

Article

Lessons Learned in Transcribing 5000 h of Air Traffic Control Communications for Robust Automatic Speech Understanding

Juan Zuluaga-Gomez ^{1,2,*} , Iuliia Nigmatulina ^{1,3} , Amrutha Prasad ^{1,4}, Petr Motlicek ^{1,4,*} , Driss Khalil ¹, Srikanth Madikeri ¹, Allan Tart ⁵, Igor Szoke ⁴, Vincent Lenders ⁶, Mickael Rigault ⁷ and Khalid Choukri ⁷

¹ Speech & Audio Processing Group, Idiap Research Institute, 1920 Martigny, Switzerland; iuliia.nigmatulina@idiap.ch (I.N.)

² LIDIAP, Ecole Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland

³ Institute of Computational Linguistics, University of Zurich, 8050 Zurich, Switzerland

⁴ Faculty of Information Technology, Brno University of Technology, 60190 Brno, Czech Republic; szoke@replaywell.com

⁵ OpenSky Network, 3400 Burgdorf, Switzerland

⁶ Cyber-Defence Campus, Armasuisse, 3602 Thun, Switzerland

⁷ Evaluations and Language Resources Distribution Agency (ELDA), 75013 Paris, France; mickael@elda.org (M.R.)

* Correspondence: juan-pablo.zuluaga@idiap.ch (J.Z.-G.); petr.motlicek@idiap.ch (P.M.)

Abstract: Voice communication between air traffic controllers (ATCOs) and pilots is critical for ensuring safe and efficient air traffic control (ATC). The handling of these voice communications requires high levels of awareness from ATCOs and can be tedious and error-prone. Recent attempts aim at integrating artificial intelligence (AI) into ATC communications in order to lessen ATCOs's workload. However, the development of data-driven AI systems for understanding of spoken ATC communications demands large-scale annotated datasets, which are currently lacking in the field. This paper explores the lessons learned from the ATCO2 project, which aimed to develop an unique platform to collect, preprocess, and transcribe large amounts of ATC audio data from airspace in real time. This paper reviews (i) robust automatic speech recognition (ASR), (ii) natural language processing, (iii) English language identification, and (iv) contextual ASR biasing with surveillance data. The pipeline developed during the ATCO2 project, along with the open-sourcing of its data, encourages research in the ATC field, while the full corpus can be purchased through ELDA. ATCO2 corpora is suitable for developing ASR systems when little or near to no ATC audio transcribed data are available. For instance, the proposed ASR system trained with ATCO2 reaches as low as 17.9% WER on public ATC datasets which is 6.6% absolute WER better than with "out-of-domain" but gold transcriptions. Finally, the release of 5000 h of ASR transcribed speech—covering more than 10 airports worldwide—is a step forward towards more robust automatic speech understanding systems for ATC communications.

Keywords: air traffic control communications; automatic speech recognition and understanding; OpenSky Network; callsign recognition; ADS-B data



Citation: Zuluaga-Gomez, J.; Nigmatulina, I.; Prasad, A.; Motlicek, P.; Khalil, D.; Madikeri, S.; Tart, A.; Szoke, I.; Lenders, V.; Rigault, M.; et al. Lessons Learned in Transcribing 5000 h of Air Traffic Control Communications for Robust Automatic Speech Understanding. *Aerospace* **2023**, *10*, 898. <https://doi.org/10.3390/aerospace10100898>

Academic Editor: Joost Ellerbroek

Received: 5 September 2023

Revised: 10 October 2023

Accepted: 11 October 2023

Published: 20 October 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

There has been a growing interest in the development of automatic speech recognition (ASR) and understanding systems for air traffic control (ATC) due to their potential to enhance the safety and efficiency of the aviation industry. The application of ASR and understanding technologies in ATC has resulted in the creation of advanced proof-of-concept engines that can assist air traffic controllers (ATCOs) in their daily tasks. These systems are designed to analyze spoken ATC communications and convert them into machine-readable texts, allowing for faster and more accurate processing. Previous works such as MALORCA [1], HAAWAI [2] or SESAR2020's Solution 97.2 [3] have shown mature

enough methods to reduce ATCOs' workload while increasing safeness, e.g., see [4,5]. The authors concluded that integrating novel ASR-based tools can reduce the total amount of time that ATCOs expend on entering and confirming the clearances in their workstations by 20% absolute points. As a result, ASR and understanding technologies are becoming more advanced and capable of handling the complexities of ATC communications, leading to improved safety and efficiency in the aviation industry. The paragraphs below summarize three current challenges—while working with ATC voice communications—addressed by this paper:

(1) Previous works on ASR to analyze air traffic communication is built for a specific domain, e.g., one airport or en-route/approach scenarios. The process of adapting machine learning models to different airports or control areas requires new in-domain data, which remain challenging to collect and annotate. For instance, ATC audio data collected from one airport, e.g., airport X , in general, do not transfer well to airport Y .

(2) ATC data collection and their transcription [6] are expensive and time-consuming tasks. The data collection phase includes data capture and preprocessing; this task can be automated. The data transcription phase aims at producing the word-by-word transcript of the given ATC utterance; this task is carried out by hand by humans. It becomes expensive as several man hours are needed to transcribe one hour of ATC speech without silence. For some solutions, such as those targeting small airports, this cost may be prohibitive. This raises the question of what is the most efficient manner to collect and process large-scale ATC audio data.

(3) In addition, audio data from ATC communication are considerably noisier with regard to standard ASR corpora when they are captured via very-high-frequency (VHF) receivers. In some cases, SNR (signal-to-noise) levels may range from 5 to 20 dB. Thus, it becomes challenging to develop an ASR system and later use its outputs for downstream tasks, e.g., natural language processing (NLP), due to the high word error rates (WER). In contrast, higher SNR ATC data sourced from operation rooms and characterized by close-mic recordings and substantially reduced noise can be obtained from air navigation service providers (ANSPs), albeit being limited to private use in most cases.

In this paper, we answer these questions by extending our previous work on the ATCO2 project and its resulting corpora [7]; see detailed information in Appendix A. The ATCO2 project aimed to reduce the human effort required to collect, preprocess, and transcribe ATC voice communications by employing state-of-the-art ASR and NLP systems [8]. ATCO2 releases the largest corpus of ATC voice communications to date, consisting of more than 5000 h of automatically transcribed audio data and their correspondent surveillance data [9]. In addition, four hours of human-transcribed data (i.e., gold transcriptions) were also released, where we quantified that the transcription process can be significantly accelerated by providing the annotators with automatically transcribed data (i.e., output from an in-domain ASR system), rather than requiring them to produce transcriptions from scratch. According to [7], the real-time factor (RTF; time needed to generate the gold word-by-word transcription of the ATC audio with regard to its duration) for transcribing the data can be reduced from 50 to 20. An overview of the composition of ATCO2 corpora is given in Figure 1.

This paper covers several aspects and lessons learned (see Section 6) related to the data collection and the transcription pipeline, including its primary actors. Also, it covers the main AI-based systems that can be developed with the ATCO2 corpora, and we set baselines on ASR and understanding.

The rest of the paper is organized as follows. Section 2 covers related work on automatic speech recognition and understanding for ASR. We describe the ATCO2 system, data collection pipeline, and the main contributions of this paper in Section 3. In Section 4, we cover technical details about the data collection platform (front end and back end) and how the community of volunteers interacts with them. In Section 5, we cover the main technologies that can be developed with ATCO2 corpora. We conclude the paper and discuss the main lessons learned in Section 6.

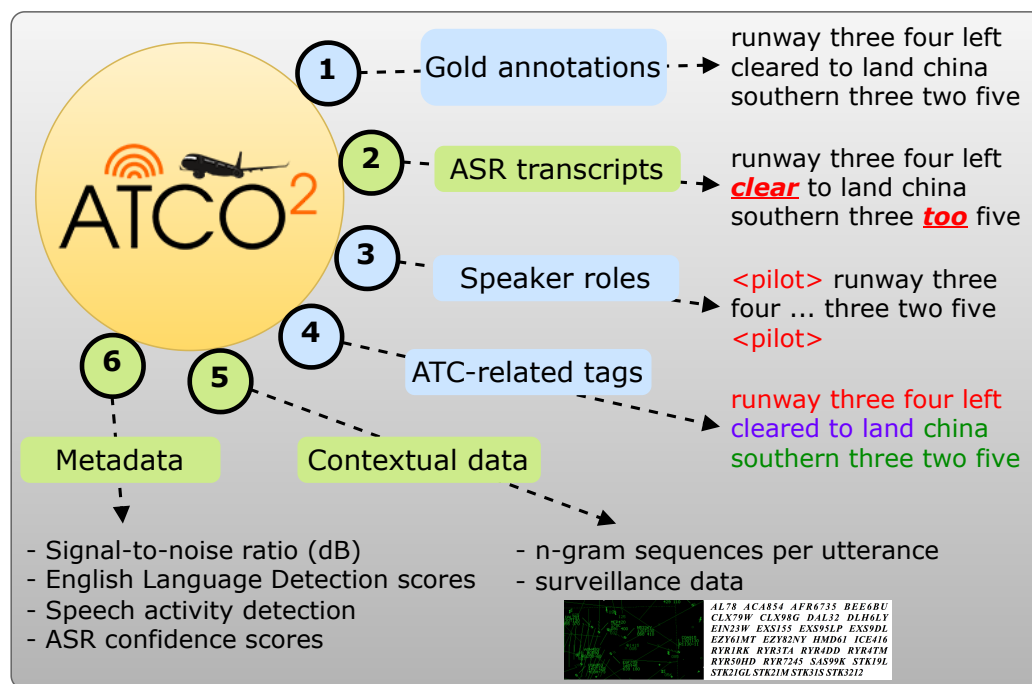


Figure 1. ATCO2 corpora. Blue circles denote transcriptions only available for ATCO2 test set corpus. Green circles denote transcriptions and metadata available for both ATCO2 test set and ATCO2 transcribed corpus sets. Taken from previous work in [7].

2. Early Work on Automatic Speech Recognition and Understanding in ATC

Recent work in ASR and understanding of ATC communications has been documented for trainee's ATCo training by AENA (Aeropuertos Españoles y Navegación Aérea) [10] and MITRE corporation [11]; also including workload estimation with ASR systems [12]. In recent years, more research-oriented work has focused on pure ASR. For example, ref. [13] established the first benchmark on ASR for different ATC communications-focused databases. Furthermore, there has been a significant effort to integrate novel semisupervised learning algorithms for boosting the ASR performance with surveillance data such as [14]. This supports the idea of the growing interest in research in ASR and understanding towards ATC, with mature proof-of-concept engines that can assist ATCos in their daily tasks. Our previous work related to the large-scale automatic collection of ATC audio data from different airports worldwide was in [9]. Additionally, recent work targeted to improve callsign recognition by integrating surveillance data into the pipeline has been explored in [15] or, for instance, automating pilots report extraction with ASR tools [16].

Another line of work has been directed at open-sourcing ATC-related databases: for US-based communications [17], in Czechia [18], and [19] for several accents in English. Recently, there was an Airbus-led challenge [20] for ATC communications, with French-accented recordings from France [21]. Private databases such as VOCALISE [22] and ENAC [23] have also targeted ATC communications. For a general overview of ATC-related databases, we redirect the reader to Table 1 in ref. [7], and for the databases released by the ATCO2 project, to Table 1 in this paper.

Table 1. Air traffic control communications corpora released by ATCO2 project. [†] Full database after silence removal. ^{††} Speaker accents depend on the airport's location and on the airline origin (e.g., Air France in Australia may contain French-accented audio); accents of pilots are not known at any time of the communication due to privacy regulations.

Database	Details	Licensed	Accents	Hours [†]	Ref.
Released corpora by ATCO2 project					
<i>ATCO2 corpora</i>	Data from different airports and countries: public corpora catalog.elra.info/en-us/repository/browse/ELRA-S0484/ (accessed on 10 October 2023)		Several ^{††}		[7]
<i>ATCO2-test-set</i>	Real life data for ASR and NLP research.	✓	Several ^{††}	4	[7]
<i>ATCO2-T set</i>	ASR-based transcribed dataset. Real data for research in ASR and NLU.	✓	Several ^{††}	5281	[7,9]
Free access databases released by ATCO2 project					
<i>ATCO2-test-set-1h</i>	'ASR dataset': public 1 h sample, a subset of <i>ATCO2-test-set</i> . https://www.atco2.org/data (accessed on 10 October 2023)	✓	Several ^{††}	1	[9]
<i>ATCO2-ELD set</i>	'ELD dataset': public dataset for English language detection. https://www.atco2.org/data (accessed on 10 October 2023)	✓	Several ^{††}	26.5	[24]

3. ATCO2 Corpora

It is well known that AI-based tools need large amounts of reliably transcribed data during their training process. For instance, ASR or NLP tools for ATC could work better if we had large-scale data. The ATCO2 corpora was designed to target this data scarcity issue by solving four big challenges:

(1) Current corpora related to air traffic control are primarily focused on automatic speech recognition. However, for an AI engine to be successfully deployed in the control room, it must not only accurately transcribe ATC communication but also understand it. This includes the ability to detect speaker roles (SRD) as well as extract and parse callsigns and commands. The ATCO2 corpora provides a comprehensive solution to this challenge by including detailed tags for SRD and callsign and command extraction. This, in turn, will improve the accuracy and efficiency of AI-based systems in ATC operations.

(2) Out-of-domain ASR and NLP-based corpora transfer poorly to the ATC domain. ATC communication follows a unique grammatical structure and employs a specific set of the vocabulary defined by ICAO [25], making it a niche application. This poses a significant limitation to the use of out-of-domain corpora (previous studies [13] have shown that employing non-ATC related corpora such as LibriSpeech [26], CommonVoice [27] or SWITCHBOARD [28], does not match the acoustics of ATC communication, and therefore does not contribute substantially to ASR training). As such, the ATCO2 project collected and publicly released a large amount of ATC-specific data to aid in the development of ASR and understanding engines for ATC.

(3) The research community working on ATC is hindered by a severe lack of openly available annotated data. To address this issue, the ATCO2 project has released a vast corpus of over 5000 h of automatically transcribed data (i.e., *ATCO2-T set*), as well as 4 h of manually annotated data (i.e., *ATCO2-test-set-4h*). It is worth noting from Table 1, that the transcriptions generated by the automatic tools have been proven to be robust, with WERs as low as 9%. These errors are achieved when training an ASR engine with ATCO2 corpora only. See the prior results for the *MaLorca-Vienna-test* set coming from the MALORCA project in [7].

(4) There is no standardized metric to evaluate quality of nontranscribed data prior to their transcription process. Currently, when a new corpus for ASR is in its collection and labeling phase, few filtering stages are performed to ensure high-quality audio data selection. In contrast, in Section 3.3, and specifically in Equation (1), ATCO2 uncovers the

quality estimation that helped to select the best audio files for gold transcription generation by humans.

3.1. ATCO2 System and Generalities

The ATCO2 system is described in Figure 1. During the collection of the ATCO2 corpora, we followed several preprocessing steps in order to normalize the generated transcriptions. Here, we aim at minimizing errors produced by phonetic dissimilarities, e.g., “descent to two thousand” and “descend two two thousand”. We performed several text normalization steps in order to unify the gold and automatic transcriptions following ICAO rules [25] and well-known ontologies for ATC communications [5]. A summary of the transcription protocol is depicted in Figure 2. Additionally, we direct the reader to a more detailed overview on text normalization and lexicon for transcript generation in Section 3 of ref. [7]. Furthermore, the ATCO2 corpora are composed of *ATCO2-T set* corpus and *ATCO2-test-set* corpus, described below:

- First, the *ATCO2-T set corpus* is the first ever release of a large-scale dataset targeted to ATC communications. We recorded, preprocessed, and automatically transcribed ~5281 h of ATC speech from ten different airports (see Table 2). To the best of the authors’ knowledge, this is the largest and richest dataset in the area of ATC ever created that is accessible for research and commercial use. Further information and details are available in [7].
- Second, *ATCO2-test-set-4h corpus* was built for the evaluation and development of automatic speech recognition and understanding systems for English ATC communications. This dataset was annotated by humans. There are two partitions of the dataset, as stated in Table 1. The *ATCO2-test-set-1h corpus* is a ~1 h long open-sourced corpus, and it can be accessed for free at <https://www.atco2.org/data> (accessed on 10 October 2023). The *ATCO2-test-set-4h corpus* contains *ATCO2-test-set-1h corpus* and adds to it ~3 more hours of manually annotated data. The full corpus is available for purchase through ELDA at <http://catalog.elra.info/en-us/repository/browse/ELRA-S0484> (accessed on 10 October 2023).

