Training-Free Analog Circuit Design: Unlocking Latent Reasoning Capabilities in Logic-Optimized LLMs

Haiyan Qin*, Gengfei Li*, and Wang Kang†
National Key Laboratory of Spintronics, Hangzhou International Innovation Institute
School of Integrated Circuit Science and Engineering, Beihang University, China
Email: haiyanq@buaa.edu.cn, ligengfei@buaa.edu.cn, wang.kang@buaa.edu.cn

***Abstract*—Domain-specific Large Language Models (LLMs) typically rely on computationally expensive fine-tuning to acquire specialized engineering knowledge. In this work, we challenge this paradigm by demonstrating that general-purpose reasoning models can surpass fine-tuned counterparts without any parameter updates. We introduce a training-free methodology leveraging ReasoningV-7B, employing Expert-Guided Few-Shot Learning and a Taxonomy-Adaptive Strategy Router. On the AMSBench benchmark, our inference-optimized framework outperforms the fine-tuned AnalogSeeker in 5 out of 6 tasks, achieving an overall accuracy of 86.66% (+7.65%), with 93.32% on Text QA (+8.32%) and 81.6% on LDO analysis (+35.6%). Notably, on Graduate-level TQA questions, we achieve 96.12% accuracy, approaching near-perfect performance. Critically, applying identical optimization strategies to AnalogSeeker yields no overall improvement (84.97% unchanged) and even degrades TQA performance by 2.38%, suggesting that supervised fine-tuning may impair in-context learning capabilities. Ablation studies reveal that few-shot learning contributes the largest gain (20-30%), while strategies combine additively. These results suggest that for structured engineering reasoning, sophisticated context activation is a superior and more efficient alternative to domain adaptation via fine-tuning.**

***Index Terms*—Analog Circuit Design, Large Language Models, Prompt Engineering, AMSBench, In-Context Learning**

I. Introduction

The automation of analog circuit design has long been a "holy grail" in the EDA industry. While digital design flows are highly automated, analog design remains heavily dependent on human intuition and expertise. Recently, Large Language Models (LLMs) have been explored as potential assistants for this task [1], [2].

Prior work, such as AnalogSeeker [3], posits that general LLMs lack the specific knowledge required for analog circuits and thus require extensive fine-tuning on domain-specific datasets. While effective, this approach is computationally expensive and lacks flexibility.

Validating this hypothesis, our work adopts a training-free paradigm. Crucially, we challenge the assumption that analog expertise requires domain-specific pre-training. We hypothesize that the reasoning structures required for hardware code generation (HDLs) are *isomorphic* to those needed for analog circuit analysis. Verilog synthesis demands strict causal reasoning and feedback loop analysis, which mirrors the topological dependencies in analog circuits (e.g., LDO stability analysis requires tracing signal propagation through error amplifiers and feedback networks).

Guided by this **Code-to-Analog Transfer** hypothesis, we leverage ReasoningV-7B—a model optimized for Verilog generation with *no explicit analog training*. **Surprisingly**, it exhibits remarkable aptitude for analog circuit reasoning. Recent studies suggest that code training instills a generalized "reasoning scaffold" into LLMs that transfers to structured tasks [4]. We demonstrate that activating this latent scaffold through sophisticated inference strategies is superior to both generic conversational priors and rigid fine-tuned weights. The bottleneck is not the lack of knowledge, but the lack of appropriate context activation [5].

We propose a purely inference-time optimization framework that leverages:

1) **Expert-Guided Few-Shot Learning**: Providing structured circuit analysis examples.
2) **Taxonomy-Adaptive Strategy Router**: Dynamically selecting prompting strategies based on semantic classification, ensuring optimal reasoning paths for different question types.
3) **Deterministic Execution**: Eliminating generation randomness for engineering rigor.

II. Related Work

A. LLMs for Hardware Design

Recent advances have demonstrated the potential of LLMs in hardware design. ChipNeMo [1] introduced domain-adapted LLMs for chip design through extensive fine-tuning on proprietary datasets. VeriGen [2] focused on Verilog code generation using large-scale pre-training. However, these approaches require significant computational resources and domain-specific training data.

B. Prompt Engineering

The emergence of prompt engineering techniques has shown that LLMs can be adapted to new tasks without parameter updates. Chain-of-thought prompting [6] demonstrated that

explicit reasoning steps improve performance on complex tasks. Zero-shot reasoning [7] showed that simple prompt modifications can unlock latent capabilities. Our work extends these ideas to the specialized domain of analog circuit design.

C. Analog Circuit Design Automation

Traditional approaches to analog design automation have relied on optimization algorithms [8]. AnalogSeeker [3] represents the first foundation model specifically trained for analog circuit tasks, achieving strong performance on the AMSBench benchmark through domain-specific fine-tuning.

III. Methodology

Figure 1 illustrates our overall methodology framework, which consists of three key components working in concert to achieve superior performance without fine-tuning.

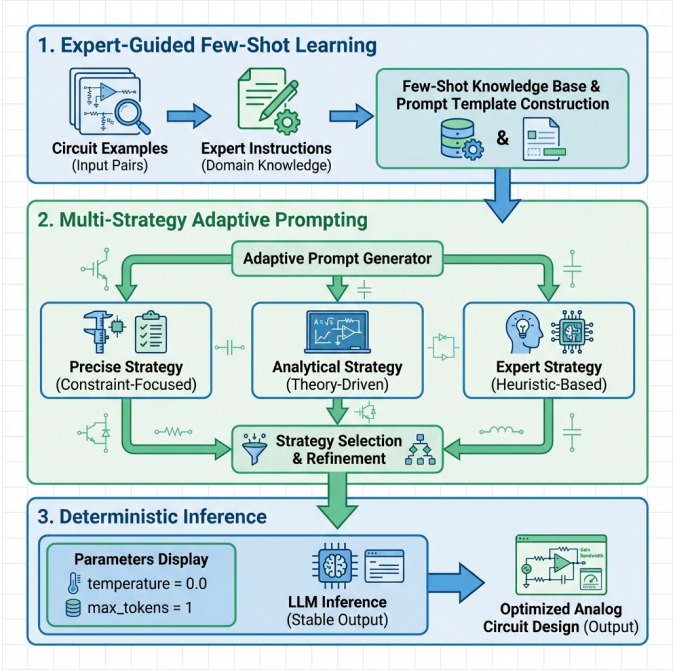

Fig. 1. Overview of our adaptive prompting methodology for analog circuit design.

