Aviation Parser: A Knowledge-Guided Self-Evolving Optimization Framework with LLMs for NOTAM Understanding

Anonymous ACL submission

Abstract

Accurate parsing of Notices to Airmen (NOTAMs) is critical for aviation safety, yet many existing methodologies struggle with template rigidity that hinders effective handling of non-standard syntax, regional expression ambiguities and the semantic-practice gap. In this paper, we propose a knowledge-guided self-evolving optimization framework that integrates Large Language Models (LLMs) with an Aviation Knowledge Graph (AviationKG) in order to achieve efficient structured NOTAM parsing. This framework consists of three innovative modules: 1) Knowledge-Enhanced Retrieval (KG-TableRAG), which resolves semantic ambiguities through binding of knowledge graph relations with infrastructure tables to constrain search spaces; 2) Self-Evolving Optimization (SEVO), which employs dynamic preference alignment and error-driven curriculum learning to iteratively enhance complex instruction compliance; 3) Consensus Inference Engine (CIE), which improves edge-case robustness via terminology-preserved input diversification and majority voting decoding. Experimental results demonstrate that our framework achieves a *30.4%* accuracy improvement over the base model within 3-5 iterations on a labeled dataset of 10,000 global NOTAMs. Ablation studies further validate the collaborative effectiveness of its modular components. This research establishes the first knowledge-driven, iteratively optimized LLM solution for aviation text parsing, with a methodology extensible to other high-precision-demanding professional domains.

1 Introduction

Accurate NOTAM (Notice to Airmen) interpretation is a critical yet challenging component of modern flight operations. These specialized bulletins contain time-sensitive information regarding temporary airspace restrictions and navigational hazards, characterized by linguistic features distinct from conventional technical documentation. With over one million active NOTAMs published annually worldwide (Morarasu and Roman, 2024), the aviation industry urgently needs robust automated analysis to reduce manual workload and errors. Existing systems mostly use rule-based template matching, focusing research on automated rule discovery or basic classification (Dieter et al., 2024; Mi et al., 2022).

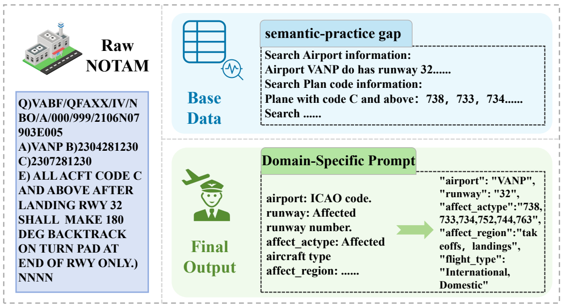

Figure 1: An illustration of NOTAM analysis task

However, NOTAMs present unique parsing challenges due to their heavy dependence on 300+ standardized abbreviations and non-standard syntactic structures (e.g., "RWY 09L/27R CLSD DUE BIRD ACT"), which frequently violate conventional parsing rules. Furthermore, practical scenarios introduce additional complexity through regional expression variations (e.g., "EGBA" encompassing both EGBA1A and EGBA1B) and typographical/grammatical errors. Beyond syntactic, a key challenge is the **semantic-practice gap**: the disconnect between textual descriptions and operational impacts requires implicit correlation with aviation infrastructure status. For instance, interpreting "APCH LGT U/S" necessitates knowledge of specific runway configurations, yet NOTAMs may reference non-existent runways or omit critical identifiers (Patel et al., 2023). These operational constraints rely on factual data from regularly updated official sources, as illustrated in Figure 1.

Recent advancements in large language models (LLMs), characterized by their sophisticated natural language understanding capabilities, are paving new frontiers for NOTAM analysis. While no prior studies specifically address LLM applications in this domain, recent breakthroughs in complex instruction following and generic information extraction (Morarasu and Roman, 2024) establish critical technical foundations. Essentially, LLMs can be adapted to take raw NOTAM text as input and, with appropriate guidance, produce structured information as output. However, NOTAMs present unique challenges, including domain-specific language and the need to link the text to structured aviation data. Building on this, our LLM-adapted framework for NOTAM analysis makes three pioneering contributions to bridge this gap and enable effective NOTAM parsing:

- **Knowledge-Driven Architecture**: Innovating the inaugural application of LLMs to NOTAM parsing, our framework integrates an aviation knowledge graph with TableRAG retrieval to overcome domain-specific challenges through constraint-aware information extraction.
- **Self-Optimizing Pipeline**: Through synergistic integration of dynamic preference alignment, error-driven curriculum learning, and consensus inference mechanisms, we establish an end-to-end optimizable system capable of self-evolution without manual intervention.
- **Empirical Performance Leap**: Experimental validation demonstrates our optimized model achieves a 30.4% accuracy improvement over base LLMs, with multi-perspective analysis and majority voting decoding.

2 Related Work

2.1 NOTAM Analysis

NLP plays a key role in reducing manual operations in aviation, particularly in NOTAM analysis (Mogillo-Dettwiler, 2024; Mi et al., 2022). Early efforts explore transformer-based models trained on unlabeled NOTAMs for filtering and inconsistency correction (Bravin et al., 2020). NLP workflows (TF-IDF, topic modeling, NER) are applied to a large dataset for automated segmentation and tagging (Clarke et al., 2021). Further work utilizes pre-trained BERT models on 1.2 million NOTAMs for aviation knowledge extraction (Arnold et al., 2022). These pioneering studies, while significant, reveal persistent challenges (e.g., ambiguous abbreviations, semantic-practical mismatches, regional variations) that impact safety and efficiency (Morarasu and Roman, 2024). Our work builds upon these insights by proposing a more adaptive and robust LLM-based framework specifically designed to handle these complexities in practical NOTAM parsing.

2.2 Recent Advances in Aviation NLP

Recent advances in aviation NLP have significantly improved flight operations and safety management through enhanced text understanding and information extraction. Spoken instructions are integrated into flight trajectory prediction, improving accuracy (Guo et al., 2024). A graph-based approach captures trajectory point attribute relationships (Fan et al., 2024). Trajectory prediction is framed as a language modeling task using fine-tuned LLMs, though facing latency issues (Luo and Zhou, 2025). For communication, an NLP agent is introduced for pilot training phraseology compliance (Liu et al., 2024). Additionally, deep learning is applied to classify flight phases in safety reports, automating analysis (Nanyonga et al., 2023). These developments demonstrate aviation NLP's progress in improving safety, traffic management, and communication efficiency.