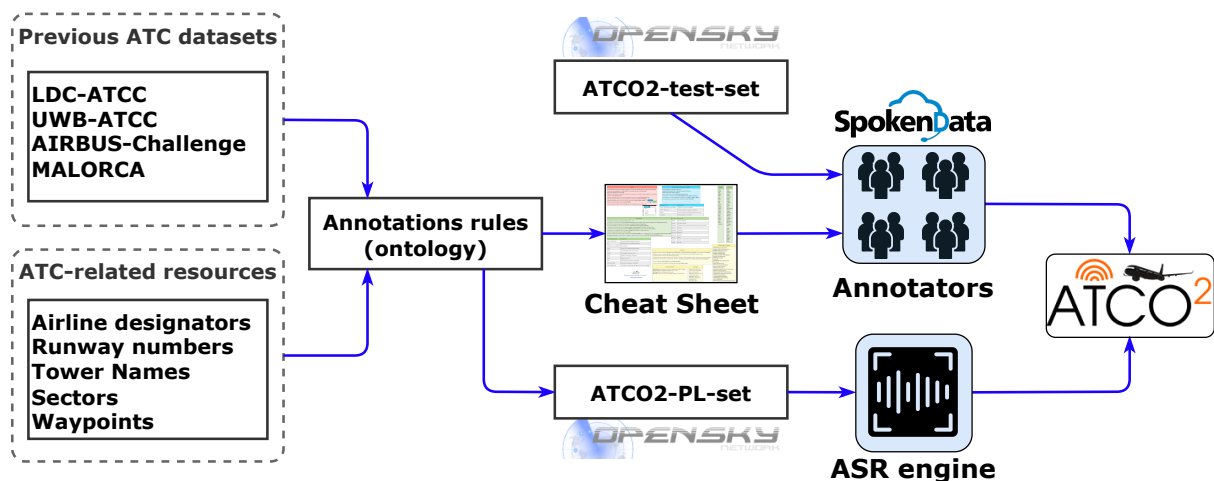


Figure 2. Transcription protocol. ATCO2 corpora follow a rigorous transcription protocol based on previous ATC-related corpora and resources. Additionally, a cheat sheet for ATC transcript generation was developed during the project. The cheat sheet is available in Appendix E.

Table 2. Total accumulated duration. (in hours) of speech after voice activity detection per airport in *ATCO2-T set*. [†] English language detection (ELD) (0–1) score. This score shows how confident our ELD system is in detecting whether there is only English spoken inside the ATC communication. Note that the first word for each name denotes the ICAO airport identifier.

ICAO (airport identifier)—City										
EETN—Tallinn	EPLB—Lublin	LKPR—Prague	LKTB—Brno	LSGS—Sion	LSZB—Bern	LSZH—Zurich	LZIB—Bratislava	YBBN—Brisbane	YSSY—Sydney	others
English Data (language score ≥ 0.5) [†]										
131	<1	1762	888	330	699	921	24	170	77	<1
Non-English Data (language score < 0.5) [†]										
2	<1	187	611	83	55	49	26	10	3	<1

3.2. Data Collection Pipeline

The processing pipeline is implemented as a Python script that follows a configuration file \rightarrow `worker.py`. The configuration file allows us to modify the logic and flow of the data in the pipeline on-the-fly. It allows parallelism, forking, and conditions. In principle, `worker.py` consists of global definitions (constants), blocks (local definitions), and links (an acyclic oriented graph) between blocks. The processing pipeline is given in Figure 3. For instance, we address earlier implementations of each technology from the previous work [9], e.g., segmentation and diarization, ASR, or named entity recognition (NER). All the technologies and tools are encapsulated in BASH scripts with a unified interface.

The first row of blocks from Figure 3 refers to segmentation and demodulation. Initially, an antenna and a recording device jointly capture the radio signal, which is divided into segments containing portions where the transmission was “active”, and the silent parts are not recorded (push-to-talk is used in ATC voice communication). This functionality is part of the RTLSDR-Airband audio recording software, from which we dump the raw I/Q signal. Second, we convert this complex I/Q radio signal into a waveform signal by a software-defined radio CSDR. The first part is performed in the recording device, while the second is performed at the OpenSky Network (OSN). The OSN is a nonprofit community-based receiver network which has been continuously collecting air traffic surveillance data since 2013. Unlike other networks, OpenSky keeps the complete unfiltered raw data and makes them accessible to academic and institutional researchers).

Next, we perform “signal-to-noise ratio (SNR) filtering” (second row); the purpose is to remove the recordings that are too noisy. In bad recording conditions, we can end up in a situation in which the voice is not intelligible. The following step is “diarization” (third row). In the automatically segmented data, some recordings contain more than one speaker. This is a problem because we intend to automatically transcribe speaker turns of single speakers. And, for subsequent NLP/SLU tasks, it is important to separate the speaker turns as well. The diarization solves this by splitting the audio into segments with single speakers and assigning them speaker labels. In the ASR step, we simply convert “speech-to-text”. This is performed by our ASR system that we build with tools from the Kaldi toolkit [29]. The outputs from this step are transcripts, which inevitably contain some errors. To improve the accuracy of the transcripts, we use callsign lists from surveillance as contextual information. The callsign lists come from the air traffic monitoring databases of OpenSky Network. Further details can be found in Section 5.2.2 and [15].

Next, the transcripts are used as input for the English language detection (ELD) system. The purpose is to be able to discard non-English audio data. The typical state-of-the-art language identification system is based on acoustic modeling and uses audio as input. For the ATC speech, we do not need to “identify” the non-English languages, so we developed

a “lexical English detection system” which uses transcripts and confidence scores produced by ASR as its inputs (see previous work at Interspeech in 2021 [24]). For ATC speech, this worked better than the “traditional” acoustic language identification method. The last automatic operation is “post-processing by NLP”. Currently, the pipeline performs a callsign-code extraction step. It returns the callsign in ICAO format, like “DLH77RM”, belonging to an aircraft. Finally, some processed data go through “human correction”, and some data are kept with automatic labels. The former case produced *ATCO2-test-set-4h corpus*, while the latter, *ATCO2-T set corpus*. A more detailed description of the data collection flow and data transcription is given in Appendices C and D.

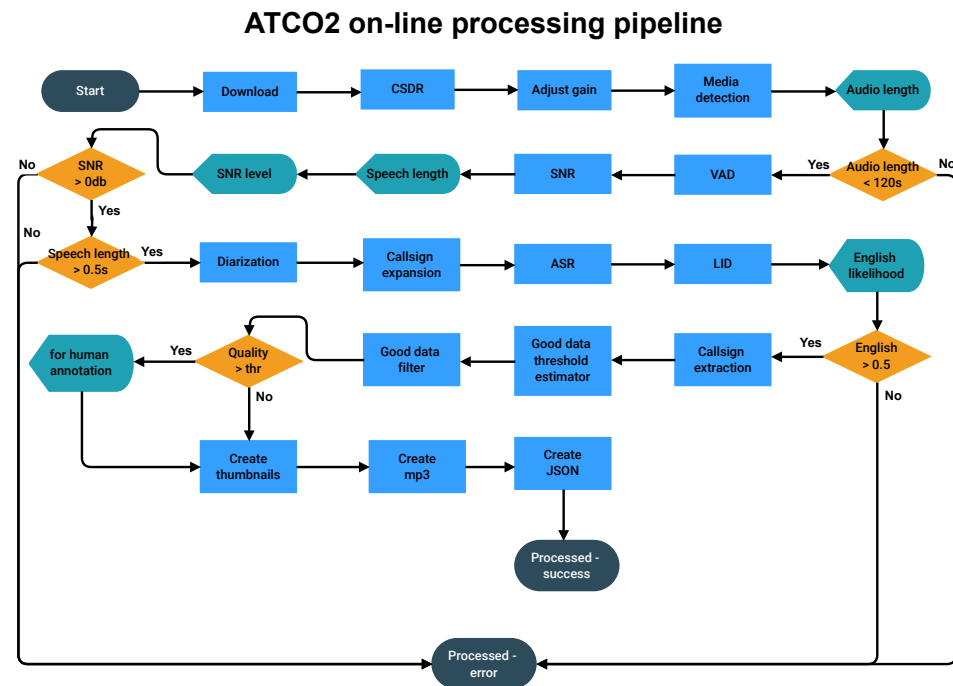


Figure 3. ATCO2 workflow for processing data collected by a community of feeders. Initially, the data are sent and stored on OSN servers. The audio data go through several modules to filter out recordings with a high level of noise and too-long or too-short segments. **Blue rectangles** are processes. The **cyan arrow blocks** are internal callback events, where the pipeline informs the master node about progress and sends intermediate results. The **orange rhombuses** are conditions, where intermediate results are taken into account (e.g., an SNR level), i.e., whether to continue (clean audio) or stop processing. A final internal callback is run when the pipeline finishes. It triggers the API to call the OSN server with the particular callback, for instance, the processing has finalized as OK or ERROR.

3.3. Quality Estimation for Data Transcription

As mentioned at the end of Section 3.2, the captured, processed, and automatically transcribed data (see Figure 3) can be annotated by humans. This in turn would generate “gold transcriptions” that we use to evaluate the proposed ASR and NLP systems. The *ATCO2-test-set-4h corpus* went through all these steps. As the data are continuously being recorded by OSN, we need to select the most intelligible and clean data. We developed a score that ranks the recordings depending on their quality. This score integrates seven metrics that assess the quality of each recording present in *ATCO2 corpora*. For instance, we used Equation (1) to measure, rank, and select the ATC communications with the highest quality. Later, these recordings were shortlisted for human transcription (see Section 4.2). The data annotators generated the ground truth transcripts and tags that are part of the *ATCO2-test-set-4h corpus*. The ranking score is given as follows:

$$Score = \log(avg_{SNR} + e) + \log(num_{spk} + e) + \log\left(\frac{speech_{len}}{audio_{len}} + e\right) + ELD_{score} \times 3 + avg_{WordConf} \times 3 + \log(wrd_{cnt} + e), \quad (1)$$

where

- avg_{SNR} —provides average SNR of speech in range $\langle 0, 40 \rangle$. SNR needs to be as high as possible;
- num_{spk} —provides the number of speakers in the audio in the range of $\langle 1, 10 \rangle$. The more speakers detected in audio, the better;
- $speech_{len}$ —provides the amount of speech in seconds;
- $audio_{len}$ —provides the overall audio length. More speech detected in audio is better;
- ELD_{score} provides “probability” of audio being English in the range $\langle 0.0, 1.0 \rangle$. The higher the ELD score, the better;
- $avg_{WordConf}$ —provides average confidence of the speech recognizer $\langle 0.0, 1.0 \rangle$. We want data where the recognizer is confident. Higher is better;
- wrd_{cnt} —provides the number of words spoken in the range of $\langle 0, \sim 150 \rangle$. The more words, the better.

A breakdown of the outputs of these steps for a single day is given in Figure 4. For instance, ~ 0.6 h of data are selected for gold transcriptions from an initial 26 h pool of audio data. We believe this is a robust quality scoring method because it gathers information from different systems, e.g., ASR, SNR, and ELD estimation. A day-to-day estimation of the output of each of these steps is available on the SpokenData website: <https://www.spokendata.com/atc> (accessed on 10 October 2023).

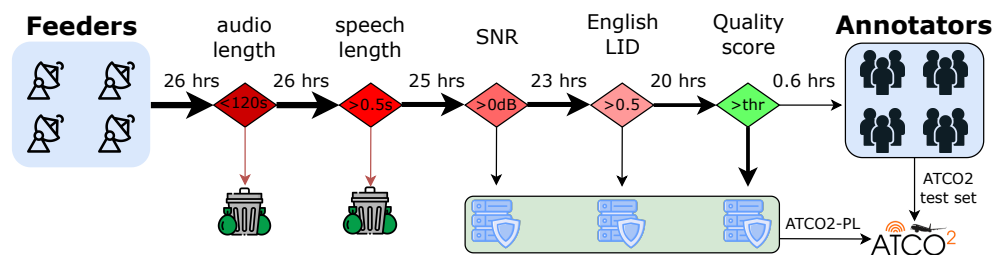


Figure 4. Breakdown of data flow yield from raw data (recordings from data feeders) w.r.t the human-annotated transcripts throughout the pipeline. This is a one-day snapshot from 9 February 2022.

3.4. Runtime Characteristics

We also measured the running time for individual components of our processing pipeline. In Table 3, we list the relative time spent by each module, such as ASR and speaker diarization; both accounting for 65% of the overall processing time. This disparity compared to other modules is due to the fact that both are AI-powered modules, which, in principle, needs more processing time. Other important parts are preprocessing, voice activity detection (VAD) segmentation, and ELD. Audio data preprocessing involves obtaining data, demodulation by software radio, segmental gain control, detecting media format, and plotting waveform. A key metric is the real-time factor of the whole pipeline. The real-time factor is the ratio of “processing time” over “length of the audio”. Our processing pipeline has a real-time factor of 4.47. In other words, the processing is computationally demanding. For an average five-second-long recording, the processing time is 22 s. The actual running times of each component for the “average” five-second-long recording are shown in Table 3.

Table 3. Processing time per component in the transcription pipeline. The values in the second column are for an average 5.016 s long recording. The average was computed over 10,334 recordings (14.4 h), recorded on 4 December 2021.

Processing Step	Time [s]	Percentage [%]
Preprocessing	2.5	11.6
VAD segmentation	2.4	11.1
SNR estimation	0.6	3.0
Diarization	7.1	32.6
Callsign expansion	0.5	2.1
Speech-to-text (ASR)	7.0	32.1
English detection	1.3	5.9
Callsign extraction	0.1	0.4
Post-processing	0.2	1.2
Total time	21.6	100.0

4. Collection Platform and Community of Volunteers

In this section, we summarize the data collection and distribution. In addition, a short description of the roles involved in data processing is provided. We also cover some high-level statistics about the collected data. First, data are captured and fed into the OpenSky Network by the volunteers who operate their own receiver equipment (see Figure 5). These individuals are often aviation enthusiasts with previous operational experience, or people with an interest in aviation technology, e.g., conducting domain-related research. But anyone with little to no background in aviation or technology can become a feeder. To become a feeder, one must have an internet connection and access to a VHF receiver. An affordable low-complexity setup is covered in the ATCO2 corpora paper [7] and the guide for setting it up is provided <https://ui.atc.opensky-network.org/set-up> (accessed on 10 October 2023). It is important to recall that in some countries, it is prohibited by law to record air traffic management (ATM) data. Readers interested in the legal aspect are directed to the legal and privacy aspects for collection of ATC recordings section in [7].

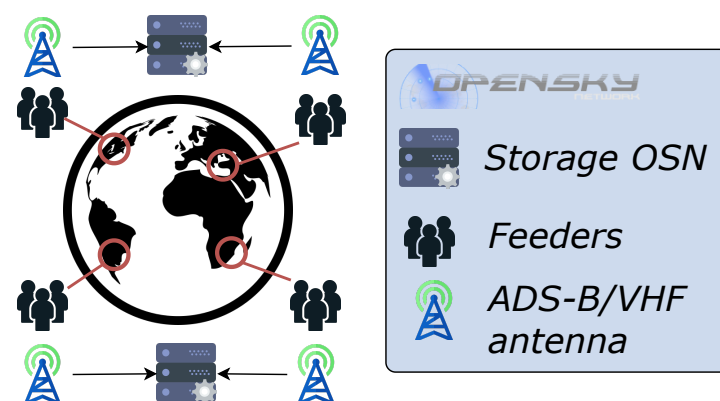


Figure 5. Data feeders pipeline. The data users have set up a VHF receiver and feed data to OSN servers.

4.1. The Platform

The high-level architecture is given in Figure 6. As one can observe, the platform has been divided into three distinct groups: (i) feeder equipment, (ii) back-end, and (iii) front-end. The architecture was decided during the design phase of ATCO2, with the main objective to achieve scalability of the entire system. That means keeping the complexity relatively low within all the groups, which allows it to

- Support a similar number of users to the current OpenSky Automatic Dependent Surveillance–Broadcast system (ADS-B);
- Keep the feeder equipment simple and affordable;
- Provide data to different types of users in a simple and intuitive way;

- Interface external services (e.g., voice annotation) in a simple and intuitive way;
- Keep maintenance and error handling as simple as possible.

A better overview of the OSN platform is also listed in Appendix B. As mentioned above, the platform has been divided into three parts. Below, we describe each of these platform's groups.

