

Detection of Aircraft Pushback Segments Using Deep Learning

Kevin Hänggi^{*} and Manuel Waltert

Centre for Aviation, Zurich University of Applied Sciences, Winterthur, Switzerland

^{*}Corresponding author: kevin.haenggi@zhaw.ch

Abstract

Aircraft pushback information is relevant for analysing airport surface movements, taxiing behaviour, congestion, fuel consumption, and emissions. However, pushback events are generally not explicitly identified in publicly available surveillance data and must instead be inferred from aircraft trajectories. Existing algorithmic approaches rely on fixed geometric and movement-based criteria, which may fail due to variable pushback manoeuvres, incomplete trajectories, inaccurate ground positions, or delayed transponder activation. This study investigates whether aircraft pushback segments can be detected more reliably using deep learning methods and openly available Automatic Dependent Surveillance–Broadcast (ADS-B) data from the OpenSky Network. We develop a non-causal Pointwise Residual Dilated Temporal Convolutional Network (PRD-TCN) pipeline using trajectory data from Stockholm Arlanda, Zurich, Tokyo Haneda, and Perth airports, with model configurations selected through leave-one-airport-out experiments. The final model estimates a pushback probability for each trajectory point, from which pushback segments are extracted through post-processing. Performance is evaluated on an independent manually labelled dataset from Brussels, Boston Logan, and Aeroparque Jorge Newbery airports, none of which were used during pipeline development, and compared with the parking-area-based and parking-position-based pushback-detection methods implemented in the open-source Python library *traffic*. The proposed pipeline achieves a macro F_1 -score of 98.4%, compared with 66.8% and 44.0% for the two rule-based methods, respectively. A macro specificity of 97.7% and a macro balanced accuracy of 97.9% further demonstrate that the pipeline not only reliably detects pushbacks but also correctly identifies flights without pushback. The detected segments achieve a macro mean temporal intersection-over-union of 0.864, with mean absolute start and end timing errors of 5.1 s and 10.2 s. These results demonstrate the potential of deep learning to automate the reliable extraction and temporal localisation of pushback information from open ADS-B trajectories and reduce the manual and airport-specific preprocessing required for airport surface-operation research.

Keywords: Aircraft Pushback; Automatic Dependent Surveillance–Broadcast; Deep Learning; Temporal Convolutional Networks; Trajectory Segmentation

Abbreviations: ADS-B: Automatic Dependent Surveillance–Broadcast PRD-TCN: Pointwise Residual Dilated Temporal Convolutional Network

1. Introduction

Future growth in aircraft movements at aerodromes is expected to place increasing pressure on airport infrastructure and operations in the coming years [1]. Since expanding airport capacity is often constrained by spatial limitations, environmental considerations, and high investment costs, improving the efficiency of existing infrastructure and resources is often the only viable way for airports to accommodate growing demand. A detailed understanding of airport operations is therefore required to identify inefficiencies, evaluate operational procedures, and assess potential improvements. Aircraft surface trajectories derived from surveillance data, particularly openly available Automatic Dependent Surveillance–Broadcast (ADS-B) data, provide an important basis for such analyses, as they allow aircraft movements on the airport movement area to be examined across large numbers of flights. However, ADS-B data primarily provide time-stamped position, altitude, speed, direction,

and identification information [2]. They do not explicitly encode operational phases such as parking, pushback, and taxiing, nor do they indicate where these phases begin and end [3]. This information must therefore either be obtained from complementary operational data sources, which are often proprietary, or inferred from the recorded trajectories.

One particularly relevant operational phase of airport surface operations is pushback. During pushback, an aircraft parked at a nose-in stand is moved away from its parking position by an external ground vehicle and positioned on the apron or taxiway, from where it can subsequently commence taxiing under its own power [4]. Pushback marks the transition between stand operations and taxi-out. Consequently, reliable identification of pushback events, together with their start time, end time, and duration, provides an important basis for analysing airport surface operations, including stand and apron utilisation, taxi-out performance, surface traffic congestion, aircraft fuel consumption, and associated emissions.

Existing approaches for extracting pushback segments from surface trajectories include two rule-based methods provided by the open-source Python library *traffic* [5]. The `parking_position` method first determines the aircraft's parking stand by intersecting the trajectory with slightly buffered stand geometries documented in OpenStreetMap. Starting from the first entry into this stand, it detects the onset of movement as the beginning of the first movement segment lasting more than 30 s in which the filtered, position-derived ground speed exceeds 2 kt. The pushback is then delimited from this onset to the first abrupt change of more than 90° in the filtered, position-derived ground track between consecutive trajectory points, searched between 30 s and 5 min after the onset of movement. The `parking_area` method instead retains only the points at which the filtered, position-derived ground speed of the aircraft exceeds 1 kt and groups them into consecutive movement segments separated by temporal gaps of more than one minute. Such gaps may result either from an extended interval below the speed threshold or from missing position reports. The method classifies the first segment as pushback if it intersects a documented apron area and if the absolute numerical difference between the median position-derived ground track over the last 20 s of this segment and that over a 20 s window starting 5 s after the beginning of the following segment is at least 90°. Although both methods of the *traffic* library are straightforward to apply, their reliance on fixed spatial, temporal, and directional criteria limits their ability to represent the variability of pushback manoeuvres observed in actual operations. Depending on the airport layout, parking position, aircraft type, and local operating procedures, pushbacks may follow simple or complex paths, including L-shaped, S-shaped, U-shaped, and push-pull manoeuvres [4]. In particular, a large directional change within a push-pull manoeuvre may cause `parking_position` to terminate the detected segment before the manoeuvre is completed, while `parking_area` may return only the initial part of the manoeuvre or reject it entirely when the required directional difference between consecutive movement segments is not present. Stops that do not produce a temporal gap of more than one minute between retained observations may cause `parking_area` to merge pushback and subsequent taxiing into a single movement segment. Conversely, positional noise can create apparent directional changes during regular taxiing or apparent movement while the aircraft is stationary, both of which may satisfy the rule-based criteria and contribute to false detections. These methods further rely on documented stand or apron geometries, which may be incomplete or inaccurate, while geometry-based assignment using narrow spatial tolerances can be affected by positional inaccuracies. Late initiation of ADS-B trajectory recordings, for example when pilots switch on the transponder only after commencing taxiing, inaccurate ground positions, and missing trajectory points can further lead to incomplete or misleading observations. Together, these limitations can cause missed pushback operations, false detections, or incorrectly delimited segments, as illustrated in Figure 1.

