Variable Taxi-Out Time Prediction Using Graph Neural Networks

Yixiang Lim†, Fengji Tan†‡, Nimrod Lilith†, Sameer Alam†
† Saab-NTU Joint Lab, Nanyang Technological University, Singapore
‡ Saab Singapore
{yixiang.lim, fengji.tan, nimrod.lilith, sameeralam}@ntu.edu.sg

***Abstract*—Airport Collaborative Decision Making (ACDM) is an important initiative that aims at more efficient and optimised use of airport resources. Variable taxi time prediction is one of the key elements in ACDM, supporting the tactical planning needed to ensure smooth traffic flow and optimal use of taxiway resources. This paper presents a mesoscopic data-driven model for the prediction of variable taxi-out times together with the associated data processing stages. To support operational implementation, the model utilises features that are readily available as part of the pre-departure tactical planning phase – namely, information on the intended taxi route and milestone time estimates. By using a Graph Neural Network (GNN) framework, each trajectory can be represented as a sub-graph of the airport taxi network, and GNN convolution operations be performed on this subgraph to extract meaningful features. Both impeded and unimpeded taxi time predictions from the GNN model are compared against standard methodologies by the Federal Aviation Authority (FAA) and EUROCONTROL, as well as against predictions made by Gradient Boosted Machines (GBM), a popular tree-based machine learning technique. Results show that both GNN and GBM models outperform standard FAA and EUROCONTROL methods (with RMSE and MAE of the former group lower by 40% to 60% relative to the latter), and the novel GNN model slightly outperforms the GBM model by around 2 seconds, or a 2% to 4% improvement in model performance.**

***Keywords*—Air Traffic Control; Airport Collaborative Decision Making; A-SMGCS data; taxi time prediction; Graph Neural Networks**

I. Introduction

An increasing digitalisation of Air Traffic Control (ATC) towers provides opportunities for stakeholders to work together to improve the safety, efficiency, resilience and sustainability of airport airside operations. A key initiative that is being pursued over the last decade is Airport Collaborative Decision Making (ACDM). The key concept behind ACDM is to enable airport users to work together to make better operational decisions through sharing a common picture. ACDM can lead to more optimised use of resources within the air transport network, reduced delays, waiting or buffer times, along with greater predictability and responsiveness to adverse conditions [1]–[3].

The ACDM elements set out in EUROCONTROL's ACDM Implementation Manual [4] describe a number of fundamental building blocks, starting with information sharing, then followed by monitoring and prediction of aircraft milestone times as well as accurate variable taxi time prediction. The work presented in this paper focuses on this latter aspect, as accurate taxi time predictions can be particularly important for effective pre-departure sequencing, translating to optimised runway usage particularly for high-capacity runway configurations such as mixed-mode operations. Tactical decision support tools providing such functionalities can support Air Traffic Controllers (ATCOs) in ensuring efficient use of airport resources, while maintaining system-wide transparency across different airport stakeholders. Additionally, accurate taxi time predictions can be beneficial for the optimisation of downstream air traffic flow and capacity in Network Operations through provision of accurate Calculated Take Off Times.

Currently, at most airports, a default taxi time is assumed for all movements, otherwise a set of taxi times are associated with particular runway configurations [4]. While this might be sufficient at small airports, taxi times can vary significantly at medium and large airports where the taxiway layout, operational mode, aircraft type, taxi route, as well as traffic and weather conditions, all can have a significant impact on actual taxiing times. In such cases, the calculation of variable taxi times can provide more accurate taxiing time estimates. While the desired accuracy of these estimates varies over different planning horizons, as described in Table I, the requirements of the tactical planning phase are the most stringent, with a desired accuracy of within 2 minutes.

TABLE I. Desired accuracy of taxi time estimates over different planning phases, based on [4].

Planning phase	Prediction horizon	Desired accuracy
Strategic	2 hrs to 3 hrs	±7 mins
Pre-tactical	30 mins to 2 hrs	±5 mins
Tactical	Up to 30 mins	±2 mins

II. Literature Review

A. Taxi Time Modelling

Taxi time models can be broadly grouped into microscopic, mesoscopic, or macroscopic models, according to the level of detail required of the model. Microscopic models simulate the taxiing dynamics of single aircraft, and can also account for complex interactions with other taxiing aircraft [5]. Mesoscopic models aggregate the taxi data of individual aircraft movements into higher-level flow metrics, usually over as attributes of individual taxiways and runways within graph-based, node-link representation of the airport [6]. In

macroscopic models, data is further aggregated into high-level flow or congestion metrics capturing taxiing characteristics over large parts of the airport (e.g., stand-runway pairs, as opposed to individual taxiway links) [7]–[9]. When deciding between these different models, one of the main considerations tends to be the trade-off between model fidelity and simplicity.

As microscopic models simulate the taxiing of individual aircraft, they can yield highly accurate and detailed outputs (e.g., taxi speed profile, precise holding times and locations, holding reason, etc.), however, a fairly complex model is required to provide such a high level of detail. Event-based simulations such as SOSS [10], SIMMOD [11] or TAAM [12] utilise detailed aircraft and airport models, along with handcrafted rules for surface movement (e.g., taxiing, holding or queueing), making it challenging to adapt such models to new airports or operating conditions. Additionally, the inherently dynamic nature of microscopic models introduces complex interdependencies between the simulated agents and their environment, which tends to incur a relatively higher computational cost, and also makes it more difficult to translate into tactical decision support tools for ATCOs.