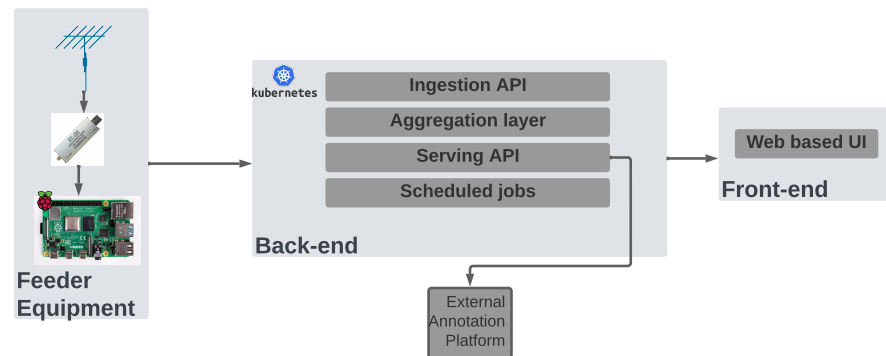


Figure 6. The high-level architecture of the data collection platform.

Feeder equipment: the main task of the feeder equipment is to capture the conversation between the pilot and the ATCo and feed the data, together with some relevant metadata, to the back-end. For the recording part, we recommend using RTLSDR-Airband together with RTLSDR dongle. RTLSDR is a set of tools that enables USB dongles based on the Realtek RTL2832U chipset to be used as cheap software defined radios, given that the chip allows transferring raw I/Q samples from the tuner straight to the host device (see further documentation in <https://osmocom.org/projects/rtl-sdr/wiki/Rtl-sdr>; accessed on 10 October 2023). The latter is an affordable and widely used combination within the aviation enthusiast community for this exact purpose—to capture and stream ATC voice.

The feeder software is responsible for transmitting the recordings from the receiver to the remote server. It is a rather simple piece of software that monitors the output directory of the RTLSDR-Airband and transfers any new data it finds to the back-end using a gRPC (gRPC; remote procedure calls) connection. The fact that the feeder software only looks for specific types of data from the output folder suggests that the feeder is free to choose any other software for capturing and storing the voice data. Care must be taken to assure that the output is suitable for the feeder software. A simple, step-by-step guide is provided to simplify the setup process. It can be found at <https://ui.atc.opensky-network.org/set-up> (accessed on 10 October 2023).

Back-end: the main tasks for the back-end are (i) to store recordings, transcripts, and any other relevant metadata, and (ii) to provide interfaces for external users. The external users in this are data feeders, transcription service providers, data users, or any other parties contributing to the dataset or making use of it. The back-end is deployed on Kubernetes, an open-source container orchestration system. As one can observe from Figure 6, there are several processing layers involved. These layers are as follows:

- Ingestion API: receives recording segments and metadata and queues them for processing in Kafka/S3 compatible object storage;
- Aggregation layer: converts raw data to flac audio, stores metadata, and triggers transcription using Kafka Streams, S3, and Serving API;
- Serving API: provides external interfaces to consume metadata, store, and consume transcript and statistics;
- Scheduled jobs: run processes that are not part of the streaming process like statistics aggregation and data housekeeping.

Interfacing the back-end is performed using API, which is well documented in <https://api.atc.opensky-network.org/q/swagger-ui> (accessed on 10 October 2023). In order to access the back-end and make use of the available APIs, one needs to register on <https://auth>.

opensky-network.org/auth/ (accessed on 10 October 2023), contact OpenSky Network (mailto: contact@opensky-network.org), and give a short description of what one needs the access for.

Front-end: the front-end is a web-page (<https://ui.atc.opensky-network.org/>; accessed on 10 October 2023) and it provides access to public stats, links to documentation, e.g., API documentation, and external web pages, e.g., SpokenData transcription service. In addition, this is a place for an user to set up their receivers, see some statistics about the receiver performance, and so on.

Statistics: since the public opening of the service (5 March 2023), the ATCO2 project has recorded speech from 24 different airports in 14 different countries. In Figures 7 and 8, names of countries and airports, together with corresponding recording lengths, are shown. Please note that only the airports/areas with the length of regrinding ≥ 1 h are included. This also applies for the ATCO2 corpora released in ELDA.

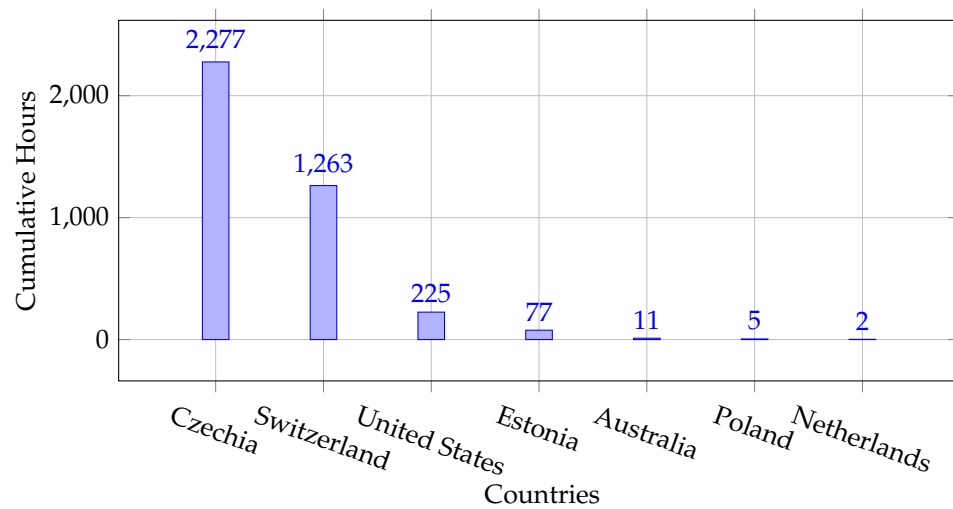


Figure 7. Length of recordings per country from the beginning of the service until 5 March 2023. Countries where the length of recordings is longer than 1 h are given. Note that some countries (e.g., United States) were not part of the official release of the ATCO2 corpora (see Table 2). Still, they are currently being collected in the OSN Platform.

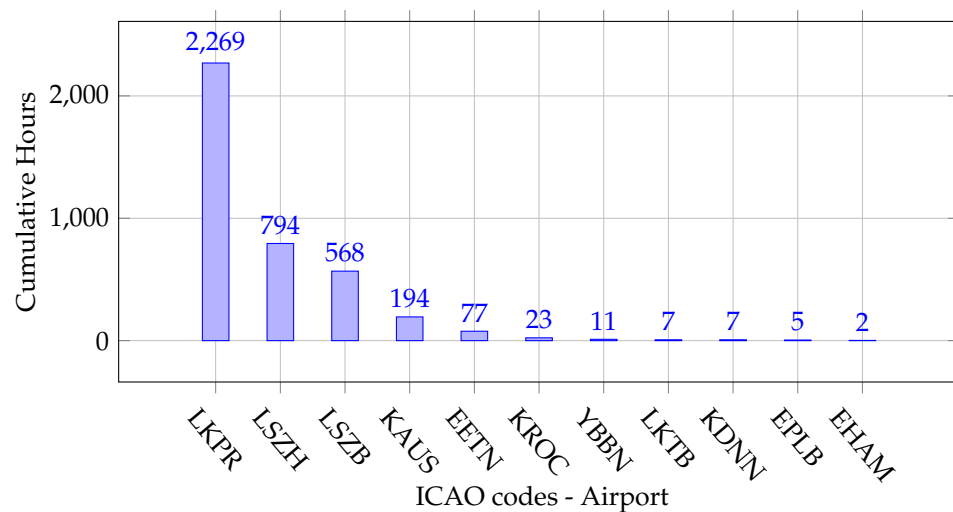


Figure 8. Length of recordings per airport from the beginning of the service until 5 March 2023. Airports where the length of recordings is longer than 1 h are given. LKPR: Vaclav Havel Airport Prague; LZSH: Zurich Airport; LSZB: Bern Airport; KAUS: Austin-Bergstrom International Airport; KROC: Frederick Douglass Greater Rochester International Airport; YBBN: Brisbane Airport; LKTB: Brno Airport; KDNN: Dalton Municipal Airport; EPLB: Lublin Airport; EHAM: Amsterdam Airport Schiphol.

4.2. Data Annotators

Apart from the data feeder, there is another type of volunteers who have contributed to the project and will continue to contribute in the future. These are called “Annotators”. The data annotators are volunteers who write down the transcripts of the ATC voice communications, including assigning speakers and annotating named entities, i.e., callsigns and commands. For the ATCO2 project, we relied on both volunteers and paid transcribers. Our data processing pipeline (as seen in Figure 3) generates transcripts and NLP tags for each communication. By generating transcriptions with AI tools, we are able to speed up the overall transcription process (if you are interested in becoming an annotator, please create an account on the SpokenData transcription platform: <http://www.spokendata.com/atco2>; accessed on 10 October 2023). The amount of human transcribed data is the package of a four-hour test set, i.e., *ATCO2-test-set-4h corpus*. The data annotators are the final actors involved in the transcription step, as shown in Figure 4.

5. Technologies

In this section, we cover the main tools developed with the ATCO2 corpora. We also list some potential topics that can be explored with it. Moreover, note that the ATCO2 corpora are not limited to the fields covered in this paper e.g., ASR or NLP, but also can be used for text-to-speech (TTS), which is somehow opposite to ASR. We expect the community will build on top of ATCO to foster and advance speech and text-based technologies for ATC.

5.1. Automatic Speech Recognition

One of the principal components of the ATCO2 project is the strong ASR system, used in order to provide high-quality automatic transcriptions for the collected ATC data. An ASR system is trained to predict the best text translation for the input acoustic signal. Formally speaking, ASR aims to find the best probability candidate output sequence of words from a set of all possible word combinations (or sentences) in a language given a noisy acoustic observation sequence. End-to-end ASR models learn a direct mapping of speech S , to the output text W :

$$\hat{W} = \operatorname{argmax}_{W \in \mathcal{V}^*} p(W|S).$$

The hybrid (conventional) ASR systems combine three separately trained models: acoustic model (AM), pronunciation model, and language model (LM). The model calculates the conditional probability $p(W|S)$, where W is a sequence of words ($W = w_1, \dots, w_n$), S is a sequence of input feature vectors representing the acoustic observations ($S = s_1, \dots, s_t$), and \mathcal{V} is the vocabulary of all possible words [30,31] or subwords [32], as shown in Equation (4).

$$\hat{W} = \operatorname{argmax}_{W \in \mathcal{V}^*} p(W|S) \tag{2}$$

$$= \operatorname{argmax}_{W \in \mathcal{V}^*} p(S|W)p(W) \tag{3}$$

$$= \operatorname{argmax}_{W \in \mathcal{V}^*} \sum_P p(S|P)p(P|W)p(W), \tag{4}$$

where $p(S|P)$ is an AM, $p(P|W)$ is a pronunciation model, and $p(W)$ is an LM; we use \mathcal{V}^* to represent the collection of all word sequences formed by words in \mathcal{V} . One of the advantages of conventional pipeline models is a more transparent optimization of an objective function [33]. Moreover, the LM is trained with unpaired text data and can be easily adapted to a specific domain. This gives conventional models more flexibility and makes them convenient for use in industrial projects, such as ATC.

5.1.1. Training Data Configuration

To measure the effectiveness of using automatically transcribed data (ATCO2-T set) versus using fully supervised gold transcriptions, we defined three training scenarios.

- **Scenario (a) only supervised data:** we employ a mix of public and private supervised ATC databases (recordings with gold transcriptions). It comprises ~190 h of audio data (or 573 h after speed perturbation);
- **Scenario (b) only ATCO2-T 500 h dataset:** we use only a subset of 500 h from the ATCO2-T corpus (see introductory paper [7]);
- **Scenario (c) only ATCO2-T 2500 h dataset:** same as **scenario (b)**, but instead of only using 500 h subset, we use five times more, i.e., a 2500 h subset. This subset is only used to train a hybrid-based ASR model (CNN-TDNNF; convolutional neural network and time-delay neural network) to test the boosting experiments in Section 5.2.2.

5.1.2. Test Data Configuration

Two ATCO2 test sets are used for ASR evaluation, as shown in Table 4: ATCO2-test-set-1h (in short ATCO2-1h) and ATCO2-test-set-4h (in short ATCO2-4h). The same test sets are used for boosting experiments presented in Section 5.2.2.

Table 4. WER results for the public ATCO2 test sets with CNN-TDNNF models trained on different data; this includes from scenario (a) to scenario (c).

Model	Test Sets	
	ATCO2-1h	ATCO2-4h
Scenario (a) only supervised 573 h dataset	24.5	32.5
Scenario (b) only ATCO2-T 500 h dataset	18.1	25.1
Scenario (c) only ATCO2-T 2500 h dataset	17.9	24.9

5.2. Conventional ASR

To obtain automatic transcriptions of the best possible quality for ATCO2 corpora audio, we use a strong hybrid model trained on ATC data only. We train a hybrid-based model for each of the scenarios described above. For **scenario (a)**, an AM was built to include all available 190 h datasets, speech augmentation accounting for 573 h of data. The model dictionary consists of 30,832 words coming from diverse sources. This includes (i) a list of airline designators for callsigns taken from Wikipedia: https://en.wikipedia.org/wiki/List_of_airline_codes (accessed on 10 October 2023); (ii) all five-letter waypoint names in Europe retrieved from the Traffic project, see <https://pypi.org/project/traffic/> (accessed on 10 October 2023); (iii) additional words, such as countries, cities, airport names, airplane models and brands, and some ATC acronyms. For training the acoustic model, we use the Kaldi toolkit [29]. The system follows the standard Kaldi recipe, which uses MFCC and i-vectors features [34] with time-delay neural networks (TDNN) [35,36]. The standard chain training is based on lattice-free maximum mutual information (LF-MMI [37], which includes threefold speed perturbation and one-third frame subsampling). The acoustic model is a CNN-TDNNF [38], which comprises a convolutional network and a factorized-TDNN. The LM is 3G trained on the same data as the acoustic model with additional text data coming from additional public resources such as airlines names, airports, ICAO alphabet, and way-points in Europe.

Results and analysis: the results are presented in Table 4. We compared three models trained with the same conventional CNN-TDNNF architecture but on different data: scenarios (a), (b), and (c) (see Section 5.1.1). The model (a) in Table 4 is trained on the “out-of-domain” for ATCO2 but supervised data. The models (b) and (c) are trained on the “in-domain” ATCO2 data and the difference is only in the size of the training set: 500 h vs. 2500 h. We can see that training on completely unsupervised data yields good performance in comparison to (a). Increasing the size of unsupervised data from 500 h to 2500 h, however, does not bring too much improvement: the WER goes from 18.1% to 17.9% and from 25.1% to 24.9% only for ATCO2-1h and ATCO2-4h, respectively.

Our main hypothesis is that ATCO2 test sets contain higher levels of noise compared to the audio data present in (a), i.e., mainly clean data from ATCos. Moreover, ATCO2 test sets also contain speech from pilots collected via VHF receivers, which in turn degrades the

SNR levels, i.e., reduced audio quality. Hence, when the system is trained on “clean data”, i.e., scenario (a) and later tested on ATCO2, it creates a large train–test set mismatch. Yet, when we use ATCO2 training data, scenario (b) or scenario (c), this mismatch is reduced, and therefore we obtain substantially better results.

5.2.1. End-to-End ASR

Differently from hybrid-based ASR, there exists another paradigm for performing ASR [39], named end-to-end (E2E) ASR [40]. Here, we aim at directly transcribing speech to text without requiring alignments between input features and output words or characters (i.e., standard procedure in hybrid-based ASR); see Equation (4). Recent work on encoder–decoder ASR has shown that this step can be removed [41]. E2E can be divided into connectionist temporal classification (CTC)-based [42], attention-based encoder–decoder modeling [43], or hybrid [44]. Previous work based on self-supervised learning [45] for ASR includes Wav2Vec2.0 [46], vq-Wav2Vec [47], and, most recently, WavLM [48] and multilingual XLS-R [49] models. E2E ASR aims at reducing the expert knowledge needed. This makes the overall ASR development simpler; thus, it could have a significant impact on ATC [50]. This work focuses on data novelty (including their collection and preparation) rather than investigating (i) different E2E architectures for ASR, e.g., Conformer [51], HyperConformer [52], Conner [53], or BranchFormer [54]; or (ii) toolkits for E2E ASR such as SpeechBrain [55], ESPnet [56], NeMo [57], or WeNet toolkits [58]. Therefore, we leave these lines of research for future work.

