Issue 24(4), 2024, pp. 133-153
ISSN: 1567-7141
TU Delft OPEN

EJTIR

SI: Sesar Innovation Days 2023
https://ejtir.tudelft.nl
DOI: 10.59490/ejtir.2024.24.4.7531

Minimum effort adaptation of automatic speech recognition system in air traffic management

Mrinmoy Bhattacharjee[1], Petr Motlicek[2], Srikanth Madikeri[3], Hartmut Helmke[4], Oliver Ohneiser[4], Matthias Kleinert[4], Heiko Ehr[4]
[1] corresponding author mrin.mb@gmail.com, Idiap Research Institute, Martigny, Switzerland; 0000-0002-5348-9378
[2] Idiap Research Institute, Martigny, Switzerland; Brno University of Technology, Czech Republic; 0000-0001-6467-1119
[3] Idiap Research Institute, Martigny, Switzerland;
[4] Institute of Flight Guidance, German Aerospace Center (DLR), Braunschweig, Germany;

Keywords
Speech Recognition
Model Adaptation
Integration of prior knowledge
Customization of models
Rare-word integration

Publishing history
Submitted: 26 April 2024
Revised date(s): 04 September 2024, 15 October 2024
Accepted: 28 October 2024
Published: 31 December 2024

Cite as
Bhattacharjee, M., et al., (2024). Minimum effort adaptation of automatic speech recognition system in air traffic management. *European Journal of Transport and Infrastructure Research, 24*(4), 133-153.

Abstract

Advancements in Automatic Speech Recognition (ASR) technology is exemplified by ubiquitous voice assistants such as Siri and Alexa. Researchers have been exploring the application of ASR for Air Traffic Management (ATM) systems. Initial prototypes utilized ASR to pre-fill aircraft radar labels and achieved a technological readiness level before industrialization (TRL6). However, accurately recognizing infrequently used but highly informative domain-specific vocabulary is still an issue. This includes waypoint names specific to each airspace region and unique airline designators, e.g., "DEXON" or "POBEDA". Traditionally, open-source ASR toolkits or large pre-trained models require substantial domain-specific transcribed speech data to adapt to specialized vocabularies. However, typically, a "universal" ASR engine capable of reliably recognizing a core dictionary of several hundreds of frequently used words suffices for ATM applications. The challenge lies in dynamically integrating the additional region-specific words used less frequently. These uncommon words are crucial for maintaining clear communication within the ATM environment. This paper proposes a novel approach that facilitates the dynamic integration of these new and specific word entities into the existing universal ASR system. This paves the way for "plug-and-play" customization with minimal expert intervention and eliminates the need for extensive fine-tuning of the universal ASR model. The proposed approach demonstrably improves the accuracy of these region-specific words by a factor of ≈*7* (from *10%* F1-score to *70%*) for all rare words and ≈*5* (from *13%* F1-score to *64%*) for waypoints.

1 Introduction

A joint study by DLR and MITRE examined millions of words spoken by Air Traffic Controllers (ATCos) and pilots in radio telephony. It was observed that the ten digits (zero to nine) make up a whopping *40%* of all spoken words [1]. In the US alone, *550* different words account for *95%* of the radio communication. However, the complete dictionary used by ATCos and pilots is significantly larger, reaching tens of thousands of words. This includes nearly *10,000* airline designators (such as "Speedbird", "Air France", or "Ocean") and an even greater number of artificial waypoints, fixes, and navigation aid names (e.g., "DEXON", "MOBSA", "DOMUX") used globally across airports and airspaces. Further issues are that these vast word lists require frequent updates (often monthly) to incorporate some new terms. This constant evolution can lead to performance degradation in speech recognition and related processing systems over time. Maintaining these extensive and dynamic vocabularies presents a significant lifecycle challenge, especially when scaling applications to encompass multiple Air Traffic Control (ATC) sectors and facilities.

Early attempts at integrating Automatic Speech Recognition (ASR) into Air Traffic Management (ATM) focused on replacing simulation pilots with ASR systems [2]. These initial applications were used by Air Navigation Service Providers (ANSPs). These applications simulated a limited number of airspaces with static waypoint sets. However, the required waypoints change when simulating different airspaces (e.g., Frankfurt vs. Heathrow approach).

1.1 Problem

The DIAL[1] project [3] aims to develop a digital assistant for ATCos. This assistant handles routine tasks for less complex aircraft, thereby freeing up ATCos' cognitive resources so they can focus on more demanding tasks. Additionally, DIAL utilizes a complex recognition and understanding solution to reduce the number of simulation pilots needed per training speech exercise. Originally, the ASR engine was developed by Idiap for the Vienna approach area. This ASR system relied on *260* waypoints such as "ABIRI" or "BALAD". However, DIAL considers the Celle sector in upper airspace, usually controlled by Maastricht upper airspace center (UAC), as relevant airspace [4]. Therefore, *565* different waypoints were required to be well recognized by ASR for the new use-case in DIAL, e.g., "KOSEK", "WYK", or "DOR" ("DOR" is pronounced as "WICKEDE" being a village in western part of Germany and "WYK" is pronounced as "WIPPER"). Contrary to this, "KOSEK" for the Celle sector or "ABIRI" for the Vienna approach are artificial five-letter words composed of vowels and consonants. Others like "OSNABRÜCK" (German) or "LIÈGE" (French) have pronunciation complexities due to non-English letters or regional accents. These waypoints are typically absent from the ASR's training data, leading to recognition issues. While some pronunciations can be inferred from spelling, others are highly context-dependent on the ATCos' regional dialect. Previous trials in DLR's labs using a universal ASR solution designed for the Vienna approach achieved a Word Error Rate (WER) of *3.1%* on *120,000* words [5]. However, when applying this ASR solution to the DIAL project's initial trials, we have observed a more than four-fold increase in WER (refer to Table 3). This performance drop is directly linked to the presence of spoken waypoint names unseen during the ASR training [6]. From now on, this issue will be referred to as Out-Of-Training-Data (OOTD) words. The language modelling system that relies on context to generate transcripts struggles in such scenarios and makes additional errors. This means that poorly recognized waypoints also lead to errors in nearby words (i.e., contextualisation). With availability of data for model adaptation, previous works on adapting ASR systems to better recognize rare words have shown impressive results [7] [8]. The use of ASR to detect spoken terms has also been studied a lot in the past (e.g., in our work from 2010 [9], including multilingual data). However, the problem was typically solved by inserting rare words

[1] The project "Individual and Automated Air Transport" (Der Individuelle und Automatisierte Luftverkehr; DIAL), initiated by the DLR, bundles research to expand the automation of air traffic.

directly to training, lexicon extension, or indirectly through model adaptation, which were not feasible approaches for ATC (e.g., waypoints are unique for each airspace and thus the model would need to be re-trained/adapted for each new airspace). Hence, this paper proposes an appealing solution to address this specific issue of recognising OOTD words.

The challenge addressed by this paper is related to the contextual mismatch of test data concerning training data, specifically for the incorporation of waypoint (word) entities. Waypoints are specialized terms in the field of ATC, corresponding to coordinates. ATCos and pilots use these terms to follow or adjust flight paths. Typically, these waypoint terms are specific to particular areas. For example, if we focus on a specific domain, like enroute navigation for Germany, the waypoints are tailored to that region. However, if the German ASR system were to be applied for enroute navigation in Switzerland, the waypoints used would be different. Consequently, the system might not readily recognize them. In this context, our current work empowers customers to easily incorporate these new waypoints into the system, even without specialized expertise. The OOTD issue differs from the known Out-Of-Vocabulary (OOV) problem addressed in the past [10]. OOV words are often artificial words not part of the ASR dictionary. Our OOTD words are typically pre-defined and exist in the ATC dictionary. However, they can have more pronunciation variants than those in the standard dictionary. Moreover, OOTDs are not seen during training the components of the conventional ASR system. A conventional ASR system is composed of a separate acoustic model that learns to map speech to distinct sound units (see subsection 2.2 and a language model that models the context in which different words can appear in an utterance (see subsection 2.6).

1.2 Paper structure

This paper presents the challenge of recognizing ATC-specific spoken words that are rarely found in training data. Section 2 reviews existing research addressing this issue. It also offers a concise overview of the current state of the art in ASR for ATC. Following this, Section 3 details the experimental setup used to evaluate the recognition performance of OOTD words after customization. Section 4 presents the core of the paper and describes various solutions implemented within the DIAL project to tackle the OOTD problem. Section 5 presents the results achieved, analysing performance at both the word and semantic levels. Finally, Section 6 concludes the paper by summarizing the key takeaways.

