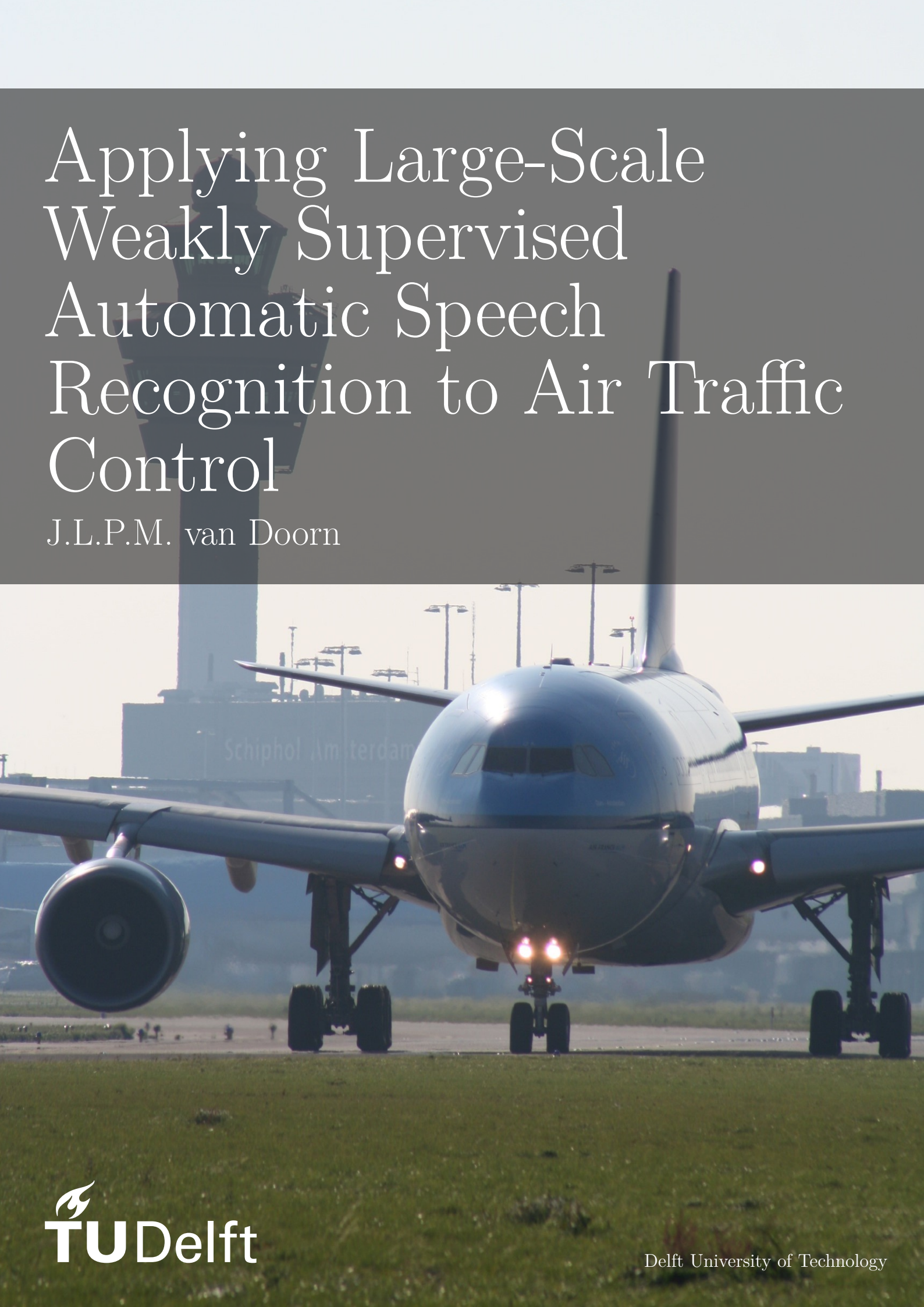
Applying Large-Scale Weakly Supervised Automatic Speech Recognition to Air Traffic Control

J.L.P.M. van Doorn



TU Delft

Delft University of Technology

Applying Large-Scale Weakly Supervised Automatic Speech Recognition to Air Traffic Control

Thesis report

by

J.L.P.M. van Doorn

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on December 11, 2023 at 13:00

Thesis committee:

	Prof. dr. ir. J.M. Hoekstra	TU Delft, professor
	Dr. J. Sun	TU Delft, supervisor
	Ir. P. Jonk	Koninklijke NLR, examiner
	Dr. X. Wang	TU Delft, examiner
Place:	Faculty of Aerospace Engineering (AE), Delft	
Project Duration:	April, 2023 - December, 2023	
Student number:	4664140	

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

Faculty of Aerospace Engineering (AE) · Delft University of Technology

Preface

Air traffic control has relied on voice-based communications for a long time. How can it be that everything on a modern airplane, except communication with the ground, is automated? It was on the 8th of January 2023 when I got to experience real air traffic control communications in the cockpit of KLM flight 836 on its way from Denpasar, Indonesia to Singapore. The moment that I was handed over the headset I started questioning myself: "How can these pilots understand what is happening?" After asking them how they do it, the captain laughed and answered: "I speak 25 different versions of English". This real-life experience is what got me thinking, how can I solve this problem? That is what ultimately led me to do the research described in this report. Using this thesis, I try to lay the groundwork for fundamentally solving the problem of miscommunication in air traffic control.

This document is the final result of my thesis to obtain the degree of "Master of Science". During my thesis, I experienced a lot of good help and effort from some people. First of all, I would like to thank Prof. Dr. Ir. Jacco M. Hoekstra. Your critical mindset made me develop a more professional and nuanced version of myself and my work. I have always enjoyed your positive attitude and our good conversations. Second, I would like to thank Dr. Junzi Sun for the ability to perform this research and for being my daily mentor. You taught me a lot and really helped me shape this research. Third, I would like to thank Patrick Jonk and Vincent de Vries from NLR for helping me look at the subject from a different perspective.

Moreover, I would like to thank my parents Laurens and Connie, and lovely brothers Willem and Gijs for their help and enthusiasm. Of course, I also want to thank Einstein for his emotional support. Last but not least, I would like to express my gratitude to all my friends for their constructive feedback and endless support: Bouwe, Cas, Chris, Emile, Fleur, Floris, Gerben, Hidde, Isabelle, Kiet, Mirthe, Simone and Thomas.

J.L.P.M. van Doorn
Delft, November 2023

Contents

Part I

Research Paper

Applying Large-Scale Weakly Supervised Automatic Speech Recognition to Air Traffic Control

J.L.P.M. van Doorn 4664140 J.L.P.M.vanDoorn@student.tudelft.nl

***Abstract* — The application of automatic speech recognition in the air traffic control domain has been researched extensively. However, its primary application remains in the training and simulation of air traffic controllers. This is due to the insufficient performance of automatic speech recognition in specific environments, such as air traffic control, where strong performance and safety requirements are paramount. This study demonstrates how a large-scale, weakly supervised automatic speech recognition model, Whisper, could meet these performance requirements and establish a new approach to air traffic control communication. Fine-tuning Whisper in the air traffic control domain resulted in a word error rate of 13.5% on the ATCO2 dataset and 1.17% on the ATCOSIM dataset. Furthermore, the study reveals that fine-tuning with region-specific data can enhance performance by up to 60% in real-world scenarios.**

1 Introduction

Automatic Speech Recognition (ASR) has been studied extensively, with some research even dating back to the 1950s [1]. The introduction of machine learning has spurred significant progress in the development of ASR models. In machine learning, there are two common approaches for the development of a learning algorithm. On the one hand, there are *supervised* learning models with long-time-ruling examples such as DeepSpeech [2], [3] and SpeechStew [4]. On the other hand, there are *unsupervised* learning models with examples such as Wav2Vec [5], [6] and BigSSL [7]. The difference between those methods is in the labeling of the data. Supervised learning models exclusively rely on labeled data, leading to a limited amount of training data because of the labor-intensive process of creating the labels. In contrast, unsupervised learning models operate with unlabeled data, leading to significantly larger volumes of training data [8].

Supervised ASR models reached around 5,000 hours of labeled training data [4]. In contrast, unsupervised models reached up to 1,000,000 hours of unlabeled training data [7]. However, neither of the two approaches can be depicted as being the best. A gap existed between small-scale supervised and large-scale unsupervised ASR models. A newly introduced automatic speech recognition model, from September 2022, tried to efficiently fill this gap. Whisper, created by OpenAI, closed the gap with the introduction of 680,000 hours of weakly supervised learning [9].

The goal of Whisper was to have a robust automatic speech recognition model characterized by high reliability and usability. This is achieved by using a vast amount of diverse training data which resulted in a very good and broad generalization. This generalization ensures that Whisper can transcribe out-of-domain audio without a significant drop in performance compared to training data, which is ideal for niche domains with distinct phraseology such as air traffic control (ATC). Therefore, studying the application of Whisper in the ATC domain is an appealing choice.

ASR technology is at the cusp of being technically feasible for application in ATC. Although numerous studies have already been conducted on applying ASR to ATC, the performance of ASR models is not yet satisfactory for broad application in air traffic control [10], [11]. The only ATC-related field of application where ASR is actively used is in training and simulation sessions [12], [13].

This is the only field where ASR is implemented due to the softer performance requirements compared to real-life scenarios in light of strong performance and safety requirements. Moreover, human performance in speech recognition is very high. According to Adriaan E. van der Groef (personal communication, October 19, 2023), a word error rate (WER) of almost zero can be expected [14]. The current performance of ASR in ATC does not match human performance. Whisper has the potential to change this. Whisper does not necessarily need to match human performance but it can be used to reduce the workload of air traffic controllers (ATCos). Possible applications may for example include incident analysis, clearance control, or safety analysis. These applications and more examples, together with a qualitative analysis of their required performance, will be discussed later in this paper.

The goal of this research was to find out how large-scale weakly supervised automatic speech recognition could be applied to air traffic control. During this study, Whisper was used to discover the potential speech recognition performance of a robust and

reliable model in an ATC context. The methodology involved assessing the out-of-the-box performance of Whisper in the ATC domain and determining the possible performance increase that could be reached by fine-tuning Whisper on global and local ATC data. To stimulate future research and reproducibility, the complete code, all public ASR models, and other needed resources are published in a GitHub repository[1].

2 Methodology

This section describes how the research has been performed. An overview can be found in Figure 1.

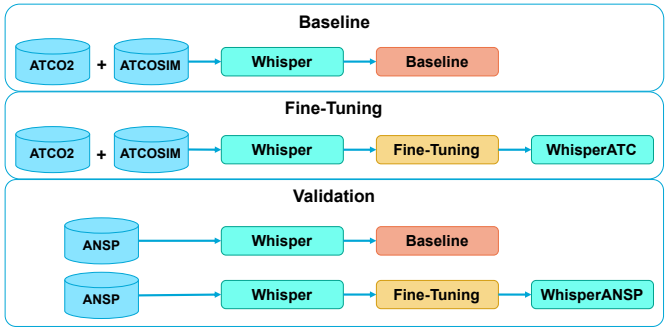


Figure 1: An overview of the methodology used in this research. First, the baseline performance of Whisper on the ATCO2 and ATCOSIM datasets was determined. Second, Whisper was fine-tuned on the ATCO2 and ATCOSIM datasets to create the WhisperATC models. Lastly, air navigation service provider (ANSP) data was used to validate the effect of fine-tuning Whisper in real-world operations.

2.1 Data Pre-Processing

The datasets used in the research and the preparation thereof will be described in this subsection.

2.1.1 Datasets

At the moment of writing, three public and free ATC-related datasets are available, namely the ATCO2 dataset [15], the ATCOSIM dataset [16] and the UWB-ATCC dataset [17]. However, the latter contains transcriptions that are nested in a way that it was hard to extract the literal text for training purposes despite serious effort. Moreover, the ATCO2 and ATCOSIM datasets do form a benchmark in the development of ASR in ATC [18]. Therefore only the ATCO2 and ATCOSIM datasets were used. An overview of the datasets with their corresponding characteristics can be found in Table 1.

The ATCO2 dataset, as part of the ATCO2 project[2] started in 2020, is a collection of speech from multiple airports, mainly around Europe. The free and publicly accessible portion accounts for a total of one hour of speech. It consists of speech collected from pilots and ATCos. Moreover, it is a community-driven project where the audio is captured with simple very high frequency (VHF) antennas. Therefore it contains relatively noisy data. Further, it includes radar data (e.g., nearby waypoints and call signs) that was augmented using the OpenSky network. The audio was captured at a sampling frequency (f_s) of 16 kHz, which is common for audio capturing [15].

The ATCOSIM dataset consists of ten hours of speech collected during a simulated session in the Eurocontrol experimental center in France. It only includes the speech of the ATCo role, more specifically from ten ATCos at the en-route position. Moreover, since it concerns a simulated session, the clarity of the data is much higher. Therefore some artificial noise was added for a more realistic representation. Additionally, the audio was captured at a higher sampling frequency of 32 kHz which, together with the low noise footprint, resulted in better audio quality [16].

Table 1: An overview of the used datasets.

	ATCO2	**ATCOSIM**
Size (hrs)	1.1	10
Origin	Europe	France
Speaker	Pilot & Controller	Controller
Language	English	English
Misc.	Radar Data	-
f_s (Hz)	16,000	32,000

2.1.2 Pre-Processing

The first step was to process the transcripts. Since both datasets, as mentioned previously, were not created in an equal way, the transcripts needed to be put into a single format. The actual text was extracted from extensible markup language files and purified by removing text in between brackets, e.g., [...], (...), and < ... >. Furthermore, for the ATCO2 dataset, the radar data were converted into a simple machine-readable format. Lastly, all audio and transcript pairs of which the transcript was empty (i.e., empty audio files) were filtered out.

After the transcripts had been processed, the audio was processed by re-sampling the files at 16,000 Hz, the required frequency for Whisper to work.

After processing the audio and corresponding labels, the dataset was divided into two parts called splits. A part of the data was used as training data, i.e., for the fine-tuning of Whisper. The remaining portion was used for the validation process during the fine-tuning of Whisper. Commonly, the dataset is divided into 80% training and 20% validation data [19]. These numbers were used in a random process to generate the training respective validation portions of the dataset. When the datasets were created they were uploaded to the HuggingFace Hub, a development platform and repository system for machine learning models. Apart

[1] https://www.github.com/jlvdoorn/WhisperATC
[2] https://www.atco2.org

from the baseline performance assessment, only the validation split is used for evaluation.

The final datasets, as listed in Table 2, were structured as follows. The ATCO2 dataset contained 446 training samples and 113 validation samples. Each of the samples consisted of an audio file, a transcript, and an extra file containing radar data of the corresponding audio. The ATCOSIM dataset contained 7,646 training samples and 1,913 validation samples. Each sample consisted of an audio file paired with a transcript, but, without any radar data files. In addition, the datasets were also combined into the ATCO2-ATCOSIM dataset which ultimately contained 8,092 training samples and 2,026 validation samples. Each sample consisted of an audio file, a transcript, and a radar data file when available. This dataset was only used for fine-tuning the models.

Table 2: An overview of the pre-processed datasets.

	ATCO2	**ATCOSIM**
Total Size (hrs)	1.1	10.46
Train Size (hrs)	0.86	8.37
Validation Size (hrs)	0.23	2.09
Total Samples	559	9,559
Train Samples	446	7,646
Validation Samples	113	1,913

2.2 Baseline Performance

This subsection describes the process of assessing the performance of the out-of-the-box Whisper model, i.e., determining the baseline for the rest of this research.

2.2.1 Evaluation Metrics

The most common evaluation metric in automatic speech recognition is the word error rate [10]. It compares the transcript with the reference (ground truth) as supplied by the dataset on a per-word basis. The WER can be calculated by the following formula:

$$WER(\%) = \frac{S + I + D}{N} \times 100\%. \tag{1}$$

Here, S represents the number of replaced words, I represents the number of wrongly inserted words, and D represents the number of missing words in the transcript when compared with the reference. N, on the other hand, represents the total number of words in the reference. The WER score is often expressed as a percentage.

Most ATC speech consists of two segments, the call sign and the command [20]. The call sign is used to address the aircraft to whom or from whom the communication is going. The command is the actual instruction given. So it is also possible to assess the model's performance by looking at the call sign and command extraction rate. Although conceptual understanding is more important for some applications, the first step is to focus on correctly transcribing audio. After that, the attention can be redirected toward call sign and command recognition as will be explained later in Subsection 4.6.

The WER score was determined by comparing all the generated transcriptions with all the references at once. By calculating the WER over the whole set of transcriptions instead of per utterance, a weighted average method was applied. This ensured that utterances with fewer words do not hold the same significance as those with a larger word count when computing the overall measure. Instead, each word held the same importance. This method resulted in an unbiased view of the actual performance.

2.2.2 Prompting

Whisper's diverse and vast amount of training data resulted in zero-shot capabilities. That means it can recognize specific scenarios without being trained on data that is paired explicitly with that scenario. In contrast, within specialized contexts such as air traffic control, the broad generalization may potentially result in a counterproductive deficiency of contextual awareness. This was solved with a so-called prompting scheme.

A prompt is a free text input that can enhance Whisper's textual awareness when transcribing audio. It gives importance to the words given in the prompt and tries to look for words that are closely related to those in the given prompt. Ideally, a prompt will contain all the hard-to-transcribe items such as local waypoints (e.g., ARTIP, WOODY), entities (e.g., Bordeaux Radar, Kines-Saint Prex), or possible airlines and call signs (e.g., KLM Six Eight One, Lufthansa, Speedbird Five Alpha).

The construction of the ideal prompting scheme is illustrated at the end of the next paragraph.

2.2.3 Normalization

Another property of Whisper is that it has been trained to output "intelligent" transcripts. Instead of outputting plain text, as is the standard in natural language processing, Whisper generates complete texts including items such as punctuation and accents. Yet, the datasets often contain labels in plain text format. Therefore, some processing was needed to be able to accurately compare the generated transcript with the ground truth and thus, correctly calculate the word error rate.

This process is called normalization. Whisper contains a built-in normalizer that was used as a foundation for the normalization process and was adapted to certain ATC-specific terminology and phraseology. The difference between prompting and normalization can be seen in Figure 2.

An iterative process was followed to construct the ideal prompting and normalization scheme. In each iteration, ten audio files were transcribed with a certain prompt and were then passed through a normalizer. After each iteration, a manual comparison of the generated transcript and the reference resulted in adaptations on the given prompt and the used normalizer. This way,

the most ideal prompt and normalizer were constructed empirically.

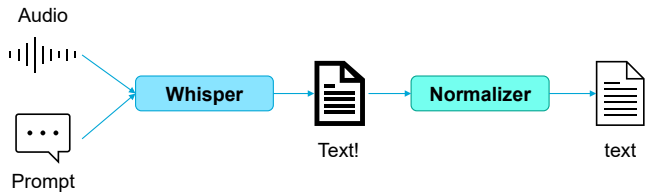


Figure 2: The difference between prompting and normalization.

2.2.4 Inference

During inference, the actual performance of the out-of-the-box Whisper model was assessed. This was done by loading the datasets from the HuggingFace Hub and loading the *large-v2* model from the Whisper Python package. Then, each audio file was processed systematically. First, the so-called log-mel spectrogram of the audio file was calculated. The log-mel spectrogram is a regular logarithmic audio spectrogram with the y-axis being adapted to the human hearing frequency levels expressed as mels. Second, the prompt was set up. The prompt contained the words *Air Traffic Control Communications* to indicate the context. Further, it was extended with some terminology (e.g., ILS, knots, heading) and the NATO alphabet. If available in the dataset, the radar information was also added to the prompt. Moreover, automatic language detection was turned off and the language was manually set to English to maximize the performance. Afterward, each audio file was transcribed twice: with and without the prompt. The results were then saved in a data frame for later use.

After each file had been transcribed twice, the word error rate was calculated. This was also done twice, the first time with normalizing both the reference and the transcript and the second time without normalization. Then, the WER scores were also calculated twice, once for the whole dataset (training + validation split) and once for the validation split only. The latter was done to have a fair comparison between the blank and any fine-tuned models later on.

In total, the inference of one model gave eight results per dataset, so sixteen results for both the ATCO2 and ATCOSIM datasets.

2.3 Model Fine-Tuning

For fine-tuning Whisper, the *large-v2* variant was used since it would most-likely result in the best achievable performance. To facilitate the fine-tuning, the HuggingFace *transformers* Python package was used [21]. This created an easily usable solution for fine-tuning any available model on the HuggingFace Hub with any dataset. The *Whisper Large v2* model was fine-tuned on the three created datasets: *ATCO2*, *ATCOSIM*, and *ATCO2-ATCOSIM* to determine the best available model. A list of all created models can be found in Appendix B.

First, the data were prepared. All audio was resampled at 16 kHz to make sure that it was processed correctly. Second, the Whisper tokenizer transformed the labels into input tokens. These tokens represented the words in the lexicon of the language model of Whisper. Third, Whisper's feature extractor was used to calculate the log-mel spectrogram expressed as an array of input features (e.g., an array containing the power of the signal for each frequency at a given time point). Fourth, a data collator was constructed to ensure the input tokens and the spectrograms were of the same length, time-wise. That means, in case of a silence in the audio, it was expressed as a certain input token in the token array to ensure that there was no information mismatch between the input IDs and the label. Fifth, a metric calculation function was created to calculate the WER score during training. Finally, the training and validation splits of the dataset were designated for the training process (train and evaluation set).

