An NLP-Focused Pilot Training Agent for Safe and Efficient Aviation Communication

Xiaochen Liu, Bowei Zou, Ai Ti Aw
Institute for Infocomm Research (I2R), A*STAR, Singapore
{liu_xiaochen, zou_bowei, aaiti}@i2r.a-star.edu.sg

Abstract

Aviation communication is vital for safe and efficient flight operations. However, pilots often struggle to adhere to strict phraseology due to diverse backgrounds and language proficiency levels. Traditional training methods involve expensive setups and reliance on human-in-the-loop simulations. To overcome these challenges, we propose an NLP-focused training agent. Our approach leverages natural language capabilities and involves fine-tuning on communication data to generate instructions based on input scenarios (keywords). Given the absence of prior references for this business problem, we explored the feasibility of our proposed solution by 1) generating all instructions at once and 2) generating one instruction while incorporating conversational history in each input. Our findings affirm the feasibility of this approach, emphasizing the effectiveness of fine-tuning pre-trained models and large language models in advancing aviation communication training.

1 Introduction

Efficient and accurate communication is crucial in air traffic management (Cardos et al., 1998). During real flying scenarios, Air Traffic Controllers (ATCos) engage in timely communication with numerous aircraft in designated airspace, including aviation instructions, aeronautical announcements, traffic advisories, aerodrome announcements, and weather updates. Unfortunately, miscommunications frequently occur, attributed to pilot readback errors (Hamzah, 2018; Yang et al., 2023) and phraseology issues (Helmke et al., 2021). This long-standing problem, highlighted in an analysis of NextGen 2013 found pilot mishearing (28%) and no pilot readback (20%) as predominant factors, constituting 74% of human errors based on 382 miscommunication messages (Skaltsas et al., 2013). A 2023 study showed that 92% of aviation respondents believed language training was essential (Hamzah et al., 2023). Therefore, providing training in pilots' phraseology and domain-specific language is crucial to addressing communication challenges for safety and operational efficiency.

On the other hand, the aviation industry has been actively seeking solutions amid the rapid advancement and widespread adoption of artificial intelligence technologies (Kashyap, 2019). AI-driven tools, such as Natural Language Processing (NLP), are seamlessly integrated into every facet of modern aviation (Kabashkin et al., 2023). Noteworthy implementations include human-in-the-loop training involving flight simulations and communications (Williams et al., 2014), aviation ontology construction (Helmke et al., 2022), readback error detection (Helmke et al., 2021), and phraseology training (Zuluaga-Gomez et al., 2023). Despite these efforts, none have addressed the requirements for enhancing pilots' communication. Training that incorporated human actors often incurred substantial costs for setup (Brudnicki et al., 2005), significant resources (Williams et al., 2014; Kabashkin et al., 2023), and necessitated labor-intensive annotated data for simulation (Wu et al., 2021). For instance, an MITRE report (Johnson, 2010) revealed that simulating an air traffic scenario for communication training in a small class required the involvement of four domain experts, four AI assistants, and one traffic simulation system, all within a meticulously configured environment integrating AI, network, and interfaces.

To address the above challenges, we propose an NLP-focused training agent that exclusively utilizes aviation communication data. This agent offers language training to pilots, aiming to enhance their proficiency in domain-specific phraseology while minimizing resource utilization and costs. Essentially, pilots can effortlessly create a tailored aviation communication environment, generating a list of aviation instructions with just a few clicks on a single machine. Subsequently, they can systematically perform readbacks of each instruction

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track), pages 89–96

through an application interface, transmitting natural language data to the backend for processing. Throughout the training loop, pilots receive prompt feedback on their readbacks and are prompted for repetition in the event of a readback error. Upon successfully executing readbacks for all instructions in a training session, pilots have the flexibility to opt for another training session with a different set of instructions using the same setup. The core of this method revolves around aviation instruction generation, for which we leveraged transformer-based pre-trained language models, including GPT-2 (Radford et al., 2019), BART(Seq2Seq model) (Lewis et al., 2020), and Llama2 (Touvron et al., 2023). Our contributions include:

1. **Cost-Effective Data Setup**. The proposed method is dedicated to generating contextually-aware aviation instructions using only communication data. Unlike traditional approaches, it eliminates the need for a massive amount of annotated input and expensive flight simulation setups, providing promising task outcomes at a fraction of the cost.