2 Current State-of-the-art ASR Solution for ATC

2.1 Automatic Speech Recognition (ASR)

ASR, also often referred to as a speech-to-text system, automatically converts the input speech to a textual form, i.e., a sequence of words. In the case of ATC communication, we refer to the voice conversation captured by microphones on the side of ATCos or pilots with input speech. The most advanced ASR technology developed in the recent past for ASR for ATM applications comes from HAAWAII project [11] (HAAWAII = Highly Advanced Air Traffic Controller Workstation with Artificial Intelligence Integration[2]). The project needed to focus not only on developing innovative ASR approaches in ATM but also on reaching a certain level of maturity so that the developed ASR solution can provide sufficient recognition performance to be useful for subsequent downstream applications, e.g., callsign highlighting [12], pre-filling radar label entries [5], or readback error detection [13]. Furthermore, we have decided to re-use a conventional ASR solution trained on relatively large manually transcribed speech data available from the HAAWAII project and other past projects.

[2] HAAWAII Project: https://www.haawaii.de/wp/

As shown in Figure 1 the ASR engine consists of a combination of independently trained Acoustic Models (AM) and Language Models (LM). This work trains the acoustic model as a classical Deep Neural Network instead of using new end-to-end architectures [14]. A conventional ASR system employs separate AM and LM. The AM is trained with speech recordings with a corresponding text transcript. On the other hand, LM is trained on text data only; for example, the text corresponding to the speech recordings available for training AM is typically used. However, much larger textual resources are generally available than speech data. The AM represents the relationship between a speech signal and phonemes or other linguistic units that make up the speech. The LM is usually represented by a probability distribution over sequences of words. The LM in the form of a Finite State Transducer (FST) provides context to distinguish between words and phrases that sound similar. Using the knowledge of AM and LM, a decoding graph is usually built as a Weighted Finite State Transducer (WFST) [15], [16] using the open-source library called OpenFST [17]. The WFST graph generates text output given an observation sequence as shown in Figure 1. A decoder module uses the decoding graph to predict the best probable transcript corresponding to an input speech signal.

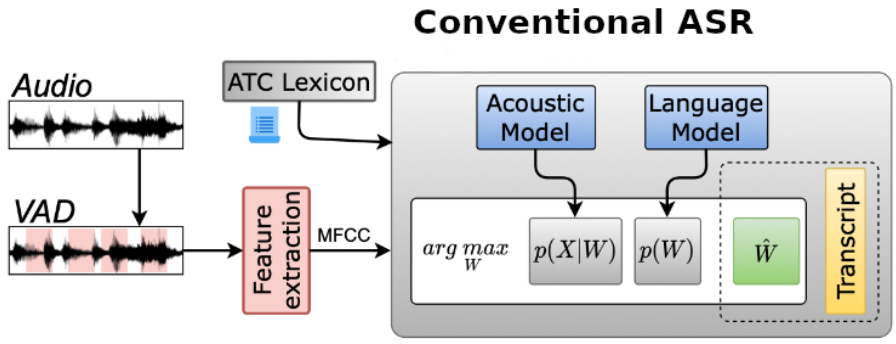

Figure 1. A conventional ASR system, where acoustic model combines hidden Markov models and modern deep neural network architectures. The language model is then used during the decoding phase. Here, VAD stands for the pre-processing step of Voice Activity Detection. MFCC stands for Mel-Frequency Cepstral Coefficients, a popular feature representation in speech processing.

2.2 *Acoustic Modelling (AM)*

The acoustic model in conventional ASR is built around Hidden Markov Models (HMMs) combined with Deep Neural Networks (DNNs) [18]. DNNs effectively estimate the posterior probability of a given set of phonemes called tri-phones, specifically, context-dependent phonemes. These posterior probabilities can be seen as pseudo-likelihoods or "scaled likelihoods", which can be interfaced with HMM modules. HMMs provide a structure for mapping a temporal sequence of acoustic features extracted from the input speech, e.g., Mel Frequency Cepstral Coefficients (MFCC), to a sequence of states [19]. End-to-end models have become popular in recent years as they can be trained as non-autoregressive systems that can model the long future context during training. Nevertheless, conventional ASR systems remain one of the best approaches for building ASR engines, allowing them to reach high recognition accuracies. An HMM/DNN based ASR is still considered as the state-of-the-art solution for ATC domain. It was also used in the HAAWAII project [13] (and already partially tested in preceding MALORCA project) as well as in DIAL project.

The acoustic model in those projects was typically trained in a supervised mode, i.e., manually transcribed data is required for the target ATC domain. Some of the recent works have shown further improvements with semi-supervised model training. Such training relies on exploiting automatically labelled speech, i.e., using some universal ASR engine or an ASR engine developed using a small amount of manually transcribed speech recordings. More details on leveraging non-transcribed ATC speech data by semi-supervised learning can be found in [20], [21]

where authors extract semantic knowledge and [22], which aimed to improve callsign recognition performance. An advantage of semi-supervised learning is that a large set of unlabelled speech data is easily available and can be employed for training. Large here means 10 or 100 times more than in the case of manually transcribed data. One of the sources of reliable large-scale collection of ATC speech data from different airports worldwide is available from the ATCO2 project [23]. Additionally, innovative research to improve word recognition belonging to spoken callsigns is possible by integrating surveillance data into the pipeline [22], [24]. As promising as this line of work may seem, self-supervised model training cannot solve the problem targeted in this work.

In order to train the conventional ASR, additional knowledge is required for its development. In the case of a monolingual ASR system (English in this case), the minimum knowledge relevant for ATM is a set of phonemes and an input dictionary. Both play an important role in further model customization. The ATC dictionary consists of words that usually do not appear in English conversations but are specific to ATC. On the one hand, there are frequently used and static terms such as "QNH" and "WILCO". On the other hand, there are seldom used and dynamically changing words (such as artificial waypoint names). Such words may have never been seen during the training of the ASR model but might occur in the field when the ASR system is deployed. This work proposes solutions to easily customize the running ASR system to better detect such words in the field.

This work considers two types of AM for a rigorous analysis of the proposed boosting methods. The first model is known as Convolutional Neural Network (CNN) - Time-Delay Neural Network (TDNN) models [25]. The second type of AM used in this work is the Cross-lingual Representation Learning for Speech Recognition (XLSR) based model [26] (see subsection 2.4 for more details). The two acoustic models are described in detail below.

2.3 CNN-TDNN model

The CNN-TDNN model is relatively small compared to the current trends of using large deep-learning models. CNN-TDNN can be directly trained on the target data relatively quickly, thus allowing it to be used in diverse setups with limited computational resources. CNN-TDNN type architecture is deployed in Kaldi and uses the Lattice-Free Maximum Mutual Information (LF-MMI) training criterion. LF-MMI is implemented as a sequence discriminative training, wherein each sequence (typically an utterance of speech) is evaluated by two values: the numerator, which computes the probability of the observation given the ground truth and the denominator, which computes the probability over all possible sequences. LF-MMI is considered a state-of-the-art criterion for hybrid ASR. CNN-TDNN models [25] are a type of artificial neural network specifically designed to handle sequential data. By incorporating time delays in their architecture, TDNNs excel at capturing long-range dependencies within data, making them particularly effective for tasks like speech recognition. Unlike traditional feed-forward neural networks, CNN-TDNNs process input data in a variable order, allowing them to model complex temporal patterns.

2.4 XLSR-based model

The second AM considered in this work is the XLSR-based model [27]. XLSR is a large pre-trained end-to-end model (i.e., encoder) trained on large amounts of multilingual data without supervision. The XLSR-based encoder is then extended with Factorized Time-Delay Neural Network (TDNNF) layers [26], and the whole architecture is further trained (or fine-tuned) on target domain data. The XLSR model [27] is a multilingual speech recognition model that leverages self-supervised learning to learn general-purpose speech representations. Built upon the wav2vec *2.0* architecture [28], XLSR is trained on massive amounts (≈*56*k hours) of unlabelled speech data from *53* languages by using LF-MMI training criterion. Such a huge training corpus enables the model to capture shared acoustic patterns across linguistic contexts. This cross-lingual pre-training significantly boosts fine-tuned performance on specific speech recognition tasks, especially in low-resource language settings like the ATC domain.

