Structured Command Extraction from ATC Communications Using Open and Fine-Tuned Language Models

Ana Maria Mekerishvili*, Junzi Sun*, Patrick Jonk†, Vincent de Vries†

*Faculty of Aerospace Engineering
Delft University of Technology
The Netherlands

†Aerospace Operations Division
Royal Netherlands Aerospace Centre
The Netherlands

***Abstract*—Radiotelephony remains the primary medium for pilot-controller communication, yet extracting structured information from spoken exchanges is challenging. Deep learning approaches often depend on large annotated datasets, limiting use in data-scarce environments. This study evaluates open-source Large Language Models for Structured Information Extraction from ATC communications, with applications in assisting or automating pseudo-pilot tasks. We evaluate Llama 3.3 (70B) with baseline prompting and Gemma 3 (4B) with baseline and fine-tuned variants on 496 utterances from NLR's ATM simulator: NARSIM (NLR ATC real-time simulator). Performance is assessed on human transcripts and ASR outputs from Whisper models, with varying prompt contexts. Cross-sector generalization is tested across two ATC sectors. Using manual scoring, Llama 3.3 achieves micro-F1 0.95 on human transcripts and 0.86 on fine-tuned Whisper outputs. While Gemma 3 performed weaker in its baseline form, fine-tuning on a small sample led to notable improvements. Results demonstrate the potential of LLMs for ATC applications without the need for large annotated datasets.**

***Keywords*—Radiotelephony, speech-to-text, language model, structured command extraction**

I. Introduction

The exchange of operational information between pilots and controllers is primarily conducted through voice-only radiotelephony (RT) communications. Automating structured information extraction from these exchanges supports applications such as performance assessment, safety monitoring, scenario analysis, and simulator training. In simulator environments, automatic parsing of Air Traffic Controller (ATCO) commands assists pseudo-pilots and potentially automates a subset of their tasks, which is particularly valuable given the workload, high demand, and scarce availability for this specialized task [1].

Previous studies have researched using BERT-family encoders for Named Entity Recognition (NER) and Slot Filling tasks in the ATC domain [2], [3]. BERT, or Bidirectional Encoder Representations from Transformers, is a transformer encoder pre-trained with masked language modeling and, in its original form, next-sentence prediction [4], [5]. It is typically fine-tuned on labeled data for downstream tasks. BERT-based approaches have required domain-specific labeled datasets, a challenge in ATC where annotated open data is scarce [2], [6]. F1 scores between 0.66 and 0.97, depending on dataset size and quality, have been achieved by [2], [7], [8]. More complex architectures, such as RoBERTa-Attention-BiLSTM-CRF, have been explored [9], and prompt-based approaches like SLKIR shows promise with low-resources [10].

While most prior work relies on large, high-quality annotated corpora and focuses on NER or slot filling, relatively little research has addressed the generation of structured information extraction (SIE) from audio, especially in scenarios with limited human-labeled training data. The SIE task in this research entails extracting structured entities from ATC instructions according to an ontology adapted from SESAR PJ16-04 [7]. For each ATCO utterance, the parsed information includes the callsign, and for each instruction: category, command, value, unit, qualifier, and condition.

Unlike traditional natural language processing tasks like NER, SIE must handle complete ATC commands and capture relationships between entities, including situations where multiple instructions appear in a single utterance.

Recent advances in large language models (LLMs) have opened new opportunities for such systems, as these models have been shown to generalize effectively from very limited data [5], [11]. Outside ATC, LLMs and prompt-based methods have demonstrated strong performance for SIE and NER, especially in few-shot or low-resource contexts [12], [13]. For ASR, end-to-end models such as Whisper have outperformed previous hybrid approaches in the ATC domain [14], [15].

Our work in this paper addresses the gap in end-to-end SIE by evaluating open-source LLMs for structured extraction from both human and ASR transcripts, leveraging Whisper for ASR given its strong performance and open availability. A high-level overview is shown in Figure 1.

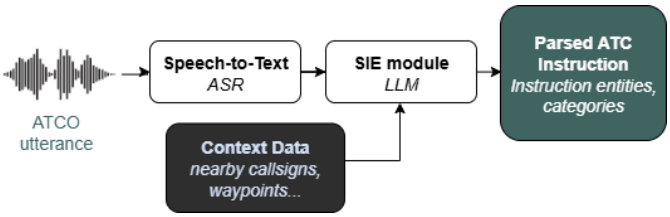


Figure 1. Simple overview of ASR and context-enhanced Structured Information Extraction from Air Traffic Controller utterances

The aim of this study is to evaluate the ability of open-source LLMs to extract structured ATC commands from both human transcripts and ASR outputs, incorporating varying

levels of contextual information. We examine: (i) few-shot prompting of LLMs on human transcripts to accelerate labeling and estimate an upper bound on clean-text performance, and (ii) few-shot prompting of LLMs on ASR outputs to assess realistic end-to-end performance. Most experiments use a large model (Llama-3.3 [16])

We further compare with a smaller architecture (Gemma 3 [17]), evaluated both in its baseline form and after fine-tuning on approximately 350 labeled samples. The effect of including different levels of contextual information in prompts is also examined.

The ASR component uses both a baseline and a fine-tuned Whisper model [14], [15]. Experimental data are drawn from NARSIM, NLR's real-time ATM simulator[1]. Performance is measured using F1 scores, and sector generalizability is assessed by evaluating results on two different sectors separately. Finally, a qualitative analysis of the LLM outputs is performed.

II. Dataset

The dataset used in this research was collected from NARSIM during airspace restructuring experiments. These simulations involve operational participants, including LVNL (Air Traffic Control The Netherlands), the Dutch air navigation service provider, and MUAC (Maastricht Upper Area Control Centre), which manages upper airspace over parts of northwest Europe. The dataset includes recordings of controller–pilot exchanges on separate radio frequencies, as well as scenario metadata such as flight entry times, flight routes, radio frequency assignments, and airline designators. All audios are digitally recorded and therefore have better audio quality compared to actual ATCO recordings.

