Easy Adaptation of Speech Recognition to Different Air Traffic Control Environments using the DeepSpeech Engine

Matthias Kleinert[1], Narasimman Venkatarathinam[1,2], Hartmut Helmke[1], Oliver Ohneiser[1], Maximilian Strake[2], Tim Fingscheidt[2]

[1]Institute of Flight Guidance, German Aerospace Center (DLR), Braunschweig, Germany,

[2]Institute for Communications Technology, Technische Universität Braunschweig, Braunschweig, Germany

[1]firstname.lastname@dlr.de

[2]{n.venkatarathinam; m.strake; t.fingscheidt}@tu-bs.de

***Abstract*—Nowadays, recognizing and understanding human speech is quite popular through systems like Alexa®, the Google Assistant or Siri®. Speech also plays a major role in air traffic control (ATC) as voice communication between air traffic controllers (ATCos) and pilots is essential for ensuring safe and efficient air traffic. This communication is still analogue and ATCos are forced to enter the same communication content again into digital systems with additional input devices. Automatic speech recognition (ASR) is a solution to automate this digitization process and an important necessity in optimizing ATCo workflow. This paper investigates the applicability of DeepSpeech, an open source, easy to adapt, end-to-end speech recognition engine from the Mozilla Corporation, as a speech-to-text solution for ATC speech. Different training approaches such as training a model from scratch and adapting a model pre-trained on non-ATC speech are explored. Model adaptation is performed by employing techniques such as fine-tuning, transfer learning, and layer freezing. Furthermore, the effect of employing an additional language model in conjunction with the end-to-end trained model is evaluated and shown to lead to a considerable relative improvement of 61% in word error rate. Overall, a word error rate of 6.0% is achieved on voice recordings from operational and simulation environment of different airspaces, resulting in command recognition rates between 85% and 97%. The achieved results show that DeepSpeech is a highly relevant solution for ATC-speech, especially when considering that it includes easy to use adaptation mechanisms also for non-experts in speech recognition.**

Keywords-automatic speech recognition; ASR; air traffic control; ATC; DeepSpeech; ontology; domain adaptation

I. INTRODUCTION

A. Problem

Speech is mankind's most important means of communication. Its complexity and expressive possibilities allow the exchange of a wide range of information. In recent years, applications that allow recognition and understanding of human speech have become increasingly popular. Systems like Alexa®, the Google Assistant or Siri® work reasonably well for everyday speech and enable a wide variety of applications ranging from searching simple answers for voice queries to controlling home automation systems. Speech also plays a major role in ATC. The voice communication between ATCos and pilots primarily consists of instructions, reports, and readbacks which clearly indicate the future behavior and action of aircraft. This workflow is essential for ensuring a safe and efficient navigation of aircraft through different airspaces. However, to this day, this communication is still analogue and ATCos are forced to enter the same information – the communication content – again into digital systems with mouse, keyboard or touch devices. Automatic speech recognition (ASR) offers a solution to automate this digitization process. It helps to optimize ATCo's workflow and enables more complex applications based on ASR output. The popular systems from Amazon, Google or Apple are not suitable for ATC speech, as they were created for non-ATC purposes. Furthermore, these systems are cloud-based and proprietary, which introduces data-security issues and does not allow any domain-specific adaptation through the user.

Projects in the recent past have shown that methods and technologies exist which allow the adaptation of speech recognition to ATC speech. The Active Listening Assistant (AcListant®) project [1] introduced assistant-based speech recognition (ABSR) [2] a new type of ASR. ABSR can support controller assistant systems and thus ATCos [3],[4]. ATCos can benefit from reduced workload [5], which leads to more efficient air traffic control [6]. The Horizon 2020 funded project MALORCA successfully expanded the ABSR approach to reduce the costs of adaptation to different environments [7]. Current SESAR projects such as HAAWAII [8], PJ.10-96-W2 [9], and PJ.05-97-W2 [10] are also working on different aspects of ABSR to further improve the technology. Such projects make ABSR usable for pilot speech and enable more applications such as workload estimation or readback error detection.

One experiment of the SESAR funded solution PJ.16-04 CWP HMI used a commercial off-the-shelf speech recognition engine from Nuance Communications. For the purpose of the project, the grammar of the engine was adapted to cover ATC speech in the terminal maneuvering area. The command recognition rates of this approach ranged between 31% and 82% for different ATCos in a lab environment [11]. This was for the most part far away from the reported 83% to 95% achieved in MALORCA [7] and AcListant® [3], [5]. In contrast to PJ.16-04

the speech recognition component in these ABSR-related projects heavily relied on ASR experts to achieve the reported recognition quality. Hence, a transfer of these technologies and know-how to non-speech recognition experts from ATC or air traffic management (ATM) industry is difficult and risky.

B. *Solution*

In 2017, the Mozilla Corporation started DeepSpeech [12], an open-source, end-to-end speech recognition engine, which is originally based on a research paper from Baidu Research [13]. In its current state, the project not only offers a pre-trained English model for everyday speech, it also provides the necessary means, tools, and descriptions for easy adaptations to different domains and languages. The easy to use nature especially for non-speech recognition experts and the platform maturity makes it interesting as a solution for ATC speech.

The aim of this paper is to evaluate the quality of DeepSpeech as a speech recognition component for ABSR. For this purpose, different training approaches such as training a model without prior knowledge from scratch and also adapting a model pre-trained on non-ATC speech are explored. The model adaptation is performed via different techniques, e.g., fine-tuning, transfer learning, and layer freezing. A further contribution is the evaluation of using an additional language model for rescoring in conjunction with the end-to-end trained model and the resulting effects on performance when using ATC speech data. This is motivated by previous research showing improved results by using an additional language model with end-to-end speech recognition, especially with medium-sized training data sets [14].