2.5 Streaming XLSR model

Recently, the streaming solution for the XLSR model was developed to address real-time processing requirements (including those for ATC). Whereas the XLSR LF-MMI model operates in an offline fashion (i.e., the decoding process commences only when the entire audio is available), the streaming model waits only for a limited amount of audio (e.g., a few hundreds of milliseconds to 3 seconds) to produce transcripts. There is a significant loss when using models trained with full attention in streaming mode [29]. This loss in performance can be minimized by training the models with causal attention.

More specifically, the online decoder outputs three types of transcripts: partial, endpoint, and final. Given the current decoding state, a partial output is the best scoring word sequence. An endpoint refers to one of several conditions of the state of the decoder (e.g., end of word, short silence, long silence, long utterance length, etc.). When one of such conditions is reached, the endpoint output is generated. The final output is available once the entire audio has been processed. In this work, the streaming model starts producing transcripts when it receives at least *1.2*s of audio. During the generation of each transcript (partial or otherwise), the full left context is used along with the current chunk.

The XLSR LF-MMI model is compiled with PyTorch's torchscript to be loaded in Kaldi. PyTorch's C++ Application Programming Interface (API) was integrated with Kaldi. Special hardware known as Graphics Processing Units (GPU) are used in machine learning to significantly speed up the running time of models. However, standard implementations of the systems need to be modified to run them on GPUs. For our needs, the online GPU decoder for LF-MMI models was extended to support the XLSR LF-MMI models. The support for offline GPU decoding is available as an added benefit of the implementation.

2.6 Language Modelling (LM), Dictionary

As part of conventional ASR, LM still plays a crucial role [30]. The main advantage of deploying LM is its large power to bring the generic ASR technology to the target, i.e., the ATC domain. Standard conventional ASR approaches still rely on word-based dictionaries, as is the case of the ASR solution developed for HAAWAII and DIAL projects. This paper utilizes a word-based LM trained directly on transcripts from the ATC domain. We chose a trigram LM (n-gram with n=3). However, word-based LMs struggle with words the ASR engine has never encountered. The ASR needs a pre-defined dictionary containing known words. This dictionary can be quite large, ranging from thousands of words for ATC to hundreds of thousands for general English conversations. In subsequent sections, we discuss a customization step to tackle the issue of rarely seen words during LM training (see section 4).

2.7 Recognition process

The process of recognition, i.e., generating the recognition output from the input speech, is briefly described here. Trained AM and LM are combined with the conventional-ASR solution applied for HAAWAII[3] and DIAL data. These models are combined using the concept of FST leveraging the Kaldi framework [31], one of the main toolkits researchers and companies use for ASR. Both the acoustic models as described previously (see section 2.3 and section 2.4) are used in the Kaldi framework along with a language model trained from the corresponding text data. It is to be noted that the XLSR model is not used in an end-to-end framework, as is done with the wav2vec *2.0* [28] model. Moreover, Kaldi solutions for decoding streaming speech data are also employed in this work[4]. The true power of the XLSR model is derived from the Transformer layers in its architecture. The Transformer layers use a mechanism called attention that helps them learn

[3] Project website: Check footnote on page 3
[4] Kaldi streaming decoding: https://kaldi-asr.org/doc/online_programs.html

the temporal dependencies of the data. The transformer layers in the standard XLSR model attend to both past and future content while training. However, in a streaming setup, future content will not be available. Hence, we employ a masking mechanism to hide future content from the model during training. Thereby, the model can work even as a true streaming model.

The trained AM and LM are composed as FSTs together with a dictionary, and the final decoding graph is used through the process called "decoding". During decoding, the input speech is first used to extract speech features (MFCC mentioned above), which are then inferred through the Deep Neural Network architecture. Using the decoding graph, the output set of phonemes represented by a set of posterior probabilities is passed to the decoder to map the phoneme sequence to the most likely word sequence. The output can then be seen as a set of word recognition hypotheses, i.e., word sequences represented by lattice data structures. The lattices carry not only information about word sequences but also information about confidence for each word. Decoding can be run in an offline mode, i.e., after detecting the endpoint in utterance, the speech is decoded, and the recognized word sequence is returned. Meanwhile, in online decoding mode, partial word recognition is available in real-time during decoding with a minimum latency of *200-300*ms.

3 Experimental setup

3.1 Training data

The data for training the AM for the DIAL project is partially leveraged from the past works. This dataset stems from an exploratory research initiative to investigate and create a dependable and flexible system for automatically transcribing voice commands provided by ATCos and pilots alike. The dialogues between ATCos and pilots were sourced from two ANSPs: (i) NATS for the London approach and (ii) ISAVIA for Icelandic enroute. For training the acoustic models, a total of *195* hours of labelled ATC data have been used [14]. The training data is augmented with other internal ATC databases and also using speed perturbation during training. The corresponding reference text has been used to train the baseline language model. The language model is a tri-gram model trained using the SRILM toolkit in Kaldi [32].

3.2 Test data

The testing data for evaluating the ASR system was collected through proof-of-concept exercises involving ATC utterances from ATCos to pilots. The data was collected at DLR, Germany. Despite the diverse English accents of the speakers, the recording conditions were generally clean. Furthermore, the exercises included spoken words not often encountered during training. Additionally, utterances that especially contain rare words, e.g., waypoints, in the correct context were recorded by different speakers, e.g., "AIR FRANCE TWO SIX ALFA PROCEED HAMM". The test set consists of approximately *52* minutes of audio data with a total of *673* test utterances, comprising a total of *1157* commands like CONTACT, CONTACT_FREQUENCY, DIRECT_TO, etc., containing rare words. According to the annotation ontology [1], a "command" is a high-level concept that represents an ATC instruction. The number of waypoints considered for boosting in this work is discussed in the next subsection.

Waypoints are names given to a latitude-longitude pair representing a geographic location. A waypoint is represented by an abbreviation consisting of a sequence of letters and numbers like "DL455" or "WYK". These waypoint names may appear in ATCo-pilot communication spelled by the ICAO alphabet as "DELTA LIMA FOUR FIVE FIVE" for "DL455" or "WHISKEY YANKEE KILO" for "WYK". "WYK" is not just an artificial geographic location but represents the river "WIPPER" in Germany. Therefore, "WYK" can be spoken as mentioned by the ICAO alphabet or just "WIPPER". The waypoint point "MOBSA" is again a completely artificial name, which can be spoken as "MOBSA" or "MIKE OSCAR BRAVO SIERRA ALFA".

When the waypoint is referred to by pronouncing its sequence of letters, the ASR system can easily detect it, as the ICAO phonetic alphabet is commonly encountered as part of the English

dictionary during training. However, challenges arise when ATCos or pilots use artificially created waypoint names like "WIPPER", either newly introduced or infrequently encountered during model training. The discussion on waypoints cannot be complete without mentioning that the baseline ASR dictionary contains many waypoints from different sectors. However, only a specific set of waypoints is important when deploying an ASR system for a particular sector. The remaining "unimportant" waypoints in the dictionary are unwanted for this particular use case. Therefore, we define such waypoints as non-target or old waypoints.

Table 1. Statistics related to the Word Boosting Task

Test set size	*673* utterances / *52* minutes audio
Dictionary size	*30,821*
Number of valid Waypoints in the sector	*565*
Number of Waypoints in the test set	*84*
Number of OOTD Waypoints in the test set	*12*
Total occurrences of Waypoints	*443*

Table 1 lists important statistics about the test data and the waypoints selected for boosting in the present case. The DIAL database consists of a list of *565* rare words that were required to be detected correctly by the ASR system. From this list of rare words, *84* unique word entities were present in the test set collected for evaluating the ASR system, and these occurred for a total of *443* times. However, a fraction of the rare words was never seen during training the LM and can be referred to as OOTD words. Therefore, in order to correctly evaluate the performance of the proposed boosting methods in improving the detection of the *84* rare words in the test set, the set of OOTD words was manually added (see subsection 4.3) to the dictionary and the LM training data in the form of synthetic data created for augmentation. The synthetic data is generated by replicating utterances for the most frequent waypoint in the data by replacing the actual waypoint with the target waypoints, including OOTD. This step is important since the proposed customization methods assume the words to be boosted are in the dictionary. Nevertheless, this step is easily performed with the tools developed in this work. Once all the *84* words are known to the ASR engine, we use the boosting methods presented in this work to improve the weights of all the *84* words, including contact frequencies like "MAASTRICHT" and waypoints.