2.3 Large Language Models

Advances in LLMs (Zhao et al., 2023) drive progress in specialized domains. NOTAM analysis requires integrating key capabilities like information extraction, tabular understanding, and complex instruction following. Transformer architectures (Brown et al., 2020; Chowdhery et al., 2022), enhanced by parameter scaling (Rae et al., 2021; Le Scao et al., 2022), show strong few-shot learning capabilities suitable for aviation's often sparsely labeled data (Xu et al., 2023). Key aspects include:

Knowledge Enhancement: Integrating knowledge graphs (KGs) with LLMs is shown to improve reasoning and reduce hallucinations (Ji et al., 2024; Zhang et al., 2024). KG structure reorganization and instruction fine-tuning address hallucination in complex question answering (Ji et al., 2024). KG structure is leveraged for factual reasoning enhancement (Zhang et al., 2024). The SAC-KG framework achieves high precision in domain-specific KG construction, significantly outperforming prior methods (Chen et al., 2024a). In law, Legal-LM combines KGs with keyword extraction to boost performance (Shi et al., 2024). These studies show KG-enhanced LLMs improve reliability for complex reasoning tasks.

Information Extraction: This task benefits from in-context learning (prompt engineering) techniques (Li et al., 2023) or supervised fine-tuning with instruction datasets (Wang et al., 2023). Innovations like code-style prompting (Sainz et al., 2024) and hierarchical schema representations (Li et al., 2024) improve output consistency, but conventional methods struggle with aviation's dynamic semantics and regional variations.

Tabular Understanding: Methods in this area have evolved from Text2SQL systems (Zhong et al., 2017) to more advanced neuro-symbolic approaches like TableRAG (Chen et al., 2024b). While TableRAG's query expansion helps with large tables, limitations in NOTAM contexts include context window constraints and poor cell localization with sparse schemas.

Complex Instruction Following: Research indicates that progressively adding constraints can improve a model's ability to comply with complex instructions (Mukherjee et al., 2023; Luo et al., 2024). Frameworks like Conifer (Sun et al., 2024) show potential for handling multi-level constraints. Building on this, we employ curriculum learning to optimize performance on NOTAMs' heterogeneous instruction complexity.

3 The Proposed Framework

3.1 Problem Formulation

Our primary objective is to extract structured aviation information from an input NOTAM text sequence $X = [x_1, \ldots, x_n]$, by leveraging a collection of aviation reference tables $\mathcal{T} = \{T_1, \ldots, T_m\}$ within a knowledge-enhanced generative framework. Formally, this task is defined as maximizing the conditional probability:

$$p_\theta(Y \mid X, P, K) = \prod_{i=1}^{m} p_\theta(Y_i \mid X, P, K, Y_{<i}), \quad (1)$$

where $Y = [Y_1, \ldots, Y_m]$ denotes the target structured output sequence. The term θ represents the parameters of the LLM, P encapsulates task-specific prompts and instructions, and $K = \kappa(X, \mathcal{T})$ corresponds to the factual knowledge retrieved from the aviation reference tables $\mathcal{T}$.

3.2 Framework Overview

As illustrated in Fig. 2, our framework has three stages: (1) The *Retrieval Stage* grounds predictions in aviation domain knowledge via dynamic table retrieval; (2) The *Optimization Stage* enables iterative self-improvement of the foundation model through adaptive preference learning; (3) The *Inference Stage* ensures robust parsing via diversified input generation and consensus decoding. This architecture addresses key NOTAM analysis challenges: knowledge grounding, error propagation and stability.

3.3 Knowledge-Guided TableRAG

To ensure factual consistency in NOTAM parsing results, this study proposes a knowledge graph-enhanced Table Retrieval-Augmented Generation framework (KG-TableRAG). It integrates real-time aviation infrastructure tables to overcome conventional TableRAG limitations in specialized domains. Conventional table retrieval in aviation often shows domain-specific bias from insufficient structural knowledge representation. For instance, conventional vector retrieval often fails to capture implicit cross-table correlations in "runway closure" events, such as those with lighting systems and navigation equipment. Furthermore, existing ReAct-based table retrieval methods have multi-step reasoning inefficiencies, making them impractical for time-sensitive operations.

Our KG-TableRAG framework improves upon TableRAG (Chen et al., 2024b) by systematically integrating knowledge graphs, which are constructed using open-source methodologies with manual refinements due to limited structured corpora. Upon receiving raw NOTAMs (Notices to Airmen), the framework employs LLMs to decompose queries, executes graph queries based on extracted keywords, and subsequently performs vector searches. For a detailed illustration of the domain knowledge graph architecture, please refer to Figure 4 in Appendix B.

Our implementation specifics include explicit mappings between knowledge nodes and table columns, such as dynamically binding the graph relationship `[Airport]`→`[Owns]`→`[Runway]` to operational columns like `RWY-STATUS`. This design constrains the search space to mitigate interference from irrelevant columns. A lightweight single-step inference mechanism replaces traditional multi-round decision processes by utilizing predefined