C. *Paper Structure*

In the next section we present related work on speech recognition in ATC. Section III outlines the conventional ASR pipeline and end-to-end models. Section IV gives insights into DeepSpeech and the applied training approaches. Section V shows semantic ATC interpretation on speech-to-text output coming from a DeepSpeech model. Section VI explains the performed experiments and used metrics to determine the quality of word recognition and semantic interpretation, whereas Section VII presents the results. Section VIII concludes the paper and gives an outlook on what could be next.

II. Related Work On ASR in ATC

DLR and Saarland University developed ABSR for ATC-domain speech recognition [3]. ABSR uses speech recognition embedded in a controller assistant system. It makes use of the contextual information from the assistant system to reduce the search space of a speech recognizer, thereby increasing the recognition performance [3]. The ASR models used so far for ABSR are based on Kaldi [15], an open-source speech recognition toolkit. The decoder is based on weighted finite-state transducers [16]. In some controller areas, ABSR reached command recognition rates exceeding 95% [3]. It also proved to significantly increase ATM efficiency. Fuel reductions of 60 liters per flight and a throughput increase by two arrivals per hour are possible due to reduction of controller workload when using ABSR [6]. Adapting ABSR for different airports or control sectors is difficult, as it requires human experts and many manual adaptations [17]. This problem was tackled in the project MALORCA, which performs semi-automatic adaptation of ABSR to other airports based on radar and audio recordings [7].

In 2018, Airbus held an ATC speech recognition challenge. The participants were provided with 40 hours of ATC-speech data. The objective of the participants was to develop a model that can perform automatic speech-to-text transcription for ATC-speech. The models from all participants were evaluated on a data set of about five hours. The best ranked team achieved a word error rate (WER) of 7.62% [18].

The suitability of several state-of-the-art deep neural network architectures for ASR in the ATC domain was evaluated in [19]. All the models were developed using Kaldi. The models were trained on about 176 hours of ATC-speech data. The proposed ASR engine for ATC speech achieved an averaged WER of 7.75% across four test sets with different accents and the results suggest that training on cross-accent data helps in the overall system's performance, rather than limiting the amount of data to single-accent data sets [19].

III. ASR Pipelines

A. *Conventional ASR Pipeline*

ASR is the process of converting input speech signals into their underlying textual representation by means of a computing system. The most fundamental approach of speech recognition involves the following steps:

- The digital speech signal is processed to extract a sequence of feature vectors $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t, \dots \mathbf{x}_T)$, where $\mathbf{x}_t$ represents a feature vector at discrete audio frame index t and T is the number of frames in the sequence.
- Given this sequence of feature vectors $\mathbf{X}$, the corresponding sequence of words $\mathbf{W}^* = (\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_r, \dots, \mathbf{w}_R)$ with word index r and length R is found. The word sequence $\mathbf{W}^*$ has the maximum posterior probability $P(\mathbf{W} \mid \mathbf{X})$ which is computed using Bayes theorem [20] as

$$\mathbf{W}^* = \underset{\mathbf{W}}{\operatorname{argmax}}\, P(\mathbf{W}|\mathbf{X}) = \underset{\mathbf{W}}{\operatorname{argmax}}\, \frac{p(\mathbf{X}|\mathbf{W})P(\mathbf{W})}{p(\mathbf{X})}, \qquad (1)$$

where

- $p(\mathbf{X}|\mathbf{W})$ is called the acoustic model,
- $P(\mathbf{W})$ is called the language model.

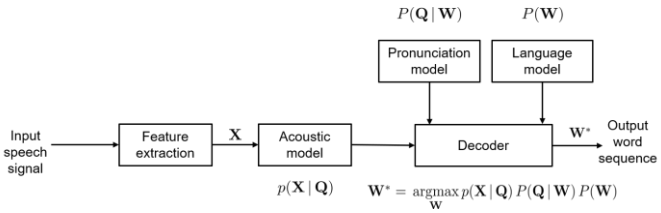

Figure 1. Conventional ASR pipeline

The various components involved in a classical speech recognition system are shown in Figure 1. The feature extraction module provides a high-dimensional feature vector for every

frame of the pre-processed signal, thereby producing the feature vector sequence **X**. Among different types of features, Mel-Frequency Cepstral Coefficients (MFCC) [21] and Perceptual Linear Predictive (PLP) [22] are classical choices.

The acoustic model gives an estimate of the probability of how the given phonetic sequence sounds like in terms of the feature vector sequence, expressed by the term $p(\mathbf{X}|\mathbf{W})$ in (1). These statistical representations are modelled using different classification methods such as Hidden Markov Models (HMMs), Deep Neural Networks (DNNs), and sequence-to-sequence acoustic modeling [23].

The language model, i.e., $P(\mathbf{W})$, is independent of the acoustic observations. The objective of the language model is to incorporate restrictions on the way, in which the words of the vocabulary can be concatenated to form whole sentences [24]. The language model expresses these constraints by assigning a probability to each sequence of words. It provides an estimate of the prior probability $P(\mathbf{W})$ in (1). Many different language modelling types such as n-gram, bidirectional, exponential modelling etc., are used depending on the application.

