SIEVE: Sample-Efficient Parametric Learning from Natural Language

Parth Asawa [1] **Alexandros G. Dimakis** [1 2] **Matei Zaharia** [1]

Abstract

Natural language context—such as instructions, knowledge, or feedback—contains rich signal for adapting language models. While in-context learning provides adaptation via the prompt, parametric learning persists into model weights and can improve performance further, though is data hungry and heavily relies on either high-quality traces or automated verifiers. We propose SIEVE, a method for sample-efficient parametric learning from natural language context that requires as few as three query examples. SIEVE uses a novel synthetic data generation pipeline, SIEVE-GEN, that leverages the insight that context is *decomposable*. Decomposing context allows us to generate higher quality rollouts by pairing synthetic queries with only the applicable context rather than the entirety, then using context distillation to internalize context into the model. We evaluate in reasoning settings where context is necessary, including custom domains and the RuleArena and Machine Translation from One Book tasks. Our results show that SIEVE outperforms prior context distillation methods using just three query examples, demonstrating how to achieve sample-efficient parametric learning from natural language.

1. Introduction

Language models today rely heavily on in-context learning (ICL) to adapt to new tasks: users provide examples, instructions, feedback, or domain knowledge directly in prompts to guide model behavior (Brown et al., 2020). This approach has become ubiquitous—from developers writing extensive system prompts with coding standards and API documentation, to users providing personal preferences and writing styles, to domain experts supplying specialized knowledge for medical diagnosis or legal reasoning (Xu et al., 2024; Liu et al., 2025; Sahoo et al., 2025). However, ICL has fundamental limitations: it cannot leverage the benefits of parametric learning, such as eliminating context window constraints, enabling persistent improvements that survive across sessions, and improving performance through additional training compute.

Due to these limitations, a line of work has emerged exploring parametric learning—directly internalizing context into model weights through training. Context distillation approaches (Snell et al., 2022; Bhargava et al., 2024; Upadhayayaya et al., 2025) "bake" instructions and examples into weights by training a student model to imitate a teacher with access to context. Methods for training with natural language feedback (Scheurer et al., 2022; Chen et al., 2024) show that textual corrections can improve model outputs. However, these parametric methods face a critical bottleneck: they are data-hungry, typically requiring many examples of queries or expensive expert-generated traces and automated verifiers. This creates a wide gap: ICL works with minimal examples but cannot access parametric benefits, while parametric methods offer these benefits but demand abundant data. **Can we achieve the advantages of parametric learning with the sample efficiency of in-context learning?**

We demonstrate that the answer is yes. With **as few as three examples of task queries**, models can internalize natural language context requiring multi-step reasoning, outperforming prior context distillation methods and even matching or exceeding ICL performance without needing context at inference time.

To create a sample-efficient method, we seek to leverage synthetic queries and rollouts that are created from a natural language context ($\mathcal{C}$) and very few examples of queries. Our key insight is that **natural language context is decomposable**. Natural language context often consists of independent context units (u) where only a subset applies to any given query. For example, a list of rules can be broken into individual parts, with each task requiring only a few; a grammar specification decomposes into individual constraints, with each translation needing only relevant ones. The quality of context distillation training is heavily linked to the quality of the rollouts with context (r). Decomposing context allows us to filter and only include the context units that are applicable to a particular query ($c_a \subseteq \mathcal{C}$), leading to

[1]University of California, Berkeley, CA, USA [2]Bespoke Labs. Correspondence to: Parth Asawa <pgasawa@berkeley.edu>. Code available at: https://github.com/pgasawa/sieve.

Preprint. February 4, 2026.

higher quality rollouts for training than prior methods that indiscriminately provide all context for all queries.

Using this, we propose SIEVE, a method that achieves sample-efficient parametric learning from natural language context. The heart of our approach is **SIEVE-GEN**, a novel synthetic data generation pipeline that requires natural language context and just a few examples of queries. SIEVE-GEN breaks context into context units, leverages a base language model to create diverse sets of context units for generating synthetic queries, then filters which units actually apply to each query. This produces training data where queries are paired with *only* their applicable context, yielding higher-quality rollouts. We then apply standard context distillation techniques (Snell et al., 2022), distilling the model's behavior when conditioned on applicable context into weights that can perform the same reasoning without context. Avoiding prior limitations, SIEVE enables parametric learning from natural language both without access to many examples of queries over context nor high quality traces or verifiers.