5.2.2. Callsign Boosting

To further improve the prediction made by an ASR system, along with speech input, one can use other information available from context. For the ATC domain, such context information may be the data received from radar. Every moment, radar registers aircraft that are currently in the airspace, listing unique identifiers of those aircraft, i.e., “callsigns”. With the radar data, we know exactly what callsigns are especially likely to appear in the conversation. This knowledge allows us to bias the system outputs towards these registered callsigns and to increase the probability that they are recognized correctly. A callsign is typically a sequence of an ICAO airline identifier, letters, and digits, which in speech turns into a sequence of words. In ASR, the target sequences of words can be boosted during decoding with WFST (weighted finite state transducer) by adjusting the weights in the prediction graphs, called “lattices”. The rescoring technique with WFST was proposed earlier and applied for biasing towards use’s play lists [59], contact names [60], and named entities [61]. Recently, a similar biasing approach has proved to be useful in improving callsign recognition [9]. The rescoring of lattices is performed with the finite state transducer (FST) operation of composition between lattices produced by an ASR system and an FST created with the target transcript and discount weights (Equation (5)):

$$\text{biased_Lattices} = \text{Lattices} \circ \text{biasing_FST} \quad (5)$$

Biasing the lattice toward the context callsigns usually allows us to considerably improve their recognition in the final outputs (Table 5). The results of different experiments on the ATC data proved that applying the lattice rescoring method on top of ASR predictions leads to higher accuracy of automatic transcriptions, first of all, callsigns [14]. Therefore, lattice rescoring was used for all transcriptions of the ATCO2 data.

Table 5. Results for boosting experiment on ATCO2 corpora. Results are listed for the CNN-TDNNF model trained with either all supervised data or 500 h or 2500 h of ATCO2 corpora. The top results per block are **highlighted in bold**. The best result per column is marked with underline. [†] 1h public test set. [‡] 4h full test set. Results are obtained with offline CPU decoding. [¶] word error rates only on the sequence of words that compose the callsign in the utterance. CallWER: callsign word error rate; ACC: accuracy.

Boosting	ATCO2-test-set-1h [†]			ATCO2-test-set-4h [‡]		
	WER	CallWER [¶]	ACC	WER	CallWER [¶]	ACC
scenario (a) only supervised dataset						
Baseline	24.5	26.9	61.3	32.5	36.7	42.4
Unigrams	24.4	25.5	63.2	33.1	35.0	45.8
N-grams	23.8	23.8	66.4	31.3	33.7	47.9
GT boosted	22.9	19.1	75.2	29.7	29.1	58.5
scenario (b) only ATCO2-T 500 h dataset						
Baseline	18.1	16.2	71.2	25.1	24.8	62.6
Unigrams	19.1	14.6	74.2	26.0	22.8	65.6
N-grams	17.5	13.5	75.3	24.3	21.4	66.6
GT boosted	16.3	6.9	88.9	22.5	13.0	82.9
scenario (c) only ATCO2-T 2500 h dataset						
Baseline	17.9	16.7	70.5	24.9	24.2	62.0
Unigrams	18.3	14.4	73.8	25.6	22.0	65.9
N-grams	17.3	14.2	74.3	24.3	21.1	66.5
GT boosted	15.9	6.5	89.4	22.2	12.5	83.9

Results and analysis: in Table 5, we report the results for the out-of-domain (ATC supervised) and in-domain (ATCO2-500 h/2500 h) ATC models. Both acoustic models are trained with CNN-TDNNF architecture following the standard Kaldi recipe, as described in Section 5.2. The results are reported with three metrics: WER (word error rate), CallWER (WER calculated on the sequence of n-grams that correspond to callsigns only), and ACC (accuracy).

To rescore a decoding lattice according to the current context, we perform the following steps: (1) we receive all the callsigns registered by the radar at the current timestamp in the ICAO format; (2) we expand the ICAO callsigns to word sequences to include all possible callsign variations, i.e., ways this callsign can be spoken; (3) we use the expanded callsigns to bias the decoding lattice towards the current context. See our previous work [15] for more details on callsign verbalization.

Biasing multiple callsigns registered by the radar, compared to biasing only a ground truth (GT) callsign, can be used in a real-life scenario and with real-time ASR. To allow it, a new contextual FST with expanded callsigns is generated on the fly every time when new data come from radar. The results of biasing a GT callsign are given in Table 5 to illustrate the oracle performance of the biasing method. Overall, decoding with n-grams biasing always helps to achieve better performance, especially for callsigns, with a relative improvement of 15.0% and 12.8% for callsign recognition and of 3.4% and 2.4% for the entire utterance on ATCO2-test-set-1h and ATCO2-test-set-4h test sets, respectively.

The size of biasing FST depends on the number of callsigns and their variations we want to boost. Too many callsigns may decrease the effectiveness of the biasing method, as the more nontrue callsigns are boosted, the less the correct sequence is prominent. The previous results show that the optimal size of biasing FST highly depends on the data, but generally, the performance begins to degrade when the number of biased word sequences exceeds 1000 [62]. For our experiments, we have, on average, 214 biased callsigns variations per utterance in the ATCO2-test-set-4h and 140 biased callsigns variations per utterance in the ATCO2-test-set-1h corpus.

5.3. Natural Language Understanding of Air Traffic Control Communications

Natural language understanding (NLU) is a subfield of NLP that focuses on the ability to understand and interpret human language. NLU involves the development of algorithms and models that can extract meaning and intent from text and/or spoken communication. NLU involves several subtasks, including (i) named entity recognition [63], which aims at identifying entities in text, such as people, places, and organizations [64]; (ii) part-of-speech tagging (POS), identifying the grammatical role of each word in a sentence [65], similar to sequence classification (see Section 5.3.2); (iii) sentiment analysis, identifying the emotional tone of a piece of text [66]; (iv) relationship extraction, identifying the relationships between entities in text [67]; (v) question answering, understanding, and answering natural language questions [68]. The following subsections cover each of the proposed NLU submodules that can be developed with ATCO2 corpora, like the ones presented in Figure 9.

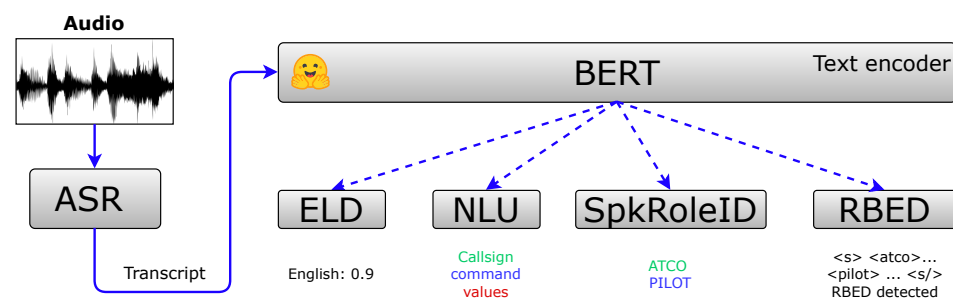


Figure 9. Main automatic speech recognition and understanding tasks that can be achieved with the ATCO2 corpora. ELD: English language detection; NLU: natural language understanding, e.g., callsign highlighting; SPKRoleID: speaker role identification; RBED: read-back error detection.

5.3.1. Named Entity Recognition for Air Traffic Control Communications

In ATC communications, NLU can be used to automatically analyze and interpret the meaning of spoken messages between pilots and ATCOs, which can aid ATCOs in downstream tasks, such as assisting in identifying emergency situations and other critical events. NLU can help to extract important information, such as flight numbers, callsigns, or airport codes, which in turn can aid ATCOs to manage traffic more efficiently.

Overall, the use of NLU in ATC helps improve communication accuracy and efficiency, aids in reduction of ATCOs' workload by prefilling aircraft radar labels, and provides valuable data for analysis and decision making. In this work, one of the main tasks is to understand and extract high-level information within ATC communication. Therefore, we develop an NER system tasked to extract this information, as depicted in Figure 10a. For instance, consider the following transcribed communication (taken from Figure 1):

ASR transcript: runway three four left cleared to land china southern three two five, would be converted to high-level entity format with the NER system to:

Output: <value> runway three four left </value> <command> cleared to land </command> <callsign> china southern three two five </callsign> .

In this work, we developed two systems based on transformers [69] to extract and tag this information from ATC communications, i.e., a pretrained BERT [70] model and RoBERTa [71] model.

Experimental setup: we fine-tune a pretrained BERT and RoBERTa model on the NER task, as shown in Figure 9). We employed the pretrained version of BERT-base-uncased [70] with 110M parameters, URL: <https://huggingface.co/bert-base-uncased> (accessed on 10 October 2023). Also, the pretrained version of RoBERTa-base [71] is composed of 123M parameters, URL: <https://huggingface.co/roberta-base> (accessed on 10 October 2023). We download the pretrained models from HuggingFace [72,73]. For training, we use the full ATCO2-test-set-4h, which contains ~3k sentences. In this dataset, each word is annotated together with a predefined class, as follows: callsign, command, values, and

UNK (everything else). In order to fine-tune the model, we append a layer on top of the BERT model by using a feedforward network with a dimension of 8 (we define two outputs per class, see the class structures in Section 3.3 of ref. [8] and in [15]). Due to the lack of gold transcriptions, we perform a fivefold cross-validation scheme to avoid overfitting. The reader interested in developing their own NER system for ATC is redirected to the open-source GitHub repository of the ATCO2 corpora (GitHub repository: <https://github.com/idiap/atco2-corpora>; accessed on 10 October 2023). We fine-tune each model on an NVIDIA GeForce RTX 3090 for ~ 10 k steps. During experimentation, we use a linear learning rate scheduler with an initial learning rate of $\gamma = 5 \times 10^{-5}$, dropout [74] of $dp = 0.1$, and GELU (Gaussian error linear unit) activation function [75]. We also employ gradient norm clipping [76] for regularization and AdamW as optimizer [77]. Each model during the cross-validation scheme uses an effective batch size of 32.

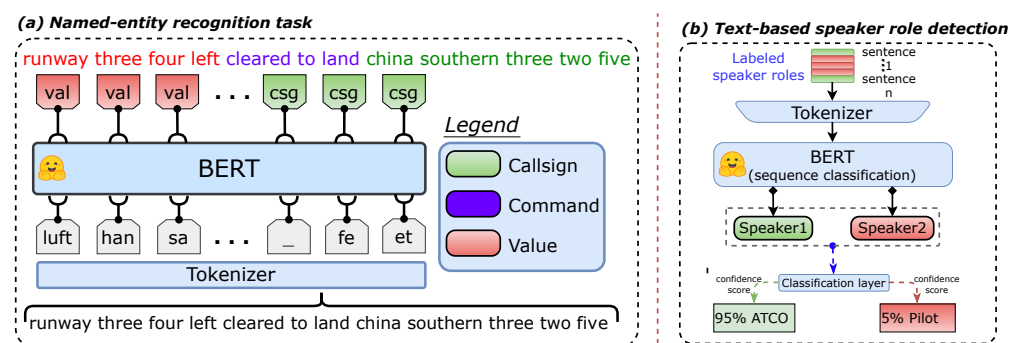


Figure 10. (a) Named entity recognition and (b) speaker role detection module based on sequence classification (SC). Both systems are based on fine-tuning a pretrained BERT [70] model on ATC data. The NER systems recognize callsign, command, and values, while the SC assigns a speaker role to the input sequence. Taken from [7].

Evaluation metric: we evaluate both BERT and RoBERTa NER systems with a binary classification metric named, F-score. Particularly, the F1-score, defined in Equation (8), represents the harmonic mean of precision and recall. Recall, as defined in Equation (7), is the ratio of TP to all samples that should have been identified as positive (including false negatives (FN)). Precision, as described in Equation (6), is the ratio of true positive (TP) results to all positive results (including false positives (FP)):

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \quad (8)$$

Results and analysis: the NER system's performance is evaluated on the ATCO2-test-set-4h corpus using a fivefold cross-validation scheme, with five fine-tuning runs using different training seeds. Table 6 presents the performance metrics for callsign, command, and values classes of two transformer-based [69] models, namely, BERT-base and RoBERTa-base. Although both models achieve similar F1-scores, we provide analysis for the BERT-based NER system, which achieves an F1-score of over 97% for the callsign class, while the command and values classes lag behind with F1-scores of 81.9% and 87.1%, respectively. We hypothesize that the command class contains higher complexity when compared to the other two classes, values and callsigns. Values are mostly composed of defined keywords (e.g., flight level) followed by cardinal numbers (e.g., "one hundred"), while callsigns follow a well-defined structure of airline designators and a set of numbers or letters spoken

in ICAO format [25]. These characteristics make it easier for the NER system to correctly detect them.

One potential method for increasing the performance of the NER system for the command and values classes is to incorporate plausible commands and values in real time, depending on the situation of the surveillance data. This can be achieved using the boosting technique, as described in Section 5.2.2. Although the results with boosting callsigns are reported in Table 5, further investigation is needed to assess the impact of boosting on the command and values classes.

Table 6. Different performance metrics for callsign, command, and values classes of the NER system. Results are averaged over a fivefold cross-validation scheme on *ATCO2-test-set-4h corpus* in order to mitigate overfitting. We run five-times fine-tuning with different training seeds (2222/3333/4444/5555/6666). Results are reported on two transformer-based models. @P, @R, and @F1 refer to precision, recall, and F1-score, respectively.

Model	Callsign			Command			Values		
	@P	@R	@F1	@P	@R	@F1	@P	@R	@F1
Bert-base	97.1	97.8	97.5	80.4	83.6	82.0	86.3	88.1	87.2
RoBERTa-base	97.1	97.7	97.5	80.2	83.7	81.9	85.6	88.6	87.1

5.3.2. Text-Based Speaker Role Detection

Sequence classification (SC) is a type of machine learning (ML) task that involves assigning a label or a category to a sequence of data points [78,79]. The data points in the sequence can be of various types, such as text, audio, or numerical data, and the label assigned to the sequence can also be of different types, such as binary (e.g., positive or negative sentiment [66]) or multiclass. Sequence classification can also be used to automatically classify ATC communication sequences into various categories. This technique can be applied to both audio and text data, making it a versatile tool to provide a high-level understanding of the communication at hand.

In scenarios where only a monaural communication channel exists, it can be challenging to recognize the identity of the speaker. Hence, it is especially important to distinguish between the ATCo and the pilot over the target communications. As a potential solution, we propose an alternative approach that utilizes a speaker role detection (SRD) system based on SC. The system receives text as an input, and it returns as output a category where the communication falls, either uttered by the ATCo or the pilot. In recent years, there has been a growing interest in using deep learning techniques, such as the transformer-based models [69], to improve the performance of SC for SRD in ATC communications. Here, we ablate three types of such models, (i) BERT [70], (ii) RoBERTa [71], and (iii) DEBERTA [80]. These models have been shown to achieve state-of-the-art performance on a wide range of sequence classification tasks, including SRD for ATC. The proposed SRD is illustrated in Figure 10b.

Overall, the SRD and speaker diarization (see Section 5.3.3) tasks can leverage the fact that ATC dialogues follow a well-defined lexicon and dictionary with simple grammar. This standard phraseology has been defined by the ICAO [25] for ATCos. The main idea is to guarantee safety and reduce miscommunications between the ATCos and pilots. Therefore, previous work has shown the potential in performing SRD in an E2E manner on the text-level, as presented here (see in [8,81]).

Experimental setup: The SRD system is built on top of pretrained models (BERT [70], RoBERTa [71], and DEBERTA [80]), which are downloaded from HuggingFace [72,73]. Here, the experimental setup is exactly the same as the one described for the NER system, including the training hyperparameters. For further details, we redirect the reader to Section 5.3.1. Still, the SRD model is fine-tuned on the SC rather than on the NER task. Further, we define an output layer with two units (classes): one for ATCo and one for pilot.