It must be noted that tens of thousands of different waypoints are used across the ATC sectors worldwide. Only a specific set of waypoints is important for any airspace, while the others can be discarded. However, this is not as simple as it appears. The universal ASR system was trained using the ATC data that was available at that time. The training data was collected from diverse airspaces containing different sets of waypoints used in those sectors. Hence, the trained universal ASR system knows about a set of waypoints in the training data. Subsequently, such waypoints will be referred to as Old Waypoints (OW) until otherwise mentioned. With this training and testing data background, we will present the baseline performance in the following subsection.

3.3 Baseline ASR performance

In this subsection, the performance of the baseline ASR system is described. Originally, an ASR system trained for different airport/airspace scenarios was developed. However, when the same system was evaluated on the test data, it was observed that the important waypoints and other airport-dependent names were not properly recognized. The required customization of this ASR system forms the crux of this work. However, before presenting the results obtained from the model customization, it is necessary to discuss the results for the unmodified system that will serve as the baseline for this work.

We use four metrics to report the results. The WER measures the percentage of erroneous insertions, substitutions, and deletions caused by the ASR system concerning the total number of reference words in the test data. This metric covers all words and not only the performance of the rare words. A lower WER is preferred. Next, we report the precision, recall, and F1-score of detecting the rare words in the test data. These metrics are computed using standard recognition statistics in Table 2. For validating or falsifying the hypotheses of the last subsection, we use the metrics introduced in [33] resulting in a simple scheme for measuring performance on a semantic level. It is independent of semantic concept type or sub-components and treats all semantic components equally.

Table 2. Definition of basic Metric elements

Name	Definition
TP: True Positive	Total number of True Positives: The concept (a sequence of words indicating a high-level command) is present and correctly and completely detected (including all subcomponents). In the case of Waypoints, the target waypoint is correctly detected.
FP: False Positive	Total number of False Positives: The concept is incorrectly detected, i.e., either the concept is absent, or one or more subcomponents are incorrectly detected. In the case of Waypoints, a waypoint is detected in place of another.
TN: True Negative	Total number of True Negatives: The concept is not correctly detected because it is not present. In the case of Waypoints, none is detected if there aren't any.
FN: False Negative	Total number of False Negatives: A concept is not detected when it should have been. In the case of Waypoints, the target waypoint is not detected.
TA: Total	Total number of annotated concepts, i.e., ground truth concepts that are sometimes referred to as gold-standard concepts.

The metric can be used to measure the performance of multiple tasks like command extraction, callsign extraction, waypoint extraction, etc. In this work, the metrics are used to report the performance of waypoint and command extraction. Table 2 lists definitions that are the building blocks for the performance metrics. From the five building blocks, viz. TP, FP, TN, FN, and TA (see Table 2), we can derive accuracy using equation (1). Additionally, in equation (3) the F1-Scores by defining Recall and Precision in equation (2).

$$Accuracy = \frac{TP+TN}{TA} \tag{1}$$

$$Recall = \frac{TP}{TP+FN}; Precision = \frac{TP}{TP+FP} \tag{2}$$

$$\text{F1 } Score = \frac{2*Recall*Precision}{Recall+Precision} \tag{3}$$

This work considers the CNN-TDNN model as the baseline system. The XLSR and XLSR-streaming models extend the baseline system that works in an End-to-End (E2E) fashion. The word-boosting methods proposed in this work are applied only to the baseline CNN-TDNN model. Exploring the effects of the boosting methods on the E2E systems will be considered in the future as an extension of this work. As observed from the F1-score in Table 3, the baseline ASR system fares quite poorly in detecting the rare words. Both performance values, WER and recall, which are most important for waypoint recognition, are poor. Recall of ≈*5%* roughly means that *95* (of *100*) waypoints are substituted or deleted. Similarly, a precision of ≈*5%* would mean that out of *100* times a waypoint is predicted, only *5* is predicted at the correct position. The baseline system has good precision but extremely poor recall, leading to an overall F1-score of *13%*. Also, WER above *10%* is significantly higher than observed on data such as those from the HAAWAII project (*3.1%* from Table 1 in [5]). In section 4, we discuss the proposed customization algorithms applied to the baseline ASR system to improve the performance of the rare words.

Table 3. Performance of the Baseline ASR systems

AM	WER (%)	Waypoint Detection		
		Precision (%)	Recall (%)	F1-Score (%)
CNN-TDNN	13.85	100	7	13
XLSR	8.53	95	45	61
XLSR-streaming	12.65	99	27	43

3.4 Baseline performance on semantic level

Measuring the accuracy of ASR (typically done on the word level) is not directly related to the problem of speech understanding (i.e., extraction of information on a semantic level). As a complementary evaluation to ASR, we measure performance on a semantic level (i.e., understanding part by processing ASR output) using the metrics presented above considering the commonalities of speech recognition and understanding for ATC applications on both sides of the Atlantic [1]. The above data set in total consists of *1157* commands with *85* commands of type CONTACT, as shown for command types DIRECT_TO, STATION, and CLIMB in Table 4. The test set is the same as the one described in subsection 3.2. We did not show all command types, like HEADING, SPEED, etc.

Table 4. Semantic Performance of the Baseline system

Baseline	Number of commands	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
All Type	*1157*	*58*	*91*	*61*	*73*
CONTACT	*85*	*36*	*79*	*39*	*53*
DIRECT_TO	*139*	*5*	*47*	*6*	*10*
STATION	*170*	*13*	*85*	*13*	*23*
CLIMB	*90*	*90*	*95*	*94*	*94*

According to annotation ontology [1], CONTACT results from the utterance "CONTACT **BOERDE**". DIRECT_TO results from "PROCEED TO **NIENBURG**" and STATION is the semantic interpretation of "SPEED BIRD FOUR ONE **MAASTRICHT** GO AHEAD". The airspace-dependent words are marked in boldface. The mixture of command types is not representative of real-life utterances. We added many more airspace-dependent names to our test set. We see the bad performance with accuracy far below *50%* for the CONTACT, DIRECT_TO, and STATION commands, whereas CLIMB is much better. It consists of keywords, numbers, and units that are not airspace-dependent.

4 Customization of ASR for new Domains

This section describes the process of ASR adaptation or customization necessary to port the ASR engine to a new airport sector. The main concept targeted by our work is to minimize the requirements for expert knowledge, allowing target users to customize the ASR technology on their premises. Customization can be done in several ways, as described below.

4.1 AM and LM retraining using target-domain data

The most obvious way of adapting the ASR is to use the concept of model adaptation by applying data-driven approaches leveraging a set of speech and/or text transcripts available from the target domain. More specifically, the AM can be efficiently adapted to a new domain, e.g., to a target use-case airport/airspace, by exploiting some target data (speech aligned with available transcripts) from the domain to fine-tuning the model parameters [22]. In the case of LM, the n-gram interpolation of an original LM with transcripts available from the target domain can be used.

AM and LM adaptation processes usually require expert knowledge, which is usually not part of an ATM personnel's skill set. Thus, collaboration with the ASR developers is required. This also includes collecting speech data sets from target domains, which could require a certain amount of time. Additionally, manual data transcription, typically done by humans, can take enormous time. Furthermore, this process requires certain unit testing to ensure the new models perform well and the target models are not over-trained. Also, when porting the ASR system to a new domain, such as the DIAL project for the Celle sector, the set of new waypoints is expected to be a priori available.

4.2 Dynamic customization of existing LM

The second method involves end-users customizing existing LM to enhance the recognition of new words. In contrast to the method above, this paper presents an approach that enables incorporating new words or word entities into the existing ASR framework without requiring additional manual transcriptions or retraining of existing models. Kocour et al. [34] discuss the use of contextual information to improve the accuracy of call sign recognition where a priori information about the list of callsigns from radar data was obtained to correlate ASR output. Other works on contextual biasing for online ASR [35] and improving waypoint recognition [36] [6] employ boosting the target entities (waypoints in this work) to improve their recognition. The original ASR models are typically trained on huge data sets, but data at similar scales are unavailable during customization. Hence, this work tackles the problem in a more user-centric manner. The proposed methods do not require ASR retraining, nor do they require expert knowledge. Therefore, we have not delved further into the adaptation approach. Instead, we have concentrated on strategies for customizing models to recognize new waypoints within the Celle Sector for "enroute" positions. Customizing the model in this study involves two distinct approaches: the first is called G-Boosting, and the second is termed Lattice-Rescoring. Both of these approaches will be explained in the following subsections.