A. Base Model: ReasoningV

We selected ReasoningV-7B as our base model not for its domain knowledge (which is digital), but for its **inference engine**. The model's pre-training on hardware description languages (HDLs) equips it with a strong prior for *structured dependency analysis*—understanding how component A affects component B given condition C. We hypothesize that this capability, rather than raw domain knowledge, is the true bottleneck in analog circuit reasoning. While SFT aligns models for conversational fluency, it often incurs an "alignment tax" that degrades pure reasoning capabilities [9]. Traditional analog LLMs acquire domain knowledge through fine-tuning but may lack the systematic reasoning structures that HDL training provides. By using a base reasoning model without aggressive domain-specific SFT, we avoid this degradation and retain the model's latent logical adaptability.

B. Adaptive Prompting Strategies

1) Expert-Guided Few-Shot Learning: For structural analysis tasks (LDO, Comparator, Caption), we observed that zero-shot performance was suboptimal (e.g., LDO at 46.0%). We introduced expert instructions combined with N-shot examples.

- **LDO Task**: $N = 3$. Instructions focus on pass transistors, error amplifiers, and feedback networks.
- **Comparator Task**: $N = 2$. Systematic analysis of input stages and output drivers.
- **Caption Task**: $N = 8$. Evaluating descriptions of circuit schematics.

The expert instruction for LDO, for example, guides the model to check: (1) Pass transistor with source fixed at VDD, (2) Error amplifier comparing VREF with feedback, (3) Stable bandgap reference, and (4) Resistive divider feedback network. For complex circuits like LDOs, outcome supervision alone is insufficient. We employ a step-by-step verification strategy [10] via expert checklists, forcing the model to validate intermediate circuit states before concluding.

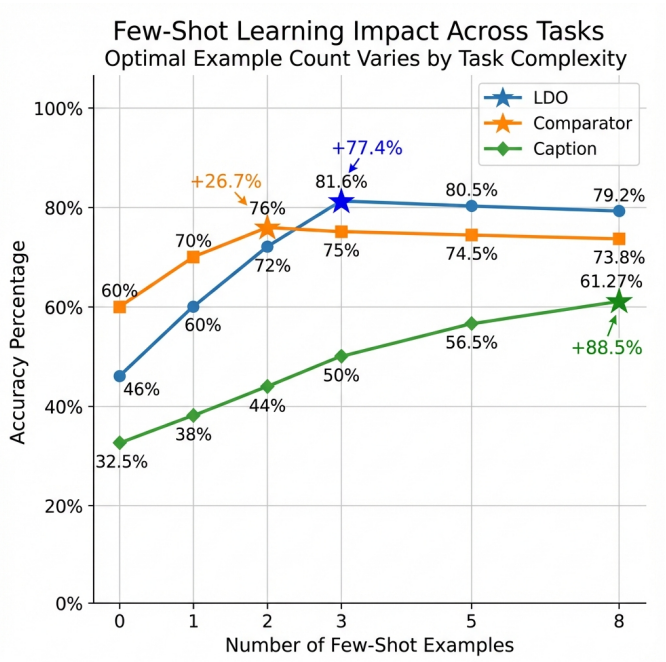

Fig. 2. Impact of Few-shot learning on accuracy across different tasks. The optimal number of examples varies by task complexity, with LDO benefiting significantly from 3 examples and Caption from 8 examples.

2) Multi-Strategy Optimization for TQA: To automate strategy selection during inference, we developed a **keyword-based semantic router** derived from validation set insights. The router analyzes question text using regular expression matching to compute matching scores for each question type. Questions are classified into five categories based on interrogative keywords and domain context:

- **Factual (Router → Precise):** Triggered by "What is", "What are", "Define", "Which of the following" (∼60% of questions).
- **Reasoning (Router → Analytical):** Triggered by "Why", "How does", "Explain", "Because", "Leads to" (∼15% of questions).
- **Calculation (Router → Calculate):** Triggered by "Calculate", "Compute", "Determine", "Find", "What is the value" (∼5% of questions).
- **Analysis (Router → Analytical):** Triggered by "Analyze", "Examine", "Evaluate", "Compare", "Difference" (∼10% of questions).
- **Comparison (Router → Compare):** Triggered by "Better", "Best", "Prefer", "Optimal", "Superior" (∼5% of questions).

The router computes a matching score for each category and selects the highest-scoring type. If all scores are zero, it defaults to the Factual category. This routing mechanism successfully applied specialized strategies to **104 out of 1,257 questions**, achieving a **61.7% error correction rate** on these targeted questions (fixing 116 out of 188 baseline errors). Figure 3 illustrates the strategy selection process.

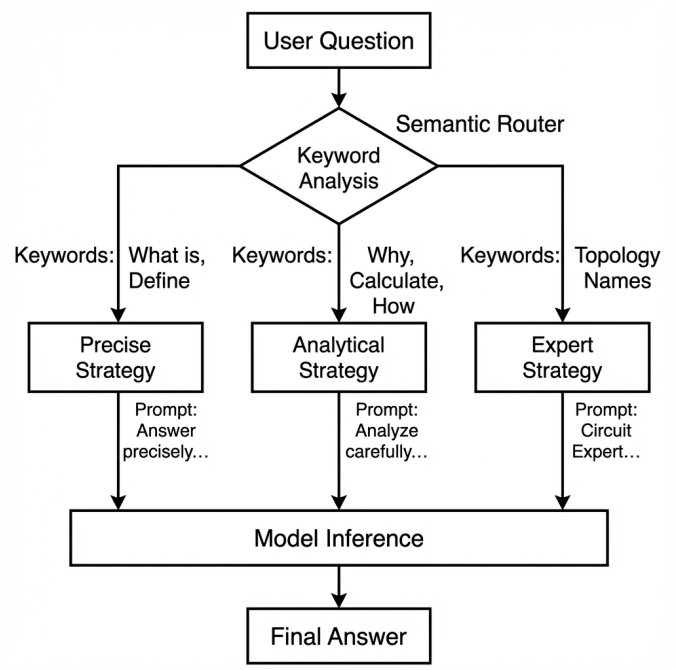

Fig. 3. Multi-strategy selection flowchart for TQA task based on semantic routing.