Results and analysis: we evaluate the SRD system on ATCO2-test-set-4h corpus. Differently from the NER system, here, we have access to two training corpora. (1) The Air Traffic Control Corpus (LDC-ATCC) corpus, see URL: <https://catalog.ldc.upenn.edu/LDC94S14A> (accessed on 10 October 2023). It consists of audio recordings in the area of ASR for air traffic control communications. We use the metadata along the transcripts to perform research on NLU for ATC, i.e., speaker role detection. The data files are sampled at 8 kHz, 16 bit linear, with continuous monitoring and without squelch or silence elimination. (2) the UWB-ATCC corpus by the University of West Bohemia, which can be downloaded for free at the following URL: <https://lindat.mff.cuni.cz/repository/xmlui/handle/11858/00-097C-0000-0001-CCA1-0> (accessed on 10 October 2023). The UWB-ATCC corpus contains recordings of air traffic control communication. The speech is manually transcribed with the speaker information; thus, it can be used for speaker role detection) datasets. We evaluate the SRD under two considerations: (i) ablations of different pretrained models for SRD on ATC communications, and (ii) low-resource and incremental training scenarios.

(i) *Analysis of the Impact of Pretrained Models and Training Data Type.* In this scenario, we evaluate the impact of pretrained models and training data on the SRD task for ATC data. To this end, we compare the performance of three transformer-based [69] models, including BERT, RoBERTa, and deBERTa-V3, trained on two different corpora, LDC-ATCC and UWB-ATCC, and evaluate them on the ATCO2-test-set-4h corpus. The F1-scores for SRD are reported separately for ATCO and pilot speakers in Table 7. Our results show that all the models achieved comparable F1-scores, ranging from 87–88% for ATCO and 84–85% for pilots. These findings suggest that the SRD task for ATC data is not significantly sensitive to the choice of pretrained models. However, we observe that models trained on UWB-ATCC outperform those trained on LDC-ATCC, with up to 4% absolute improvement in F1-scores. For instance, BERT-model with LDC-ATCC → UWB-ATCC gives a comparison of 82.4% → 86.2% for ATCO and 79.2% → 83.2%, for Pilot. Additionally, we find that combining both datasets leads to a 1% absolute improvement in F1-scores. Overall, our study highlights the importance of selecting appropriate training data for the SRD task in ATC data and suggests that using multiple datasets can lead to improved performance. The findings also suggest that the choice of pretrained models has a relatively minor impact on the SRD task for ATC data.

Table 7. ATCO/PILOT F1-scores for speaker role identification based on full ATC utterances for ATCO2-test-set-4 test set. Each utterance represents one sample. Metrics reported with three different transformer-based models (BERT [70], RoBERTa [71], deBERTa-V3 [80]). All models are the “base” version, e.g., bert-base. Numbers in **bold** refer to the top performance per split, i.e., **ATCO** or **PILOT**. Results are averaged over a fivefold cross-validation scheme on ATCO2-test-set-4h corpus in order to mitigate overfitting. Each round of fine-tuning is run five times with different training seeds (2222/3333/4444/5555/6666).

Training Corpora	BERT		DEBERTA		ROBERTA	
	ATCO	PILOT	ATCO	PILOT	ATCO	PILOT
LDC-ATCC	82.4	79.2	82.4	79.6	84.0	80.2
UWB-ATCC	86.2	83.2	86.8	84.0	87.0	82.8
↔ + LDC-ATCC	87.6	85.2	88.8	85.8	88.0	84.2

(ii) *Analysis of the Impact of Data Quantity on Speaker Role Detection.* In this study, we aim to evaluate the impact of the number of text samples on the performance of SRD. The results of this analysis are illustrated in the left panel of Figure 11, where the F1-score on the ATCO2-test-set-4h is plotted against the number of samples in a logarithmic scale on the x -axis. Interestingly, we found that as few as 100 samples are necessary to achieve a reasonably good F1-score of 60% on SRD. Notably, the UWB-ATCC appears to be more informative for the BERT model, which achieves an F1-score of 71% with only 100 training samples. Increasing the training data to 1000 samples further improves the performance,

resulting in F1-scores near 80% (LDC-ATCC + UWB-ATCC). These findings are significant, considering that the gold transcription of ATC communications is generally expensive and time-consuming. In the right panel of Figure 11, we present a box plot that shows the variation of the BERT model's performance when fine-tuned on SRD with different training seeds. Each box represents the variation of the model between the ATCo and pilot subsets, over the fivefold cross-validation scheme. Overall, the results indicate that increasing the training data leads to better performance and more consistent results. These observations highlight the importance of selecting a suitable training set size for speaker role detection tasks.

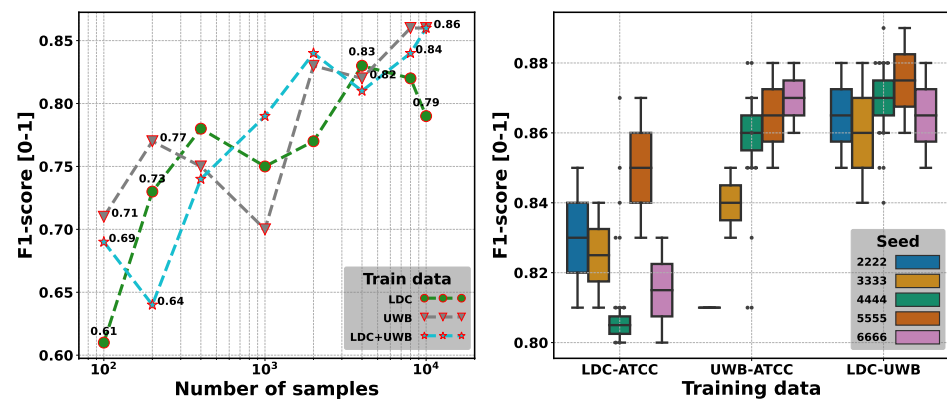


Figure 11. Metrics for the speaker role detection system (introduced in [7]). Metrics are reported only on ATCO2-test-set-1h corpus with a bert-base-uncased model trained with different datasets from Table 1. Left plot: ablation of the F1-score versus the number of samples used to train the system. Right plot: F1-score for models trained with different training seeds. The box plot depicts the performance variability when splitting the test set into ATCo and pilot subsets.

5.3.3. Text-Based Diarization

In addition to only detecting roles in a given ATC communication (e.g., SRD), there are cases where multiple segments end up in the same recording/communication. The task that solves this issue is known as speaker diarization (SD). SD answers the question “who spoke when?”. Here, the system receives an audio signal or recording (or text, in our case) and detects the speaker changes or segmentation and the speaker role. The main parts of an SD system are (i) segmentation, (ii) embedding extraction, (iii) clustering, and (iv) labeling (similar to SRD). SD is normally performed on the acoustic level, and previous work based on mel filterbank slope and linear filterbank slope was covered in [82]. Speaker discriminative embeddings such as x-vectors are investigated in [83], and, more recently, a variational Bayesian hidden Markov model (VBx) was investigated in [84], which is the SD system used during the data collection stage of ATCO2 (see Section 3.2). State-of-the-art SD systems are based on the E2E paradigm, named E2E neural diarization (EEND) [85]. This approach was introduced in [86] where an SD model is trained jointly to perform extraction and clustering [87]. Here, differently from SRD, we only used the BERT [70] pretrained model.

Experimental setup: The SD system is built on top of a pretrained BERT model downloaded from HuggingFace [72,73]. As in the NER and SRD system, here, the experimental setup is the same; this also includes the training hyperparameters. For further details we redirect the reader to Section 5.3.1. The SD model is fine-tuned on the NER task, where each speaker role (ATCo or pilot) is a class. Therefore, we have two tags per class, accounting for four classes in total. Readers are directed to our paper on text-based SD presented at The 2022 IEEE Spoken Language Technology Workshop (SLT 2022), see [8].

Evaluation metric: to score the text-based SD system, we use the Jaccard error rate (JER) metric. JER is a recent metric introduced in [88] that aligns with speaker diarization.

JER aims at avoiding the bias that the predominant speaker might cause, i.e., JER evaluates all speakers equally. The JER is defined in Equation (9):

$$JER = 1 - \frac{1}{\#\text{speakers}} \sum_{\text{speaker}} \max_{\text{cluster}} \frac{|\text{speaker} \cap \text{cluster}|}{|\text{speaker} \cup \text{cluster}|} \quad (9)$$

where (i) speaker is the selected speaker from reference and (ii) \max_{cluster} is the cluster from the system with maximum overlap duration with the currently selected speaker.

Results and analysis: we evaluate the SRD system on ATCO2-test-set-4h corpus. Differently from the NER system, but similar to SRD, here, we have access to two training corpora: LDC-ATCC and UWB-ATCC datasets. We evaluate the SD under one consideration: (i) low-resource and incremental training scenario.

(i) Analysis of the Impact of Data Quantity on Text-based Speaker Diarization In this study, we aim to evaluate the impact of the number of text samples on the performance of SD. The results of this analysis are illustrated in the left panel of Figure 12, where the JER (the lower the better) on the ATCO2-test-set-4h is plotted against the number of samples in a logarithmic scale on the x -axis. We found that as few as 100 samples are necessary to achieve a JER score of 45.6% (LDC-UWB). Similar to SRD, the UWB-ATCC dataset seems to be more informative in the SD system. For instance, under the 1000 samples scenario, we noted a 5% absolute JER reduction if UWB-ATCC is used. Furthermore, increasing the training data to 10k samples improved the performance, resulting in JER scores near to 20% (LDC+UWB). A more appropriate comparison of text and acoustic-based SD for ATC communications can be found in our previous work [8]. Additionally, in the right panel of Figure 11, we present a box plot that shows the variation of the BERT-based SD model's performance when fine-tuned with different training seeds. Each box represents the variation of the model between the two proposed classes: ATCo and pilot, over the fivefold cross-validation scheme. The results are listed with F1-scores. Overall, we can conclude that the UWB-ATCC dataset is more informative for the SD model in comparison to the LDC-ATCC dataset.

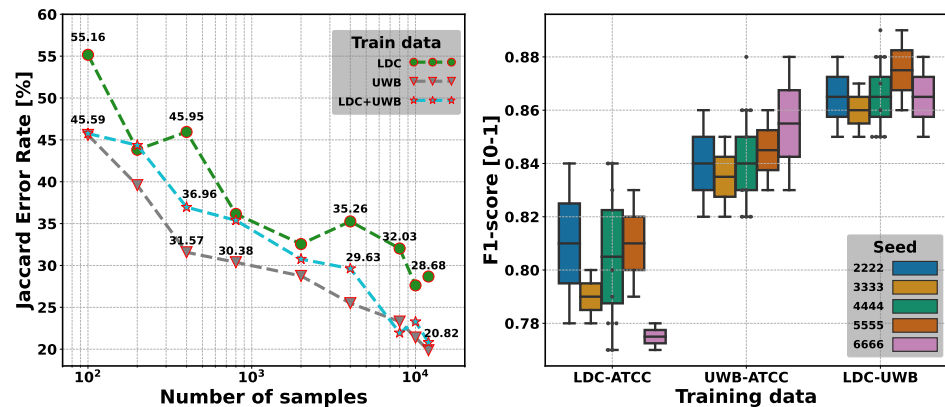


Figure 12. Metrics for the text-based diarization system (introduced in [7,8]). Metrics are reported only on ATCO2-test-set-4h corpus with a bert-base-uncased model trained with different datasets from Table 1. Left plot: ablation of the Jaccard error rate versus the number of samples used to train the system. Right plot: F1-score for models trained with different training seeds. The box plot depicts the performance variability when splitting the test set by ATCo and pilot subsets.

5.4. Future Work Enabled by ATCO2

In this subsection, we discuss several research directions that can be explored with the ATCO2 corpora. We cover (i) end of communication detection (akin to VAD), (ii) read-back error detection, and (iii) English language detection.

5.4.1. End of Communication Detection

In the ATC domain, it is crucial to detect the end of communications. While push-to-talk (PTT) signals are commonly acquirable in the ATC operations room or in the cockpit, there are cases where PTT is not available, and in such scenarios, the ATCO2 corpora can be leveraged to develop end-of-communication detection systems using either acoustic- or text-based approaches. Acoustic-based systems, known as VAD, perform their task prior to the ATC communication being sent to the ASR system [89], but may require the integration of a new independent module into the recognition pipeline. Text-based systems rely on strong artificial intelligence models like BERT, and previous studies in ATC [8] have shown their effectiveness in detecting callsigns [90], commands [7], and end-of-communication signals from transcripts generated by an ASR system.

5.4.2. Read-Back Error Detection

Pilot read-backs happens when a pilot speaks back the relevant instructions initially uttered by the ATCo. In practice, the ATCo is listening and checking the conformity of each read-back. Therefore, it is important to have a procedure in place, e.g., a read-back error detection (RBED) system. Despite the infrequency of communication errors in ATC, they still have the potential to cause significant safety issues, with some transmissions containing multiple errors. Authors in [91] show that in every hundredth ATC communication, an error may occur, and in [92], the authors show that the error may occur in every sixteenth communication. The possibility to detect such error still remains a challenge, as shown in this recent work [93]. Although, in general, read-back errors are quite rare, preventing even one incident due to automatic RBED can make an important difference in ensuring ATM safety. To support ATCos in this task, previous projects employ ASRU engines to extract high-level information from ATC communications [5]. Previous work in [93] has proposed two approaches for performing RBED. One system is based on rules, while a second system is a data-driven sequence classifier based on a BERT-alike pretrained encoder, named RoBERTa [71]. Here, the input sequence is a concatenation of ATCo and pilot utterance transcriptions with a special separator token [SEP] between them. They show that combining these approaches results in an 81% RBED rate in real-life voice recordings from Isavia's en-route airspace. They also cover a proof-of-concept trial with six ATCos producing challenging, artificial read-back error samples.

A main issue with well-known past projects, such as HAAWAI or MALORCA, is that their data cannot be publicly shared. In contrast, ATCO2 corpora are open to the public, e.g., *ATCO2-test-set-1h* set can be accessed for free, and practitioners can follow previous research to implement an RBED module.

5.4.3. English Language Detection

Currently, we have developed and deployed a suitable English language detection system (ELD) to discard non-English utterances in newly collected data. We tested a state-of-the-art acoustic-based system with an x-vector extractor. We also came up with the idea of using an NLP approach that processes ASR output with word confidence for the ELD. Finally, our experiments show that the ELD based on NLP is superior to the acoustic approach in both detection accuracy and computational resources. Moreover, the NLP approach can use outputs from several ASR systems jointly, which further improves the results. For the processing pipeline, we integrated the NLP-based English detector operating on Czech and English ASR. The integrated English detector consists of TF-IDF (term frequency-inverse document frequency) for reweighting the accumulated "soft" word counts and a logistic regression classifier to obtain the English/non-English decision [24].

We created the development and evaluation dataset consisting of data from various airports, data with various English accents, and code-mixing of English and local languages. The data are selected from our ATCO2 corpora introduced in Table 1. The development set is used to estimate the model parameters of our English language detector (the logistic regression classifier). The evaluation set is used for testing. The rules for manually tran-

scribing the utterances are mentioned below. We found several interesting properties of the ATC data during listening and tagging the ELD dataset:

- Various noise conditions. The majority of data are clean, but there are some very noisy segments;
- Strongly accented English. The speakers' English accent varies widely. From native speakers (pilots) to international accents (French, German, Russian, etc.) (pilots and ATCos) and strong Czech accents (pilots and ATCos);
- Mixed words and phrases. For example, the vocabulary of Czech ATCos is a mix of Czech and English words. They use standard greetings in Czech which can be a significant portion of an "English" sentence if a command is short. On the other hand, they use many English words (alphabet, some commands) in "Czech" sentences. Moreover, they use a significant set of "Czenglish" words.

We use the language of spoken numerals as a rule of thumb to decide on the language of a particular ATCo-pilot communication utterance. The language has to be consistent within the audio recording. More detailed information, including experimental results, is covered in our previous work [24].

