Proceedings of the ASME 2026
International Design Engineering Technical Conferences and
Computers and Information in Engineering Conference
IDETC-CIE2026
August 23-26, 2026, Houston, TX

DETC2026-194314

DOMAIN KNOWLEDGE ACQUISITION IN AI-ASSISTED SYSTEMS ENGINEERING

Sara Sourani Yancheshmeh, Shih-Chun Deng, Yan Jin*

Department of Aerospace and Mechanical Engineering
University of Southern California
Los Angeles, CA

souraniy@usc.edu, dengshih@usc.edu, yjin@usc.edu

(*Corresponding Author)

ABSTRACT

Large language models (LLMs) face persistent challenges in specialized engineering environments, such as limited domain expertise, hallucinations, and an absence of explicit validation mechanisms. This paper presents a locally deployable, modular QA pipeline for regulatory engineering documents that addresses these issues by combining LoRA-based domain adaptation, lightweight graph-augmented retrieval, and post-hoc verification. To comply with industrial data privacy requirements, LLaMA 3 (8B) was locally deployed and adapted using Low-Rank Adaptation (LoRA). The system was evaluated using an LLM-assisted QA pipeline applied specifically to American Bureau of Shipping (ABS) regulatory documents. The results of the experiments indicate that the retrieval substantially improves the groundedness and the overall quality of the answers relative to generation alone. The integrated retrieval model significantly outperformed the standalone generator, achieving a mean score of 4.17/5.0 compared to 3.07/5.0 and improving groundedness scores from 2.68/5.0 to 4.14/5.0. Furthermore, the verifier provides an additional safeguard against unsupported outputs and demonstrates a high degree of precision in differentiating correct from incorrect answers. The verifier identified missing evidence (49.1%) and numerical inaccuracies (39.4%) as the primary error types. Overall, this study of applied architecture demonstrates that integrating these components yields a highly practical and effective domain-specific prototype for regulatory QA.

Keywords: Systems Engineering, Large Language Models, Retrieval-Augmented Generation, Domain Adaptation, Model Validation.

1 INTRODUCTION

Recent advances in large language models (LLMs) have demonstrated impressive general reasoning and text generation capabilities across a wide range of tasks. However, despite their scale and versatility, these models continue to exhibit critical limitations when deployed in structured reasoning environments. In particular, three persistent challenges remain: limited domain specialization, hallucination arising from reliance on parametric knowledge, and the absence of explicit mechanisms for validating generated outputs [1]. While increasing model scale has been shown to improve benchmark performance, scaling alone does not guarantee robustness, traceability, or computational efficiency in real-world applications. Smaller models offer a more deployable alternative but typically lack sufficient domain adaptation without targeted fine-tuning. At the same time, generative models conditioned solely on internal knowledge may produce outputs that are factually unsupported or misaligned with authoritative documents [2]. Retrieval-augmented generation (RAG) has been proposed as a mechanism to mitigate hallucination by incorporating external evidence at inference time; however, retrieval alone does not eliminate logical inconsistencies or prevent numerically incorrect responses. Furthermore, most generative pipelines lack an independent validation stage capable of detecting unsupported or unsafe outputs before deployment [3]. To address these intertwined challenges, this study investigates a modular architecture that integrates parameter-efficient fine-tuning, retrieval augmentation, and post-generation verification within a unified framework. Specifically, we evaluate whether domain-adapted smaller models can achieve competitive performance

relative to larger general-purpose models, whether retrieval improves grounding and factual alignment, and whether a dedicated verification component enhances overall system reliability. By systematically analyzing the contribution of each component, this work aims to demonstrate that combining specialization, retrieval, and validation results in a computationally efficient and operationally dependable reasoning system.

2 RELATED WORK

2.1 Model-Based Systems Engineering (MBSE) with SysML Modeling

The advancement of modern technical systems necessitates a shift toward Model-Based Systems Engineering (MBSE) to foster collaboration across diverse engineering domains. Although SysML has emerged as a standard for defining system structures and behaviors, effective linkage of these models with domain-specific tools such as CAD or thermodynamic simulations remains a significant challenge due to fragmented communication channels and inconsistent requirements. Recent research into SysML process chains has revealed that, while structural and behavioral aspects are reused across engineering tasks, the transformation of model information often relies on application-specific interfaces that lack a unified framework [4]. To address the discrepancy between unstructured design documentation and formal models, DocUDS was proposed as a solution. DocUDS is an automated system that extracts design knowledge, specifically requirements, functions, and solutions, from extensive reports. This extraction is performed using fine-tuned language models. This approach facilitates the alignment of sentences with particular design entities, thereby enhancing the comprehension of the design knowledge embedded within textual documents [5].

2.2 Domain Adaptation through Parameter-Efficient Fine-Tuning

While LLMs have been shown to exhibit remarkable general reasoning capabilities, achieving superior performance on specialized engineering tasks often requires fine-tuning on domain-specific datasets [6]. However, the high computational cost of full-parameter training has led to the development of efficient strategies such as Low-Rank Adaptation (LoRA) and Quantized LoRA (QLoRA). Research indicates that the combination of these parameter-efficient techniques with Distilling Step-by-Step (DSS) enables models to maintain high performance even under constrained GPU memory budgets, offering a practical workflow for industrial applications with limited resources [7]. Furthermore, exploration into Supervised Fine-Tuning (SFT) and model merging has revealed that integrating multiple specialized models can drive substantial advancements in technical capabilities, creating synergistic functionalities that individual parent models could not achieve alone [8]. These findings underscore the importance of strategic fine-tuning to enhance an LLM's in-domain generalization while maintaining its ability to adapt to diverse engineering tasks [6, 8].