We evaluate SIEVE on domains requiring reasoning over context (not just factual recall): our Retail domain tests compositional rule application over 30 discount rules, RuleArena (NBA) (Zhou et al., 2025) evaluates complex sports regulation reasoning, and MTOB (Tanzer et al., 2024) requires translating extremely low-resource languages from grammar specifications. Unlike prior work on parametric internalization that focuses on memorizing facts (Eyuboglu et al., 2025; Lin et al., 2025; Yang et al., 2024), these tasks require models to reason over internalized context at inference time.

Our contributions are:

1. We demonstrate that sample-efficient parametric learning from natural language context is achievable with *as few as three task examples*, bridging the gap between ICL's sample efficiency and parametric learning's advantages.
2. We introduce SIEVE-GEN, a novel synthetic data generation method that exploits context decomposability to create diverse, high-quality training data by pairing queries with only their applicable context.
3. We show empirically that models trained with SIEVE outperform prior context distillation methods and can match or exceed in-context learning performance without context at inference time, across multiple reasoning domains and model families.

These findings establish that parametric learning can be practical for incorporating natural language context, enabling persistent improvements from minimal input.

2. Related Work

Context Distillation. Work in context distillation, also known as prompt baking or prompt distillation, seeks to internalize information from prompts directly into model weights (Bhargava et al., 2024; Upadhayayaya et al., 2025; Snell et al., 2022; Shenfeld et al., 2026). These methods typically employ a teacher-student framework, where a student model is trained to mimic the output distribution of a teacher model that has access to additional context. While sharing the goal of parametric internalization, all of these works assume access to a distribution of queries or expert demonstrations (traces) from which to sample rollouts. Our work differs by focusing on **sample-efficient learning from only minimal query examples**.

Training with Natural Language Feedback. Several approaches integrate natural language context during training (Scheurer et al., 2022; Chen et al., 2024; Hübotter et al., 2026). Rather than relying on scalar preference scores, these methods use rich textual feedback to generate refined outputs for training. For instance, Scheurer et al. (2022) uses human feedback to generate multiple refinements and train on the highest-quality version. While we also adopt this pattern of using natural language context to create training examples, our contribution focuses on achieving this under extreme data constraints **without requiring expert-curated traces, verifiers, or task-specific datasets**.

Synthetic Data for Parametric Knowledge Injection. Recent work explores using synthetic data to inject knowledge into model parameters. Cartridges and related knowledge injection methods (Eyuboglu et al., 2025; Kujanpää et al., 2025) generate synthetic conversations from long-context document corpora and apply context distillation or lightweight adapters to compress knowledge into KV caches or LoRAs. Active Reading and Synthetic Continued Pretraining (Lin et al., 2025; Yang et al., 2024) train models to acquire factual knowledge via self-generated synthetic data during pretraining. These approaches primarily target fact recall and memorization, evaluating whether models can retrieve stored information from parameters. In contrast, our work studies internalizing natural language context that requires multi-step reasoning and selective applicability at inference time. Our method and benchmarks therefore emphasize **reasoning over context**, rather than simply recalling stored facts.

3. Methods

We seek to design a method that integrates natural language context into model weights in a sample-efficient way. Unlike prior work that focuses primarily on memorization or factual recall, our approach targets the internalization of knowledge that requires complex reasoning at inference