Macroscopic models adopt a different approach by abstracting away from simulated movements of individual aircraft, instead modelling the aggregated surface flow characteristics at the airport-level. These can be further divided into empirical and analytical models. Empirical models adopt a data-driven approach for modelling taxi times based on statistical regression [13], as well as machine-learning methods [7], [9]. Analytical models include additional processes for capturing specific flow characteristics — queueing theory-based models, most prominently, have been used to model congestion dynamics at runway, crossing or ramp locations [8]. Owing to their lower complexity, macroscopic models have simpler data processing requirements, are generalizable across a wider range of operating environments, and can also be more easily integrated into decision support tools. Historically, the macroscopic approach is considered to be the most established, evident from methodologies for calculating taxi-out times, developed separately by FAA's Aviation Policy and Planning Office (APO) [14] and EUROCONTROL's Performance Review Unit (PRU) [15]. However, we note that macroscopic models reflect flow metrics at the airport level, and can suffer from a lack of detail when modelling local flow conditions, and therefore will tend to under-perform when predicting the taxi times of larger and more complex airports.

Mesoscopic models represent a middle-ground between the microscopic and macroscopic approaches. While there is also an aggregation of individual aircraft movement data, the level of granularity in the data is finer compared to that of macroscopic models – at node-link level as compared to airport level. As such, mesoscopic models present an opportunity to exploit the strengths of both microscopic and macroscopic models. Compared to microscopic models, the aggregation of aircraft data allows for a lower computational complexity and greater ease of integration into decision support tools, while the higher model fidelity relative to macroscopic models allows for potentially greater detail and accuracy in the model prediction.

B. Graph Neural Networks

In recent years, Graph Neural Networks (GNNs) have been gaining popularity as a means of performing deep learning on graph-structured data. Compared to classical deep learning approaches, which typically operate on Euclidean data, GNNs allow for machine learning operations to be performed on entities and their relations, modelled by explicit, non-Euclidean data structures – specifically graphs – which can represent entities, their properties, and the relationships between other entities. GNNs have found applications in several domains, including social networks [16], physical [17], biological [18], [19] and multi-agent systems [20], as well as in traffic forecasting [21].

A graph is defined as $G = \{\boldsymbol{V}, \boldsymbol{E}\}$ with V denoting the set of nodes in the graph and E denoting the set of edges connecting the nodes. Furthermore, G may contain a combination of node attributes $\boldsymbol{v}_i \in \boldsymbol{V}$, edge attributes $\boldsymbol{e}_{ij} \in \boldsymbol{E}$, and global attributes $\boldsymbol{u}$, and can be either directed or undirected. If G is directed, $\boldsymbol{e}_{ij}$ represents the one-way relationship between source node $\boldsymbol{v}_i$ and target node $\boldsymbol{v}_j$; otherwise, if G is undirected, a two-way relationship exists such that $\boldsymbol{e}_{ij} = \boldsymbol{e}_{ji}$. GNN tasks can be decomposed into node, edge as well as graph-level tasks – where inferences are made about individual node, edge or the entire graph's properties. Common operations performed by a GNN layer include graph convolutions and graph pooling. Graph convolutions follow a similar principle as the convolution operations used by Convolutional Neural Networks (CNNs), extracting higher-level features through the use of localised filters. In the same manner, graphs are downsampled through graph pooling operations in the same way that CNN pooling layers work to reduce the dimensionality of its input feature maps.

1) Graph Convolution: Graph convolutions can be performed on the graph's nodes and/or edges. Convolution layers are typically made up of a combination of message-passing and update steps. In the message-passing step, messages are created at nodes and/or edges based on the attributes of other nodes/edges in its neighbourhood $\mathcal{N}$ through a message-passing function M.

$$m_i^{t+1} = M(v_i^t, v_j^t, e_{ij}^t) \forall j \in \mathcal{N}_i$$

In the update step, the messages at each node/edge are collected and then used to update the node's/edge's attributes in the next layer with an update function U

$$v_i^{t+1} = U_v(v_i^t, m^{t+1})$$
$$e_{ij}^{t+1} = U_e(e_{ij}^t, m_i^{t+1}, m_j^{t+1})$$

Most of the well-known graph convolution layers, such as GCNConv [22], SAGEConv [23] and GATConv [24] were designed to operate only on node attributes, and later extended to include graph attributes (e.g., EGAT [25]). The Graph Transformer layer with edge features [26] applies the

transformer architecture to graph datasets, with edge and node message functions $m_e^{k,t+1}$ and $m_v^{k,t+1}$ for $k = 1 : H$ attention heads:

$$m_e^{k,t+1} = \left(\frac{Q^k v_i^t \cdot K^k v_j^t}{\sqrt{d_k}} \right) \cdot E^k e_{ij}^t \tag{1}$$

$$m_v^{k,t+1} = \sum_{j \in \mathcal{N}_i} softmax_j(m_{e_{ij}}^{k,t}) V^k v_j^t \tag{2}$$

where d_k is a scaling parameter based on the output dimension, and Q^k, K^k, V^k, E^k are the (learnable) query, key, node value and edge value matrices. The edge and node update steps U_e and U_v are further defined as:

$$U_e = O_e \Big\|_{k=1}^{H} (m_e^{k,t+1}) \tag{3}$$

$$U_v = O_v \Big\|_{k=1}^{H} (m_v^{k,t+1}) \tag{4}$$

where $||$ denotes the concatenation operator performed over all attention heads, and O_e, O_v are learnable matrices.