2.3 Retrieval-Augmented Generation and Dataset Synthesis

A significant impediment to the implementation of LLMs in engineering is the paucity of high-quality, domain-specific labeled datasets. To address this, recent methodologies leverage generative models like GPT-4 to synthesize high-fidelity, ontology-based textual datasets. These datasets can achieve performance levels comparable to manually labeled benchmarks. The combination of synthetic data generation and rigorous evaluation metrics offers a scalable solution to the limitations imposed by dataset shortages in technical domains [9]. Despite the implementation of specialized training, generative models demonstrate a vulnerability to hallucinations. Consequently, Retrieval-Augmented Generation (RAG) has emerged as a method to ground model outputs in external evidence. RAG is a system that separates knowledge storage from language modeling. This approach ensures that generated answers are explicitly supported by authoritative document segments. Consequently, this improves factual alignment and reliability in structured reasoning environments [10].

3 MODULAR RETRIEVAL-AND-VERIFICATION AUGMENTED ARCHITECTURE

This paper presents a locally deployable, modular QA pipeline for regulatory engineering documents that combines LoRA-based domain adaptation, graph-augmented retrieval, and post-hoc verification. In the ABS document setting, retrieval substantially improves groundedness and overall answer quality relative to generation alone, while the verifier provides an additional safeguard against unsupported outputs. The Generator produces an initial structured answer conditioned on the user query. The retrieval module retrieves relevant document segments from a vector database and provides contextual grounding. The Verifier evaluates the generated answer against the retrieved evidence and assigns a structured verdict along with an error classification when applicable. The system operates through an orchestrated workflow. Given a user query q, the orchestrator first triggers the retrieval module to obtain the $top - K$ relevant document segments. The Generator then produces an answer conditioned on both the query and the retrieved context. The Verifier subsequently evaluates the pair and outputs a structured assessment consisting of a verdict (e.g., correct or incorrect) and, when necessary, an associated error type. Unlike confidence-threshold gating architectures, the system does not block or release answers based on a scalar confidence score. Instead, it separates generation, retrieval, and verification into independent modules, enabling structured analysis of factual alignment, grounding, and error categorization. This modular orchestration improves traceability and allows each component to be evaluated and refined independently.

3.1 Global vs Local LLM Deployment

Two model deployment paradigms were considered. Global models (e.g., GPT-class systems) offer strong general reasoning due to massive pretraining. However, they present limited transparency over training data, restricted fine-tuning control, dependency on external infrastructure, and data privacy concerns. These limitations are significant in regulatory and industrial contexts. Given collaboration with Japanese engineering companies, strict data privacy and compliance requirements necessitate local model deployment. Therefore, this study employs LLaMA 3 (8B parameters) as the base model. Benefits of local models include full control over weights and fine-tuning, private or in-house deployment compatible with enterprise policies, no transmission of sensitive regulatory or proprietary documents, and lower long-term operational cost. Their limitations are lower base capability compared to frontier global models and greater responsibility for optimization and evaluation.

3.2 Fine-Tuning Methodology

The generator model is initialized using LLaMA 3 (8B parameters). This model was selected due to compatibility with local deployment, balanced trade-off between performance and computational cost, and suitability for parameter-efficient adaptation. To adapt the base model to the regulatory domain, Low-Rank Adaptation (LoRA) was employed. Instead of updating all model parameters (~8 billion), LoRA introduces trainable low-rank matrices into selected transformer layers. The modified weight matrix is defined as:

$$W' = W + BA \tag{1}$$

Where W is the frozen pretrained weight matrix, $A \in R^{d\times r}$, $B \in R^{r\times d}$, $r \ll d$ is the adaptation rank, rank $r = 16$, only attention projection layers were adapted, total trainable parameters ≈ 15–20 million, original 8B parameters remained frozen. This reduces memory consumption and training time significantly while preserving base model knowledge. Fine-tuning was conducted using the Unsloth framework, which enables efficient LoRA integration, reduced GPU memory usage, faster training throughput, stable fine-tuning on consumer GPUs. Training data consists of structured regulatory question–answer (QA) pairs, where the input includes a user question and the associated context (regulatory clause), and the output is a structured compliance-oriented answer. The model is trained using standard cross-entropy loss:

$$L_{SFT} = -\sum_{t=1}^{T} log\ P_\theta(y_t \mid y_{<t}, x) \tag{2}$$

where x represents input tokens, y_t represents output tokens, and θ are trainable LoRA parameters. The model was trained using the AdamW optimizer with a learning rate of 1×10^{-5} and a batch size of 1. Gradient accumulation was set to 8, resulting in an effective batch size of 8 and the model was trained for 2-3 epochs with early stopping based on validation Exact Match performance. Validation performance was monitored after each epoch to select the best checkpoint. The chosen setup provides domain specialization without full retraining, reduces the risk of overfitting, lowers computational requirements, and enables feasible industrial deployment while maintaining compatibility with strict data privacy constraints.