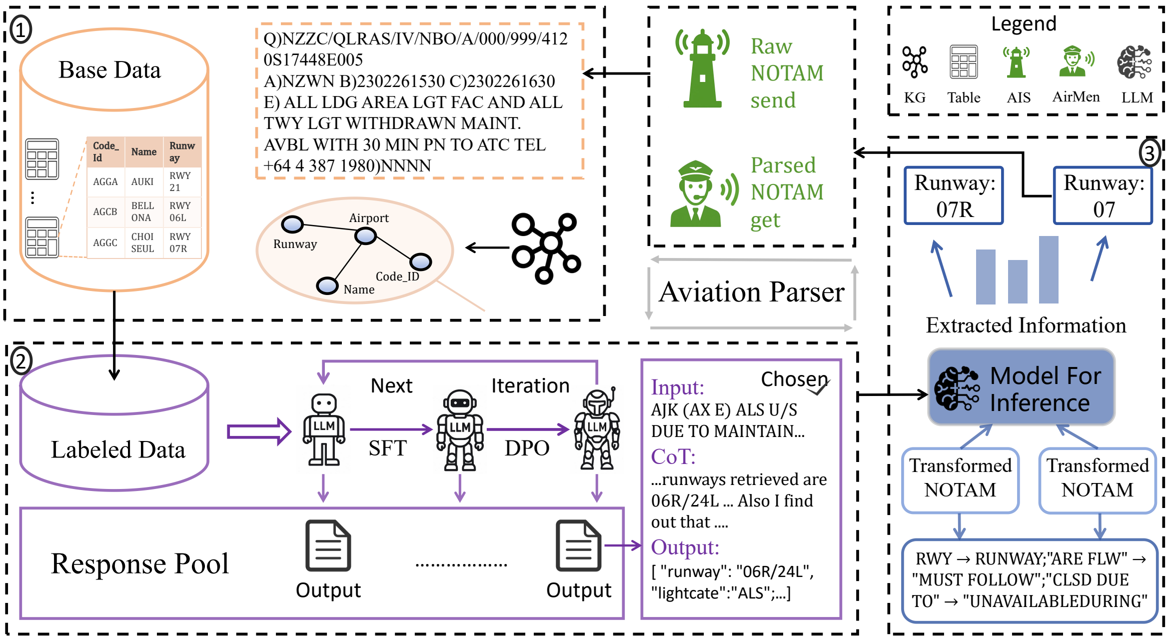

Figure 2: Overall framework of the proposed Aviation Parser: (1) Retrieval Stage: The final outputs are based on a set of base tables that represent real-world conditions, e.g., the number of runways at an airport. (2) Optimization Stage: Our foundational model gains proficiency in handling complex instructions within NOTAM analysis scenarios through iterative self-evolution. (3) Inference Stage: We rephrase the original NOTAM without altering its core content and then extract information from multiple texts to determine the final answer via a voting mechanism.

graph paths (e.g., the chained pattern *"Restriction Type-Impacted Equipment-Applicable Time Period"*) for direct Cypher query generation.

We prioritize SmolAgents over ReAct for task-specific processing due to their superior computational efficiency and scalability in complex KG-integrated queries. This substitution streamlines workflows, improves system robustness and responsiveness, and collectively yields measurable accuracy and efficiency gains in NOTAM information retrieval and analysis.

3.4 Self-Evolving Supervised and Preference Optimization

This stage iteratively refines the base model using a combination of supervised learning on correct predictions and preference learning on error signals.

Initialization Setup The process starts with:

- **Data Partitioning**: An annotated dataset $\mathcal{D}_0 = \{(x \circ K, Y^*)\}$ is partitioned into training ($\mathcal{D}_{\text{train}}$) and test ($\mathcal{D}_{\text{test}}$) sets (e.g., 8:2 ratio). Here x is the NOTAM text, $K = \kappa(x, \mathcal{T})$ is retrieved knowledge, and Y^* is the ground truth structured output.
- **Base Model**: An initial model π_0, typically an untuned open-source LLM (π_{base}).
- **Response Pool**: An indexed set $\mathcal{R}$, initially empty, to store input-output pairs $(x, Y^*, \hat{Y})$ generated across iterations e.

Iterative Optimization Loop Each iteration e (from 1 to a maximum E) involves intertwined SFT and DPO stages (See Fig. 2 and Algorithm 1 in Appendix A).

First, using the current model π_e, responses $\hat{Y}^{(e)}$ are generated for inputs $x \in \mathcal{D}_{\text{train}}$. These responses are compared with Y^* to label them as correct or incorrect, and the repository $\mathcal{R}$ is updated. The error rate for an input x, $\xi(x)$, is estimated as the fraction of the last K' generated responses that were incorrect:

$$\xi(x) = \frac{\sum_{k=1}^{K'} \mathbb{I}(\hat{Y}^{(k)} \neq Y^*)}{K'} \tag{2}$$

where K' is a hyperparameter defining the look-back window.

Next, supervised fine-tuning (SFT) is performed. Correct input-output pairs $(x \circ K, Y^*)$ are extracted from $\mathcal{R}$ to form the SFT dataset $\mathcal{D}^{(e)}_{\text{SFT}}$. The model π_e is fine-tuned by minimizing the standard negative log-likelihood loss (where m is the target sequence length):

$$\mathcal{L}_{\text{SFT}}^{(e)} = -\mathbb{E}_{(x,Y^*)\sim\mathcal{D}_{\text{SFT}}^{(e)}}\left[\sum_{i=1}^{m}\log\pi_\theta(Y_i^*|\right.$$
$$\left. x \circ K, Y_{<i}^*)\right] \quad (3)$$

Following SFT, the dynamic preference optimization (DPO) stage begins. A preference dataset $\mathcal{D}_{\text{pref}}^{(e)}$ is constructed by sampling triples (x, y^*, y^-) from $\mathcal{R}$, where y^* is a known correct response and y^- is incorrect for input x. Dynamic data augmentation is applied for inputs x with a high error rate ($\xi(x) \geq \tau$, where τ is a threshold), generating N_{aug} semantic-preserving variants $\mathcal{V}_x$. Corresponding preference triples (v, y^*, y^-) for $v \in \mathcal{V}_x$ are added to form the full DPO dataset:

$$\mathcal{D}_{\text{DPO}}^{(e)} = \mathcal{D}_{\text{pref}}^{(e)} \cup \bigcup_{\substack{x\in\mathcal{D}_{\text{train}} \\ \xi(x)\geq\tau}} \{(v, y^*, y^-) \mid v \in \mathcal{V}_x\} \quad (4)$$

Weighted curriculum learning is implemented when sampling from $\mathcal{D}_{\text{DPO}}^{(e)}$. The sampling weight $w_e(x)$ for input x at iteration e adaptively focuses on harder examples using the error rate $\xi(x)$ and a curriculum schedule $\alpha_e = \min(e/E, 1)$. Let $N = |\mathcal{D}_{\text{DPO}}^{(e)}|$, β_{weight} controls error emphasis, and E is the total scheduled iterations:

$$w_e(x) = (1-\alpha_e)\frac{1}{N} + \alpha_e \frac{\exp(\beta_{\text{weight}}\xi(x))}{\sum_{j=1}^{N}\exp(\beta_{\text{weight}}\xi(x_j))} \quad (5)$$

This transitions sampling from uniform towards error-weighted as iterations progress. Finally, the DPO loss is optimized using the SFT-updated model as the policy π_θ and the model from the start of the iteration π_e as the reference π_{ref}. Samples (x, y^*, y^-) are drawn according to $P_e(x) \propto w_e(x)$:

$$\mathcal{L}_{\text{DPO}}^{(e)} = -\mathbb{E}_{\substack{(x,y^*,y^-) \\ \sim P_e(x)}}\left[\log\sigma\left(\beta_{\text{DPO}}\log\frac{\pi_\theta(y^*|x)}{\pi_{\text{ref}}(y^*|x)} - \beta_{\text{DPO}}\log\frac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)}\right)\right] \quad (6)$$

Here, β_{DPO} is the DPO hyperparameter. The sigmoid function $\sigma(\cdot)$ transforms the scaled log-probability difference into a probability [0, 1], representing the preference likelihood (y^* over y^-), enabling direct learning from preferences. The model after DPO becomes π_{e+1}.