2. **Enhanced Method Explainability**. The method enhances explainability by highlighting specific keywords in the input that significantly impact the generated content. This feature ensures clarity for non-technical users, offering a transparent understanding of the rationale behind decision-making processes.

3. **Efficient Domain Adaptation**. The method integrates existing NLP techniques to achieve broader adaptation in solving aviation-related problems. It aids in the analysis of deeper contextual constraints and provides valuable data insights for similar tasks in cases where other forms of data are unavailable.

2 Training Agent Workflow

According to the International Civil Aviation communications Organization (ICAO), aviation communication phraseology is precisely defined with specific vocabulary and content (ICAO, 2020). While aviation instructions draw from a relatively limited set of words, their meanings within this context can significantly diverge from general language, especially concerning terminology (ICAO, 2020). For example, consider a message, *FAKEAIR ONE TWO NINER DIRECT TO NOVEMBER ECHO PAPA WHISKEY WHISKEY*, which signifies that it is an Air Traffic Control (ATC) communication directed at a pilot flying the aircraft. It consists of domain-specific language for alphabets, numbers, the placement of call-sign, and aviation instruction for aircraft manipulation and readback.

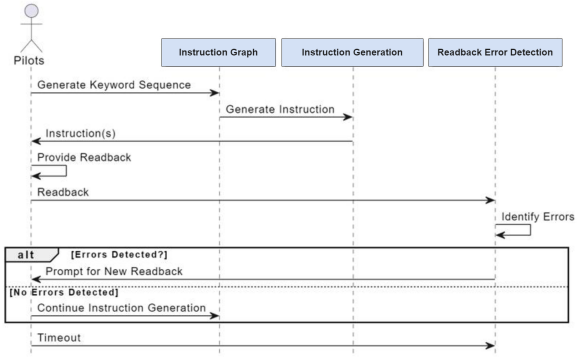

Figure 1: Workflow of pilot training agent.

In a real-world scenario involving communication between a controller and an aircraft, a set of aviation instructions would be issued sequentially. If pilots provided correct readbacks, or if ATCos rectified readback errors through clarification, communication proceeded smoothly. However, instances of miscommunication could occur during live flying operations. This becomes a focal point for us, aiming to minimize such occurrences through cost-effective simulations of these interactions with pilots. These interactions include user input processing, context-aware instruction generation, pilot response/readback detection, and a reproducible workflow for generating content with various instruction scenarios.

Consequently, we come up with a training agent with an integrated solution leveraging existing NLP techniques, as illustrated in Figure 1. At the application level, pilots initiate a request for a flight communication scenario, specifying the communication channel and desired instruction length for practice. The request is transmitted to the *Instruction Graph* module, which generates a sequence of instruction keywords based on transitional probabilities specific to the chosen communication channel. Subsequently, the instruction scenario is forwarded to the *Instruction Generation* module to produce instruction content. The module then returns either a list of instructions, offloading the instruction issuance to the application layer, or transmits one instruction at a time along with conversational history. Pilots provide readbacks of individual instructions

at the application layer and transcribe into natural language data. Each readback is processed by the *Readback Detection* module to raise a signal if a readback error is found. In case errors are detected, pilots are prompted for a new readback; otherwise, the issuance of instructions continues. Once all instructions are successfully read back and verified, pilots can initiate the process for further practice.

3 Methodology

The proposed training agent aims to generate aviation instructions that are both meaningful and contextually aware. To achieve this goal, we devise a specific input format termed *Instruction Scenario*, which comprises a collection of distinct keywords representing various types of instructions and communication channels. These instruction scenarios are retrieved and stored in *Instruction Graphs*, serving as inputs for generation at the application layer. Subsequently, the instruction generation process utilizes the instruction scenario as input and generates words auto-regressively, where each token is predicted based on the previously generated tokens. For this task, we employ transformer-based language models and fine-tuned them with domain-specific data.