In large-vocabulary speech recognition, words are considered as sequences of states **Q**. A lexical model also known as pronunciation dictionary describes how words are pronounced phonetically. It is created by human experts with the help of a phone set which is specific to a language. The pronunciation model defines the relationship between orthographic and phonemic representations of words. Incorporating the pronunciation model, the fundamental equation of speech recognition can be rewritten as

$$\mathbf{W}^* = \underset{\mathbf{W}}{\operatorname{argmax}}\ p(\mathbf{X}|\mathbf{Q}) \cdot P(\mathbf{Q}|\mathbf{W}) \cdot P(\mathbf{W}), \qquad (2)$$

where $P(\mathbf{Q}|\mathbf{W})$ is the pronunciation model.

In speech recognition, making a search decision is also referred to as decoding. The decoding process of a speech recognizer is to find a sequence of words whose corresponding acoustic and language models best match the input signal [20]. Decoding is basically a process of finding a word sequence $\mathbf{W}^*$ which has maximum posterior probability $P(\mathbf{W}|\mathbf{X})$ as shown in (1). Search strategies based on dynamic programming or the Viterbi algorithm have been applied successfully to a wide range of speech recognition tasks [20].

B. *End-to-End Models*

In the conventional ASR pipeline, various training methods exist for each module of the previous subsection. Every module is independently optimized with its own optimization objective function. Consequently, this makes the training process complex and difficult to be globally optimized [25]. In recent years, end-to-end speech recognition models have received significant interest due to the simplification of the training process. End-to-end refers to the point that most modules of the conventional pipeline are replaced by one neural network-based model. This model is able to perform decoding on its own, which means it can directly map a feature vector sequence **X** to a sequence of words $\mathbf{W}^+$ as shown in Figure 2. This allows to model dependencies in the acoustic sequence and the word sequence and therefore already integrates knowledge about the acoustics and the language using a joint optimization process.

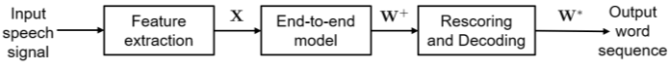

Figure 2. End-to-end ASR pipeline

The fact that an end-to-end model is able to perform decoding on its own makes the usage of a language model optional [26]. Nevertheless it is known that on medium-sized data sets combining an end-to-end model with an additional language model on top, see "Rescoring and Decoding" with output $\mathbf{W}^*$ in Figure 2, can improve the performance, e.g., in conjunction with so-called transformers [14]. A popular training approach for end-to-end models is Connectionist Temporal Classification (CTC) [27].

IV. DEEPSPEECH RECOGNITION ENGINE

DeepSpeech is an open-source end-to-end speech recognition engine launched in 2017 by the Mozilla Corporation. It was developed based on deep learning algorithms and is originally inspired by the research papers [13], [28]. DeepSpeech can be trained using supervised learning techniques without any external "sources of intelligence", like a grapheme-to-phoneme converter or forced alignment on the input [29]. All the models discussed in this paper are trained using DeepSpeech version 0.9.3 [30]. This section describes the model architecture of DeepSpeech and training approaches used in this paper.

A. *Architecture*

The core of DeepSpeech is a deep recurrent neural network [31],[32] that receives audio features as input and outputs the corresponding transcription. The model architecture of DeepSpeech for the first three exemplary frames of the input feature sequence **X** is illustrated in Figure 3.

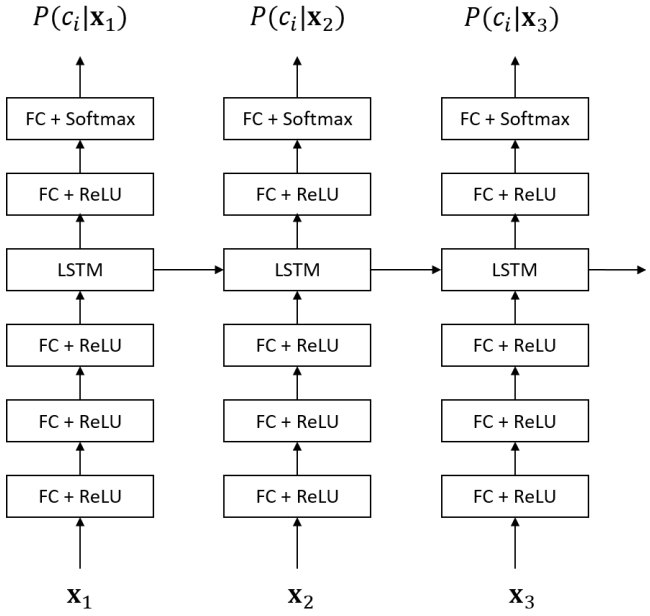

Figure 3. Model architecture of DeepSpeech

The deep recurrent neural network has overall six layers. MFCC features [21] are extracted from the input audio signal and fed into three fully connected (FC) layers with rectified linear unit (ReLU) activation function. The fourth layer is a unidirectional long short-term memory (LSTM) [33] unit with tanh activation function. This is followed again by an FC layer with ReLU activation function. The output layer of the network gives a matrix of character probabilities over time, which is a fully connected layer with softmax activation function. For each step, the network outputs a probability $P(c_i|\mathbf{x}_t)$ of each character c_i in the character set $\mathcal{C}$, corresponding to that particular input audio frame t. DeepSpeech uses the CTC-loss function [27],[34], the gradients with respect to all model parameters are computed via back-propagation [35], and the Adam optimizer [36] is used for training.