The iterative loop terminates when the model π_{e+1} achieves a target accuracy η on the test set $\mathcal{D}_{\text{test}}$:

$$\frac{1}{|\mathcal{D}_{\text{test}}|}\sum_{(x,Y^*)\in\mathcal{D}_{\text{test}}}\mathbb{I}(\pi_{e+1}(x \circ K) = Y^*) \geq \eta \quad (7)$$

where $\pi_{e+1}(x \circ K)$ is the model's prediction.

Empirical results show that the framework achieves commercial SOTA-level NOTAM parsing accuracy within 3-5 iterations without model distillation.

3.5 Integrated Inference Strategy

Standard QA paradigms struggle with NOTAM analysis due to models' limited complex instruction-following, often causing structural output errors. Particularly for edge cases where minor reasoning path variations could determine correctness, we observe that the baseline model (π_{R1}) generates inconsistent predictions despite demonstrating partial comprehension. To mitigate instability and preserve domain integrity, we use input diversification with consensus-based decoding. The approach begins with generating $N = 5$ semantically-equivalent NOTAM variants through controlled paraphrasing that strictly maintains original aviation terminology (e.g., preserving "RWY" abbreviations), spatiotemporal constraints, and safety-critical numerical values. Each variant undergoes independent model processing to yield candidate structured outputs $\{\hat{Y}^{(k)}\}_{k=1}^{N}$, followed by majority voting to determine the final prediction $\hat{Y}_{\text{final}} = \arg\max_Y \sum_{k=1}^{N}\mathbb{I}(Y = \hat{Y}^{(k)})$. The paraphrasing mechanism combines lexical substitution (e.g., "`CTAM`" ↔ "Controller Advisory Message"), syntactic restructuring through voice alternation, and contextual expansion with optional ICAO phraseology clarifications. Experimental validation in Section 4.3 demonstrates this technique's effectiveness, achieving 1.3% accuracy improvement by resolving 23% of borderline cases where single-pass decoding produced partially correct outputs.

4 Experiments

4.1 Experimental Setup

Datasets. We construct a specialized dataset from global NOTAMs (2024), with rule-based annotations meticulously verified by an expert aviation team for quality and operational relevance. Our dataset emphasizes real-world complexity, unlike

Model	Light	Area	Runway	Taxiway	AVG
Popular Models					
Regex Template Rule-based Matching	0.370	0.491	0.443	0.396	0.425
UIE (Lu et al., 2022)	0.270	0.380	0.320	0.430	0.350
qwen2.5-7B (Yang et al., 2024)	0.560	0.777	0.412	0.748	0.624
Mistral-7B (Jiang et al., 2023)	0.405	0.655	0.588	0.492	0.535
Llama3.1-8B-instruct (Dubey et al., 2024)	0.440	0.476	0.392	0.490	0.450
Deepseek-R1-Distill-Qwen-7B	0.410	0.484	0.446	0.492	0.458
qwen2.5-7b-instruct (SFT)	0.590	0.793	0.730	0.864	0.744
Deepseek-R1-Distill-Qwen-7B (SFT)	0.18	0.226	0.236	0.204	0.212
Deepseek-R1-Distill-Qwen-7B (ours)	0.620	0.725	**0.836**	0.868	0.762
Commercial Models					
GPT-4o (Achiam et al., 2023)	0.605	0.851	0.770	0.914	0.785
Deepseek-R1 (DeepSeek-AI et al., 2025)	**0.725**	**0.871**	0.792	**0.924**	**0.828**

Table 1: Performance comparison on four NOTAM analysis tasks. Models are grouped into Popular (including traditional methods and open-source LLMs) and Commercial (references). Underlined: Best result within the Popular Models group or the Commercial Models group, respectively. **Bold**: Overall best result across all models.

simpler benchmarks (Arnold et al., 2022), by requiring structured information extraction grounded in temporal-aligned aeronautical knowledge tables ($\mathcal{T}$). Key characteristics, including global operational patterns, significant textual variability, short validity periods, and evaluation task distributions, are detailed in Table 2, necessitating robust parsing for realistic assessment.

Statistic	Value / Count
Overall Characteristics	
Total Samples	10,000
Avg. Word Count	39.2
Avg. Valid Days	8.1
Top Region (%)	Asia (38.8%)
Top Q-Code (%)	Movement Area (M, 49.8%)
Evaluation Task Distribution	
Light	1,000
Area	4,000
Runway	2,500
Taxiway	2,500

Table 2: NOTAM Dataset Details

Baselines. We benchmark our framework against several baseline categories (detailed in Table 1). These comprise: traditional methods (a Regex Template Rule-based system and the UIE information extractor (Lu et al., 2022)); popular untuned open-source LLMs (qwen2.5-7B (Yang et al., 2024), Mistral-7B (Jiang et al., 2023), Llama3.1-8B-instruct (Dubey et al., 2024), and our specific base model, Deepseek-R1-Distill-Qwen-7B); their SFT counterparts where applicable (qwen2.5-7b-instruct and Deepseek-R1-Distill-Qwen-7B (SFT)); and high-performance commercial models as performance references (GPT-4o (Achiam et al., 2023) and Deepseek-R1 (DeepSeek-AI et al., 2025)). To ensure fairness, all LLM evaluations utilize the same domain-specific prompts.

Evaluation Rule: A prediction is considered correct only if it exactly matches the ground truth format and all annotated field values.

Implementation Details. Our implementation is based on the DeepSeek-R1-Distill-Qwen-7B model. Fine-tuning (standard SFT and our iterative optimization) efficiently utilizes the Unsloth framework with its recommended configurations. The KG-TableRAG module's core involves constructing the knowledge graph via the GraphFusion methodology (Pan et al., 2024), with human verification (KG architecture in Appendix A). Experiments were conducted on a single NVIDIA A800-80GB-PCIe GPU.

4.2 Main Results

We evaluate our optimized model (*Deepseek-R1-Distill-Qwen-7B (ours)*) against baselines on four key NOTAM analysis tasks.

Table 1 summarizes these results. Our optimized model achieves an excellent average score (AVG), significantly surpassing traditional methods. Crucially, it achieves a 30.4% absolute improvement compared to its base model (Deepseek-R1-Distill-Qwen-7B), directly validating the effectiveness of our optimization pipeline. Furthermore, our model outperforms other tested untuned open-source large language models (e.g., Mistral-7B, Llama3.1-8B) and the best-performing SFT (Supervised Fine-Tuning) baseline model (qwen2.5-7b-instruct).

