Exploring the Reasoning Depth of Small Language Models in Software Architecture: A Multidimensional Evaluation Framework Towards Software Engineering 2.0

Ha Vo[§], Nhut Tran[§], Khang Vo[§], Phat T. Tran-Truong[✉]
Faculty of Computer Science and Engineering
Ho Chi Minh City University of Technology (HCMUT), VNU-HCM
Ho Chi Minh City, Vietnam
{ha.voxuancs22, nhut.trannov25th, khang.vohuu, phatttt}@hcmut.edu.vn

Son Ha
The Business School
RMIT University
Ho Chi Minh City, Vietnam
ha.son@rmit.edu.vn

***Abstract*—In the era of "Software Engineering 2.0" (SE 2.0), where intelligent agents collaborate with human engineers, Generative AI is advancing beyond code generation into Software Architecture (SA). While Large Language Models (LLMs) demonstrate superior capabilities, computational costs and data privacy concerns drive interest in Small Language Models (SLMs) with fewer than 7 billion parameters. However, the reasoning limits of these resource-constrained models remain unexplored. This study benchmarks 10 state-of-the-art SLMs on Architectural Decision Records generation, introducing a multidimensional framework evaluating Technical Compliance and Semantic Diversity. Our empirical results reveal a significant reasoning gap: 7B-parameter models demonstrate robust zero-shot capabilities, while 1B-parameter models require Fine-Tuning for structural validity. Contrary to assumptions regarding context saturation, Few-Shot prompting serves as a highly effective calibration mechanism, enhancing architectural correctness across mid-sized models. Furthermore, high semantic diversity in off-the-shelf small models often correlates with hallucination rather than productive exploration. These findings establish a rigorous baseline for deploying sustainable, locally hosted architectural assistants.**

***Index Terms*—SE 2.0, SLM, Software Architecture, ADRs**

I. Introduction

The field of Software Engineering (SE) is witnessing a paradigm shift toward "Software Engineering 2.0", where software development transforms into collaboration between engineers and autonomous intelligent agents [1]. Recent advances in generative AI have enabled software developers to shift focus toward higher-level design activities: requirements analysis, system design, and Software Architecture (SA) [2], [3], while we can dedicate low-level tasks such as implementation, code comprehension, debugging, .etc to software agents [4]–[6]. SA serves as the backbone of every digital system, requiring abstract thinking, trade-off analysis between quality attributes, and long-term maintainability forecasting. However, this shift prompts a critical question: can generative AI be trusted to support—or even make—architectural decisions requiring balancing competing quality attributes and reasoning about complex trade-offs?

Large Language Models (LLMs) have demonstrated exceptional capabilities in generating Architecture Decision Records (ADRs). Dhar et al. showed that GPT-4 achieves noteworthy capabilities in generating architectural design decisions, while smaller models like Flan-T5-base achieve comparable results with few-shot prompts and fine-tuning [7]. However, LLMs come at significant cost. Their extensive computational resources lead to prohibitively high operational expenses and substantial carbon emissions, making the search for efficient solutions both a moral and economic imperative [8]. Additionally, architectural documents often contain business secrets and system vulnerabilities, causing organizations to prohibit sending such data to public cloud APIs.

Small Language Models (SLMs) [9], typically defined as having fewer than 7 billion parameters, have emerged as a promising solution. They enable on-premise deployment, ensuring data security while reducing latency and inference costs through techniques such as model pruning, knowledge distillation, and low-rank factorization [10]. Parameter-Efficient Fine-Tuning (PEFT) methods like LoRA make fine-tuning accessible even on consumer-grade hardware.

Despite SLMs' growing prominence, a notable gap exists in their systematic evaluation for SA tasks. Existing benchmarks like SLM-Bench [11] evaluate diverse NLP tasks, while code generation benchmarks like HumanEval [12], MBPP [13], and SWE-bench [14] primarily assess functional correctness rather than architectural reasoning. Critically, existing ADR generation studies [7] rely on textual similarity metrics (ROUGE, BLEU), which cannot detect architecturally incorrect but grammatically correct designs.

To address this gap, we systematically evaluate open-source SLMs in generating ADRs, making three significant contributions:

(1) SLM-ArchBench Framework: A comprehensive eval-

[§]These authors contributed equally.
[✉] Corresponding Author: Phat T. Tran-Truong (phatttt@hcmut.edu.vn)

uation framework moving beyond textual similarity to assess architectural reasoning, probing the parameter threshold where models transition from lexical mimicry to genuine architectural compliance.

(2) Domain Adaptation Strategy Comparison: Systematic comparison of In-Context Learning versus Parameter-Efficient Fine-Tuning to identify optimal deployment configurations for resource-constrained environments.

(3) Quality-Diversity Trade-off Analysis: Empirical boundaries distinguishing productive architectural exploration from stochastic hallucination in generated ADRs.

We evaluate 10 open-source SLMs across varying parameter sizes, providing a principled basis for deploying sustainable and reliable architectural assistants balancing resource efficiency with real-world applicability in SA.

II. Related Work

A. Language Models for SA

The application of language models to SA represents a natural progression from earlier successes in code generation. Dhar et al. [7] demonstrated that GPT-4 exhibits noteworthy capabilities in generating Architecture Decision Records (ADRs), while smaller models like Flan-T5-base achieve comparable results with few-shot prompts and fine-tuning. Contemporary research has explored design rationale mining [15], [16], knowledge graph construction [17], [18], and domain-specific evaluation resources such as the QuArch dataset [19]. Despite these advances, the systematic evaluation of SLMs for architectural reasoning remains underexplored, leaving practitioners without clear guidance on whether resource-efficient alternatives can adequately support architectural decision-making in privacy-sensitive or resource-constrained environments.