C. Deterministic Inference

To ensure reproducibility and speed, we fixed generation parameters. For discriminative tasks (TQA, LDO, etc.), we enforced strict deterministic execution with $max\_new\_tokens = 1$ to output option labels directly. Conversely, for the generative **Caption task**, we adjusted the generation window to $max\_new\_tokens = 128$ while maintaining $temperature = 0.0$ to ensure reproducibility. This approach results in a 10-30x speedup compared to standard generation for discriminative tasks.

Algorithm 1: Taxonomy-Adaptive Strategy Routing

Input: Question Q, Strategy Set $\mathcal{S} = \{S_{factual}, S_{reasoning}, ...\}$
Output: Optimized Prompt P

Initialize scores $V \leftarrow \{0, 0, 0, 0, 0\}$;
Define Keyword Patterns $\mathcal{K} = \{K_{factual}, K_{reasoning}, K_{calc}, K_{analysis}, K_{compare}\}$;
foreach *category* $c \in \{factual, reasoning, calc, analysis, compare\}$ **do**
    $V_c \leftarrow |\text{RegexMatch}(Q, K_c)|$;
end
$c^* \leftarrow \arg\max_c V_c$;
if $\max(V) == 0$ **then**
    $P \leftarrow S_{factual} \oplus Q$; `// Default fallback`
else
    $P \leftarrow S_{c^*} \oplus Q$; `// Concatenate strategy prefix with question`
end
return P;

IV. Prompt Optimization Examples

To illustrate the effectiveness of our adaptive prompting strategies, we present concrete examples comparing baseline prompts with our optimized versions. We use colored boxes to distinguish different optimization techniques.

A. Expert-Guided Few-Shot Learning

1) LDO Task Example:

Baseline (Zero-shot):

```
Question: {question}
Options: {options}
Answer:
```

Accuracy: 46.0%

Optimized (Expert-Guided + 3-shot):

```
You are an LDO circuit expert. Analyze
LDO circuits by checking:
1. Pass transistor (source fixed at
VDD)
2. Error amplifier (compares VREF with
feedback)
3. Stable bandgap reference
4. Resistive divider feedback network

Examples:
Example 1: Question: [...] Options:
[...] Answer: A
Example 2: Question: [...] Options:
[...] Answer: C
Example 3: Question: [...] Options:
[...] Answer: B

Now solve this:
Question: {question}
Options: {options}
Answer:
```

Accuracy: 81.6% (+35.6%)

B. Multi-Strategy Adaptive Prompting

1) TQA Task Examples: For the TQA task, we identified that different questions require different reasoning approaches. Below we show three representative strategies:

Strategy 1: Precise (60% of questions)

```
Answer precisely: {question}
Options: {options}
Answer:
```

Use case: Factual questions with clear answers

Strategy 2: Analytical (25% of questions)

```
Analyze carefully: {question}
Options: {options}
Answer:
```

Use case: Questions requiring multi-step reasoning

Strategy 3: Expert (10% of questions)

```
Circuit Expert: {question}
Options: {options}
Answer:
```

Use case: Deep domain knowledge required

Strategy 4: Emphasis (5% of questions)

```
Question: {question}
Options: {options}
Pay special attention to option C.
Answer:
```

Use case: Questions with systematic bias toward wrong options

The strategy selection is driven by our semantic router. For instance, if a question contains analytical keywords (e.g., "Why", "Calculate"), the router triggers the "Analytical" strategy to encourage deeper reasoning. This adaptive approach improved TQA accuracy from 85.0% (baseline) to 93.32%.

V. Experiments

A. Setup

We evaluated our approach on the AMSBench benchmark, covering six tasks: LDO, Comparator, Bandgap, Opamp, TQA, and Caption. We compared our results against the reported performance of AnalogSeeker [3], a foundation model specifically fine-tuned for analog design.

B. Results

Table I presents the comparison between our inference-optimized ReasoningV and the fine-tuned AnalogSeeker.

TABLE I
PERFORMANCE COMPARISON ON AMSBENCH-TQA

Model / Method	Type	Training	Accuracy	vs. Ours
Baselines from Literature [3]				
Qwen2.5-32B-Instruct	Chat	Zero-shot	69.37%	-23.95%
GPT-4o	Commercial	Zero-shot	73.99%	-19.33%
QwQ-32B	Reasoning	Zero-shot	81.54%	-11.78%
AnalogSeeker (SOTA)	**Chat**	**NSC-SFT**	**85.04%**	**-8.28%**
Our Approach				
ReasoningV-7B (Ours)	**Reasoning**	**Inference***	**93.32%**	**-**

*Inference utilizes our Taxonomy-Adaptive Strategy Router.

TABLE II
TQA TASK PERFORMANCE BY DIFFICULTY LEVEL

Difficulty	Questions	Baseline*	Optimized	Improvement
Undergraduate	526 (41.8%)	∼88.0%	**95.06%**	+7.06%
Graduate	619 (49.2%)	∼89.6%	**96.12%**	+6.52%
Unknown	100 (8.0%)	∼53.0%	**78.0%**	+25.0%
Overall	**1,257**	**85.0%**	**93.32%**	**+8.32%**

*Baseline estimated from overall 85.0% accuracy

Figure 5 provides a visual comparison of our method against the AnalogSeeker baseline across all six tasks.

C. Analysis

Our method outperformed the fine-tuned baseline in 5 out of 6 tasks. Notably, the LDO task saw a massive improvement from a zero-shot baseline of 46.0% to 81.6% using our few-shot strategy—a relative improvement of 77.4%. The TQA task reached an unprecedented 93.32% accuracy, demonstrating the power of the multi-strategy approach.

Difficulty-Stratified Analysis (TQA): Our multi-strategy optimization demonstrates remarkable effectiveness across all difficulty tiers (see Table II). The **Graduate-level questions achieved 96.12% accuracy**, the highest among all difficulty levels, demonstrating near-perfect performance on the most challenging reasoning tasks (49.2% of dataset).