4.3 LM expansion using synthetic data

The first step of performing the proposed modifications to the LM involves expanding the existing LM using synthetically generated text. For the present use case of waypoints, synthetic texts are generated that correspond to the usage of such words in ATCo-Pilot communication. In order to perform this expansion, let us consider WP_{freq} the most frequent waypoint in the original LM training text. Also, let $T(WP_{freq})$ be the text data containing WP_{freq} in the training data. Let WP_{boost} be one of the waypoints to be boosted. To generate the synthetic text for WP_{boost}, WP_{freq} in $T(WP_{freq})$ is replaced with WP_{boost}. The text obtained by substituting WP_{boost} instead of WP_{freq} is added to the LM training corpus. This process is repeated for the remaining *84* rare words considered in this work (see Table 1). Once the LM training data is augmented with the synthetic data, a new LM is trained on the augmented data. This LM trained using additional synthetic text data is considered the baseline LM in this work and is kept constant in all the experiments.

4.4 G-Boosting

The G-Boosting approach modifies the trained FST graph that represents the LM for the ASR task [35] [6]. The trained LM is also stored as an FST (as mentioned previously). The LM is trained to learn the most likely sequences or the context in which a word appears in the data. The acoustic model information predicts many possible word sequences for a given test utterance during decoding. The LM provides the likelihood score for a predicted word sequence to be present in the

current context. When there are many possible word sequences to select for a particular utterance, the one obtaining the highest LM score is generally selected. Although this is a very logical method to train and use the LM, it has disadvantages. The LM, being a statistical model, gets biased to the most frequent of the words seen during training data. In other words, a word seen many times during training will get a higher LM score than a similar but less frequent word. Thus, in cases where the acoustic model is not very confident about the predicted word in a particular context, the LM would select the most frequent among the probable options. Our approach tries to solve this problem for the waypoints we are concerned about.

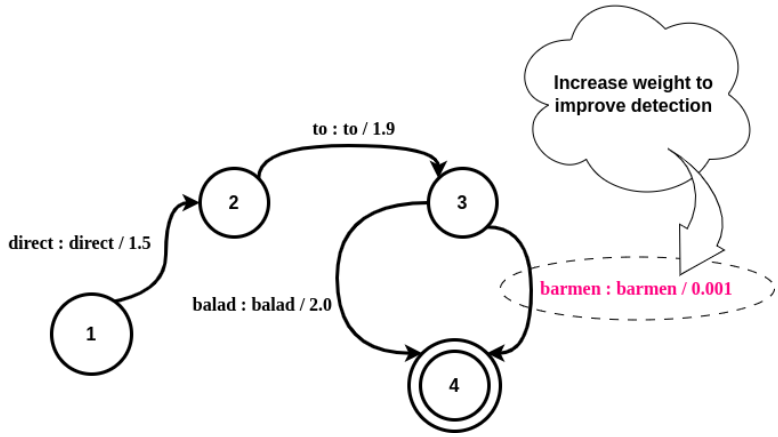

Figure 2. Word-boosting Operation in the Language Model.

Since the words are never seen or rarely seen during training, the final ASR output rarely predicts these words. To customize the ASR system to recognize these words, we update the weights in a pre-trained LM in a certain way so that the likelihood of the new words being predicted increases. Figure 2 graphically illustrates a toy example of the weight update step. For instance, if the word "BALAD" is more frequently seen during LM training, it will be associated with a higher weight than a less frequent "BARMEN". In such a scenario, to improve the detection of "BARMEN", we update the LM FST so that the word "BARMEN" also has a decent weight. The boosting factor is empirically decided based on the performance of a validation set. The LM FST discussed previously is a graph where all the correct sequence of words is represented as arcs of the graph with their respective weights. Our approach searches the arcs in the FST corresponding to each new word and artificially boosts the weights of the words under consideration. The modified LM is then used to create a new decoding FST graph to replace the previous one. Subsequently, when the decoding is performed again using the new decoding FST graph, the new words are detected much better than earlier (see Table 5). The best part of this approach is that there is no need for expert intervention to perform this operation. An ATM personnel member has to perform very basic steps, like preparing a list of waypoints to be boosted and passing them to a program that automatically performs the required modifications to the FST graph. G-Boosting performed quite well during experiments in improving the detection of rare words.

Table 5. Performance of the boosting techniques for conventional ASR

Method	WER (%)	Waypoint Detection		
		Precision (%)	Recall (%)	F1-Score (%)
CNN-TDNN	*13.85*	*100*	*7*	*13*
+G-Boosting	*9.55*	*93*	*55*	*69*
+Lattice-Rescoring	*14.04*	*93*	*9*	*17*

+G-Boosting + Lattice-Rescoring	*9.43*	*88*	*58*	*70*

4.5 Lattice-Rescoring

As the name suggests, the second approach of Lattice-Rescoring is performed on the decoded word recognition lattices (mentioned previously). The lattices are a data structure that stores the most likely decoded paths or word sequences for a given test utterance, along with their scores from the AM and the LM. In a normal decoding setup, the transcript for a given test utterance is the path that obtains the overall best score among all the possible sentences present in the corresponding lattice. As with new and rare words discussed previously, even if they are present in one of the possible paths, they usually correspond to low LM scores and, hence, do not get selected as the best path. We observed that there is a scope to improve the recognition of rare words by modifying the decoded lattices and re-scoring them. Specifically, we first create a small biased FST comprising rare words with boosted weights. Subsequently, a first-pass decoding using the baseline LM is performed to obtain the initial lattices. These lattices are then composed with the biased FST so that wherever the rare words are present in the decoded lattices, we update their corresponding LM scores with the boosted weight. After this operation, the modified lattice is used to compute the best path. The boost provided to the rare words' LM scores improves the chance of selecting the path that consists of the rare words. As with G-Boosting, we also avoid needing expert intervention to perform Lattice-Rescoring.

The main distinction between G-Boosting and Lattice-Rescoring is that G-Boosting can improve the recognition of OOTV words (when combined with language model expansion), while Lattice-Rescoring can only re-rank hypotheses but cannot add new words. Moreover, G-Boosting is a permanent modification of the LM, whereas Lattice-Rescoring runs without permanently modifying the LM. In our experiments, we found Lattice-Rescoring improved the detection of rare words (see Table 5).

5 Results after Boosting the Waypoints

This section discusses the results obtained using the proposed boosting methods. Subsection 5.1 presents the word-level performance. Subsection 5.2 discusses the effect of the context size used for the boosting. The effect of tuning the G-Boosting factor is discussed in subsection 5.3. We also show the effect of de-boosting the old waypoints in subsection 5.4. Finally, the gain in the semantic performance resulting from improving waypoint recognition is detailed in subsection 5.5.

5.1 Word level performance

The performance of the word-boosting techniques is tabulated in Table 5. It can be observed that the **G-Boosting** technique reduces the overall WER of the system from *13.85%* to *9.55%*. Regarding rare word detection, the baseline system detects *24* out of *443* occurrences, whereas the detection improves to *256* out of *443* after performing G-Boosting. Moreover, G-Boosting improves the precision, recall and F1-scores of detecting the rare words from *92%*, *5%*, and *10%* to *93%*, *55%*, and *69%*, respectively.

Lattice-Rescoring improves the precision, recall and F1-scores of detecting the rare words to *93%*, *9%*, and *17%* respectively. Also, combining G-Boosting and Lattice-Rescoring reduces the precision of detecting the waypoints. Another experiment worth trying is to apply Lattice-Rescoring on top of the G-Boosting method. Since the two methods are independent, the combination is pretty straightforward. The first-pass decoding required for Lattice-Rescoring can be performed using the G-Boosting method, followed by the second-pass re-scoring using the Lattice-Rescoring method. Combining the G-Boosting and Lattice-Rescoring methods reduces the overall WER to *9.43%*, while the rare word detection recall and F1-scores improve to *58%* and *70%*, respectively. Interestingly, the overall WER increases to *14.04%* with the Lattice-Rescoring approach, which is worse than the baseline. Such results indicate that the Lattice-Rescoring method introduces some

false negatives in the modified output. In other words, in rare cases, the combination of both methods causes over-boosting of the waypoints, so they may get detected in the wrong places in the speech. A possible reason for such over-boosting might be that this work applies the Lattice-Rescoring approach using a three-word context match around the rare words with a fixed boosting factor. A longer context would lead to a stricter match, thereby minimizing over-boosting, but it could also cause poorer recognition in genuine cases. In the future, we will try to optimize the context length and the boosting factor for Lattice-Rescoring. Nevertheless, the improvements obtained using these two methods are far more significant. Such results indicate the effectiveness of the proposed method in improving the detection performance of rare words in ASR.