B. Benchmarks for Code and Reasoning Tasks

Established benchmarks such as HumanEval [12], MBPP [13], and SWE-bench [14] have become standard tools for evaluating code generation capabilities, yet predominantly assess functional correctness at the implementation level, neglecting higher-order architectural reasoning. Recent efforts like SLM-Bench [20] integrate environmental metrics alongside accuracy across diverse NLP tasks, while multi-agent frameworks [21], [22] have been proposed for iterative evaluation enhancement. However, these approaches remain largely general-purpose, lacking the domain-specific criteria necessary for assessing architectural competence, including anticipating trade-offs, ensuring documentation-implementation alignment, and analyzing change impact propagation [20].

C. Evaluation Metrics: Limitations and Advances

Traditional text similarity metrics such as ROUGE and BLEU exhibit fundamental limitations when applied to technical domains [23], [24] — a design decision may achieve high textual similarity while violating core architectural principles. Recognition of these limitations has driven the development of model-based metrics including BERTScore [25] and COMET [26], the LLM-as-a-Judge paradigm [27], [28], and multi-dimensional frameworks like Meta-rater [29] and AXCEL [30] that integrate multiple quality dimensions. These advances inform our proposed framework, which extends beyond textual similarity to assess semantic diversity, structural compliance, and impact prediction capabilities.

III. Study Design and Execution

This section documents the design and execution of our empirical study. We begin by defining the research goal (Section III-A) and formulating the specific research questions (Section III-B). Next, we detail the model selection criteria (Section III-C) and the data collection methodology (Section III-D). The study then proceeds to the experimental procedure (Section III-E), describing the execution pipeline and generation strategies, finally concluding with the definition of the multi-dimensional evaluation metrics (Section III-F). A visualization of the complete experimental workflow is provided in Figure 1.

A. Goal

The primary goal of this research is to develop and apply a novel evaluation methodology that probes the reasoning depth of SLMs in the domain of SA. While existing benchmarks focus predominantly on code generation or general NLP tasks with surface-level metrics, this study aims to establish a systematic framework—SLM-ArchBench—that quantifies architectural reasoning capabilities through semantically and structurally grounded evaluation dimensions. Specifically, this research seeks to:

- **Quantify the architectural reasoning capabilities** of state-of-the-art SLMs when faced with complex SA problems.
- **Analyze the trade-offs** between model size, reasoning performance providing actionable insights for practitioners seeking to deploy AI-assisted architectural tools in resource-constrained or privacy-sensitive environments.

We define "reasoning depth" through three fundamental aspects: the ability to explore a diverse solution space rather than producing superficial variations, strict compliance with established structural rules and design patterns, and accurate prediction of the ripple effects caused by architectural changes. This multidimensional definition ensures that evaluation reflects not merely linguistic fluency but genuine architectural understanding.

B. Research Questions

To achieve our goal of benchmarking efficient models for architectural decision support, we address the following four research questions (RQ):

RQ1 *How capable are off-the-shelf SLMs in generating semantically accurate and architecturally compliant design decisions?*

This question establishes the baseline capability of resource-constrained models to generate coherent ADRs without prior training. We specifically investigate whether these efficient models possess sufficient pre-trained knowledge to propose technically sound architectural patterns that comply with industry best practices, or if they merely replicate the terminology without grasping the underlying design rationale.

RQ2 *Does In-Context Learning improve the architectural compliance and semantic quality of SLMs?*

This question examines the impact of providing limited examples ($k = 2$) within the prompt context. We investigate whether the addition of valid ADR examples serves as an effective calibration mechanism to guide the models toward standard architectural patterns, thereby increasing their compliance scores, or if the increased context length introduces noise that degrades reasoning in constrained attention windows.

RQ3 *To what extent does domain-specific Fine-Tuning maximize architectural compliance in resource-constrained models?*

This question explores the efficacy of Parameter-Efficient Fine-Tuning (LoRA) as a domain adaptation strategy. We analyze whether fine-tuning is strictly necessary to correct the reasoning deficits and compliance gaps of the smallest models (1B parameters) and whether it offers significant advantages over prompting strategies for larger, more capable architectures.

RQ4 *To what extent do SLMs exhibit semantic diversity when proposing solutions to open-ended architectural requirements?*

This question analyzes the breadth of the solution space explored by the models. We seek to differentiate between "productive exploration"—where the model proposes varied but valid trade-offs—and "stochastic variance," where high diversity signals hallucination or lack of compliance. We further investigate how different adaptation strategies (In-Context Learning vs. Fine-Tuning) influence this balance between convergence on compliant solutions and creative exploration.

C. Model Selection

1) Selection Criteria and Rationale: Following established benchmarking practices [11], we define selection criteria ensuring comprehensive coverage while maintaining experimental feasibility. Our model selection adheres to four fundamental principles: (1) **Parameter Constraint:** Models must contain fewer than 7 billion parameters, aligning with the conventional SLM definition [9] and representing a practical boundary for consumer-grade hardware deployment; (2) **Open-Source Availability:** All models must be publicly accessible with permissive licenses to ensure reproducibility; (3) **Instruction-Tuned Variants:** We exclusively utilize instruction-tuned versions as architectural design tasks require understanding complex instructions rather than simple text completion, with instruction-tuned models demonstrating superior performance in reasoning tasks [31]; (4) **Model Lineage and Training Diversity:** We selected models from diverse providers (Meta, Microsoft, Google, Alibaba, Mistral) to enable comparative analysis of how different training methodologies influence architectural reasoning capabilities.

2) Selected Models: Based on these criteria, we selected 10 instruction-tuned SLMs representing diverse approaches to model design, training data composition, and architectural innovation.