Then, the fine-tuning process started. The samples in the training split were, in pre-defined batch sizes, sent over to the model to predict an output. Based on the difference between the predicted output and the correct output (the data labels) the model parameters were adjusted. The fine-tuning process cycled over the training dataset a fixed number of times (epochs). After the fine-tuning, the models were uploaded to the HuggingFace Hub[3].

2.3.1 Hyperparameters

Each of the three datasets had its own fine-tuning process. Each process needed dedicated hyperparameters due to the difference in dataset size. Table 3 lists the hyperparameters used for each process. It was decided to train each model for around 100 epochs, and from there, the rest of the hyperparameters were calculated. During the fine-tuning process, the learning process was monitored to verify whether the model "learned" as required.

Table 3: The hyperparameters used in each fine-tuning process.

Parameter	ATCO2	ATCOSIM	A2-ASIM
Max. Steps	2,800	12,500	12,644
Train Samples	446	7,646	8,092
Batch Size	16	64	64
Epochs	100	104.2	99.56
Eval Steps	100	1,000	250
Save Steps	100	2,000	1,000

2.3.2 Hardware Resources

Fine-tuning a machine learning model requires high-level hardware. It was decided to use the Delft High

[3]https://huggingface.co/jlvdoorn

Performance Computing cluster (DHPC) as part of the facilities of the Delft University of Technology [22].

Each fine-tuning process had its own hardware requirements. Whisper was fine-tuned on the ATCO2 dataset using one NVIDIA Tesla V100S graphics card with 32 GB of video memory. Further, one AMD EPYC 7402 24C CPU was used together with 128 GB of working memory. All resulted in a fine-tuning time of approximately 21.5 hours.

Due to the large size, the ATCOSIM and ATCO2-ATCOSIM fine-tuning processes needed more hardware. There, four NVIDIA Tesla V100S graphics cards were used, each with 32 GB of video memory. Moreover, the number of CPUs was increased to four AMD EPYC 7402 24Cs. The same working memory of 128 GB was used. The time needed for fine-tuning was 32.5 and 38 hours respectively for the ATCOSIM and ATCO2-ATCOSIM datasets.

2.4 Validation

To validate whether the methodology worked as intended, real-world data were used. These data were supplied by a real-world air navigation service provider. The blank Whisper model was first used to set a baseline on the ANSP dataset. Later, the ANSP data were used to fine-tune Whisper.

2.4.1 ANSP Dataset

To validate the illustrated methodology, ANSP data were used. The supplied data consisted of a week of audio (3rd Oct. 2022 - 9th Oct. 2022) originating from the tower controller ATCo-position. The audio contained files of unequal length and consisted of both controller and pilot speech. Furthermore, it did not include any metadata such as speaker IDs. Since the recording was done on the controller's site, the audio contained relatively large quantities of noise, especially for the pilot speech. This noise reduced the quality of the data enormously. Moreover, it was collected at a sampling frequency of only 8 kHz which is low compared to standards. This further reduced the quality of the recordings. Lastly, only the audio was provided, thus the labels, i.e., transcripts, still needed to be created.

The first step in creating the dataset was selecting the data that would be used. Since the total amount of ANSP data (139 hours across 50,000 files) was far more than needed compared to the amount of data used for fine-tuning, only a small slice, of approximately three hours was used. During the labeling process, the actual selection of the data was based on the noise level, amount of non-English speech, and lack of utterances.

After setting the dataset requirements as illustrated in the previous paragraph, the data were processed. This was done first by re-sampling the audio from 8 kHz to 16 kHz and removing empty audio files, i.e., audio without any utterances. Then, the audio was listed in random order to remove any time-based (and thus also speaker-based) biases.

The third step in the dataset creation process was the labeling of the ANSP data. Since this would be a labor-intensive task, it was chosen to run the best-performing fine-tuned model, as created according to the previous subsection, on the ANSP data to create pseudo-labels. These pseudo-labels, together with their corresponding audio files, were used in a program called *Prodigy*. This software presented an audio file together with its pseudo-label. The label could then be edited manually while listening to the audio file. That way, the correct labels of the ANSP dataset were created.

During the labeling process, some audio files were rejected due to their noise level, amount of non-English speech, or lack of utterances. These metrics are standard to take into account during the creation of a dataset. The audio was labeled up until the point that the total duration of three hours was reached. The final dataset consisted of a duration of three hours across 1001 files. Again, this dataset was randomly divided into a training and a validation partition using the 80% to 20% ratio. This resulted in a training partition containing 799 samples with a duration of 2 hours and 23 minutes and a validation partition of 202 samples and a duration of 37 minutes respectively. A complete overview of all used datasets in this research can be found in Appendix A.

2.4.2 Inference

The performance assessment on the ANSP audio was executed in the same manner as the baseline performance assessment on the ATCO2 and ATCOSIM datasets (Subsection 2.2). First, the blank model was tested on the ANSP dataset, and afterward, all three fine-tuned models were tested on the ANSP dataset. The results showed what performance could be achieved in a real-world scenario. The WER was calculated in the four configurations of prompting and normalization. It gave sixteen WER scores in total, four per model.

2.4.3 Fine-Tuning

It could be expected that the ANSP dataset would contain a lot of location-specific terminology and vocabulary. Thus, a general ATC-ASR model would presumably only be able to reach a certain WER on domain-shifted data such as the ANSP dataset. Therefore, it could be beneficial to adapt to those location-specific irregularities. This was solved by fine-tuning Whisper on a piece of the ANSP data. The performance of this fine-tuned model evaluated on the ANSP data gave insight into the effect of in-domain training (e.g., fine-tuning and evaluating on the same dataset).

Now, due to security reasons, the ANSP data could only be processed on specialized NLR (Royal Netherlands Aerospace Centre) hardware. Therefore, the fine-tuning process could not be performed anymore on the DHPC hardware. It was decided to use the NLR equivalent for the fine-tuning process. This machine consisted of 188

GB of RAM, 48 Intel Xeon Gold 6136 CPUs, and one NVIDIA Tesla V100 32 GB GPU.

The fine-tuning was performed for 125 epochs, using a total batch size of 64. With a total of 799 training samples, this resulted in 1500 training steps. The evaluation was carried out every 125 steps and the model was saved at four checkpoints (every 375 steps) to create the best working model. Time-wise, 12.5 hours were needed per training session. In total, this gave 50 hours of total training for the four models.

3 Results

In this section, the results and outcomes of the research are discussed.

3.1 Prompting and Normalization

Figure 3 shows the effect of prompting and normalization on the word error rate. It is seen that both had approximately the same effect. Yet, a combination of the two yielded the best result. The word error rate on the ATCO2 dataset was reduced from 42.27% to 9.73%, a 33% absolute reduction. On the ATCOSIM dataset, the WER was reduced by an absolute 76%, from 78.72% to 3.12%.

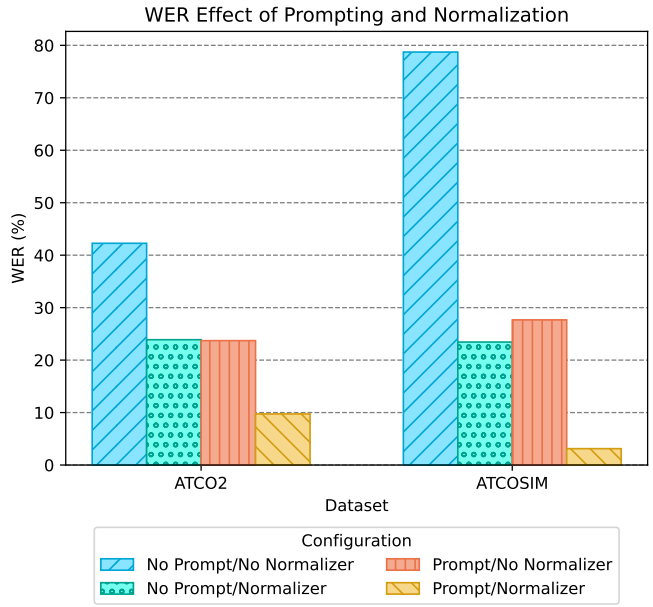


Figure 3: The effect of prompting and normalization on the word error rate. Each group of bars represents a slice of a dataset.

During the construction of the prompting scheme, the first step was to state the context (i.e., "air traffic control communications"). Afterward, it was extended with a list of airlines (e.g., KLM, Lufthansa, and Speedbird). Later, all location-specific items such as full call signs, waypoints, and entities were added (e.g., KLM Six Eight One, WOODY, Amsterdam Radar). Ultimately the NATO alphabet and some terminology such as ILS, VFR, etc. were added to the prompt.

Table 4: The effect of prompting and normalization on the word error rate. The best WER for each dataset slice is printed in *italic*.

Prmpt.	Norm.	ATCO2	ATCOSIM
no	no	42.27	78.72
no	yes	23.89	23.44
yes	no	23.71	27.66
yes	yes	*9.73*	*3.12*

On the other hand, for normalization, the built-in Whisper normalizer was used as a foundation. It was extended with several functions and filters. It first made sure that only text was passed through (i.e., removing any non-alphanumerical characters). Later it ensured that all numbers were written in numerical form and it split all numbers into digits (e.g., 501 becomes 5 0 1). Further, it also standardized certain wordings (e.g., "goodbye" became "good bye") to have a consistent format. Lastly, it also ensured that everything was processed as lowercase text.

3.2 Baseline Performance

The performance of the blank Whisper Large v2 model on the ATCO2 and ATCOSIM datasets can be seen in Table 5 (and in Figure 4 as part of the comparison with the fine-tuned models).

The blank Whisper model reached as low as 24.03% word error rate on the ATCO2 dataset and 16.74% on the ATCOSIM dataset. These results were the best available performance, with both normalization and prompting enabled.

Table 5: The performance of out-of-the-box Whisper on the training (T) and validation (V) splits of the ATCO2 and ATCOSIM datasets for each configuration of prompting and normalization. The best score for each dataset and split is printed in *italic*.

Prmpt.	Norm.	ATCO2		ATCOSIM	
		T+V	V	T+V	V
no	no	71.60	71.62	79.70	79.11
no	yes	31.45	29.05	18.14	17.98
yes	no	63.91	61.08	64.10	63.62
yes	yes	*27.14*	*24.03*	*17.10*	*16.74*

3.3 Fine-Tuned Performance

The performance of the fine-tuned and blank models of Whisper is displayed in Figure 4. In the rest of this section, only the validation part of the data is presented in the results as this was not used for training. For the y-axis (WER), the best-performing configuration (prompting/normalization) was taken for each score. The complete set of results can be found in Appendix C.

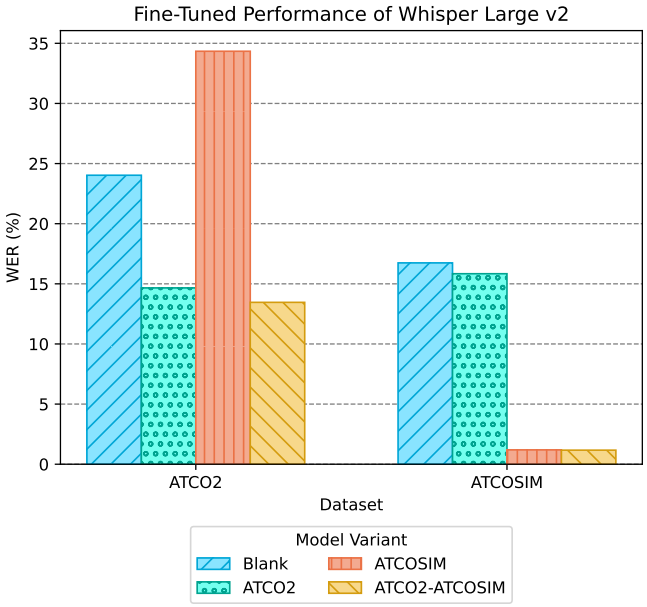


Figure 4: The performance of the blank and fine-tuned models on the ATCO2 and ATCOSIM datasets.

The fine-tuned performance results show that, apart from a single outlier, training Whisper on air traffic control speech significantly reduced the word error rate. This is especially true for the model that had been fine-tuned on both ATCO2 and ATCOSIM. It was the best-performing model across all datasets.

Looking at the results, fine-tuning Whisper on ATCO2 resulted in a WER score of 14.66% on the ATCO2 dataset. Further training of the model, on ATCOSIM, resulted in a small performance increase with a WER of 13.46%.

On the other hand, fine-tuning Whisper resulted in an even bigger WER drop on the ATCOSIM dataset. Blank Whisper reached 16.74% WER on the ATCOSIM dataset. By fine-tuning Whisper on the 8.37 hours of training data in the ATCOSIM dataset, a WER of just 1.19% is reached. Extending the training data with an additional 0.86 hours from ATCO2, the WER is decreased just a bit more, to 1.17%.

Table 6: The performance of the blank and fine-tuned models on the ATCO2 and ATCOSIM datasets. The table's index lists the fine-tuned model version and the columns list the datasets. The best score for each column is printed in *italic*.

Mdl	ATCO2	ATCOSIM
-	24.03	16.74
A2	14.66	15.84
AS	34.34	1.19
A2-AS	*13.46*	*1.17*

The effect of fine-tuning becomes tangible when looking at some examples. For example, the blank model predicted the following: "Telegraph, Skyfinal 25 for touch and go" whereas fine-tuned Whisper predicted: "Hotel Echo X-ray final two five for touch and go". It is seen that the fine-tuned model produces meaningful sentences instead of falsely predicting some words. The blank model already understood some *terminology*, but by fine-tuning, it learned to understand *phraseology*, a big difference.

3.4 Model Validation

After the creation of the models, the methodology of this research, fine-tuning Whisper on the ATC domain to increase ASR performance, was validated. This was done using ANSP data.

3.4.1 ANSP Baseline Performance

Blank Whisper was evaluated on the created ANSP dataset. The word error rate was compared with the performance on the ATCO2 and ATCOSIM datasets. The results are listed in Table 7.

Whisper achieved a 32.02% word error rate on the ANSP dataset. This was more compared to the 24.03% and 16.74% on the ATCO2 and ATCOSIM datasets respectively. This difference was explained by the fact that the ANSP dataset contained the highest level of noise and the lowest sampling frequency. Thus, the overall quality of the audio in the ANSP dataset was the worst among all three used datasets.

Further, the ANSP dataset was much more location-centric than the broad ATCO2 and ATCOSIM datasets. It only contained audio from a single location. Moreover, it also contained a lot of region-specific items (e.g., waypoints, SIDs, STARs) since it concerns a tower controller.

On the other hand, still more than two-thirds of the audio in the ANSP dataset was transcribed correctly and it could only be expected that this amount would increase after fine-tuning Whisper.

Table 7: The performance of the blank Whisper model on the ATCO2, ATCOSIM, and the ANSP dataset. The first two columns indicate the configuration of prompting and normalization that was used. The best score for each dataset is printed in *italic*.

P	N	ATCO2	ATCOSIM	ANSP
no	no	71.62	79.11	78.49
no	yes	29.05	17.98	*32.02*
yes	no	61.08	63.62	68.73
yes	yes	*24.03*	*16.74*	35.09

3.4.2 ANSP Fine-Tuned Performance

It can be argued that to achieve perfection, training on a local dataset is the best approach. A good generalization for transcribing ATC-related speech can be reached using broad and diverse training data, such as the combination

[4]International Civil Aviation Organization

of ATCO2 and ATCOSIM (Table 11). Nonetheless, each ATC station has its standards and procedures that might deviate from the ICAO[4] guidelines. This is reflected in the ATC communication of the station. To reach a perfect transcription, it is inevitable to fine-tune on local, region-specific, data. Therefore, a test was carried out to see the effect of fine-tuning and evaluating within the same dataset. The main focus here was on the ANSP dataset. The results are presented in Figure 5.

The figure shows how fine-tuning a model to a certain environment increased the performance in that environment. The blank Whisper model set a baseline WER of 32.02% on the ANSP dataset. Fine-tuning on the 2 hours and 23 minutes of the training set resulted in a WER of 13.28% which was an absolute reduction of 18.74% translating into a relative reduction of 59%. In this context, a relative reduction was far more important since it concerned the fine-tuning effect.

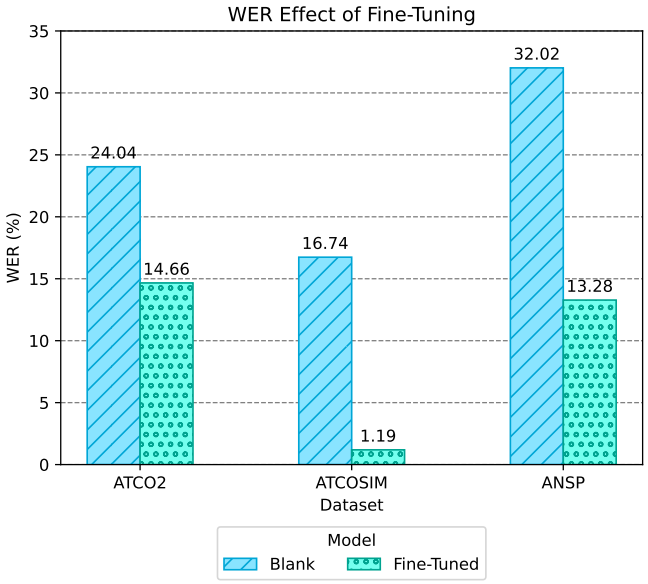


Figure 5: The effect of fine-tuning and evaluating on the same dataset (training respective validation partition).

The fine-tuning effect can be seen in the following example. In one sample the blank model predicted: "total green warfare elevator level four vhf final eight right", whereas fine-tuning on the ANSP dataset delivered: "tower good morning KLM eight zero four we is final eighteen right". Even though the fine-tuned model was not perfect, i.e., "we is" should have been "we are at", it produced far better results than the blank model. The latter transcript contained a meaning whilst the blank model produced useless text.

This reduction is even more extreme with the ATCOSIM dataset. The blank Whisper model reached a WER of 16.74%, transcribing one in six words incorrectly, which was already quite low. With fine-tuning on the ATCOSIM dataset it reached a word error rate of a mere 1%. Only one in 85 words were transcribed wrongly, a relative reduction of 93%.

Table 8: The effect of fine-tuning on the evaluation on in-domain data.

Model	ATCO2	ATCOSIM	ANSP
Blank	24.04	16.74	32.02
Fine-Tuned	14.66	1.19	13.28

The out-of-the-box Whisper model reached a WER of 24.04% on the ATCO2 dataset. By fine-tuning the model on the training set of the ATCO2 dataset, the WER was reduced to 14.66%. Without training less than one in four words was transcribed incorrectly. After training this was reduced to almost one in seven words, a 40% relative reduction.

4 Discussion

This section discusses the results, shows limitations of the work, and illustrates future work suggestions.

4.1 Results

First of all, the effect of prompting and normalization shows that both may have a substantial effect on the word error rate. However, it must also be said that prompting relies on a priori knowledge and thus may not always be expected during transcribing. On the other hand, it is possible to implement using a data augmentation system (e.g., using radar data). In contrast, normalization is based on a posteriori knowledge, and therefore it seems reasonable to expect normalization to be applied after transcribing.

The baseline performance assessment shows that the performance on the validation split of the datasets approximates the performance on the whole dataset (the training and validation split). From there, it can be argued that both partitions of the dataset are created in an equal manner. Thus, both partitions (training and validation respectively) consisted of a similar level of difficulty. This can be said for the ATCO2 and the ATCOSIM datasets. Since the combined dataset (ATCO2-ATCOSIM) was a summation of the two individual datasets, the combined dataset logically also contained partitions with equal difficulty.

Moreover, it is visible that the model performed much better on the ATCOSIM dataset compared to the ATCO2 dataset. The main reason for this is the difference in noise level in the audio of the datasets. ATCO2 is a community-driven project that collects audio with a collection of simple VHF antennas. They pick the audio from the air and send it over the internet to a database [15]. This way of collecting includes a lot of transmission noise in the audio. In contrast, the ATCOSIM dataset had been produced in a simulated environment where the audio is captured at the source, resulting in less noise and thus better audio quality. That way, it is understandable that the model performs better on ATCOSIM compared to ATCO2.

The fine-tuned models reached a WER of 13.46% on the ATCO2 and a WER of 1.17% on the ATCOSIM datasets respectively. The previous state-of-the-art was set at 15.4% on ATCO2 and 5.0% on ATCOSIM [15], [18]. Thus, Whisper sets a new benchmark for ASR on ATCO2 and ATCOSIM.

When looking at the amount of training data needed, Whisper is even more conspicuous. The previous ATCO2 benchmark was set with 3600 hours of training (originating from the full ATCO2 dataset)[15]. For Whisper, training on only the 0.86 hours of the training partition of ATCO2 already resulted in a WER of 14.66% on the ATCO2 validation set. Extending the training to include ATCO2 and ATCOSIM (totaling 9.23 hours), the before-mentioned WER of 13.46% was reached. This indicates that fine-tuned Whisper outperformed the best available model on the ATCO2 dataset.

This is also true for the ATCOSIM dataset, where the best-performing model still required 176.4 hours of training data [18]. That is drastically more than the 8.37 hours needed for the ATCOSIM model. Again, the test set-up was similar, 80% of the ATCOSIM dataset was included in the 176.4 hours of training and the validation was carried out on the remaining 20% of the ATCOSIM dataset.