Beyond the two methods available in the *traffic* library, several studies have developed rule-based methods tailored to their respective applications. Waltert, Figuet, and Felux [4] proposed an alterna-

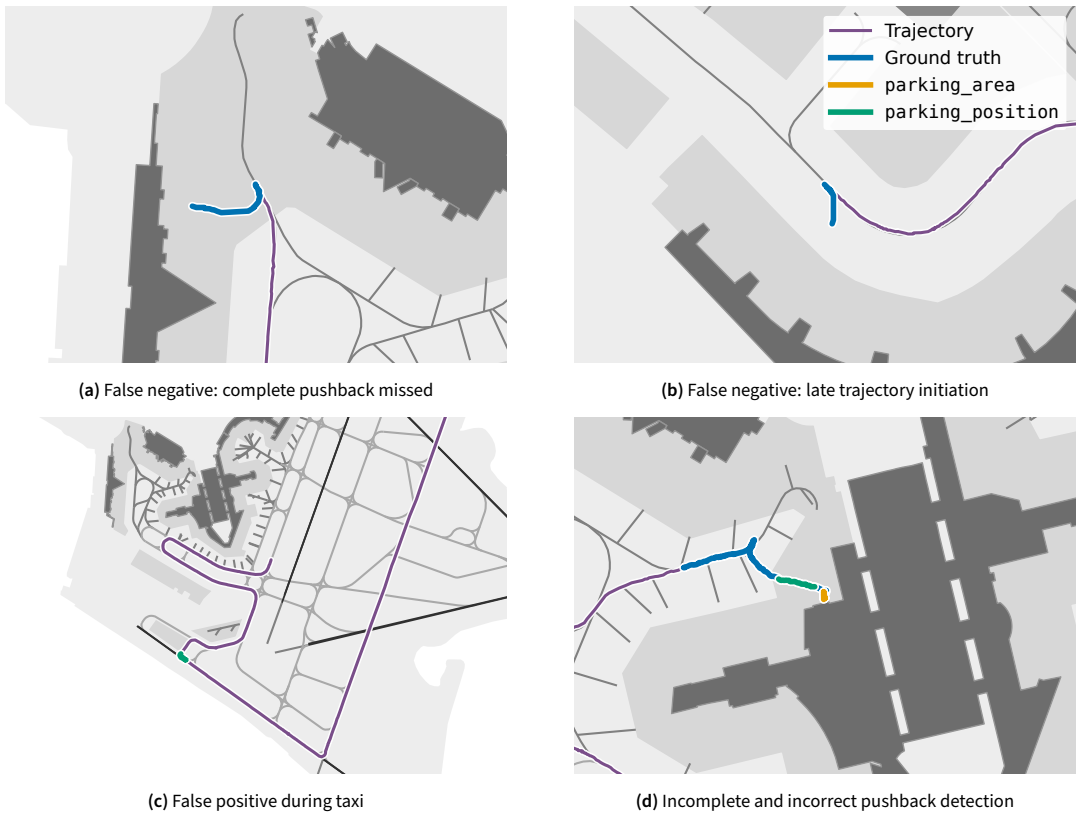


Figure 1. Failure cases of the rule-based pushback methods at Boston Logan International Airport, drawn from the manually labelled independent test dataset (Section 2.1.2). (a) Complete pushback missed. (b) False negative because the observed trajectory begins after pushback initiation. (c) False positive during regular taxiing; a wider spatial extent is shown for context. (d) Incomplete and incorrectly delimited pushback segments. Purple denotes the observed trajectory, blue the ground truth, and orange and green segments detected by the parking_area and parking_position methods, respectively.

tive to the existing `pushback()` implementation in *traffic* after observing that its reliance on sudden changes in the position-derived ground track may not reliably capture more complex manoeuvres. The authors defined polygons covering nose-in, push-out stand areas at Zurich Airport and classified moving trajectory segments crossing their boundaries as pushback. Stloukal, Had, and Hospodka [6] assigned departing aircraft to individual stands and retained only trajectories for which the onset of pushback could be observed at the stand. The authors then identified the beginning and end of pushback from transitions between stationary and moving states of the aircraft, while applying minimum-duration requirements to distinguish these transitions from brief stops during pushback and subsequent taxiing. Hoole, Booker, and Cooper [7] used changes in the bearing between consecutive position reports of the aircraft and marked a change exceeding 60° as the pushback apex, after which preceding turning manoeuvres were relabelled as occurring during pushback. These studies demonstrate that pushback-related information can be derived from surveillance trajectories using different rule-based methods. However, each method was developed for the respective airport, dataset, and research objective and relies on geometries, threshold values, and assumptions selected accordingly. Consequently, generalizing these approaches to other airports may require adaptation of the underlying rules and parameters as well as additional validation.

A possible alternative to rule-based pushback detection is deep learning. Although pushback it-

self can be described in operational terms, translating this description into fixed spatial, temporal, and directional criteria that remain applicable across different airports is more difficult. Instead of relying on manually defined rules, a learning-based model can infer the relevant characteristics of pushback directly from labelled examples. This avoids explicitly encoding particular parking geometries, thresholds, or heuristic decision rules in the model and may therefore better accommodate the variability observed across airports and operations. A further advantage is that label generation and model application do not need to rely on the same information, allowing training labels to be derived from additional data sources that are not required during later application.

To the best of our knowledge, deep learning has not yet been used to detect and temporally delimit aircraft pushback segments from surveillance trajectories. It has, however, already been applied to related event-detection tasks in aircraft trajectories. Olive et al. [8] developed a deep-learning model for detecting holding-pattern segments, including incomplete and atypically flown patterns. The authors employed cross-airport validation by training models on subsets of airports and testing them on airports excluded from training. Jarry et al. [9] used deep-learning models to detect single-engine taxiing during the taxi-in phase and estimate its starting point from movement patterns in ground trajectories. Proprietary Quick Access Recorder data were used to derive the reference labels, whereas the model inputs were restricted to trajectory features that are also available from ADS-B or ground-radar data. These studies demonstrate the feasibility of using deep-learning models to infer operational events from aircraft trajectories even when those events are not explicitly represented in the available trajectory variables.

Against this background, our study addresses two research questions. First, using openly available ADS-B trajectory data from the OpenSky Network, we examine whether a deep-learning-based detection pipeline can determine whether a pushback occurs and identify the corresponding segment at airports not used during development of the proposed pipeline. Second, we evaluate how accurately the proposed pipeline can identify pushback occurrences and their temporal segment boundaries compared with the rule-based pushback detection methods implemented in the *traffic* library. To address these questions, we develop and evaluate a complete detection pipeline comprising trajectory preprocessing, a deep-learning model as its central component, and post-processing of the model outputs. The evaluation is performed on an independently and manually labelled reference dataset comprising trajectories from three airports not used during pipeline development. More broadly, this work aims to provide a transferable and automated means of extracting pushback information from open trajectory data without detection rules tailored to a particular airport. Such an approach can reduce the manual and airport-specific preprocessing required for subsequent research and enable larger-scale analyses of airport surface operations.

The remainder of this paper is structured as follows. Section 2 describes the datasets, development of the proposed pushback detection pipeline, and evaluation methodology. Section 3 presents the evaluation results and comparison with the rule-based methods implemented in the *traffic* library. Section 4 discusses the findings and limitations, while Section 5 summarises the main conclusions and provides an outlook for future research.

2. Methods

The methodology comprises three main stages. Section 2.1 describes the construction and reference labelling of separate development and independent test datasets. Section 2.2 introduces the proposed pushback detection pipeline, while Section 2.3 describes model development and configuration selection. Finally, Section 2.4 describes the evaluation of the final pipeline on the independent test dataset against existing rule-based methods.

2.1 Data Collection and Reference Labelling

Separate development and independent test datasets were constructed from ADS-B trajectory data obtained through the OpenSky Network [10]. To assess the generalisability of the final pushback detection pipeline, the test dataset comprised airports and trajectories not used during model development. Development reference segments were generated using a rule-based procedure exploiting additional directional information available in the OpenSky historical database, whereas test reference segments were annotated manually. The rule-based reference-labelling procedure was not applied to the test dataset.