Model Name	Provider	Parameters (B)	Context Length	Release Date
Llama-3.2-1B	Meta	1.2	131,072	Sept 2024
Llama-3.2-3B	Meta	3.2	131,072	Sept 2024
Phi-3-mini	Microsoft	3.8	4,096	Apr 2024
OLMo-2-1B	Allen Institute (Ai2)	1.0	4,096	Apr 2025
OLMo-2-7B	Allen Institute (Ai2)	7.0	4,096	Nov 2024
Qwen2.5-1.5B	Alibaba	1.54	32,768	Sept 2024
Qwen2.5-3B	Alibaba	3.09	32,768	Sept 2024
Gemma-3-1B	Google DeepMind	1.0	32,768	Mar 2025
SmolLM2-1.7B	Hugging Face	1.7	8,192	Nov 2024
Mistral-7B-v0.3	Mistral AI	7.0	32,768	May 2024

3) Instruction-Tuned Variants Justification: Our exclusive focus on instruction-tuned variants is motivated by three factors. First, generating ADRs fundamentally requires instruction-following with structured outputs adhering to specific formats—a capability where instruction-tuned models demonstrate superior performance [32]. Second, this reflects practical SE 2.0 deployment scenarios [7] where practitioners naturally use instruction-tuned models for architectural assistance. Third, instruction tuning enhances both format compliance and complex reasoning capabilities [33], essential for architectural decision-making involving multi-step trade-off analysis. This ensures our benchmark reflects actual performance software architects would experience in resource-constrained environments.

D. Data Collection

1) Dataset Selection Rationale: We adopt the dataset curated by Dhar et al. [7], specifically designed for evaluating LLMs on ADR generation, motivated by three considerations. First, this established ICSA 2024 benchmark ensures direct comparability with prior work. Second, unlike general NLP datasets, it comprises 95 domain-specific ADRs systematically collected from five GitHub repositories, created by practicing architects following standardized formats [7]. Third, current SLM benchmarks [11] lack evaluation scenarios reflecting real-world architectural decision-making.

Data Characteristics: The dataset comprises 95 Context-Decision pairs from archane-framework (17), winery (17), joelparkerhenderson (32), cardano (14), and island (15) [7]. Each ADR follows standard format: Context describes the problem and constraints; Decision articulates the solution and rationale.

2) Dataset Limitations and Scope: We acknowledge inherent limitations. The 95 samples are modest but reflect domain constraints—high-quality ADRs require expert knowledge, making large-scale collection infeasible [7]. ADRs from open-source projects [7] may not fully represent enterprise contexts. We converted Markdown to CSV for: (1) seamless integration with data loading libraries, and (2) removing formatting noise for smaller models. Following baseline methodology [7], we used 80% training and 20% validation splits, with validation data for zero/few-shot experiments and few-shot examples from training data to prevent leakage. All ADRs are from publicly accessible repositories with open-source licenses [7].

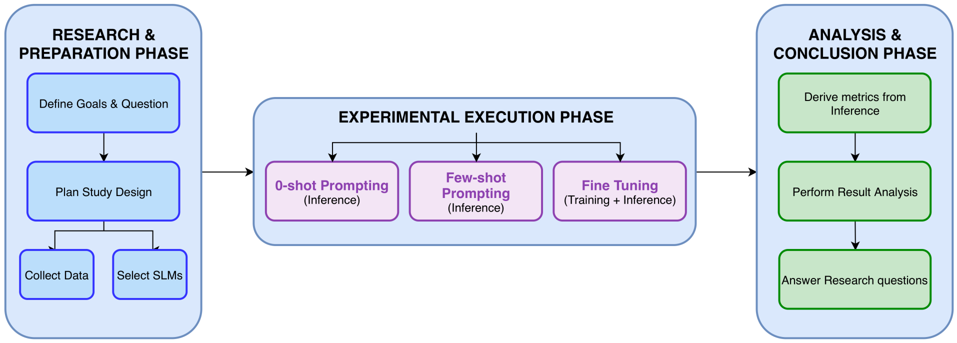

Fig. 1: Study Design

E. Experimental Procedure

To rigorously evaluate the architectural reasoning capabilities of SLMs, we executed a multi-stage experimental pipeline. All models were loaded using 4-bit Normal Float (NF4) quantization. The experiment was conducted in three distinct phases:

1) Zero-Shot: In this baseline setting, we evaluated the models' innate capability to generate ADRs without prior examples. The model was provided solely with the decision context and a specific instruction to generate the decision. The exact prompt template is illustrated in Figure 2.

> You are a SA assistant. Given the following architecture decision context, write a clear and complete Architecture Decision Record (ADR) decision.
> Context: {target_context}
> Decision:

Fig. 2: Zero-shot prompt

2) Few-Shot: To assess the impact of few-shot approach (in-context learning), we employed a few-shot prompting strategy with $k = 2$. We selected 2 examples from the training set according to the golden samples in [7] and added them to the target prompt. The structure of the few-shot prompt is presented in Figure 3.

3) Parameter-Efficient Fine-Tuning (PEFT): We specialized the SLMs for the ADR generation task using Low-Rank Adaptation (LoRA). We targeted all linear modules within the models using a rank $r = 16$, alpha $\alpha = 32$, and a dropout rate of 0.5. The models were fine-tuned for 10 epochs using the AdamW optimizer with a learning rate of $2e - 4$ and a linear warmup over the first 10% of steps. To accommodate memory constraints, we utilized a batch size of 2 with 4 gradient accumulation steps.

F. Metrics

To provide a holistic evaluation of the generated architectural decisions, we utilized a multi-dimensional metric suite

> You are a SA assistant. Given the following architecture decision context, write a clear and complete Architecture Decision Record (ADR) decision.
> Context: {example_context_1}
> Decision: {example_decision_1}
> Context: {example_context_2}
> Decision: {example_decision_2}
> Context: {target_context}
> Decision:

Fig. 3: Few-shot prompt

covering lexical overlap, semantic accuracy, structural validity, and solution diversity.