Specifically, two frequencies are used in this study: one from LVNL's area control center and another from MUAC, responsible for the Delta sector. Table I summarizes the dataset used for evaluation in the case of few-shot prompting, as well as for fine-tuning and evaluating the Gemma 3 model. The audio segments from MUAC-Delta and LVNL-ACC1 are transcribed in approximately equal proportions. A separate dataset, LVNL-dev, is used exclusively for prompt engineering and is described in Table II.

TABLE I. Overview of the main NARSIM dataset: number of instructions and the total duration of the transcribed audio segments per frequency

Dataset	Sector	Instructions	Duration
MUAC	Delta (Upper Area Control)	251	16 min
LVNL	ACC1 (Area Control)	263	15 min

TABLE II. LVNL-dev set used for prompt design and validation, sampled separately from the main LVNL dataset with no shared audio segments between the two.

Dataset	Sector	Instructions	Duration
LVNL-dev	ACC1 (area control)	93	6 min

[1]https://www.nlr.org/newsroom/facility/narsim/

A. Transcription and Annotation

Audio segmentation is performed based on audio energy thresholds to isolate individual utterances. ATCO commands are transcribed by first running a Whisper model on the segments and then manually correcting them using Prodigy[2]. The audio recordings also include occasional greetings between ATCOs in Dutch, which are not included in the dataset for this study.

B. Context Data

Next to the recorded audio from NARSIM, complementary context data from the experiments is also used in this study to provide extra context information when prompting the LLMs. In particular, this includes the callsigns of all flights, along with their associated flight plans, consisting of scheduled entry times and route waypoints. The context data used in this study corresponds to scenario information that is typically available prior to the start of ATM simulations. So, no real-time or in-simulation information is incorporated into context data. For some aircraft, the available route data is incomplete, meaning that certain utterances lack corresponding waypoint information. To address this, synthetic augmentation is applied to five utterances from the combined LVNL and MUAC datasets by inserting the missing waypoint into the list of possible waypoints at a random position.

III. Experimental Setup

All experiments are conducted on data obtained from NARSIM. The Structured Information Extraction (SIE) module is evaluated on both human and ASR transcriptions. The small development subset (LVNL-dev), shown in the previous Table II, is used to refine prompts, and performance on the LVNL-ACC1 and MUAC-Delta sets is not used for prompt iteration.

A. ASR Module

We evaluate two ASR systems: 1) the baseline Whisper large-v3 from OpenAI, and 2) a fine-tuned Whisper large-v2 model from our predecessor study [15] using human-labeled ASR data. Performance of both these models is shown in Table III.

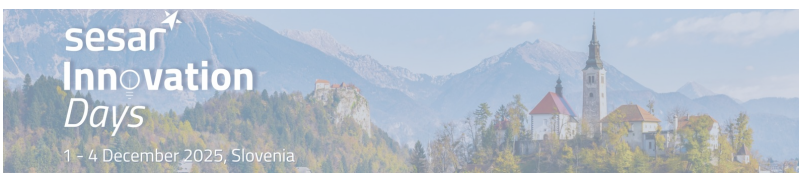
TABLE III. Word error rate of each Whisper model on the LVNL-ACC1 and MUAC-Delta NARSIM sets.

Dataset	Whisper Baseline (large-v3)	Whisper Fine-tuned (large-v2)
LVNL	40.7%	12.2%
MUAC	28.3%	17.2%
Total	34.8%	14.6%

The word error rates (WER) shown in this table are calculated after normalization, including case folding, number normalization, and mapping NATO phonetic alphabet tokens to letters according to [15]. In some outputs of the fine-tuned model, a spurious fixed string occasionally appears at the start, which is easily removed in post-processing.

It is worth noting that, although the fine-tuned model is not trained on NARSIM data, its training set includes operational

[2]https://prodi.gy/



data from LVNL, which may explain its stronger performance on data from the simulated LVNL sector compared to the MUAC sector.

B. Structured Information Format and Ontology

The SIE module first identifies the commands in transcribed ATCO utterances, then determines their *category* and extracts command *entities* in a structured JSON format. An example of a SIE input-output pair is shown in Figure 2. The utterance in this example contains two instructions: a climb command and a heading command. The instruction also contains a correction of a misspoken value. The SIE returns the command category along with all instruction entities for each command. In this case, neither command includes a qualifier or condition, so the module returns *null* for these fields.

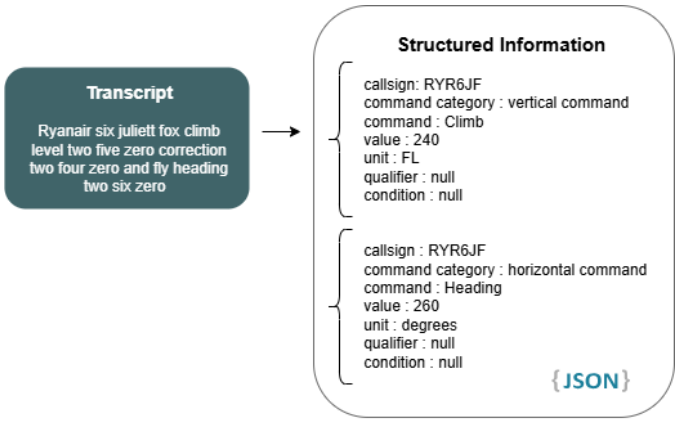


Figure 2. Example of structured information extraction (SIE) from a human-annotated ATC transcript.

The *Command Category*, or *Instruction Category*, can be regarded as a multi-class prediction task with labels *vertical command*, *horizontal command*, *speed command*, *changing frequency command*, or *other*, which is a subset of the categories mentioned in SESAR PJ16-04 [7]. An overview of the frequency at which each of these categories *appears* in the LVNL and MUAC datasets is shown in Figure 3.