Notably, employing only SFT can lead to a degradation in the base model's reasoning capabilities, suggesting that basic SFT may be insufficient for analyzing and processing such complex structured tasks, which further motivates our adoption of our SEVO (Self-Evolving Optimization) approach. Finally, as detailed in Table 1, despite our model having a significantly smaller parameter count than GPT-4o and Deepseek-R1, its performance is comparable to them. This thoroughly demonstrates the effectiveness of our proposed overall framework, showcasing its ability to achieve performance comparable to leading large-scale commercial models.

4.3 Ablation Study

To validate our design choices, we conduct systematic ablation analyses removing key components: (1) KG-TableRAG knowledge integration and (2) the Reasoning Integration mechanism. Table 3 shows the impact on AVG performance compared to the full system (0.762 AVG):

Removing KG-TableRAG (-KG) results in a 2.2% drop (0.740 AVG), particularly affecting knowledge-dependent interpretations (e.g., Q-code mapping).

Removing Reasoning Integration (-Inf. Integr.) lead to a larger 4.1% drop (0.721 AVG), confirming its importance for handling semantic ambiguity and constraints.

Removing Both (-KG -Inf. Integr.) yield the lowest performance (0.690 AVG), demonstrating the complementary necessity of both components.

KG-TableRAG	Inf. Integr.	AVG
✓	✓	0.762
✓	×	0.721
×	✓	0.740
×	×	0.690

Table 3: Ablation Study Results with Structured Knowledge (KG-TableRAG) and Reasoning Integration Components. Gray background indicates full configuration.

Figure 3 shows clear benefits from our iterative SEVO strategy. Over three iterations, complex tasks like Taxiway saw accuracy improve from 64.6% to 86.8%, and Light accuracy rose from 45% to 62%.

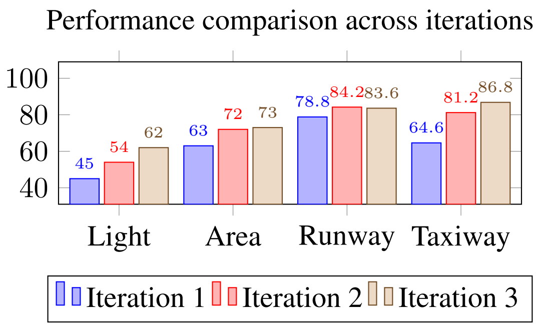

Figure 3: Iterative Optimization Performance (Accuracy %) across NOTAM Categories.

Overall, the ablation and iterative results validate the distinct contributions of each framework component: KG-TableRAG for essential knowledge grounding, Reasoning Integration for robust deduction, and SEVO's iterative optimization for continuous performance enhancement, collectively addressing the core challenges of accurate NOTAM parsing.

4.4 Complexity Analysis

We rigorously analyze the computational characteristics of our framework through three fundamental components. The dynamic preference optimization process is governed by the response pool $\mathcal{R}_x^{(t)} = \{(Y^*, \hat{Y}^{(k)})\}_{k=1}^{3K}$ containing outputs from three models per input, the sample-wise error rate $\xi(x) = \frac{\sum_{k=1}^{3K} \mathbb{I}(\hat{Y}^{(k)} \neq Y^*)}{3K}$ from (4), and the active preference pairs $\mathcal{D}_{\text{pref}}^{(t)} = \{(x, y^*, y^-) | y^* \in \mathcal{Y}_x^*, y^- \in \mathcal{Y}_x^-\}$.

As demonstrated in Table 4, the response pool grows linearly as $|\mathcal{R}_x^{(t)}| = 3Kt$ with each iteration's triple-model generation, while the actual preference pair creation instead follows quadratic scaling, modulated by accuracy progression:

$$\begin{aligned} |\mathcal{D}_{\text{pref}}^{(t)}| &= \sum_{x \in \mathcal{D}_0} |\mathcal{Y}_x^*| \cdot |\mathcal{Y}_x^-| \\ &\approx 9K^2t^2(1-\eta) \end{aligned} \tag{8}$$

where $\eta_x = \frac{1}{t} \sum_{i=1}^{t} \mathbb{I}(\hat{Y}^{(i)} = Y^*)$ tracks per-input accuracy and η denotes global performance. Our experiments reveal accuracy improvements from initial 45% to final 62%, causing the error suppression term $(1-\eta)$ to decrease from 0.55 to 0.38 through three iterations.

Metric	Iter.1	Iter.2	Iter.3
Theoretical pairs	2,415	5,915	11,320
Effective pairs	1,449	3,549	6,792
Time (h)	0.58	1.5	3.2
Scale factor	1.0×	2.6×	2.1×

Table 4: Iterative Complexity Metrics with Scaling Factors

The computational cost per iteration combines preference pair volume with curriculum learning dynamics, as quantified in Table 4:

$$\mathcal{T}_{\text{DPO}}^{(t)} = E \cdot |\mathcal{D}_{\text{pref}}^{(t)}| \cdot \mathbb{E}_{w_e(x)}[1/P_e(x)]$$
$$P_e(x) \propto (1-\alpha_e)\frac{1}{N} + \alpha_e \frac{\exp(\beta\xi(x))}{\sum_j \exp(\beta\xi(x_j))} \quad (9)$$

where E denotes training epochs and $\alpha_e = \min(e/E, 1)$ implements our phased curriculum strategy. Three mechanisms suppress theoretical $O(t^2)$ scaling to observed 2.3× average per-iteration growth: 1) Error threshold filtering (4) removes 40% of low-difficulty samples, 2) Weighted curriculum sampling reduces effective batch size by 38%, and 3) Accuracy saturation limits error response generation through $(1-\eta)$ decay (0.55 $\rightarrow$ 0.46 $\rightarrow$ 0.38).

The framework maintains practical tractability through exponential complexity bounding:

$$\mathcal{T}_{\text{DPO}}^{(t)} \leq 2.3^t \mathcal{T}_{\text{DPO}}^{(0)}, \quad \lim_{t\to\infty} \mathcal{T}_{\text{DPO}}^{(t)} = O(1) \quad (10)$$

with complete convergence achieved in 3 iterations at 62% accuracy. Total wall-clock time ranges from 35 minutes to 3.2 hours on NVIDIA A800 GPUs, with DPO training utilizing 1,449-6,792 filtered preference pairs per iteration as detailed in Table 4.