6. Conclusions

This paper expands upon our previous work [7] and discusses the main lessons learned from the ATCO2 project. The aim of the ATCO2 project was to develop a platform for collecting, preprocessing, and posterior ASR-based transcription generation of ATC communications audio data. With over 5000 h of ASR transcribed audio data, ATCO2 is the largest public ATC dataset to date, thus pushing the research boundaries on robust automatic speech recognition and natural language understanding of ATC communications. The main lessons learned from ATCO2 are sixfold, as follows:

- **Lesson 1:** ATCO2's automatic transcript engine (see Appendix B) and annotation platform (see Appendix C) have proven to be reliable ($\sim 20\%$ WER on ATCO2-test-set-4h) for collection of a large-scale audio dataset targeted to ATC communications;
- **Lesson 2:** Good transcription practices for ATC communications have been developed based on ontologies published by previous projects [5]. A cheat sheet (see Appendix E) has been created to provide guidance for future ATC projects and reduce confusion while generating transcripts;
- **Lesson 3:** The most demanding modules of the ATCO2 collection platform are the speaker diarization and automatic speech recognition engines, each accounting for $\sim 32\%$ of the overall system processing time. The complete statistics regarding runtime are covered in the Table 3. In ATCO2, we make these numbers public so they can be used as baselines in future work aligned to reducing the overall memory and runtime footprint of large-scale collection of ATC audio and radar data;
- **Lesson 4:** Training ASR systems purely on ATCO2 datasets (e.g., *ATCO2-T 500h set corpus*) can achieve competitive WERs on ATCO2 test sets (see Table 4). The ASR model can achieve up to 17.9%/24.9% WERs on ATCO2-test-set-1h/ATCO2-test-set-4h, respectively. More importantly, these test sets contain noisy accented speech, which is highly challenging in standard ASR systems;
- **Lesson 5:** ATC surveillance data are an optimal source of real-time information to improve ASR outputs. The integration of air surveillance data can lead to up to 11.8% absolute callsign WERs reduction, which represents an amelioration of 20% (62.6% no boosting \rightarrow 82.9% GT boosted) absolute callsign accuracy in ATCO2-test-set-4h, as shown in Table 5;
- **Lesson 6:** ATCO2 corpora can be used for natural language understanding of ATC communications. BERT-based NER and speaker role detection modules have been developed based on ATCO2-test-set-4h. These systems can detect callsigns, commands, and values from the textual inputs. Additionally, speaker roles can also be detected based on textual inputs. For instance, as few as 100 samples are necessary to achieve

60% F1-score on speaker role detection. Furthermore, the NLU task is of special interest to the ATC community because this high-level information can be used to assist ATCos in their daily tasks, thus reducing their overall workload.

In addition to these six lessons learned, this paper brings substantial improvements in the domain of automatic speech recognition and understanding for ATC domain, i.e., Tables 5 and 6 show the current best-performing ASR and NLU engines developed on open-source data, and, thus, are replicable by the community. Furthermore, to the authors' knowledge, there is no other research or commercial activity at this moment which would demonstrate a more accurate engine for an ATC domain built on publicly open data.

Author Contributions: Conceptualization, J.Z.-G., S.M., I.S., V.L., M.R. and K.C.; Data curation, J.Z.-G. and I.N.; Formal analysis, J.Z.-G. and I.S.; Funding acquisition, P.M.; Investigation, J.Z.-G.; Methodology, J.Z.-G., I.N., A.P. and A.T.; Project administration, P.M.; Resources, J.Z.-G.; Software, J.Z.-G., S.M., A.T. and I.S.; Supervision, P.M., S.M., I.S. and V.L.; Validation, J.Z.-G., A.P. and I.S.; Visualization, J.Z.-G. and M.R.; Writing—original draft, J.Z.-G. and I.N.; Writing—review and editing, J.Z.-G., I.N., A.P., P.M., D.K., V.L., M.R. and K.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This paper presents a continuation of our previous work [7], the *ATCO2 Corpora*. ATCO2 is derived from a joint contribution from Clean Sky 2 Joint Undertaking (JU) and EU-H2020, which was supported by Clean Sky 2 Joint Undertaking (JU) and EU-H2020, under Grant Agreement No. 864702—ATCO2 (Automatic collection and processing of voice data from air traffic communications). See our website at <https://www.atco2.org/> (accessed on 10 October 2023).

Conflicts of Interest: The authors declare no conflicts of interest.

Nomenclature

AI	Artificial Intelligence
AM	Acoustic Model
ATC	Air Traffic Control
ATM	Air Traffic Management
ASR	Automatic Speech Recognition
ATCo	Air Traffic Controller
ATCC	Air Traffic Control Corpus
ADS-B	Automatic Dependent Surveillance–Broadcast
CTC	Connectionist Temporal Classification
Conformer	Convolution-augmented Transformer
dB	Decibel
DNN	Deep Neural Networks
E2E	End-To-End
ELDA	European Language Resources Association
FST	Finite State Transducer
ICAO	International Civil Aviation Organization
GELU	Gaussian Error Linear Units
LF-MMI	Lattice-Free Maximum Mutual Information
LM	Language Model
ML	Machine Learning
NER	Named Entity Recognition
NLP	Natural Language Processing
NLU	Natural Language Understanding
PTT	Push-To-Talk
SNR	Signal-To-Noise
VAD	Voice Activity Detection
VHF	Very-High Frequency
WER	Word Error Rate
WFST	Weighted Finite State Transducer

RBE	Read-back Error
RBED	Read-back Error Detection

Appendix A. ATCO2 Project

The ATCO2 project developed an unique platform that allows the collection, organization, and preprocessing of air traffic control (voice communication) data from airspace. The project considers real-time voice communication between air traffic controllers and pilots available either directly through publicly accessible radio frequency channels, or indirectly from air-navigation service providers (ANSPs). In addition to the voice communication, the contextual information available in a form of metadata (i.e., surveillance data) is exploited.

More specifically, data acquisition was based on off-the-shelf automatic-dependent surveillance-broadcast (ADS-B) technology already exploited by OpenSky Network (OSN), collecting detailed (live) aircraft information over the publicly accessible 1090 MHz radio frequency channel. ADS-B sensors are distributed among volunteers (i.e., community of users) throughout the world (at <https://opensky-network.org/network/facts>; accessed on 10 October 2023), making it possible to analyze billions of ADS-B messages. The aim of ATCO2 was to extend the current cloud setting (i.e., central server responsible for managing the sensors, collecting the received ADS-B messages, and storing all received information in a database) so that ATC voice communication of both channels (ATCo and pilot's read back) will be captured, time correlated with the surveillance data, and stored in the database for further processing. These data will also be complemented by air traffic voice communication data provided by ANSPs. Below are listed some links that might be of interest to the reader.

- The latest news and blog posts from ATCO2 project are located in the following website: <https://www.atco2.org/>; accessed on 10 October 2023.
- The ATCO2 corpus can be downloaded for a fee at <https://catalog.elra.info/en-us/repository/browse/ELRA-S0484/>; accessed on 10 October 2023.
- The ATCO2-test-set-1h can be downloaded for free at <https://www.atco2.org/data>; accessed on 10 October 2023.
- Stats and voice feeding of ATC data is listed at <https://ui.atc.opensky-network.org/set-up>; accessed on 10 October 2023.
- ATC training and transcription service is provided by SpokenData at: <https://www.spokendata.com/atco2>; accessed on 10 October 2023.

Appendix B. Automatic Transcription Engine

This appendix describes in detail how we collected the audio and metadata that brought to life the *ATCO2 corpus*. We mainly rely on the automatic transcription engine, described in more detail in Section 3.2. The automatic transcription engine is implemented as a scalable cloud service. It communicates with other services (or partners) using APIs. This service is designed to process large flows of data produced by data feeders. Data feeders are enthusiasts that act as “feeders” of ATC speech and contextual ATC data (e.g., surveillance), see Section 4.2.

The data are pushed to this service by OSN (OpenSky Network: <https://opensky-network.org/>; accessed on 10 October 2023) servers by calling an API request and providing a job setting JSON file. After the request is accepted, settings parameters are processed and the job is stored in an internal queue for processing. The user (in this case, OSN) may have an ability to tweak the settings and to affect the processing pipeline and the result, namely:

- Audio input format choices;
- Rejection threshold for too-long audio;
- Rejection threshold for too-short audio;
- Rejection threshold for too-noisy audio;
- Rejection threshold for non-English audios;
- Switching the language of automatic speech recognizer.

Most of these are actually disabled due to security reasons (not to interrupt the processing pipeline), but may be easily enabled on the fly if needed. The overall data flow model is described in Figure A1. Any new job (request for a full automatic transcription of recording) accepted via API on the SpokenData (Industrial partner: <https://www.spokendata.com/atco2>; accessed on 10 October 2023) side is processed by a master processing node. The job is enqueued into a workload manager queue. Once there is a free processing slot, the job is submitted to a processing server, or worker. The master processing node then informs the OSN server about the state of the job by calling a callback.

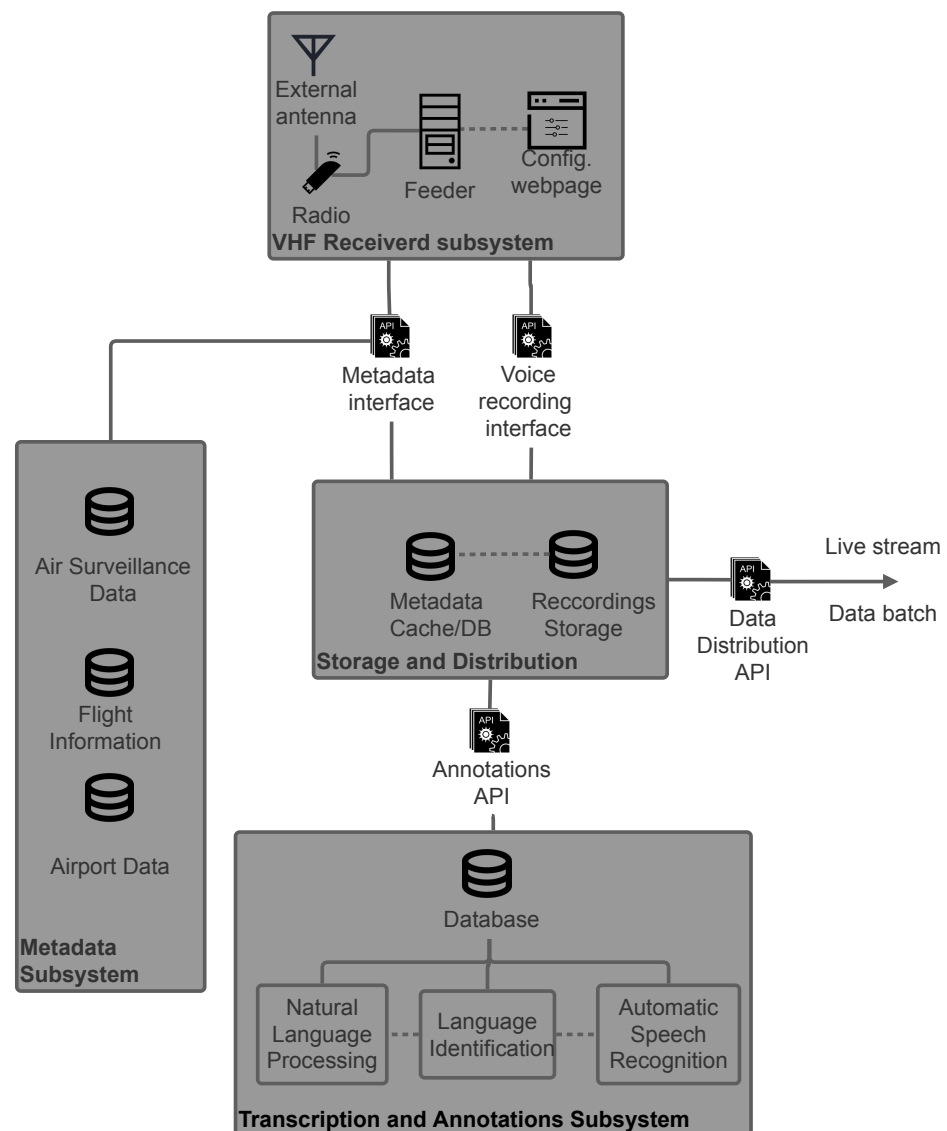


Figure A1. ATCO2 communication schema.

Appendix C. Transcription Platform: Data Flow

The data (the recording for human transcription) life cycle is split into four main states: The **new recording** state is set as queued and is untouched when the recording is pushed into the transcription platform from the transcription engine. The recording is placed into a queue of transcription jobs and is immediately visible to all annotators. The queue is shown in the open jobs screen. Annotators can interact with the queue—listen to recordings and select some for transcription. Recording in this state may drop off the queue in the case: they are old—no one is interested in annotating them; three annotators marked the recording by thumbs down. The dropped-off recordings are deleted after 7 days.

Once an user selects the recording for transcription, revises the automatic transcription, and saves it, the recording is set as **queued and annotated**. This state prevents the recording from being dropped from the queue and deleted. Also, it is indicated as (to) re-check in the open jobs screen, to inform other annotators that it was modified (annotated) and they should recheck if the transcription is correct rather than annotate from scratch. If any annotator indicates the existence of personal information in the recording (by “Anonymize” label), the recording is dropped off the queue and deleted.

The next state is **annotated**. If the recording is successfully rechecked, then the recording is considered as annotated and the transcription is final. The recording is removed from the open jobs queue and placed on a stack of finished recording transcriptions. The stack is periodically exported to ELRA for further packaging and distribution to the community. This state also triggers a callback to the OSN platform, informing them that the human transcription is completed, and they can download the transcription. After the recording is exported to ELRA, we set the state as **Finished**. Here, the recording can be archived or deleted. The detailed data flow schema is depicted in Figure A2.

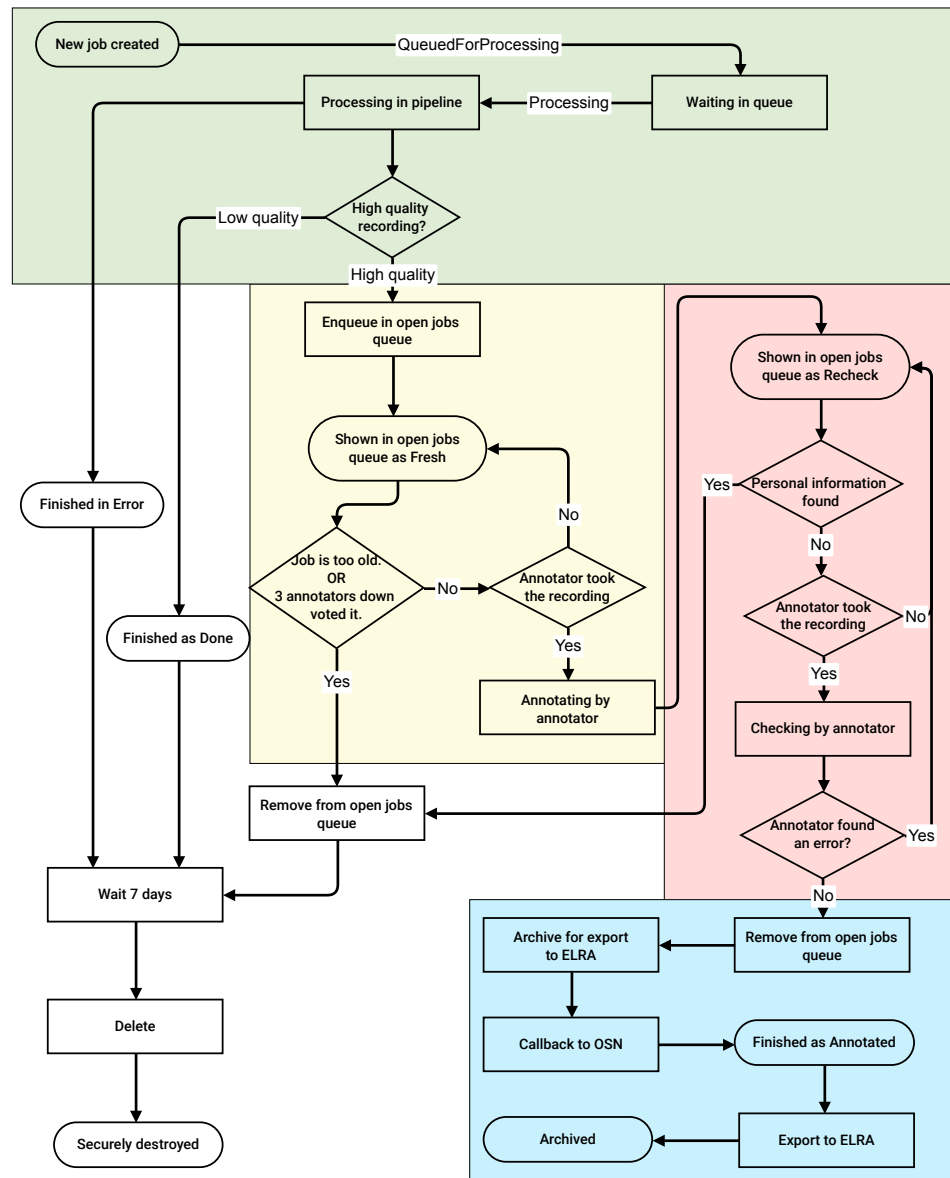


Figure A2. Diagram of the data flow (lifetime) in the transcription platform. Transcription engine in green. Queued and untouched state in yellow. Queued and annotated state in red. Annotated state in blue. The rest (white) is for state, securely destroyed.