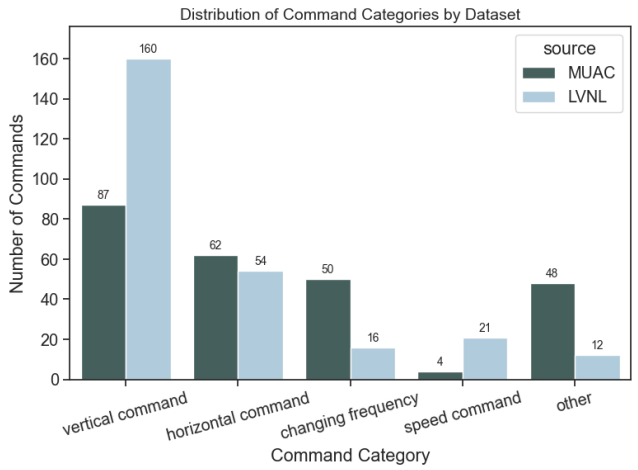


Figure 3. Instruction frequencies per sector by category.

Overall, vertical commands (altitude changes, vertical speeds) make up 48.7% of all commands, followed by horizontal commands *making up* another 22.6% (direction commands, heading changes), frequency change commands 12.8%, and speed changes being the least frequently occurring category with only 4.5% of total commands. There is also a difference in distribution between the LVNL and MUAC datasets, with the LVNL dataset having more vertical and speed commands and the MUAC having more frequency and horizontal commands. These differences likely reflect the operational roles of the respective sectors, with lower airspace having more climb and descent phases or commands, while upper airspace control emphasizes cruise-level coordination.

The *Instruction Entities* follow the same ontology [7]: each instruction is parsed to *command* (type), *value*, *unit*, *qualifier*, and *condition*. Examples of these fields appear in Figure 4.

Callsigns are always extracted in their ICAO format [18] for standardization. For example, in practice, callsign `KLM1999` may be spoken as `KLM one niner niner niner` or `KLM one triple niner`, yet both map to the same ICAO string.

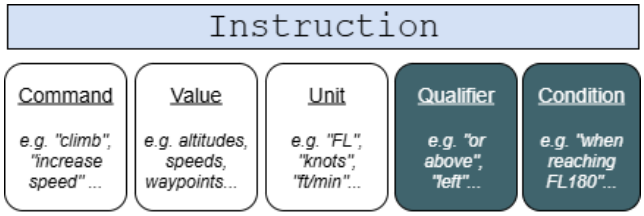


Figure 4. ATC instruction entities, based on ontology from SESAR PJ16-04 [7].

Not all instructions contain all entities. The prevalence of each field in the combined LVNL and MUAC datasets is shown in Figure 5. Callsigns and commands appear in every instruction, but not all of them include a value or unit. An example of such an instruction is `KLM123, continue climb`. Furthermore, only a small fraction of commands include a qualifier or a condition.

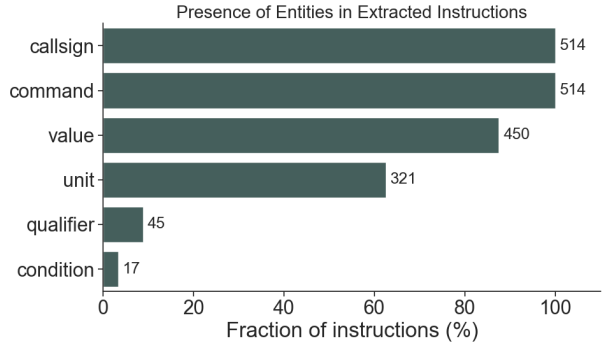


Figure 5. Occurrence frequency for each entity in the combined LVNL and MUAC dataset.

C. Prompt Design and Variants

LLM performance on information extraction is sensitive to how the prompt is formulated. Clear task description, consistent output formatting, and structured examples improve accuracy and reproducibility [11]. Prompt development is guided by small-scale trials on the *LVNL-dev* dataset, which is excluded from final evaluation. All prompts follow the same template:

- Brief task description and role specification
- Required JSON output schema
- Optional contextual information (e.g., nearby callsigns)
- Five few-shot examples (sourced from *LVNL-dev*)

We construct three prompt variants that differ only in the contextual information provided (Table IV). Each variant is used for both human and ASR transcripts. ASR prompts additionally warn about possible recognition errors and include noisy examples but remain otherwise identical. The core instruction block shared by all prompts is:

```
You are an ATC transcript parser. Extract essential
 information from each instruction and return a
JSON array of objects with fields \{callsign,
command\_category, command, value, unit, qualifier,
 condition\}. Output must be valid JSON only, with
no explanations or formatting.
```

Each example in the prompts includes both the input transcript, context information for CS and CX families, and an expected JSON output. Below is an example used in our experiments with CS family prompts on human transcripts.

```
Input transcript: corendon eight lima echo good day
 climb flight level two five zero

Nearby callsigns: CAI66JF, CAI8LE, DAL161, KLM1755,
 KLM7910, RYR8JF, SAS1555, TRA55L

Output JSON:
[
  {
    "callsign": "CAI8LE",
    "command_category": "vertical command",
    "command": "Climb",
    "value": "250",
    "unit": "FL",
    "qualifier": null,
    "condition": null
  }
]
```

D. Inference and Hardware Specifications

We use two NVIDIA A100 and V100 GPUs. A100 GPUs are used to run the Llama 3.3 70B model, while V100s are used for Whisper inference and for running/fine-tuning Gemma 3. Unless stated otherwise, the *temperature* parameter for LLM inference is set to zero in order to improve reproducibility and reduce the randomness of the outputs.

E. LLM Fine-tuning

Fine-tuning is applied to the smaller Gemma 3 (4B) model to test whether limited domain-specific data can boost performance, as its lightweight architecture makes it feasible to fine-tune within hardware constraints compared to the larger Llama 3.3. Gemma 3 (4B, instruction-tuned) is further fine-tuned using *Unsloth*[3] on input–output pairs constructed with the Callsigns-Only (CS) prompt *without* embedded examples. Five-fold cross-validation is performed on the combined LVNL-ACC1 and MUAC-Delta datasets with a split of 70/10/20 for training/validation/testing, respectively. For each fold, the model is trained for one epoch. Parameter-efficient fine-tuning (PEFT) via Low-Rank Adaptation (LoRA) is applied [19], [20]. A fixed random seed is selected to ensure reproducibility.