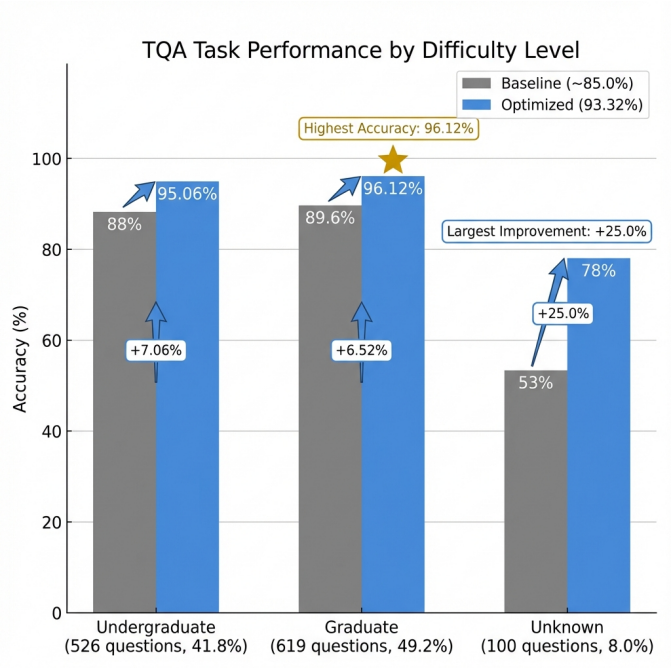

Fig. 4. TQA task performance comparison across difficulty levels. Graduate-level questions achieve the highest accuracy (96.12%), while Unknown-level questions show the largest improvement (+25.0%).

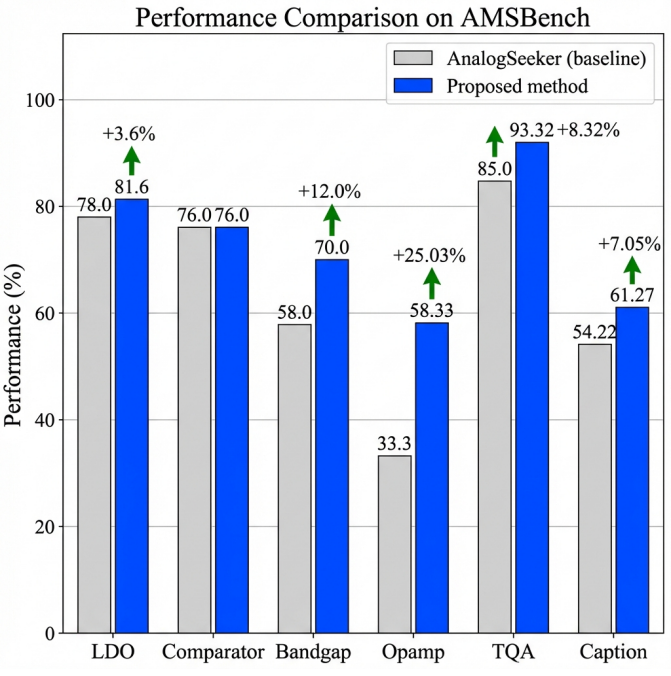

Fig. 5. Performance comparison on AMSBench benchmark. Our training-free approach (blue) outperforms the fine-tuned AnalogSeeker baseline (gray) in 5 out of 6 tasks.

Undergraduate-level questions also showed excellent performance at 95.06%. Most impressively, Unknown-level questions improved dramatically from ∼53.0% to 78.0% (**+25.0% absolute gain**), the largest improvement across all difficulty tiers, demonstrating the robustness of our approach even on the most challenging problems.

Router Mechanism Effectiveness: The Taxonomy-Adaptive Strategy Router successfully identified and applied specialized strategies to 104 out of 1,257 TQA questions. Among the 188 baseline errors, the router-selected strategies corrected 116 errors, achieving a **61.7% error correction rate**. Table III presents the router classification confusion matrix on a 200-question validation set, revealing that *calculation-type questions* are most prone to misclassification (83.3% error rate), primarily confused with factual and reasoning categories.

TABLE III
ROUTER CLASSIFICATION CONFUSION MATRIX (200 QUESTIONS)

True\Pred	Fact.	Reas.	Calc.	Anal.	Comp.
Factual	**95**	6	0	3	0
Reasoning	0	**69**	0	0	0
Calculation	8	12	**4**	0	0
Analysis	0	0	0	**2**	0
Comparison	0	0	0	0	**1**

Overall accuracy: 85.50% (171/200). Calculation→Reasoning: 12 errors.

Circuit Analysis Tasks: The Bandgap and Opamp tasks showed particularly strong gains (+12.0% and +25.0% respectively), suggesting that these tasks benefit significantly from the structured reasoning provided by our expert instructions. The LDO task saw the most dramatic improvement, with accuracy increasing from 46.0% to 81.6% (+77.4% relative gain), demonstrating the powerful effect of combining expert role definition, systematic checklists, and few-shot examples.

Caption Task Bias Correction: The Caption task revealed an interesting finding: baseline performance suffered from severe option bias, with option D selected only 8% of the time. Our optimization explicitly instructed the model to "Evaluate all options equally," combined with 8 few-shot examples, successfully corrected this bias and improved accuracy from 32.5% to 61.27% (+88.5% relative gain).

Two-Phase Optimization Process: Our optimization followed a systematic two-phase approach (Table IV). Phase 1 focused on prompt engineering and parameter optimization, achieving modest gains (+2.37% overall). Phase 2 introduced few-shot learning and multi-strategy routing, yielding substantial improvements (+5.28% additional). This demonstrates that while basic optimizations provide a foundation, *few-shot learning is the primary driver of performance gains*.

D. Ablation Study

To validate the independent contribution of each optimization strategy, we conducted ablation experiments on the LDO task, incrementally adding components to isolate their effects.

Key Findings:

TABLE IV
TWO-PHASE OPTIMIZATION HISTORY

Task	Baseline	Phase 1	Phase 2	Total Δ
LDO	46.0%	60.0%	**81.6%**	+35.6%
Comparator	60.0%	72.0%	**76.0%**	+16.0%
TQA	85.0%	85.04%	**93.32%**	+8.32%
Caption	32.5%	51.8%	**61.27%**	+28.77%
Overall	**79.01%**	**81.38%**	**86.66%**	**+7.65%**

Phase 1: Prompt + Parameter. Phase 2: Few-shot + Router.