Appendix D. Communication Schema

The communication schema developed during ATCO2 project is depicted in Figure A3.

ATCO2 communication schema

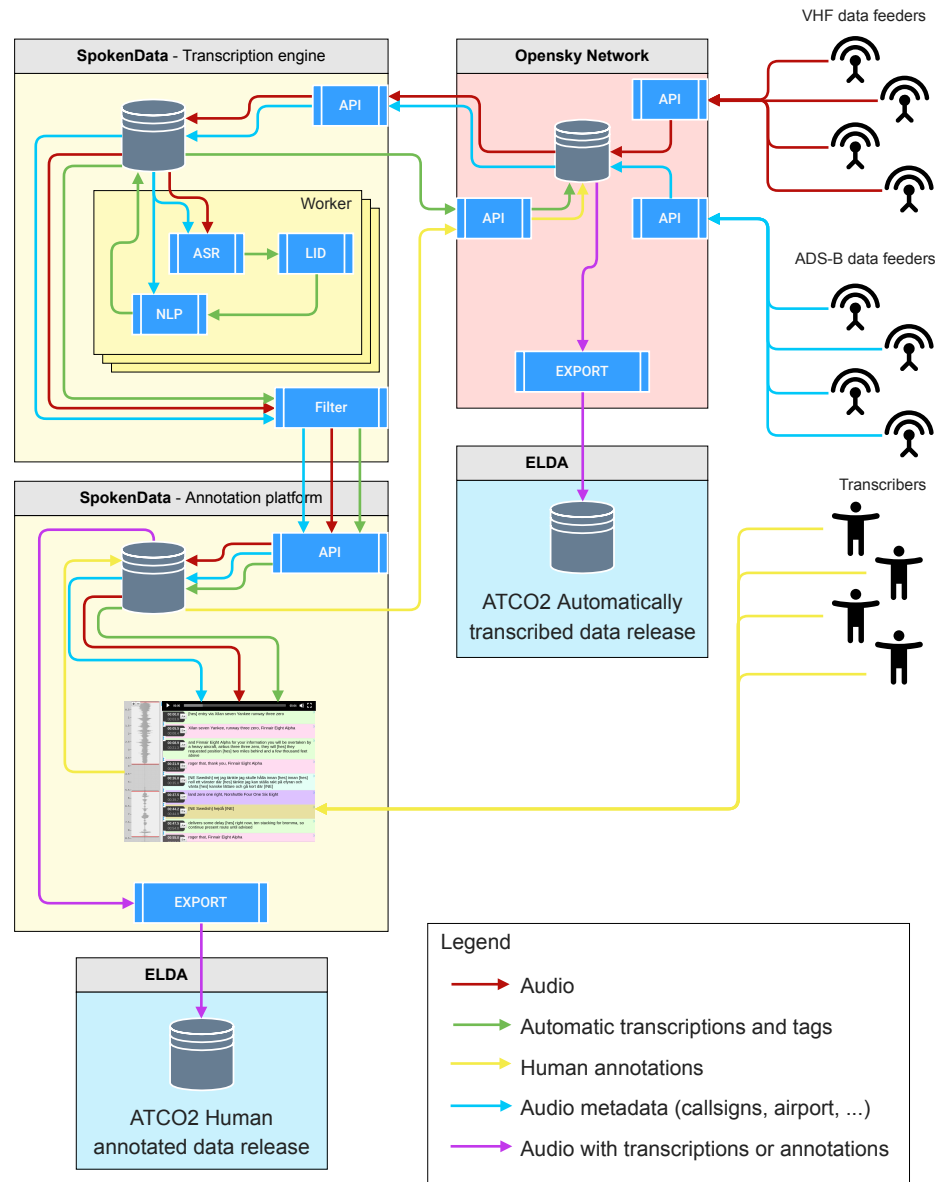


Figure A3. Other view of the ATCO2 communication schema.

Appendix E. Transcription Cheat Sheet

Figure A4 presents the transcription cheat sheet developed by ATCO2 project.

Global

Do not waste your time on really bad files or segments. Refuse the job or ignore the segment. If you do not understand -> write <UNK>, if you are not sure -> do not label.

We prefer QUALITY not quantity

Segment is in English if commands values and numerals are in English. Ignore greetings (but indicate them by using [NE] [NE] tag)

Indicate segments which are correctly transcribed by checking the **Correct transcript**.

Indicate segments which are correctly labeled by checking the **Correct labeling**.

Indicate segments which are not in English checking the **Non-English button**.

When the file is finished mark the job as **DONE**

Use provided information (FRV/VR manuals, list of waypoints and callsigns, etc).

Segmentation and speaker "identity"

audio segment = speaker utterance

segment boundaries are in pauses (check the timing)

delete segments without speech

each segment has to have attached a speaker tag

be specific in the identity, attach ATCO-Radar, ATCO-Tower, Call-sign

if the identity is not clear distinguish speakers using UNK-1, UNK-2, UNK-3

long passages of cross talks set as segment and tag Crosstalk

Abbrev.

acas	Alfa
AFIS	Bravo
AIP	Charlie
AMSL	Delta
ATC	Echo
atis	Footrot
ATS	Golf
ATZ	Hotel
fato	India
FIS	Juliett
POB	Kilo
PTT	Lima
QDM	Mike
QDR	November
QFE	Oscar
QNH	Papa
QTE	Quebec
RNP	Romeo
RTF	Sierra
RVR	Tango
SSR	Uniform
VDF	Victor
VHF	Whiskey
VFR	X-ray
volmet	Yankee
	Zulu

Speaker TAG

Speaker TAG	Example
ATCO, ATCO radar, ATCO tower	ATCO side of the conversation
UNK-1, UNK-2 etc.	Unknown identity (call-sign) of segment
LH469	Correct call-sign should also appear in the audio
Crosstalk	Whole segment is Crosstalk

Transcription

only ASCII code letters are allowed A-Z a-z [] / ' - ()

everything is in lower case except **Call-signs** (Speedbird Charlie Nine Two), Airlines, Waypoints, Geographical Names

transcribe exactly what was said including re-restarts and repetitions

indicate swallowed or cutted words if possible Lufthansa(hansa) goodbye(goodb-)

numbers are written as pronounced (one hundred / one zero zero), do not use numerals 403

if abbreviations are spelled then capitalized QNH (as queange) lowercase otherwise atis (as atis)

whatever is not intelligible mark as unknown [unk] either word(s) or whole sentences

indicate hesitations like eehh, uuhhm by [hes]

label or enclose non-English parts of sentence into [NE] [unk] [NE] if you understand then be specific [NE French] bonjour [NE]

Any personal data must be labeled or tagged for later removal: "morning cpt. [PERS] John Doe [PERS] please"

Transcription TAG

Transcription TAG	Example
UNK-1, UNK-2 etc.	Unknown identity (callsign) of segment
XT	Whole segment is Crosstalk (blocking)
[PERS] Jon Doe [PERS]	Personal data
[unk]	Word(s) is not legible/understandable
[hes]	Clear hesitation (umh, uhh, hmhm)
[noise]	Non-speaker noise (alarm etc)
[bpk]	Speaker noise (laugh, cough etc)
[key]	Double-press PTT
[XT]	Small part of the segment is crosstalk
[NE] [unk] [NE]	Non-English language not identified
[NE langID] [NE]	Non-English but language identified e.g. [NE German] [NE]

correct

takeoff	take off, take-off
callsign	call sign, call-sign
stand by	standby (to not be mixed up with "taxi to your stand byby")
startup	start up
readback	read back
flight level	flightlevel
good bye	goodbye
line up	lineup
descend	descent (+ still correct but not preferred), decent
taxiway	taxi way

incorrect

take off, take-off	
call sign, call-sign	
standby (to not be mixed up with "taxi to your stand byby")	
start up	
read back	
flightlevel	
goodbye	
lineup	
descent (+ still correct but not preferred), decent	
taxi way	

Labeling

Assign words into classes: Call-sign, Command, Value, Unnamed Phrase, Anonymize and optionally non-English.

Call-sign consists of: airline identifier / name and alphanumeric code, or only the alphanumeric code.

Command / Value (upto(s)) usually follows the Call-sign

Unnamed Phrase use when it does not fit Call-sign, Command, Value category, but it is obvious it carries an information.

Prepositions and variations are part of the command: continue to, expect a turn after, contact now, descend for

You can indicate personal data using Anonymize label

You can indicate non-English part of transcript using non-English label

Commands and Values

descend flight level one five

climb to flight level one two zero

maintain flight level one five

continue heading value

left/right heading value

continue present heading value

maintain heading value

heading two one six

turn right

set altitude three thousand feet

direct taxi

vector for LS six five

cross runway two eight seven

line up runway two eight

vacate zero five zero

cleared to cross runway two eight seven

speed two five zero knots

reduce speed two five zero knots

contact ground one two one seven five


contact Zurich Approach

call Swiss two six eight zero

request startup and clearance

report established

expect sectors by ILS approach runway four



Annotation manual cheat sheet: v6 10.02.2022
<https://atc.spokendata.com>

Figure A4. Cheat sheet for ATC communications annotation. This document was created during the transcription process of ATCO2 corpora, which can be used to transcribe air traffic control communications data from different airports.

References

- Helmke, H.; Ohneiser, O.; Buxbaum, J.; Kern, C. Increasing ATM efficiency with assistant based speech recognition. In Proceedings of the 13th USA/Europe Air Traffic Management Research and Development Seminar, Seattle, WA, USA, 27–30 June 2017.
- Shetty, S.; Helmke, H.; Kleinert, M.; Ohneiser, O. Early Callsign Highlighting using Automatic Speech Recognition to Reduce Air Traffic Controller Workload. In Proceedings of the International Conference on Applied Human Factors and Ergonomics (AHFE2022), New York, NY, USA, 24–28 July 2022; Volume 60, pp. 584–592.
- Ohneiser, O.; Helmke, H.; Shetty, S.; Kleinert, M.; Ehr, H.; Murauskas, Š.; Pagirys, T.; Balogh, G.; Tønnesen, A.; Kis-Pál, G.; et al. Understanding Tower Controller Communication for Support in Air Traffic Control Displays. In Proceedings of the SESAR Innovation Days, Budapest, Hungary, 5–8 December 2022.
- Helmke, H.; Ohneiser, O.; Mühlhausen, T.; Wies, M. Reducing controller workload with automatic speech recognition. In Proceedings of the 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC), Sacramento, CA, USA, 25–29 September 2016; pp. 1–10.
- Helmke, H.; Slotty, M.; Poiger, M.; Herrer, D.F.; Ohneiser, O.; Vink, N.; Cerna, A.; Hartikainen, P.; Josefsson, B.; Langr, D.; et al. Ontology for transcription of ATC speech commands of SESAR 2020 solution PJ. 16-04. In Proceedings of the IEEE/AIAA 37th Digital Avionics Systems Conference (DASC), London, UK, 23–27 September 2018; pp. 1–10.
- Guo, D.; Zhang, Z.; Yang, B.; Zhang, J.; Lin, Y. Boosting Low-Resource Speech Recognition in Air Traffic Communication via Pretrained Feature Aggregation and Multi-Task Learning. *IEEE Trans. Circuits Syst. II Express Briefs* **2023**, *70*, 3714–3718. [CrossRef]
- Zuluaga-Gomez, J.; Veselý, K.; Szöke, I.; Motlicek, P.; Kocour, M.; Rigault, M.; Choukri, K.; Prasad, A.; Sarfjoo, S.S.; Nigmatulina, I.; et al. ATCO2 corpus: A Large-Scale Dataset for Research on Automatic Speech Recognition and Natural Language Understanding of Air Traffic Control Communications. *arXiv* **2022**, arXiv:2211.04054.
- Zuluaga-Gomez, J.; Sarfjoo, S.S.; Prasad, A.; Nigmatulina, I.; Motlicek, P.; Ondre, K.; Ohneiser, O.; Helmke, H. BERTTraffic: BERT-based Joint Speaker Role and Speaker Change Detection for Air Traffic Control Communications. In Proceedings of the IEEE Spoken Language Technology Workshop (SLT), Doha, Qatar, 9–12 January 2023.
- Kocour, M.; Veselý, K.; Szöke, I.; Kesiraju, S.; Zuluaga-Gomez, J.; Blatt, A.; Prasad, A.; Nigmatulina, I.; Motlíček, P.; Klakow, D.; et al. Automatic processing pipeline for collecting and annotating air-traffic voice communication data. *Eng. Proc.* **2021**, *13*, 8.
- Ferreiros, J.; Pardo, J.; De Córdoba, R.; Macias-Guarasa, J.; Montero, J.; Fernández, F.; Sama, V.; González, G. A speech interface for air traffic control terminals. *Aerosp. Sci. Technol.* **2012**, *21*, 7–15. [CrossRef]

11. Tarakan, R.; Baldwin, K.; Rozen, N. An automated simulation pilot capability to support advanced air traffic controller training. In Proceedings of the 26th Congress of ICAS and 8th AIAA ATIO, Anchorage, AK, USA, 14–19 September 2008.
12. Cordero, J.M.; Dorado, M.; de Pablo, J.M. Automated speech recognition in ATC environment. In Proceedings of the 2nd International Conference on Application and Theory of Automation in Command and Control Systems, London, UK, 29–31 May 2012; pp. 46–53.
13. Zuluaga-Gomez, J.; Motlicek, P.; Zhan, Q.; Vesely, K.; Braun, R. Automatic Speech Recognition Benchmark for Air-Traffic Communications. In Proceedings of the Interspeech, Shanghai, China, 25–29 October 2020; pp. 2297–2301. [CrossRef]
14. Zuluaga-Gomez, J.; Nigmatulina, I.; Prasad, A.; Motlicek, P.; Vesely, K.; Kocour, M.; Szöke, I. Contextual Semi-Supervised Learning: An Approach to Leverage Air-Surveillance and Untranscribed ATC Data in ASR Systems. In Proceedings of the Interspeech, Brno, Czech Republic, 30 August–3 September 2021; pp. 3296–3300. [CrossRef]
15. Nigmatulina, I.; Zuluaga-Gomez, J.; Prasad, A.; Sarfjoo, S.S.; Motlicek, P. A two-step approach to leverage contextual data: Speech recognition in air-traffic communications. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Virtual, 7–13 May 2022.
16. Chen, S.; Kopald, H.; Avjian, B.; Fronzak, M. Automatic Pilot Report Extraction from Radio Communications. In Proceedings of the 2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC), Portsmouth, VA, USA, 18–22 September 2022; pp. 1–8.
17. Godfrey, J. The Air Traffic Control Corpus (ATC0)—LDC94S14A. 1994. Available online: <https://catalog ldc.upenn.edu/LDC94S14A> (accessed on 4 September 2023).
18. Šmídl, L.; Švec, J.; Tihelka, D.; Matoušek, J.; Romportl, J.; Ircing, P. Air traffic control communication (ATCC) speech corpora and their use for ASR and TTS development. *Lang. Resour. Eval.* **2019**, *53*, 449–464. [CrossRef]
19. Segura, J.; Ehrette, T.; Potamianos, A.; Fohr, D.; Illina, I.; Breton, P.; Clot, V.; Gemello, R.; Matassoni, M.; Maragos, P. The HIWIRE Database, a Noisy and Non-Native English Speech Corpus for Cockpit Communication. 2007. Available online: <http://www.hiwire.org> (accessed on 10 October 2023).
20. Delpech, E.; Laignelet, M.; Pimm, C.; Raynal, C.; Trzos, M.; Arnold, A.; Pronto, D. A Real-life, French-accented Corpus of Air Traffic Control Communications. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan, 7–12 May 2018.
21. Pellegrini, T.; Farinas, J.; Delpech, E.; Lancelot, F. The Airbus Air Traffic Control speech recognition 2018 challenge: Towards ATC automatic transcription and call sign detection. *arXiv* **2018**, arXiv:1810.12614.
22. Graglia, L.; Favennec, B.; Arnoux, A. Vocalise: Assessing the impact of data link technology on the R/T channel. In Proceedings of the 24th IEEE Digital Avionics Systems Conference, Washington, DC, USA, 30 October–3 November 2005; Volume 1.
23. Lopez, S.; Condamines, A.; Josselin-Leray, A.; O'Donoghue, M.; Salmon, R. Linguistic analysis of English phraseology and plain language in air-ground communication. *J. Air Transp. Stud.* **2013**, *4*, 44–60. [CrossRef]
24. Szöke, I.; Kesiraju, S.; Novotný, O.; Kocour, M.; Vesely, K.; Černocký, J. Detecting English Speech in the Air Traffic Control Voice Communication. In Proceedings of the Interspeech, Brno, Czech Republic, 30 August–3 September 2021; pp. 3286–3290. [CrossRef]
25. International Civil Aviation Organization. *ICAO Phraseology Reference Guide*; International Civil Aviation Organization: Montreal, QC, Canada, 2020.
26. Panayotov, V.; Chen, G.; Povey, D.; Khudanpur, S. Librispeech: An ASR corpus based on public domain audio books. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, Australia, 19–24 April 2015; pp. 5206–5210.
27. Ardila, R.; Branson, M.; Davis, K.; Henretty, M.; Kohler, M.; Meyer, J.; Morais, R.; Saunders, L.; Tyers, F.M.; Weber, G. Common voice: A massively-multilingual speech corpus. *arXiv* **2019**, arXiv:1912.06670.
28. Godfrey, J.J.; Holliman, E.C.; McDaniel, J. SWITCHBOARD: Telephone speech corpus for research and development. In Proceedings of the Acoustics, Speech, and Signal Processing, IEEE International Conference on IEEE Computer Society, San Francisco, CA, USA, 23–26 March 1992; Volume 1, pp. 517–520.
29. Povey, D.; Ghoshal, A.; Boulianne, G.; Burget, L.; Glembek, O.; Goel, N.; Hannemann, M.; Motlicek, P.; Qian, Y.; Schwarz, P.; et al. The Kaldi speech recognition toolkit. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding IEEE Signal Processing Society, Waikoloa, HI, USA, 11–15 December 2011.
30. Jurafsky, D.; Martin, J.H. *Speech and Language Processing*, 2nd ed.; Prentice-Hall, Inc.: Upper Saddle River, NJ, USA, 2009.
31. Wang, D.; Wang, X.; Lv, S. An Overview of End-to-End Automatic Speech Recognition. *Symmetry* **2019**, *11*, 1018. [CrossRef]
32. Sennrich, R.; Haddow, B.; Birch, A. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*; Association for Computational Linguistics: Berlin, Germany, 2016; pp. 1715–1725. [CrossRef]
33. Kingsbury, B. Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling. In Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, Taipei, Taiwan, 19–24 April 2009; pp. 3761–3764.
34. Dehak, N.; Kenny, P.J.; Dehak, R.; Dumouchel, P.; Ouellet, P. Front-end factor analysis for speaker verification. *IEEE Trans. Audio Speech Lang. Process.* **2010**, *19*, 788–798. [CrossRef]