[3]https://docs.unsloth.ai/

F. Evaluation Metrics

The manually annotated labels (JSON-formatted, with entities and categories) are used as reference for evaluating the performance of the SIE module. Since utterances may contain multiple commands, a greedy one-to-one matching algorithm is used to align human annotated and LLM predicted commands. For each human annotated command, all unmatched predictions are scored based on the number of identical fields after normalization (command category, command, value, unit, qualifier, condition, and callsign). Normalization consists of lowercasing and collapsing extra whitespace to ensure consistent comparison. The prediction with the highest score is selected as the match, with each prediction used at most once.

For each matched command pair, entity fields are compared individually. A True Positive (TP) is counted when the manual annotation and the prediction contain the same value. A False Negative (FN) occurs when the human annotation specifies a value but the prediction either omits it or gives a different value. Conversely, a False Positive (FP) occurs when the prediction provides a value that is absent from the human annotation or differs from it. We evaluate the command category assignment performance using the F1 score, defined as the harmonic mean of precision and recall:

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{1}$$

where Precision $= TP/(TP+FP)$ and Recall $= TP/(TP+FN)$.

The entity extraction task covers six structured entity fields: `callsign`, `command`, `value`, `unit`, `qualifier`, and `condition`. Micro-F1 is chosen as the primary metric because it reflects overall extraction accuracy by weighting each prediction equally, regardless of the entity type. This is particularly important in our setting, where the dataset is imbalanced (e.g., callsigns are much more frequent than conditions), and we aim to measure the model's aggregate performance across all entities. Micro-F1 is calculated by aggregating true positives (TP), false positives (FP), and false negatives (FN) across all six fields and all utterances within an evaluation split, as shown in Equation 2.

$$\text{Micro-F1} = \frac{2 \cdot TP_{\text{total}}}{2 \cdot TP_{\text{total}} + FP_{\text{total}} + FN_{\text{total}}} \tag{2}$$

Finally, entity frequency is defined as the number of times a specific entity appears in human annotations (TP + FN).

IV. Results

First, we evaluate the performance of SIE using the Llama model on different prompts and transcripts. Figure 6 summarizes micro-F1 scores across the three different prompt strategies and across human and Whisper transcripts. Performance increases consistently with both higher-quality transcripts and richer prompts. Moving from *No Context* (NC) to *Callsigns*

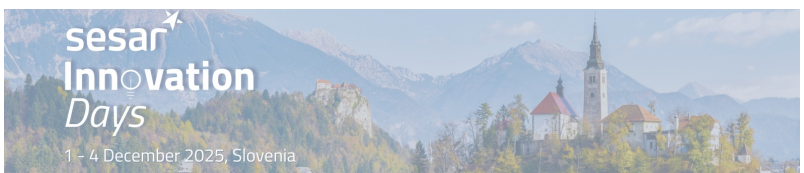


TABLE IV. PROMPT VARIANTS AND INCLUDED CONTEXT. ALL PROMPTS ALSO INCLUDE FIVE EXAMPLES AND JSON OUTPUT FORMAT REQUIREMENTS.

Prompt family	Context included
No Context (NC)	Airline designators only, no nearby callsigns or any sector-specific information.
Callsigns Only (CS)	Nearby callsigns written in ICAO format (2-hour scenario window) and relevant airline designators only.
Additional Context (CX)	Two-stage prompting: extract callsign and command category, then add targeted context based on category (route waypoints for horizontal, sector frequencies for frequency change, plausible speed ranges for speed, sector altitude ranges for vertical).

Only (CS) yields gains of approximately 0.05-0.07 F1 across all transcript types, with the largest improvement observed for baseline Whisper transcriptions. Adding *Additional Context* (CX) provides a further but smaller improvement of 0-0.02, with the effect diminishing as transcription quality increases. When using human transcripts, the model reaches a ceiling of **0.91** micro-F1 under both CS (0.912) and CX (0.914) prompts.

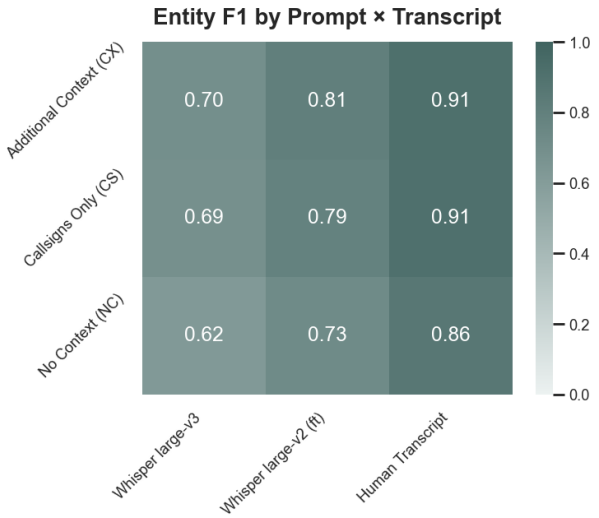


Figure 6. Micro-F1 by prompt family and transcript source. Each cell reports the micro-averaged F1 score for one prompt (rows) and transcription condition (columns).

Transcript quality seems to be more impactful than context-enhanced prompting: fine-tuned Whisper-v2 significantly outperforms Whisper-v3 by around 0.09–0.11 by micro-F1 scoring, with gold transcripts making an additional difference of 0.10–0.13. The impact of transcript quality on the SIE output can also be visualized in Figure 7, where the three data points corresponding to the three different transcripts indicate a strong correlation between the transcript WER and the SIE Micro-F1. This relation seems to be somewhat stronger when going from golden transcripts to fine-tuned Whisper outputs than going from fine-tuned to baseline Whisper. Furthermore, CS and CX prompts show a somewhat less steep decline in quality between human transcripts and baseline Whisper models, which can be explained by the fact that these prompts enhance the recognition of certain callsigns, waypoints, and values even when they are not transcribed fully correctly.