B. *External Scorer*

CTC-based end-to-end models often use additional language models (often called scorer) to improve the recognition performance. Similarly, DeepSpeech can use an additional language model, which is referred to as "scorer", to improve the accuracy of the predicted transcripts. The scorer is used to compute the likelihood (also called a score, hence the name "scorer") of sequences of words or characters in the output, to guide the decoder towards more likely results [37]. The scorer consists of two components, the

- KenLM language model [38],
- and a vocabulary as a text file.

The probability estimate from the language model is included as a factor in the inference problem [34], which is given by

$$\mathbf{W}^* = \underset{\mathbf{W}}{\operatorname{argmax}} P(\mathbf{W}|\mathbf{X}) \cdot P(\mathbf{W})^{\alpha} \cdot R_{\mathbf{W}}^{\beta}. \qquad (3)$$

The language model probability is raised with the language model weight α. The number of words $R_{\mathbf{W}}$ in the sequence $\mathbf{W}$ is raised by the word insertion weight β. The decoder preference to a smaller number of longer words or a larger number of shorter words is controlled using β. Both α and β are determined empirically by tuning them on a development data set [20].

C. *Training Approaches*

As part of the DeepSpeech 0.9.3 release a model for regular English language is provided. The *pre-trained* English model was trained on approximately 6,500 hours of data not related to ATC, which can be directly used for inference. DeepSpeech also provides means to train a completely new model without prior knowledge from *scratch* for any domain-specific data set.

In addition to training a model from scratch, DeepSpeech allows to adapt models pre-trained on other data sets. There are two approaches supported for adaptation by DeepSpeech namely *fine-tuning* and *transfer learning* [39]. For *fine-tuning*, checkpoints of a pre-trained model can be used to bootstrap the training process [39]. The requirement is that the transcriptions of the target data set also come from the same character set as the model which is used for adaptation.

DeepSpeech's *transfer learning* allows to remove certain layers from a pre-trained model, to initialize new layers for the target data, to stitch together the old and new layers, and to update all layers via gradient descent. This training approach is suggested when the character set of the pre-trained model does not match with the target data [40]. The pre-trained output layer (and optionally more layers) can be removed and reinitialized to fit the target alphabet.

In addition to the implementation provided by DeepSpeech for transfer learning, [41] discusses another approach called *layer freezing*. In this process, many parameters of the original model may be "frozen", i.e., held constant during training. In the transfer learning approach discussed previously, the weights of certain chosen layers of a pre-trained model are dropped, i.e., they are reinitialized and then the weights of all the layers are updated during the training process. On the other hand, in the *layer freezing* approach, the weights of certain chosen layers of a pre-trained model are held constant throughout the training process. Only the weights of those layers, which are not frozen are updated during the training process. The first (and optionally more subsequent layers) can be frozen during the training process.

V. SEMANTIC INTERPRETATION OF ATC SPEECH

A DeepSpeech model provides recognized words for a given audio utterance as output. When comparing this speech-to-text output with the gold (correct) transcription, the WER can be calculated. For an ATC application this is more or less useless, as long as no semantic interpretation exists. An ATCo does not care if the words of a greeting are correctly recognized. A wrong recognition of a greeting should, accordingly, not disturb the correct recognition of, e.g., a descend command. But, what does semantic interpretation mean? Given the utterance "*three nine two papa continue heading zero six zero descend altitude six thousand*", the semantic interpretation is:

```
AUA392P MAINTAIN HEADING 060 none
AUA392P DESCEND 6000 none
```

The CWP HMI project [42], in which more than 20 European partners from ATM industry, research and air navigation providers worked together from 2016 to 2018 has developed a so-called ontology, i.e., rules for transforming ATC utterances into their semantic interpretations. An utterance consists of one or more instructions.

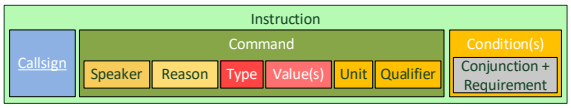

Figure 4. Elements of an instruction consisting of a callsign, a command, and optional condition(s).

Each instruction (Figure 4) consists of one callsign, one command, and either no, one or more conditions. A command always has a type and depending on the type values, a unit and a qualifier can follow. In the above example the words "*three nine two papa continue heading zero six zero"* are transformed into the callsign "AUA392P", which is only possible, if additional

context information from, e.g., surveillance data is available. The type, extracted from "*continue heading*" consists of the two words "MAINTAIN HEADING". The utterance "*zero six zero*" is transformed into the value 060. As no qualifier LEFT or RIGHT is specified, none is added, since a qualifier is mandatory for the MAINTAIN HEADING command.

In each instruction the callsign is repeated, although spoken only once. If no callsign is said or could not be extracted, respectively, NO_CALLSIGN is used. The utterance "*descend altitude*" is mapped to the type DESCEND, while "*six thousand*" results in the value "6000". No unit (e.g., feet or flight level) is provided. Therefore, again "none" is used. A qualifier is optional for a DESCEND command type.

The word sequence "*four eight six november i call you for preceding traffic turn left heading seven zero initially*" results in

```
NLY486N INFORMATION TRAFFIC none
NLY486N HEADING 070 LEFT
```

The heading value according to the ontology always has three digits, although the leading zero was not said, which is a deviation from the International Civil Aviation Organization (ICAO) phraseology. The word sequence "*danke schoen four miles final speed one sixty or less tower one correction four miles final speed one sixty tower one two three eight good bye*" results in

```
NO_CALLSIGN CORRECTION
NO_CALLSIGN SPEED 160 none UNTIL 4 NM FINAL
NO_CALLSIGN CONTACT TOWER
NO_CALLSIGN CONTACT_FREQUENCY 123.800
NO_CALLSIGN FAREWELL
```

The semantic interpretation is needed to calculate comand recognition rates when comparing the automatic inerpretation with the gold (corret) interpretation.