4.5 Case Study

This case study illustrates our framework's advantage in reasoning about implicit, hierarchical relationships in NOTAMs, where high-level restrictions affect unmentioned components.

Consider a NOTAM for airport AGGC:

```
E) CHOISEUL L BAY AIRPORT CLOSED TO ALL
OPERATIONS...
```

Correctly interpreting this airport-wide closure means inferring effects on associated, unlisted components like runways.

Typical baseline systems, lacking structured knowledge (e.g., airport-runway relationships) or advanced reasoning, often fail this inference. They might parse the airport closure but omit the runway, providing incomplete awareness:

```
{"airport": "AGGC", "runway": "", ...}
```

Our framework addresses this challenge. KG-TableRAG queries the aviation knowledge graph with the airport identifier ('AGGC'), retrieving that `"RWY 07R"` belongs to airport `"AGGC"`. This fact supplies the missing structural context.

The LLM then integrates the input instruction ("airport closed") with this retrieved fact. Through semantic reasoning, it correctly infers the operational consequence - the runway must also be closed because it is part of the closed airport, leading to the accurate output:

```
{"airport": "AGGC", "runway": "RWY 07R",
...}
```

This correct inference of runway `"RWY 07R"` is critical for operational safety (e.g., preventing routing to a closed runway). It highlights our approach's advantage: integrated knowledge and reasoning for comprehensive understanding beyond simple text extraction. For additional detailed examples, please refer Appendix D

5 Conclusion

We present a knowledge-guided framework that combines LLMs with aviation expertise to address key challenges in NOTAM parsing task. Our self-evolving architecture addresses semantic-factual contradictions through dynamic integration of infrastructure knowledge and operational constraints.

The framework achieves a 30.4% accuracy gain over its base model via iterative optimization on 10,000 NOTAMs, effectively bridging NLP capabilities with aviation requirements while maintaining terminology integrity. This research establishes a new paradigm for NOTAM analysis, with principles extensible to other high-precision domains requiring robust knowledge integration and adaptive learning.

The results underscore the transformative potential of LLM-driven solutions in enhancing airspace management automation, mitigating human error risks, and advancing real-time decision-making capabilities for global aviation systems.

6 Limitation

While our self-evolving framework improves iteratively, limitations exist. First, the computational cost per optimization iteration increases (as detailed in our Complexity Analysis), necessitating potentially significant training time to achieve optimal performance, akin to reinforcement learning paradigms. Second, the inherent complexity of NOTAMs, with intricate temporal-spatial dependencies and specialized terminology, makes creating perfectly accurate ground truth annotations challenging. This difficulty in capturing all nuances can subsequently limit the model's performance ceiling, even with constraint-aware methods. Future work could explore LLM-assisted annotation combined with expert validation to further refine training data quality.

7 Ethical Considerations

While this work demonstrates potential for automating NOTAM analysis, current accuracy levels do not meet the stringent safety requirements for direct, real-time deployment in aviation systems. Therefore, the system is intended solely as a decision-support tool for ground analysts. Crucially, all outputs, especially critical flight information, must undergo rigorous manual verification by qualified personnel before operational use or transmission to pilots, adhering to established aviation safety protocols.

Our study utilizes publicly available NOTAM data, and annotations are performed by domain experts, ensuring transparency and data integrity. Continuous refinement of the model is ongoing, but the technology must currently be treated as supplementary, augmenting rather than replacing essential expert judgment in this safety-critical domain.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, et al. 2023. Gpt-4 technical report.

Alexandre Arnold, Fares Ernez, Catherine Kobus, and Marion-Cécile Martin. 2022. Knowledge extraction from aeronautical messages (notams) with self-supervised language models for aircraft pilots. In *Proceedings of NAACL-HLT 2022: Industry Track Papers*, pages 188–196.

Marc Bravin, Sita Mazumder, Daniel Pfäffli, and Marc Pouly. 2020. Automated smartification of notices to airmen. In *Proceedings of the 7th Swiss Conference on Data Science (SDS)*, pages 51–52.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, et al. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Hanzhu Chen, Xu Shen, Qitan Lv, Jie Wang, Xiaoqi Ni, and Jieping Ye. 2024a. SAC-KG: Exploiting large language models as skilled automatic constructors for domain knowledge graph. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4345–4360, Bangkok, Thailand. Association for Computational Linguistics.

Si-An Chen, Lesly Miculicich, Julian Martin Eisenschlos, Zifeng Wang, Zilong Wang, Yanfei Chen, et al. 2024b. Tablerag: Million-token table understanding with language models. *ArXiv*, abs/2410.04739.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, et al. 2022. Palm: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Stephen S. B. Clarke, Patrick Maynard, Jacqueline A. Almache, Satvik G. Kumar, Swetha Rajkumar, Alexandra C. Kemp, and Raj Pai. 2021. Natural language processing analysis of notices to airmen for air traffic management optimization. In *Proceedings of the AIAA Aviation Forum 2021*, pages 1–26.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Jun-Mei Song, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

Michelle Dieter, Eric Sprenger, Otilia Pasnicu, Josefine Staudt, and Nils Ellenrieder. 2024. Virtual flight deck crew assistance utilizing artificial intelligence methods to interpret notams: a user acceptance study. *CEAS Aeronautical Journal*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. 2024. The llama 3 herd of models. *ArXiv*, abs/2407.21783.

Yuqi Fan, Yuejie Tan, Liwei Wu, Han Ye, and Zengwei Lyu. 2024. Global and local interattribute relationships-based graph convolutional network for flight trajectory prediction. *IEEE Trans. Aerosp. Electron. Syst.*, 60(3):2642–2657.

Dongyue Guo, Zheng Zhang, Bo Yang, Jianwei Zhang, Hongyu Yang, and Yi Lin. 2024. Integrating spoken instructions into flight trajectory prediction to optimize automation in air traffic control. *Nature Communications*, 15(1):9662.

Yixin Ji, Kaixin Wu, Juntao Li, Wei Chen, Mingjie Zhong, Xu Jia, and Min Zhang. 2024. Retrieval and reasoning on KGs: Integrate knowledge graphs into large language models for complex question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7598–7610, Miami, Florida, USA. Association for Computational Linguistics.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, et al. 2023. Mistral 7b. *ArXiv*, abs/2310.06825.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilic, Daniel Hesslow, et al. 2022. BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100.