For both datasets, a reference segment defines the sequence of trajectory observations belonging to a pushback manoeuvre. A pushback extends from the start to the end of the associated aircraft movement and includes intermediate stationary observations within multi-stage or push-pull manoeuvres. Stationary observations preceding or following the manoeuvre, including those associated with tug coupling or decoupling, are excluded. At most one complete pushback is assigned to each departure. This definition is applied consistently to the automatically generated development references and manually annotated test references.

2.1.1 Development Dataset

Development data were collected for departures from Stockholm Arlanda Airport (ESSA), Zurich Airport (LSZH), Tokyo Haneda Airport (RJTT), and Perth Airport (YPPH). These airports were selected based on ground coverage, availability of the directional information required for reference labelling, and the number of fully observed pushback manoeuvres. Departures were identified using the track-based method of the `takeoff()` function provided by the *traffic* library. To capture seasonal variation, data were collected for each local meteorological season using one week per season for ESSA, LSZH, and RJTT and three consecutive weeks per season for YPPH because of its lower flight volume.

Construction of the development dataset comprised two stages, summarised in Figure 2: (i) reference generation using trajectories retrieved directly from the OpenSky historical database, and (ii) model-input construction using `traffic.history()`. Direct database access was required because reference generation uses directional information from the heading field of the `position_data4` table.

Reference Generation. A dedicated rule-based procedure was developed to generate reference segments for model development. It exploits the difference between aircraft heading, which describes the orientation of the aircraft's longitudinal axis, and ground track, which describes its direction of movement. The two are generally similar during forward ground movement but typically differ by approximately 180° during backward movement within a pushback manoeuvre. The rule-based procedure comprised four steps, as summarised in Figure 2.

- 1. Candidate pushback detection and validation.** Each database trajectory was resampled to 1 Hz, and ground speed, ground track, and cumulative distance were derived from consecutive positions using the `cumulative_distance()` function of *traffic*. The derived speed and track are hereafter referred to as position-derived ground speed and position-derived ground track to distinguish them from reported OpenSky values. Trajectories with no more than one distinct reported directional value over the first 0.06 NM of cumulative distance were excluded because they contained insufficient directional information for reference labelling. Candidate pushback observations were identified where the absolute circular difference between the reported directional value and position-derived ground track exceeded 100°, position-derived ground speed was between 2.5 kt and 15 kt, and the change in position-derived ground track between consecutive observations was below 5°. The speed range restricts candidates to low-speed ground movement, while the track-stability criterion reduces false detections due to position noise. If no candidate

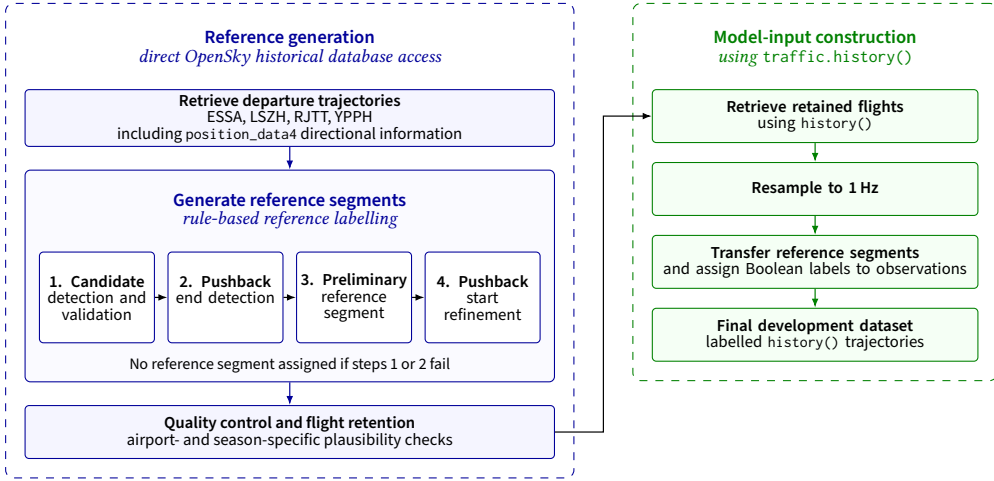


Figure 2. Construction of the development dataset. During reference generation, departure trajectories are retrieved directly from the OpenSky historical database and additional directional information from the `position_data4` table is used to generate and quality-control pushback reference segments. During model-input construction, the retained flights are retrieved using `traffic.history()`, and the reference segments are transferred to these trajectories to obtain the labelled development dataset used for the development of the pushback detection pipeline.

observation was found, no reference segment was assigned. Otherwise, the median timestamp of the candidate observations defined the pushback reference time, reducing sensitivity to isolated observations caused by noise. A 60 s interval centred on this time was extracted from the unfiltered trajectory, and the candidate pushback was retained only if the median absolute circular difference between the reported directional value and position-derived ground track across all observations in this interval was at least 60° . Unlike the preceding 100° criterion, this threshold applies to all observations within the interval rather than only the candidate observations.

2. **Pushback end detection.** The unfiltered trajectory was searched forward from the pushback reference time for observations where the absolute change in position-derived ground speed between consecutive observations was below 9×10^{-8} kt. This numerical tolerance identifies effectively unchanged ground speed and serves as a proxy for stationary observations. Such observations were grouped when separated by no more than 2 s, and groups lasting at least 10 s were retained as candidate stationary periods. If none was found, no reference segment was assigned. Multiple candidate stationary periods were considered chronologically. If the median absolute circular difference between the reported directional value and position-derived ground track between two consecutive stationary periods exceeded 100° , the earlier period was interpreted as an intermediate stop and discarded, allowing multi-stage pushbacks to be bridged. The beginning of the first candidate stationary period not discarded by this criterion defined the pushback end.
3. **Construction of a preliminary reference segment.** The preliminary reference segment comprised all observations from the beginning of the available unfiltered trajectory to the identified pushback end. Segments shorter than 30 s were retained unchanged because the available trajectory was considered likely to have begun after the pushback had already started. For longer segments, the start was refined.
4. **Pushback start refinement.** The initial aircraft position was estimated as the median latitude and longitude over the first 15 s. Observations more than 30 m from this position and with position-derived ground speed above 0.5 kt were retained, and their median timestamp defined the start-refinement reference time. If no observation satisfied these conditions, the preliminary reference segment was retained unchanged. To estimate the pushback start, the prelimi-

nary reference segment was independently restricted to observations with an absolute change in position-derived ground speed exceeding 8×10^{-9} kt, thereby excluding effectively unchanged speed. Starting from the start-refinement reference time, these observations were traversed backwards in time until a temporal gap exceeding 10 s was encountered. Because observations with effectively unchanged position-derived ground speed had been removed, the first observation following the first temporal gap exceeding 10 s encountered during the backward traversal was interpreted as the transition from stationary to moving conditions and taken as the estimated pushback start. If no such gap was found, the earliest remaining observation was used. The estimated start was shifted 20 s backwards, and the final reference segment was extracted from the unfiltered trajectory between this adjusted start and the previously identified pushback end. If refinement could not be performed, the preliminary reference segment was retained unchanged.

The 20 s margin recovers observations removed by the preceding filters and reduces the risk of excluding the onset of movement, but may include a short preceding stationary interval. The automatically generated reference segments therefore approximate the pushback definition and are intended primarily to identify the presence and approximate temporal location of a pushback rather than its exact boundaries. All thresholds, minimum durations, and spatial and temporal margins were selected empirically.