VI. Experimental Setup and Metrics

Multiple DeepSpeech models for word recognition were trained with the training approaches described in IV.C. For semantic interpretation, an already implemented rule-based model was used [43]. This section describes the used data sets for training, validation, and evaluation, as well as the used metrics to determine the quality of word recognition and semantic interpretation.

A. ATC Communication Data Sets

The used data sets consist of voice and radar recordings from operational (ops) and simulation (lab) environment of Prague Approach (Czech Republic) and Vienna Approach (Austria) from the SESAR projects *MALORCA* and *PJ.16-04 CWP HMI*. For DeepSpeech model training, these data sets were combined with the *ATCOSIM* data set, a publicly available lab data set that contains ATC voice recordings (no radar recordings) for the en-route sectors Söllingen, Geneva, and Zürich [44]. The combined data set contains voice recordings with a total duration of 33.6 hours in 28,452 voice utterances. All the voice utterances have corresponding manually created and verified transcriptions. For the experiments, the data set was split as shown in Table I.

TABLE I. SPLITTING OF DATA SET FOR DEEPSPEECH MODEL TRAINING

Data Set	# Utterances	Duration
Training	17,877	21.0 h
Validation	7,662	9.0 h
Test	2,913	3.6 h

The data in Table I is used for training and evaluation of a DeepSpeech model for word recognition. For the evaluation of the semantic interpretation as described in Section V, manually created/verified semantic interpretations (gold annotations) are required. The ATCOSIM data set does not have any gold annotations so it is excluded for the semantic interpretation experiments. Half the utterances from the Vienna ops room environment have a gold annotation. From Prague ops/lab and Vienna lab environment, all manually transcribed utterances also have a gold annotation, resulting in the number of utterances shown in Table II. Table III shows the corresponding number of ATC commands.

TABLE II. NUMBER (#) OF UTTERANCES AND AVERAGE NUMBER OF WORDS PER UTTERANCE (UTT)

Data Set	All	Training + Validation	Test	Words/Utt (Average)
Ops Prague	3,039	2,368	671	13.2
Lab Prague	4,219	3,877	342	13.2
Ops Vienna	2,956	2,785	171	unknown
Lab Vienna	3,566	3,201	365	13.9

TABLE III. NUMBER (#) OF COMMANDS (CMDS) AND AVERAGE NUMBER OF COMMANDS PER UTTERANCE (UTT)

# Commands	All	Training + Validation	Test	Cmds / Utt (Average)
Ops Prague	6,120	4,765	1,355	2.0
Lab Prague	6,904	6,313	591	1.6
Ops Vienna	4,713	4,419	294	1.6
Lab Vienna	6,014	5,384	630	1.7

B. Evaluation Metrics

For speech recognition systems the commonly used metric for performance comparison is the word error rate (WER), which is derived from the Levenshtein distance. It determines the quality of a recognition by the three types of errors which can appear:

- Substitution (S) – an incorrect word was substituted for the correct one in the recognized word sequence
- Deletion (D) – a correct word was omitted in the recognized word sequence
- Insertion (I) – an extra word was added in the recognized word sequence

Based on these error types, the WER is given by

$$WER = \frac{S+I+D}{N}, \qquad (4)$$

where *N* is the number of words in the reference (gold) transcription and *S*, *I*, and *D* as described above.

The quality of semantic interpretations is evaluated on ontology instructions (see Section V). The so-called command recognition rates are computed by comparing the correct semantic interpretation of a voice utterance (gold annotation) to the results of the automatic semantic interpretation (automatic annotation). For a given voice utterance, each instruction in a gold or automatic annotation is treated as one big word. Then, similar to WER, the Levenshtein distance between the gold and automatic annotation is calculated, resulting again in the number of substitutions (S), insertions (I), and deletions (D). Table IV gives an overview about the different metrics, where #gold means the total number of commands in the gold annotation and #matches = #gold - S - D. More information and examples are provided in [45].

TABLE IV. METRIC DEFINITIONS

Metric	Calculation
Command Recognition Rate (**RcR**)	**RcR** = #matches / #gold
Command Recognition Error Rate (**ErR**)	**ErR** = (S + I) / #gold
Callsign Recognition Error Rate (**CaE**)	Same as ErR, but only for callsigns without instructions

For calculating the CaE, we just compare callsigns from the gold and automatic annotation. For each voice utterance we consider the callsign only once, except if more than one callsign is annotated or extracted, then the calculation is done for each callsign.

VII. RESULTS AND DISCUSSION

Various DeepSpeech models for word recognitions were created/used and evaluated. Creation of the models, except for one, is based on the data sets *Train* and *Validation*. Evaluations are based on the data set *Test*, described in section VI.A. Table V shows results for following models (details in section IV.C):

- Pre-train – English model, no ATC speech adaptation.
- Scratch – New model trained only on ATC speech.
- Fine-Tune – Pre-trained model adapted with ATC speech.
- Trans-Learn-X – Pre-trained model adapted with ATC speech via transfer learning. X refers to the number of reinitialized layers, where these are preceding layers starting from the output layer.
- Layer-Freeze-X – Pre-trained model adapted with ATC speech via layer freezing. X refers to number of frozen layers, where these are subsequent layers starting from the input layer.