Instruction Scenario and Graph. We establish two state spaces: one for instruction keywords, denoted as $KW = \{kw_1, kw_2, ..., kw_n\}$, and another for communication channels, represented by $C = \{c_1, c_2, ..., c_m\}$. To calculate the transitional probabilities among different types of instructions, the transitional probability $p_{i,j}$ from one instruction kw_i to another instruction kw_j is calculated by dividing the count of transitions from kw_i to kw_j by the total count of transitions from kw_i to all possible instructions on a communication channel C. Thus, each instruction scenario is defined as a combination of a communication channel keyword and a sequence of instruction keywords (KW) that have occurred on that channel (C).

Subsequently, we organize the keywords and their corresponding transitional probabilities into matrices $P = \{p_1, p_2, ...p_m\}$, which are of size $(n + 2, m)$ to accommodate an initial state and an end state to KW. Equation 1 elucidates that the summation of each row's transitional probabilities for KW always equals 1, where i and j signifies rows and columns respectively, and $p_{i,j}$ denotes the transitional probability from kw_i to kw_j. Consequently, we derive directed graphs for KW, denoted as $\overrightarrow{G} = \{g_1, g_2, ...g_m\}$, incorporating the transitional matrices P. The graph can be symbolized as $\overrightarrow{G} = (V, E, W)$, where $E = \{kw_i, kw_j \in V\}$ and $W = \{w_{i,j} | p_{i,j}, w_{i,j} \in \mathbb{R}\}$.

$$\sum_{j=1}^{N} p_{i,j} = \sum_{j=1}^{N} \mathbb{P}(kw_{n+1} = j | kw_n = i) = 1 \quad (1)$$

Instruction Generation. We employ the transformer neural network architecture (Vaswani et al., 2017) for the generation, including both decoder-only and Seq2Seq architectures to explore and compare methods. Our goal is to generate instructions for a given instruction scenario by predicting the next token in the sequence based on the context of the preceding tokens. Consequently, each training instance is designed with a set of domain-specific vocabulary V, an instruction scenario $q = \{x_1, x_2, ..., x_m\}$ where $x_i \in V$, and a sequence of instructions $S = \{y_1, y_2, ..., y_n\}$ where $y_i \subseteq V$. In summary, Equation 2 elucidates how the model factorizes the probability of vocabulary across instructions via the chain rule (Biswas et al., 2022) and subsequently generates the current instruction y_i given q and all previous instructions $y_{1:i-1}$.

$$p\left(S|q\right) = \prod_{i=1}^{n} P\left(y_i | y_{1\,:\,i-1}, q\right) \quad (2)$$

To systematically generate consecutive and correlated instructions given an instruction scenario, we design two data templates, as illustrated in Figure 2. These templates compose the input context differently, aiding in the coherent generation of instructional content.

- *Generating the entire sequence of instructions at once*. We utilize natural language features for text generation, including the use of special tokens embedded in text generation, incorporating contextual vocabulary to augment contextual awareness among generated tokens, and leveraging punctuation to discern lexical boundaries. This enables us to predict the entire sequence of instructions at a specific timestamp, considering the context of the entire instruction scenario. Further details can be found in the first example depicted in Figure 2.

- *Generating a single instruction with context*. To generate instructions one at a time, we