1) Standard Evaluation Metrics: We employed a comprehensive set of NLP metrics to measure the textual similarity between the generated decisions and the ground truth:

BERTScore (F1): Utilized as our primary measure of semantic accuracy, this metric evaluates the contextual embedding similarity between candidate and reference texts, capturing meaning beyond exact word matches.

ROUGE Family (1, 2, L): These recall-oriented metrics quantify the extent to which the generated text covers the content of the ground truth, measuring overlap at the unigram, bigram, and longest-common-subsequence levels.

BLEU: This metric assesses the precision of the generation, penalizing the inclusion of irrelevant or hallucinated tokens that do not appear in the reference text.

METEOR: To account for linguistic variability, this metric evaluates alignment by considering synonyms, stemming, and paraphrasing, providing a more flexible measure of correctness than strict exact-matching.

2) Architectural Compliance Score (LLM-as-a-Judge): Standard metrics often fail to capture technical correctness or adherence to specific architectural patterns. To address this, we implemented an automated Architectural Compliance Score using *Gemini-2.5-Flash* as an expert evaluator. The evaluator compares the model's output against the ground truth and assigns a scalar score $(0 - 100)$ based on the technical validity of the decision rationale and its alignment with architectural best practices. The exact prompt and scoring rubric are defined in 4.

> You are an expert SA evaluator.
> Given a model's generated architectural decision and the ground truth, rate how well the model's answer aligns with standard architectural patterns and the ground truth decision.
> Consider:
> - Correctness of architectural approach
> - Alignment with the ground truth rationale
> - Compliance with architectural best practices (e.g., MVC, microservices, layered architecture)
>
> Model Answer: {model_answer}
> Ground Truth: {ground_truth}
>
> Rate the compliance from 0 to 100:
> - 0-20: Completely wrong or irrelevant
> - 21-40: Partially correct but major issues
> - 41-60: Somewhat aligned but significant gaps
> - 61-80: Good alignment with minor differences
> - 81-100: Excellent alignment, equivalent or better
>
> Respond with ONLY a number between 0 and 100. No explanation.

Fig. 4: Automated Compliance Evaluator Prompt

3) Semantic Diversity Score: To quantify the breadth of the solution space explored by the models ($RQ4$), we moved beyond single-point estimates. For each test input, we generated 3 distinct candidate solutions using nucleus sampling.

We encoded these solutions into dense vector representations to calculate the Semantic Diversity Score, defined as the mean pairwise cosine distance between all candidate solutions for a given input. A higher score indicates that the model proposed architecturally distinct solutions, while a score near zero indicates mode collapse or repetitive generation.

IV. Results

In this section, we present the findings of our experimentation in line with the research questions that frame this study (Section III-B). Tables I, II, III summarize the metric scores obtained from the zero-shot, few-shot, and fine-tuning configurations, respectively, with the best score for each metric highlighted in bold. Our main evaluation measure, BERTScore (F1), is visualized in 5

A. Results RQ1: How capable are off-the-shelf SLMs in generating semantically accurate and architecturally valid design decisions

To address RQ1, we evaluated the zero-shot baseline of 10 SLMs. Our analysis reveals a distinct divergence between lexical capability and architectural reasoning, particularly in smaller models. The Mistral-7b-v0.3 model demonstrated the strongest zero-shot performance, achieving a BERTScore of 0.827 and an Compliance Score of 66.9%. This indicates that models in the 7B-parameter class possess sufficient pre-trained knowledge to autonomously propose valid architectural patterns. However, a significant discrepancy was observed in smaller architectures. While Gemma-3-1b achieved a competitive BERTScore of 0.805, its Compliance Score was the lowest in the cohort at 45.4%. This suggests that while low-parameter models can replicate the terminology of SA, resulting in high semantic overlap scores, they frequently fail to construct technically sound or coherent decision rationales without further guidance.

B. Results RQ2: Does In-Context Learning (Few-Shot Prompting) improve the architectural correctness and semantic quality of SLMs?

RQ2 investigates the efficacy of In-Context Learning (ICL) by providing two example ADRs in the prompt. The results demonstrate that ICL significantly enhances alignment with standard architectural patterns across the evaluated models. For instance, Phi-3-mini demonstrated a substantial increase in capability, achieving a semantic score of 0.831 alongside competitive quality metrics. The consistent improvement in Compliance Scores suggests that the provided examples function as a calibration mechanism. Rather than merely adjusting formatting, the examples enable the models to better distinguish between plausible-sounding but incorrect solutions and the rigorously correct architectural approaches preferred in the ground truth. This confirms that even resource-constrained models can effectively utilize their small context window to refine their architectural reasoning.

C. Results RQ3: To what extent does domain-specific Fine-Tuning maximize architectural correctness in resource-constrained models?

RQ3 examines the impact of Parameter-Efficient Fine-Tuning (LoRA) on model performance. The data indicates that fine-tuning is a critical mechanism for domain adaptation in the smallest model classes. Gemma-3-1b, which exhibited significant reasoning deficits in the zero-shot setting, achieved its peak semantic performance (0.828 F1) following fine-tuning. This confirms that the training process successfully encodes domain-specific architectural constraints into the model parameters, compensating for limited inference-time reasoning. In contrast, for more capable architectures such as Mistral-7b, fine-tuning yielded strong results (0.830 F1) but did not significantly surpass the performance achieved via few-shot prompting (0.835 F1). Furthermore, certain architectures like Olmo-2-1b exhibited performance degradation, with its Architectural Quality Score dropping to 46.6%. This highlights the susceptibility of certain architectures to overfitting during the adaptation process, suggesting that training is not always the optimal strategy for every model architecture.