3.3 Verification Model

Even when domain adaptation and retrieval augmentation are applied, generative models remain probabilistic systems that may produce structurally plausible but incorrect outputs. Such errors may include numeric inconsistencies, clause misattribution, incomplete condition handling, or unsupported claims derived from partial retrieval context. In structured reasoning environments, these failure modes can undermine system reliability. To address this limitation, a post-generation verification model is introduced as an independent validation layer. The verifier operates as a supervised multi-class classifier trained to assess the consistency between the generated claim $\hat{y}$ and the retrieved evidence $d_{1:K}$. The verifier predicts:

$$y \in \{Correct, Incorrect, Uncertain\} \tag{3}$$

The training objective minimizes categorical cross-entropy:

$$L_{verifier} = -\sum_i\ log\ P_\phi(y_i \mid \hat{y}, d_{1:K}). \tag{4}$$

where ϕ denotes the verifier parameters. The inclusion of an "Uncertain" class allows the model to abstain when evidence is insufficient or ambiguous. This design prevents overconfident, incorrect classifications and supports safer deployment. Synthetic negative examples are systematically generated to expose the verifier to realistic model failure modes, including numeric perturbations, clause mismatches, unit inconsistencies, condition or scope alterations, and missing evidence cases. Additionally, real generator errors observed during validation are incorporated into the verifier training set to improve generalization to authentic failure patterns. At inference time, the verifier produces a confidence score $s \in [0,1]$. A release decision is made according to a threshold τ, defined previously. The threshold is calibrated using validation data to balance reliability and coverage. System reliability is quantified using the false-release rate:

$$False\ Release\ Rate = \frac{Incorrect\ answers\ released}{Total\ released\ answers} \tag{5}$$

By separating generation from validation, the architecture reduces the propagation of unsupported outputs and introduces an explicit control mechanism over answer release. This modular separation enhances robustness and provides an interpretable decision boundary within the pipeline.

3.4 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is incorporated to mitigate hallucination and improve factual grounding by conditioning generation on externally retrieved evidence rather

than relying solely on parametric knowledge. While fine-tuning enhances domain adaptation, it does not ensure that outputs are explicitly endorsed by authoritative documents. RAG addresses this limitation by separating knowledge storage from language modeling, enabling dynamic access to a structured document corpus at inference time. Given a user query q, the retrieval module first encodes the query into a dense vector representation:

$$q = f_{enc}(q) \tag{6}$$

where f_{enc} denotes the embedding function. Each document segment d_i in the corpus is pre-embedded into vector form d_i. Retrieval is performed using cosine similarity:

$$sim(q, d_i) = \frac{q \cdot d_i}{\|q\| \|d_i\|} \tag{7}$$

The top-K segments are selected:

$$D_K = TopK_{d_i} sim(q, d_i) \tag{8}$$

These retrieved segments are concatenated with the query and passed to the generator model:

$$y_{cand} \sim p_\theta(y \mid q, D_K) \tag{9}$$

This mechanism enhances factual alignment by constraining the generator's context to pertinent regulatory clauses. However, conventional vector-based retrieval primarily operates on local semantic similarity and may fail in scenarios requiring multi-hop reasoning, entity disambiguation, or cross-document synthesis. The retrieved segments may possess individual relevance; however, when considered collectively, they may be incomplete or logically inconsistent. To address this limitation, Graph-based Retrieval-Augmented Generation (GraphRAG) is adopted. In this work, GraphRAG is a lightweight, graph-augmented retrieval method. Paragraph chunks act as nodes, and automatically extracted entities are attached to these chunks. Chunk relations are induced by document co-occurrence. First, seed chunks are retrieved using TF-IDF cosine similarity. Then, one-hop neighboring chunks are appended as supporting context. Instead of treating document segments as independent vectors, the corpus undergoes transformation into a knowledge graph.

$$G = (E, R, C) \tag{10}$$

where E denotes entities, R denotes typed relations between entities, and C represents higher-level community clusters. Entity extraction and relation linking are performed offline during corpus preprocessing. At inference time, the query is mapped to seed entities:

$$E_{seed} = EntityLink(q, G) \tag{11}$$

A 1-hop graph expansion retrieves related entities and relations:

$$N_k = kHop(E_{seed}) \tag{12}$$

Associated text spans corresponding to these graph nodes are aggregated to form structured evidence:

$$E_q = AssembleText(N_k) \tag{13}$$

This approach broadens retrieval beyond isolated similarity scores by capturing neighborhood context. Rather than formal multi-document reasoning chains, broader contextual relationships are preserved through implicit co-occurrence links, which improves coherence in structured regulatory queries compared to standard dense retrieval. The candidate answer produced by the generator is therefore conditioned on this graph-augmented evidence:

$$y_{cand} \sim p_\theta(q, E_q) \tag{14}$$

The output and its supporting evidence are subsequently forwarded to the verification model described in Section 3.3. This integration transforms retrieval grounding into a controlled decision process rather than a soft prompt constraint. Although this lightweight graph expansion introduces additional preprocessing and inference latency compared to standard vector search, it improves cross-document consistency and compatibility with the verification interface. Within the modular architecture, GraphRAG functions as the grounding layer, fine-tuning provides domain specialization, and verification enforces output reliability. The integrated system thus achieves a balance between computational efficiency and operational robustness under the constraints of industrial deployment.

3.5 Document Dataset Generation

The QA dataset was generated through a structured pipeline applied to ABS regulatory documents, as illustrated in Figure 1. First, the source documents were converted into text using an LLM-based extraction process. The extracted text was then segmented into smaller passages using a regex-based chunking method. In parallel, document-level summaries were produced using an LLM to capture higher-level context. Both the summaries and the text chunks were used as inputs for the QA generation stage, where an LLM generated grounded question–answer pairs based on the regulatory content. After generation, the QA pairs were embedded and tagged using a semantic encoder to obtain topic-level annotations. These annotations enabled visualization of the dataset distribution and supported uniform sampling across regulatory domains, ensuring balanced coverage of the rule space. The resulting QA pairs were filtered to retain only high-quality samples grounded in the source documents. As summarized in Table 1 in the Results section, a total of 5,482 QA pairs were initially generated, of which 2,337 pairs passed the filtering criteria and were retained for training the generator model. This dataset was divided into 70% training, 15% validation, and 15% testing subsets. A separate synthetic

dataset containing 4,326 samples was constructed to train the verifier model using controlled perturbations of ground-truth examples. This dataset followed the same 70/15/15 split. For final system evaluation, the test subsets of the generator and verifier datasets were combined to form an evaluation pool of 1,000 instances, from which 200 samples were manually reviewed by two human evaluators and the remaining 800 samples were evaluated using an LLM-based judge. All dataset splits were performed at the question level to prevent overlap between training and evaluation data.