Table 6. Performance of the boosting techniques for XLSR ASR

Method	WER (%)	Waypoint Detection		
		Precision (%)	Recall (%)	F1-Score (%)
XLSR	*8.53*	*95*	*45*	*61*
+G-Boosting	*6.95*	*93*	*70*	*80*
+Lattice-Rescoring	*8.16*	*97*	*63*	*76*
+G-Boosting + Lattice-Rescoring	*7.17*	*93*	*73*	*82*

Table 6 presents the performance evaluation of different customization techniques applied to an XLSR AM-based ASR. The baseline model without customization achieved a WER of *8.53%* and moderate waypoint detection performance with an F1-score of *61%*. Applying the G-Boosting technique enabled a significant reduction in WER to *6.95%* and improved waypoint detection F1-score to *80%*. While improving waypoint detection precision to *97%*, Lattice-Rescoring had a less pronounced effect on WER (*8.16%*) and overall F1-score (*76%*). Combining both techniques yielded a WER slightly higher than G-Boosting alone (*7.17%*) but achieved the best overall waypoint detection performance with an F1-score of *82%*. These results indicate that both G-Boosting and Lattice-Rescoring can enhance the performance of the speech recognition system in different ways. G-Boosting is particularly effective in improving overall accuracy (WER) and waypoint detection, while Lattice-Rescoring is better at refining the system's ability to pinpoint the exact locations of waypoints. Combining these techniques provides a balanced approach, leading to the best overall performance. In essence, these findings demonstrate that by carefully applying these customization techniques, we can significantly enhance the capabilities of speech recognition systems. Such improved results compared to the CNN-TDNN model demonstrate the effect of using a significantly more powerful AM at the core of the conventional ASR system. However, it is encouraging to note that the proposed word-boosting methods can improve Waypoint recognition even with the powerful XLSR model. Nevertheless, in the rest of the work, we continue presenting results and analysis with the CNN-TDNN model currently deployed at the client site since it requires comparatively less computational overhead and is easy to set up for production.

Table 7. Semantic Level Performance

Method	ALL		CONTACT		DIRECT_TO		STATION	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Baseline	58	73	37	53	6	11	14	24
+G-Boosting	77	86	94	96	11	20	88	93
+Lattice-Rescoring	59	74	35	52	9	17	22	36

+G-Boosting + Lattice-Rescoring	77	86	95	97	12	21	89	94

Table 7 shows the performance for all command types (Column "All") and the airspace-dependent command types CONTACT, DIRECT_TO, and STATION for the different boosting techniques. The improvement of G-Boosting and the combined technique is observed on both the word and semantic levels. A command is considered recognized if the callsign, the type, the value, the conditions, etc., are correctly extracted from the recognized sequence of words. We see a dramatic improvement from baseline to the combined technique for CONTACT by almost *60%* absolute and STATION with more than *75%* absolute. Both command types have an F1-score around *95%*. The location names uttered in connection with those two command types appear on a more regular basis and are of a low number, making it easier to extract the semantics. The accuracy of DIRECT_TO also improved by a factor of two but on a very low level. The list of potential waypoints that could be correct seems to be too big to choose the correct one. A potential solution to look into in the future would be to decide the target waypoint like "MOBSA" that only appears with DIRECT_TO and frequency names such as "MAASTRICHT", "HOLSTEIN", etc. that appear with CONTACT, on semantic level knowing the aircraft trajectory and the likelihood of mentioning one of the rare words from the N-best hypotheses.

5.2 Effect of n-gram context while G-Boosting

Rare words like "MAASTRICHT" are recognized comparatively better than the waypoints. Hence, this section reports experiments performed to improve the recognition of only the waypoints that follow the DIRECT_TO command as they are more critical. The G-Boosting approach discussed previously was performed using n-gram contexts automatically from the data. However, not all n-grams may be equally useful in waypoint boosting. Table 8 reports the performance of G-Boosting with different n-gram contexts. The performances are reported as the precision, recall, and F1-scores of only the target waypoints. The first row reports the baseline ASR performance without any boosting performed. This result is similar to that reported in Table 3, but selecting only the waypoints during performance computation. The second row reports the performance obtained using all possible tri-gram combinations where waypoints can appear at any position of the tri-gram (position-independent n-grams). This is the same setup that is used to report results in Table 5. The only difference here is that the results reported in Table 8 {are computed only for the} target waypoints, whereas Table 5 reports results for other rare words as well. It may be observed that the performance obtained by boosting using position-independent n-grams is significantly better (*33%* F1-score) than the baseline. Next, various position-dependent n-grams were explored, and the best results were obtained using the tri-grams, where waypoints can appear only as the last word in the trigram (position-dependent n-grams). The third row in Table 8 indicates that the position-dependent n-grams are even better than the position-independent n-grams and improve the performance by ≈*30%* F1-score. Finally, a different set of n-grams was explored, which are tri-gram or bi-gram waypoint contexts selected with human supervision (human-supervised n-grams) based on the validity of the word sequence to appear in a conversation. The results with this set of n-grams are reported in the fourth row. The human supervised n-grams provide ≈*20%* improvement on F1-score than the position independent n-grams but fall short of the performance of position-dependent n-grams. Therefore, it appears that the position of the waypoints in the selected n-grams used for boosting impacts their detection performance significantly.

Table 8. Performance of boosting only waypoints with different context

Method	**Waypoint Detection**		
	Precision (%)	**Recall (%)**	**F1-Score (%)**
CNN-TDNN	*100*	*7*	*13*

+Boosting (position independent n-grams)	*75*	*21*	*33*
+Boosting (position dependent n-grams)	*80*	*50*	*62*
+Boosting (human supervised n-grams)	*73*	*43*	*54*

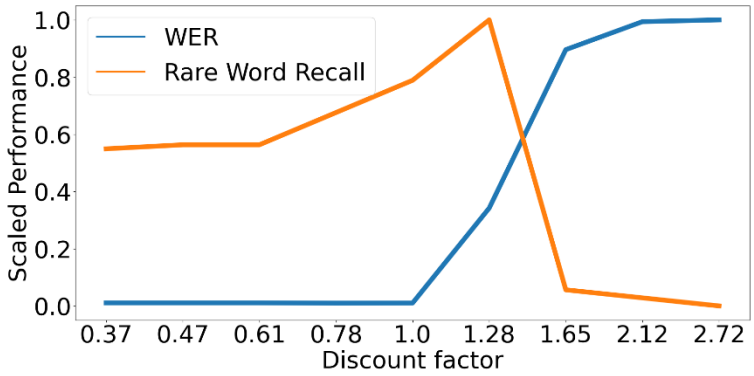

Figure 3. Effect of tuning discount factor on G-Boosting performance [6].

5.3 Effect of tuning the G-Boosting factor

As mentioned previously in subsection 4.4, the modification of the weights of a pre-trained LM is performed using a predetermined hyper-parameter known as the discount factor. The discount factor can be tuned to optimize the performance of the G-Boosting method. The impact of the discount factor used within the G-Boosting framework on overall word error rate (WER) and rare word recall [6] is illustrated in Figure 3. The x-axis quantifies discount factor values (p), while the y-axis displays scaled WER and rare word recall. The analysis reveals a nuanced relationship between the discount factor and model performance.

Discount factors below *1.0* exert minimal influence on rare word recall. Conversely, as p surpasses this threshold, a discernible improvement in rare word recall is observed concurrently with an increase in WER. This trend persists until approximately p=*1.3*, where a critical inflection point is reached. Beyond this value, the model's performance deteriorates rapidly as over-boosting leads to a surge in false positive predictions for non-target words.

These findings underscore the importance of meticulous tuning of the discount factor to achieve optimal model performance. While judicious application of boosting can enhance the recognition of rare words, excessive boosting can harm overall accuracy. The observed trade-off between rare word recall and WER highlights the need for a balanced approach to discount factor selection.

5.4 Effect of non-target waypoints seen during ASR training

Until now, the paper discusses the ways to improve the ASR performance of detecting a list of target waypoints. When the universal ASR system is deployed for a different sector that uses a list of waypoints different from those seen during training, the OW may hinder the performance of the system. Such hindrances can be observed in two forms:

- **Rare target waypoints substituted by more frequent OW**: The target waypoints can either be OOVs or very rarely seen by the ASR system. Therefore, whenever there is confusion between a waypoint and an OW during the ASR inference, the language model will predict the OW if seen more frequently than the target waypoints.