model	rouge1	rouge2	rougeL	bleu	meteor	BERTScore			diversity	compliance
						precision	recall	f1		
gemma-3-1b	0.171	0.026	0.107	0.006	0.161	0.78	0.833	0.805	0.397	45.421
llama-3.2-1b	0.181	0.032	0.108	0.015	0.173	0.802	0.839	0.82	0.417	53.947
llama-3.2-3b	0.192	0.032	0.112	0.016	0.186	0.809	0.845	0.826	0.342	65.421
mistral-7b-v0.3	**0.202**	**0.042**	**0.117**	**0.018**	**0.189**	**0.813**	0.844	0.827	0.28	66.947
olmo-2-1b	0.18	0.029	0.102	0.017	0.177	0.809	0.843	0.825	0.298	60.474
olmo-2-7b	0.187	0.03	0.104	0.009	0.184	0.805	0.835	0.819	0.274	62.368
phi-3-mini	0.173	0.029	0.104	0.013	0.171	0.801	0.837	0.818	0.499	66.421
qwen2.5-1.5b	0.166	0.02	0.089	0.012	0.173	0.809	0.841	0.824	0.365	56.316
qwen2.5-3b	0.184	0.03	0.104	0.013	0.184	0.806	0.842	0.823	0.283	**71.737**
smollm2-1.7b	0.165	0.033	0.098	0.021	0.168	0.799	0.834	0.815	**0.541**	51.053

TABLE I: Zero-shot Result

model	rouge1	rouge2	rougeL	bleu	meteor	BERTScore			diversity	compliance
						precision	recall	f1		
gemma-3-1b	0.202	0.06	0.139	0.045	0.198	0.791	0.84	0.814	0.463	46.474
llama-3.2-1b	0.186	0.055	0.119	0.042	0.193	0.8	0.839	0.818	0.426	44.105
llama-3.2-3b	0.195	0.042	0.11	0.023	0.188	0.811	0.85	0.83	0.392	57.105
mistral-7b-v0.3	0.224	0.076	0.147	**0.055**	0.216	**0.823**	0.85	**0.835**	0.303	62.0
olmo-2-1b	0.2	0.068	0.131	0.042	0.207	0.81	0.844	0.826	0.36	56.211
olmo-2-7b	**0.229**	0.071	0.14	0.049	0.211	0.819	**0.853**	**0.835**	0.252	**73.632**
phi-3-mini	0.219	0.068	0.14	0.051	0.198	0.817	0.848	0.831	0.554	72.105
qwen2.5-1.5b	0.172	0.022	0.092	0.016	0.169	0.807	0.843	0.824	0.439	52.053
qwen2.5-3b	0.203	0.063	0.134	0.043	**0.221**	0.8	0.848	0.822	0.296	67.316
smollm2-1.7b	0.216	**0.081**	**0.148**	0.05	0.214	0.815	0.843	0.828	**0.642**	65.684

TABLE II: Few-shot Result

model	rouge1	rouge2	rougeL	bleu	meteor	BERTScore			diversity	compliance
						precision	recall	f1		
gemma-3-1b	**0.202**	**0.042**	**0.116**	0.022	**0.203**	0.812	0.846	0.828	0.382	47.895
llama-3.2-1b	0.174	0.028	0.101	0.012	0.164	0.804	0.838	0.82	0.437	53.842
llama-3.2-3b	0.197	0.04	0.113	0.02	0.193	0.808	0.845	0.825	0.404	**58.263**
mistral-7b-v0.3	0.2	0.035	0.109	**0.024**	0.198	0.813	0.85	**0.83**	0.377	54.211
olmo-2-1b	0.181	0.039	0.107	0.023	0.172	0.767	0.797	0.781	0.452	46.632
olmo-2-7b	0.195	0.034	0.11	0.016	0.178	**0.816**	0.846	**0.83**	0.355	57.368
phi-3-mini	0.193	0.04	**0.116**	**0.024**	0.194	0.813	0.845	0.828	**0.515**	47.789
qwen2.5-1.5b	0.179	0.029	0.097	0.017	0.182	0.802	0.847	0.823	0.427	46.0
qwen2.5-3b	0.194	0.036	0.106	0.016	0.196	0.807	0.844	0.825	0.429	56.053
smollm2-1.7b	0.185	0.037	0.11	0.022	0.197	0.81	0.846	0.827	0.437	50.842

TABLE III: Fine-tune Result

D. Results RQ4: To what extent do SLMs exhibit semantic diversity when proposing solutions to open-ended architectural requirements?

RQ4 analyzes the Semantic Diversity Score alongside the Architectural Quality Score to differentiate between productive exploration and stochastic variance. We observed an inverse correlation between output diversity and architectural quality in the zero-shot baselines. Smollm2-1.7b exhibited the highest diversity (0.541) but a mediocre Compliance Score (51.1%). This high variance implies a lack of convergent reasoning, where the model generates disparate solutions due to uncertainty rather than creative exploration. Conversely, higher-performing models like Mistral-7b demonstrated lower diversity (0.280) paired with higher quality (66.9%). This indicates that capable models tend to exhibit convergent reasoning, confidently identifying and adhering to the optimal architectural pattern rather than oscillating between multiple valid options. Finally, In-Context Learning facilitated a balanced expansion of the solution space; for models such as Olmo-2-1b, the inclusion of examples increased diversity to 0.360 while maintaining semantic relevance. This suggests that ICL empowers models to explore valid architectural variations without devolving into incoherence.

V. Discussion

This study provides evidence that Small Language Models (SLMs) can be effectively leveraged to generate Architectural Design Decisions, challenging the assumption that massive cloud-based models are a prerequisite for this task. Below, we address each research question by examining the outcomes of our experiments and drawing conclusions regarding the efficacy of using SLMs for ADR generation.