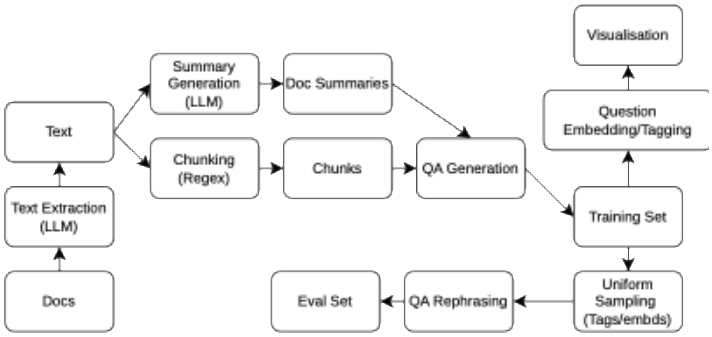

FIGURE 1: QA DATASET GENERATION AND TRAINING PIPELINE

In order to support accurate, maritime-specific generation, a localized GraphRAG database was constructed from American Bureau of Shipping (ABS) regulatory documents that had been converted into Markdown format. The pipeline employs a structure-aware chunking strategy that prioritizes natural paragraph boundaries while enforcing a strict 1,200-character limit. Monolithic sections are subdivided, and smaller contiguous segments are merged to preserve semantic context. This ensures that the retrieval mechanism maintains high granularity without losing surrounding meaning.

The database integrates chunk-level retrieval with a knowledge graph, wherein entity nodes are extracted via a frequency-based protocol, thereby filtering out domain-specific noise. The establishment of relationship edges is achieved implicitly through the co-occurrence of documents, facilitating the mapping of domain-specific ontologies without the necessity of manual annotation. A global TF-IDF vectorization approach is utilized for indexing, employing smoothed term frequencies and L2-normalization across a 12,000-term vocabulary. At inference, the system identifies seed chunks through the implementation of cosine similarity, followed by a 1-hop graph expansion. This hybrid structure facilitates the localized model's ability to address complex, multi-hop regulatory queries that might be overlooked by traditional dense retrieval methods.

4 RESULTS AND DISCUSSION

4.1 Dataset Evaluation

4.1.1 Generator Dataset Evaluation

The dataset evaluation process involved a multi-stage filtering pipeline designed to rigorously assess the quality of question-answer (QA) pairs. The total dataset consisted of 5,482 records. The pipeline was initiated with the utilization of vector-based RAG, wherein source documents were segmented and indexed through the application of OpenAI embeddings and FAISS. For each question-answer pair, relevant context chunks were retrieved based on the user's question, thereby establishing the ground truth necessary for factual verification. Subsequent to retrieval, the basic check function implemented a series of heuristic filters to identify evident quality issues without incurring substantial computational expenses. These checks included structural validation, which entailed detecting empty or improperly sized content; numeric verification, the process of ensuring that numbers in the answer appeared in the source text; and citation consistency, which involved verifying the existence of referenced tables or clauses in the retrieved context. The software also calculated keyword overlap. Records that did not pass these checks were identified and flagged at an early stage. Subsequently, a Large Language Model (GPT-5-mini) is employed to act as semantic judges. These models evaluated the remaining pairs for correctness, faithfulness to the source text, and answer relevance. Each pair was scored and provided with a final "accept," "repaired," or "reject" recommendation. The final results indicate that of the total records, 2,337 (42.6%) passed the rigorous evaluation criteria, while 3,145 failed. The pass rates exhibited variability across the sub-datasets. This stringent filtration process guarantees that the final dataset utilized for training or fine-tuning comprises exclusively high-quality, verifiable QA pairs that are anchored in the source documentation.

4.1.2 Synthetic Dataset Generation for Verifier Model

To robustly train and evaluate the verifier model, a synthetic dataset was constructed by using GPT-4o to perturb ground-truth samples, creating a realistic distribution of factual errors and knowledge gaps. The dataset follows a target distribution of 40% Correct, 40% Incorrect, and 20% Uncertain examples. The "Correct" subset comprises 400 verified, factual claims accompanied by their original supporting evidence, thereby establishing a baseline for the system.

The "incorrect" examples are generated through four distinct perturbation strategies: numeric perturbation (adjusting values by ±20%), unit swaps (e.g., substituting bar for psi), scope modifications, and a logic-based missing evidence approach where claims are paired with disjoint evidence. These adversarial samples are designed to expose the verifier to specific failure modes, such as numeric inconsistencies and clause misattributions. This systematic exposure guarantees that the verifier can identify unsupported claims prior to their inclusion in the final dataset.

The "uncertain" category is characterized by the deliberate degradation of supporting evidence, which involves the intentional removal of quantitative data and definitive statements. Vague terminology such as "approximate" or "standard" is employed. This model simulates real-world scenarios in which reference material is lacking, preventing the formulation of a definitive judgment. Through training on these

ambiguous cases, the verifier learns to abstain rather than produce overconfident, incorrect classifications, thereby enhancing the overall reliability of the deployment.

To ensure transparency in dataset construction, Table 1 summarizes the dataset composition, training–validation–test splits for both the generator and verifier models, and the evaluation pool used for the final pipeline assessment.