2) Graph Pooling: Graph pooling layers are used to coarsen a graph by combining clusters of nodes or edges. For graph-level inference tasks, global pooling operators are used to aggregate a graph's node and edge attributes into a single feature vector $\boldsymbol{x}_G$. Examples of common global pooling operators are mean, max or sum pooling, as well as sort pooling [27], Set2Set [28] and global attention [29]. The global attention pooling operator is given by:

$$\boldsymbol{x}_G = \sum_{i=1}^{N} softmax(\phi_{gate}(v_i)) \odot \phi_\Theta(v_i) \tag{5}$$

where ϕ_{gate} is a neural network that computes attention scores, and ϕ_Θ is a neural network that maps the node attributes to the output dimension.

III. Data Pipeline

Figure 1 presents the various processing stages in the data pipeline. The pipeline comprises three main stages, with higher level data being extracted at each stage of processing.

In the first stage, depicted in gray, aircraft trajectories are extracted from raw surveillance data, provided by Saab Sensis' Aerobahn surface management system, along with other auxiliary data such as weather information. In the second stage, trajectories are fused with the airport graph data by a map-matching algorithm, then further enriched with movement status and runway queue information. The enriched trajectories are then used to obtain flow features, which are combined with trajectory and auxiliary data in the third stage to generate the dataset used to train the model.

The remainder of this section focuses on the second stage of the data pipeline, highlighting key processes used to derive key model features from trajectory data.

A. Graph Association

The four-dimensional track data obtained in the first stage of the data pipeline is fused with a directed graph of the airport surface network through a map-matching process. The map-matching algorithm calculates the probability P_e of a track datapoint being associated to edge e of the airport graph based on a combined angular/distance-based approach:

$$P_e = P_d * P_a * P_\theta.$$

The association probability P_e comprises three components: P_d, denoting the association probability as a function of shortest (perpendicular) distance between the datapoint and edge e; P_a, denoting the association probability as a function of the non-dimensional distance of the datapoint along edge e (i.e., between 0 and 1 if lying along e); and P_θ, denoting the association probability as a function of the angular alignment between the datapoint and edge e.

B. Runway Queue Identification

Departing flights taxiing out to their allocated runway are assigned to a departure queue prior to takeoff. During periods of high demand, it may be desirable to maintain greater pressure on departing runways – via longer queue lengths – in order to maximize runway efficiency at the cost of additional holding times for the queuing aircraft. To estimate the holding times that aircraft spend in a departure queue, buffers zones are defined around the entrances to each runway, illustrated in Fig 2. Any holding occurring within these queue zones is assumed to be due to runway queuing, as opposed to flow-related congestion.

C. Holding Time Disaggregation

To gain a deeper understanding of the distribution and cause of taxi holding, holding times are further broken down into the separate components described in Table II.

TABLE II. Holding times breakdown.

Hold type	Description
H_{rwy}	Holding in the departure queue zone (departures only).
$H_{crossing}$	Holding when the aircraft is crossing a runway.
H_{flow}	Holding along taxiways, when the downstream flow is greater than a set threhsold.
$H_{twy,gate}$	Holding along taxiways, before the assigned gate becomes available (arrivals only).
$H_{ramp,gate}$	Holding on the apron, before the assigned gate becomes available (arrivals only).
$H_{twy,res}$	Any residual holding along taxiways that is not accounted for by $H_{crossing}$, H_{flow} or $H_{twy,gate}$.
$H_{ramp,res}$	Any residual holding on the apron that is not accounted for by $H_{ramp,gate}$.
H_{twy}	The sum of $H_{crossing}$, H_{flow}, $H_{twy,gate}$ and $H_{twy,res}$.
H_{ramp}	The sum of $H_{ramp,gate}$ and $H_{ramp,res}$.
H_{total}	The sum of H_{rwy}, H_{twy} and H_{ramp}.