On the other hand, the split in the used ATCOSIM dataset was created randomly. However, since the ATCOSIM dataset consists of only ten speakers and does have speaker labels, it could be of interest to divide the dataset based on speaker IDs. It is chosen, to use *sm2* and *zf2* for the validation set and the rest for the training set. Then, fine-tuning the model on that training set gives a score of 3.88% on the new validation set. This is higher than the 1.17% mentioned before, yet it still sets a new benchmark compared to the current state-of-the-art of 5.0%.

The ATCO2 project claimed that previous work showed that the use of standard speech corpora (e.g., LibriSpeech[23], CommonVoice[24]) for training an ASR model is not effective in increasing speech recognition performance in the ATC context [15], [25]. However, Whisper has been trained on 680,000 hours of audio from diverse sources to create a good generalization. That causes that, as has just been shown, only a small amount of ATC-related training is needed to create a model that outperforms literature in transcribing ATC conversations. The good generalization of the blank Whisper model is the main power that enables the small amount of training data to suffice for low WER scores in the ATC domain.

Overall, the fine-tuned results show that a low word error rate is possible, even with the small amount of useful, publicly accessible, and free datasets. It can be said that Whisper has learned to understand the phraseology and terminology of ATC conversations by fine-tuning it on the ATCO2 and ATCOSIM datasets. The created models set a new benchmark for the application of automatic speech recognition in the air traffic control domain.

Lastly, it can be argued that the ATCOSIM and ANSP datasets are similar in the way that they were constructed in a single physical location or region. Contrary, ATCO2 was created by collecting data from a much broader area. On the other hand, the ANSP dataset best represents the real world among the three datasets. First, since the ANSP dataset is a slice of the real-world ATC speech, but further, because ATCOSIM was collected in a simulated environment. When looking at the results with all this in mind, it can be argued that fine-tuning on a region-specific dataset delivers the best and more importantly the most realistic performance increase.

4.2 Possible Applications

As presented in Section 3, the ATCO2 and ATCOSIM fine-tuned ASR model reached up to 13.46% WER on the ATCO2 and 1.17% WER on the ATCOSIM dataset respectively. Further fine-tuning on local data (ANSP dataset) resulted in a WER of 13.28%.

Multiple works showed that the application of ASR in the ATC domain is feasible [26], [27]. However, a severe problem that has not yet been tackled is the required performance. Currently, the only field where ASR is applied in air traffic control is the field of training and simulation [12], [13]. In this field, the required performance is not as high as in real-world applications since it only concerns training in a simulated environment. Nonetheless, the current reachable performance is sufficient for a lot more applications. This subsection will mention those, together with possible future applications, in an ascending order of required performance. It briefly describes each application, gives a qualitative analysis of the required performance levels, and compares the required performance with the current achievable performance.

The applications with the lowest required performance would, instead of real-time processing, need to focus on the post-processing of audio. Examples such as incident analysis, where an ASR model can be used to transcribe parts of audio for a text-based incident investigation, could be included. It does not rely on call sign or command recognition per se, but just on a correct transcription with a relatively low word error rate. The performance of the current model suffices for these purposes. Further, this application would be the easiest to implement since the model can be used as-is, without any real-time adaptations.

Following up would be an application like a clearance control monitoring system. An ASR model could be used to monitor clearances and prevent potentially dangerous situations by catching the speech of the responsible ATCo and comparing it with radar data. This would require a small shift of focus toward the understanding of the ATC speech to recognize the word “cleared” and the corresponding call sign and command. The performance of the current model is high enough for such an application, but it would require some (real-time)

calculation framework to include radar data and focus on the clearance keywords.

As discussed, the field of training and simulation is the only field where applications are implemented in real-world usage. An ASR model is used to assist or even replace a pseudo-pilot. In the former, the focus would be on the WER with a slight shift toward understanding, to pre-fill some given instructions in the pilot's interface. The latter would require a good understanding of the speech for a correct extraction of call signs and commands for the ASR model (pilot) to be able to adhere to the given instructions by the ATCo. The current model will have a sufficient performance to assist the pseudo-pilot but it is not accurate enough yet to completely replace a pseudo-pilot. Also, some adaptions are needed for the model to be implemented in a real-time working application.

A more demanding application would be a safety monitoring system. The automatic speech recognition model could be applied in real-time to analyze the speech of the ATCo. By analysis of the speech, the safety monitoring system could catch potentially dangerous situations. Like the clearance controller example mentioned earlier, but now in a general form for approach-, area- or upper-area controllers. The model could catch potentially dangerous instructed route, altitude, or speed commands and warn the controller. In that case, a very good understanding of the speech and context is needed, which is not yet achievable with the current model. However, it could have a major impact in the long term.

Miscommunication is a large problem in air traffic control. Even though standard phraseology exists for ATC communication, a large portion can be traced back to wrong understandings of speech [28]. The leading cause of this is a difference in language proficiency [29], especially in non-western countries where the level of English is below par. An ASR model like Whisper could help solve this problem. For example, by creating a kind of subtitle that could be sent over a data-link channel to reduce the amount of misunderstanding. Although it can have an effect already by only focusing on the transcription itself. For it to succeed and have more impact, almost perfect transcriptions and a deep understanding of the matter are needed. Moreover, it would need to work no matter the accent or geographical region, thus requiring a robust and multi-lingual model. Again, the high requirements make this a project for the long term since the current model does not have the needed capabilities.

It can be seen that Whisper could make an appearance in some applications already. However, most of them still have a more robust performance requirement, leaving a gap. As this gap becomes narrower, Whisper could form the foundation of many more ASR applications in ATC in the future. Some suggestions to close this gap will be discussed in Subsection 4.6.

4.3 Data Availability

It is difficult to get hold of data that is suitable for training an ASR model for the ATC domain. The ATCO2 and ATCOSIM datasets are the only two free publicly available datasets that suffice the needs. However, these datasets are not very large (1 hour and 10 hours respectively). In many works, the lack of data is stated as being an issue [10], [15]. To further improve the performance of ATC-related ASR models and stimulate open research, it is important to have more free, public data available.

Further, the diversification of the data is of great importance. Ideally, the model would be trained on data from all over the world (i.e., different accents, phraseology differences, and ATCo positions). However, the ATCO2 and ATCOSIM datasets are both limited in that sense. Both are mainly from Europe and only contain a small amount of diversity in terms of accents and ATCo positions. That causes the models fine-tuned on ATCO2 and ATCOSIM to perform worse on left-out datasets compared to datasets used for training. This lack of generalization is a general problem in machine learning and will be tackled by training on more diverse data. Ultimately this will remove the need for fine-tuning on a local dataset as was performed in this research (Subsection 2.4).

4.4 Environmental Impact

This research involved creating machine learning models. The rise in popularity of artificial intelligence and machine learning does not come without consequences. A common problem of training models is the large amounts of electricity needed, therefore it is more and more important to take into account the environmental impact that is made during research that involves machine learning. As mentioned before, the Delft High Performance Computing cluster is used for training Whisper on the ATCO2 and ATCOSIM datasets. This computing cluster has an intelligent scheduler that minimizes the computing load by making optimal use of the available resources. By use of this scheduler, the environmental impact of the fine-tuning was taken down to a minimum. Future research should also allocate time to consider the environmental impact of their study.

4.5 Limitations

First of all, the main focus of this research was on the literal transcription (i.e., from speech to text). The concepts of call sign and command recognition, i.e., the understanding of speech, are briefly mentioned but are not researched in this work. The first step in ASR is creating a good transcription, later this can be extended by looking at the semantic level. As can be seen above, simpler applications will primarily rely on the word error rate, but more complex applications will rely on a good understanding of the speech.

Second, this work focused on the *creation* and *development* of automatic speech recognition models.

However, the *application* or *implementation* was out of the scope of this research. Though, it is at least as important. To have a working application, effort needs to be put into the actual implementation of the model. Not only into the literal real-time working of the model in an application but also into the long lists of regulations, hardware, and possibly weight requirements for applications. The model needs to fulfill all sorts of requirements to be implemented into an ATC system or aircraft.

Third, the accuracy and reliability of machine learning models are greatly involved by the hyperparameters. The hyperparameters will influence the learning process during the fine-tuning of a model. Wrong hyperparameters could result in under- or overfitting of a model. Hyperparameter-tuning is needed to overcome this problem. In this study, there has been a lack of focus on the tuning of hyperparameters.

Moreover, the model stability has not been assessed. There has been no focus on techniques like cross-validation to determine the generalization of the model. Despite this, the evaluation of the model trained on both ATCO2 and ATCOSIM does reveal some degree of generalization.

4.6 Future Work

Future work should move the focus from speech recognition to actual speech understanding. This means that the model should not only be able to recognize the speech but also understand the message it conveys. Good call sign and command recognition are, for some applications, far more important than having a low word error rate. This is especially true for more complex applications with harder performance and safety requirements. The development of ATC speech understanding is a big step toward high-demanding applications of ASR in the ATC domain.

A big challenge is the actual implementation of the model. As this research aimed to find the best available performance, the best available variant was used (Whisper Large V2). This logically contained the largest number of parameters, 1.5 billion. This large number of parameters required some very powerful hardware to run on. This can be a problem for real-time implementation. Then it is important to have a fast response time. Thus, a compromise needs to be made between powerful but complex hardware, and a quick response time. A possible solution could be to use a smaller model, such as Whisper Medium, which has 770 million parameters and a relative speed of two compared to Whisper Large V2 [9].

It should be investigated whether fine-tuning this model on the ATCO2 and ATCOSIM datasets would result in a WER that is close to the WER of the Whisper Large V2 model. In addition, the English-centered version of the Whisper Medium model could be taken to reduce the size of the model even more. This is trained only on English speech, but since ATC mainly concerns English speech, this could be a good solution. Another possible solution could be to use a refined version of Whisper called Distil-Whisper [30]. The main goal of Distil-Whisper was faster processing. It reduced overall processing times by up to 5.8 times while minimizing the performance decrease by at most 1% WER.

An assumption that has been made during the inference is that the only possible language is English, the model has been forced to use English since the ATC datasets mostly contain English speech. However, it could be of high interest to look at full multilingual speech recognition. In air traffic control, greetings, for example, are often spoken in a local language. Further, in countries where the development of English is below par, the amount of non-English speech is often higher. Therefore, it could be of great interest to look at multilingual speech recognition. For example, the recently launched Massively Multilingual Speech model could be used [31]. Enabling the recognition of speech in other languages than English or more accents could result in better recognition of non-English or accented speech.

5 Conclusion

This research focused on applying automatic speech recognition to air traffic control. Whisper was used, a large-scale weakly supervised automatic speech recognition model. The question to answer was: How can Whisper be used for application in air traffic control?

From the results presented in Subsection 3.2 it can be concluded that blank Whisper performs relatively well on the out-of-domain ATCO2 and ATCOSIM datasets. A word error rate of 24% and 17% respectively show that Whisper understands large parts of the ATC speech. Further, fine-tuning on those datasets significantly enhances the performance. Whisper sets the new state-of-the-art word error rates of 14% on ATCO2 and 1.2% on ATCOSIM. The model learned to comprehend ATC-specific vocabulary and, more importantly, phraseology. Lastly, from the validation results presented in Subsection 3.4, it can be concluded that applying Whisper in a real-world ATC environment is possible with promising results.

As seen in Subsection 4.2, there is a lot of potential for ASR to be applied in ATC. Some of the applications require a more robust performance than currently can be delivered, a long-term focus is of more importance there. On the other hand, the current performance of the created models already suffices for simple applications. Especially incident analysis, where post-processing is required instead of real-time processing, is an auspicious example.

All in all, it can be said that Whisper has already formed a solid foundation. However, fine-tuning Whisper on ATC can provide even better speech recognition performance in the ATC domain. It provides the drive needed to bring ASR technology to a mature and robust model for application in air traffic control. It has the needed performance to start reducing the workload of

air traffic controllers in the short term, using simple yet powerful applications. In the long term, it could have enough potential to make a significant difference in the way of working for air traffic controllers.

References

[1] R. Pieraccini, *From AUDREY to Siri: Is speech recognition a solved problem?* San Francisco, 2012. [Online]. Available: http://www.icsi.berkeley.edu/pubs/speech/audreytosiri12.pdf.

[2] A. Hannun, C. Case, J. Casper *et al.*, 'Deep Speech: Scaling up end-to-end speech recognition,' *arXiv preprint arXiv:1412.5567*, Dec. 2014. DOI: 10.48550/arXiv.1412.5567. [Online]. Available: https://arxiv.org/abs/1412.5567v2.

[3] D. Amodei, R. Anubhai, E. Battenberg *et al.*, 'Deep Speech 2: End-to-End Speech Recognition in English and Mandarin,' *33rd International Conference on Machine Learning, ICML 2016*, vol. 1, pp. 312–321, Dec. 2015. [Online]. Available: http://arxiv.org/abs/1512.02595.

[4] W. Chan, D. Park, C. Lee, Y. Zhang, Q. Le and M. Norouzi, 'SpeechStew: Simply Mix All Available Speech Recognition Data to Train One Large Neural Network,' *arXiv preprint arXiv:2104.02133*, Apr. 2021. DOI: 10.48550/arXiv.2104.02133. [Online]. Available: https://doi.org/10.48550/arXiv.2104.02133.

[5] S. Schneider, A. Baevski, R. Collobert and M. Auli, 'wav2vec: Unsupervised Pre-training for Speech Recognition,' in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2019-September, International Speech Communication Association, Apr. 2019, pp. 3465–3469. DOI: 10.21437/Interspeech.2019-1873. [Online]. Available: https://arxiv.org/abs/1904.05862v4.

[6] A. Baevski, H. Zhou, A. Mohamed and M. Auli, 'wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations,' *Advances in Neural Information Processing Systems*, vol. 2020-December, Jun. 2020, ISSN: 10495258. [Online]. Available: https://arxiv.org/abs/2006.11477v3.

[7] Y. Zhang, D. S. Park, W. Han *et al.*, 'BigSSL: Exploring the Frontier of Large-Scale Semi-Supervised Learning for Automatic Speech Recognition,' *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1519–1532, 2022. DOI: 10.1109/jstsp.2022.3182537. [Online]. Available: https://doi.org/10.1109%2Fjstsp.2022.3182537.

[8] J. Delua, *Supervised vs. Unsupervised Learning: What's the Difference?* 2021. [Online]. Available: https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning.

[9] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. Mcleavey and I. Sutskever, *Robust Speech Recognition via Large-Scale Weak Supervision*, 2022. DOI: 10.48550/arXiv.2212.04356. [Online]. Available: https://arxiv.org/abs/2212.04356.

[10] S. Badrinath and H. Balakrishnan, 'Automatic Speech Recognition for Air Traffic Control Communications,' *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2676, no. 1, pp. 798–810, Jan. 2022. DOI: 10.1177/03611981211036359. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/03611981211036359.

[11] V. N. Nguyen and H. Holone, 'Possibilities, Challenges and the State of the Art of Automatic Speech Recognition in Air Traffic Control,' *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, vol. 9, no. 8, pp. 1940–1949, Jun. 2015. DOI: 10.5281/zenodo.1108428.

[12] J.-P. Imbert, H. Christophe and J. Yannick, 'Think different: how we completely changed the visualization of Pseudo-Pilots,' in *Graphics Interface*, Halifax: Canadian Information Processing Society, Jun. 2015, pp. 257–264, ISBN: 978-099478680-7. [Online]. Available: https://enac.hal.science/hal-01166368/file/iipp_gi2015.pdf.

[13] R. Tarakan, K. Baldwin and N. Rozen, 'An Automated Simulation Pilot Capability to Support Advanced Air Traffic Controller Training,' in *The 26th Congress of ICAS and 8th AIAA ATIO*, Anchorage, Alaska: American Institute of Aeronautics and Astronautics, Sep. 2008, ISBN: 978-1-60086-997-6. DOI: 10.2514/6.2008-8897. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2008-8897.

[14] A. E. van der Groef, H. Nguyen and H. Vink, *Personal Conversation at Luchtverkeersleiding Nederland*, Oct. 2023.

[15] J. Zuluaga-Gomez, K. Veselý, I. Szöke *et al.*, 'ATCO2 corpus: A Large-Scale Dataset for Research on Automatic Speech Recognition and Natural Language Understanding of Air Traffic Control Communications,' *arXiv*, Nov. 2022. DOI: 10.48550/arXiv.2211.04054. [Online]. Available: https://arxiv.org/pdf/2211.04054.pdf.

[16] K. Hofbauer, S. Petrik and H. Hering, 'The ATCOSIM Corpus of Non-Prompted Clean Air Traffic Control Speech.,' in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco: European Language Resources Association (ELRA), Jan. 2008. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2008/pdf/545_paper.pdf.

[17] Š. Luboš, *Air Traffic Control Communication*, 2011. [Online]. Available: http://hdl.handle.net/11858/00-097C-0000-0001-CCA1-0.

[18] J. Zuluaga-Gomez, P. Motlicek, Q. Zhan, K. Vesely and R. Braun, 'Automatic Speech Recognition Benchmark for Air-Traffic Communications,' in *Proc. Interspeech 2020*, Shanghai, Oct. 2020, pp. 2297–2301. DOI: 10.21437/Interspeech.2020-2173. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-2173.

[19] V. R. Joseph, 'Optimal ratio for data splitting,' *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 15, no. 4, pp. 531–538, Aug. 2022, ISSN: 1932-1864. DOI: 10.1002/sam.11583. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/sam.11583.

[20] International Civil Aviation Organization, *ICAO Annex 10 Volume ii: Aeronautical Telecommunications*, 2001. [Online]. Available: https://www.icao.int/Meetings/anconf12/Document%20Archive/AN10_V2_cons%5B1%5D.pdf.

[21] T. Wolf, L. Debut, V. Sanh *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*, Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. DOI: 10.48550/arXiv.1910.03771. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[22] Delft High Performance Computing Centre, *DelftBlue Supercomputer (Phase 1)*, 2022. [Online]. Available: https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1.

[23] V. Panayotov, G. Chen, D. Povey and S. Khudanpur, 'Librispeech: an ASR corpus based on public domain audio books,' in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2015, pp. 5206–5210. DOI: 10.1109/ICASSP.2015.7178964.

[24] R. Ardila, M. Branson, K. Davis *et al.*, 'Common Voice: A Massively-Multilingual Speech Corpus,' in *Proceedings of the 12th International Conference on Language Resources and Evaluation (LREC 2020)*, 2020, pp. 4211–4215. DOI: 10.48550/arXiv.1912.06670.

[25] J. Zuluaga-Gomez, K. Veselý, A. Blatt *et al.*, 'Automatic Call Sign Detection: Matching Air Surveillance Data with Air Traffic Spoken Communications,' *Multidisciplinary Digital Publishing Institute Proceedings*, vol. 59, no. 1, p. 14, Dec. 2020, ISSN: 2504-3900. DOI: 10.3390/PROCEEDINGS2020059014. [Online]. Available: https://www.mdpi.com/2504-3900/59/1/14/htm%20https://www.mdpi.com/2504-3900/59/1/14.

[26] 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, p. 490, May 2023, ISSN: 22264310. DOI: 10.3390/aerospace10050490. [Online]. Available: https://doi.org/10.3390/aerospace10050490.

[27] Y. Lin, L. Deng, Z. Chen, X. Wu, J. Zhang and B. Yang, 'A Real-Time ATC Safety Monitoring Framework Using a Deep Learning Approach,' *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 11, pp. 4572–4581, Nov. 2020, ISSN: 15580016. DOI: 10.1109/TITS.2019.2940992.

[28] H.-H. Yang, Y.-H. Chang and Y.-H. Chou, 'Subjective measures of communication errors between pilots and air traffic controllers,' *Journal of Air Transport Management*, vol. 112, p. 102 461, 2023, ISSN: 0969-6997. DOI: 10.1016/j.jairtraman.2023.102461. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0969699723001047.

[29] M. Oda, 'Emergency! Do We Still Have to Speak English?: English as a Lingua Franca for Aviation,' in *unctional Variations in English: Theoretical Considerations and Practical Challenges*, R. A. Giri, A. Sharma and J. D'Angelo, Eds., vol. 37, Cham: Springer International Publishing, 2020, pp. 45–59, ISBN: 978-3-030-52225-4. DOI: 10.1007/978-3-030-52225-4{\textbackslash}{_}4.

[30] S. Gandhi, P. von Platen and A. M. Rush, 'Distil-Whisper: Robust Knowledge Distillation via Large-Scale Pseudo Labelling,' *arXiv*, Nov. 2023. DOI: 10.48550/arXiv.2311.00430.

[31] V. Pratap, A. Tjandra, B. Shi *et al.*, 'Scaling Speech Technology to 1,000+ Languages,' *arXiv*, May 2023. DOI: 10.48550/arXiv.2305.13516. [Online]. Available: https://doi.org/10.48550/arXiv.2305.13516.

Appendix A Datasets

Table 9: An overview of all the used datasets in the research.

	ATCO2	**ATCOSIM**	**ATCO2-ATCOSIM**	**ANSP**
Total Size (h)	1.1	10.46	11.56	3.00
Train Size (h)	0.86	8.37	9.23	2.38
Validation Size (h)	0.23	2.09	2.32	0.62
Total Samples	559	9,559	10,118	1,001
Train Samples	446	7,646	8,092	799
Validation Samples	113	1,913	2,026	202

Appendix B Models

Table 10: An overview of all the used or created models in the research. Note: "WLV2" stands for Whisper Large v2, all the created models are based on it. Due to security reasons, the models that have been fine-tuned on ANSP data are not released to the public. Yet, the other models can be found on the HuggingFace Hub.