Dataset Component	Count	Split	Description
Raw QA Pairs	5482	-	QA pairs generated from ABS docs
QA Pairs Accepted After Evaluation	2337	-	Verified QAs used for generator training
Generator Training Set	1636	70%	LoRA fine-tuning of the generator model
Generator Validation Set	351	15%	Used for hyperparameter tuning
Generator Test Set	351	15%	Used for generator evaluation
Verifier Synthetic Set	4326	-	Synthetic dataset used for verifier training
Verifier Training Set	3028	70%	Used for training the verifier classifier
Verifier Validation Set	649	15%	Used for threshold calibration
Verifier Test Set	649	15%	Used for verifier evaluation
Total Evaluation Instances	1000	-	Combination of generator test set (351) and verifier test set (649)
Human-Evaluated Subset	200	-	Manually reviewed QA instances
LLM-judged Subset	800	-	Automatically evaluated using calibrated LLM judge

TABLE I: DATASET CONSTRUCTION AND EVALUATION SUMMARY

4.2 Method Comparison Result

To better understand the contribution of each component in the proposed architecture, the evaluation was conducted using several system configurations that progressively add different modules of the pipeline. These configurations were designed to isolate the impact of domain adaptation, retrieval grounding, and post-generation verification. The evaluated configurations include: (1) a base local model, consisting of the original LLaMA 3 (8B) model without LoRA fine-tuning and without retrieval; (2) a fine-tuned model, where LLaMA 3 (8B) is adapted to the ABS regulatory domain using LoRA but without retrieval; (3) a RAG baseline, which combines the fine-tuned model with standard vector-based retrieval; (4) a GraphRAG model, which replaces the standard retrieval method with the proposed graph-based retrieval mechanism; and (5) the full pipeline, which integrates the fine-tuned model, GraphRAG retrieval, and the verifier module.

For evaluation, a total of 1,000 questions were processed through the system to produce generated answers, retrieved evidence, verification verdicts, and error classifications where applicable. From these 1,000 instances, 200 question–answer cases were selected for detailed human review. During this manual evaluation phase, the outputs of the Generator and RAG-based configurations were independently scored using four criteria: accuracy, groundedness, completeness, and hallucination, each measured on a 0–5 scale. Accuracy measures factual correctness relative to the provided evidence. Groundedness evaluates how well the answer can be traced to retrieved document segments. Completeness measures whether all required elements of the answer are addressed. Hallucination measures the degree to which unsupported information appears in the generated response. The Verifier model was evaluated separately using two criteria: verdict accuracy, which measures whether the correctness label assigned by the verifier is appropriate, and error-type accuracy, which measures whether the predicted error category correctly characterizes the issue in the generated answer. These manually reviewed scores served as the calibration reference for automated evaluation.

For the remaining 800 question–answer instances, an LLM-based automated judge was used to perform the same evaluation using a structured scoring prompt calibrated with the previously reviewed examples. The automated judge produced detailed sub-scores for each evaluation criterion and aggregated performance metrics for all evaluated configurations. This process resulted in scored outputs for the full set of 1,000 evaluation questions, enabling systematic comparison across the baseline models, retrieval configurations, and the full Generator–GraphRAG–Verifier pipeline. The evaluation pipeline used in this study is illustrated in Figure 2:

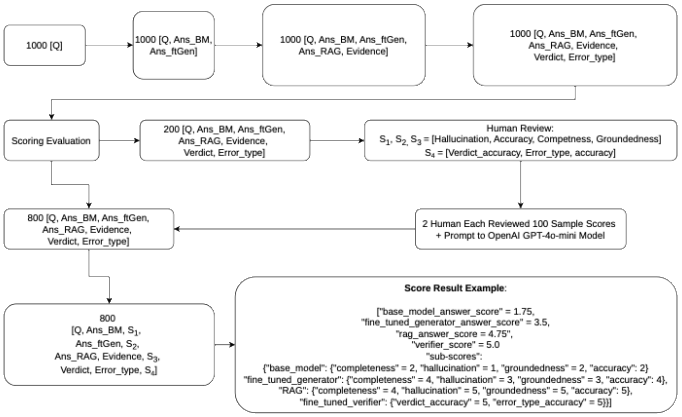

FIGURE 2: EVALUATION PIPELINE AND SCORING FRAMEWORK:

Figure 2 illustrates the evaluation pipeline for comparing the performance of a base Llama model (BM), a fine-tuned generator (ftGen), and a GraphRAG-based system using a multi-stage scoring process validated by human review and GPT-4o-mini.

The scoring prompt for the 800 examples included the previously reviewed 200 examples as few-shot references so that the model could follow the same scoring pattern. So, rather than relying on a single aggregate quality score, this approach decomposes answer quality into interpretable criteria that reflect factual correctness, evidence traceability, completeness, and fabrication risk. Each criterion is independently scored on a bounded ordinal scale ranging from 0 to 5, where 0 represents extremely poor performance and 5 represents excellent performance. Final scores are computed from these sub-criteria using predefined weighting schemes. The Generator model produces answers without guaranteed access to structured retrieval constraints at generation time. The first criterion, accuracy, measures factual correctness relative to the supplied evidence. The second criterion, hallucination, measures the degree to which the answer introduces unsupported or fabricated information. Importantly, this scale is defined such that higher values indicate fewer hallucinations. The third criterion, groundedness, evaluates how well claims in the answer can be directly traced to the evidence. High groundedness indicates explicit alignment with specific evidence statements, thresholds, or terminology. The fourth criterion, completeness, assesses whether the answer covers all required elements implied by the question and the evidence. The Verifier model assesses whether an answer is correct and whether its error type classification is appropriate. Its evaluation, therefore, focuses on classification reliability rather than content generation quality. Two criteria are used: verdict accuracy and error type accuracy. Verdict accuracy measures whether the Verifier correctly labels an answer as correct or incorrect, given the evidence. Error type accuracy measures whether the identified error category appropriately characterizes the nature of the problem.