While an in-depth discussion of Table II is outside the scope of this paper, the breakdown presented above is primarily used to identify outlier trajectories from the dataset, based on inputs from subject matter experts. For example, the residual

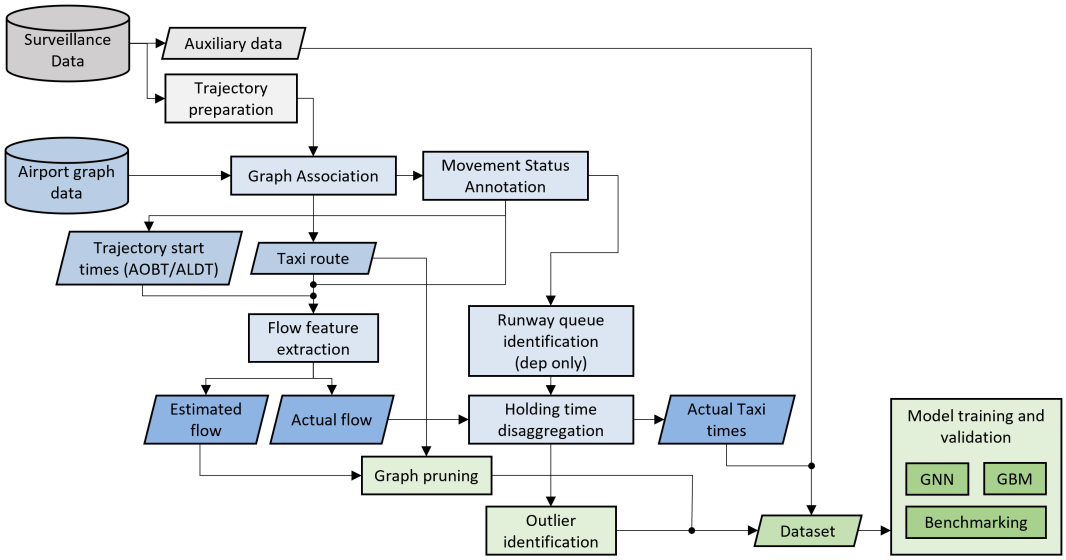

Figure 1. Data pipeline, with key processing stages and data structures color-coded by processing stage; grey denotes trajectory extraction, blue denotes graph fusion (light blue data boxes represent base data and dark blue data boxes represent derived data) and green denotes model training.

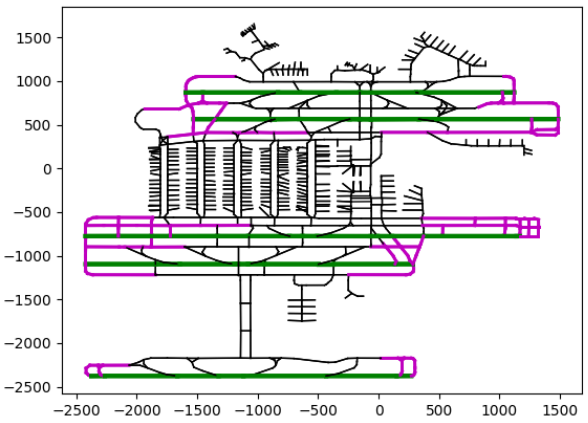

Figure 2. Designated queue zones of Atlanta airport, highlighted in magenta. Runways are colored green. Queue zones are specific to the particular runway.

holding times $H_{twy,res}$ and $H_{ramp,res}$ are good metrics for identifying trajectories that hold for non flow-related reasons (e.g., technical/passenger issues).

D. Flow Features

Traffic conditions are one of the key variables to be considered in the estimation of taxi times, since congested traffic is known to adversely impact surface movement (e.g., affecting taxiing speeds and leading to queuing delays). Although congestion metrics – such as the number of taxiing in/out aircraft and arrival/departure count – have been previously used as features in macroscopic models, in our data exploration we found that local flow conditions were more important than global flow conditions, particularly for large airports where congestion in a localised area might not translate to increased taxi times for aircraft in other parts of the airport.

In our model, surface flow represents the spatio-temporal movement of aircraft within the airport at a mesoscopic level, aggregated from individual aircraft over different taxiway links. From the fused trajectory data obtained from Section III-A, the taxi intent – assumed here to be the taxi route directly derived from the fusion process (i.e., not accounting for tactical changes) – is extracted as a series of edges. Trajectory start time estimates were obtained from movement data, as it was found that the provided milestone timings were too inaccurate. Moreover, while milestone times were given in minute-level precision, the movement data was resampled to second-level precision, therefore allowing for more accurate estimates of the trajectory start time. For arrivals, the landing time was taken to be the trajectory start time, while the start-up time was used for departures. A 'first-order' estimate (i.e., not accounting for other interactions in the environment, for example with other aircraft) of the route time was obtained by assuming fixed taxiing speeds over different taxiway types (e.g., apron, taxiway and runway). With the start time as a reference point, the route timings were processed to obtain occupancy intervals – the time intervals where each airport taxiway would be occupied by aircraft. Furthermore, if some statistical distribution is assumed (e.g., Gaussian), the probability of an aircraft occupying the edge at any particular time interval can be derived from the occupancy interval projections. In total, the following edge flow features can be obtained:

- Aircraft count: the cumulative number of aircraft passing through a particular taxiway within a specified time window;
- Edge occupancy probability: the probability of a taxiway being occupied within a specified time window, taken

to be the highest occupancy probability of all aircraft (or equivalently, the occupancy probability of the nearest aircraft); and
- Flow potential: the sum of the occupancy probabilities of all aircraft within a specified time window. If only one aircraft passes through the taxiway during the time window, the flow potential is equal to the edge occupancy probability.

E. Auxiliary Data

In addition to the trajectory data described in the sections above, a number of other auxiliary features are also processed within the data pipeline.