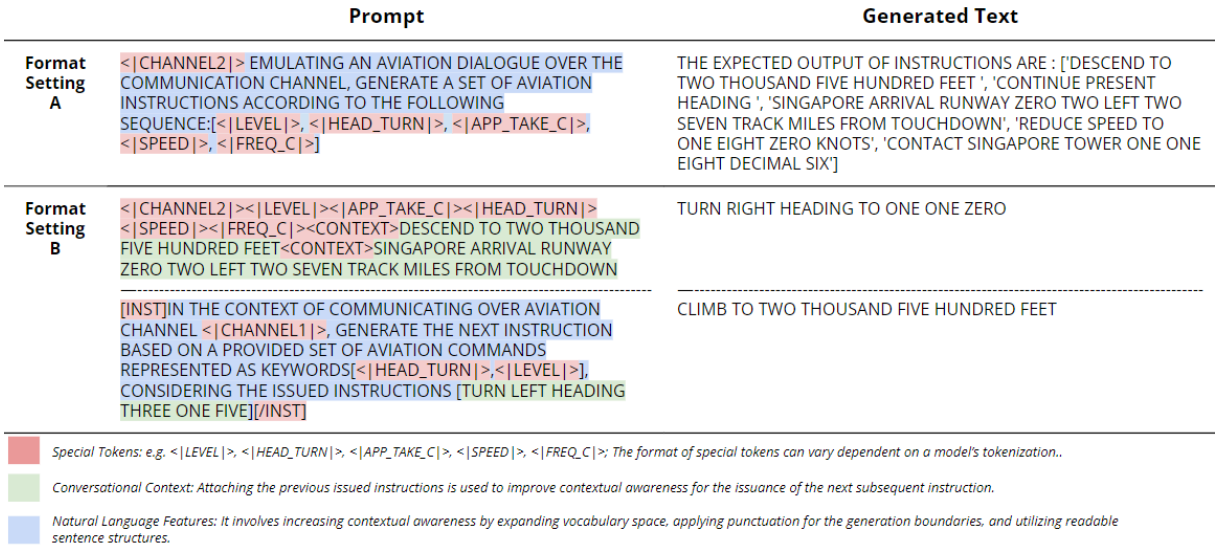

Figure 2: Prompt and generated instructions under two distinct experimental settings. Format A: Generation of all instructions with a specified instruction scenario. Format B: Sequential generation, producing one instruction at each timestamp with the inclusion of previously issued instructions.

append previously issued instructions as dynamic context to the instruction scenario. It involves feeding the model the instructions already generated, allowing it to build upon its understanding of the scenario as it progresses. As demonstrated in the second example of Figure 2, this approach facilitates the sequential generation of each instruction with additional context.

Finetuning. Similar to predicting the next sequence of tokens based on previous tokens as input (Radford et al., 2019), our objective is to minimize the language modeling loss through fine-tuning the domain-specific dataset E. This fine-tuning process is devised to minimize the negative log-likelihood L with parameters θ, as illustrated in Equation 3.

$$L\left(E\right) = -\sum_{j=1}^{|E|} log\ p_\theta\left(y_i|y_{1:i-1}, q\right) \quad (3)$$

We adopt LoRA (Hu et al., 2021) for parameter fine-tuning within finite GPU resources. LoRA preserves the LLM parameters while introducing trainable rank-decomposition matrices into each layer. In Equation 4, A and B represent the decomposed matrices, where B has dimensions $B \subseteq \mathbb{R}^{d\times r}$ and A has dimensions $A \subseteq \mathbb{R}^{r\times k}$, with the number of rank denoted by r (where $1 \leq r \leq 4$), and n refers to the model's dimension of its dense layer. The objective is to customize an LLM for a specific task by updating its parameters using trainable matrices with dimensions $n \times r + r \times n$, without altering the original LLM parameters, which are of size $n \times n$.

$$h = W_x + \Delta W_x = W_0 x + BAx \quad (4)$$

Readback Error Detection. This component takes in a pair of instruction texts from different speakers to indicate whether an error is detected in the pilot's readback when compared against the Air Traffic Controller's (ATCo) instruction. The definition of an error may vary depending on the implementation, such as the outcome of basic string matching, the semantic distance between two strings, or the matching of words only within the essential semantic attributes of a specific type of instruction, akin to the approach by Helmke et al. (2022). For the sake of simplicity in this experiment, we adopt the string-matching approach.