Peng Li, Tianxiang Sun, Qiong Tang, Hang Yan, Yuanbin Wu, Xuanjing Huang, and Xipeng Qiu. 2023. CodeIE: Large code generation models are better few-shot information extractors. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15339–15353, Toronto, Canada. Association for Computational Linguistics.

Zixuan Li, Yutao Zeng, Yuxin Zuo, Weicheng Ren, Wenxuan Liu, Miao Su, Yucan Guo, Yantao Liu, et al. 2024. KnowCoder: Coding structured knowledge into LLMs for universal information extraction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8758–8779, Bangkok, Thailand. Association for Computational Linguistics.

Xiaochen Liu, Bowei Zou, and AiTi Aw. 2024. An nlp-focused pilot training agent for safe and efficient aviation communication. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track, NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 89–96. Association for Computational Linguistics.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. *ArXiv*, abs/2203.12277.

Kaiwei Luo and Jiliu Zhou. 2025. Large language models for single-step and multi-step flight trajectory prediction. *CoRR*, abs/2501.17459.

Xihaier Luo, Xiaoning Qian, and Byung-Jun Yoon. 2024. Hierarchical neural operator transformer with learnable frequency-aware loss prior for arbitrary-scale super-resolution. *ArXiv*, abs/2405.12202.

Baigang Mi, Yi Fan, and Yu Sun. 2022. Notam text analysis and classification based on attention mechanism. *Journal of Physics: Conference Series*, 2171(1):012042.

Anna Mogillo-Dettwiler. 2024. Filtering and sorting of notices to air missions (notams).

Miruna Maria Morarasu and Cătălin Horațiu Roman. 2024. Ai-driven optimization of operational notam management. In *2024 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, pages 1–6. IEEE.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, et al. 2023. Orca: Progressive learning from complex explanation traces of GPT-4. *CoRR*, abs/2306.02707.

Aziida Nanyonga, Hassan Wasswa, and Graham Wild. 2023. Aviation safety enhancement via nlp & deep learning: Classifying flight phases in atsb safety reports. In *2023 Global Conference on Information Technologies and Communications (GCITC)*, pages 1–5. IEEE.

Lei Pan, Wuyang Luan, Yuan Zheng, Junhui Li, Linwei Tao, and Chang Xu. 2024. Graphfusion: Integrating multi-level semantic information with graph computing for enhanced 3d instance segmentation. *Neurocomputing*, 602:128287.

Krunal Kishor Patel, Guy Desaulniers, Andrea Lodi, and Freddy Lecue. 2023. Explainable prediction of qcodes for notams using column generation. *Preprint*, arXiv:2208.04955.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *ArXiv*, abs/2112.11446.

Oscar Sainz, Iker García-Ferrero, Rodrigo Agerri, Oier Lopez de Lacalle, German Rigau, and Eneko Agirre. 2024. GoLLIE: Annotation guidelines improve zero-shot information-extraction. In *The Twelfth International Conference on Learning Representations*.

Juanming Shi, Qinglang Guo, Yong Liao, Yuxing Wang, Shijia Chen, and Shenglin Liang. 2024. Legal-lm: Knowledge graph enhanced large language models for law consulting. In *Advanced Intelligent Computing Technology and Applications - 20th International Conference, ICIC 2024, Tianjin, China, August 5-8, 2024, Proceedings, Part IV (LNAI)*, volume 14878 of *Lecture Notes in Computer Science*, pages 175–186. Springer.

Haoran Sun, Lixin Liu, Junjie Li, Fengyu Wang, Baohua Dong, Ran Lin, and Ruohui Huang. 2024. Conifer: Improving complex constrained instruction-following ability of large language models. *ArXiv*, abs/2404.02823.

Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuansen Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, et al. 2023. InstructUIE: Multi-task instruction tuning for unified information extraction. *arXiv preprint arXiv:2304.08085*.

Fan Xu, Nan Wang, Xuezhi Wen, Meiqi Gao, Chaoqun Guo, and Xibin Zhao. 2023. Few-shot message-enhanced contrastive learning for graph anomaly detection. *2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 288–295.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. 2024. Qwen2.5 technical report. *ArXiv*, abs/2412.15115.

Yichi Zhang, Zhuo Chen, Lingbing Guo, Yajing Xu, Wen Zhang, and Huajun Chen. 2024. Making large language models perform better in knowledge graph completion. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 233–242.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *CoRR*, abs/2303.18223.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *ArXiv*, abs/1709.00103.

A Training Algorithm Implementation Details

Algorithm 1 Self-Evolving SFT & DPO Optimization

Require: Initial dataset $\mathcal{D}_0 = \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{test}}$ (8:2 split)

Base model π_{base}, empty response pool $\mathcal{R} = \emptyset$

Max iterations T, error threshold τ, temperature β, total epochs E

procedure MAIN
  $\pi_{\text{current}} \leftarrow \pi_{\text{base}}$ ▷ Model initialization
  for $t = 1$ **to** T **do**
    GENERATERESPONSES($\pi_{\text{current}}, \mathcal{D}_0$)
    UPDATERESPONSEPOOL($\mathcal{R}$) ▷ Record correct/incorrect responses
    $\mathcal{D}_{\text{SFT}} \leftarrow \{(x \circ K, Y^*) | Y^* \in \mathcal{Y}_x^*\}$
    $\pi_{\text{SFT}} \leftarrow$ SFT-TRAIN($\pi_{\text{current}}, \mathcal{D}_{\text{SFT}}$)
    GENERATERESPONSES($\pi_{\text{SFT}}, \mathcal{D}_0$)
    UPDATERESPONSEPOOL($\mathcal{R}$)
    $\mathcal{D}_{\text{pref}} \leftarrow$ BUILDPREFERENCEPAIRS($\mathcal{R}$)
    if $\mathcal{D}_{\text{pref}} \neq \emptyset$ **then**
      $\pi_{\text{DPO}} \leftarrow$ DPO-TRAIN($\pi_{\text{SFT}}, \mathcal{D}_{\text{pref}}$)
      $\pi_{\text{current}} \leftarrow \pi_{\text{DPO}}$
    end if
  end for
end procedure