1) Weather: Inclement weather may have an adverse impact on airport throughput – primarily reducing runway capacity, but also impacting the taxiway efficiency, and might impose additional operational constraints, such as the need for departing aircraft to perform de-icing during freezing conditions. Following the methodology developed by [30], TAF and METAR records were retrieved from the Aerobahn system and processed to derive scores for each of the five weather classes (visibility, wind, precipitation, freezing conditions, dangerous phenomena). Minor modifications were made to the scoring method to obtain a finer division of scores for some of the weather classes. However, in our study, it was found that weather did not play a significant role in taxi time prediction, and was ultimately omitted from the input feature set.

2) Air Carrier: The air carrier information can be a relevant feature when predicting aircraft ground movement, and is used in the FAA APO methodology. Different carriers might have different taxiing policies, with some possibly favoring expediency while others preferring to taxi at lower, fuel-efficient speeds. Additionally, at particular airports, air carriers might have been assigned to a set of default apron stands. If their assigned stands are fully occupied, arriving aircraft belonging to the carrier might be forced to wait for occupying aircraft to vacate before continuing their taxi into their assigned stand.

While the air carrier information could be fed directly as a model input via one-hot encoding, this might not be feasible at large airports, where the large number of operating airlines translates to a corresponding large input dimensionality – greater than 90 in our case study. For a more compact feature representation, embeddings were generated from a set of carrier characteristics (e.g., daily airport utilisation, fleet mix, flight mix, apron mix, etc.), derived from airport traffic. The specific details of the embedding methodology are not discussed in this paper.

IV. GNN Model

We let the airport be defined as a directed graph $G = \{V, E\}$ with the set of nodes V and set of edges E. Then, each trajectory can be represented by a subset of G, $G' = \{V', E', \boldsymbol{u}\}$ with node attributes $\boldsymbol{v}_i \in V'$, edge attributes $\boldsymbol{e}_k \in E'$, and global attributes $\boldsymbol{u}$. To support implementation in an operational context, we limit the model inputs to features that can be extracted from tactical planning data. In particular, we include features that represent the structural layout of the airport, route features derived from the trajectory intent, as well as flow projections obtained from the first-order trajectory time estimates. These features are listed in Table III, with edge attributes decomposed into static ($e_{k,s}$) and dynamic $e_{k,d}$ attributes $\boldsymbol{e}_k = [\boldsymbol{e}_{k,s}, \boldsymbol{e}_{k,d}]$, separately describing the airport topology and taxiway flow conditions. Node attributes describe the taxi intent and its relation to local taxiways. Global attributes capture information that otherwise cannot be included in the graph representation, such as carrier, runway, queue position, the first-order time/distance estimates, etc.

TABLE III. Model attributes.

Input type	Features
Node attrs	Est. route dist
	Node anchor
	Distance to anchor
	Anchor weight
Edge attrs (static)	Edge type
	Edge length
	Inverted edge length
Edge attrs (dynamic)	Est. aircraft count
	Est occupancy probability
	Est flow potential
Global attrs	Est rwy takeoffs and landings
	Est rwy takeoffs and landings (normalised)
	Queue position
	Queue position (normalised)
	Est route distance
	Est unimpeded time
	Rwy ID
	Aircraft wakecat
	Carrier embeddings

A. Dataset

The dataset comprises taxi movements at Atlanta airport over two months (December 2019 and January 2020), with approximately 136,000 movements (arrivals and departures combined) over 62 days. As there are certain factors in taxi time prediction that are specific to arrival and departure movements, separate taxi-out and taxi-in models can allow for more accurate results. Currently, we just focus on the taxi-out time prediction for departing flights, comprising around 67,000 movements prior to outlier rejection.

B. Outlier identification

Outliers are designated as movements with excessively long delay times, as well as trajectories with excessively short movement times. The former group of movements accounts for trajectories that hold for non flow-related reasons, while the latter group of movements could account for trajectories that were incorrectly processed. Table IV lists the set of identification criteria used. Approximately 62,000 departure trajectories remain after outlier rejection, which is then divided into training, validation and testing sets via a 70-20-10 split. The test set comprises the last 10% of movements in the dataset (in chronological order), while the remainder of the data is evenly divided between the train and validation sets.

TABLE IV. OUTLIER IDENTIFICATION CRITERIA.

Criteria	Description
$H_{rwy} > 600s$	Excessively long holding time at the runway.
$H_{twy} > 600s$	Excessively long holding time at the taxiway.
$H_{ramp} > 600s$	Excessively long holding time on the apron.
$H_{twy,res} > 60s$	Residual (i.e., non-flow related) taxiway holding time.
$H_{total} > 1200s$	Excessively long total holding time.
$t_{unimp,rwy} < 12.5s$	Excessively short unimpeded taxi time at the runway.
$t_{unimp,total} < 45s$	Excessively short total unimpeded taxi time.
$t_{imp,total} < 60s$	Excessively short total impeded taxi time.
$d_{est} < 300m$	Excessively short 'first-order' approximation of taxi distance.