Name	**Created from**	**Fine-Tuned on**
WLV2	-	-
WLV2-ATCO2	WLV2	ATCO2
WLV2-ATCOSIM	WLV2	ATCOSIM
WLV2-ATCO2-ATCOSIM	WLV2	ATCO2 & ATCOSIM
WLV2-ANSP	WLV2	ANSP
WLV2-ATCO2-ANSP	WLV2-ATCO2	ANSP
WLV2-ATCOSIM-ANSP	WLV2-ATCOSIM	ANSP
WLV2-ATCO2-ATCOSIM-ANSP	WLV2-ATCO2-ATCOSIM	ANSP

Appendix C Results

Table 11: All WER scores on the ATCO2, ATCOSIM, and ANSP datasets (validation split).

Model	**Dataset**	**Prompting**	**Normalization**	**WER (%)**
Whisper Large v2 - Blank	ATCO2	no	no	71,6192
		no	yes	29,0538
		yes	no	61,0765
		yes	yes	**24,0338**
	ATCOSIM	no	no	79,1119
		no	yes	17,9776
		yes	no	63,6183
		yes	yes	**16,7376**
	ANSP	no	no	78.4873
		no	yes	**32.0160**
		yes	no	68.7349
		yes	yes	35.0947

Continued on the next page

Continued from the last page

Whisper Large v2 - ATCO2	ATCO2	no	no	19,7064
		no	yes	17,8587
		yes	no	18,1495
		yes	yes	**14,6602**
	ATCOSIM	no	no	48,4812
		no	yes	**15,8392**
		yes	no	44,6147
		yes	yes	15,9515
	ANSP	no	no	40.1268
		no	yes	26.8650
		yes	no	38.5568
		yes	yes	**26.4506**
Whisper Large v2 - ATCOSIM	ATCO2	no	no	53,5142
		no	yes	**34,3403**
		yes	no	54,4039
		yes	yes	35,3621
	ATCOSIM	no	no	1,2502
		no	yes	1,2634
		yes	no	**1,1854**
		yes	yes	1,1932
	ANSP	no	no	63.7077
		no	yes	**51.7318**
		yes	no	66.1232
		yes	yes	56.1575
Whisper Large v2 - ATCO2-ATCOSIM	ATCO2	no	no	17,2598
		no	yes	14,5713
		yes	no	19,395
		yes	yes	**13,4607**
	ATCOSIM	no	no	**1,1669**
		no	yes	1,1792
		yes	no	1,2271
		yes	yes	1,2447
	ANSP	no	no	36.9263
		no	yes	23.9195
		yes	no	28.6685
		yes	yes	**22.9278**
Whisper Large v2 - ANSP	ANSP	no	no	26.3436
		no	yes	15.2457
		yes	no	24.4112
		yes	yes	**13.2771**
Whisper Large v2 - ATCO2-ANSP		no	no	23.3545
		no	yes	12.5814
		yes	no	23.3394
		yes	yes	**12.3742**
Whisper Large v2 - ATCOSIM-ANSP		no	no	24.0942
		no	yes	**13.2475**
		yes	no	25.2868
		yes	yes	13.9284
Whisper Large v2 - ATCO2-ATCOSIM-ANSP		no	no	26.1322
		no	yes	16.4594
		yes	no	23.3998
		yes	yes	**12.5518**

Part II

Preliminary Report
–
Previously Graded on AE4020

Abstract

The development of automatic speech recognition (ASR) models has been boosted over the last decades by the introduction of machine learning (ML). However, a persistent problem is the gap between small-scale supervised and large-scale unsupervised models. Recently OpenAI filled this gap with the introduction of Whisper: the ChatGPT among speech recognition models. The unique character of Whisper makes it highly interesting to research the potential of applying it in the air traffic control (ATC) domain. Historically, a lot of research has already been done on speech recognition techniques in the ATC environment. However, Whisper might form a basis for fundamental changes in the future way of working of air traffic control.

This research first assesses the performance of the out-of-the-box Whisper model. It finds that Whisper achieves a word error rate (WER) of 24% and 17% respectively on the commonly-known ATCO2 and ATCOSIM datasets. Based on these results, it proposes the fine-tuning of Whisper on the datasets in order to increase the performance of speech recognition in air traffic control communication. Further, it includes an outlook on the application of a fine-tuned Whisper model in the ATC domain.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Meaning
ABSR	Assistant-Based Speech Recognition
ACC	Area Control Centre
AM	Acoustic Model
ANSP	Air Navigation Service Provider
APP	Approach Controller
ASR	Automatic Speech Recognition
ATC	Air Traffic Control
ATCo	Air Traffic Controller
CTA	Control Area
CTR	Control Zone
CPDLC	Controller Pilot Data Link Communication
CPU	Central Processing Unit
CRR	Command Recognition Rate
DNN	Deep Neural Network
FFT	Fast Fourier Transform
FIR	Flight Information Region
FL	Flight Level
GAS	Gradient Accumulation Steps
GMM	Gaussian Mixture Model
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
HMI	Human-Machine Interface
HMM	Hidden Markov Model
IAF	Initial Approach Fix
ICAO	International Civil Aviation Organization
ILS	Instrument Landing System
LM	Language Model
LVCSR	Large Vocabulary Continuous Speech Recognition
ML	Machine Learning
MMS	Massively Multilingual Speech
NATO	North Atlantic Treaty Organisation
NLU	Natural Language Understanding
SID	Standard Instrument Departure
STAR	Standard Arrival Route
TMA	Terminal Control Area
TWR	Tower
UAC	Upper Area Control
UTA	Upper Control Area
VFR	Visual Flight Rules
VHF	Very High Frequency
WER	Word Error Rate
XML	Extensible Markup Language

List of Symbols

Symbol	Meaning
$b_{s,e}$	Evaluation Batch Size
$b_{s,t}$	Training Batch Size
D	Number of Deletions
E	Number of Epochs
$F1$	Call Sign Recognition Rate
f	Train-Test Split Ratio
g_s	Gradient Accumulation Steps
I	Number of Insertions
N	Number of Words in a Reference Text
n_e	Evaluation Samples
n_t	Training Samples
p	Probability
p_e	Number of Evaluation Points
S	Number of Substitutions
s_e	Evaluation Steps
s_t	Maximum Training Steps
t	Time
η	Learning Rate
ω	Frequency

1 Introduction

This report describes the ongoing research on applying a large-scale weakly supervised automatic speech recognition model in air traffic control. It particularly focuses on the progress so far and proposes plans to complete the research. That starts with the introduction of state-of-the-art automatic speech recognition (ASR) and the application of automatic speech recognition in air traffic control. This chapter closes with the research gap and describes the research goal.

1.1. Automatic Speech Recognition

Automatic speech recognition has been studied for a very long time already, with some research even dating back to the mid-20th century [1]. Back in that time, ASR research was based on using mechanical devices that pick up audio signals and performed filtering to recognize the 10 digits in the decimal system [2]. Soon after that, the first voice-activated typewriter was built, also using signal-filtering [3]. It was the point where dynamic programming was introduced when the interest in the technology began to grow. Using this approach, it was getting easier and easier to create models with a large vocabulary. With these growing language models (LMs), a new method was needed to recognize words fast enough. The solution for that was the introduction of statistical models. Specifically, the Hidden Markov Model (HMM) made its appearance in automatic speech recognition models [1]. Using the HMM approach, now a probability could be given for the occurrence of a certain word in an utterance instead of looking for sound patterns in the audio. Sometime later, Gaussian Mixture Models (GMM) were used to model the perceived states of the HMM model [2]. The combination of HMM-GMM models became the golden standard in ASR models and is still forming a big part of those models today. However, the introduction of deep learning can be seen as the next big thing in automatic speech recognition. Recent work done by several big companies such as Google [4], Meta [5], and Microsoft [6] makes it clear that the introduction of deep learning significantly improves the automatic recognition of speech.

One technique to construct automatic speech recognition models is by using supervised learning. This method results in end-to-end models that do not need any fine-tuning as will be explained later in chapter 4. Many successful ASR models exist that rely on the supervised learning approach. More noteworthy examples include DeepSpeech [7] which set a new benchmark for automatic speech recognition. They successfully built a model that could not only recognize speech in a clear and conversational manner but also in noisy environments. Reaching a word error rate of around 12% they even outperformed the larger publicly available speech recognition systems such as Apple Dictation, Bing Speech, and the Google speech API [7].

Further, SpeechStew is a more recent noteworthy example [8]. A common problem with supervised learning ASR models was the lack of data, SpeechStew tried to solve this by combining seven different datasets from several sources into one dataset. By training on this new, larger, dataset they reached near-state-of-the-art performance (in 2021). They saw that multi-domain training and transfer learning could result in WERs that directly competed with the latest work available [8].

Another approach to take is the approach of unsupervised learning resulting in the training on much

larger datasets as no labeled data is needed. Wav2Vec took this approach by training on around 1120 hours of unlabeled data [9]. Soon, a new iteration was created where Wav2Vec2.0 [10] increased the amount of unlabeled data to 53.2 thousand hours by using a pre-processing technique found in [11]. Still, this is nothing compared to unsupervised learning on one million hours of audio performed in BigSSL [12]. Since the scaling of data is extremely easy and the available computing power grew significantly over the past years, the performance of unsupervised learning ASR models experienced some great improvements.

DeepSpeech trained their model on around 5000 hours of data and even though SpeechStew mixed several datasets, this still resulted in 'only' 5140 hours of supervised data. On the other hand, large-scale unsupervised models like BigSSL still need fine-tuning (as said, this will be explained later chapter 4). Supervised models are often far more robust than unsupervised models as they are trained on actual labels. Due to the fine-tuning of unsupervised models, they often tend to perform much worse when tested on held-out datasets [13]. It can be concluded that neither supervised nor unsupervised models deliver really promising results due to their lack in size and robustness respectively.

Whisper suggests that an automatic speech recognition model should work reliably out-of-the-box, even in a multi-domain setup without the need for additional processing [14]. Therefore, it focuses on closing the gap between small-scale supervised learning and large-scale unsupervised learning ASR models. It does so by building a large-scale weakly supervised robust speech recognition model. The model not only focuses on extending the training data to 680.000 hours, but it also emphasizes that multitask and multilingual learning only increases the performance compared to previously existing models [14], leading to some distinct capabilities.

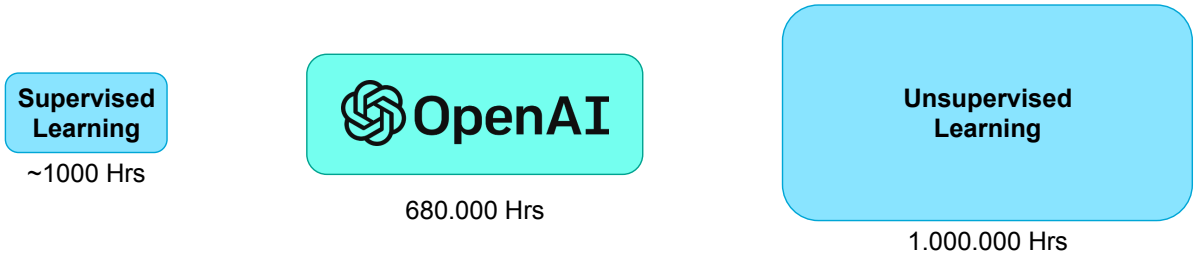


Figure 1.1: Whisper fills the gap between large-scale unsupervised learning and small-scale supervised learning.

The first property of the model is that the data used for training comes from a vast number of sources on the internet, therefore the model spans a very diverse set of audio. That makes that using the model on out-of-domain data should not lead to a significant drop in performance compared to the baseline scores of the model. Without having seen a certain dataset, 'zero-shot', it can still correctly transcribe speech in that dataset. This property makes it very appealing to use the model in air traffic control since ATC typically is an area where transcribed audio is scarce.

Another feature of the model is that it is trained on a lot of languages and dialects, it includes training on a total of 96 languages and dialects. The primary focus is English, but the multilingual capabilities could be useful as will be discussed in chapter 5. A model that performs far better in terms of multilingual capacity is the Massively Multilingual Speech (MMS) model by Meta-AI [15]. It focuses entirely on extending speech recognition, synthesis, and identification to a large number of languages. Reaching a total of 4000 languages. However, due to this attention on a massive amount of languages, MMS performs much worse on English than Whisper.

Third, speaker diarization is also trained during the creation of the model. This could too be useful in the air traffic control domain in order to determine whether speech is coming from an air traffic controller (ATCo) or a pilot and apply speaker-dependent processing.

1.2. Air Traffic Control Applications

The air traffic control domain has encountered multiple studies on the topic of automatic speech recognition. Especially in the last two decades, since the rapid development of automatic speech recognition. Works like [16], [17], and [18] have looked into applying ASR in the ATC domain. Numerous applications have been mentioned as having potential for ASR models.

1.2.1. Training and Simulation
One of them is the training and simulation application [19]. Here soon-to-be ATCos are trained in a simulated session where a pseudo-pilot receives commands from the ATCos and gives a response while 'controlling' the aircraft. An ASR model could be implemented together with a speech synthesizer to assist or even replace the pseudo-pilot. This could help in the development and training of more ATCos as there are ongoing operational challenges with the availability of these pseudo-pilots [20]. They often do the support in the training sessions next to their main job (as real-world pilots) which brings in lots of constraints.

1.2.2. Safety Monitoring Framework
Another approach that can be taken is the implementation of automatic speech recognition into air traffic control through a safety monitoring network. For example, the work done in [18] suggests using an ASR model which transcribes the ATCo's speech. The controller's intent is then extracted from this transcription and sent to a safety system. The safety system then checks whether the commands will lead to potentially dangerous situations by comparing them with radar data.

1.2.3. Operational Analysis
Other applications include the determination of the workload of an air traffic controller [21]. Since ATCos are the bottleneck in the operational capacity of the airspace, they are under immense pressure. Therefore, it can be of high value to correctly measure the workload of a controller by analyzing the speech. Further, analyzing the speech can also open the opportunity to study the compliance of the controller with certain standardized procedures performed in air traffic control as suggested in [17].

1.2.4. Data Link Integration
Further research that was conducted, focused on applying automatic speech recognition in a next-generation controller-pilot data link communication (CPDLC) system. The study concludes that applying automatic speech recognition and synthesis in combination with CPDLC in aircraft could have a lot of potential [22].

1.2.5. Implementation
Even though a lot of research has already been done on applying automatic speech recognition in air traffic control, almost none of the applications have been implemented in real-life. The only examples of an implemented application are in the training and simulation sectors. Here two works [23], [19] have been used actively in real-world training of air traffic controllers.

The lack of implementation could be explained by the high accuracy requirements and the unique challenges in the ATC context [24]. Although the performance of ASR systems has increased massively over the past years, the niche domain of air traffic control communications still is too hard of a challenge.

1.2.6. Datasets
To apply Whisper in a certain application, some transcribed ATC audio data is needed to assess the performance of the Whisper model. This performance assessment is crucial for selecting a suitable application. A problem that arises in a lot of studies is the lack of sufficient high-quality data. Some works attempted to build ATC speech corpora [25], [26]. However, only three of those are publicly available for use. The ATCO2 corpus [27], ATCOSIM corpus [28], and a corpus as part of the CLARIAH LINDAT-CZ project [29].

Several ASR models have been built and tested on ATC data already examples are the works [30] and [21]. The company of Airbus even held a competition to see the state-of-the-art of speech recognition in the ATC domain [31]. Using a dedicated dataset created in [25], the best scoring model achieved a word error rate of 7.6 % [31]. Since there are not many public datasets, these datasets can be seen as a benchmark measure for ASR models in the ATC domain.

1.3. Research Gap
Every speech recognition model is a unique model, however, it can be concluded that Whisper is the latest and greatest in ASR. By filling the gap between small-scale supervised learning and large-scale unsupervised learning with large-scale weakly supervised learning on a broad range of data, it

successfully seemed to fulfill its goal: creating a reliable out-of-the-box model that works well on out-of-domain data without the need for additional dataset specific fine-tuning. By setting a new standard, Whisper paves the way for endless new opportunities in the context of automatic speech recognition.

The distinct properties of Whisper make it an appealing choice for usage in the air traffic control domain. Although there are lots of interesting applications, to the writer's best knowledge only one of them has been implemented in a real working environment. One more thing that all these applications have in common is that they all focus on one side of the communication spectrum. On one hand, there is the ATCo side and on the other hand, there is the pilot side. It seems that almost all applications focus on the former. That brings unique opportunities to take the opposite perspective. Or, for the same reason, why not think outside the box and look at the communication spectrum as a whole?

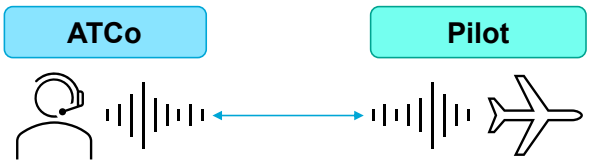


Figure 1.2: The communication spectrum in air traffic control.

For example, one of the many challenges faced by pilots and controllers is the lack of language proficiency. These inequalities in the level of English (or a local language) can lead to some serious miscommunications. Sending along a transcription of the speech over a data link and presenting it as some kind of subtitle could help in solving this problem.

It is also possible to go one step further. Air traffic control has relied on voice-based communication ever since its introduction. Yet, CPDLC made its entrance a few decades ago to declutter radio channels in congested airspace. However, CPDLC is used for some clearances and reporting only. This is to reduce 'heads-down' time in the cockpit. The next step in ATC communication might be a switch from voice-based to text-based communication. Then, one side of the line could speak (even in their local language) after which a transcript is made. This transcript is then sent over a data link channel, as they are much more reliable and less noise-susceptible than VHF (Very High Frequency) channels. At the receiving end, this transcript is then synthesized and presented to the receiving party (which could also be in a local language).

1.4. Research Goal

All in all, it is seen that Whisper is the next big thing with a combination of abilities that is never seen before. Whisper demonstrates that applying automatic speech recognition in (commercial) applications in the ATC domain is at the cusp of being technologically feasible. In terms of applications, a lot of research is still left open. This is especially true for the pilot side of the communication spectrum. Here, Whisper could pave the way for a fundamental change in the way of working in air traffic control.

Therefore, the goal of this research is to explore the possibilities of applying large-scale weakly supervised automatic speech recognition to air traffic control.

1.5. Report Structure

The rest of this report is structured as follows, the foundations of automatic speech recognition will be illustrated in chapter 2. Afterward, chapter 3 discusses the language processing of the transcriptions coming from the audio. An introduction to machine learning will be given in chapter 4. Possible applications will be discussed in chapter 5. Further, chapter 6 contains the experimental setup of the research. Logically, chapter 7 discusses the results and outcomes of the experiments, and chapter 8 contains the conclusion that can be drawn from the work done so far and makes an outlook to future challenges that may arise.

2

Automatic Speech Recognition

Automatic Speech Recognition, simply called ASR, is the recognition of words and sentences from spoken language. It is a technology that has been studied for a long time and has recently made some serious advancements. This chapter starts by illustrating the historical development of ASR. Furthermore, it describes the general working principle and shows the recent advancements in ASR technology.

2.1. Historical Development

The first automatic speech recognition machine dates back to the 1950s [1]. It then consisted of a mechanical device that only could recognize 10 digits (in English) by the use of filters. The recognition using filter-based mechanisms developed rapidly and in the 1960s, specialized hardware for the recognition of speech was built. Due to the application of dynamic programming, the vocabulary of speech recognition grew. It grew until a point where the current technology was not efficient anymore. In the 1980s, the first statistics-based technology made its appearance in the field of automatic speech recognition. Using a Hidden Markov Model, speech recognition made significant progress. Soon after the introduction of HMM, GMMs were combined with HMM-based speech recognition. After extensive research of the HMM-GMM framework in the following two decades, the framework's performance was approaching its maximum in the late 2000s. Luckily, at that time the phenomenon of deep learning, a machine learning principle, was brought to life. Using a combination of context-based Deep Neural Networks (DNN) and a Hidden Markov Model, Large Vocabulary Continuous Speech Recognition (LVCSR) models could be made that outperformed the traditional HMM-GMM-based models [2].

2.2. Working Principle

The overall working principle of an ASR system is that it processes spoken language in such a way that it predicts the spoken words in the audio fragment. In the end, it all comes down to the probability of certain words being in the audio fragment. The probability of these words determines the performance of the model. If the model is accurate enough, the probability of the predicted words will be high and vice versa. To predict the words that are said in an audio fragment, four parts of processing are needed. The first part is the extraction of certain 'features' from the audio fragment, these are contained in a set of feature vectors. The second part maps the 'phonemes' (or even sub-phonemes) of the audio fragment to a sequence of feature vectors, this happens in the so-called acoustic model (AM). The third part is the pronunciation model or lexicon, it maps the pronunciation (i.e. the phonemes and sub-phonemes) to a sequence of words. The last part is the language model, which simply contains the vocabulary of a certain language and determines the most likely sequence of words given the audio input and context [32]. An overview of the working principle of automatic speech recognition can be found in Figure 2.1.