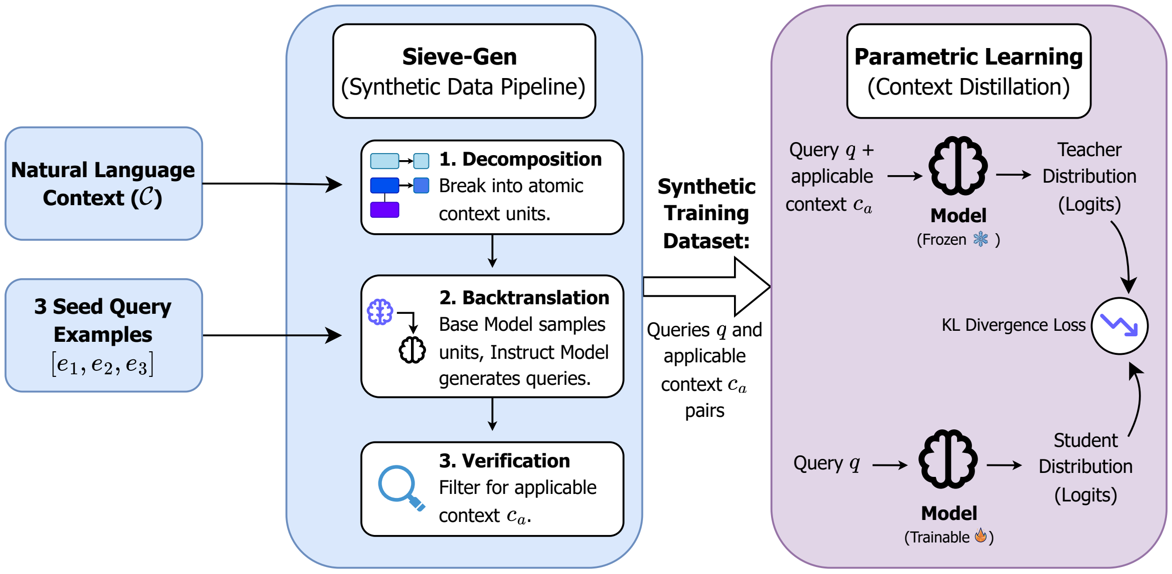
Natural Language Context ($\mathcal{C}$)
3 Seed Query Examples $[e_1, e_2, e_3]$
Sieve-Gen (Synthetic Data Pipeline)
1. Decomposition Break into atomic context units.
2. Backtranslation Base Model samples units, Instruct Model generates queries.
3. Verification Filter for applicable context c_a.
Synthetic Training Dataset:
Queries q and applicable context c_a pairs
Parametric Learning (Context Distillation)
Query q + applicable context c_a
Model (Frozen)
Teacher Distribution (Logits)
KL Divergence Loss
Query q
Model (Trainable)
Student Distribution (Logits)

Figure 1. **SIEVE system overview.** Given a natural language context corpus and as few as 3 seed query examples, SIEVE-GEN generates synthetic training data composed of (query, applicable context) pairs. These pairs are used for context distillation, where a student model learns to match a teacher's distribution conditioned on applicable context, internalizing the knowledge into weights for inference without context.

time. Importantly, our method is **domain-agnostic**—we use no task-specific prompt engineering, applying the same approach across all domains we evaluate (Appendix C). Further gains could likely be achieved per-domain by doing so.

Context distillation methods typically make use of many example queries and sample rollouts with the entire context to update the policy. Synthetic data provides an avenue around this, though high quality synthetic data is challenging. As motivated in Section 1, SIEVE leverages a key insight: when integrating natural language context into weights, **not all context applies to every query**. Simultaneously, natural language context is often compositional; multiple portions may apply to any given query. Decomposing the context we wish to distill into context units enables us to precisely filter context into only what is applicable for particular queries. This filtering leads to higher quality rollouts which in turn leads to higher quality context distillation.

Overview. Leveraging these insights, our approach consists of four steps (Figure 1):

1. Obtain a set of natural language context $\mathcal{C}$ (from humans, automated systems, other LLMs, etc.) relevant to a task or domain, and example queries $[e_1, e_2, e_3]$ that serve as examples to a generator.
2. Generate synthetic tuples of (query, applicable context units) pairs (q, c_a) where $c_a \subseteq \mathcal{C}$ using SIEVE-GEN.
3. Sample a response r from the model conditioned on the query and only the applicable context: $(q, c_a) \to r$.
4. Train the model via context distillation to produce r given only q, internalizing the context into weights.