4 Experiments

The objective of this experiment is to objective the potential of generating correlated aviation instructions exclusively with the proposed instruction scenario and natural language input. For instruction generation, we finetune pre-trained language models using domain-specific communication data, including GPT-2 (Radford et al., 2019), BART-base (Lewis et al., 2020), and Llama2-7b (Touvron et al., 2023). Due to lack of aviation metadata, we assume

Channel	No. of conversations	Avg. no. of tokens per conv.	Avg. no. of instr. per conv.
a	761	43	5
b	1,553	38	5
c	1,179	69	7
d	3,185	40	4
e	1,068	38	4
f	5,763	41	4
g	683	55	5
h	2,789	35	3

Table 1: Data distribution. "conv.": conversation; "instr.": instruction.

that training the models to reconstruct original aviation communication would enable them to generate meaningful communication context and content. Additionally, several in-house developed tools and components are employed to support the experiments and the eventual integration at the business level. However, their performance is not reported in this experiment. Refer to Table 2 for references to these complementary components contributing to the final solution.

4.1 Dataset and Settings

The data is transcribed from live aviation communications across seven communication channels at a certain airport, consisting of approximately three months of audio transcriptions involving a total of 500 different aircraft. Table 1 lists the distribution of the original data per communication channel. Each conversation within the dataset represents an instruction scenario, comprising spoken utterances exchanged between one Air Traffic Controller (ATCo) and one pilot at specific time intervals. Subsequently, an in-house module for instruction extraction is employed to process these conversations, by which individual sequences of ATCos' instructions are extracted and associated with keywords for categorization. Each instruction sequence serves as the foundation for developing both the graphs of application input and the instructions' generation process. As a result, we eventually obtain a training set consisting of 14,433 instances, a development set comprising 2,548 instances, and a domain-specific test set containing 10 annotated conversations.

We instantiate GPT2, BART-base, and LLama2-7B models from *HuggingFace*[1] and conduct these experiments independently, including the instantiation of their respective tokenizers. For adding

In-house module	Function
Instruction Extraction	Extract instructions from an aviation dialogue.
Read-back Error Detection	Detect and yield errors in the read-back of and instruction.
Semantic-slot Filling	Tag aviation instruction into semantic slots.
Front-end Application	Fetch user input and system response; User interaction.

Table 2: In-house components for training agent development.

domain-specific vocabulary, we include keywords representing instruction categories and communication channels as special tokens. Additionally, in fine-tuning LLama2-7B, we utilize an existing *PEFT* implementation of LoRa [2], with a rank-size of 4 and the task type of *CASUAL_LM*. Furthermore, we utilize an existing quantization package, *bitsandbytes*[3], to load the model in 4-bit mode for memory efficiency, enabling completion of experiments within three GTX3090 GPUs.

In this experiment, we utilize a proprietary test set derived from domain-specific data, expertly annotated to assess the quality of the generated instructions. The test set comprises 221 instructions across 10 conversations, exhibiting an above-average volume of instructions. To evaluate the content overlap between the generated instructions and the provided references, we conduct an automated evaluation using BLEU-4 and ROUGE-L metrics. In essence, the BLEU-4 score serves to measure the precision of the generated instructions in comparison to the reference. A higher BLEU-4 score signifies a substantial N-gram overlap between the generated and reference instructions. Similarly, the ROUGE-L metric is adopted to evaluate the recall of the generated content against the reference. ROUGE-L focuses on the ability to generate the longest sub-sequence of tokens compared to the references, with a higher ROUGE-L score indicating a more comprehensive coverage of the intended meaning.

4.2 Main Results

As shown in Table 3, the results for instruction generation are presented for two experimental settings: generating all instructions (Setting A) and generating one instruction at a time (Setting B). In the results for Setting A, Llama2-7b achieves

[1] https://huggingface.co/

[2] https://huggingface.co/docs/peft/en/package_reference/lora

[3] https://huggingface.co/docs/bitsandbytes/

Setting A	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L
GPT2	0.512	0.296	0.215	0.180	0.471
BART-base	0.184	0.058	0.030	0.021	0.208
Llama2-7b	**0.650**	**0.456**	**0.375**	**0.320**	**0.639**

Setting B	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L
GPT2	0.289	0.157	0.109	0.089	0.490
BART-base	0.445	0.284	0.207	0.173	0.446
Llama2-7b	**0.644**	**0.509**	**0.427**	**0.372**	**0.582**

Table 3: Performance of instruction generation under Settings A and B on test set.

the highest BLEU-4 and ROUGE-L scores among the various models, followed closely by GPT2 and BART-base. It indicates that Llama2-7b not only captures the meaning of the reference instructions but also produces a substantial portion of content that closely aligns with the reference instructions. While GPT2 demonstrates a closely ROUGE-L score of 0.471 compared to Llama2-7b, it exhibits a relatively lower BLEU-4 score than Llama2-7b. It implies that the GPT2 model is relatively effective in capturing the meaning (content) of the references during generation but lacks a strong capability to precisely replicate the exact wording or sequence of words found in the reference instructions. Such phenomenon may be attributed to the scope of the models and the amount of natural language data used in pre-training.