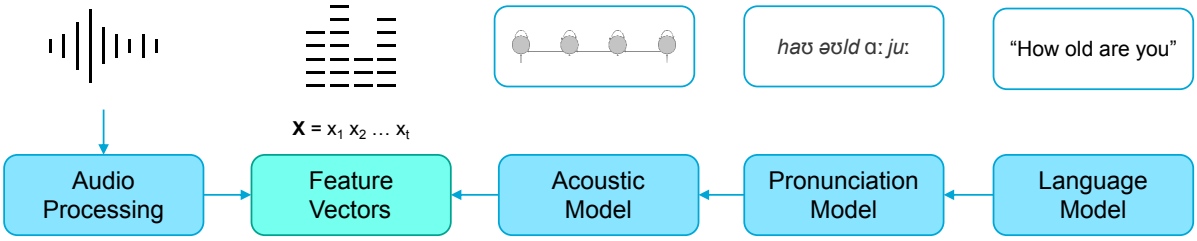


Figure 2.1: The working principle of automatic speech recognition. The transcription is created by matching the feature vectors of the audio with those determined by the probabilistic model.

2.2.1. Feature Extraction

Each audio fragment has a set of features that is unique to the audio fragment, this can be described by a feature vector. The first step in extracting this feature vector is transforming the audio signal from the time domain to the 'frequency domain' using a Fast Fourier Transform (FFT). Since an audio signal is built up from several sinusoids, each having its own frequency and amplitude, the frequency domain will give us much more information about the audio signal than the time domain. This can be shown mathematically by:

$$f(t) = A_0 + \sum_{n=1}^{\infty} \left(A_n \cos(\frac{2\pi nt}{P}) + B_n \sin(\frac{2\pi nt}{P}) \right) \tag{2.1}$$

It can be seen that every signal, can be reconstructed by adding up an arbitrary number of sinusoids and a constant. Each sinusoid has its own frequency and amplitude. A visual representation of the Fourier transform is illustrated in Figure 2.2.

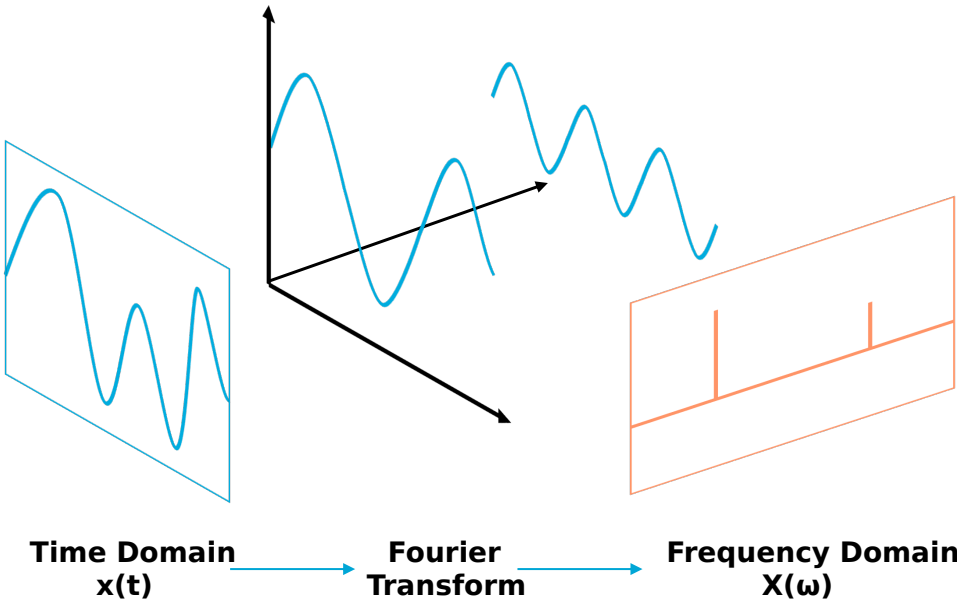


Figure 2.2: The Fourier transform maps a signal from the time domain (t) to the frequency domain (ω). Every signal can be described by an infinite sum of sinusoids, each with its own frequency and amplitude.

The points of interest in the frequency domain are the frequencies at which the peaks occur (in the frequency domain) and the amplitude of these peaks. By analyzing the amplitude of the peaks, the strength of the signal can be calculated. By graphing the amplitudes of a signal against the frequencies, the so-called spectrum of a signal is created. In Figure 2.2 the spectrum is represented by the orange graph.

Now, the only problem is that each spectrum is only valid for a certain point in time of the audio signal. As the audio signal is continuous, so is the spectrum. That means that the spectrum will

change over time. If the audio fragment is divided into smaller segments, each a few milliseconds long, and a spectrum is made from each segment, then one can represent the complete audio signal through a spectrogram. A spectrogram is nothing more than a whole set of FFTs put together. The only difference is the representation of the values, the frequency is now represented by the y-axis. The amplitude is represented by the z-axis (color scale). Lastly, the added dimension, time, is represented by the x-axis.

As the spectrogram contains all the necessary information of an (audio) signal, the feature vector is simply represented by the peak amplitudes, the frequencies at which they occur, and the points in time at which they occur. Below, in Figure 2.3, a spectrogram is visualized. It can be seen that speech recognition now comes close to image recognition as each phone has a unique representation in a spectrogram. The transcript corresponding to the spectrogram below is *alitalia two three four bonjour squawk five seven seven five*.

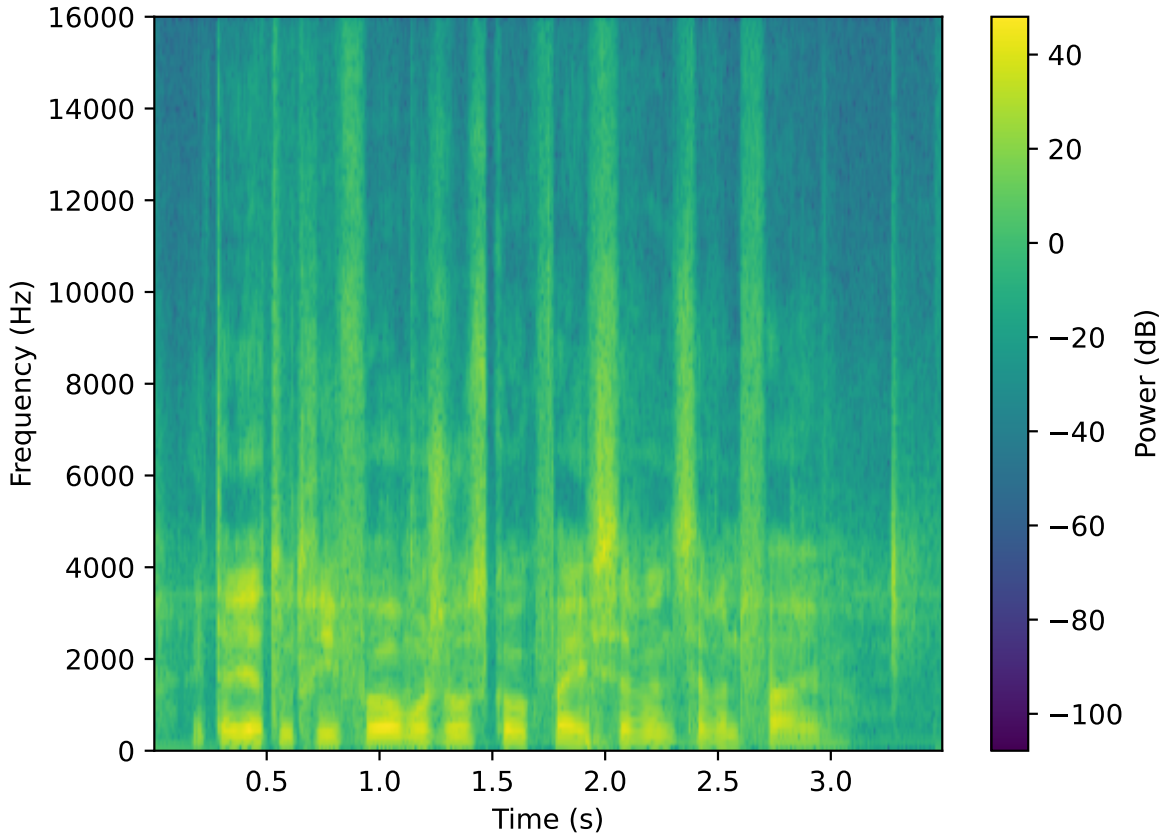


Figure 2.3: A spectrogram, in essence, is a visualization of audio. The corresponding transcript to this spectrogram is *alitalia two three four bonjour squawk five seven seven five*. The numbers in the squawk code are represented by the bars in the figure between 1.7 and 2.8 seconds.

2.2.2. Acoustic Model

As said, the task of the acoustic model is to map a sequence of phonemes and sub-phonemes to a sequence of feature vectors. In ASR, the acoustic model will try to find the highest probability of feature vectors (**X**) given an input of phonemes (**W**), in mathematical terms, this can be described by finding:

$$arg_W max\ P(\mathbf{X}|\mathbf{W}) \tag{2.2}$$

An acoustic model is typically represented by a Hidden Markov Model. HMMs are symbolized by a set of nodes, each having a probability of staying at that node or transitioning to another node. Such a node is called a state, a state can be (and most likely is) hidden. That is where the name 'hidden' Markov model comes from. Each (hidden) state can be observed by an observation. In this case, the

states are the 'phonemes' and the observations are the feature vectors belonging to these phonemes. The connections between the nodes represent the probabilities of staying at a certain node or moving to the next node, the so-called transition probability. Furthermore, the probability that a specific node will emit a certain observable is the so-called emission probability. In the case of ASR, a sequence of nodes is represented by a sequence of phonemes (and in the observable domain by a sequence of feature vectors). Now what the AM does is that it finds to most likely way that a sequence of phonemes flows through the chain and thereby determines the sequence of feature vectors that belongs to the sequence of phonemes [32]. An overview of the Hidden Markov Model can be found in Figure 2.4.

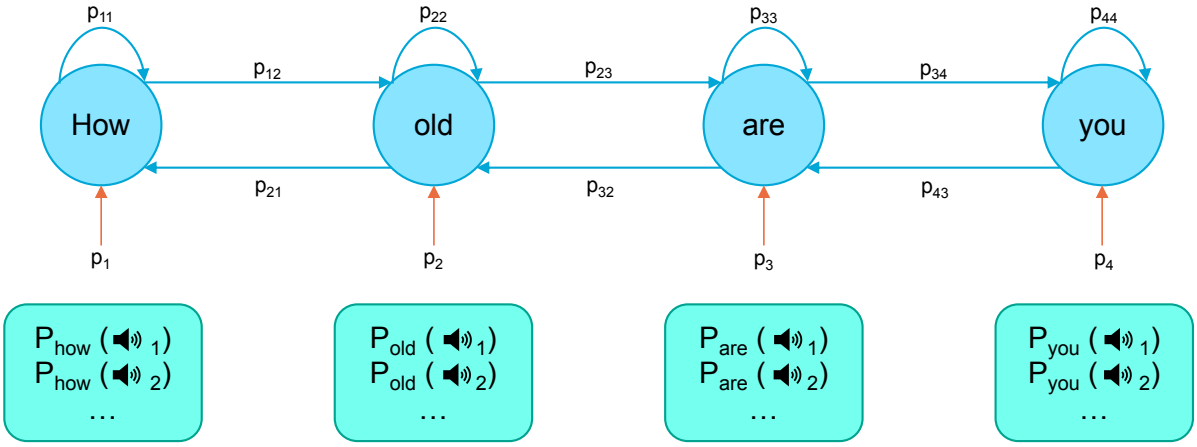


Figure 2.4: An overview of the Hidden Markov Model from the perspective of speech recognition. Here the transmission probabilities are represented by the lowercase letter p. The emission probabilities are represented by the capital letter P.

2.2.3. Pronunciation Model

The pronunciation model is the model that pairs a word with a set of phonemes and sub-phonemes that form the pronunciation of the words. This model is also called the lexicon as it contains the collection of words (lexis) with the speech of these words.

2.2.4. Language Model

The language model is the model that contains the actual vocabulary of the ASR. It is simply a collection of pieces of text from a certain (or multiple) language(s). What a language model does is that it tries to predict the next word in a sequence of words based on the previous N words. A language model is often described as an 'N-gram' language model, here the N refers to the number of words that are used to determine the probability of the next word in the sequence. The goal of the language model can be described mathematically by:

$$p(\mathbf{W}) = p(W_t|W_{t-1}W_{t-2}\ldots W_{t-N}) \tag{2.3}$$

Here, **W** is the sequence of words, t is the point in time, W_t is the next word in the sequence, and W_{t-1} until W_{t-N} is the set of the N previous words. When the dataset that is used is large enough, the chance of having a certain word can easily be calculated by counting the number of instances that a certain word occurs after a given sequence of N words. For example, the probability of having 'you' after 'How old are...' is then calculated as:

$$p(you|How\ old\ are) = \frac{count(How\ old\ are\ you)}{count(How\ old\ are)} \tag{2.4}$$

If the dataset that is used to build the language model is large enough, so will the reliability of the language model. By using the web, a large enough dataset can be built. Also, one could say that having a higher N number will result in a much more precise language model. That is not necessarily the case, if the next word will be predicted on the previous 10 words (e.g. 10-gram LM) then the dataset will need to be much much larger than for example having a 5-gram LM in order to have a reliable prediction of what the next word will be [33].

2.3. State-of-the-Art

Automatic speech recognition technologies have come a long way since their first development. The current models involve machine learning and try to provide an end-to-end solution for transcribing the audio. The newest of such models is called Whisper, an automatic speech recognition model built by OpenAI [14].

Whisper is the next big thing among automatic speech recognition models. It efficiently fills the gap between small size supervised and large scale unsupervised learning models by creating a large scale weakly supervised model. They made a model using 680.000 hours of labeled audio. The distinct properties of Whisper make it an appealing choice for ASR applications. The goal of Whisper is to be a model that does not need any environment-specific fine-tuning afterward, e.g. to work in a zero-shot environment [14].

Further, Whisper works in a different way than people may be used to. Instead of voice assistants such as Amazon's Alexa, Apple's Siri, and Google's Assistant, Whisper works offline. The model will be downloaded to the device where the speech recognition will be performed. The voice data that is processed by Whisper all happens on the device itself instead of being sent to the cloud. This also has a big advantage in terms of privacy.

Whisper is created using the Python programming language. Specifically, the model is created using the PyTorch framework. With the correct requirements installed (primarily PyTorch, NumPy, and FFmpeg), the model can be inferred using a Python script.

3

Language Processing

After the transcripts have been generated, some processing is needed. How can it be determined whether the transcript is correct? Further, is it the transcript that is of importance or is it the aspects in the transcript that are important? Those are all questions that will be discussed in this chapter. This chapter contains the processing of the 'language' in the transcript, e.g. the processing of the outcome from the automatic speech recognition.

3.1. Assessment Criteria

Now that the transcripts are generated, it is time to determine how well the Whisper model works by assessing the transcripts. This is done by looking at the assessment criteria. Three aspects are of importance when it comes to these transcripts in the air traffic control domain. The first aspect is the word error rate, the second is the call sign recognition rate and the third is the command recognition rate (CRR). The importance of each of those criteria depends on the application in the ATC domain, for example, for speech analysis, the word error rate is more important but for assistant-based speech recognition (ABSR) systems the call sign and command recognition rates are far more important than the word error rate. More on that in chapter 5. In this section, all three assessment criteria will be illustrated together with their shortcomings.

3.1.1. Word Error Rate

The most simple way to assess whether an automatically generated transcript is correct is by comparing it with a manually made transcript (e.g. the 'label' of the data, sometimes also called the 'ground truth'). This can be done on a per-word basis but is generally done on a per-sentence basis, where every word in two sentences is compared instead of every character in two words. The most common metric for assessing the correctness of the transcript is called the word error rate, it says how many of the words in the automatic transcript ('the transcript') are also in the manually given reference text ('the reference'). The WER can be calculated with the following equation:

$$WER\ (\%) = \frac{S + I + D}{N} \tag{3.1}$$

In this equation, S is the number of substitutions, I is the number of insertions and D is the number of deletions in the transcript. Furthermore, N is the total number of words in the reference. So for example, the word error rate of the transcript 'Can I get to-dos?' with reference 'Can I get two shoes?' will be 40% since the transcript contains one substitution, zero insertions, and one deletion whilst the total number of words in the reference is five.

A common problem that arises is that reference transcripts often have everything written out in words and have no punctuation at all. The Whisper model has a certain level of intelligence through which, for example, it writes numbers in number format instead of in text format and adds all needed punctuation, accents, etc... to the transcript whereas the manually created reference almost always lacks this kind of intelligence. Therefore it can be very hard to systematically determine the WER scores of the transcripts. This problem will is visualized in Figure 3.1 and will be addressed in section 3.2.

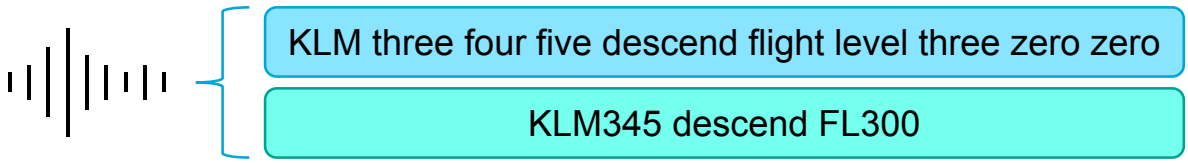


Figure 3.1: Strictly taken, the word error rate would not be 0%. However, the two sentences have the exact same meaning.

A third problem that occurs is that the transcription model does not have any contextual awareness. Since the air traffic control audio contains very location-specific naming of for example waypoints (i.e. ARTIP, VALKO, TULIP), IAFs (Initial Approach Fixes), and local entities (i.e. Praha Radar) which are not part of any standard language model, it is very hard to correctly transcribe these items. Luckily some input can be given to Whisper when transcribing audio, more on that in section 3.3.

3.1.2. Call Sign Recognition Rate

Since all the audio data is directly related to air traffic communication, almost all of them contain a call sign by which the air traffic controller addresses the pilot or by which the pilot identifies itself to the ATCo. The call sign is one of the most valuable items in the audio as it can be used to identify the aircraft to or from which the communication is. Also, it can be used to find out who is speaking at a certain point in time. An ATCo begins its phrase with the call sign while a pilot ends its communication with the call sign. That makes it key to correctly extract the call sign from the automatically generated transcript. The score which indicates how many of the aircraft call signs are correctly extracted from the transcript is called the 'F1-score'.

A problem that arises when extracting the aircraft call sign from the transcript is that there are many ways of expressing the same aircraft call sign. 'Lufthansa three delta echo', 'DLH three delta echo', 'hansa three echo', 'three delta echo', and 'delta lima hotel three delta echo' are all examples of expressing the same call sign (e.g. DLH3DE). Even-though air traffic control communication is standardized as specified in the International Civil Aviation Organization (ICAO) guidelines [34], there are hundreds of ways of expressing a certain phrase whilst keeping the same meaning. This problem will partially be addressed later on in section 3.2 and section 3.3.

3.1.3. Command Recognition Rate

An utterance in air traffic communication mainly consists of two parts, the first is the call sign which was discussed in the previous section, and the second is the command or instruction (see Figure 3.2). The command is the action that the pilot has to perform or the validation that the pilot gives to the ATCo when complying with the instruction. An example of such a command may be: *descend flight level one zero zero* or *contact Vienna center on one three four decimal three five*. When a pilot has been given a command by an ATCo, he will do the readback of the command. This is used to confirm that the pilot will comply with the given instruction: *KLM681 climb to flight level three seven zero*. *Flight level three seven zero, KLM681*. Here, the same holds as does for the call sign, the command can be expressed in multiple ways. Even though there are fewer ways of expressing a command, it is still more than one. Furthermore, sometimes there is no command at all. For example, a simple *willco* or *negative* but also some non-scripted conversations about special circumstances or greetings do occur often. The command recognition can be graded by the so-called Command Recognition Rate, which states from how many percent of the transcripts the command (if there is any) has correctly been recognized.

Since there is some kind of logic in an utterance in terms of call signs and commands, it should be possible to have some sort of systematic and mathematical approach to the extraction of these items from a generated transcript. That will be discussed later in chapter 6.

3.2. Normalizing

As described in the previous section, it can be very hard to compare the transcript with the ground truth as there will be a difference in 'writing style'. In order to correctly calculate the word error rate, some sort of normalization is needed to bring both pieces of text, transcript or hypothesis and reference, to the same format. This normalizer will contain multiple steps that need to be performed on both pieces

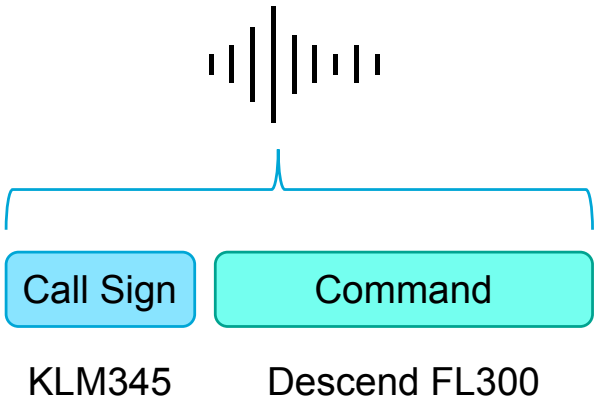


Figure 3.2: A utterance predominantly consists of a call sign and a command.

of text.