We also evaluate the performance of the Llama 3.3 model per entity to understand how these micro-F1 scores are constructed. Table V shows the performance on human transcripts using the CS prompt. In general, the LLM output contains more false positives than false negatives, explained by the

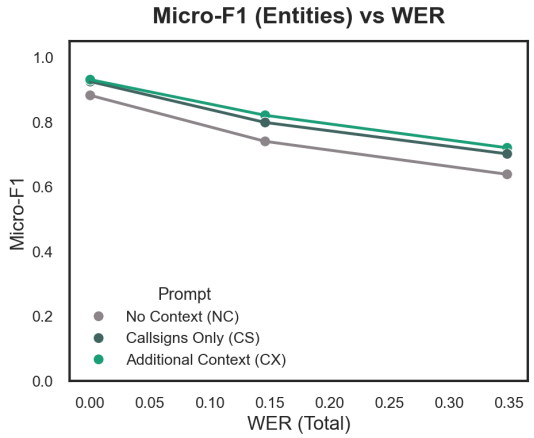


Figure 7. Effect of ASR WER on SIE micro-F1 per prompt family. The three data points correspond to gold transcriptions (WER assumed to be 0%), fine-tuned Whisper model (WER = 14.6%), and baseline Whisper model (WER = 34.8%)

fact that the LLM overpredicts the number of commands present in the transcriptions. Performance is generally higher for frequent entities, such as unit (F1 = 0.96), callsign (F1 = 0.94), value (F1 = 0.94), and command category (F1 = 0.88). In contrast, rare entities perform worse: qualifier (F1 = 0.69) and especially condition (F1 = 0.38), where both false positives and false negatives outnumber true positives. This is explained by the fact that conditions tend to be more complex and longer than other entities, and they may have multiple ways of wording, making them less likely to be an exact string match with the human annotation. For qualifiers, false positives strongly dominate false negatives, even more so than for other entities, indicating that the LLM tends to overpredict what counts as a qualifier.

TABLE V. ENTITY-LEVEL F1 SCORES FOR THE LLAMA MODEL USING THE *CS* PROMPT AND HUMAN TRANSCRIPTS. MICRO-F1 = 0.912

Entity	TP	FP	FN	Frequency	F1
callsign	489	40	25	514	0.94
command	460	72	54	514	0.88
value	426	27	24	450	0.94
unit	313	19	8	321	0.96
qualifier	37	25	8	45	0.69
condition	6	9	11	17	0.38

A. Sector Differences

Furthermore, sector generalizability of the Llama-based SIE is evaluated by calculating evaluation metrics on the LVNL and MUAC datasets separately, with the results displayed in Figure 8. It seems that although prompts are tailored according to the LVNL-dev dataset, only a difference of 0.03–0.04 in F1 score

is observed on human transcription and baseline Whisper-large-v3 model performances across sectors. There is a larger discrepancy in the fine-tuned Whisper output performance, but this can also partially be attributed to the difference in WER between the sectors as shown in Table III.

The sector difference in the fine-tuned Whisper performance is also evident from the fact that callsign recognition is better for the MUAC-Delta sector than for LVNL-ACC1 in human transcriptions, but the opposite is true for the fine-tuned Whisper SIE, where there are likely more callsign transcription errors (Tables VI and VII).

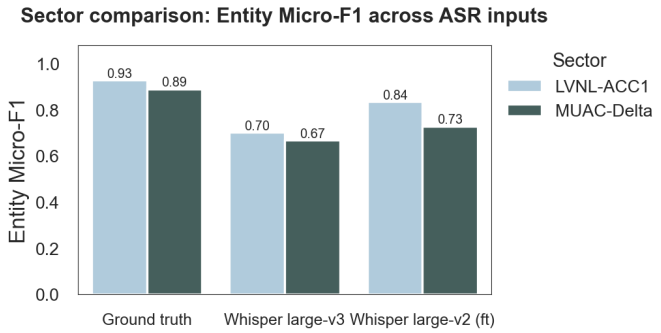


Figure 8. Entity micro-F1 scores per sector. CS prompt used.

TABLE VI. ENTITY-LEVEL MICRO-F1 SCORES BY SECTOR FOR THE CS PROMPT AND HUMAN TRANSCRIPTS.

Entity	LVNL-ACC1	MUAC-Delta
callsign	0.93	0.97
command	0.89	0.81
condition	0.67	0.00
qualifier	0.95	0.64
unit	0.97	0.94
value	0.94	0.93

TABLE VII. ENTITY-LEVEL MICRO-F1 SCORES BY SECTOR FOR THE CS PROMPT AND THE FINE-TUNED WHISPER TRANSCRIPTS.

Entity	LVNL-ACC1	MUAC-Delta
callsign	0.76	0.71
command	0.86	0.76
condition	0.42	0.00
qualifier	0.83	0.53
unit	0.95	0.92
value	0.82	0.66

B. Latency Evaluation

To assess the impact of prompting strategies on inference speed, we measured latency with Llama 3.3 on two A100 GPUs. Table VIII reports average prompt length (tokens) and inference time for three schemes: no context (NC), callsigns only (CS), and additional context (CX).

NC prompts have the shortest length (828 tokens) and the fastest inference time (2.6 s). In contrast, the CS scheme increases prompt length (1956 tokens) by including nearby callsigns in examples and the input transcript, which correspondingly raises latency (3.3 s). The CX scheme, which uses a two-step process, results in 1560 tokens in the first step and 882 in the second, with a total inference time of 3.8 s.

These measurements show that prompt structure measurably affects latency, with differences of around 1-1.2 s between the simplest (NC) and most complex (CX) schemes. The reported numbers are indicative rather than absolute, as inference times depend on hardware and system configuration.