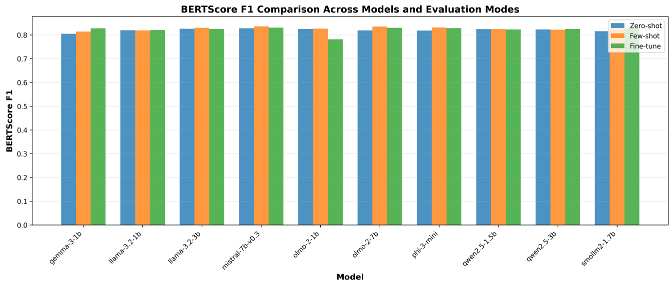

Fig. 5: BERTScore f1

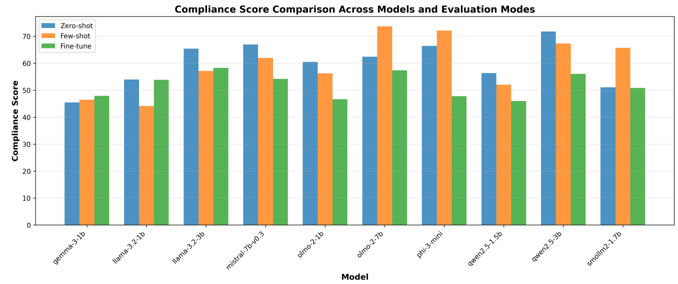

Fig. 6: Compliance Score

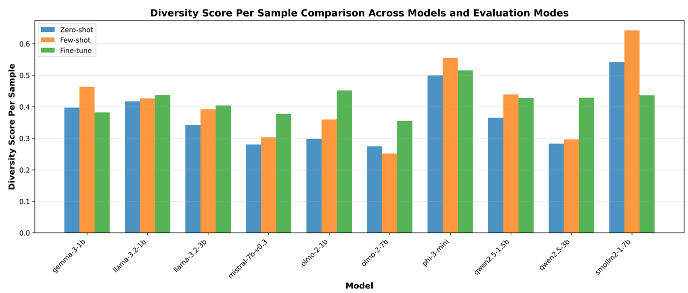

Fig. 7: Diversity Score

A. *How capable are off-the-shelf SLMs in generating semantically accurate and architecturally compliant design decisions?* (RQ_1)

Yes, but capability is strictly tied to model scale.

Our zero-shot experiments reveal that SLMs in the 7B-parameter class, such as Mistral-7b-v0.3, demonstrate a strong innate ability to propose valid architectural solutions, achieving high compliance scores without any task-specific training. This contradicts earlier assumptions that only massive proprietary Large Language Models can handle such reasoning tasks autonomously.

However, this capability is not universal. We identified a clear reasoning gap in the 1B-parameter class. Models like Gemma-3-1b and Smollm2-1.7b, while capable of generating text with high lexical overlap, frequently failed to align with architectural best practices, evidenced by their low compliance score. This suggests that for ultra-lightweight models, the pre-training data alone is insufficient to instill the complex decision logic required for SA.

B. *Does In-Context Learning (Few-Shot Prompting) improve the architectural compliance and semantic quality of SLMs?* (RQ_2)

Yes, In-Context Learning is a highly effective calibration mechanism. Providing just three examples enabled efficient models like Phi-3-mini to achieve performance parity with larger baselines, boosting its compliance score to 66.4This result indicates that the 1024-token context limit of these efficient models is not a bottleneck. Instead, the examples serve as a critical calibration signal, helping the model to adopt the correct tone and structural rigor of an ADR. For practitioners, this implies that few-shot prompting is the most cost-effective deployment strategy for mid-sized SLMs (3B–7B), as it yields state-of-the-art results without the computational overhead of fine-tuning.

C. *To what extent does domain-specific Fine-Tuning maximize architectural compliance in resource-constrained models?* (RQ_3)

Fine-tuning is essential for the smallest models but optional for larger ones. Our results highlight a nuanced trade-off. For models lacking innate architectural intuition like Gemma-3-1b, fine-tuning was the only intervention that successfully bridged the reasoning gap. This confirms that weight updates are necessary to internalize architectural patterns that 1B-parameter models cannot dynamically infer from a prompt. However, for capable architectures like Mistral-7b or Olmo-2-7b, fine-tuning yielded diminishing returns, producing results comparable to or slightly lower than the few-shot approach. Furthermore, the regression observed in Olmo-2-1b serves as a cautionary tale: over-optimizing small models on narrow datasets can degrade their general reasoning capabilities. Thus, fine-tuning should be reserved for ultra-low-resource scenarios where models under 3B parameters are required.

D. *To what extent do SLMs exhibit semantic diversity when proposing solutions to open-ended architectural requirements?* (RQ_4)

High diversity in small models often signals hallucination, not creativity. We observed a crucial distinction between ”productive exploration” and ”stochastic variance.” In zero-shot settings, the smallest models exhibited the highest diversity scores but the lowest compliance. This suggests that their

”diverse” outputs were largely incoherent or factually inconsistent. In contrast, capable models like Mistral-7b exhibited lower diversity paired with higher compliance, indicating convergent reasoning—the model consistently identified the optimal architectural pattern across multiple runs. However, we found that In-Context Learning could successfully balance this trade-off. For most models, few-shot prompting maintained or slightly increased diversity while significantly boosting compliance. This finding suggests that providing examples empowers the model to explore a valid range of architectural trade-offs without drifting into incoherence, fostering genuine decision support rather than random generation.

VI. Threats to Validity

In this section, we discuss the potential threats to the validity of our study and the mitigation strategies employed, following the guidelines by Wohlin et al [34].

Internal Validity: Prompt engineering bias may favor specific model families. We mitigated this through standardized, neutral prompt templates across all models, ensuring fair baseline comparison of instruction-following capabilities. Data leakage poses risks that ADRs appeared in pre-training corpora; we rigorously separated training/testing splits to prevent immediate leakage. The use of 4-bit NF4 quantization, while aligned with resource-constrained deployment goals, may introduce precision loss affecting observed performance ceilings.