The first step is to remove any non-alpha-numerical characters as exclamation marks and question marks, in order to have some 'flat text'. Further steps include rewriting numbers as digits instead of text; separating numbers and text; separating numbers into digits; remove any spoken separators ('decimal', 'comma', 'point'). It is also important to take into account the NATO (North Atlantic Treaty Organization) alphabet, for example, Whisper will sometimes recognize 'Alpha' as 'A' so the transcript must also have this intelligence. Also, some terminology like flight level is often transcribed as 'FL' instead of 'flight level'.

3.3. Prompting

In order to give Whisper more contextual awareness, a feature called 'prompting' can be used. This is an extra input that can be given to the model when transcribing a certain audio file. It is worth taking note that prompting can only be used when inferring the model, it can not be used when training or fine-tuning the model. The format of the prompt is a free text field. When a prompt is given, during inference, it will be used as some sort of priority vocabulary. The model will first try to see if the audio matches any of the words in the prompt before continuing with the 'normal' vocabulary. Also, it will try to find words that are closely related to (e.g. in the same domain as) those in the prompt. Therefore it would make sense to input a list of hard-to-transcribe items combined with the context of the audio.

The first item to include in the prompt is the context, e.g. 'Air Traffic Control Communication'. After it can be extended with the airlines and or call signs of the aircraft that are in the controlled area. This can then be appended by the list of waypoints and entities (e.g. Praha Radar) in the controlled area. Further, a list of terminology like 'ILS', 'flight level', 'wilco', and 'affirm' can be added to further improve the recognition of these words. At last, the nato alphabet can be added in order to put focus on the recognition of those words.

4

Machine Learning

Machine learning significantly improved the development and performance of automatic speech recognition. One could say that it really formed the basis of ASR in general. Since a large part of this research is based on machine learning models, it makes sense to have some theory on the topic. This chapter will first give some background on Machine Learning and illustrate the main working principle, afterward, it focuses on the fine-tuning technique of existing ML models. Last, it also explains the purpose of and ideas behind hyperparameters and how they influence the fine-tuning process.

4.1. Background

Machine learning is a term used for having a computer, the machine, learn a specific task. This happens by creating a model, often some kind of network, with one input and one output. Then a dataset is fed to the model where it will have to predict the output based on the input. The model will learn to correctly predict the output by iterating multiple times over this dataset. Each time a piece of the dataset is sent to the model as input, the model will adapt its internal parameters in order to match its predicted output with the given output in the dataset. At the first iteration, the parameters will be changed much more than at the last iteration, as the predicted output of the model gets closer and closer to the 'ground truth' as provided by the dataset. That happens up until a point where the difference between the predicted output and the ground truth is almost zero, then the model has been trained sufficiently and accurately represents the desired connection between input and output.

The creation of a model by machine learning mainly involves two flavors, supervised learning, and unsupervised learning. Both these flavors, their differences, and their similarities will be illustrated in the next two subsections.

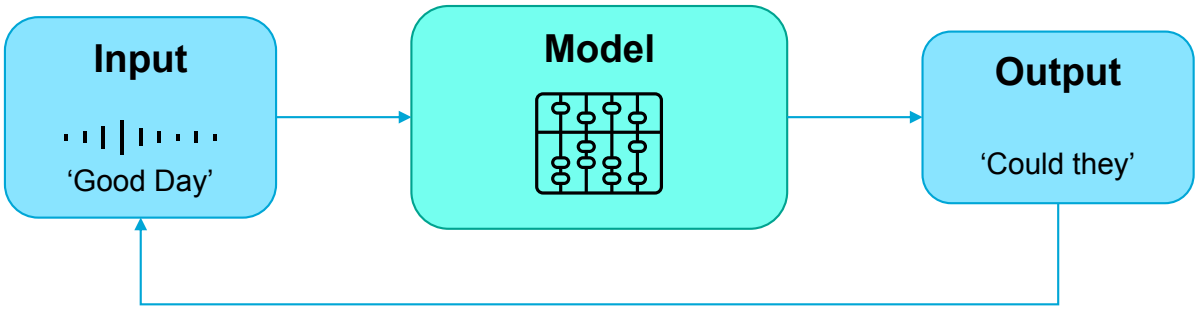


Figure 4.1: A graphical representation of machine learning. Input is given to the model from which it will try to predict the output. In each iteration, the internal parameters of the model are updated based on the difference between the predicted output and the correct output.

4.1.1. Supervised Learning

Supervised learning is based on the point that for every sample (input) in the training dataset, a label exists with the corresponding output. During training a sample is given to the model, together with the ground truth of the corresponding sample (e.g. the label). The model then has to predict the label of the input for each and every sample, which creates the estimated outputs. Based on the difference between the estimated outputs and the ground truth, the model will adapt its internal parameters. Eventually, convergence will be reached.

This approach to machine learning involves a lot of manual interaction to be able to train the model. The dataset needs to be labeled for each and every sample, therefore these datasets are often smaller in size. The advantage of supervised learning is that the model produces a usable output. For every single input the model will be able to produce a label. Thus, it needs no processing after the training, e.g. the model can be used as-is.

4.1.2. Unsupervised Learning

Unsupervised learning is the exact opposite of supervised learning as here there are no labels at all in the dataset. The model will simply try to group together several inputs that look similar. It tries to discover patterns in the data without the need for human interaction [35].

As the approach of unsupervised learning does not have a need for any labeled datasets, the datasets are often much larger as it is easy to accumulate a lot of the data, for example from the internet. Also, if the model is finished training and an input is given to the model it will only output to which cluster or group of data the input probably belongs. That is why an unsupervised model still needs some processing after training. All the clusters or groups of data need to be labeled. The difference between supervised and unsupervised machine learning is visualized in Figure 4.2.

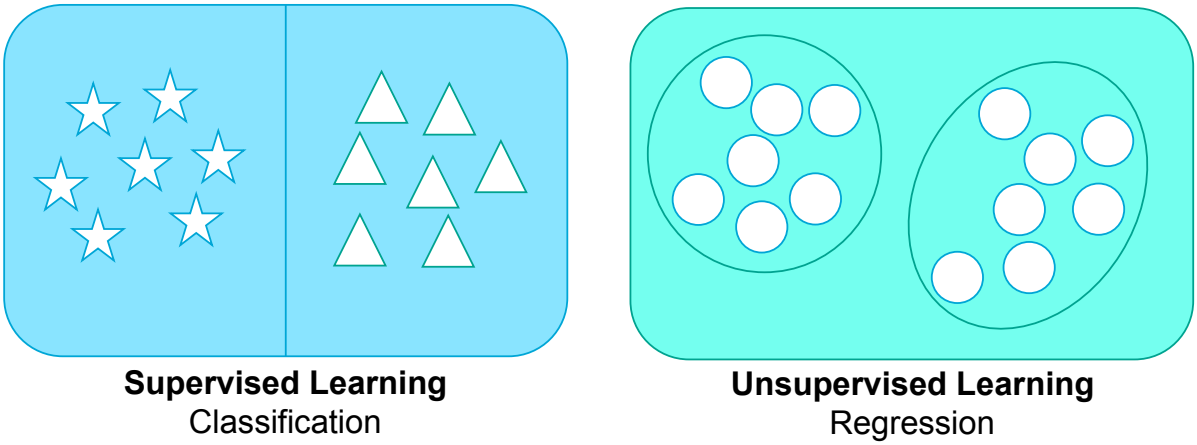


Figure 4.2: A visualization of supervised learning and unsupervised learning. The difference is in the labeling of the data. A supervised model will give a classification to an input whereas an unsupervised model will separate different inputs based on a regression.

As an example, let us take a model that predicts whether a certain picture is from a dog or a cat. A supervised model will need a collection of pictures of dogs and cats with for each picture the label whether it is a dog or a cat. When trained, the model will output either the label 'dog' or the label 'cat'. When training a model without supervision, it will create two groups, one with pictures of dogs and one with pictures of cats. When a picture is given to the final model, it will output whether it belongs to 'group 1' or 'group 2'. As that does not say something useful, an extra step is needed where the mapping is done between the groups and the labels 'dog' and 'cat'.

4.2. Fine-Tuning

If an existing machine learning model is trained on a very diverse set of data, it is then also assumed to have a good performance on a very diverse dataset. However, the performance on out-of-domain data may still not be sufficient. Then, the performance on this out-of-domain data could be increased by training the existing model on this dataset. That is what is called fine-tuning. In essence, it is training

an existing machine learning model on a new dataset. The model will then start to adapt its parameters toward the newly learned dataset.

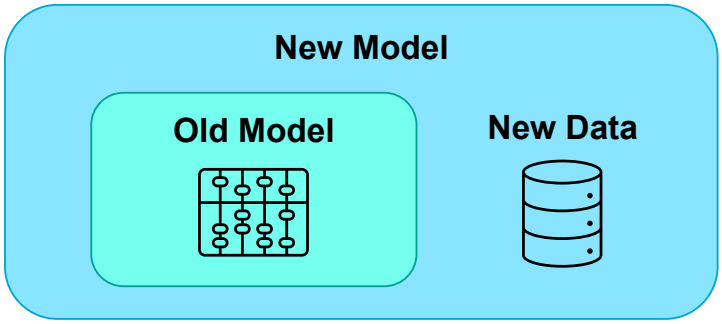


Figure 4.3: A schematic overview of the fine-tuning principle. In essence, it is training an existing model on a new dataset in order to create a model that performs better in the domain of the newly trained dataset.

Since the main idea behind Whisper is to have an out-of-the-box reliably working automatic speech recognition model, it is trained on a very diverse dataset. Thus, it could be of high interest to fine-tune Whisper on ATC data to, further, improve the recognition of words in air traffic control communications.

4.3. Hyperparameters

One can imagine that training a machine learning model requires a lot of parameter tweaking. Each of the parameters has its own effect on the training process. As they do not concern the internal model but rather the broader picture they are called the hyperparameters. They each control the learning process of the model [36]. The most important hyperparameters and their effects on the learning process are listed in table 4.1.

Table 4.1: The most important hyperparameters and their effect on the learning process of machine learning models [36].

Hyperparameter	Symbol	Influence on the learning procecss
Train-Test Split Ratio	f	The percentage of the data that is used for training and validation
Optimization Algorithm	-	The algorithm that is used in order to find the best gradient
Learning Rate	η	Sets the (initial) pace at which the optimization algorithm learns the parameter estimation
Training Batch Size	$b_{s,t}$	The number of samples that are sent to the model each step during training
Gradient Accumulation Steps (GAS)	g_s	The number of steps before the model parameters are updated
Max Training Steps	s_t	The maximum number of parameter updates during training
Training Samples	n_t	The number of samples in the training data
Evaluation Samples	n_e	The number of samples in the evaluation data
Number of Epochs	E	The number of iterations that the complete training data passes the model
Evaluation Steps	s_e	After how many steps the model is evaluated using the validation dataset
Evaluation Batch Size	$b_{s,e}$	The number of samples that are sent to the model each step during evaluation
Number of Evaluation Points	p_e	The number of (check)points at which a model is evaluated

Some of the parameters are related to each other. For example, the number of epochs is defined as:

$$E = s_t \frac{b_{s,t} g_s}{n_t} \tag{4.1}$$

Whereas the train-test split ratio is defined as the relation between the number of samples used for training and for evaluation:

$$f = \frac{n_t}{n_t + n_e} \tag{4.2}$$

And the number of evaluation points can be described by:

$$p_e = \frac{s_t}{s_e} \tag{4.3}$$

5

Air Traffic Control Applications

This chapter describes the possible applications in the air traffic control domain for ASR models. It starts by illustrating the working environment of air traffic control and the areas of application. It then presents applications that have already been implemented in real-life and possible future applications that have not been implemented yet. At last, it finishes off with a description of the different available air traffic control datasets.

5.1. Working Environment

Air traffic control is the control and guidance of air traffic in a certain geographical area. Often, each country has its dedicated airspace for which it is responsible. The largest division of airspace is called the Flight Information Region (FIR) and spans (a piece of) the whole country. As this airspace is too large to control without any structure, it is divided into several areas, each having its type of controller.

First of all, the control zone (CTR) spans the area around an airport. It is therefore controlled by the air traffic control tower (TWR) at the airport. When an aircraft departs or approaches an airport it flies through the terminal control area (TMA) which is adjacent to the CTR. It is controlled by the so-called 'approach' controller (APP). The actual airspace where flights follow the highways in the sky is the control area (CTA), which is controlled by the area control center (ACC). If an aircraft climbs to higher altitudes, it transitions from the control area into the upper control area (UTA) which is controlled by the upper area control center (UAC) [37]. One UAC often controls the airspace above multiple CTAs. The division of airspace is visualized in Figure 5.1.

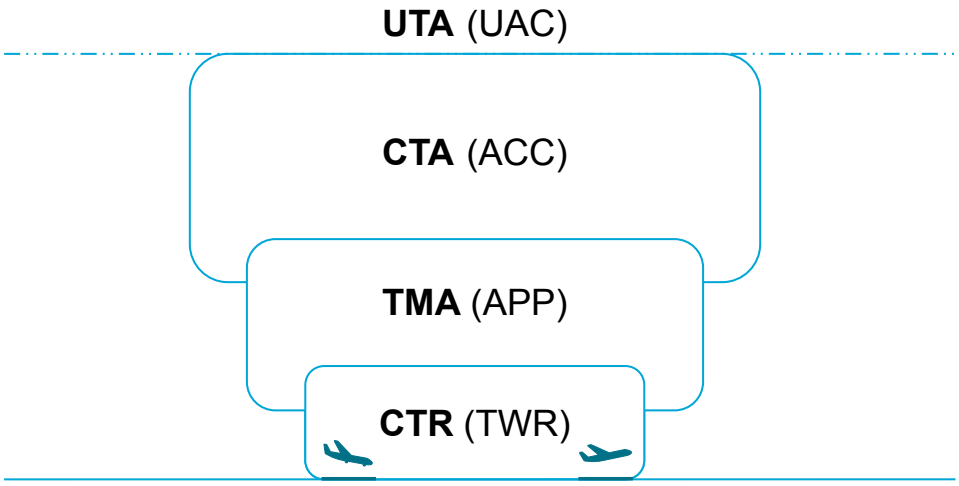


Figure 5.1: A graphical overview of all different divisions of airspace in a flight information region.

Even though all air traffic control communication has to adhere to the ICAO guidelines [34], each airspace has its own style of communication. The higher airspace, UTA, is less cluttered than the lower airspaces. This is due to the fact that the in this airspace the aircraft primarily follow fixed routes according to their flight plan. Examples of communication in the UTA are *KLM six eight two descend flight level three five zero* and *Transavia five six four three contact Marseille on one two six decimal one five zero*.

The ACC is a more congested control center. The area controllers have their hands full on merging several streams of incoming traffic to a set initial approach fixes. In addition, they guide outgoing traffic along their standard instrument departure (SID) routes. At last, they also guide the overflying traffic that passes the CTA on their route. Some examples of communication in the CTA are: *Martinair three four seven direct RIVER* and *Amsterdam Radar, Easyjet two niner niner two is inbound HELEN 2 alpha.*

Further, the approach controllers are merging the incoming streams from the ACC into one or two streams toward the runways that are in use for landing. Additionally, they guide the outgoing traffic toward their designated standard arrival routes (STARs). They control aircraft just after being airborne until they approach the borders of the TMA at around 8000 ft. In addition, they control aircraft that are delivered at one of the IAFs until they are on final approach to the runway. So an example of a command given by an approach controller is *KLM five niner two cleared ILS one eight right.*

At last, the tower controls everything at and around the airport. They primarily control clearances, for example, *Lufthansa three delta echo cleared for takeoff runway two seven* or *Swiss three seven two seven, Amsterdam Ground, Runway three six left, taxi via alpha five, bravo, zulu, victor, and hold short of three six left on victor four*.

5.2. Current Applications

Applying automatic speech recognition in air traffic control requires some serious robustness due to the high required performance and highly involved safety risks. One place where this is of less importance is the field of simulation and training. In a simulated environment, it has less importance and impact if a speech recognition model makes some mistakes. That is presumably also why the only place in air traffic control where speech recognition is applied is in the field of simulation and training.

An example of implementation is made by the *Ecole Nationale de l'Aviation Civile*, the national civil aviation school in France [23]. They reimagined the current human-machine interface (HMI) of the pseudo-pilots acting in the simulation session. There, automatic speech recognition was implemented as an assistant-based system in the HMI of an ACC controller. Normally if an ATCo gives a command to a certain aircraft, the pseudo-pilot monitors the given command in the order stack for execution. The role of the ASR model was to prefill the order stack based on the ATCo speech.

While the previous example only covers ASR as an assistant for the pseudo-pilot, it is also possible to completely remove the pseudo-pilot as presented by the Center for Advanced Aviation System Development [19]. The enrouteTrainer uses automatic speech recognition and speech synthesis to form an 'entity' for the replacement of pseudo-pilots. Although this would suffice for simpler training sessions, it still lacks the full 'human behavior'. Features like unruly pilot behavior or communication errors can be implemented based on a mathematical model, but still would only be a very basic implementation. The human is a much more complex system than a simple mathematical model.

5.3. Future Applications

As described in chapter 1 there are numerous fields of applications in air traffic control where ASR can play a role. The most promising example is assistant-based speech recognition where automatic speech recognition is used to assist the air traffic controller. Several works successfully implemented ASR deeply into the ATCo HMI resulting in massive workload reduction [38], [39].

Applications like ABSR do not rely anymore on word error rates. Yet, call sign and command recognition are of much greater importance. The rapid development in the use of machine learning brings great improvement in the field of natural language understanding (NLU) in air traffic control. Several methods exist that can successfully extract the call signs and commands from annotated speech [40], [41]. These great prospects make it highly likely that ABSR will be used in numerous ATC centers in the nearby future.

An additional example of an application is the presence of an ASR model in a safety framework. It has been shown that implementing ASR in a safety framework could result in a lower ATCo workload,

increased safety, and decreased misunderstanding [18]. However, the reliance on a computer model for the sake of safety still has its risks. It must be 100% reliable to be put in the foundation of air traffic control. At this moment in time, it still does not achieve this level of robustness [42]. Although it could have a significant meaning in air traffic control in the future, safety monitoring systems will not be forming the basis of ATC in the near future.

The goal of applying ASR in ATC can also take a different approach. As described in chapter 1 almost all of the applications in the literature focus on the air traffic controller side of the communication spectrum. Looking at the pilot side of the spectrum or the whole spectrum, in general, could open numerous new fields of application. Currently, a big problem in the communication between pilots and ATCos is the difference in language proficiency. Especially in non-western countries, where the level of English speech is below par, the understanding of speech is a big problem.

Possibly an ASR system could transcribe a certain utterance from the sender and send it along a data link channel to the receiving party. Using this transcription, the level of misunderstanding and wrong conceptional analysis of the speech will be much lower. An advantage of this possible application is that the audio can be captured and recognized at the source, resulting in less noise and thus a better transcription.

Another application could look at changing the long-lived method of voice-based communication. It could transcribe a message and then send the transcription instead of the voice itself. Where at the receiving end, a speech synthesis model would transform the text into a spoken message. This could even happen over digital data link channels to reduce possible noise compared to old-school VHF channels. Since Whisper has also been trained massively on translational data, this application could have the potential to let people speak in their local language. The audio will then be transcribed into English or synthesized from English into a local language.

5.4. Datasets

To test the ASR model and develop applications, ATC-related datasets are needed. The only problem is that there are not many ATC speech datasets available publicly. Either they are private, or they are behind a paywall. The only three public datasets that exist and are free to use are from the ATCO2 project [27], the ATCOSIM project [28], and the LINDAT project from the Charles University in Prague, Czech Republic [29]. Table 5.1 shows the datasets' specifications.

Table 5.1: The ATC audio datasets that are available, together with their specifications

Dataset	Hours	Languages	Geographical Source
ATCO2	1.1	En	Czech-Republic, Bratislava, Switzerland, Sydney
ATCOSIM	10	En	Eurocontrol Experimental Centre France
ZCU-CZ-ATC	13.2	En	Prague Region

The ATCO2 project aims a creating a platform to collect, organize and process air traffic communication data [27]. The latest release of the dataset (2022) contains pilot and controller voice communication in the English language. The data is collected from several airports scattered mainly across Europe and accounts for a total duration of around 1 hour of audio, making it relatively small. The full version of the ATCO2 data totals at around 1500 hours of audio, however, it is not freely accessible.

ATCOSIM is an abbreviation for Air Traffic Control Simulation and is a speech corpus of ATCo-only data. This dataset, made in 2008, contains roughly 10 hours of pure speech in 52 hours of audio recordings of air traffic controllers. It is all obtained at the Eurocontrol Experimental Centre in France. The dataset only contains English speech, but the speech is spoken by non-native speakers [28].

Finally, the ZCU-CZ-ATC dataset is a dataset created in 2011 by the research department of Language Technologies, Arts and Humanities (LINDAT) at Charles University in Prague, Czech Republic. It contains only English language and the total length of the audio is 13 hours. All the speech is transcribed and labeled manually. The research department plans to release a bigger dataset in the future [29].

All three datasets bring the total available speech data to almost 25 hours of transcribed audio. The audio is only available in the English language with some local accents. The audio data is gathered mostly from the European continent as can be seen in Figure 5.2 and involves ATCos and pilots. It can be argued that testing the Whisper model on the current datasets does not accurately give a

representation of the performance for general use worldwide. It would be preferred to have a very diverse dataset that contains audio fragments from all over the world, however, this is still a utopia.