function DPO-TRAIN($\pi_{\text{ref}}, \mathcal{D}_{\text{pref}}$)
  $\mathcal{D}_{\text{aug}} \leftarrow \emptyset$
  for $x \in \mathcal{D}_{\text{pref}}$ **do**
    if $\xi(x) \geq \tau$ **then** ▷ Data augmentation trigger
      $\mathcal{V}_x \leftarrow \bigcup_{n=1}^{N} \text{Augment}(x, n)$
      $\mathcal{D}_{\text{aug}} \leftarrow \mathcal{D}_{\text{aug}} \cup \{(v, Y^*, Y^-)\}$
    end if
  end for
  for $e = 1$ **to** E **do** ▷ Curriculum learning
    $\alpha_e \leftarrow \min(e/E, 1)$
    for $x_i \in \mathcal{D}_{\text{aug}}$ **do**
      $w_e(x_i) \leftarrow (1 - \alpha_e)\frac{1}{N} + \alpha_e \frac{\exp(\beta\xi(x_i))}{\sum_j \exp(\beta\xi(x_j))}$
    end for
    Sample batch $\sim P_e(x) \propto w_e(x)$
    Update π_θ using $\mathcal{L}_{\text{DPO}}$ (Eq. 6)
  end for
  return π_θ
end function

function BUILDPREFERENCEPAIRS($\mathcal{R}$)
  $\mathcal{D}_{\text{pref}} \leftarrow \emptyset$
  for $x \in \mathcal{D}_0$ **do**
    if $\exists (Y^*, Y^-) \in \mathcal{R}_x$ **then** ▷ Valid preference pairs exist
      $\mathcal{D}_{\text{pref}} \leftarrow \mathcal{D}_{\text{pref}} \cup \{(x, Y^*, Y^-)\}$
    end if
  end for
  return $\mathcal{D}_{\text{pref}}$
end function

B Knowledge Graph Structure

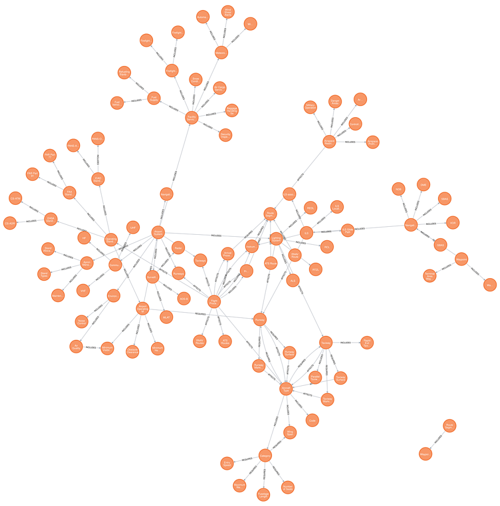

Figure 4: Domain Knowledge Graph Architecture.

C Task Prompt

```
You are an AI assistant specialized in parsing
    ↪NOTAMs. Your task is to extract information
    ↪about the runway status from the given NOTAM
    ↪ text. Please follow the guidelines below :

1. Identify Runway Status :
   Closed (MRLC, MRXX): Contains keywords like
       ↪CLOSED, CLSD, CLOSURE, NOT AVBL,
       ↪UNAVAILABLE, SUSPENDED, etc.
   - Limited (MRLT, MRXX): Contains phrases like
       ↪RESTRICTED, LIMITED, RESERVED FOR, etc.,
       ↪ and is combined with "only".
   - Open (MRAH): Contains keywords like OPEN, OPN
       ↪TO TFC, CANCELLED CLOSURE, etc.
2. Evaluate the Impact:
   - Determine if it affects takeoffs, landings, or
       ↪ both (based on the semantics).
   - Identify the affected flight types (
       ↪International, Domestic, Regional).
     - If the restricted flight type is not
         ↪explicitly mentioned, assign "
         ↪International, Domestic, Regional".
     - If explicitly mentioned, assign only the
         ↪restricted flight type.
3. Output Format:
   Please return the result in the JSON array
       ↪format. Each element represents a record
       ↪ and contains the following fields:
   - `airport`: ICAO code of the airport.
   - `runway`: Affected runway number.
   - `affect_actype`: Affected aircraft type. Fill
       ↪in the field only when it involves
       ↪wingspan, CODE C/D, or the number of
       ↪engines.
   - `affect_region`: The scope of closure or
       ↪restriction, with values "TAKEOFFS", "
       ↪LANDINGS", or "TAKEOFFS,LANDINGS".
   - `flight_type`: The affected flight type (
       ↪International, Domestic, Regional).
4. Notes:
   - Partial closure/restriction of a runway is
       ↪also considered as a complete closure/
       ↪restriction.
```

```
    - Create a separate record for each runway if
        ↪the message involves multiple runways.
    - Extract only the information explicitly
        ↪mentioned in the NOTAM; do not make
        ↪assumptions.
    - When facing the flight type, consider using
        ↪Chinese words to represent the flight
        ↪type as mentioned in the prompt and
        ↪examples.
    - Focus on the impact on aircraft, such as the
        ↪aircraft category (CODE C/D), wingspan (
        ↪in meters), etc.
    - Fill in the `affect actype` field based on the
        ↪ affected aircraft type. The wingspan
        ↪length may need to be converted from FT
        ↪to M for comparison.
    - The message may mention restrictions on
        ↪aircraft categories or CODEs, which need
        ↪ to be matched with the aircraft type
        ↪table.
```

D Example

—INPUT—

```
<instruction>:
Q)KZDV/QMRLC/IV/NBO/A/000/999/3952
    ↪N10440W005
A)KDEN B)2301010254 C)2301011200
E) DEN RWY 17L/35R CLSD

)
```

—OUTPUT—

```
<result>:
<think>
Runway Identification: Extracted the airport ICAO
    ↪code as "KDEN" from "A) KDEN".
Runway Status Analysis:
    "RWY 17L/35R CLSD" -> Both runways are
        ↪completely closed (CLSD=Closed).
    The default impact of closure is on takeoffs and
        ↪ landings -> affect_region: "TAKEOFFS,!
LANDINGS".
Aircraft Type Restriction:
    No mention of wingspan, CODE category, or engine
        ↪ restrictions -> affect_actype: null.
Flight Type Inference:
    No specific restriction mentioned -> Default
        ↪impact on all types -> flight_type: "
        ↪International,Domestic,Regional".
Handling Multiple Runways:
    Create a separate record for each runway
        ↪according to the rules (one each for 17L
        ↪ and 35R).

```json
[
  {
    "airport": "KDEN",
    "runway": "17L",
    "affect_actype": null,
    "affect_region": "TAKEOFFS,!
LANDINGS",
    "flight_type": "International,!
Domestic,Regional"
  },
  {
    "airport": "KDEN",
    "runway": "35R",
    "affect_actype": null,
    "affect_region": "TAKEOFFS,!
LANDINGS",
    "flight_type": "International,!
Domestic,Regional"
  }
]
```
```