- **Homophonic waypoints substituted by OW**: When an OW sounds very similar to a target waypoint, the acoustic model score for the OW may be higher than for the target waypoint. Thus, the target waypoint may be substituted with an OW.

Considering such possibilities, exploring ways to remove the OW from the ASR predictions is logical. As it has already been discussed that retraining the ASR model is not feasible, the intervention to remove the OW words have to be done during the inference time. The approach proposed in this work is motivated by the previous experiment combining the G-Boosting and Lattice-Rescoring methods (see Table 5). Results indicate that applying Lattice-Rescoring on top of G-Boosting can further improve the detection of the target waypoints. We proposed to use Lattice-Rescoring in a modified formulation to remove the OW. The basic premise of Lattice-Rescoring involves creating a biased FST with boosted weights for the target waypoints. The removal of OW can be viewed as a reverse process of boosting waypoints. In other words, similar to boosting target waypoints, the unwanted OW can be *de-boosted* so that the probability of their prediction reduces. Such a de-boosting can be done using the same procedure as boosting using Lattice-Rescoring. More specifically, a new biased FST (say Biased-FST-OW) is created using the list of OW that needs to be removed from the ASR predictions. The OW is selected from the training data of the universal ASR system. For example, waypoints like "SASAL" and "VADOV" are not required to be predicted in the current use case. While creating the Biased-FST-OW, the de-boosting weights are carefully chosen. Whenever the OWs appear in any of the probable hypotheses for a given utterance, its score gets reduced after the lattice is composed with the Biased-FST-OW. This results in a reduction in the prediction of the OW words. Therefore, previously, the cases where OW words were predicted in place of the target waypoints could have been reduced using the de-boosting approach.

The performance of the de-boosting approach using the Lattice-Rescoring method is provided in Table 9. The performances are again reported as the precision, recall, and F1-scores of only the target waypoints. The first row lists the performance of the baseline system without any boosting. The second row shows the performance of the de-boosting method on the baseline system. It can be observed that the F1-score of detecting the target waypoints improves by *2%* by simply de-boosting the unwanted OWs. This indicates that the presence of OWs hamper the detection of the target waypoints. In the third row, the performance of the best G-Boosting scenario (row 3 in Table 8) is combined with the de-boosting based Lattice-Rescoring. It can be observed that the F1-score of this combination is further *2%* better than the only G-Boosting results of *62%* F1-score. Thus, it can be concluded along expected lines that removing unwanted OW from the ASR prediction can positively impact the performance.

Table 9. Performance of target waypoints by de-boosting Old Waypoints (OW)

Method	Waypoint Detection		
	Precision (%)	**Recall (%)**	**F1-Score (%)**
No boosting	*100*	*7*	*13*
De-Boosting (unwanted waypoint n-grams)	*100*	*8*	*15*
Boosting (human supervised n-grams) + De-Boosting (unwanted waypoint n-grams)	*79*	*54*	*64*

5.5 Semantic evaluation results with modified ASR system

A modified version of our ASR system was validated in a second, bigger human-in-the-loop simulation campaign of the DIAL project with eight ATCos in April 2024. The modification includes the boosting and de-boosting methods discussed above applied to the CNN-TDNN based ASR system. On the complete dataset of *52* human-in-the-loop simulation runs containing *27,576*

words, the WER was *9.7%* with a sentence error rate of *48.5%*. Semantic-wise, the dataset consists of *1315* ATCo utterances, including *4645* commands. Only *152* of the uttered words (*0.6%*) belong to the group of rare waypoint names. One third of them (*33.6%*) go back to the waypoints OGBER, MOBSA, EKERN, and BATEL. The other *57* uttered waypoint names sum up to just *101* occurrences. The WER of the waypoint names was *67.8%*. When looking at single waypoint names, it has to be noted that the WER was either *100%* like for a row of waypoints that have only been uttered once or twice, but for all other waypoint names, the WER had a maximum of *30%*.

Table 10. Semantic Performance of the Modified system

Baseline	Number of commands	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
All Type	*4645*	*73*	*95*	*76*	*84*
CONTACT	*517*	*67*	*93*	*71*	*80*
DIRECT_TO	*171*	*41*	*68*	*50*	*58*
STATION	*205*	*43*	*89*	*45*	*60*
CLIMB	*571*	*86*	*94*	*92*	*93*

The seven entity names for radio frequency stations, such as "MAASTRICHT", "CELLE", "LANGEN", or "RHEIN", were uttered *526* times (*1.9%* of all words). The WER of those entity names was *59.3%*. The word "MAASTRICHT" accounts for *388* (*73.8%*) occurrences of those entity names. In *70%* of the *141* cases when "MAASTRICHT" was misrecognized, it has been substituted by another word beginning with the letter "m" such as "miles".

When analysing the extracted commands in Table 10, we see roughly the same performance for the CLIMB command and a slight improvement for all types compared to the baseline runs in Table 4. However, there is a significant improvement, especially in accuracy, recall, and F1-score for CONTACT, DIRECT_TO, and STATION. Almost every second DIRECT_TO command, which contains the rare waypoint names, is correctly recognized. This performance can already help some applications. However, it has to be noted that there are several limitations.

Depending on the familiarity with the airspace, ATCos may switch between DIRECT_TO and HEADING commands to different extents. The selection of waypoints for DIRECT_TO commands that ATCos chose also differs from the simulation done with the baseline system. Furthermore, waypoint names such as MOBSA might additionally be spelled as "MIKE OSCAR BRAVO SIERRA ALFA" in an utterance which tremendously helps to extract the intended waypoint name for a DIRECT_TO command even if the single-word name itself has not been recognized.

6 Conclusions

This paper introduces two methods, G-Boosting and Lattice-Rescoring, to improve how ASR systems handle unfamiliar terms in new airspaces. These terms, like waypoint names or specific radio station names, are rarely encountered during training and often lead to recognition errors. This ease of use makes them valuable tools for ANSPs who need to adapt simulation environments or transfer ASR functionality to new airports or airspaces. These methods can even be used for prototyping new applications. Our results show significant improvements in detecting rare words like waypoints. G-Boosting alone achieves a more than *500%* increase in rare word detection F1-score compared to the baseline system. By combining G-Boosting and Lattice-Rescoring, the improvement jumps to a factor of *7*, raising the F1-score from *10%* to *70%*. Additionally, recognizing airspace-specific names within various ATC command types improves by a factor of *2* to *3*. We also show that the n-gram context in which the waypoints are boosted plays an important

role in performance improvement. Finally, a combination of boosting target words and de-boosting unwanted different sector waypoints results in a recognition F1-score of *64%*, an improvement by a factor of ≈*5* over the baseline.

Contributor Statement

Conceptualization: Mrinmoy Bhattacharjee, Petr Motlicek, Hartmut Helmke

Methodology: Mrinmoy Bhattacharjee

Data Curation: Matthias Kleinert, Heiko Ehr, Mrinmoy Bhattacharjee

Visualization: Mrinmoy Bhattacharjee

Writing –Original Draft: Mrinmoy Bhattacharjee

Writing –Review & Editing: Mrinmoy Bhattacharjee, Petr Motlicek, Srikanth Madikeri, Hartmut Helmke, Oliver Ohneiser, Matthias Kleinert

Supervision: Petr Motlicek, Hartmut Helmke

Conflict Of Interest (COI)

There are no conflicts of interest.

References

[1] S. Chen, H. Helmke, R. M. Tarakan, O. Ohneiser, H. Kopald and M. Kleinert, "Effects of Language Ontology on Transatlantic Automatic Speech Understanding Research Collaboration in the Air Traffic Management Domain," *Aerospace,* vol. 10, no. 6, pp. -29, 2023.

[2] D. Schäfer, "Context-sensitive speech recognition in the air traffic control simulation," Universität Der Bundeswehr München Fakultät für Luft- und Raumfahrttechnik, 2001.

[3] I. Gerdes, M. Jameel, R. Hunger, L. Christoffels and H. Gürlük, "The automation evolves: Concept for a highly automated controller working position," in *Proc. 33nd Congress of the International Council of the Aeronautical Sciences (ICAS)*, 2022.