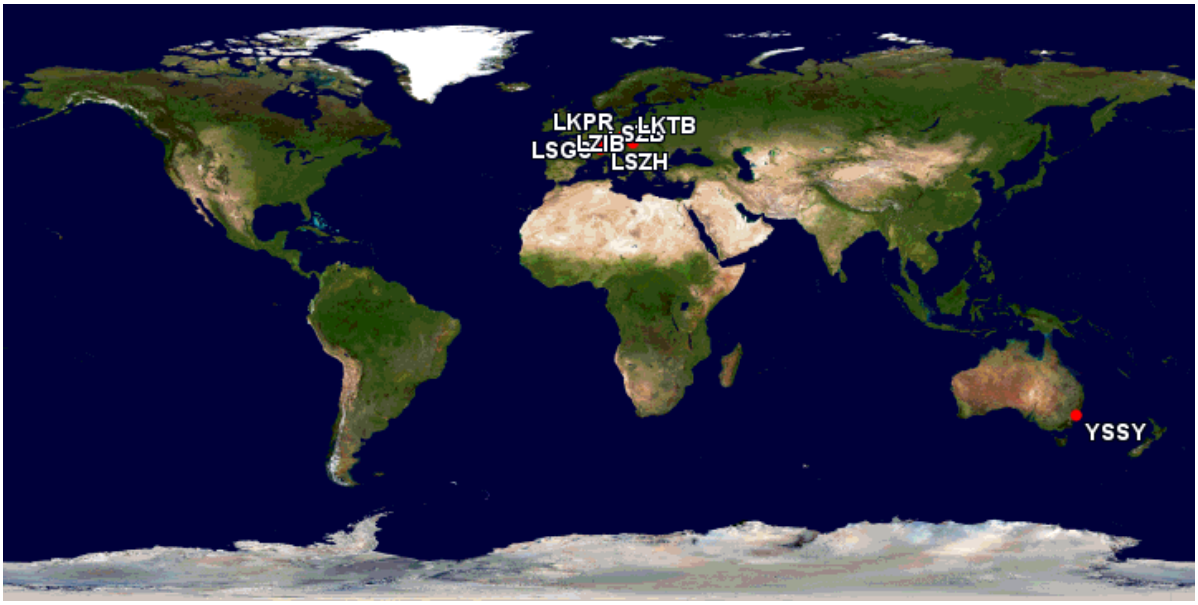


Figure 5.2: The audio data in the publicly available ATCO2 is primarily coming from the European continent.

6

Experiments

This section will cover the past and future experiments in the research. It starts with describing the performance assessment of the out-of-the-box Whisper model. Then it continues with illustrating the fine-tuning part where the blank model will be fine-tuned on ATC-specific datasets. Further, it discusses the research into applying the final ASR model to the ATC environment. At last, it illustrated the required hardware that is used.

6.1. Performance Assessment

The first step in the research is to determine whether Whisper performs as well as it says. By transcribing the datasets using the blank Whisper model, a baseline will be set for the rest of the study. To have this baseline, the audio files in the datasets mentioned in the previous chapter are transcribed, and from these transcriptions, the word error rate is calculated.

6.1.1. Data Preparation

The first step in transcribing the audio files is to pre-process the data in a format that can easily be used by the model. Each dataset is different, thus each dataset requires separate pre-processing. As Whisper will handle the standardization of the audio (like re-sampling to the correct frequency), the only work will be on extracting the actual ground truth from the datasets. For example, some of the datasets use an XML (Extensible Markup Language) format for the reference text, from there the plain text needed to be extracted.

Another step that is taken is the creation of a so-called split in the data. The data is divided randomly into two parts. One part is labeled as 'training data' and contains 80% of the audio files, the other part is called 'validation' and contains the remaining 20% of the files. This split is originally created with the purpose of fine-tuning the model, however, it can also be very useful for comparison of the performance of Whisper.

6.1.2. Transcribing

Then, the actual audio files can be transcribed. This is simply done by creating an inference code in Python. Whisper has multiple versions, each version is of a different size and has a different amount of internal parameters. Since the focus is on transcribing the audio as perfectly as possible, the largest version (*large-v2*) is used. This is the most complete version of the model and contains around 1.5 billion parameters.

A systematic approach is taken to transcribe each file separately, this is done by writing a *for loop* in Python. In addition, to maximize the WER scores, automatic language detection is disabled and the language is forced to English. The input of the model is the audio file, possibly in addition to the prompt, whereas the output of the model is the transcription of the audio file.

After all the files are transcribed, and normalized (dependent on the experiment), the word error rate is calculated by comparing each transcript with its corresponding ground truth that is extracted from the dataset. The experiment is repeated for every setup: with/without normalization and or prompt and on the whole dataset or only on the validation split.

6.1.3. Normalization

After transcribing the files, it is time to compare the hypothesis with the reference. As said in subsection 3.1.1 it can be very hard to directly compare Whisper's generated hypothesis with the ground truth. Therefore a process called normalization is used, here both pieces of text are flattened out into the same format.

The Whisper package itself comes with a standard normalizer that removes items like interpunction and non-alphanumerical characters. However, that will not suffice. As the speech contains air traffic control communications, more aspects are of importance for a correct WER calculation. An example is the mapping into the NATO alphabet where 'A' is represented as 'alpha', 'B' as bravo, etc. Another step to do is mapping the airline call sign designator to the actual airline itself. Sometimes Whisper is smart enough to transcribe, for example, *Eurowings one two alpha* into *EWG 12A*. Therefore a mapping need to be made from the call sign designator to the actual airline name.

To determine the steps that need to be done in the normalization, a batch of 10 audio files is transcribed. From the difference between the hypothesis and the reference, it is determined what can be added to the normalizer to correctly calculate the word error rate.

6.1.4. Prompting

Another problem that arises with transcribing the audio files is the lack of context awareness as described in subsection 3.1.1. This is solved using a process called prompting. Whisper can give some extra input arguments that need to be taken into account when transcribing audio.

Again, a batch of 10 audio files is transcribed multiple times. In each iteration, a new prompt is given to find some sort of prompt that would minimize the word error rate. The first step was to just mention the context of the audio, e.g. 'air traffic control communications'. This was then expanded in each iteration.

6.1.5. Variables

The inputs for the process of transcribing are the audio file itself and, if used, the prompt that was created. Now, for the ATCO2 dataset, each audio file contains a supplementary file that lists all possible airlines, call signs, entities, etc... that were nearby at the moment of recording. Thus, for testing the performance on the ATCO2 dataset, a prompt was created using these files. For ATCOSIM, on the other hand, the prompt just includes the terminology, context, and NATO alphabet as mentioned in the previous subsection. Even though the prompt shares a connection to the audio file, for the ATCO2 dataset, these variables are independent as they are created separately from each other.

Now, the output of Whisper is, of course, the transcription. Since it is created based on two independent variables, it makes sense that it is a dependent variable. Then, it is paired together with the reference or ground truth of the audio file and the word error rate is calculated. Thus, the WER is (heavily) dependent on the transcript and ground truth making it a dependent variable. It is yet to discover whether the WERs of Whisper on the ATCO2 and the ATCOSIM datasets are dependent or independent.

6.1.6. Hypothesis

Since Whisper has been trained on a large and diverse quantity of audio data, it is assumed that it will have decent performance on both the ATCO2 and the ATCOSIM datasets. A long-time ruling model, Wav2Vec2.0, has been tested on the ATCO2 data to see how Wav2Vec2.0 performs on out-of-domain datasets. They achieved a word error rate of 24.7% [30]. It is hypothesized that Whisper will achieve a comparable WER on the ATCO2 dataset. It is hypothesized that Whisper will have a lower WER on the ATCOSIM dataset. This is supported by the fact that ATCOSIM contains less noisy data and clearer speech since it is captured at the source.

Further, it is expected that the effect of normalization will be bigger compared to prompting. Whisper is trained intelligently on a diverse set of data. Thus, it should be able to produce intelligent transcriptions. The datasets, on the other hand, always have simple 'flat' text transcriptions. Thus, it is highly likely that a lot of normalization needs to be applied in order to have a comparable transcript. At last, the diverse training data will probably reduce the effect of and need for the use of prompting.

At last, it is anticipated that the relative effect of normalization and prompting will be comparable in both datasets. This could be enforced by the fact that both datasets cover the same domain. Take note that it is about the relative effect since the result of these datasets is independent of one another.

6.2. Fine-Tuning

Now that a baseline performance has been set, the model will be fine-tuned to create an ASR model that focuses on transcribing ATC data. The blank Whisper *large-v2* model will be used for the fine-tuning as again this is the most complete model and thus will presumably deliver the best performance. Fine-tuning is done on two datasets: ATCO2 and ATCOSIM. These are selected since they are the easiest to work with and greatly represent a diverse set of ATC environments such as TWR, APP, and ACC. Further, they also represent both the pilot and the ATCo. In addition, it contains the best audio quality and the ATCO2 dataset also includes radar data from the moment of recording.

In order to fine-tune the model on a particular dataset, a division need to be made between training and validation data. This so-called split is set to be 80% for training and 20% for validation. In the dataset, a random list of all audio files is created. From this list, 80% of the items are put in the training split and the other 20% of the items are put in the validation split.

The fine-tuning itself will result in two models, it is yet to be determined which model will be used. Later, it can also be decided to fine-tune on a combination of the ATCO2 and ATCOSIM datasets.

Several iterations of fine-tuning will happen, each with different hyperparameters. From there it will be determined which parameters are best suited for the fine-tuning of each model.

Fine-tuning the model will happen for around 100 epochs, where the training batch size will be 16 samples and the validation batch size will be 8 samples. Further, the number of gradient accumulation steps will be set to 1. For optimization, the Adam Weighted (AdamW) optimizer will be used and the initial learning rate is set to 1e-5.

6.3. Application

After the model has reached decent performance, it is time to implement it in some kind of application. As the experiments here are heavily dependent on the specific application, it is impossible to already determine what exact experiments will be performed.

However, what can be said is that the model needs to be run in real time without the need for heavy hardware. Therefore, the size of the model can be optimized in the experimental phase. This can be done by either fine-tuning smaller versions of Whisper or by the use of quantization, which recalculates the model parameters in a data-saving format.

6.4. Hardware

As one can imagine, inference or fine-tuning of a machine learning model could require some serious computing power. Since a simple laptop will almost certainly not have this required power, a high-power computing cluster is used. The Delft High Performance Computing Centre as part of the university facilities is used [43]. It contains multiple computing nodes with high-end CPU and GPU hardware that enables the research but also massively reduces the computing time compared to 'simple' computers.

7

Results

This section will describe the current and future results and outcomes of the research. It starts with presenting the performance assessment of the out-of-the-box Whisper model. Then it will describe the future results that will come from fine-tuning the model and researching the application of the final model. It finishes by explaining how the model will be verified and validated.

7.1. Performance Assessment

The performance of Whisper on the ATCOSIM and ATCO2 datasets has been determined to set a baseline for future experiments. These datasets have first been processed in such a way as to be saved in the HuggingFace hub. Then the inference could easily be done in a systematic way using a Python script. As said, the processes of normalization and prompting can have a great effect on the WER scores. Therefore, some tests are done based on 10 files from which the most optimal normalizer and prompt are constructed empirically.

7.1.1. Normalization

Normalizing the reference and generated hypothesis can have an immense effect on the word error rate. By comparing the manual and the generated transcription for 10 files it is decided what can be done in order to decrease the WER. The Whisper model itself also comes with a built-in normalizer that can be used, this formed the baseline of the normalization experiment.

This normalizer, called *EnglishTextNormalizer*, already is quite extensive. For example, it replaces contractions as *haven't* with the full words *have not*. Further, it applies British-American word mapping to bring everything into one format. At last, it also converts any spelled-out numbers into Arabic numbers.

Further additions to the standard normalizer include the mapping of loose characters into the NATO alphabet (i.e. A C D becomes alpha charlie delta), this is extended with some NATO similarities (i.e. alfa becomes alpha and gulf becomes golf). Another addition is the identification of airline call sign designators and replacing them with the full airline name (i.e. EWG becomes eurowings).

7.1.2. Prompting

Another feature of the Whisper model is the use of prompting. This can be used to create contextual awareness. Again, the most optimal prompt was created using an iterative process on the 10 audio files. The experiment started by bringing context, the initial prompt stated 'air traffic control communications'. Later, this was extended by general aviation terminology such as 'flight level', 'ILS', 'VFR', and 'squawk'. An addition was done by implementing the NATO alphabet to further emphasize that it is used in the audio.

The prompt used when transcribing an audio file is utilized as some kind of extra vocabulary, therefore the hard-to-transcribe pieces are the most valuable to put in the prompt. Therefore, the prompt was extended to include all the local airlines, waypoints, and entities that occurred in the 10 audio files. This resulted in a massive reduction in WER. However, in order to do this, all the airlines, waypoints, and entities need to be known beforehand. Also, it requires manual labor to list all these items

from the ground truth transcriptions. When applying automatic speech recognition in real-time, it could only work if there exists some augmentation system that would include this data based on a preset list of waypoints or radar data for example. The results of the experimenting with the normalization and prompting can be found below in Figure 7.1 and Table 7.1.

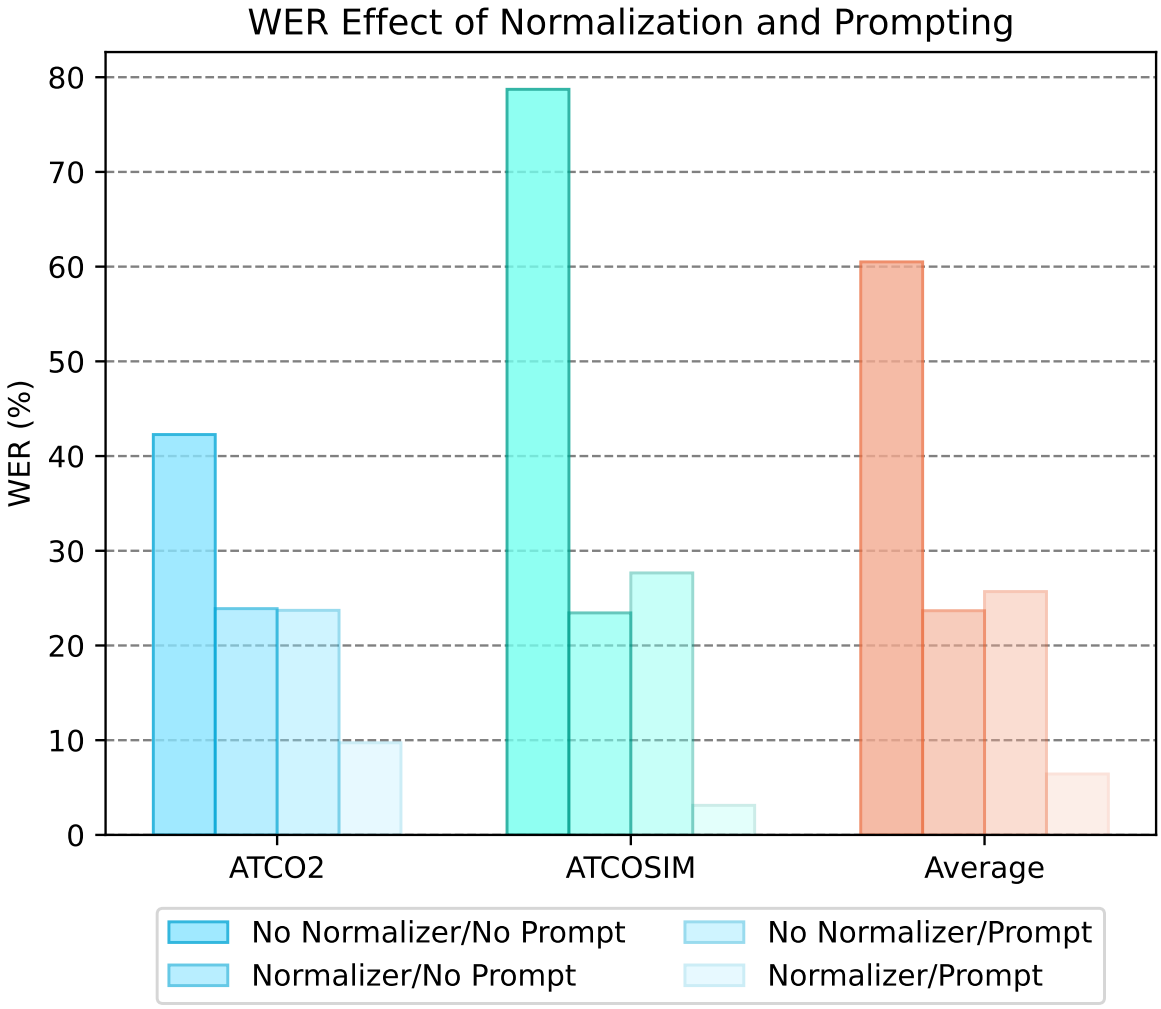


Figure 7.1: Normalization and prompting can massively reduce the word error rate. From left to right, the bars represent each combination of normalization/prompting.

From the results, it can be seen that the influence on the word error rate caused by the normalization and the prompting is approximately the same. Yet, applying the combination of normalization and prompting is what really makes the difference. The WER on the ATCO2 data reached 9,73% whilst the WER on the ATCOSIM data reached 3,12%. That makes an average word error rate of 6,43%.

Table 7.1: Normalization and prompting can massively reduce the word error rate.

Normalization	Prompt	ATCO2-ASR	ATCOSIM	Average
Basic	No	42,27 %	78,72 %	60,50 %
Yes	No	23,89 %	23,44 %	23,67 %
Basic	Yes	23,71 %	27,66 %	25,69 %
Yes	Yes	9,73 %	3,12 %	6,43 %

7.1.3. Word Error Rate

After the experimentation with normalization and prompting, the baseline performance of the Whisper model was set. This was all done using the *large-v2* version as it is the most complete model version. For each dataset and split, a separate test has been done. The test has been performed in multiple iterations, to test the model on all possible combinations of normalization and prompting. As can be seen in Table 7.2 and Figure 7.2 the word error rate without any normalization/prompting is significantly higher than with normalization and or prompting.

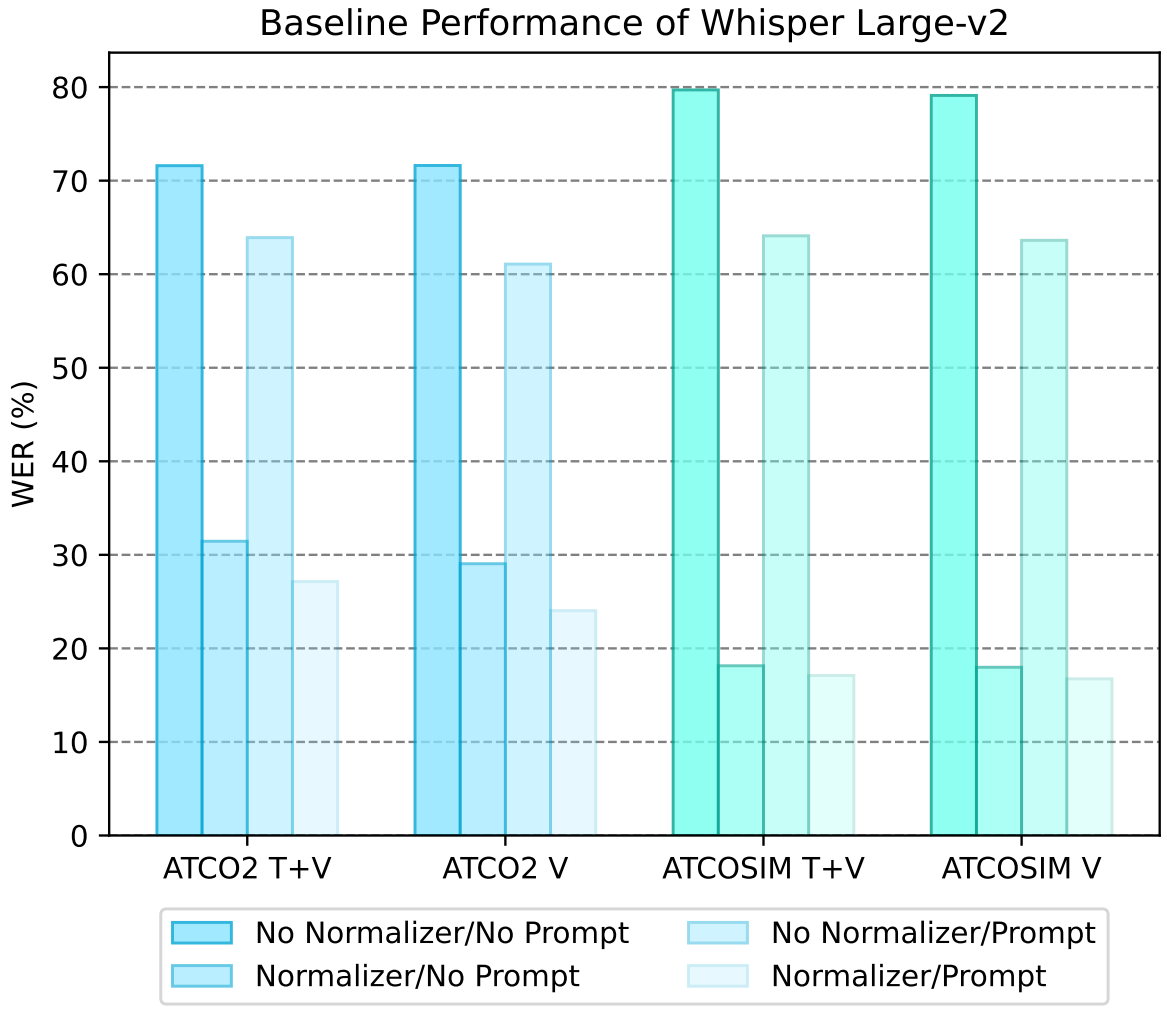


Figure 7.2: The baseline performance of the Whisper Large-v2 model. The test is performed on the whole dataset (T+V) and the validation split only (V). For comparison, the pre-trained Wav2Vec2.0 model reached a WER of 24.7% on the ATCO2 T+V set, which is comparable to Whisper's performance [30].