C. Graph pruning

Based on the methodology presented in Section III-A, a set of anchor nodes, $V_\alpha \in V$, can be extracted from the taxi route. V' can be derived from V_α by filtering out the nodes exceeding some distance threshold d_{thresh} to V_α:

$$V' = v \in V \; \forall \; mindist(v, V_\alpha) < d_{thresh}$$

where the $mindist(v, V)$ operator simply returns the minimum path distance between v and the set of nodes in V. Subsequently, E' is obtained by only retaining edges with source and target nodes in V'. A second stage of pruning is performed by only retaining edges with non-zero flow features, then removing nodes that are not connected to any edges. The result of the pruning for a particular trajectory is shown in Figure 3.

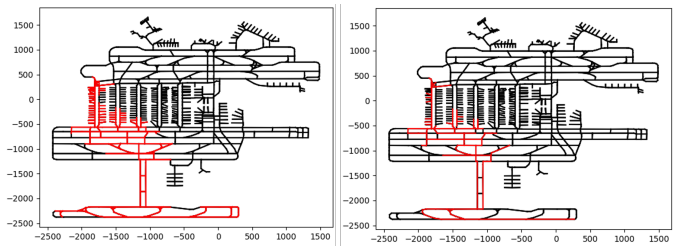

Figure 3. Results of the first (left) and second (right) stages of the graph pruning with $d_{thresh} = 500m$; remaining edges highlighted in red.

As shown in Table V, graph pruning greatly reduces the size of the resulting graph, by more than an order of magnitude in most trajectories when $d_{thresh} \leq 500m$, thereby improving CPU memory requirements and computational efficiency.

TABLE V. GRAPH PRUNING, SHOWING THE NUMBER OF NODES/EDGES IN THE ORIGINAL GRAPH, THEN AT DIFFERENT VALUES OF d_{thresh}

	Original		500m		250m		200m		50m	
	N_V	N_E	N_V	N_E	N_V	N_E	N_V	N_E	N_V	N_E
Mean	1882	2713	208	266	112	140	85	101	81	99
Min			25	24	25	24	25	24	25	24
Max			468	712	268	397	209	291	190	269
95%			275	367	163	212	132	165	119	152

D. Model Architecture

The GNN model architecture is presented in Figure 4. At the start of the model, different input types are passed through a series of dense layers to obtain their respective encodings. Graph features, comprising the node, as well as static and dynamic edge encodings, are then processed in a spatio-temporal block, comprising a spatial block nested inside a temporal block. Graph convolutions are performed within the spatial block, and a global pooling is performed on the transformed graph to obtain global graph features at each convolution step. The outputs of all convolution steps are passed through a dense layer to obtain the spatial encoding for time *t*. The spatial encoding timeseries are then passed through another dense layer to obtain the spatio-temporal encodings. Finally, the spatio-temporal encodings are concatenated with the global encodings, then passed through another series of dense layers to obtain the final output.

We use the Graph Transformer with edge features (Eqs 1 to 4) as the graph convolution operator, and the soft attention mechanism (Eq 5) as the global pooling operator.

V. RESULTS AND DISCUSSION

We compare the GNN results with the following models:

- Naive: We take a naive estimate by assuming the mean value of the training/validation dataset;
- APO: Predictions are made using the FAA APO methodology [14];
- PRU: Predictions are made using EUROCONTROL's PRU methodology [15]; and
- GBM: The GNN's global features used to train a tree-based model based on Gradient-Boosted Machines.

Both impeded and unimpeded taxi-out time predictions on the test dataset are used for comparison, with the error distribution presented in Figure 5 and comparison metrics presented in Table VI, comparing the root mean squared error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE) of the different models. As the PRU and APO methods were originally developed to estimate the unimpeded taxi time, slight modifications have been made to account for the impeded taxi time. A point to note is that these two methods were originally intended to be used as high-level metrics for airport performance monitoring and review, not for tactical planning. However, it is useful to include them here as a baseline when comparing the prediction performance of our model.

TABLE VI. TAXI-OUT TIME PREDICTION RESULTS.

Model	Impeded taxi-out time			Unimpeded taxi-out time		
	RMSE	MAE	MAPE	RMSE	MAE	MAPE
Naive	216.4s	171.9s	39.3%	126.2s	102.0s	32.6 %
PRU	199.2s	159.8s	33.9%	115.1s	86.5s	22.1%
APO	123.1s	101.3s	26.8%	114.3s	91.4s	34.9%
GBM	78.4s	57.3s	11.6%	51.1s	39.5s	10.9%
GNN	76.0s	55.2s	11.0%	50.1s	38.4s	10.4%

GBM and GNN-based models outperform the standard models by quite a significant margin – about 60% reduction