TABLE VIII. AVERAGE INPUT LENGTH (TOKENS) AND TOTAL INFERENCE TIME FOR DIFFERENT PROMPT SCHEMES USING LLAMA 3.3 ON TWO A100 GPUS. TESTED SCHEMES ARE NO CONTEXT (NC), CALLSIGNS ONLY (CS), AND ADDITIONAL CONTEXT (CX).

Prompt	Avg. Length (tokens)	Avg. Time (s)
NC	828	2.6
CS	1956	3.3
CX	1560 (step 1), 882 (step 2)	3.8

C. Gemma 3 (4B) Baseline Performance

To evaluate the effect of LLM model size and family, the Gemma 3 4B model is evaluated with the CS prompt on both human transcriptions and Whisper-large-v3 transcriptions. Gemma is a smaller model and, while it is newer than Llama 3.3, the expectation is that due to the difference in the number of parameters it still performs worse.

Both entity micro-F1 and category extraction F1 scores are shown and compared with Llama 3.3 in Table IX. It is evident that the Gemma model performs significantly worse than the Llama model on both category and entity extraction. Gemma 3 4B reaches an entity micro-F1 score of 0.70 under the CS prompt.

TABLE IX. GEMMA VS LLAMA ON CS PROMPT FOR TWO TRANSCRIPT SOURCES.

Transcript	Metric	Gemma 3-4B (baseline)	Llama-3.3-70B
Human Transcript	Entity F1	0.70	0.91
	Category F1	0.80	0.98
Whisper large-v3	Entity F1	0.58	0.70
	Category F1	0.65	0.88

D. Gemma 3 (4B) fine-tuned Performance

While the performance of the baseline Gemma 3 model is lower than that of Llama 3.3, it has the advantage of being a lighter and faster model. This applies both to inference and fine-tuning. The effects of fine-tuning and testing this 4B model on the combined LVNL/MUAC dataset are shown in Table X. It is evident that fine-tuning improves the entity extraction performance significantly, with the entity micro-F1 increasing after fine-tuning on fewer than 400 samples from 0.70 (baseline) to 0.81 (fine-tuned). On the other hand, the command classification does not perform significantly better. This suggests a steeper learning curve for the Gemma model to understand the task and what each extracted field entails, rather than understanding the broader context of the utterances.

V. INSIGHTS ON ERROR CASES

While F1 scores provide a good overview of the SIE module's overall performance, it's also important to qualitatively examine the LLM outputs to understand the nature of the

TABLE X. PERFORMANCE OF DIFFERENT LLMS ON HUMAN TRANSCRIPTS, USING THE CALLSIGNS ONLY PROMPT.

Model	Entity F1	Category F1
Llama 3.3	0.91	0.98
Gemma 3 (baseline)	0.70	0.80
Gemma 3 (fine-tuned)	0.81	0.84

errors being made, not just their frequency. This is particularly relevant when considering real-world deployment, such as using the model to assist or automate tasks for a pseudo-pilot.

A. Beyond Exact String Matching

It is important to note that exact string matching tends to underestimate performance, as many predicted outputs are semantically equivalent to the human labels despite lexical differences, for example:

- `Climb` vs. `Climb to`
- `128.4` vs. `128.400`
- `maintain speed` vs. `keep speed`
- `Amsterdam sector 2` vs. `Amsterdam sector two`
- `call` vs. `contact`

Moreover, the boundaries between certain entities are not always clearly defined, even for human annotators. For example, an instruction like `expedite your climb` could be interpreted as a single command (`expedite climb`) or as a command with an associated value (`expedite`, `climb`). Such ambiguity is particularly common in the *qualifier* and *condition* fields, which partly explains why these fields consistently have the lowest accuracy and recall across experiments.

To better reflect semantic understanding, a subset of experiment outputs was manually re-scored with semantic matching in mind. Semantically equivalent entities and commands were counted as correct, resulting in higher micro-F1 scores. For CX on human transcriptions, the micro-F1 increased from 0.91 to 0.95 (see Table XI). For fine-tuned Whisper with CX, it increased from 0.81 to 0.86. As expected, per-entity scores, particularly for fields with more ambiguous definitions or complex structures such as *qualifier* and *condition*, are also notably higher than in the exact matching evaluation (see Table V).

TABLE XI. FIELD-LEVEL SCORES WITH MANUAL SEMANTIC MATCHING (CX ON HUMAN TRANSCRIPTS).

Field	Precision	Recall	F1
command category	0.99	0.99	0.99
command	0.96	0.96	0.96
value	0.98	0.96	0.97
unit	0.98	0.98	0.98
qualifier	0.72	0.89	0.80
condition	0.67	0.59	0.62
callsign	0.96	0.96	0.96

B. Representative Error Types

Even after adjusting for semantically correct outputs that were penalized by exact string matching, errors remain in the SIE outputs, even for gold transcriptions. After a manual inspection, certain types of recurring errors and trends emerge which are described below.

a) Logical/format errors (numbers and corrections).: The model occasionally misinterprets verbal numerals or fails to apply correction phrases as intended. These errors typically affect callsign and value extraction and are traceable to specific patterns in the input. Table XII shows examples where the predicted callsign was incorrect despite the correct callsign being evident from the transcript, and the predicted one not appearing in the list of nearby callsigns. A mitigation strategy here would be to add an extra check ensuring the retrieved callsign is in the list of nearby callsigns.

TABLE XII. LOGICAL HANDLING OF NUMERALS AND CORRECTIONS.

Utterance	Human Label	Prediction
KLM one triple nine climb FL two one zero	CS = `KLM1999`	CS = `KLM139`
KLM eight eight, correction KLM one one eight eight, descend FL seven zero	CS = `KLM1188`	CS = `KLM88`

b) Hallucinations.: Occasionally, the model generates values that are not supported by the input text, even when it seems to parse the rest of the instruction perfectly correctly. An example of this type of error is given in Table XIII. This instruction is parsed completely correctly, except for the value (frequency). Although this was the only such hallucination to be observed in the CX prompt/human transcript outputs, this type of errors particularly problematic because they lack any apparent grounding in the input. These errors cannot be explained by transcription noise or misinterpretation, and therefore represent true model hallucinations.