The generated reference segments were subjected to quality control based on travelled distance, cumulative trajectory distance at the segment start, segment duration, average speed calculated from distance and duration, and maximum, mean, and median position-derived ground speed. Airport- and season-specific plausibility thresholds were applied, and flights violating any criterion were removed. Trajectories for which no reference segment was generated were additionally inspected visually in map-based batches for each airport and season, and flights containing a missed pushback were removed.

Model-Input Construction. After quality control, the retained flights were retrieved using the `history()` function of *traffic* and resampled to 1 Hz. The reference segments generated from the corresponding database trajectories were transferred to these trajectories by assigning each observation a Boolean reference label indicating whether it belonged to the reference segment. Because the database and `history()` trajectories did not always start at the same time, their temporal coverage was reconciled during mapping. Where the database trajectory started earlier, only the portion of the reference segment covered by the `history()` trajectory was retained, including cases where this reduced the segment to a single observation. Where the `history()` trajectory started earlier, it was truncated to the start of the database trajectory because no reference information was available for preceding observations. The resulting labelled `history()` trajectories constitute the final development dataset and are summarised in Table 1.

Table 1. Airport-specific observation periods and numbers of labelled `history()` trajectories in the final development dataset. Seasons refer to the local meteorological seasons at each airport.

Airport	Spring	Summer	Autumn	Winter
ESSA	07–13 Apr 2025 (1577)	07–13 Jul 2025 (1562)	06–12 Oct 2025 (1758)	13–19 Jan 2026 (1283)
LSZH	20–26 Apr 2026 (2464)	21–27 Jul 2025 (2334)	20–26 Oct 2025 (2214)	26 Jan–01 Feb 2026 (1798)
RJTT	19–25 May 2025 (2693)	18–24 Aug 2025 (2858)	17–23 Nov 2025 (2207)	16–22 Feb 2026 (3001)
YPPH	03–23 Nov 2025 (646)	02–22 Feb 2026 (581)	05–25 May 2025 (819)	04–24 Aug 2025 (1031)

2.1.2 Independent Test Dataset

The independent test dataset comprised departures from Brussels Airport (EBBR), Boston Logan International Airport (KBOS), and Aeroparque Jorge Newbery (SABE). These airports were selected based on suitable ground coverage and sufficient numbers of departures with and without pushback. Observation days between 1 January 2025 and 30 June 2026 were selected to provide coverage across local meteorological seasons where possible, with approximately 200–300 flights targeted per represented season. Two days per season were selected for SABE and generally one for EBBR and KBOS. The final KBOS selection was an exception, comprising two spring days and no summer day.

Reference segments were annotated manually using custom annotation software¹ and the pushback definition introduced above. For each identifiable pushback, the start and end of the complete manoeuvre were marked. Individual push and pull movements within multi-stage or push-pull manoeuvres were additionally annotated where identifiable but were not used for quantitative evaluation. Trajectories for which the presence or temporal extent of a pushback could not be determined with sufficient confidence were labelled *not clear* and excluded from quantitative evaluation. Additional tags documented unusual or noteworthy cases for qualitative inspection but were neither systematically validated nor used in the quantitative evaluation. The observation periods and numbers of manually labelled trajectories contained in the test dataset are reported in Table 2.

Table 2. Airport-specific observation periods and numbers of manually labelled trajectories in the independent test dataset. Seasons refer to the local meteorological seasons at each airport.

Airport	Spring	Summer	Autumn	Winter
EBBR	18 Apr 2025 (233)	17 Jul 2025 (276)	05 Nov 2025 (223)	05 Dec 2025 (236)
KBOS	17 Mar 2026 25 May 2026 (623)	–	18 Oct 2025 (271)	08 Feb 2025 (235)
SABE	10 Oct 2025 29 Nov 2025 (281)	14 Jan 2025 11 Dec 2025 (260)	11 Mar 2025 27 Apr 2025 (257)	08 Aug 2025 04 Jun 2026 (272)

All reference annotations were performed by a single annotator. Consequently, the annotations do not provide information on inter-annotator agreement, and the interpretation of individual cases and exact segment boundaries may have evolved with increasing annotation experience. The manually annotated test dataset is publicly available on Zenodo [11].

2.2 Pushback Detection Pipeline

The proposed pipeline operates on individual departure trajectories retrieved using the `history()` function of `traffic` and identifies at most one pushback segment per departure from position-based trajectory information. It comprises trajectory preprocessing and feature extraction, pointwise pushback classification, and conversion of the resulting probabilities into a pushback segment. The pipeline neither requires the directional information used for reference generation nor airport stand or apron geometries.

2.2.1 Trajectory Preprocessing and Input Features

Each trajectory is first restricted to the portion relevant to the departure airport by truncating it at the last observation within the airport footprint with geometric altitude (`geoaltitude`) either below 500 ft above airport elevation or unavailable. The airport footprint and elevation are obtained from

¹The annotation software is available in the project repository at https://github.com/hanvkev/osn26_pushback_detection/tree/main/01_data_generation/02_test/02_label_data/pushback_labeler.

the airport information provided by the *traffic* library, with the footprint based on OpenStreetMap data. Geometric altitude is taken directly from the input trajectory. The airport footprint, elevation, and geometric altitude are used only for preprocessing and are not provided to the model as input features.

Following truncation, ground speed and ground track are derived from consecutive aircraft positions using the `cumulative_distance()` function of *traffic*. As in Section 2.1, these quantities are referred to as position-derived ground speed and position-derived ground track. All candidate model inputs are derived from these quantities and therefore require no additional directional information reported by the aircraft.

The candidate input features comprise (i) position-derived ground speed, (ii) its change between consecutive observations, (iii) the absolute circular difference in position-derived ground track between consecutive observations, and (iv) longer-term directional change, calculated as the absolute circular difference between the circular mean ground tracks over windows of up to n moving observations immediately before and from each observation onward. One or several window sizes can be used, with each contributing a separate model input.

A ground-speed threshold of 0.5 kt is applied during feature construction to reduce the influence of position noise while the aircraft is stationary. Ground-speed-change and pointwise track-difference values are set to zero below this threshold. For the longer-term directional-change features, observations at or below the threshold are excluded before the windows are formed, such that each circular mean is calculated from up to n observations with ground speed above the threshold. The input-feature combination and longer-term window sizes were selected during model development, see Section 2.3.

2.2.2 Pointwise Pushback Classification

Pushback detection is formulated as a pointwise binary classification problem in which the model assigns each trajectory observation a probability of belonging to a pushback segment. A temporal convolutional network (TCN) is used because it preserves the temporal resolution of the input sequence while dilated convolutions provide a large temporal context at moderate network depth [12, 13]. Since detection is performed retrospectively rather than in real time, the network is non-causal and can use trajectory information both before and after each observation.

The resulting model, referred hereafter to as PRD-TCN (pointwise residual dilated temporal convolutional network), consists of eight residual blocks with 64 filters, a kernel size of five, and dilation rates of [1, 2, 4, 8, 16, 32, 64, 96]. Each block contains two dilated convolutions with symmetric padding, each followed by a rectified linear unit activation and dropout, with a residual connection around the block. The residual stack is followed by a pointwise prediction head consisting of two 1×1 convolutions with 64 hidden units and a rectified linear unit activation between them. The final convolution produces one logit per observation, which is transformed using the sigmoid function to obtain the corresponding pushback probability. The theoretical receptive field spans 1785 observations, corresponding to approximately 29.8 min at the 1 Hz sampling rate. The model was implemented in *PyTorch* [14].