External Validity: Our ADR dataset, while standard, represents a limited subset of SA domains and may not cover enterprise-scale distributed systems or legacy modernization scenarios. Model selection faces challenges from rapid AI development; we mitigated this by selecting 10 representative SLMs (1B-7B parameters) with diverse training methodologies to ensure broad coverage of the current SLM landscape.

Construct Validity: The LLM-as-a-Judge mechanism for Architectural Compliance Score may harbor biases favoring verbose or stylistically similar answers. We implemented a strict scoring rubric focusing on technical correctness rather than style, with strong correlation between Compliance Score and BERTScore suggesting consistent signals. High semantic diversity scores present interpretation challenges; we addressed this by analyzing diversity alongside compliance to distinguish productive exploration from incoherent hallucination.

Conclusion Validity: Generative model non-determinism may yield outlier results. We generated three distinct sequences per test input using Nucleus Sampling to ensure reliability and reduce random seed variation impact. For semantic metrics like BERTScore with compressed dynamic ranges, we interpreted values contextually based on recent literature confirming that small deltas correspond to significant qualitative performance shifts.

VII. Conclusion and Future Work

A. Conclusion

This study presents SLM-ArchBench, a multi-dimensional evaluation framework that systematically probes the architectural reasoning capabilities of SLMs in generating ARDs. Through empirical investigation of 10 state-of-the-art SLMs, we establish that a clear capability boundary exists around the 7B-parameter threshold, with models like Mistral-7b-v0.3 achieving compliance scores exceeding 66% in zero-shot settings while sub-3B models struggle to generate architecturally sound decisions autonomously. Few-Shot Learning emerges as a highly effective calibration mechanism, enabling mid-sized models (3B–7B) to achieve performance parity with larger baselines using merely two examples, as evidenced by Phi-3-mini’s compliance improvement from 66.4% to 72.1%. Parameter-Efficient Fine-Tuning proves essential only for ultra-lightweight models (1B), successfully bridging reasoning deficits but yielding diminishing returns or performance degradation in larger architectures. Our semantic diversity analysis reveals that high output variance in off-the-shelf small models often correlates with hallucination rather than productive exploration, while Few-Shot Learning successfully balances architectural correctness with valid solution space exploration. These findings establish actionable deployment guidelines: 7B models excel with prompting strategies, 3B–7B models benefit most from Few-Shot Learning, and 1B models require fine-tuning to achieve minimal viability, providing practitioners with evidence-based guidance for deploying sustainable, locally hosted architectural assistants in Software Engineering 2.0 environments.

B. Future Work

While this study establishes a rigorous baseline for evaluating SLMs in architectural reasoning, several directions warrant further investigation. Future work should extend the benchmark to encompass broader architectural tasks including system decomposition, quality attribute trade-off analysis, and architecture refactoring recommendations [20] to validate whether the identified parameter thresholds generalize beyond ADR generation. Incorporating multi-modal evaluation with visual architectural representations (UML, C4) and long-context scenarios would assess models’ ability to bridge abstract design and concrete implementations [17] [18]. Investigating interactive refinement protocols where architects provide corrective feedback would inform the design of effective human-in-the-loop architectural assistants for SE 2.0 environments [20]. Finally, integrating comprehensive energy consumption metrics alongside reasoning performance [8] and evaluating model robustness under ambiguous requirements would provide critical insights for responsible production deployment of sustainable architectural assistants.

By pursuing these research directions, the software engineering community can advance toward truly effective AI-assisted architectural design—systems that augment human expertise rather than replace it, respect resource and privacy constraints, and maintain the rigorous reasoning standards required for sustainable software evolution in the SE 2.0 era.

VIII. CODE AVAILABILITY

The source code developed for this study and results can be found at https://anonymous.4open.science/r/SLM-ArchBench

REFERENCES

[1] D. Lo, "Trustworthy and synergistic artificial intelligence for software engineering: Vision and roadmaps," in *IEEE/ACM International Conference on Software Engineering: Future of Software Engineering, ICSE-FoSE 2023, Melbourne, Australia, May 14-20, 2023*. IEEE, 2023, pp. 69–85. [Online]. Available: https://doi.org/10.1109/ICSE-FoSE59343.2023.00010

[2] V. Terragni *et al.*, "The future of ai-driven software engineering," *ACM Trans. Softw. Eng. Methodol.*, vol. 34, no. 5, pp. 120:1–120:20, 2025. [Online]. Available: https://doi.org/10.1145/3715003

[3] X. Hou *et al.*, "Large language models for software engineering: A systematic literature review," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 8, pp. 220:1–220:79, 2024.

[4] J. Yang, C. E. Jimenez, A. Wettig, K. Lieret, S. Yao, K. Narasimhan, and O. Press, "Swe-agent: Agent-computer interfaces enable automated software engineering," *Advances in Neural Information Processing Systems*, vol. 37, pp. 50 528–50 652, 2024.

[5] Y. Wei, C. S. Xia, and L. Zhang, "Copiloting the copilots: Fusing large language models with completion engines for automated program repair," in *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2023, pp. 172–184.

[6] C. Lee, C. S. Xia, L. Yang, J.-t. Huang, Z. Zhu, L. Zhang, and M. R. Lyu, "Unidebugger: Hierarchical multi-agent framework for unified software debugging," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025, pp. 18 248–18 277.