35. Snyder, D.; Garcia-Romero, D.; Povey, D. Time delay deep neural network-based universal background models for speaker recognition. In Proceedings of the 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Scottsdale, AZ, USA, 13–17 December 2015; pp. 92–97.
36. Peddinti, V.; Povey, D.; Khudanpur, S. A time delay neural network architecture for efficient modeling of long temporal contexts. In Proceedings of the Interspeech, Dresden, Germany, 6–10 September 2015; pp. 3214–3218. [[CrossRef](#)]
37. Povey, D.; Peddinti, V.; Galvez, D.; Ghahremani, P.; Manohar, V.; Na, X.; Wang, Y.; Khudanpur, S. Purely sequence-trained neural networks for ASR based on lattice-free MMI. In Proceedings of the Interspeech, San Francisco, CA, USA, 8–12 September 2016; pp. 2751–2755.
38. Povey, D.; Cheng, G.; Wang, Y.; Li, K.; Xu, H.; Yarmohammadi, M.; Khudanpur, S. Semi-orthogonal low-rank matrix factorization for deep neural networks. In Proceedings of the Interspeech, Hyderabad, India, 2–6 September 2018; pp. 3743–3747.
39. Nassif, A.B.; Shahin, I.; Attili, I.; Azzeh, M.; Shaalan, K. Speech recognition using deep neural networks: A systematic review. *IEEE Access* **2019**, *7*, 19143–19165. [[CrossRef](#)]
40. Graves, A.; Jaitly, N. Towards End-to-End Speech Recognition with Recurrent Neural Networks. In Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21–26 June 2014; Volume 32, pp. 1764–1772.
41. Karita, S.; Chen, N.; Hayashi, T.; Hori, T.; Inaguma, H.; Jiang, Z.; Someki, M.; Soplin, N.E.Y.; Yamamoto, R.; Wang, X.; et al. A comparative study on transformer vs rnn in speech applications. In Proceedings of the 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Singapore, 14–18 December 2019; pp. 449–456.
42. Graves, A.; Fernández, S.; Gomez, F.; Schmidhuber, J. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, Pennsylvania, 25–29 June 2006; pp. 369–376.
43. Chorowski, J.; Bahdanau, D.; Serdyuk, D.; Cho, K.; Bengio, Y. Attention-Based Models for Speech Recognition. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 577–585.
44. Watanabe, S.; Hori, T.; Kim, S.; Hershey, J.R.; Hayashi, T. Hybrid CTC/attention architecture for end-to-end speech recognition. *IEEE J. Sel. Top. Signal Process.* **2017**, *11*, 1240–1253. [[CrossRef](#)]
45. Oord, A.v.d.; Li, Y.; Vinyals, O. Representation learning with contrastive predictive coding. *arXiv* **2018**, arXiv:1807.03748.
46. Baevski, A.; Zhou, Y.; Mohamed, A.; Auli, M. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–12 December 2020.
47. Baevski, A.; Schneider, S.; Auli, M. vq-wav2vec: Self-Supervised Learning of Discrete Speech Representations. In Proceedings of the 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 26–30 April 2020.
48. Chen, S.; Wang, C.; Chen, Z.; Wu, Y.; Liu, S.; Li, J.; Kanda, N.; Yoshioka, T.; Xiao, X.; Wei, F.; et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *arXiv* **2021**, arXiv:2110.13900.
49. Babu, A.; Wang, C.; Tjandra, A.; Lakhota, K.; Xu, Q.; Goyal, N.; Singh, K.; von Platen, P.; Saraf, Y.; Pino, J.; et al. XLS-R: Self-supervised cross-lingual speech representation learning at scale. *arXiv* **2021**, arXiv:2111.09296.
50. Zuluaga-Gomez, J.; Prasad, A.; Nigmatulina, I.; Sarfjoo, S.; Motlicek, P.; Kleinert, M.; Helmke, H.; Ohneiser, O.; Zhan, Q. How Does Pre-trained Wav2Vec2.0 Perform on Domain Shifted ASR? An Extensive Benchmark on Air Traffic Control Communications. In Proceedings of the IEEE Spoken Language Technology Workshop (SLT), Doha, Qatar, 9–12 January 2023.
51. Gulati, A.; Qin, J.; Chiu, C.C.; Parmar, N.; Zhang, Y.; Yu, J.; Han, W.; Wang, S.; Zhang, Z.; Wu, Y.; et al. Conformer: Convolution-augmented Transformer for Speech Recognition. In Proceedings of the Interspeech, Shanghai, China, 25–29 October 2020; pp. 5036–5040. [[CrossRef](#)]
52. Mai, F.; Zuluaga-Gomez, J.; Parcollet, T.; Motlicek, P. HyperConformer: Multi-head HyperMixer for Efficient Speech Recognition. In Proceedings of the Interspeech, Dublin, Ireland, 20–24 August 2023.
53. Radfar, M.; Lyskawa, P.; Trujillo, B.; Xie, Y.; Zhen, K.; Heymann, J.; Filimonov, D.; Strimel, G.; Susanj, N.; Mouchtaris, A. Conmer: Streaming Conformer without self-attention for interactive voice assistants. In Proceedings of the Interspeech, Dublin, Ireland, 20–24 August 2023.
54. Peng, Y.; Dalmia, S.; Lane, I.; Watanabe, S. Branchformer: Parallel mlp-attention architectures to capture local and global context for speech recognition and understanding. In Proceedings of the International Conference on Machine Learning, Guangzhou, China, 18–21 February 2022; pp. 17627–17643.
55. Ravanelli, M.; Parcollet, T.; Plantinga, P.; Rouhe, A.; Cornell, S.; Lugosch, L.; Subakan, C.; Dawalatabad, N.; Heba, A.; Zhong, J.; et al. SpeechBrain: A general-purpose speech toolkit. *arXiv* **2021**, arXiv:2106.04624.
56. Watanabe, S.; Hori, T.; Karita, S.; Hayashi, T.; Nishitoba, J.; Unno, Y.; Enrique Yalta Soplin, N.; Heymann, J.; Wiesner, M.; Chen, N.; et al. ESPnet: End-to-End Speech Processing Toolkit. In Proceedings of the Interspeech, Hyderabad, India, 2–6 September 2018; pp. 2207–2211. [[CrossRef](#)]
57. Kuchaiev, O.; Li, J.; Nguyen, H.; Hrinchuk, O.; Leary, R.; Ginsburg, B.; Krizan, S.; Beliaev, S.; Lavrukhin, V.; Cook, J.; et al. Nemo: A toolkit for building ai applications using neural modules. *arXiv* **2019**, arXiv:1909.09577.
58. Zhang, B.; Wu, D.; Peng, Z.; Song, X.; Yao, Z.; Lv, H.; Xie, L.; Yang, C.; Pan, F.; Niu, J. Wenet 2.0: More productive end-to-end speech recognition toolkit. *arXiv* **2022**, arXiv:2203.15455.
59. Hall, K.; Cho, E.; Allauzen, C.; Beaufays, F.; Coccaro, N.; Nakajima, K.; Riley, M.; Roark, B.; Rybach, D.; Zhang, L. Composition-based on-the-fly rescoring for salient n-gram biasing. In Proceedings of the Interspeech, Dresden, Germany, 6–10 September 2015; pp. 1418–1422.

60. Aleksic, P.; Ghodsi, M.; Michaely, A.; Allauzen, C.; Hall, K.; Roark, B.; Rybach, D.; Moreno, P. Bringing Contextual Information to Google Speech Recognition. 2015. Available online: <https://research.google/pubs/pub43819/> (accessed on 4 September 2023).
61. Serrino, J.; Velikovich, L.; Aleksic, P.S.; Allauzen, C. Contextual Recovery of Out-of-Lattice Named Entities in Automatic Speech Recognition. In Proceedings of the Interspeech, Graz, Austria, 15–19 September 2019; pp. 3830–3834.
62. Chen, Z.; Jain, M.; Wang, Y.; Seltzer, M.L.; Fuegen, C. End-to-end contextual speech recognition using class language models and a token passing decoder. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 6186–6190.
63. Yadav, V.; Bethard, S. A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, 20–26 August 2018; pp. 2145–2158.
64. Sharma, A.; Chakraborty, S.; Kumar, S. Named Entity Recognition in Natural Language Processing: A Systematic Review. In *Proceedings of the Second Doctoral Symposium on Computational Intelligence*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 817–828.
65. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537.
66. Birjali, M.; Kasri, M.; Beni-Hssane, A. A comprehensive survey on sentiment analysis: Approaches, challenges and trends. *Knowl.-Based Syst.* **2021**, *226*, 107134. [[CrossRef](#)]
67. Qiao, B.; Zou, Z.; Huang, Y.; Fang, K.; Zhu, X.; Chen, Y. A joint model for entity and relation extraction based on BERT. *Neural Comput. Appl.* **2022**, *34*, 3471–3481. [[CrossRef](#)]
68. Zaib, M.; Zhang, W.E.; Sheng, Q.Z.; Mahmood, A.; Zhang, Y. Conversational question answering: A survey. *Knowl. Inf. Syst.* **2022**, *64*, 3151–3195. [[CrossRef](#)]
69. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762
70. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2018**, arXiv:1810.04805.
71. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
72. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*; Association for Computational Linguistics: Berlin, Germany, 2020; pp. 38–45.
73. Lhoest, Q.; del Moral, A.V.; Jernite, Y.; Thakur, A.; von Platen, P.; Patil, S.; Chaumond, J.; Drame, M.; Plu, J.; Tunstall, L.; et al. Datasets: A Community Library for Natural Language Processing. *arXiv* **2021**, arXiv:2109.02846.
74. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
75. Hendrycks, D.; Gimpel, K. Gaussian error linear units (gelus). *arXiv* **2016**, arXiv:1606.08415.
76. Pascanu, R.; Mikolov, T.; Bengio, Y. On the difficulty of training recurrent neural networks. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 1310–1318.
77. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
78. He, Z.; Wang, Z.; Wei, W.; Feng, S.; Mao, X.; Jiang, S. A Survey on Recent Advances in Sequence Labeling from Deep Learning Models. *arXiv* **2020**, arXiv:2011.06727.
79. Zhou, C.; Cule, B.; Goethals, B. Pattern based sequence classification. *IEEE Trans. Knowl. Data Eng.* **2015**, *28*, 1285–1298. [[CrossRef](#)]
80. He, P.; Gao, J.; Chen, W. Deberv3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv* **2021**, arXiv:2111.09543.
81. Zuluaga-Gomez, J.; Prasad, A.; Nigmatulina, I.; Motlicek, P.; Kleinert, M. A Virtual Simulation-Pilot Agent for Training of Air Traffic Controllers. *Aerospace* **2023**, *10*, 490. [[CrossRef](#)]
82. Madikeri, S.; Boudlard, H. Filterbank slope based features for speaker diarization. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 111–115.
83. Sell, G.; Snyder, D.; McCree, A.; Garcia-Romero, D.; Villalba, J.; Maciejewski, M.; Manohar, V.; Dehak, N.; Povey, D.; Watanabe, S.; et al. Diarization is Hard: Some Experiences and Lessons Learned for the JHU Team in the Inaugural DIHARD Challenge. In Proceedings of the Interspeech, Hyderabad, India, 2–6 September 2018; pp. 2808–2812.
84. Landini, F.; Profant, J.; Diez, M.; Burget, L. Bayesian HMM clustering of x-vector sequences (VBx) in speaker diarization: Theory, implementation and analysis on standard tasks. *Comput. Speech Lang.* **2022**, *71*, 101254. [[CrossRef](#)]
85. Fujita, Y.; Watanabe, S.; Horiguchi, S.; Xue, Y.; Nagamatsu, K. End-to-end neural diarization: Reformulating speaker diarization as simple multi-label classification. *arXiv* **2020**, arXiv:2003.02966.
86. Fujita, Y.; Kanda, N.; Horiguchi, S.; Xue, Y.; Nagamatsu, K.; Watanabe, S. End-to-end neural speaker diarization with self-attention. In Proceedings of the 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Singapore, 14–18 December 2019; pp. 296–303.
87. Fujita, Y.; Kanda, N.; Horiguchi, S.; Nagamatsu, K.; Watanabe, S. End-to-end neural speaker diarization with permutation-free objectives. *arXiv* **2019**, arXiv:1909.05952.

88. Ryant, N.; Church, K.; Cieri, C.; Cristia, A.; Du, J.; Ganapathy, S.; Liberman, M. The Second DIHARD Diarization Challenge: Dataset, Task, and Baselines. In Proceedings of the Interspeech, Graz, Austria, 15–19 September 2019; pp. 978–982.
89. Ariav, I.; Cohen, I. An end-to-end multimodal voice activity detection using wavenet encoder and residual networks. *IEEE J. Sel. Top. Signal Process.* **2019**, *13*, 265–274. [[CrossRef](#)]
90. Lin, Y. Spoken Instruction Understanding in Air Traffic Control: Challenge, Technique, and Application. *Aerospace* **2021**, *8*, 65. [[CrossRef](#)]
91. Cardoso, K.M. *An Analysis of en Route Controller-Pilot Voice Communications*; NASA STI/Recon Technical Report N; Federal Aviation Administration: Washington, DC, USA, 1993; Volume 93, p. 30611.
92. Prasad, A.; Zuluaga-Gomez, J.; Motlicek, P.; Sarfjoo, S.; Nigmatulina, I.; Ohneiser, O.; Helmke, H. Grammar Based Speaker Role Identification for Air Traffic Control Speech Recognition. *arXiv* **2021**, arXiv:2108.12175.
93. Helmke, H.; Ondřej, K.; Shetty, S.; Arilfusson, H.; Simiganoschi, T.; Kleinert, M.; Ohneiser, O.; Ehr, H.; Zuluaga-Gomez, J.; Smrz, P. Readback Error Detection by Automatic Speech Recognition and Understanding—Results of HAAWAI Project for Isavia’s Enroute Airspace. In Proceedings of the 12th SESAR Innovation Days. Sesar Joint Undertaking, Budapest, Hungary, 5–8 December 2022.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.