Each flight is processed as a single sequence. Within a batch, sequences are padded to the length of the longest sequence, with a mask preventing padded observations from contributing to the loss or performance calculations. Input features are standardised to zero mean and unit variance using the `StandardScaler` implementation of *scikit-learn* [15]. Scaler fitting, model training, and selection of variable model settings are described in Section 2.3.

2.2.3 Pushback Segment Extraction 317

The pointwise probabilities produced by PRD-TCN are converted into at most one predicted pushback segment per departure using a two-stage procedure. First, a core segment representing the principal model prediction is identified using a comparatively strict probability threshold. Second, its temporal boundaries are expanded using a lower threshold, allowing observations with lower probabilities to contribute to the segment boundaries without determining whether a pushback is detected. 318
319
320
321
322
323

To identify the core segment, observations at or above a core probability threshold are treated as candidate pushback observations and grouped into the same candidate segment when the intervening interval without an observation reaching the threshold does not exceed a specified maximum interruption. This permits temporary interruptions in the model output to be bridged, consistent with the inclusion of intermediate stationary periods in the pushback definition of Section 2.1. If several candidate segments remain, the segment with the highest mean predicted probability is selected. Ties are resolved first in favour of the longer segment and then of the segment with the higher maximum probability. If all three criteria are identical, the earlier segment is selected. If no observation reaches the core threshold, no pushback is predicted. 324
325
326
327
328
329
330
331
332

Starting from the core segment, edge expansion proceeds independently from its start and end over adjacent observations at or above an edge probability threshold, permitting interruptions up to a specified maximum duration. Expansion in either direction terminates when this limit is exceeded. Because expansion proceeds only outwards from an existing core segment, it can modify prediction boundaries but can neither create nor remove a pushback detection. No minimum duration is imposed on the resulting predicted segment. 333
334
335
336
337
338

The core probability threshold and maximum permitted interruption are selected during model development. The edge probability threshold and maximum permitted interruption are fixed at 0.5 and 30 s, respectively. These edge-expansion parameters were set a priori rather than optimised because the development reference segments were not intended to provide precise pushback boundaries. 339
340
341
342

2.3 Model Development and Configuration Selection 343

Model development used only the development dataset and comprised three stages: definition of the training procedure, leave-one-airport-out selection of input features and model hyperparameters, and final model training and post-processing configuration selection. 344
345
346

Hereafter, *model configuration* refers to the selected input features and model hyperparameters, while *post-processing configuration* refers to the core probability threshold and corresponding maximum permitted interruption. Together, these define the variable components of the pipeline configuration. 347
348
349
350

2.3.1 Training Procedure 351

Trajectories from each development airport were independently split into 85 % training and 15 % validation subsets, stratified by the presence of a reference pushback segment. These fixed airport-specific splits were reused throughout model development and combined as required while preserving their assignments. 352
353
354
355

For each model fit, input features were standardised using a scaler fitted exclusively on the corresponding training subset. The model was optimised using Adam and weighted binary cross-entropy. To account for the imbalance between pushback and non-pushback observations, class weights were calculated from the training data as 356
357
358
359

$$w_0 = \frac{N}{2N_0}, \quad w_1 = s \frac{N}{2N_1}, \quad (1)$$

where $N = N_0 + N_1$, N_0 and N_1 are the numbers of non-pushback and pushback observations, respectively, and s is a positive-class scaling factor treated as a hyperparameter.

Training was performed for a maximum of 200 epochs with a batch size of 128. The model state with the highest pointwise validation area under the precision–recall curve (AUPRC) was retained. If validation AUPRC did not improve for four consecutive epochs, the learning rate was halved to a minimum of 10^{-5} . Training was stopped after ten consecutive epochs without improvement. Initial learning rate, dropout rate, and positive-class weight scaling factor s were variable hyperparameters; all remaining training settings were fixed.

2.3.2 Leave-One-Airport-Out Model Configuration Selection

Input features and model hyperparameters were selected using leave-one-airport-out experiments across the four development airports. For each candidate configuration, four models were trained, each using the training subsets of three airports for fitting, their validation subsets for early stopping, and all flights from the remaining airport for out-of-fold prediction.

The model-configuration space in Table 3 was sampled using 30 trials of Bayesian optimisation with *Weights & Biases* [16]. For each trial, the resulting out-of-fold predictions were evaluated over the complete grid of core probability thresholds and maximum permitted interruptions reported in the table, without retraining the models. Edge expansion was excluded from optimisation because the development references were not intended to provide precise pushback boundaries.

Each resulting pipeline configuration, defined by a model configuration and a core post-processing configuration, was evaluated using

$$J = 0.50 \text{BA}_{\text{flight}} + 0.20 F_{1,\text{IoU} \geq 0.50} + 0.30 F_{1,\text{IoU} \geq 0.75}, \quad (2)$$

where $\text{BA}_{\text{flight}}$ denotes flight-level balanced accuracy and $F_{1,\text{IoU} \geq t}$ the segment-level F_1 -score at temporal IoU threshold t (Section 2.4). Metrics were calculated separately for each development airport and macro-averaged across airports. The manually selected weights prioritise pushback detection while additionally rewarding temporal localisation, with greater weight on the stricter IoU threshold. Near-perfect boundary agreement was not prioritised because the automatically generated development references provide only approximate pushback boundaries.

The model configuration associated with the highest-scoring pipeline configuration was selected for final model training. Table 3 summarises the model and post-processing search spaces, together with the fixed model settings.

2.3.3 Final Model Training

The selected model configuration was used to train a final PRD-TCN model using the combined training subsets of all four development airports, with the corresponding validation subsets used for early stopping. Feature standardisation was fitted exclusively on the combined training data.

The final core post-processing configuration was selected from the pooled development-validation predictions by maximising Equation 2 over the same grid used during leave-one-airport-out selection. The fixed edge-expansion parameters were then applied, and the resulting pipeline was fixed before evaluation on the independent test dataset.

2.4 Evaluation

The final PRD-TCN pipeline was evaluated exclusively on the independent manually labelled test dataset described in Section 2.1.2. No model, feature, or post-processing parameter was adjusted using the test data. Performance was evaluated in terms of pushback detection and temporal segment

Table 3. Model and post-processing search spaces and selected values, together with fixed model and edge-expansion settings.

Parameter	Search space or setting	Selected value
Bayesian search		
Feature variant	GS + T; GS + G + T; GS + C; GS + G + C; GS + T + C; GS + G + T + C	GS + G + T + C
Track-window preset	5, 15, 30, 5/15, 10/30, 5/15/30, 10/30/60	5/15/30
Learning rate	3×10^{-4} – 3×10^{-3} (log-uniform)	2.963×10^{-3}
Dropout	0.10–0.30 (uniform)	0.256
Positive-class weight scale	0.50–1.00 (uniform)	0.524
Core post-processing grid search		
Core probability threshold	0.35–1.00 (step 0.01)	0.84
Maximum permitted interruption	0, 15, 30, 45, 60, 90 s	15 s
Fixed model settings		
TCN filters	–	64
Kernel size	–	5
Dilations	–	1, 2, 4, 8, 16, 32, 64, 96
Dense units	–	64
Ground-speed threshold	–	0.5 kt
Final edge expansion		
Edge probability threshold	–	0.5
Edge maximum interruption	–	30 s

Note: GS = ground speed; G = ground-speed difference; T = track difference; C = circular track change. Track-window presets apply only to circular feature variants. A dash indicates a fixed or non-optimised parameter. Track windows are specified in observations; interruption durations are specified in seconds.

localisation. PRD-TCN was additionally compared with the existing rule-based pushback detection methods implemented in the *traffic* library.