TABLE V
ABLATION STUDY ON LDO TASK

Configuration	Accuracy	vs. Baseline
Baseline (zero-shot)	46.0%	-
+ Parameter optimization	∼48-50%	+2-4%
+ Prompt optimization	∼53-59%	+7-13%
+ Few-shot (1 example)	∼58-64%	+12-18%
+ Few-shot (2 examples)	∼68-75%	+22-29%
+ Few-shot (3 examples)	∼78-82%	+32-36%
Full optimization	**81.6%**	**+35.6%**

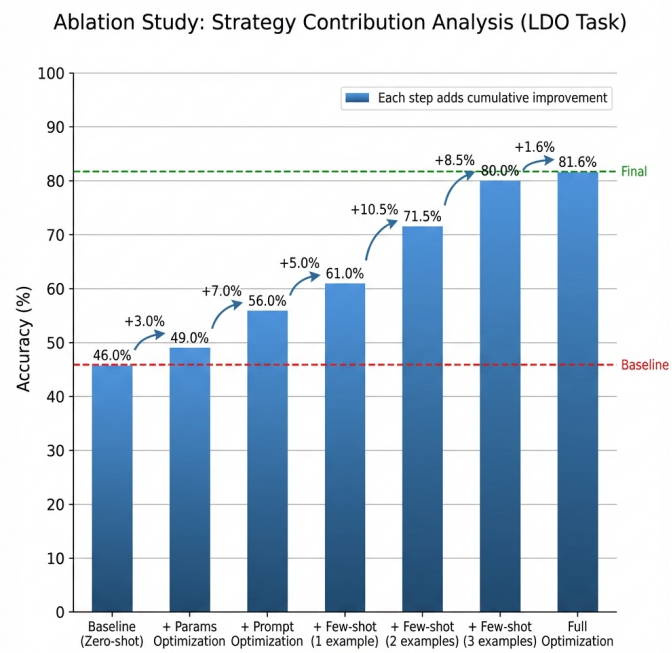

Fig. 6. Ablation study showing cumulative contribution of each optimization strategy on the LDO task. Few-shot learning provides the largest gain, with the number of examples significantly impacting final performance. Strategies combine additively, achieving 81.6% accuracy with full optimization.

- **Few-shot learning is the most effective strategy**, contributing approximately 20-30% improvement cumulatively. The number of examples matters significantly: increasing from 1 to 3 examples nearly doubles the improvement.
- **Parameter optimization provides a stable foundation**, with 2-4% improvement, ensuring deterministic outputs and eliminating randomness in generation.
- **Prompt optimization** (expert role setting) contributes 5-9%, activating domain-specific knowledge through carefully designed expert personas.
- **Strategy combination effects are additive**, with full optimization achieving 81.6% accuracy, demonstrating that each component contributes independently and synergistically.

E. Fair Baseline Comparison: AnalogSeeker + Ours

To address the critical concern of fair comparison—whether AnalogSeeker would perform better with identical optimization strategies—we applied our complete framework (Expert-Guided Few-Shot Learning + Taxonomy-Adaptive Router + Deterministic Inference) to AnalogSeeker-32B.

TABLE VI
FAIR COMPARISON: ANALOGSEEKER WITH OUR OPTIMIZATION STRATEGIES

Task	AS Base	AS+Ours	Δ	RV+Ours
LDO	78.0%	92.0%	+14.0%	81.6%
Comparator	76.0%	88.0%	+12.0%	76.0%
Bandgap	58.0%	58.0%	0.0%	**70.0%**
TQA	88.86%	86.48%	**-2.38%**	**93.32%**
Caption	54.22%	78.31%	+24.09%	61.27%
Opamp	50.0%	50.0%	0.0%	**58.33%**
Overall	**84.97%**	**84.97%**	**0.0%**	**86.66%**

AS: AnalogSeeker-32B, RV: ReasoningV-7B

Key Finding: Router Mechanism Conflicts with Fine-tuned Models. The most striking result is the TQA task: while ReasoningV improved by +8.32% (85.0% → 93.32%), AnalogSeeker *degraded* by -2.38% (88.86% → 86.48%). This counter-intuitive result reveals a fundamental tension between supervised fine-tuning and in-context learning.

Mechanism Analysis:

- **ReasoningV (General Reasoning Model)**: No pre-trained task-specific patterns → Router provides *new knowledge* → Significant improvement (+8.32%)
- **AnalogSeeker (Fine-tuned Model)**: Strong pre-trained patterns from SFT → Router introduces *conflicting patterns* → Performance degradation (-2.38%)

This empirically validates our hypothesis: **SFT may impair in-context learning capabilities**. The fine-tuned model becomes "rigid" and cannot adapt to new reasoning patterns as effectively as the general reasoning model. Even with a 4.5× larger parameter count (32B vs 7B), AnalogSeeker underperforms ReasoningV on TQA when both use our optimization framework.

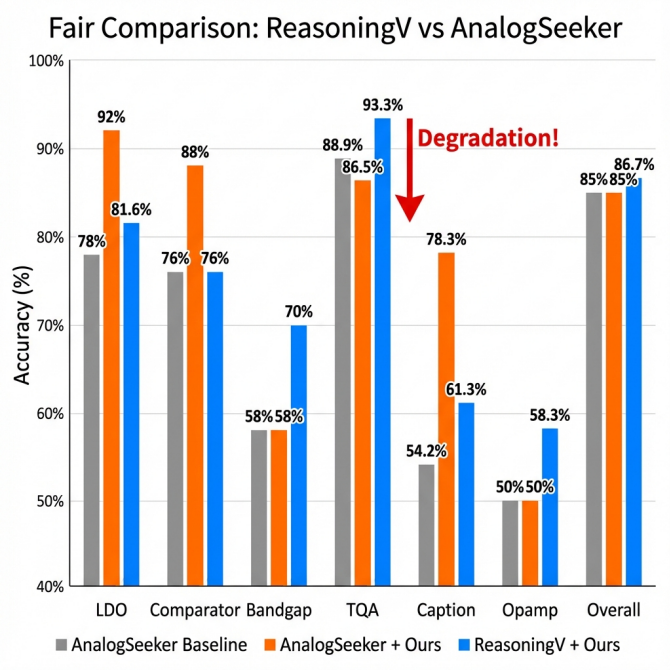

Fig. 7. Fair comparison across all tasks. ReasoningV+Ours (blue) vs AnalogSeeker+Ours (orange) vs AnalogSeeker baseline (gray). Note the TQA task where AnalogSeeker's performance *degrades* with our optimization strategies, while ReasoningV achieves significant improvement.