The different models were evaluated with an additional 3-gram, 4-gram, or no language model (LM) resp. scorer (see Section IV.B). The table shows that using the pre-trained model without any adaptation to ATC-speech is not usable, because WER varies between 85.2% and 100.0%. The other models which all incorporate an adaptation to ATC speech all reach quite good results with WERs below 10% with a 4-gram LM. Overall, the best result is achieved by adaptation of the pre-trained model with the fine-tuning method, which shows that the out-of-domain data used for pre-training can still help in training a model for the ATC domain, even though the domains are quite different. Since the *Trans-Learn-X* approaches all provide worse WERs compared to *Fine-Tune*, it can be inferred that the pre-trained parameters of all layers contribute to a better ATC domain performance. The *Fine-Tune* model achieved a WER of 15.5% without using an LM. With a 4-gram LM the WER is reduced significantly to 6.0%.

TABLE V. WER ON TEST DATA WITH DIFFERENT DEEPSPEECH MODELS

Model	No LM WER %	3-gram LM WER %	4-gram LM WER %
Pre-train	100.0	85.2	86.6
Scratch	20.0	8.6	8.2
Fine-Tune	**15.5**	**6.3**	**6.0**
Trans-Learn-1	22.6	7.7	7.4
Trans-Learn-2	24.3	8.3	7.8
Trans-Learn-3	31.4	9.1	8.6
Trans-Learn-4	32.9	10.0	9.4
Trans-Learn-5	33.1	10.4	9.8
Layer-Freeze-1	15.8	6.3	6.0
Layer-Freeze-2	16.7	6.8	6.5
Layer-Freeze-3	17.1	7.7	7.4
Layer-Freeze-4	18.5	7.9	7.6

Results of the following semantic interpretation are always based on the output of *Fine-Tune*, in combination with the 4-gram LM. Table VI presents the error rates on automatic extraction of callsigns. The column “Gold” shows the CaE on automatically extracted callsigns from the manually transcribed voice utterances and thus provides an upper limit for callsign extraction that can be reached with a perfect ASR model.

TABLE VI. CALLSIGN RECOGNITION ERROR RATE CAE

	CaE Gold	CaE Auto	WER	Delta Abs	Delta Rel
Ops Prague	0.6%	5.7%	9.1%	5.1%	90%
Lab Prague	0.0%	3.9%	1.1%	3.9%	100%
Ops Vienna	0.6%	6.4%	9.0%	5.8%	91%
Lab Vienna	1.6%	3.0%	2.6%	1.4%	45%

The performance is very good and ranges between 0% and 1.6%, but we also see that the callsign extraction is not perfect, except for the lab data from Prague. When the automatically transcribed utterances, i.e., the DeepSpeech model with a certain WER is used for word recognition, the performance of the callsign extraction decreases as seen in column “Auto”. The corresponding WER of the DeepSpeech model on the different data sets can be seen in the respective column. The column “Delta Abs” shows the absolute reduction of the CaE between “Gold” and “Auto” and column “Delta Rel” shows the relative loss.

Table VII has the same structure as before, but shows the RcR performance, i.e., the quality on extracting full instructions instead of only callsigns. The results on the manual transcripts (Gold) show that the extraction works very well on Prague utterances. Also, Vienna results are quite good, but some challenges are left. The majority of the problems result from the fact that deviations from standard phraseology occur more often

in the Vienna utterances than in the Prague utterances. The extraction performance again goes down, when automatic transcriptions from DeepSpeech are used instead of the gold transcriptions (Auto).

TABLE VII. COMMAND RECOGNITION RATE RCR

	RcR Gold	RcR Auto	WER	Delta Abs	Delta Rel
Ops Prague	98.2%	85.0%	9.1%	13.1%	15%
Lab Prague	99.7%	97.0%	1.1%	2.7%	3%
Ops Vienna	96.3%	87.4%	9.0%	8.8%	10%
Lab Vienna	97.5%	95.6%	2.6%	1.9%	2%

The WER for Vienna is comparable to the WER for Prague. Therefore, we achieved similar results on RcR. The results published in [11] (lab) and [46] (ops room) always report a better RcR and WER for Prague compared to more noisy Vienna data, but these projects created speaker specific models for Vienna and Prague. Our approach uses a common model trained on data from Vienna, Prague and ATCOSIM from ops room and lab environment plus thousands of hours of regular speech already included in the pre-trained DeepSpeech model, which enables it to be used for both environments without retraining.

TABLE VIII. COMMAND RECOGNITION ERROR RATE ERR

	ErR Gold	ErR Auto	WER	Delta Abs	Delta Rel
Ops Prague	1.5%	8.4%	9.1%	6.9%	82%
Lab Prague	0.3%	2.9%	1.1%	2.5%	88%
Ops Vienna	4.4%	8.8%	9.0%	4.4%	50%
Lab Vienna	1.1%	1.4%	2.6%	0.3%	22%

Table VIII shows the achieved ErR. Error rates of 0.6% for Prague and 3.5% for Vienna were reported in [46], but their context from available surveillance and weather data was used to optimize the word recognition and exclude unlikely extractions, which is not used here. From previous experiments it can be expected that using context information will dramatically improve the presented ErR herein and only slightly decrease the RcR.

VIII. CONCLUSION AND OUTLOOK

The aim of this paper was to identify the suitability of DeepSpeech, as an open source easy to use speech recognition engine from the Mozilla Corporation, as a speech-to-text solution for domain-specific speech in ATC. Various end-to-end models were trained based on different training approaches. Based on the performance results achieved by these models, it can be concluded that DeepSpeech provides a solution to easily create end-to-end models that deliver great performance for ATC speech even if only a relatively small amount of domain-specific training data is available.