Large language models are known to exhibit this behavior [21], [22]. While difficult to eliminate, some mitigation strategies include applying post-processing steps to LLM outputs: (i) checking if the output was mentioned in the utterance as well, (ii) lexicon/range validation for frequencies, headings, and flight levels, and (iii) a brief self-verification step that prompts the model to check and revise its own answer, which can reduce the rate of hallucinations significantly [23].

TABLE XIII. EXAMPLE OF A HALLUCINATED VALUE.

Utterance	Human label	Prediction
Ryanair six juliett fox call Amsterdam one two three seven zero five	Val = `123.705`	Val = `123.770` (altered)

c) Waypoints Or other Unfamiliar Words: It was a curious observation that the waypoint `OMELO` was parsed incorrectly on all three occasions it appeared in the dataset, across all prompt families and ASR inputs. In the full-context prompt, it was rendered as `OMELLO` or `OMELO`, while in callsign-only prompts it appeared as `OMEL` or `OMELON`. This behavior is likely due to out-of-vocabulary (OOV) effects [24]: `OMELO` is a domain-specific term that likely does not appear in Llama's training data and is tokenized into rare or unfamiliar subword units, resulting in unstable output. These errors can be mitigated by applying a post-processing step that checks against a known list of valid waypoints.

d) Entity ambiguity.: As mentioned earlier, the LLM struggled with some instructions that have a more ambiguous

mapping to structured entities. A common example involves climb or descent instructions that also specify a vertical speed. These were sometimes parsed as two separate commands, and other times as a single `Climb`/`Descend` command with the vertical speed included as a *condition*. For instance, the instruction of `Sunexpress seven tango tango, descend flight level two six zero, two thousand feet per minute or greater` could be parsed either as:

(i) two commands: a `Descend` command with value `FL260`, and a separate `Vertical Speed` command with value `2000 fpm or greater`, or
(ii) a single `Descend` command with value `FL260` and the vertical speed expressed as a *condition*.

Such inconsistencies are not critical and can likely be mitigated by incorporating clearer descriptions or representative examples into the prompt.

C. ASR Error Propagation

Finally, SIE output errors that can be traced back to ASR mistakes are considered. Table XIV presents examples of SIE errors that arise directly from transcription mistakes. Waypoints, in particular, tend to cause issues: they are often misrecognized by the ASR system, likely due to their rarity and absence from general training data. Even when the list of waypoints is provided as context in the prompt, the SIE module often fails to recover them correctly. In some cases, these errors also affect the extraction of other entities within the same utterance if they are transcribed incorrectly.

VI. Conclusions

This study demonstrates the feasibility of using open-source large language models (LLMs) for structured information extraction (SIE) from air traffic control (ATC) communications. Our central finding is that transcription quality is the most influential factor; improvements in ASR word error rate directly translate to better extraction accuracy. While larger models like Llama 3.3 achieve the strongest performance, compact, fine-tuned models such as Gemma 3 present a resource-efficient alternative. Furthermore, few-shot prompting with targeted context, like nearby callsigns, offers a practical method for boosting accuracy without large annotated datasets.

Taken together, these results provide evidence that an end-to-end speech-to-instruction pipeline can reach promising performance. However, qualitative analysis highlights recurring errors, including over-prediction and hallucinations, that necessitate verification steps before deployment. Such measures, combined with latency-aware design, could make LLM-based SIE suitable for research, training, and integration into real-time pseudo-pilot systems.

This work was limited by a relatively small dataset of area control scenarios, manual annotations, and hardware constraints that restricted the fine-tuning of larger models. The evaluation focused on open-source LLMs and did not explore advanced prompt engineering for the ASR module or the use of dynamic context, which represent avenues for future work.

In establishing a benchmark for LLM-based SIE on ATC data, this study confirms the effectiveness of lightweight prompting and fine-tuning strategies. It also identifies concrete directions for improving reliability through verification layers and tighter ASR integration. These findings position LLMs as a flexible and scalable foundation for structured ATC command extraction, with applications ranging from automated training support to future safety-monitoring tools.

VII. Discussion and Future Research Directions

This work demonstrates that open-source LLMs can extract structured ATC commands from both human and ASR transcripts with promising accuracy. Several aspects, however, warrant further discussion and identify opportunities for future research.

A. Operational Use Beyond Simulations

Although this study focuses on a simulator setting, the approach also shows potential for operational ATC environments. Structured extraction could enable applications such as conformance monitoring, early deviation detection, post-event analysis, and automated generation of structured logs. Operational deployment, particularly for safety-critical tasks, requires substantially higher reliability than simulator use. Moreover, real-world audio quality is often lower and more variable, which degrades ASR performance. Fine-tuning ASR models on operational data can mitigate this, but depends on access to high-quality transcriptions and the training of new ASR systems. Because SIE accuracy is strongly correlated with ASR performance, future work should evaluate the full pipeline on operational corpora to assess robustness under realistic conditions.

B. Generalization Across Sectors and Scenarios

Our evaluation across two en-route sectors revealed minimal differences in SIE performance between area control and upper-area control environments. However, other domains, such as tower control, introduce distinct command types and different command distributions. Evaluation across other sectors is therefore needed to determine how well the SIE module performs in environments that involve a broader and more complex set of instructions.

C. Fine-Tuning Behavior and Data Efficiency

The fine-tuned Gemma 3 model exhibited notable performance gains despite being trained on fewer than 400 samples. This data efficiency likely stems from (i) the highly structured label schema, (ii) the constrained linguistic domain of standardized ATC phraseology, and (iii) the Gemma model's instruction-tuned architecture, which benefits from light supervision. Future work should examine whether similar gains occur in other compact models and how performance scales with larger and more diverse datasets.