The proposed evaluation framework emphasizes factual correctness, evidence traceability, hallucination resistance, and completeness. By separating rubric-level evaluation from score computation and incorporating retrieval-aware adjustments, the framework provides a robust and interpretable method for comparing Base, Generator, RAG, and Verifier performance in evidence-grounded technical question-answering tasks. After receiving the judge model's JSON response, the script parses these values and computes the final scores using predefined weighting formulas. This design ensures that the evaluation process remains reproducible and consistent across runs because the aggregation of scores does not depend on the probabilistic behavior of the language model. By restricting the model to qualitative judgments and performing all operations locally, the system maintains strict control over how each criterion contributes to the final evaluation.

The deterministic scoring mechanism also allows the evaluation framework to apply task-specific weighting strategies tailored to different system components. For example, answers generated without retrieval context are evaluated with heavier emphasis on grounding and specificity, which penalizes unsupported or overly generic responses. In contrast, answers produced by the RAG system are scored using criteria that prioritize factual accuracy and alignment with the retrieved evidence to detect possible retrieval failures. If the evidence appears irrelevant, the weighting scheme adjusts to reduce penalties related to grounding, thereby preventing misleadingly low scores caused by incorrect retrieval rather than generation errors. This deterministic post-processing step ensures that evaluation results remain stable and aligned with the intended evaluation objectives of the RAG system. Based on Figure 3 and Table II, the base model achieves an average score of 2.70 (σ = 0.48), indicating relatively low answer quality and limited reliability. Fine-tuning alone improves performance to 3.29 (σ = 0.51), representing an absolute gain of 0.59 points and a relative improvement of about 22% over the base model. This suggests that fine-tuning helps moderately by improving answer structure and domain adaptation, though the improvement is still limited because the model lacks external knowledge grounding. In contrast, the RAG system achieves a substantially higher average score of 4.44 (σ = 0.68), which corresponds to a 35% relative improvement over the generator and roughly 64% over the base model, indicating that the graph-based retrieval component contributes the largest quality gain by providing relevant supporting information. The slightly larger standard deviation for RAG suggests higher variability—while many answers are significantly better, some cases still depend on retrieval quality.

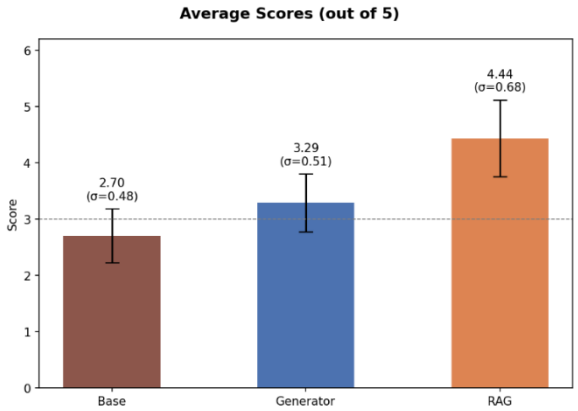

FIGURE 3: COMPARISON OF OVERALL AVERAGE WEIGHTED SCORES (OUT OF 5) BETWEEN THE BASE, GENERATOR AND RAG MODELS

Model	Average Score	Standard Deviation	Relative Improvement
Base LLaMA Model	2.70	0.48	-
Fine-tuned Generator	3.29	0.51	+22%
GraphRAG	4.44	0.68	+35%

TABLE II: AVERAGE SCORES (OUT OF 5) FOR BASE, GENERATOR, AND RAG WITH IMPROVEMENT METRICS

Relative improvement is used to show how much a method improves performance relative to the baseline, rather than only reporting the raw score difference. This helps normalize the comparison and makes it easier to understand the magnitude of improvement across different metrics, clearly illustrating how much fine-tuning and retrieval contribute to overall performance gains. It is calculated based on the following equation:

$$Relative\ Improvement = \frac{New\ Score - Baseline\ Score}{Baseline\ Score} \quad (15)$$

The sub-score breakdown further clarifies how each system improves different aspects of answer quality. Across all four metrics—completeness, hallucination control, groundedness, and accuracy—the RAG system consistently outperforms both the base model and the fine-tuned generator. For completeness, the base model scores 2.87 ($\sigma = 0.54$), which increases to 3.58 ($\sigma = 0.73$) after fine-tuning, representing an improvement of about 25%, while RAG reaches 4.25 ($\sigma = 0.90$), indicating that retrieval provides additional information that helps produce more complete responses. A similar pattern appears in hallucination reduction, where scores improve from 3.81 ($\sigma = 0.62$) for the base model to 4.31 ($\sigma = 0.66$) with fine-tuning, and further to 4.91 ($\sigma = 0.42$) with RAG, suggesting that external document grounding significantly reduces unsupported claims. The largest improvement occurs in groundedness, where the base model achieves 2.59 ($\sigma = 0.57$) and the generator improves modestly to 2.98 ($\sigma = 0.42$), but RAG jumps to 4.39 ($\sigma = 0.70$)—a relative improvement of roughly 69% over the base model—demonstrating the strong impact of retrieval in anchoring answers to relevant evidence. For accuracy, the base model scores 2.70 ($\sigma = 0.57$), the generator improves to 3.38 ($\sigma = 0.76$), and RAG reaches 4.20 ($\sigma = 0.97$), indicating that both fine-tuning and retrieval contribute to correctness, though retrieval again provides the largest gain. The standard deviations show that RAG sometimes exhibits greater variability (particularly in accuracy and completeness), which likely reflects dependence on retrieval quality, but its substantially higher means indicate consistently stronger performance overall shown in Table III. Together, these results confirm that fine-tuning alone yields moderate improvements, while the graph-based retrieval structure contributes the largest gains, especially in groundedness and hallucination reduction.