The used training approaches were based on either training a new model from scratch with only ATC related speech or adaptation of the pre-trained English model provided by DeepSpeech, which is trained on thousands of hours of non-ATC speech. The best results were achieved by a fine-tuning adaptation of the general DeepSpeech English model with about 30 hours of domain-specific ATC speech, i.e., the WER of this model on test data was 6.0% compared to 8.2% for a model trained from scratch.

Although the trained end-to-end models can perform inference for ATC speech without any additional language model, an additional scorer was used to improve the recognition performance. In our scenario, an impressive relative WER improvement of 61% was observed for the best model, when the inference was performed along with a 4-gram LM.

Performing automatic semantic interpretations on the output of the best DeepSpeech model showed command recognition rates between 85% and 97% and command recognition error rates between 1.4% and 8.8%. For some data sets, the performance on the gold transcriptions is very good, but for some there is still room for improvement on the word recognition, but also on the semantic interpretation side. However, considering the relatively simple architecture of DeepSpeech and its easy-to-use nature, that makes adaptations possible in an efficient way. These results are impressive and can be considered as state-of-the-art for ATC speech [11], [46].

In the future, the performance of the trained models could be improved further by exclusively tuning the models to a specific controller area or with enough data even for a specific controller. The best performance improvement can be expected by ABSR, i.e. the integration of surveillance data. The models can also be trained further using more data from different airports or sectors to obtain a general model which is even more robust to changes to other control areas.

ACKNOWLEDGMENT

The SESAR2020 exploratory research project *MALORCA* and the industrial research projects *PJ.16-04-W1* CWP HMI both have received funding from the SESAR Joint Undertaking under the European Union's grant agreement No. 698824 and 734141.

REFERENCES

[1] The project AcListant® (Active Listening Assistant) http://www.aclistant.de, n.d.

[2] H. Helmke, H. Ehr, M. Kleinert, F. Faubel, and D. Klakow, "Increased acceptance of controller assistance by automatic speech recognition," in 10th USA/Europe Air Traffic Management Research and Development Seminar (ATM2013), Chicago, IL, USA, 2013.

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

[4] O. Ohneiser, H. Helmke, H. Ehr, H. Gürlük, M. Hössl, T. Mühlhausen, Y. Oualil, M. Schulder, A. Schmidt, A. Khan, and D. Klakow, "Air Traffic Controller Support by Speech Recognition," in N. Stanton, S. Landry, G. Di Bucchianico, and A. Vallicelli (Eds.), Proceedings of the 5th International Conference on Applied Human Factors and Ergonomics AHFE 2014, Advances in Human Aspects of Transportation: Part II, pp. 492-503, Krakow, CRC Press, 2014.

[5] H. Helmke, O. Ohneiser, T. Mühlhausen, and M. Wies, "Reducing controller workload with automatic speech recognition," in IEEE/AIAA 35th Digital Avionics Systems Conference (DASC), Sacramento, CA, USA, 2016.

[6] H. Helmke, O. Ohneiser, J. Buxbaum, and C. Kern, "Increasing ATM efficiency with assistant-based speech recognition," in 12th USA/Europe

Air Traffic Management Research and Development Seminar (ATM2017), Seattle, WA, USA, 2017.

[7] The project MALORCA (Machine Learning of Speech Recognition Models for Controller Assistance) http://www.malorca-project.de, n.d.

[8] The project HAAWAII (Highly Automated Air Traffic Controller Workstations With Artifical Itelligence Integration) https://www.haawaii.de, n.d.

[9] PJ.10-96-W2: SESAR2020 funded industrial research projects under the European Union's grant agreement 874470, https://cordis.europa.eu/programme/id/H2020_SESAR-IR-VLD-WAVE2-10-2019/de, n.d.

[10] PJ.05-97-W2 SESAR2020 funded industrial research projects under the European Union's grant agreement 874464, see for further information https://www.remote-tower.eu/wp/?page_id=888, and https://www.remote-tower.eu/wp/?p=824 and https://www.sesarju.eu/index. php/projects/DTT, n.d.

[11] M. Kleinert, H. Helmke, S. Moos, P. Hlousek, C. Windisch, O. Ohneiser, H. Ehr, and A. Labreuil, "Reducing Controller Workload by Automatic Speech Recognition Assisted Radar Label Maintenance," 9th SESAR Innovation Days, Athens, Greece, 2019.

[12] Mozilla Corporation, "DeepSpeech Model," 2021, https://deepspeech.readthedocs.io/en/v0.9.3 (visited on 08/31/2021).

[13] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep Speech: Scaling up end-to-end speech recognition," ArXiv 1412.5567, 2014.

[14] T. Lohrenz, P. Schwarz, Z. Li, and T. Fingscheidt, "Relaxed Attention: A Simple Method to Boost Performance of End-to-End Automatic Speech Recognition," submitted to IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Cartagena, Colombia, 2021.

[15] Kaldi Toolkit http://kaldi-asr.org/doc/about.html, n.d.

[16] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," Computer Speech & Language, 2002.

[17] 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?," 13th USA/Europe Air Traffic Management Researchand Development Seminar, Vienna, Austria, 2019.

[18] 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 20th Annual Conference of the International Speech Communication Association (INTERSPEECH 2019), Austria, Sep 2019,.