D. Error Mitigation

Hallucinations, although rare, pose a significant risk to SIE performance and therefore require mitigation strategies. Simple approaches include rule-based verification layers or querying the LLM two or more times to filter out errors. A more robust and systematic method involves Retrieval-Augmented Generation (RAG), which anchors outputs to scenario-specific lexicons such as flight plans and frequencies. These methods can improve consistency and reduce the

TABLE XIV. REPRESENTATIVE ASR TO SIE ERROR PROPAGATION EXAMPLES. FINE-TUNED WHISPER AND CX PROMPT USED

Reference utterance	ASR output	Downstream extraction error
"eight one november hello climb flight level ah two five zero"	"eight two november hello climb flight level ah two five zero"	CS = EWG82N instead of KLM81N (not in nearby list)
"KLM three three yankee hello climb to flight level two five zero"	"KLM three three yank hello cleared to fly at flight level two five zero"	Command = Cleared to instead of Climb to
"one zero hotel good day continue on the arrival landing runway one eight center"	"one zero hotel good day equal to you ahm only ryanair four landing runway will be one eight center"	Command = Landing runway, Callsign = RYR
"...resume own navigation direct RAVLO"	"...resume own navigation to the right hello"	Qualifier = to the right, waypoint missed
"...direct to BUREK"	"...direct to BUREQ"	Value = BUREQ instead of BUREK

occurrence of hallucinations. Future research could investigate the integration of RAG for this task.

E. Use of Contextual Data

This study used only contextual information available prior to simulations, such as predetermined flight plans. Incorporating dynamic, real-time data, such as aircraft positions or flight parameters at the moment an instruction is issued, could provide more targeted context for LLM queries. Although integrating dynamic context increases system complexity, it is expected to improve SIE performance by resolving ambiguities and providing more focused and relevant information. Investigating how dynamic context can be integrated effectively, and how much it contributes to overall performance gains, represents another potential direction for future research.

REFERENCES

[1] 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.

[2] J. Zuluaga-Gomez et al., *ATCO2 corpus: A Large-Scale Dataset for Research on Automatic Speech Recognition and Natural Language Understanding of Air Traffic Control Communications*, Jun. 2023. DOI: 10.48550/arXiv.2211.04054.

[3] J. Li, A. Sun, J. Han, and C. Li, "A Survey on Deep Learning for Named Entity Recognition," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 50–70, Jan. 2022, ISSN: 1041-4347, 1558-2191, 2326-3865. DOI: 10.1109/TKDE.2020.2981314.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[5] A. Vaswani et al., "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[6] J. Zuluaga-Gomez et al., "Lessons Learned in Transcribing 5000 h of Air Traffic Control Communications for Robust Automatic Speech Understanding," en, *Aerospace*, vol. 10, no. 10, p. 898, Oct. 2023, ISSN: 2226-4310. DOI: 10.3390/aerospace10100898.

[7] H. Helmke 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)*, Sep. 2018, pp. 1–10. DOI: 10.1109/DASC.2018.8569238.

[8] A. K. Y. Low, L. Nimrod, S. Alam, and L. C. K. Poh, "Deep neural network-based automatic speech recognition for atc-pilot audio transcription," 2024.

[9] S. Chen, W. Pan, Y. Wang, S. Chen, and X. Wang, "Research on the Method of Air Traffic Control Instruction Keyword Extraction Based on the Roberta-Attention-BiLSTM-CRF Model," *Aerospace*, vol. 12, no. 5, 2025, ISSN: 2226-4310. DOI: 10.3390/aerospace12050376.

[10] P. Jiang, C. Zeng, W. Pan, B. Han, and J. Zhang, "SLKIR: A framework for extracting key information from air traffic control instructions Using small sample learning," en, *Scientific Reports*, vol. 14, no. 1, p. 9791, Apr. 2024, ISSN: 2045-2322. DOI: 10.1038/s41598-024-60675-6.

[11] T. B. Brown et al., "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[12] J. Dagdelen et al., "Structured information extraction from scientific text with large language models," en, *Nature Communications*, vol. 15, no. 1, p. 1418, Feb. 2024, ISSN: 2041-1723. DOI: 10.1038/s41467-024-45563-x.

[13] S. Wang et al., *GPT-NER: Named Entity Recognition via Large Language Models*, Oct. 2023. DOI: 10.48550/arXiv.2304.10428.

[14] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, *Robust Speech Recognition via Large-Scale Weak Supervision*, Dec. 2022. DOI: 10.48550/arXiv.2212.04356.

[15] J. van Doorn, J. Sun, J. Hoekstra, P. Jonk, and V. de Vries, "Whisper-atc: Open models for air traffic control automatic speech recognition with accuracy," ICRAT–2024, 2024.

[16] A. Dubey et al., *The Llama 3 Herd of Models*, Nov. 2024. DOI: 10.48550/arXiv.2407.21783.

[17] A. Kamath et al., *Gemma 3 Technical Report*, Mar. 2025. DOI: 10.48550/arXiv.2503.19786.

[18] *Manual of radiotelephony*, 4th ed., Doc 9432 AN/925, International Civil Aviation Organization, Montréal, Canada, 2007, ISBN: 92-9194-996-5.

[19] E. J. Hu et al., *LoRA: Low-Rank Adaptation of Large Language Models*, Oct. 2021. DOI: 10.48550/arXiv.2106.09685.

[20] Z. Han, C. Gao, J. Liu, J. Zhang, and S. Q. Zhang, *Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey*, Sep. 2024. DOI: 10.48550/arXiv.2403.14608.

[21] Z. Ji et al., "Survey of Hallucination in Natural Language Generation," *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, Dec. 2023. DOI: 10.1145/3571730.

[22] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, *On Faithfulness and Factuality in Abstractive Summarization*, May 2020. DOI: 10.48550/arXiv.2005.00661.

[23] S. Tonmoy et al., "A comprehensive survey of hallucination mitigation techniques in large language models," *arXiv preprint arXiv:2401.01313*, vol. 6, 2024.

[24] R. Sennrich, B. Haddow, and A. Birch, *Neural Machine Translation of Rare Words with Subword Units*, Jun. 2016. DOI: 10.48550/arXiv.1508.07909.