Metric	Base	Generator	Rag	Base → Generator	Base → RAG
Completeness	2.87±0.54	3.58±0.73	4.25±0.90	+25%	+48%
Hallucination	3.81±0.62	4.31±0.66	4.91±0.42	+13%	+29%
Groundedness	2.59±0.57	2.98±0.42	4.39±0.70	+15%	+69%
Accuracy	2.70±0.57	3.38±0.76	4.20±0.97	+25%	+56%

TABLE III: SUB-SCORE COMPARISON ACROSS BASE, GENERATOR, AND RAG MODELS WITH IMPROVEMENTS RELATIVE TO THE BASE MODEL

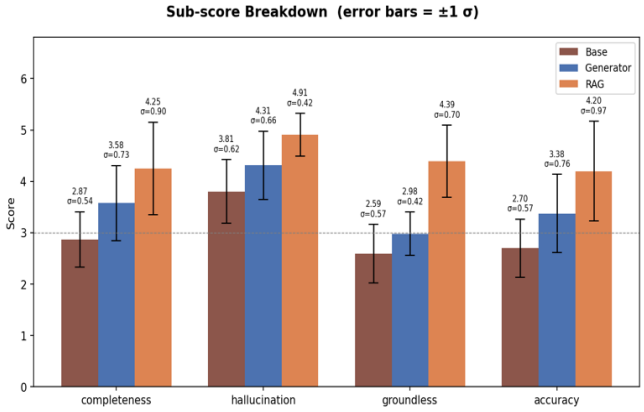

FIGURE 4: DETAILED SUB-SCORE BREAKDOWN OF COMPLETENESS, HALLUCINATION, GROUNDEDNESS, AND ACCURACY FOR THE BASE, GENERATOR AND RAG MODELS (ERROR BARS = ±1σ)

The error category analysis reveals that the majority of incorrect answers stem from missing evidence, which accounts for 207 cases (49.3%) of all errors. This indicates that nearly half of the failures occur when the system generates responses that are not sufficiently supported by retrieved or available documentation. Such errors suggest limitations in retrieval coverage or in the model's ability to incorporate relevant evidence into its responses. The second largest category is numeric errors, comprising 164 cases (39.0%). These errors arise when the system misinterprets or incorrectly reports numerical values, thresholds, or quantities from the source material. Numeric reasoning remains a known weakness of many language models, particularly when precise values must be extracted or interpreted from technical documents.

In contrast, unit-related errors represent the smallest category, with 49 cases (11.7%), indicating that mistakes involving measurement units or unit conversions are comparatively less frequent. This suggests that once numerical values are identified, the system generally preserves the associated units correctly. Overall, the distribution shows that the primary challenge is evidence grounding rather than formatting or unit handling, as nearly half of the incorrect outputs lack adequate supporting evidence. Addressing this issue may require improvements in document retrieval quality, evidence ranking, or prompt strategies that encourage the model to explicitly reference and integrate supporting information from the knowledge base.

The verifier error analysis shows how effectively the verification component distinguishes between correct and incorrect answers. Out of all evaluated responses, the verifier correctly identifies 379 answers as correct (True Correct) and 391 answers as incorrect (True Incorrect), indicating that it performs well in recognizing both valid and erroneous outputs. Notably, the system produces zero false positives, meaning the verifier does not incorrectly flag correct answers as wrong. This is an important property for user-facing systems, as it avoids

unnecessarily rejecting valid responses and maintains trust in the system's feedback. The absence of false positives suggests that the verifier applies a conservative validation strategy when assessing answer correctness.

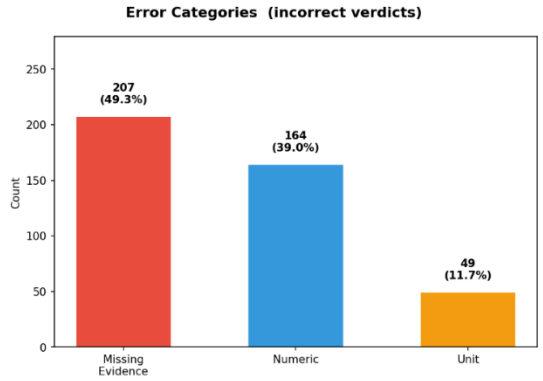

FIGURE 5: DISTRIBUTION OF ERROR CATEGORIES

However, the analysis also reveals 30 false negatives, where incorrect answers were mistakenly judged as correct by the verifier. Although this represents a relatively small portion of the total predictions, it indicates that some errors still pass through the verification stage undetected. Analysis of false negative cases suggests that these errors primarily occur when the retrieved evidence is incomplete or insufficient to clearly contradict the generated answer. In such cases, the verifier may incorrectly classify plausible but unsupported responses as correct. Additionally, subtle numerical inconsistencies and partial evidence alignment can further contribute to missed detections, indicating that both retrieval limitations and verifier sensitivity play a role in these failures. In practice, this means that while the verifier significantly improves reliability by catching most incorrect outputs, it is not perfect and may occasionally fail to identify subtle or complex mistakes. Overall, the results demonstrate that the verifier primarily enhances system reliability by accurately identifying incorrect responses without over-penalizing correct ones, though further improvements could focus on reducing the remaining false negatives to strengthen error detection.