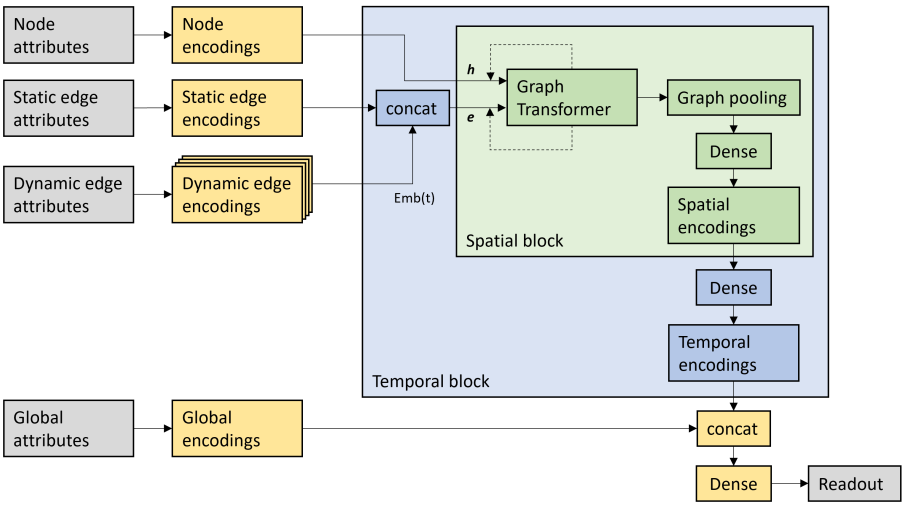

Figure 4. GNN architecture.

in RMSE and MAE relative to the PRU method, and 40% relative to the APO method. Comparing between the PRU and APO methods, the APO prediction gives a lower error in the impeded taxi-out time than the PRU prediction. This is most probably because the APO model includes a regression on the number of taxi-in and taxi-out movements for each carrier, whereas the PRU method assumes a median the unimpeded taxi time for each runway-stand pair (we further obtain the impeded taxi time by assuming a mean additional taxi time of each runway-stand pair, and adding this value to the unimpeded taxi time).

Both GNN and GBM models have a test MAE of under 60 seconds, suggesting that both models are, on average, able to satisfy the desired taxi-time accuracy of ± 2 mins for tactical movement planning. As shown in Fig 5, the impeded taxi time prediction errors for the GNN and GBM models are contained within a relatively narrow band, with the first and third quartiles of the GNN error falling between -33.2s to 48.2s, and that of the GNN falling between -52.3s to 31.5s. Impeded time prediction errors falling outside the ± 120s range correspond to the $[5.0, 96.0]$ percentiles for the GNN model and the $[4.9, 94.9]$ percentiles for the GBM model (both corresponding to around 90% of predictions in the test set). In spite of its relatively simplicity, The GBM model performs surprisingly well, with only a slightly lower accuracy when compared with the GNN model (approximately 2s difference for impeded and 1s difference for unimpeded taxi-out time prediction). This suggests that the global feature set is sufficient to obtain good estimates for both impeded and unimpeded taxi-out times.

With comparable performance of both GNN and GBM models, the relative simplicity of the GBM model gives it an advantage in terms of implementation, while also possessing a relatively high level of model interpretability due to the tree-based algorithms used. On the other hand, while relatively more complex, the GNN framework offers greater flexibility, with the potential for improved prediction performance through either the inclusion of other graph-based features, or with modifications to the model architecture. Furthermore, there is scope for extending the GNN model, beyond the graph-level task presented here, to other types of inference tasks – for example, taxiway speed/flow prediction (edge-level tasks) or hotspot prediction (node-level tasks).

VI. Conclusion

This paper presents a methodology for the prediction of variable taxi-out time using GNNs. The data pipeline describing the key data processing and feature engineering stages is first presented, followed by the model architecture and validation results. Each aircraft trajectory is represented by a subset of the airport graph, incorporating features that describe the airport taxiway topology as well as projected flow estimates. For ease of operational implementation, the model only includes features that are readily obtainable from tactical planning information. Model testing results show that the GNN model outperforms standard methodologies used by the FAA APO and EUROCONTROL's PRU. Additionally, the GNN-based model is also seen to outperform conventional machine-learning methods such as GBMs, albeit by a small margin. Results from the model testing also suggest that impeded taxi-out predictions of both GNN and GBM models are accurate enough to be used in tactical movement planning, with a test MAE of under 1 minute, and 90% of the predicted impeded times being accurate to within ± 2 mins. These findings support the use of data-driven approaches for the provision or reliable variable taxi time predictions, particularly for large airports with complex taxiway layouts. The outputs of the proposed model will be important in supporting more advanced ACDM functionalities for tactical/pre-tactical surface movement planning and optimisation, such as pre-departure sequencing, mixed-mode runway operations, or Arrival/Departure management (AMAN/DMAN) integration.

Future work will verify the model's generalisability to other airports by comparing its performance on different datasets.

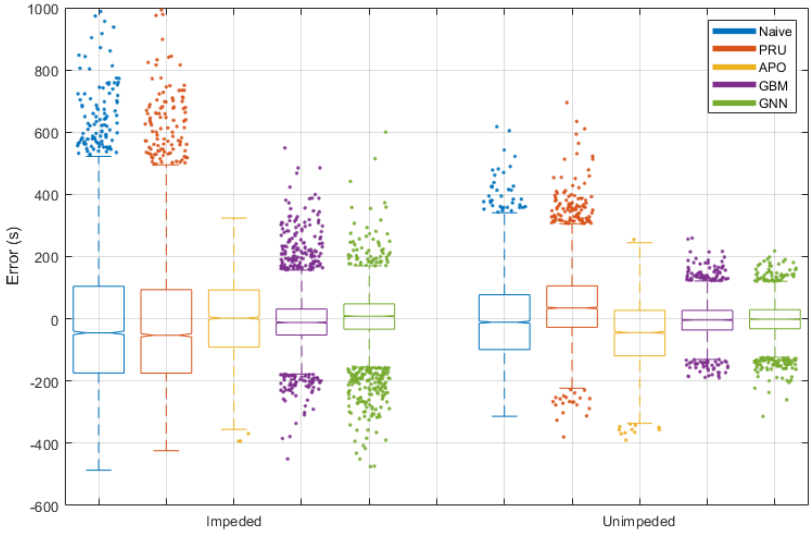

Figure 5. Error distributions of the different models. The thick bars denote the data between $q1$ and $q3$, while the whiskers denote data within $q3 \pm 1.5 \times (q3 - q1)$, where q1 and q3 are the first and third quartiles respectively.