[7] R. Dhar, K. Vaidhyanathan, and V. Varma, "Can llms generate architectural design decisions? - an exploratory empirical study," in *21st IEEE International Conference on Software Architecture, ICSA 2024, Hyderabad, India, June 4-8, 2024*. IEEE, 2024, pp. 79–89. [Online]. Available: https://doi.org/10.1109/ICSA59870.2024.00016

[8] Schwartz *et al.*, "Green ai," *Commun. ACM*, vol. 63, no. 12, p. 54–63, Nov. 2020. [Online]. Available: https://doi.org/10.1145/3381831

[9] Z. Gao *et al.*, "Small pre-trained language models can be fine-tuned as large models via over-parameterization," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, A. Rogers, J. L. Boyd-Graber, and N. Okazaki, Eds. Association for Computational Linguistics, 2023, pp. 3819–3834. [Online]. Available: https://doi.org/10.18653/v1/2023.acl-long.212

[10] L. C. M. andand others, "Teaching small language models to reason," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, A. Rogers, J. L. Boyd-Graber, and N. Okazaki, Eds. Association for Computational Linguistics, 2023, pp. 1773–1781. [Online]. Available: https://doi.org/10.18653/v1/2023.acl-short.151

[11] N. T. Pham *et al.*, "Slm-bench: A comprehensive benchmark of small language models on environmental impacts-extended version," *CoRR*, vol. abs/2508.15478, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2508.15478

[12] M. Chen, J. Tworek *et al.*, "Evaluating large language models trained on code," *CoRR*, vol. abs/2107.03374, 2021. [Online]. Available: https://arxiv.org/abs/2107.03374

[13] J. Austin, A. Odena *et al.*, "Program synthesis with large language models," *CoRR*, vol. abs/2108.07732, 2021. [Online]. Available: https://arxiv.org/abs/2108.07732

[14] C. E. Jimenez *et al.*, "Swe-bench: Can language models resolve real-world github issues?" in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024.

[15] J. Zhao *et al.*, "Drminer: Extracting latent design rationale from jira issue logs," in *Proceedings - 2024 39th ACM/IEEE International Conference on Automated Software Engineering, ASE 2024*. ACM, 2024, pp. 468–480.

[16] M. Dhaouadi *et al.*, "Rationale dataset and analysis for the commit messages of the linux kernel out-of-memory killer," in *IEEE International Conference on Program Comprehension*. IEEE, 2024, pp. 415–425.

[17] S. Palombo *et al.*, "Generate with codexhug: A dataset to enhance model cards with code usage patterns," *Lecture Notes in Computer Science*, vol. 16083 LNCS, pp. 327–344, 2025.

[18] B. Chen *et al.*, "Autokg: Efficient automated knowledge graph generation for language models," in *Proceedings - 2023 IEEE International Conference on Big Data, BigData 2023*. IEEE, 2023, pp. 3117–3126.

[19] S. Prakash *et al.*, "Quarch: A question-answering dataset for ai agents in computer architecture," *IEEE Computer Architecture Letters*, vol. 24, no. 1, pp. 105–108, 2025.

[20] Z. Wan, Y. Zhang *et al.*, "Software architecture in practice: Challenges and opportunities," in *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2023, San Francisco, CA, USA, December 3-9, 2023*, S. Chandra, K. Blincoe, and P. Tonella, Eds. ACM, 2023, pp. 1457–1469.

[21] D. Li *et al.*, "Smoa: Improving multi-agent large language models with sparse mixture-of-agents," *Lecture Notes in Computer Science*, vol. 15872 LNAI, pp. 54–65, 2025.

[22] R. Li *et al.*, "Coevol: Constructing better responses for instruction finetuning through multi-agent cooperation," in *EMNLP 2024 - 2024 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*. Association for Computational Linguistics (ACL), 2024, pp. 4703–4721.

[23] B. Goodrich *et al.*, "Assessing the factual accuracy of generated text," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2019, pp. 166–175.

[24] M. Grusky, "Rogue scores," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, vol. 1. Association for Computational Linguistics (ACL), 2023, pp. 1914–1934.

[25] Q. Pan, "Human-centered design recommendations for llm-as-a-judge," in *HuCLLM 2024 - 1st Human-Centered Large Language Modeling Workshop, Proceedings of the Workshop*. Association for Computational Linguistics (ACL), 2024, pp. 16–29.

[26] C. Leiter, "Towards explainable evaluation metrics for machine translation," *Journal of Machine Learning Research*, vol. 25, 2024.

[27] S. A. Gebreegziabher, "Metricmate: An interactive tool for generating evaluation criteria for llm-as-a-judge workflow," in *CEUR Workshop Proceedings*, vol. 3957. CEUR-WS, 2025, pp. 266–278.

[28] R. Hu *et al.*, "Training an llm-as-a-judge model: Pipeline, insights, and practical lessons," in *WWW Companion 2025 - Companion Proceedings of the ACM Web Conference 2025*. ACM, 2025, pp. 228–237.

[29] X. Zhuang, "Meta-rater: A multi-dimensional data selection method for pre-training language models," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, vol. 1. Association for Computational Linguistics (ACL), 2025, pp. 10 856–10 896.

[30] P. A. Sreekar *et al.*, "Axcel: Automated explainable consistency evaluation using llms," in *EMNLP 2024 - 2024 Conference on Empirical Methods in Natural Language Processing, Findings of EMNLP 2024*. Association for Computational Linguistics (ACL), 2024, pp. 14 943–14 957.

[31] J. Wei, M. Bosma *et al.*, "Finetuned language models are zero-shot learners," in *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*, 2022.

[32] L. Ouyang *et al.*, "Training language models to follow instructions with human feedback," in *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., 2022.

[33] H. W. Chung *et al.*, "Scaling instruction-finetuned language models," *J. Mach. Learn. Res.*, vol. 25, pp. 70:1–70:53, 2024. [Online]. Available: https://jmlr.org/papers/v25/23-0870.html

[34] C. Wohlin *et al.*, *Experimentation in Software Engineering, Second Edition*. Springer, 2024.