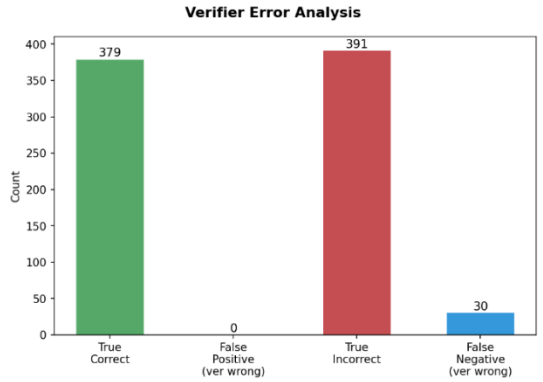

FIGURE 6: CLASSIFICATION OUTCOME ANALYSIS OF THE VERIFIER MODEL

The end-to-end inference latency per query to evaluate the practical performance of the proposed pipeline is also evaluated. The reported latency includes all components of the system, including retrieval, graph-based context expansion, answer generation using the fine-tuned LLaMA 3 model, and the verification module. After an initial warm-up run, the system achieves an average inference time of approximately 2.5 seconds per query, measured over a subset of evaluation queries. While the inclusion of GraphRAG and the verification stage introduces additional computational overhead compared to a generation-only baseline, these components significantly improve answer groundedness and reliability, reflecting a trade-off between latency and output quality.

5 CONCLUSIONS

This paper presented a locally deployable question-answering pipeline for regulatory engineering documents that combines domain-adapted language modeling, graph-augmented retrieval, and post-hoc verification. Motivated by privacy, compliance, and traceability requirements in engineering settings, the proposed framework was developed around ABS regulatory documents and evaluated as a practical alternative to generation-only systems. The results show that retrieval-augmented reasoning substantially improves answer quality and groundedness relative to standalone generation, while verification adds an additional safeguard against unsupported responses. The work makes three main contributions:

- First, it introduces a modular and locally deployable architecture for engineering-domain QA that integrates LoRA-based adaptation of a local LLaMA 3 8B model with retrieval and verification, making the system more suitable for privacy-sensitive industrial use.
- Second, it defines and implements a lightweight GraphRAG strategy in which paragraph chunks are linked through entity and co-occurrence relationships, enabling evidence expansion beyond direct lexical retrieval and improving grounded answer generation.
- Third, it provides an empirical evaluation showing that graph-augmented retrieval yields substantial gains over generator-only baselines, together with an error analysis that identifies missing evidence and numerical reasoning mistakes as the dominant failure modes.

At the same time, this study has several limitations. The evaluation pipeline remains substantially LLM-assisted, including stages of data generation, filtering, and large-scale judging, which may introduce bias despite the added human review. In addition, the experiments are limited to ABS regulatory documents, so the approach's generalizability to broader systems engineering corpora, design documents, or other regulatory domains has not yet been established. Finally, while the proposed GraphRAG formulation is effective in this setting, it is intentionally lightweight and should not be interpreted as a comprehensive graph-reasoning framework.

Future work will focus on improving both the robustness and practical deployability of the proposed architecture. The first

plan is to extend the evaluation with a larger-scale human assessment to reduce reliance on LLM-based judges and better capture real-world performance. Second is to investigate the generalizability of the framework across additional engineering domains beyond ABS regulatory documents, including design reports and technical corpora. Third is to explore stronger retrieval baselines and more advanced multi-hop retrieval strategies to further improve evidence coverage and reduce missing-evidence errors, which were identified as a dominant failure mode. Finally, future work will include improving the document preprocessing pipeline, particularly the conversion of PDF-based regulatory documents into structured text formats for both QA dataset generation and GraphRAG indexing, and also expanding human-centered evaluation. Preliminary observations suggest that higher-quality document structuring significantly improves retrieval effectiveness and downstream answer grounding, making this an important direction for improving overall system performance.

6 ACKNOWLEDGEMENTS

This paper is based on the work supported in part by the industrial sponsors: BEMAC Corporation, ClassNK, and MTI Co., Ltd. The authors are grateful for their support and collaboration on this research. The authors acknowledge the Center for Advanced Research Computing (CARC) at the University of Southern California for providing computing resources that have contributed to the research results reported within this publication. URL: https://carc.usc.edu.

7 REFERENCES

[1] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y., Madotto, A., and Fung, P., 2023, “Survey of Hallucination in Natural Language Generation,”.

[2] Zhai, X., Xiao, T., Liu, C., Cao, Y., and Zhang, C., 2024, “Parameter-Efficient Fine-Tuning of Large Language Models”.

[3] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., and Riedel, S., 2020, “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” Advances in Neural Information Processing Systems.

[4] Schumacher, T., Müller, C.-K., and Inkermann, D., 2025, "SysML Process Chains in MBSE: Systematic Literature Review and Future Research Directions," Digital Engineering.

[5] Qiu, Y., and Jin, Y., 2023, "Document Understanding-Based Design Support: Application of Language Model for Design Knowledge Extraction," ASME Journal of Mechanical Design.

[6] Yang, H., et al., 2024, "Unveiling the Generalization Power of Fine-Tuned Large Language Models," Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

[7] Marsh, B., et al., 2026, "An Efficient Strategy for Fine-Tuning Large Language Models," Frontiers in Artificial Intelligence.

[8] Buehler, M., 2025, "Fine-Tuning Large Language Models for Domain Adaptation: Exploration of Training Strategies, Scaling, Model Merging and Synergistic Capabilities," npj Computational Materials.

[9] Qiu, Y., and Jin, Y., 2025, "A Method for Synthesizing Ontology-Based Textual Design Datasets: Evaluating the Potential of Large Language Model in Domain-Specific Dataset Generation," ASME Journal of Mechanical Design.

[10] Lewis, P., et al., 2020, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Advances in Neural Information Processing Systems (NeurIPS)