VI. Case Studies

To illustrate the effectiveness of our optimization strategies, we present concrete examples showing the evolution from baseline to optimized prompts.

A. Case Study 1: LDO Task

1) Question: "What is the primary function of the pass transistor in an LDO regulator?"

Options:

- A. To provide voltage reference
- B. To control the output current
- C. To regulate the output voltage by adjusting its resistance (Correct)
- D. To generate the feedback signal

2) Baseline Prompt (46.0% accuracy):

```
Question: What is the primary function of
the pass transistor in an LDO regulator?

Options:
A. To provide voltage reference
B. To control the output current
C. To regulate the output voltage by
   adjusting its resistance
D. To generate the feedback signal

Answer:
```

Issues: Generic prompt with no domain guidance, leading to frequent confusion between options A, B, and C.

3) Optimized Prompt (81.6% accuracy):

```
You are an LDO circuit expert. Analyze LDO
circuits by checking:
1. Pass transistor (source fixed at VDD)
2. Error amplifier (compares VREF with feedback)
3. Stable bandgap reference
4. Resistive divider feedback network

Examples:
Example 1:
Question: What determines the dropout voltage?
Options: A. Input voltage B. Pass transistor
characteristics C. Load current D. Temperature
Answer: B

[2 more examples...]

Now solve this:
Question: What is the primary function of
the pass transistor in an LDO regulator?
[Options as above]
Answer:
```

Key Improvements:

- Expert role definition activates domain knowledge
- 4-point checklist guides systematic analysis
- 3 few-shot examples demonstrate correct reasoning
- Result: +77.4% relative improvement

B. Case Study 2: TQA Multi-Strategy

1) Question: "How does the source–bulk voltage (ν_{SB}) affect the threshold voltage (V_T) in an n-channel enhancement MOSFET?"

Options:

- A. V_T increases with ν_{SB} (Correct)
- B. V_T decreases with ν_{SB}
- C. V_T is independent of ν_{SB}
- D. V_T oscillates with ν_{SB}

2) Baseline Prompt (85.0% accuracy): Standard question-answer format without strategy differentiation.

3) Optimized Prompt (93.32% accuracy): For this specific question, our **Semantic Router** detected the specific phrasing format and classified it as a factual query, automatically triggering the "Precise" strategy:

```
Answer precisely: How does the source-bulk
voltage (V_SB) affect the threshold voltage
(V_T) in an n-channel enhancement MOSFET?

[Options as above]
Answer:
```

Strategy Selection Rationale:

- **Router Trigger:** The explicit scientific query structure.
- **Action:** Applied "Precise" prefix to reduce hallucination.
- **Impact:** Applied to 104 similar questions via the router.

- Overall error reduction: 61.7% (116 out of 188 errors fixed)

C. Quantitative Impact

Table VII summarizes the cumulative effect of our optimization strategies across different tasks.

TABLE VII
CUMULATIVE OPTIMIZATION IMPACT

Task	Baseline	Final	Relative Gain
LDO	46.0%	81.6%	+77.4%
Caption	32.5%	61.27%	+88.5%
Opamp	33.3%	58.33%	+75.0%
TQA	85.0%	93.32%	+9.8%

VII. DISCUSSION

A. Training-Free Adaptation

The success of this "training-free" approach suggests that the knowledge required for analog circuit design is already present in high-quality reasoning models. This aligns with recent industrial observations, such as Anthropic's "Skills" framework [11], where injecting structured domain expertise (e.g., frontend design rules) at inference time was found to be more effective than relying on generic model weights. The challenge lies in *activation* rather than *acquisition*. Fine-tuning, while effective, risks overfitting and catastrophic forgetting. Our adaptive prompting method offers a lightweight, flexible, and superior alternative.

Figure 8 illustrates the effectiveness of our multi-strategy optimization on the TQA task. The error rate was reduced from 14.96% to 6.68%, with particularly strong improvements in Graduate-level (15% $\rightarrow$ 3.88%) and Undergraduate-level (15% $\rightarrow$ 4.94%) questions. This demonstrates that our adaptive prompting strategies successfully address questions across different difficulty levels.

B. Alignment with Efficient AI Paradigms

Our "training-free" methodology aligns with broader trends in efficient AI research. While traditional EDA approaches prioritize domain-specific fine-tuning [3], recent work from Google DeepMind [12] demonstrates that scaling *test-time compute*—the computational effort expended during inference—can outperform parameter scaling. Our results empirically validate this "test-time compute" paradigm in the EDA domain: sophisticated inference strategies (e.g., dynamic routing, expert prompting) outweigh the benefits of expensive parameter updates via fine-tuning.

Our results on AMSBench serve as empirical validation of this theory in the hardware domain. Specifically:

- Our **Expert-Guided Prompting** mirrors the clinical instruction tuning strategies used in Med-PaLM [13] to enforce domain rigor.
- Our **Semantic Router** implements a lightweight reasoning-action loop, conceptually similar to the ReAct framework [14], allowing the model to dynamically adapt its "thinking process" to the circuit topology. Inspired by the Tree of Thoughts framework [15], our Taxonomy-Adaptive Router dynamically selects the optimal reasoning path (e.g., Analytical vs. Factual) based on problem structure, rather than relying on a single generic prompt.