[4] M. Jameel, L. Tyburzy, I. Gerdes, A. Pick, R. Hunger and L. Christoffels, "Enabling Digital Air Traffic Controller Assistant through Human-Autonomy Teaming Design," in *Proc. IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC)*, 2023.

[5] H. Helmke, M. Kleinert, N. Ahrenhold, H. Ehr, T. Mühlhausen, O. Ohneiser, L. Klamert, P. Motlicek, A. Prasad, J. Zuluaga-Gomez and others, "Automatic speech recognition and understanding for radar label maintenance support increases safety and reduces air traffic controllers' workload," in *Proc. 15th USA/Europe Air Traffic Management Research and Development Seminar (ATM2023)*, 2023.

[6] M. Bhattacharjee, I. Nigmatulina, A. Prasad, P. Rangappa, S. Madikeri, P. Motlicek, H. Helmke and M. Kleinert, "Contextual Biasing Methods for Improving Rare Word Detection in Automatic Speech Recognition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Seoul, Korea, 2024.

[7] T. Munkhdalai, Z. Wu, G. Pundak, K. C. Sim, J. Li, P. Rondon and T. N. Sainath, "NAM+: Towards Scalable End-to-End Contextual Biasing for Adaptive ASR," in *Proc. IEEE Spoken Language Technology Workshop (SLT)*, 2022.

[8] J. Tang, K. Kim, S. Shon, F. Wu and P. Sridhar, "Improving ASR Contextual Biasing with Guided Attention," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024.

[9] P. Motlicek, F. Valente and P. N. Garner, "English Spoken Term Detection in Multilingual Recordings," in *INTERSPEECH*, Makuhari, Japan, 2010.

[10] R. A. Braun, S. Madikeri and P. Motlicek, "A Comparison of Methods for OOV-Word Recognition on a New Public Dataset," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[11] H. Helmke, M. Kleinert, A. Linß, L. Klamert, P. Motlicek, J. Harfmann, N. Cebola, H. Wiese, H. Arilíusson and T. Simiganosch, "The HAAWAII Framework for Automatic Speech Understanding of Air Traffic Communication," in *Proc. 13th SESAR Innovation Days (SID)*, Sevilla, Spain, 2023.

[12] R. García, J. Albarrán, A. Fabio, F. Celorrio, C. P. d. Oliveira and C. Bárcena, "Automatic Flight Callsign Identification on a Controller Working Position: Real-Time Simulation and Analysis of Operational Recordings," *Aerospace,* vol. 10, no. 5, pp. 1-16, 2023.

[13] H. Helmke, M. Kleinert, S. Shetty, O. Ohneiser, H. Ehr, H. Arilíusson, T. S. Simiganoschi, A. Prasad, P. Motlicek, K. Vesel`y and others, "Readback error detection by automatic speech recognition to increase ATM safety," in *Proc. 14th USA/Europe Air Traffic Management Research and Development Seminar (ATM2021)*, 2021.

[14] J. Zuluaga-Gomez, A. Prasad, I. Nigmatulina, P. Motlicek and M. Kleinert, "A virtual simulation-pilot agent for training of air traffic controllers," *Aerospace,* vol. 10, no. 5, pp. 1-25, 2023.

[15] M. Mohri, F. Pereira and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language,* vol. 16, no. 1, pp. 69-88, 2002.

[16] M. Mohri, F. Pereira and M. Riley, "Speech recognition with weighted finite-state transducers," *Springer Handbook of Speech Processing,* pp. 559-584, 2008.

[17] M. Riley, C. Allauzen and M. Jansche, "OpenFST: An open-source, weighted finite-state transducer library and its applications to speech and language," *Proc. Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Tutorial Abstracts,* pp. 9-10, 2009.

[18] K. Vesel`y, A. Ghoshal, L. Burget and D. Povey, "Sequence-discriminative training of deep neural networks," in *Proc. INTERSPEECH*, 2013.

[19] H. A. Bourlard and N. Morgan, Connectionist speech recognition: a hybrid approach, vol. 247, Springer Science & Business Media, 2012.

[20] A. a. M. P. a. H. I. a. S. G. a. O. Y. a. H. H. Srinivasamurthy, "Semi-supervised learning with semantic knowledge extraction for improved speech recognition in air traffic control," in *Proc. INTERSPEECH*, Stockholm, Sweden, 2017.

[21] M. Kleinert, H. Helmke, G. Siol, h. Ehr, C. Aneta, K. Christian, D. Klakow, P. Motlicek, Y. Oualil, M. Singh and A. Srinivasamurthy, "Semi-supervised Adaptation of Assistant Based Speech Recognition Models for different Approach Areas," in *37th AIAA/IEEE Digital Avionics Systems Conference*, London, UK, 2018.

[22] J. Zuluaga-Gomez, I. Nigmatulina, A. Prasad, P. Motlicek, K. Vesely, M. Kocour and I. Szoke, "Contextual Semi-Supervised Learning: An Approach To Leverage Air-Surveillance and Untranscribed ATC Data in ASR Systems," in *Proc. INTERSPEECH*, Brno, Czechia, 2021.

[23] J. Zuluaga-Gomez, K. Vesel`y, I. Szöke, A. Blatt, P. Motlicek, M. Kocour, M. Rigault, K. Choukri, A. Prasad, S. S. Sarfjoo and others, "ATCO2 corpus: A large-scale dataset for research on automatic speech recognition and natural language understanding of air traffic control communications," *arXiv preprint arXiv:2211.04054,* 2022.

[24] J. Zuluaga-Gomez, K. Vesely, A. Blatt, P. Motlicek, D. Klakow, A. Tart, I. Szoke, A. Prasad, S. S. Sarfjoo, P. Kolcarek and others, "Automatic Call Sign Detection: Matching Air Surveillance Data with Air Traffic Spoken Communications," in *Proc. 8th OpenSky Symposium*, 2020.

[25] V. Peddinti, D. Povey and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. INTERSPEECH*, 2015.

[26] A. Vyas, S. Madikeri and H. Bourlard, "Comparing CTC and LFMMI for Out-of-Domain Adaptation of wav2vec 2.0 Acoustic Model," in *Proc. INTERSPEECH*, 2021.

[27] A. Conneau, A. Baevski, R. Collobert, A. Mohamed and M. Auli, "Unsupervised Cross-Lingual Representation Learning for Speech Recognition," in *Proc. INTERSPEECH*, 2021.

[28] A. Baevski, Y. Zhou, A. Mohamed and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," in *Proc. Advances in Neural Information Processing Systems*, 2020.

[29] S. Kumar, S. Madikeri, J. Zuluaga-Gomez, E. Villatoro-Tello, I. Nigmatulina, P. Motlicek, A. Ganapathiraju and others, "XLSR-Transducer: Streaming ASR for Self-Supervised Pretrained Models," *arXiv preprint arXiv:2407.04439,* 2024.

[30] F. Jelinek, Statistical methods for speech recognition, MIT press, 1998.

[31] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz and others, "The Kaldi speech recognition toolkit," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2011.

[32] A. Stolcke, "SRILM - an extensible language modeling toolkit," in *Proc. 7th International Conference on Spoken Language Processing (ICSLP)*, 2002.

[33] H. Helmke, J. Rataj, T. Mühlhausen, O. Ohneiser, H. Ehr, M. Kleinert, Y. Oualil, M. Schulder and D. Klakow, "Assistant-based speech recognition for ATM applications," in *Proc. 11th USA/Europe Air Traffic Management Research and Development Seminar (ATM2015)*, Lisbon, Portugal, 2015.

[34] M. Kocour, K. Vesel`y, A. Blatt, J. Zuluaga-Gomez, I. Szöke, J. Cernock`y, D. Klakow and P. Motlicek, "Boosting of Contextual Information in ASR for Air-Traffic Call-Sign Recognition," in *Proc. INTERSPEECH*, 2021.

[35] I. Nigmatulina, S. Madikeri, E. Villatoro-Tello, P. Motlicek, J. Zuluaga-Gomez, K. Pandia and A. Ganapathiraju, "Implementing Contextual Biasing in GPU Decoder for Online ASR," in *Proc. INTERSPEECH*, 2023.

[36] M. Bhattacharjee, P. Motlicek, I. Nigmatulina, H. Helmke, O. Ohneiser, M. Kleinert and H. Ehr, "Customization of Automatic Speech Recognition Engines for Rare Word Detection Without Costly Model Re-Training," in *Proc. 13th SESAR Innovation Days (SID)*, Sevilla, Spain, 2023.