In the results for Setting B, Llama2-7b still gains the highest BLEU-4 and ROUGE-L scores among all the models. Notably, the ROUGE-L score indicates minimal variation among all models in terms of their ability to convey the meaning of the reference text, highlighting the effectiveness of incorporating conversational context into the models' input for generating one instruction at a time. Apart from the top performer, Llama2-7b, in this setting, BART-base demonstrates the ability to generate a relatively higher amount of overlapping content with the references compared to GPT2, with BLEU-4 scores of 0.173 and 0.089 respectively. This phenomenon may be attributed to the models' sensitivity to contextual information input and controllable factors during pre-training.

4.3 Numeric Constraints in Instruction Generation

This study is to evaluate whether the generated instructions can offer numeric values that align with aviation constraints and remain contextually relevant, even in the absence of metadata. To examine the boundaries and numeric constraints for specific instruction categories, we perform a post-analysis

CHANNEL	UNIT					
	FL	FEET	KNOTS	QNH	HEAD	FREQ
a	0.89	0.94	0.75	1.00	0.68	1.00
b	0.89	1.00	1.00	1.00	0.89	1.00
c	N.A	0.91	1.00	N.A	0.90	0.80
d	N.A	N.A	N.A	N.A	N.A	0.72
e	0.62	N.A	1.00	N.A	0.75	1.00
f	0.96	N.A	1.00	N.A	0.84	0.94
g	1.00	N.A	N.A	N.A	0.7	1.00
h	0.80	N.A	N.A	N.A	0.86	1.00

Table 4: Accuracy for numeric value generation. N.A: no samples of the specified type in this channel.

on the dev set. Utilizing an in-house tool, we annotate semantic slots to each token in a given instruction, similarly to Helmke et al. (2022). We specifically focus on instructions related to *Heading Change*, *Altitude Change*, *Frequency Change*, *Speed Change*, and *Frequency Change*. These instruction types necessitate precise numeric values for aircraft manipulation and are crucial for safety. Numeric tokens are then converted to digits, and potential outliers are removed by trimming values falling between the 10th and 90th percentiles of the original distribution. Such outliers may arise from transcription or readback errors, as exemplified by instances like *DESCEND TO NINE THOUSAND THOUSAND FEET*, which deviates from aviation context and poses challenges for communication training. It is important to note that these numeric constraints are channel-specific. For each communication, we create 150 customized aviation scenarios with the instruction graph, utilizing a weighted random walk algorithm that considers transition probabilities among instructions within each communication channel. Finally, we format these instruction scenarios according to Setting A and feed them into Llama2-7b for evaluation.

The results presented in Table 4 reveal that the accuracy of generated numeric values is significantly higher for instructions designated to specific aviation channels compared to those intended for more widespread issuance with diverse constraints

based on airspace. For instance, the instruction *QNH*, used for altimeter settings, demands only a limited set of values, enabling the model to learn and generate them accurately. In contrast, *heading* instructions are commonly issued across all communication channels, each presenting unique constraints for this type of instruction. The model requires further refinement to better identify these constraints and develop an effective strategy for generation. Enhancing this adaptability is crucial for fostering a more realistic user experience when training phraseology with the bot agent.

5 Conclusion

Accurate and clear communication plays a pivotal role in ensuring aviation safety and operational efficiency. Nevertheless, pilots with diverse backgrounds frequently encounter difficulties in adhering to strict phraseology requirements, potentially hindering operational effectiveness. To tackle this challenge, we present an NLP-focused training agent that leverages natural language features and existing communication data to generate personalized instructions tailored to specific input scenarios. Our experimental results demonstrate the significant efficacy of the proposed method. Consequently, this approach eliminates the need for costly human-in-the-loop simulations and extensive annotated data entries, paving the way for a cost-effective and accessible future in aviation training.