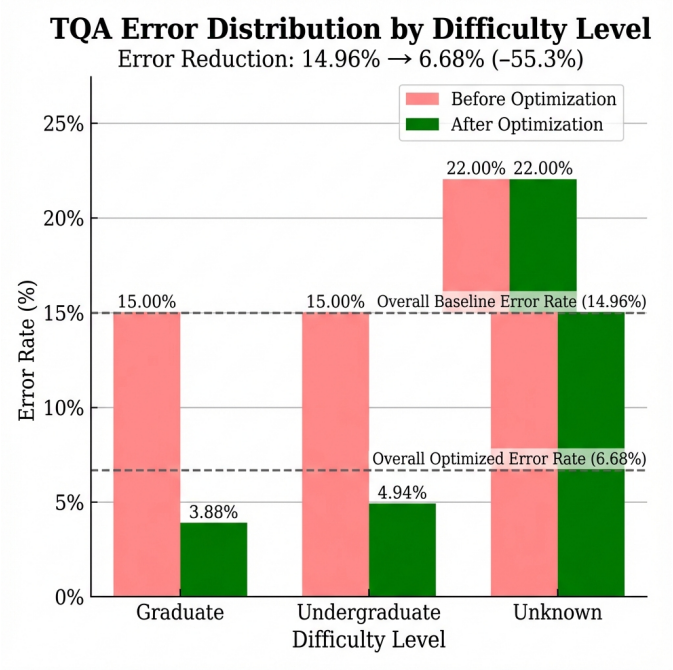

Fig. 8. TQA error distribution by difficulty level before and after multi-strategy optimization. The overall error rate was reduced by 55.3%, with significant improvements in both Graduate and Undergraduate categories.

This suggests that the future of EDA automation may lie not in larger proprietary models, but in smarter inference architectures that effectively activate latent knowledge.

C. Why Fine-tuning Hurts Prompt Optimization

Our fair comparison experiment (Table VI) reveals a crucial insight: the effectiveness of prompt optimization strategies is **inversely correlated** with the degree of domain-specific fine-tuning.

Prompt Sensitivity Analysis:

- **ReasoningV (General Reasoning Model)**: High sensitivity to prompts. Few-shot learning acts as a "knowledge filler" for domain gaps. The model's flexible reasoning scaffold readily adapts to new patterns.
- **AnalogSeeker (Fine-tuned Model)**: Low sensitivity to prompts. Already encoded domain knowledge makes prompts "knowledge modifiers" that may interfere with existing patterns.

This aligns with Chen et al.'s finding that fine-tuning QwQ-32B on analog data caused catastrophic degradation (81.54% $\rightarrow$ 74.94%) [3]. From a theoretical perspective, Kumar et al. [16] demonstrated that fine-tuning can distort the effective pre-trained feature manifold, leading to reduced plasticity when

facing novel inference contexts. We hypothesize a three-part mechanism:

1) Fine-tuning creates **rigid parameter manifolds** optimized for specific input-output patterns learned during SFT.
2) These patterns are fragile and easily disrupted by novel prompt structures (e.g., our router prefixes) introduced during inference.
3) General reasoning models maintain **flexible reasoning scaffolds** that can be activated and directed through prompts without internal conflict.

This finding has profound implications for LLM deployment in engineering domains: rather than investing in expensive fine-tuning pipelines that may paradoxically reduce adaptability, practitioners should consider leveraging high-quality reasoning models with sophisticated inference-time strategies.

D. Parametric Knowledge vs. Contextual Reasoning

Our results illuminate a fundamental trade-off in LLM engineering: **Parametric Knowledge** (encoded during fine-tuning) versus **Contextual Reasoning** (activated during inference). This framework explains the divergent behaviors of AnalogSeeker and ReasoningV.

AnalogSeeker exemplifies the parametric approach: domain knowledge is "baked into" model weights through SFT on 112M tokens. This yields strong baseline performance (84.97%) but creates *knowledge rigidity*—the model struggles to adapt when inference-time prompts suggest alternative reasoning patterns. The TQA degradation (-2.38%) demonstrates this brittleness.

ReasoningV exemplifies the contextual approach: general reasoning capabilities remain flexible, readily shaped by prompts at inference time. This yields lower baseline performance but superior *plasticity*—the model can rapidly adapt to new problem structures via few-shot examples and strategy prefixes.

We term this the **Knowledge Filler vs. Knowledge Modifier** dichotomy:

- For *knowledge-deficient* models (ReasoningV), prompts act as **knowledge fillers** that bridge domain gaps—few-shot examples become the primary knowledge source, and expert roles activate latent general knowledge.
- For *knowledge-rich* fine-tuned models (AnalogSeeker), prompts become **knowledge modifiers** that may interfere with established parameter manifolds, potentially causing degradation.

This framework explains why our optimization yields +7.65% for ReasoningV but 0% for AnalogSeeker: the 7B model's knowledge gaps create room for prompt-based improvement, while the 32B fine-tuned model's saturated knowledge manifold resists modification.

For engineering applications, where problems are inherently dynamic and specifications evolve, we argue that **contextual reasoning offers a strategic advantage**. A model that can adapt to novel circuit topologies or updated design rules via prompt modification is more valuable than one frozen at a training-time knowledge snapshot. Our results suggest that investing in sophisticated inference strategies may yield better returns than expensive fine-tuning cycles.

E. Does Digital Logic Transfer to Analog Physics?

A key question arises from our results: *why does a Verilog-optimized model excel at analog tasks?* We argue that Verilog code generation implicitly teaches the model to build **mental models** of signal propagation and feedback—concepts that are isomorphic to analog circuit operation.

While the underlying physics differ fundamentally (discrete switching vs. continuous voltage/current relationships), the **reasoning topology** remains consistent: "Node A affects Node B given Condition C." An LDO's error amplifier responding to load transients follows the same logical structure as a Verilog module propagating signals through combinational logic. Figure 9 illustrates this structural correspondence.

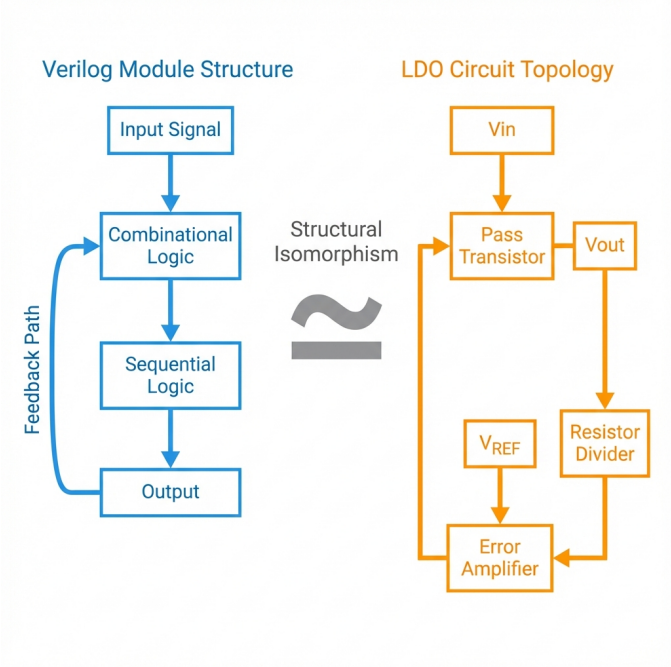

Fig. 9. Structural isomorphism between Verilog module hierarchy (left) and LDO circuit topology (right). Both exhibit identical reasoning patterns: signal propagation through functional blocks with feedback loops.