2.4.1 Comparison Methods

PRD-TCN was compared with the *parking_position* and *parking_area* pushback detection methods provided by the *traffic* library. Both methods infer pushback manoeuvres from aircraft movement in relation to airport parking infrastructure and therefore provide rule-based baselines for PRD-TCN. The baseline methods were applied using their default parameters. All three methods were applied to the same trajectories and evaluated against the same manually annotated reference segments.

2.4.2 Pushback Detection Evaluation

Pushback detection was evaluated at the flight level. A predicted and reference segment were considered to overlap only if their temporal overlap had a duration greater than zero. A prediction was considered a true positive (TP) if the departure contained a manually annotated pushback and the predicted and reference segments overlapped. A reference pushback without an overlapping prediction was counted as a false negative (FN), while a prediction without an overlapping reference pushback was counted as a false positive (FP). Consequently, a non-overlapping prediction on a departure containing a reference pushback contributed one FP and one FN. A true negative (TN) was a departure without a reference pushback for which no segment was predicted. Departures labelled *not clear* during manual annotation were excluded from the quantitative evaluation.

Detection performance was quantified using precision, recall, F_1 -score, specificity, and balanced accuracy. Specificity was calculated over departures without a reference pushback, while balanced accuracy was calculated as the mean of recall and specificity. All metrics were calculated separately for each test airport and macro-averaged across airports, giving each airport equal weight irrespective of its number of trajectories.

2.4.3 Segment-Level Evaluation

Temporal localisation was evaluated for true-positive detections, defined as predicted segments with a temporal overlap greater than zero with the corresponding reference segment. Boundary timing error was defined as

$$e_t = t_{\text{pred}} - t_{\text{ref}}, \quad (3)$$

and evaluated separately for the start and end of each segment. Negative values indicate that the predicted boundary occurred earlier than the reference boundary, whereas positive values indicate a later prediction. Mean signed error and mean absolute error (MAE) were reported for both boundaries.

Overall temporal agreement was quantified using temporal intersection-over-union (IoU),

$$\text{IoU} = \frac{|S_{\text{pred}} \cap S_{\text{ref}}|}{|S_{\text{pred}} \cup S_{\text{ref}}|}, \quad (4)$$

where S_{pred} and S_{ref} denote the predicted and reference segments, respectively, and $|\cdot|$ denotes temporal duration. Boundary-error metrics and mean IoU were calculated over all true-positive detections of each method and additionally in paired comparisons between PRD-TCN and each rule-based baseline using only departures for which both methods produced a true-positive detection. Segment-level F_1 -scores were further determined at minimum IoU thresholds of 0.50 and 0.90, with predicted-reference pairs below the respective threshold contributing one FP and one FN instead of a TP.

3. Results

This section contains the performance of the PRD-TCN pipeline and the two rule-based comparison methods on the independent test dataset.

Table 4 presents the pushback detection results for the three test airports and their macro averages, reporting precision, recall, F_1 -score, specificity, and balanced accuracy. At minimum IoU thresholds of 0.50 and 0.90, the macro segment F_1 -scores were 0.943 and 0.504 for the PRD-TCN pipeline, compared with 0.627 and 0.341 for parking_area and 0.321 and 0.017 for parking_position, respectively.

Tables 5 and 6 present the temporal localisation results. Table 5 reports mean IoU together with the absolute and signed start and end timing errors for each test airport and their macro averages. Table 6 contains the corresponding paired macro-average results for the common true-positive detections, including the number of common detections, mean IoU, and start and end MAE.

4. Discussion

With respect to the first research question of whether pushbacks can be detected and temporally delineated from openly available ADS-B trajectories at airports unseen during pipeline development, the results provide evidence that the PRD-TCN pipeline can generalise to such airports. Across the three independent test airports, the pipeline achieves a macro precision of 0.985 and a macro recall of 0.982, resulting in a macro F_1 -score of 0.984. Performance is also consistent across the

Table 4. Pushback detection performance on the independent test dataset.

Airport	Method	Precision	Recall	F_1	Specificity	Balanced Accuracy
EBBR	PRD-TCN pipeline	0.995	0.994	0.995	0.973	0.983
	parking_area	0.934	0.730	0.820	0.864	0.797
	parking_position	0.709	0.460	0.558	0.973	0.717
KBOS	PRD-TCN pipeline	0.992	0.985	0.988	0.981	0.983
	parking_area	0.782	0.262	0.392	0.959	0.611
	parking_position	0.842	0.485	0.615	0.989	0.737
SABE	PRD-TCN pipeline	0.969	0.967	0.968	0.976	0.972
	parking_area	0.926	0.690	0.791	0.957	0.824
	parking_position	0.262	0.101	0.146	0.974	0.538
Macro	PRD-TCN pipeline	0.985	0.982	0.984	0.977	0.979
	parking_area	0.881	0.561	0.668	0.927	0.744
	parking_position	0.604	0.349	0.440	0.979	0.664

Table 5. Temporal segment quality for true-positive pushback detections.

Airport	Method	Mean IoU	Start MAE [s]	Mean Start Error [s]	End MAE [s]	Mean End Error [s]
EBBR	PRD-TCN pipeline	0.847	5.1	-2.0	10.0	6.4
	parking_area	0.772	12.7	-9.1	14.7	10.8
	parking_position	0.659	14.7	13.6	23.9	2.2
KBOS	PRD-TCN pipeline	0.847	6.3	1.2	15.6	-5.2
	parking_area	0.891	6.1	-2.1	9.1	3.7
	parking_position	0.581	26.5	25.9	36.7	-19.0
SABE	PRD-TCN pipeline	0.896	4.0	2.4	4.9	0.0
	parking_area	0.881	3.3	2.7	7.2	0.9
	parking_position	0.681	24.5	24.5	16.9	7.0
Macro	PRD-TCN pipeline	0.864	5.1	0.5	10.2	0.4
	parking_area	0.848	7.4	-2.8	10.3	5.1
	parking_position	0.640	21.9	21.3	25.8	-3.3

Table 6. Paired comparison of temporal segment quality on common true-positive detections.

Comparison	Method	n	Mean IoU	Start MAE [s]	End MAE [s]
vs. parking_area	PRD-TCN pipeline	1198	0.883	4.6	7.9
	parking_area	1198	0.850	7.4	10.3
vs. parking_position	PRD-TCN pipeline	857	0.875	5.6	9.9
	parking_position	857	0.640	21.9	25.9

individual airports, with F_1 -scores ranging from 0.968 at SABE to 0.995 at EBBR. Together with a macro specificity of 0.977 and a macro balanced accuracy of 0.979, these results show that the pipeline identifies most labelled pushbacks while only rarely predicting a pushback on flights without one. The macro F_1 -scores of `parking_area` and `parking_position` are 0.668 and 0.440, respectively. The largest difference occurs in recall, where the two baseline methods achieve 0.561 and 0.349 compared with 0.982 for the PRD-TCN pipeline.