[19] J. Zuluaga-Gomez, P. Motlicek, Q. Zhan, et al, "Automatic Speech Recognition Benchmark for Air-Traffic Communications," In: Proceedings of Interspeech 2020. Sanghai: International Speech Communication Association, 2020, pp. 2297-2301.

[20] X. Huang, A. Acero, H.-W. Hon, and R. Reddy, "Spoken LanguageProcessing: A Guide to Theory, Algorithm, and System Development," 1st Edition, Prentice Hall PTR, USA, 2001.

[21] S. Davis and P. Mermelstein. "Comparison of parametric representations for mono-syllabic word recognition in continuously spoken sentences," IEEE Transactionson Acoustics, Speech, and Signal Processing 28.4, pp. 357–366, 1980.

[22] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," Journal of the Acoustical Society of America 87.4, pp. 1738–1752, 1990.

[23] S. Bhatt, A. Jain, and A. Dev, "Acoustic Modeling in Speech Recognition: A Systematic Review," International Journal of Advanced Computer Science and Applications 11.4, 2020.

[24] J. P. H. van Santen, "Handbook of Standards and Resources for Spoken LanguageSystems," Computational Linguistics 24.3, 1998.

[25] D. Wang, X. Wang, and S. Lv, "An Overview of End-to-EndAutomatic Speech Recognition," Symmetry 11.8, 2019

[26] T.-S. Nguyen, S. Stüker, and A. Waibel, "Toward Cross-Domain Speech Recognition with End-to-End Models," in Proceedings of the Life Long Learning for Spoken Language Systems Workshop colocated with ASRU Singapore, 2019.

[27] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks", Proceedings of the 23rd International Conference on Machine Learning. ICML, USA, 2006.

[28] D. Amodei, R. Anubhai, E. Battenberg, and C. Case, "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin," in Proceedings of the 33rd International Conference on Machine Learning - Volume 48, 2016.

[29] R. Morais, "A Journey to <10% Word Error Rate," 2017 https://hacks.mozilla.org/2017/11/a-journey-to-10-word-error-rate (visited on 08/31/2021).

[30] Mozilla Corporation, "DeepSpeech Model," 2020. url: https://deepspeech.readthedocs.io/en/v0.9.3/DeepSpeech.html (visited on 08/31/2021).

[31] Z. C. Lipton, "A Critical Review of Recurrent Neural Networks for Sequence Learning," arXiv:1506.00019, 2015.

[32] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and LongShort-Term Memory (LSTM) Network," Physica D: Nonlinear Phenomena Volume 404, March 2020.

[33] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory", Neural Computation, pp. 1735–1780, 1997.

[34] A. Hannun, "Sequence Modeling with CTC," Distill, 2017. url: https://distill.pub/2017/ctc/ (visited on 08/31/2021).

[35] P. Werbos, "Backpropagation through time: what it does and how to do it," Proceedings of the IEEE 78.10, pp. 1550–1560, 1990.

[36] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization", 3rd International Conference on Learning Representations ICLR, San Diego, CA, USA, 2015.

[37] Mozilla Corporation, "CTC beam search decoder," 2020. url: https://deepspeech.readthedocs.io/en/v0.9.3/Decoder.html (visited on 08/31/2021).

[38] K. Heafield, I. Pouzyrevsky, J. H. Clark, and P. Koehn, "Scalable Modified Kneser-Ney Language Model Estimation", Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, pp. 690–696, 2013.

[39] Mozilla Corporation, "Training Your Own Model," 2020. url: https://deepspeech.readthedocs.io/en/v0.9.3/TRAINING.html#training-a-model (visited on 08/31/2021).

[40] Mozilla Corporation, "Transfer-Learning (new alphabet)," 2020. url: https://deepspeech.readthedocs.io/en/v0.9.3/TRAINING.html#transfer-learning-new-alphabet (visited on 08/31/2021).

[41] O. Eberhard and T. Zesch, "Effects of Layer Freezing when Transferring DeepSpeech to New Languages", arXiv:2102.04097, 2021.

[42] H. Helmke, M. Slotty, M. Poiger, D.F. Herrer, O. Ohneiser et al., "Ontology for transcription of ATC speech commands of SESAR 2020 solution PJ.16-04," IEEE/AIAA 37th Digital Avionics Systems Conference (DASC), London, UK, 2018.

[43] M. Kleinert, H. Helmke: S. Shetty, O. Ohneiser, H. Ehr, A. Prasad, P. Motlicek, and J. Harfmann "Automated Interpretation of Air Traffic Control Communication: The Journey from Spoken Words to a Deeper Understanding of the Meaning," IEEE/AIAA 40th Digital Avionics Systems Conference (DASC), virtual, 2021.

[44] K. Hofbauer, S. Petrik, and H. Hering, "The ATCOSIM Corpus of Non-Prompted Clean Air Traffic Control Speech," Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), Marrakech, Morocco, 2008.

[45] H. Helmke, S. Shetty, M. Kleinert, O. Ohneiser, H. Ehr, A. Prasad, P. Motlicek, A. Cerna, and C. Windisch, "How to Measure Speech Recognition Performance in the Air Traffic Control domain? The Word Error Rate is only half of the truth!," Satellite Workshop of Interspeech, Brno, Czech Republic, 2021.

[46] M. Kleinert, H. Helmke, H. Ehr, Chr. Kern, D. Klakow, P. Motlicek, M. Singh, and G. Siol, "Building Blocks of Assistant Based Speech Recognition for Air Traffic Management Applications," 8th SESAR Innovation Days, Salzburg, Austria, 2018.