Additionally, ongoing research on feature engineering and architecture design can contribute towards further improving the GNN model performance. Future research will also look at incorporating the model in a surface planning and management framework for ACDM, while investigating mechanisms to support greater explainability of machine learning techniques in ATCO decision support tools. Finally, it might be useful to explore the applicability of the GNN approach presented in this paper to other aspects of air traffic management beyond surface operations, since many processes in the air traffic system can also be modelled by graphs.

Acknowledgment

This work was conducted under the Saab-NTU Joint Lab with support from Saab AB (publ). The authors are grateful to Mr Daryl Tan for his expertise and inputs to this project.

References

[1] EUROCONTROL, "A-CDM impact assessment," EUROCONTROL, Report, 2016.
[2] O. Delain *et al.*, "PJ.04 - Total Airport Management (final project report," SESAR, Report, 2019.
[3] M. Schultz *et al.*, "A-CDM lite: Situation awareness and decision-making for small airports based on ads-b data," in *9th SIDs*, 2019.
[4] EUROCONTROL, "Airport CDM implementation manual," EUROCONTROL, Report, 2017.
[5] T.-N. Tran *et al.*, "Taxi-speed prediction by spatio-temporal graph-based trajectory representation and its applications," in *ICRAT*, 2020.
[6] M. Jeong *et al.*, "Unimpeded taxi-time prediction based on the node–link model," *J Aerosp Inf Sys*, pp. 1–12, 2020.
[7] H. Lee *et al.*, "Prediction of pushback times and ramp taxi times for departures at Charlotte Airport," in *AIAA AVIATION 2019 Forum*, 2019.
[8] S. Badrinath *et al.*, "Integrated surface–airspace model of airport departures," *J Guid Cont Dyn*, vol. 42, no. 5, pp. 1049–1063, 2019.
[9] J. Yin *et al.*, "Machine learning techniques for taxi-out time prediction with a macroscopic network topology," in *37th DASC*, 2018, pp. 1–8.
[10] R. D. Windhorst *et al.*, "Validation of simulations of airport surface traffic with the surface operations simulator and scheduler," in *13th ATIO*, 2013.
[11] "SIMMOD manual," FAA, Report.
[12] J. Boesel *et al.*, "TAAM best practices guidelines," MITRE, Report, 2001.
[13] Y. Zhang and Q. Wang, "Methods for determining unimpeded aircraft taxiing time and evaluating airport taxiing performance," *Chinese J Aeron*, vol. 30, no. 2, pp. 523–537, 2017.
[14] FAA, "U.S./Europe comparison of ATM-related operational performance 2010," FAA, Report, 2012.
[15] L. Capelleras, "Additional taxi-out time performance indicator document," EUROCONTROL, Report, 2015.
[16] W. Fan *et al.*, "Graph neural networks for social recommendation," in *TheWebConf*, 2019, pp. 417–426.
[17] A. Sanchez-Gonzalez *et al.*, "Learning to simulate complex physics with graph networks," in *ICML*, 2020, pp. 8459–8468.
[18] J. Jumper *et al.*, "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.
[19] J. Gilmer *et al.*, "Neural message passing for quantum chemistry," in *ICML*, 2017, pp. 1263–1272.
[20] R. Dalmau and E. Allard, "Air Traffic Control using message passing neural networks and multi-agent reinforcement learning," in *10th SIDs*, 2020.
[21] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," *arXiv preprint arXiv:2101.11174*, 2021.
[22] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th ICLR*, 2017.
[23] W. L. Hamilton *et al.*, "Inductive representation learning on large graphs," in *31st NeurIPS*, 2017, pp. 1025–1035.
[24] P. Veličković *et al.*, "Graph attention networks," in *6th ICLR*, 2018.
[25] J. Chen and H. Chen, "EGAT: Edge-featured graph attention network," in *ICANN*, 2021.
[26] V. P. Dwivedi and X. Bresson, "A generalization of transformer networks to graphs," in *DLG-AAAI*, 2021.
[27] M. Zhang *et al.*, "An end-to-end deep learning architecture for graph classification," in *32nd AAAI*, 2018.
[28] O. Vinyals *et al.*, "Order matters: Sequence to sequence for sets," in *4th ICLR*, 2016.
[29] Y. Li *et al.*, "Gated graph sequence neural networks," in *4th ICLR*, 2016.
[30] "Describing the weather impact algorithm developed by the ATMAP project," EUROCONTROL, Report, 2011.