Acknowledgements

This work is supported by the National Research Foundation, Singapore, and the Civil Aviation Authority of Singapore (CAAS), under the Aviation Transformation Programme. (Grant No. ATP_IOP for ATM_I2R_1 and Grant No. ATP_ASRU_I2R). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not reflect the views of the Civil Aviation Authority of Singapore.

References

Biplob Biswas, Renhao Cui, and Rajiv Ramnath. 2022. Retrieval based response letter generation for a customer care setting. *NAACL-HLT 2022 Industry Track, Association for Computational Linguistics*, pages 168 – 175.

Dan Brudnicki, Bob Ethier, and Kerri Chastain. 2005. Application of advanced technologies for training the next generation of air traffic controllers. *The MITRE Corporation*.

Kim Cardos, Paul Falzarano, and Sherwin Han. 1998. Pilot-controller communication errors: An analysis of aviation safety reporting system (asrs) reports. *U.S. Department of Transportation Research and Special Programs Administration*.

Haryani Hamzah. 2018. Miscommunication in pilot-controller interaction. *ResearchGate*.

Haryani Hamzah, Pramela Krish, and Afendi Hamat. 2023. Aviation communication challenges and language training development: Perspectives from pilots and air traffic controllers. *Training, Language and Culture*, 7.

Hartmut Helmke, Matthias Kleinert, Shruthi Shetty, Karel Veselý, Karel Ondřej, Pavel Smrz, ..., and Christian Windisch. 2021. Readback error detection by automatic speech recognition to increase atm safety. *Fourteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2021)*.

Hartmut Helmke, Michael Slotty, Michael Poiger, Damián Ferrer Herrer, Oliver Ohneiser, Nathan Vink, ..., and Mario Boyero Pérez. 2022. Ontology for transcription of atc speech commands of sesar 2020 solution pj.16-04. *Conference: 2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*.

Edward Hu, Yelong Shen, Phillip Wallis Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv:2106.09685*.

ICAO. 2020. *Doc 4444, Procedures for Air Navigation Services, Air Traffic Management*. ICAO, Montréal, Canada, 2016.

Craig M. Johnson. 2010. Human-in-the-loop (hitl) simulation and analysis of optimized profile descent (opd) operations at atlanta. *The MITRE Corporation*.

Igor Kabashkin, Boriss Misnevs, and Olga Zervina. 2023. Artificial intelligence in aviation: New professionals for new technologies. *MDPI applied sciences*.

R. Kashyap. 2019. Artificial intelligence systems in aviation. in cases on modern computer systems in aviation. *IGI Global: Hershey, PA, USA*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020 Association for Computational Linguistics*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Gerasimos Skaltsas, Jasenka Rakas, Matthew G., and Karlaftis. 2013. An analysis of air traffic controller-pilot miscommunication in the nextgen environment. *Journal of Air Transport Management*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, ..., and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Meta AI*.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is all you need. *NIPS*.

Kevin W. Williams, Bonny Christopher, Gena Drechsler, Shawn Pruchnicki, Jason A. Rogers, ..., and Samuel Cotton. 2014. Aviation human-in-the-loop simulation studies: Experimental planning, design and data management. *Federal Aviation Administration*.

Xingjiao Wu, Luwei Xiao, and Sun Yixuan. 2021. A survey of human-in-the-loop for machine learning. *ResearchGate*.

Hui-Hua Yang, Yu-Hern Chang, and Yi-Hui Chou. 2023. Subjective measures of communication errors between pilots and air traffic controllers. *Journal of Air Transport Management*.

Juan Zuluaga-Gomez, Amrutha Prasad, Iuliia Nigmatulina, Petr Motlicek, and Matthias Kleinert. 2023. A virtual simulation-pilot agent for training of air traffic controllers. *arXiv:2304.07842v1*.