One possible explanation for the consistent performance across the independent test airports is that the PRD-TCN pipeline does not depend on predefined stand or apron geometries or prescribed directional changes. Instead, it learns temporal movement patterns from labelled trajectories over a theoretical receptive field of approximately 29.8 min. This broader temporal context may increase robustness when individual movement cues are noisy, incomplete, or only partially observable, whereas such conditions may prevent rule-based methods from satisfying their required geometric or directional criteria. The results therefore provide evidence that the proposed pipeline can generalise to airports not included in its development.

Regarding the second research question of how the quality of pushback detection and temporal localisation compares with the rule-based methods implemented in the *traffic* library, differences between the PRD-TCN pipeline and the rule-based methods are smaller for temporal localisation than for pushback detection. At minimum IoU thresholds of 0.5 and 0.9, the PRD-TCN pipeline achieves macro segment F_1 -scores of 0.943 and 0.504, respectively, compared with 0.627 and 0.341 for `parking_area` and 0.321 and 0.017 for `parking_position`. The smaller difference to `parking_area` at the stricter threshold reflects the greater difficulty of closely matching the labelled segment boundaries.

Among each method's own true-positive detections, the PRD-TCN pipeline achieves a macro mean IoU of 0.864, compared with 0.848 for `parking_area` and 0.640 for `parking_position`. Its macro start MAE is 5.1 s, compared with 7.4 s and 21.9 s, while its end MAE of 10.2 s is nearly identical to the 10.3 s of `parking_area` and lower than the 25.8 s of `parking_position`. These comparisons must be interpreted cautiously because each method is evaluated on a different set of successfully detected pushbacks.

The paired comparison addresses this limitation by restricting each comparison to flights detected by both methods. Across the 1,198 detections shared by the PRD-TCN pipeline and `parking_area`, their macro mean IoUs are 0.883 and 0.850, respectively, with start MAEs of 4.6 s and 7.4 s and end MAEs of 7.9 s and 10.3 s. Across the 857 detections shared with `parking_position`, the corresponding mean IoUs are 0.875 and 0.640, start MAEs 5.6 s and 21.9 s, and end MAEs 9.9 s and 25.9 s. The paired results therefore indicate that the differences in temporal localisation are not solely attributable to the methods being evaluated on different sets of successfully detected pushbacks.

The boundary-error metrics provide further insight into the temporal localisation of the PRD-TCN pipeline. Its macro start MAE of 5.1 s is lower than the end MAE of 10.2 s, indicating more precise localisation of the pushback start. One possible contributor is that trajectories beginning after pushback initiation necessarily restrict the reference start to the first available observation, whereas trajectories generally extend beyond the pushback end. Despite the larger end errors, the signed errors suggest little overall directional bias across the three test airports: the macro mean errors are only 0.5 s for the start and 0.4 s for the end. The small signed errors relative to the corresponding MAEs suggest that both start and end timing errors occur in both directions, rather than showing a consistent tendency towards early or late detection.

From a practical perspective, separating pushback detection from application-specific selection criteria allows the same pipeline to support different downstream analyses. The detection itself does not require predefined stand or apron geometries or airport-specific movement rules and can also

return short or partial pushback segments when only part of the manoeuvre is observable. Depending on the application, such detections can subsequently be filtered. For example, Stloukal, Had, and Hospodka [6] retained only trajectories for which the beginning of the pushback could be observed at the stand. Similar requirements can be imposed after detection based on the amount of trajectory available before the detected pushback start or, where stand information is available, the distance of the initial observations from a parking position. The detection procedure can thus remain independent of application-specific selection criteria.

A further practical advantage of the PRD-TCN pipeline is its limited reliance on airport-specific information required for pushback detection. Application of the complete pipeline requires time-stamped aircraft positions and geometric altitude. For trajectory preprocessing, the airport footprint and elevation are obtained from the airport information provided by the *traffic* library. The model itself operates exclusively on movement features derived from the aircraft positions. Airport footprint, elevation, and geometric altitude are not provided as model input features. The pipeline can, therefore, in principle, also be applied to trajectory data from other sources, provided that equivalent position and altitude information is available. The post-processing parameters operate on the probability sequence generated by the trained model rather than encoding a particular stand geometry or prescribed directional definition of a pushback.

The temporal accuracy achieved by the PRD-TCN pipeline is also relevant for its use in downstream analyses. The practical importance of the observed boundary deviations depends on the temporal scale of the intended analysis. For aggregate analyses of stand and apron utilisation, ground traffic, taxi-out operations, fuel consumption, or emissions across large numbers of flights, deviations of several seconds may be less consequential than for applications that depend on precise event timing. The latter includes comparisons with operational milestones such as those used in Airport Collaborative Decision Making (A-CDM), for which the less precise end boundary may be particularly relevant.

An important limitation concerns the automatically generated development labels, which rely on empirically selected thresholds and approximate the presence and temporal location of pushbacks rather than providing manually verified labels for every training trajectory. Missed pushbacks identified during visual inspection, false positives identified by the plausibility filters, and trajectories with insufficient variation in reported directional values were excluded rather than manually relabelled. While this reduced label errors, it may have underrepresented movement patterns that were not reliably captured by the labelling procedure. Conversely, incorrectly localised pushbacks satisfying the plausibility criteria could remain in the development dataset. The emphasis on the push phase introduces a further limitation: late-start trajectories showing only a pull movement cannot be identified, while subsequent pull phases of push-pull manoeuvres may be truncated or omitted. Overall, the generated labels are more reliable for identifying the presence and approximate location of pushbacks than their exact boundaries. This uncertainty is particularly relevant to the temporal localisation results and may contribute to the larger observed end errors.

Several limitations also arise from the optimisation procedure. The edge threshold of 0.5 and its maximum permitted interruption of 30 s were fixed a priori rather than optimised because the automatically generated labels are less reliable with respect to exact segment boundaries. These settings should therefore not be interpreted as globally optimal for boundary localisation and may also contribute to the variation in boundary accuracy observed across the test airports. In addition, the same validation flights were used both for early stopping of the final model and for selecting its core post-processing parameters. This reuse may lead to a somewhat optimistic estimate of the suitability of the selected core parameters for unseen data. The Bayesian optimisation comprised 30 trials and therefore sampled rather than exhaustively evaluated the model and feature search space. Other configurations within the search space may therefore perform comparably or better.

A further limitation concerns the scope of the evaluation data. Pipeline development used four airports and the independent evaluation three additional airports, all of which required suitable surface surveillance coverage. The results therefore provide evidence of transfer to airports not used during development, but they do not establish generalisability across the full range of airport layouts, operating procedures, pushback patterns, and surveillance conditions. In particular, robustness to substantially poorer surface coverage, larger data gaps, or stronger positional inaccuracies was not investigated separately. Similarly, the behaviour of the pipeline for operational concepts in which an aircraft is towed beyond what would conventionally be considered the pushback manoeuvre, including alternative ground propulsion concepts discussed for instance in [4], was not empirically assessed. Finally, 23 of the 3,167 manually labelled trajectories were considered ambiguous, marked as *not clear*, and excluded from the quantitative evaluation. The behaviour of the pipeline on such ambiguous or atypical operations was therefore not assessed quantitatively.