The best available performance of *Whisper large v2* on the complete ATCO2 dataset is just above 27%. On the other hand, the best available performance on the whole ATCOSIM dataset is around 17%. Even though the model performs slightly better on the validation split versus the whole dataset, the difference in performance is insignificant. That confirms the point that the split in the data is created using a random process.

Another notable result is that normalization has the largest effect on the word error rate. The increase in contextual awareness leads to a smaller reduction of WER than the application of a normalizer. These results could confirm the claim that Whisper has 'intelligence' in its transcriptions. Further it also slightly confirms that Whisper has been trained on a very diverse dataset as the prompting has no immensely big effect on the WER.

Furthermore, it can be seen that Whisper achieves approximately the same performance as the earlier reported 24.7% WER of pre-trained Wav2Vec2.0 on the ATCO2 T+V dataset. That confirms

Table 7.2: The baseline performance assessment of the Whisper model.

Model	Dataset/Split	Prompting	Normalizing	WER
Large v2	ATCO2-ASR/train+validation	N	N	71,60%
Large v2	ATCO2-ASR/train+validation	N	Y	31,45%
Large v2	ATCO2-ASR/train+validation	Y	N	63,91%
Large v2	ATCO2-ASR/train+validation	Y	Y	27,14%
Large v2	ATCO2-ASR/validation	N	N	71,62%
Large v2	ATCO2-ASR/validation	N	Y	29,05%
Large v2	ATCO2-ASR/validation	Y	N	61,08%
Large v2	ATCO2-ASR/validation	Y	Y	24,03%
Large v2	ATCOSIM/train+validation	N	N	79,70%
Large v2	ATCOSIM/train+validation	N	Y	18,14%
Large v2	ATCOSIM/train+validation	Y	N	64,10%
Large v2	ATCOSIM/train+validation	Y	Y	17,10%
Large v2	ATCOSIM/validation	N	N	79,11%
Large v2	ATCOSIM/validation	N	Y	17,98%
Large v2	ATCOSIM/validation	Y	N	63,62%
Large v2	ATCOSIM/validation	Y	Y	16,74%

our hypothesis as stated in subsection 6.1.6. Whisper only slightly has a higher WER, but this will presumably be decreased during fine-tuning.

At last, the difference between the ATCO2 and ATCOSIM datasets is clearly visible. The audio quality of ATCOSIM is much better, resulting in a significantly lower WER compared to the Whisper's performance on ATCO2. From there, it can be concluded that the performance of Whisper (e.g. the WER) on both the ATCOSIM and ATCO2 datasets are independent of each other.

7.2. Future Results

As mentioned, the above results just form the baseline for this research and only give an indication of how the bare model performs on the selected datasets. The next step is to build on top of this model by fine-tuning it on the selected datasets. The results and outcomes of the fine-tuning and the application of the model on the ATC domain are discussed in this section.

7.2.1. Fine-Tuning

The *large-v2* model will be used for fine-tuning on the datasets of ATCO2 and ATCOSIM. This will ultimately lead to two new models. The table below contains each of the fine-tuning processes together with their hyperparameters that will expectedly be used. A future option would be to even fine-tune the model on a combined dataset of ATCO2 and ATCOSIM in order to broaden the recognition of ATC speech.

Table 7.3: The presumed fine-tuning processes with their respective hyperparameters.

Model	Dataset	s_t	n_t	$b_{s,t}$	g_s	E
Large v2	ATCO2-ASR	50000	7646	16	1	104,63
Large v2	ATCOSIM	2800	446	16	1	100,45
Large v2	ATCO2-ASR/ATCOSIM	TBD	8092	TBD	TBD	TBD

7.2.2. Application

The desired outcome of the application phase is a model that is ready to be implemented in a real-world scenario. Ideally, this includes some sort of test setup where the model can be tested. One requirement for the model is that it must be able to run on low-level hardware.

7.3. Verification and Validation

Verifying the model and validating the results is of great importance if the model will be implemented into a real-life scenario. During the fine-tuning process, several tests are performed on the way of working before creating the final version of the fine-tuned model. In this process, it can be verified whether the fine-tuned model and the fine-tuning itself will work as desired.

Further, during fine-tuning a piece of data is dedicated to the validation of the model (e.g. the validation split). Of course, when training and validating on data from the same dataset, it will result in some bias due to the similarity in the training and validation data. That is why ultimately, the fine-tuned models will be validated on to-be-provided datasets from a real-world air navigation service provider (ANSP).

During the application phase, it is yet to be determined how the verification and validation will be performed. Presumably, the verification will happen by developing a scaled real-world example of the proposed application.

8

Conclusion

This report described the ongoing research on applying a large-scale weakly supervised automatic speech recognition model in air traffic control. It focused on the progress so far and proposed a path toward reaching the research goal. It did so by first introducing the subject and illustrating the state-of-the-art of automatic speech recognition and its application in air traffic control. Afterward, it discussed all the necessary theories behind automatic speech recognition, language processing, machine learning, and air traffic control. It continued with the description of all past and future experiments that are or will be done to reach the research goal. Further, it presents the results of the out-of-the-box performance assessment, discusses the desired results from fine-tuning Whisper, and illustrates methods for verifying and validating the model. At last, this chapter contains the conclusions that can be drawn from the experiments and the research as a whole so far.

8.1. Performance Assessment

The performance test on the ATCO2 and ATCOSIM datasets forms the baseline for future experiments in this research. A word error rate of 27,14% and 17,10%, in the most ideal case, sets the bar for the fine-tuning phase.

From the results, it can be seen that the WER on the ATCOSIM dataset is significantly lower than on the ATCO2 dataset. This is probably caused by the fact that the ATCOSIM dataset only contains ATCo speech and was recorded close to the source, in a simulated environment. The ATCO2 dataset however contains audio of much less quality. This could be explained by the way of data collection in the ATCO2.

Another conclusion that can be drawn is that the Whisper model was indeed trained on a very diverse dataset. This is visible in the performance assessment results. There, it can be seen that applying normalization results in a larger reduction of word error rate than applying prompting. Thus, the model should have had some contextual awareness already when it did the transcribing. For the same reasoning, it can also be concluded that Whisper also has some intelligence in the transcription it makes. Normalization is really necessary in order to reduce the word error rate to the lowest possible.

8.2. Fine-Tuning

As can be seen in the results, Whisper achieves a relatively low word error rate with some simple tweaks in normalization and prompting. Especially on the ATCOSIM dataset, it is possible to receive a WER of only a few percent. That makes it highly likely that fine-tuning Whisper will lead to successful results. Another reason to back up this point is the fact that Whisper has been trained on very diverse datasets, which leaves a lot of room for creating more depth. Especially for the specific phraseology in the ATC domain.

As the effect of normalization and prompting on the ATCO2 dataset is smaller than on the ATCOSIM dataset, it can be expected that the fine-tuning will work better on the latter dataset. However, since the ATCO2 is harder to transcribe than the ATCOSIM dataset, it could also create a more robust model for ATC-specific terminology.

8.3. Air Traffic Control Applications

As was discussed in chapter 1 there are numerous applications that have been mentioned as having the potential for applying automatic speech recognition. Nonetheless, most of those applications have not been implemented yet. Thus, it would create opportunities to tackle this. Especially if the fine-tuned speech recognition model gives significant results.

However, another notable point about the applications is that all of them focus on one side of the communication spectrum, the air traffic controller. By focusing on the pilot side of the spectrum, or on the method of communication in general, a completely new field of research could be created.

Based on the first results and positive prospects, it can be concluded that Whisper has the potential to go the next step. Applying automatic speech recognition to air traffic control could cross the cusp of being technically feasible. If the results are good enough, this research could even pave the way to a fundamentally new way of working in air traffic control.

References

[1] *History of Speech Recognition*. [Online]. Available: https://sonix.ai/history-of-speech-recognition.

[2] D. Wang, X. Wang, and S. Lv, "An Overview of End-to-End Automatic Speech Recognition," *Symmetry*, vol. 11, no. 8, p. 1018, Nov. 2019. DOI: 10.3390/sym11081018.

[3] H. F. Olson and H. Belar, "Phonetic Typewriter," *The Journal of the Acoustical Society of America*, vol. 28, no. 6, pp. 1072–1081, Nov. 1956, ISSN: 0001-4966. DOI: 10.1121/1.1908561. [Online]. Available: https://pubs.aip.org/asa/jasa/article-abstract/28/6/1072/758413/Phonetic-Typewriter?redirectedFrom=PDF.

[4] C. Kim, A. Misra, K. Chin, *et al.*, "Generation of Large-Scale Simulated Utterances in Virtual Rooms to Train Deep-Neural Networks for Far-Field Speech Recognition in Google Home," in *Proceedings to Interspeech 2017*, Stockholm: ISCA, Aug. 2017, pp. 379–383. DOI: 10.21437/Interspeech.2017-1510. [Online]. Available: https://static.googleusercontent.com/media/research.google.com/nl//pubs/archive/46107.pdf.

[5] A. Tjandra, N. Singhal, D. Zhang, *et al.*, "Massively Multilingual ASR on 70 Languages: Tokenization, Architecture, and Generalization Capabilities," in *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Rhodes: IEEE, Jun. 2023, pp. 1–5. DOI: 10.1109/ICASSP49357.2023.10094667. [Online]. Available: https://ieeexplore.ieee.org/document/10094667.

[6] L. Deng, J. Li, J.-T. Huang, *et al.*, "Recent advances in deep learning for speech research at Microsoft," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, May 2013, pp. 8604–8608. DOI: 10.1109/ICASSP.2013.6639345. [Online]. Available: https://ieeexplore.ieee.org/document/6639345.

[7] A. Hannun, C. Case, J. Casper, *et al.*, "Deep Speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, Dec. 2014. DOI: 10.48550/arXiv.1412.5567. [Online]. Available: https://arxiv.org/abs/1412.5567v2.

[8] W. Chan, D. Park, C. Lee, Y. Zhang, Q. Le, and M. Norouzi, "SpeechStew: Simply Mix All Available Speech Recognition Data to Train One Large Neural Network," *arXiv preprint arXiv:2104.02133*, Apr. 2021. DOI: 10.48550/arXiv.2104.02133. [Online]. Available: https://doi.org/10.48550/arXiv.2104.02133.

[9] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised Pre-training for Speech Recognition," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2019-September, International Speech Communication Association, Apr. 2019, pp. 3465–3469. DOI: 10.21437/Interspeech.2019-1873. [Online]. Available: https://arxiv.org/abs/1904.05862v4.

[10] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," *Advances in Neural Information Processing Systems*, vol. 2020-December, Jun. 2020, ISSN: 10495258. [Online]. Available: https://arxiv.org/abs/2006.11477v3.

[11] J. Kahn, M. Riviere, W. Zheng, *et al.*, "Libri-Light: A Benchmark for ASR with Limited or No Supervision," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, May 2020, pp. 7669–7673. DOI: 10.1109/ICASSP40776.2020.9052942. [Online]. Available: https://arxiv.org/abs/1912.07875.

[12] Y. Zhang, D. S. Park, W. Han, *et al.*, "BigSSL: Exploring the Frontier of Large-Scale Semi-Supervised Learning for Automatic Speech Recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1519–1532, 2022. DOI: 10.1109/jstsp.2022.3182537. [Online]. Available: https://doi.org/10.1109%2Fjstsp.2022.3182537.

[13] A. Radford, J. W. Kim, C. Hallacy, *et al.*, “Learning Transferable Visual Models From Natural Language Supervision,” *ArXiv*, Feb. 2021. DOI: 10.48550/arXiv.2103.00020. [Online]. Available: http://arxiv.org/abs/2103.00020.

[14] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. Mcleavey, and I. Sutskever, *Robust Speech Recognition via Large-Scale Weak Supervision*, 2022. DOI: 10.48550/arXiv.2212.04356. [Online]. Available: https://arxiv.org/abs/2212.04356.

[15] V. Pratap, A. Tjandra, B. Shi, *et al.*, “Scaling Speech Technology to 1,000+ Languages,” *arXiv*, May 2023. DOI: 10.48550/arXiv.2305.13516. [Online]. Available: https://doi.org/10.48550/arXiv.2305.13516.

[16] 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, p. 490, May 2023, ISSN: 22264310. DOI: 10.3390/aerospace10050490. [Online]. Available: https://doi.org/10.3390/aerospace10050490.

[17] H. D. Kopald, A. Chanen, S. Chen, E. C. Smith, and R. M. Tarakan, “Applying automatic speech recognition technology to Air Traffic Management,” in *2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC)*, Institute of Electrical and Electronics Engineers Inc., Oct. 2013, pp. 31–6. DOI: 10.1109/DASC.2013.6712620. [Online]. Available: https://ieeexplore.ieee.org/document/6712620.

[18] Y. Lin, L. Deng, Z. Chen, X. Wu, J. Zhang, and B. Yang, “A Real-Time ATC Safety Monitoring Framework Using a Deep Learning Approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 11, pp. 4572–4581, Nov. 2020, ISSN: 15580016. DOI: 10.1109/TITS.2019.2940992.

[19] R. Tarakan, K. Baldwin, and N. Rozen, “An Automated Simulation Pilot Capability to Support Advanced Air Traffic Controller Training,” in *The 26th Congress of ICAS and 8th AIAA ATIO*, Anchorage, Alaska: American Institute of Aeronautics and Astronautics, Sep. 2008, ISBN: 978-1-60086-997-6. DOI: 10.2514/6.2008-8897. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2008-8897.

[20] K. Dönmez, S. Demirel, and M. Özdemir, “Handling the pseudo pilot assignment problem in air traffic control training by using NASA TLX,” *Journal of Air Transport Management*, vol. 89, p. 101 934, Oct. 2020, ISSN: 0969-6997. DOI: 10.1016/J.JAIRTRAMAN.2020.101934. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0969699720305172.

[21] S. Badrinath and H. Balakrishnan, “Automatic Speech Recognition for Air Traffic Control Communications,” *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2676, no. 1, pp. 798–810, Jan. 2022. DOI: 10.1177/03611981211036359. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/03611981211036359.

[22] R. A. Faerber and J. L. Garloch, “Usability evaluation of speech synthesis and recognition for improving the human interface to next generation data link communication systems,” *19th DASC. 19th Digital Avionics Systems Conference. Proceedings (Cat. No.00CH37126)*, vol. 2, pp. 1–5, Oct. 2000. DOI: 10.1109/DASC.2000.884871.

[23] J.-P. Imbert, H. Christophe, and J. Yannick, “Think different: how we completely changed the visualization of Pseudo-Pilots,” in *Graphics Interface*, Halifax: Canadian Information Processing Society, Jun. 2015, pp. 257–264, ISBN: 978-099478680-7. [Online]. Available: https://enac.hal.science/hal-01166368/file/iipp_gi2015.pdf.

[24] V. N. Nguyen and H. Holone, “Possibilities, Challenges and the State of the Art of Automatic Speech Recognition in Air Traffic Control,” *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, vol. 9, no. 8, pp. 1940–1949, Jun. 2015. DOI: 10.5281/zenodo.1108428.

[25] E. Delpech, M. Laignelet, C. Pimm, *et al.*, “A Real-life, French-accented Corpus of Air Traffic Control Communications,” in *Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan, May 2018. [Online]. Available: https://hal.science/hal-01725882%20https://hal.science/hal-01725882/document.

[26] B. Yang, X. Tan, Z. Chen, *et al.*, “ATCSpeech: a multilingual pilot-controller speech corpus from real Air Traffic Control environment,” in *21st Annual Conference of the International Speech Communication Association, INTERSPEECH 2020*, Shanghai: International Speech Communication Association, Oct. 2020, pp. 399–403. DOI: 10.21437/Interspeech.2020-1020.

[27] J. Zuluaga-Gomez, K. Veselý, I. Szöke, *et al.*, “ATCO2 corpus: A Large-Scale Dataset for Research on Automatic Speech Recognition and Natural Language Understanding of Air Traffic Control Communications,” *arXiv*, Nov. 2022. DOI: 10.48550/arXiv.2211.04054. [Online]. Available: https://arxiv.org/pdf/2211.04054.pdf.

[28] K. Hofbauer, S. Petrik, and H. Hering, “The ATCOSIM Corpus of Non-Prompted Clean Air Traffic Control Speech.,” in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco: European Language Resources Association (ELRA), Jan. 2008. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2008/pdf/545_paper.pdf.

[29] Š. Luboš, *Air Traffic Control Communication*, 2011. [Online]. Available: http://hdl.handle.net/11858/00-097C-0000-0001-CCA1-0.

[30] J. Zuluaga-Gomez, A. Prasad, I. Nigmatulina, *et al.*, “How Does Pre-trained Wav2Vec 2.0 Perform on Domain Shifted ASR? An Extensive Benchmark on Air Traffic Control Communications,” in *2022 IEEE Spoken Language Technology Workshop (SLT)*, Jan. 2023, pp. 205–212. DOI: 10.1109/SLT54892.2023.10022724. [Online]. Available: https://ieeexplore.ieee.org/document/10022724.

[31] T. Pellegrini, J. Farinas, E. Delpech, and F. Lancelot, “The Airbus Air Traffic Control speech recognition 2018 challenge: towards ATC automatic transcription and call sign detection,” in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2019-September, International Speech Communication Association, Oct. 2019, pp. 2993–2997. DOI: 10.21437/Interspeech.2019-1962. [Online]. Available: http://arxiv.org/abs/1810.12614%20http://dx.doi.org/10.21437/Interspeech.2019-1962.

[32] J. Hui, *Speech Recognition — GMM & HMM*, Sep. 2019. [Online]. Available: https://jonathan-hui.medium.com/speech-recognition-gmm-hmm-8bb5eff8b196.

[33] J. Daniel and J. H. Martin, *Speech and Language Processing*, 3rd. Stanford University, 2023. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf.

[34] International Civil Aviation Organization, *ICAO Annex 10 Volume ii: Aeronautical Telecommunications*, 2001. [Online]. Available: https://www.icao.int/Meetings/anconf12/Document%20Archive/AN10_V2_cons%5B1%5D.pdf.

[35] J. Delua, *Supervised vs. Unsupervised Learning: What's the Difference?* 2021. [Online]. Available: https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning.

[36] Kizito Nyuytiymbiy, *Parameters and Hyperparameters in Machine Learning and Deep Learning*, Dec. 2020. [Online]. Available: https://towardsdatascience.com/parameters-and-hyperparameters-aa609601a9ac.

[37] International Virtual Aviation Organization, *Airspace structure*. [Online]. Available: https://wiki.ivao.aero/en/home/training/documentation/Airspace_structure.

[38] H. Helmke, M. Kleinert, J. Rataj, *et al.*, “Cost reductions enabled by machine learning in ATM: How can automatic speech recognition enrich human operators' performance?” In *13th USA/Europe Air Traffic Management Research and Development Seminar 2019*, Vienna: EUROCONTROL, Jun. 2019. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85084018759&partnerID=40&md5=b2e8f6471ae5081caeef37639d707a00.

[39] N. Ahrenhold, H. Helmke, T. Mühlhausen, *et al.*, “Validating Automatic Speech Recognition and Understanding for Pre-Filling Radar Labels—Increasing Safety While Reducing Air Traffic Controllers' Workload,” *Aerospace*, vol. 10, no. 6, p. 538, Jun. 2023, ISSN: 2226-4310. DOI: 10.3390/aerospace10060538. [Online]. Available: https://www.mdpi.com/2226-4310/10/6/538.

[40] H. Helmke, M. Slotty, M. Poiger, *et al.*, “Ontology for transcription of ATC speech commands of SESAR 2020 solution PJ.16-04,” in *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*, vol. 2018-September, Institute of Electrical and Electronics Engineers Inc., Sep. 2018, pp. 1–10, ISBN: 9781538641125. DOI: 10.1109/DASC.2018.8569238. [Online]. Available: https://ieeexplore.ieee.org/document/8569238.

[41] H. Helmke, S. Shetty, M. Kleinert, *et al.*, “Measuring Speech Recognition And Understanding Performance in Air Traffic Control Domain Beyond Word Error Rates,” in *SESAR Innovation Days*, Virtual, 2021, pp. 7–9. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85160724873&partnerID=40&md5=8ca44fb5d10f4253d2a32664039d7062.

[42] Y. Lin, M. Ruan, K. Cai, *et al.*, “Identifying and managing risks of AI-driven operations: A case study of automatic speech recognition for improving air traffic safety,” *Chinese Journal of Aeronautics*, vol. 36, no. 4, pp. 366–386, Apr. 2023, ISSN: 1000-9361. DOI: 10.1016/J.CJA.2022.08.020.

[43] Delft High Performance Computing Centre, *DelftBlue Supercomputer (Phase 1)*, 2022. [Online]. Available: https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1.