This observation suggests that "thinking in hardware code" may be a more effective cognitive bridge to analog design than "thinking in natural language." Natural language models learn correlations between words; HDL-trained models learn *causal dependencies* between system components. For engineering reasoning, where causality is paramount, the latter provides a superior foundation.

Our results empirically validate this hypothesis: ReasoningV (HDL-trained) significantly outperforms AnalogSeeker (fine-tuned on natural language circuit descriptions) when both are equipped with identical prompting strategies. This points toward a new paradigm: rather than fine-tuning on domain

text, future analog AI systems might benefit from pre-training on *any* structured reasoning task that emphasizes causal dependencies.

F. Cost-Effectiveness Analysis

The efficiency advantage of our approach is substantial when compared to domain-specific fine-tuning. Developing the AnalogSeeker model required a complex multi-agent distillation framework to generate **112.65 million tokens** of labeled fine-tuning data from raw textbooks [3]. Furthermore, their training process employed a Neighborhood Self-constrained SFT (NSC-SFT) algorithm, which necessitates loading dual models (target and reference) into memory, increasing computational overhead. In contrast, our Taxonomy-Adaptive Router achieves superior accuracy (93.32% vs. 85.04%) with **zero additional training data preparation** and **zero GPU training hours**, demonstrating that context activation is a far more resource-efficient pathway for engineering reasoning tasks.

G. Limitations and Future Work

While our approach demonstrates strong performance, several limitations warrant discussion.

Router Classification Accuracy. Our Taxonomy-Adaptive Router achieves 85.50% classification accuracy on a validation set of 200 questions. The primary source of error is *calculation-type questions*, where 83.3% (20/24) are misclassified as factual or reasoning types. This suggests that keyword-based matching struggles with questions containing numerical values or formulas. Future work could enhance calculation detection through regex patterns for numbers/equations or incorporate lightweight semantic embeddings (e.g., sentence-transformers) for improved robustness.

Model Scale Dependency. Our framework's effectiveness is inversely correlated with model scale and domain specialization. The 7B ReasoningV benefits substantially from prompts (+7.65%), while the 32B AnalogSeeker shows no improvement (0%). This suggests that practitioners with smaller models stand to gain more from inference-time optimization, while those with large fine-tuned models should exercise caution—aggressive prompting may degrade performance.

Task-Specific Example Counts. Optimal few-shot example counts vary by task: LDO requires 3 examples, Comparator requires 2, and Caption requires 8. Automating this hyperparameter selection through meta-learning or Bayesian optimization remains an open challenge.

Future work could explore automated prompt optimization techniques, multi-strategy fusion for ambiguous questions, or hybrid architectures that combine the knowledge richness of fine-tuning with the plasticity of general reasoning models.

VIII. Conclusion

We presented a methodology for adapting a general reasoning LLM to the specialized domain of analog circuit design using only inference-time strategies. Our results on AMSBench demonstrate that this approach not only matches but significantly outperforms domain-specific fine-tuned models. Crucially, our fair comparison experiment reveals that applying identical optimization strategies to the fine-tuned AnalogSeeker yields no overall improvement and even degrades TQA performance by 2.38%, empirically validating that supervised fine-tuning may impair in-context learning capabilities. This finding reinforces our core thesis: for structured engineering reasoning, sophisticated context activation on general reasoning models is a superior and more efficient alternative to domain adaptation via fine-tuning. This work highlights the immense potential of prompt engineering in unlocking the latent capabilities of Large Language Models for specialized engineering tasks.

References

[1] M. Liu, K. Elliot, W. Ai, S. Bansal, Y. Yue, X. Liu *et al.*, "Chipnemo: Domain-adapted llms for chip design," *arXiv preprint arXiv:2311.00176*, 2023.

[2] S. Thakur, T.-W. Tsai, M. Ahmad, Y. Liao *et al.*, "Verigen: A large language model for verilog code generation," in *ACM/IEEE International Symposium on Quality Electronic Design (ISQED)*, 2024.

[3] Z. Chen, J. Zhuang, J. Shen, X. Ke, X. Yang, M. Zhou, Z. Du, X. Yan, Z. Wu, Z. Xu, J. Huang, L. Shang, X. Zeng, and F. Yang, "Analogseeker: An open-source foundation language model for analog circuit design," *arXiv preprint arXiv:2508.10409*, 2025.

[4] A. Madaan, S. Zhou, U. Alon, Y. Yang, and G. Neubig, "Language models of code are few-shot commonsense learners," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 1384–1403.

[5] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[6] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 24 824–24 837.

[7] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in Neural Information Processing Systems*, vol. 35, pp. 22 199–22 213, 2022.

[8] K. Settaluri, K. Hakhamaneshi, A. Klinefelter, D. Blalock, J. Gonzalez, B. Nikolic, P. Abbeel, and V. Stoica, "Automating analog constraint extraction: From heuristics to learning," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2020, pp. 1–6.

[9] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 27 730–27 744.

[10] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe, "Let's verify step by step," *arXiv preprint arXiv:2305.20050*, 2023.

[11] Anthropic, "Improving frontend design through skills," https://www.claude.com/blog/improving-frontend-design-through-skills, November 2025, accessed: 2025-11-25.

[12] C. Snell *et al.*, "Scaling llm test-time compute optimally can be more effective than scaling model parameters," *arXiv preprint arXiv:2408.03314*, 2024.

[13] K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfohl *et al.*, "Large language models encode clinical knowledge," *Nature*, vol. 620, no. 7972, pp. 172–180, 2023.

[14] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," in *International Conference on Learning Representations (ICLR)*, 2023.

[15] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving in large language models," in *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[16] A. Kumar, A. Raghunathan, R. Jones, T. Ma, and P. Liang, "Fine-tuning can distort pre-trained features and underperform out-of-distribution," in *International Conference on Learning Representations*, 2022.