5. Conclusion and Outlook

In this study, we investigated whether aircraft pushback segments can be identified and temporally delimited from openly available ADS-B trajectory data at airports not used during development of the detection pipeline. We further assessed how the resulting detections and temporal segment boundaries compare with the rule-based pushback detection methods implemented in the *traffic* library. To this end, we developed a pipeline combining trajectory preprocessing, a pointwise residual dilated temporal convolutional network (PRD-TCN), and post-processing of the resulting pointwise probabilities. The pipeline was developed using data from four airports (ESSA, LSZH, RJTT, and YPPH), and independently evaluated on a manually labelled reference dataset from EBBR, KBOS, and SABE. Across these three test airports, the PRD-TCN pipeline achieved a macro precision of 0.985, a macro recall of 0.982, a macro F_1 -score of 0.984, a macro specificity of 0.977, and a macro balanced accuracy of 0.979, compared with macro F_1 -scores of 0.668 and 0.440 for *parking_area* and *parking_position*, respectively. The high recall and specificity demonstrate that the pipeline reliably identifies flights with pushback while also correctly recognising flights without pushback. The detected segments achieved a macro mean IoU of 0.864, with start and end MAEs of 5.1 s and 10.2 s. The same trained model and post-processing parameters were applied at all three test airports without airport-specific adaptation.

The results provide evidence that the PRD-TCN pipeline can transfer to airports not used during pipeline development. The advantage over the rule-based methods is clearest for detection, whereas the differences in the temporal localisation of the detected segments are smaller. The detection itself does not require predefined stand or apron geometries or airport-specific movement rules. Application-specific criteria can instead be applied to the detected segments afterwards, allowing the same pipeline to support different analyses of airport surface operations. In addition, the manually labelled reference dataset created for the independent evaluation provides a basis for future research on pushback detection and aircraft ground movements.

Future work should address several aspects of the proposed approach. The post-processing parameters used for boundary extension could be optimised on manually labelled data with more reliable segment boundaries. The training data could also be improved by manually labelling additional trajectories and by applying data augmentation to increase robustness to noisy or incomplete trajectory observations. Future approaches could further distinguish push and pull phases explicitly instead of representing the complete manoeuvre by a single pushback segment. Moreover, the generalisability of the pipeline should be evaluated on a larger and geographically more diverse set of airports, including airports with poorer surface coverage and ambiguous or unconventional ground operations. Finally, the pointwise classification and subsequent segment-generation approach could be extended beyond pushback to additional phases of aircraft ground movement, such as taxiing, runway entry,

and take-off, towards a more comprehensive segmentation of surface trajectories.

Acknowledgement

The authors acknowledge the contributions of two/three reviewers that greatly enhanced the value of this study. No potential conflict of interest was reported by the authors. No funding was received for this research.

Author Contributions

Conceptualization (K.H.), Methodology (K.H.), Software (K.H.), Validation (K.H., M.W.), Data Curation (K.H.), Writing – Original Draft (K.H.), Writing – Review & Editing (K.H., M.W.), Visualization (K.H.), Project Administration (M.W.)

Open Data Statement

The software code used to download the OSN-data employed in this study is available on the following repository: https://github.com/hanvkev/osn26_pushback_detection. The manually labelled test dataset is available on Zenodo [11].

Reproducibility Statement

The software code used to generate the results presented in this paper is available on the following repository: https://github.com/hanvkev/osn26_pushback_detection.

References

- [1] EUROCONTROL. *EUROCONTROL aviation long-term outlook: Flights and CO₂ emissions forecast 2024–2050*. EUROCONTROL, Dec. 2024. URL: <https://www.eurocontrol.int/publication/eurocontrol-forecast-2024-2050>.
- [2] Junzi Sun. *The 1090 megahertz riddle: A guide to decoding Mode S and ADS-B signals*. 2nd ed. TU Delft OPEN Publishing, 2021. ISBN: 978-94-6366-402-8. DOI: 10.34641/MG.11.
- [3] Junzi Sun, Joost Ellerbroek, and Jacco Hoekstra. “Flight extraction and phase identification for large automatic dependent surveillance–broadcast datasets”. In: *Journal of Aerospace Information Systems* 14.10 (Oct. 2017), pp. 566–571. DOI: 10.2514/1.I010520.
- [4] Manuel Waltert, Benoit Figuet, and Michael Felux. “Evaluating potential fuel-savings of external alternative ground propulsion systems”. In: *Journal of Open Aviation Science* 2 (May 12, 2025). Proceedings of the 12th OpenSky Symposium, pp. 1–17. DOI: 10.59490/joas.2024.7888.
- [5] Xavier Olive. “traffic, a toolbox for processing and analysing air traffic data”. In: *Journal of Open Source Software* 4.39 (2019), p. 1518. ISSN: 2475-9066. DOI: 10.21105/joss.01518.
- [6] Bo Stloukal, Petr Had, and Jakub Hospodka. “Analyzing the duration of aircraft pushback operations using ADS-B data”. In: *Proceedings of the 2026 New Trends in Civil Aviation (NTCA)*. 23–24 April 2026. Prague, Czech Republic: IEEE, Apr. 2026, pp. 113–120. DOI: 10.23919/NTCA68808.2026.11524162.
- [7] J. Hoole, J. D. Booker, and J. E. Cooper. “Impact of operator characteristics on landing gear ground manoeuvre occurrences”. In: *The Aeronautical Journal* 128 (2024), pp. 2604–2633. DOI: 10.1017/aer.2024.93.
- [8] Xavier Olive, Luis Basora, Junzi Sun, and Enrico Spinielli. “Training a machine learning model to detect holding patterns in aircraft trajectories”. In: *Journal of Open Aviation Science* 2.2 (Mar. 19, 2025). Proceedings of the 12th OpenSky Symposium. DOI: 10.59490/joas.2024.7943.

- [9] Gabriel Jarry, Philippe Very, Ramon Dalmau, Daniel Delahaye, and Arthur Houdant. “On the detection of aircraft single engine taxi using deep learning models”. In: *Proceedings of the 14th SESAR Innovation Days*. Paper 2024-029. Rome, Italy: SESAR Joint Undertaking, Nov. 2024, pp. 1–8. URL: https://www.sesarju.eu/sites/default/files/documents/sid/2024/papers/SIDs_2024_paper_029%20final.pdf.
- [10] Matthias Schäfer, Martin Strohmeier, Vincent Lenders, Ivan Martinovic, and Matthias Wilhelm. “Bringing up OpenSky: A large-scale ADS-B sensor network for research”. In: *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*. IEEE. 2014, pp. 83–94.
- [11] Kevin Hänggi and Manuel Waltert. *Manually Labelled Aircraft Pushback Segments in ADS-B Trajectories*. Version 1.0. Zenodo, 2026. DOI: 10.5281/zenodo.23548217.
- [12] Colin Lea, Michael D. Flynn, René Vidal, Austin Reiter, and Gregory D. Hager. “Temporal convolutional networks for action segmentation and detection”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1003–1012. DOI: 10.1109/CVPR.2017.113.
- [13] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. *An empirical evaluation of generic convolutional and recurrent networks for sequence modeling*. Apr. 19, 2018. arXiv: 1803.01271 [cs.LG]. URL: <https://arxiv.org/abs/1803.01271>.
- [14] Adam Paszke et al. “PyTorch: An imperative style, high-performance deep learning library”. In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019, pp. 8024–8035.
- [15] Fabian Pedregosa et al. “Scikit-learn: Machine learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- [16] Lukas Biewald. *Experiment Tracking with Weights and Biases*. Software available from wandb.com. 2020. URL: <https://www.wandb.com/>.