3.1. SIEVE-GEN: Synthetic Data Generation

The synthetic data generation process we propose, SIEVE-GEN, is crucial to the sample efficiency of our method. Unlike typical synthetic data generation that simply samples model outputs for given queries, SIEVE-GEN needs to both generate diverse queries *and* pair each query with precisely the subset of context units applicable to it, exploiting the decomposability of natural language context.

SIEVE-GEN operates in three phases: decomposition, backtranslation, and verification, and runs completely offline.

Decomposition. We first prompt an instruction-tuned model to decompose the context corpus $\mathcal{C}$ into atomic context units $\{u_1, u_2, \ldots, u_n\}$ that can be independently evaluated for applicability. Each context unit is intended to be a self-contained piece of knowledge or instruction. This decomposition is performed by the model to ensure context units are semantically coherent (domain-agnostic prompt provided in Appendix Section C). Decomposition might turn a list of rules or instructions into individual ones, grammar specifications into individual constraints, etc. We provide example context units for each of the domains we evaluate on in Appendix B.

Backtranslation. To generate diverse synthetic queries, we perform backtranslation from context to queries:

1. A base language model (trained only for next-token prediction) samples a subset of context units to serve as a seed: $c_{\text{seed}} \subseteq \{u_1, \dots, u_n\}$
2. Given the set of seed context units c_{seed} and example queries e_1, e_2, e_3, an instruction-tuned model generates a synthetic query q for which the seed context units would apply.

The use of a base model for seed selection is critical for diversity. We observed that instruction-tuned models consistently select virtually the exact same subsets of context, leading to narrow coverage of the context space (similar to results shown by Zhu et al. (2025)). Following BARE (Zhu et al., 2025), we found that base models produce substantially more diverse seeds, resulting in broader internalization of the full context corpus. We utilize a model rather than randomly sampling from the set of context units to ensure coherence in the seed context c_{seed} from which a query is generated.

Verification. After generating query q from seed context c_{seed}, we verify which context actually applies to this query.

The model iterates through all context units $\{u_1, \dots, u_n\}$ and outputs a binary decision for each, producing the verified applicable context $c_a \subseteq \mathcal{C}$. The model is prompted to assess whether each context unit is necessary to answer the query (see Appendix C). Along with the rest of SIEVE-GEN, this verification step is done entirely offline and motivated by the fact that the synthetic query may require additional context beyond the seed, or conversely, not all seed context may actually be necessary.

The resulting tuple (q, c_a) is then used to generate training data as in step 3 of our overview.

Long Context Extension. While we focus our method on natural language context that fits in context for a model to learn from, for settings where context exceeds the model's context window (e.g., MTOB, which uses 50K tokens for grammar books), we extend SIEVE-GEN to handle long contexts. We chunk the context corpus by token count (default: 8192 tokens per chunk) with 512-token overlaps to guard against splitting related content. Each chunk is then processed through the decomposition phase independently into a list of context units.

For the verification phase with long contexts, the compute required to generate data by verifying every single point of context individually can become quite large. Instead, we can batch context units together rather than evaluating them individually. Here instead of binary yes/no outputs, the model identifies which specific units from each batch are applicable to the query, reducing computational cost.

Algorithm 1 SIEVE-GEN: Synthetic Data Generation

```
Require: Context $\mathcal{C}$, base model $M_{\text{base}}$, instruction model $M_{\text{inst}}$, num. examples $N$, query examples $[e_1, e_2, e_3]$
Ensure: Synthetic dataset $\mathcal{D} = \{(q_i, c_{a,i}, r_i)\}_{i=1}^N$
1: $\{u_1, \dots, u_n\} \leftarrow$ Decompose($\mathcal{C}, M_{\text{inst}}$)
2: for $i = 1$ to $N$ do
3:     $c_{\text{seed}} \leftarrow$ SampleSubset($\{u_1, \dots, u_n\}, M_{\text{base}}$)
4:     $q_i \leftarrow$ GenerateQuery($c_{\text{seed}}, M_{\text{inst}}, [e_1, e_2, e_3]$)
5:     $c_{a,i} \leftarrow$ Verify($q_i, \{u_1, \dots, u_n\}, M_{\text{inst}}$)
6:     $r_i \leftarrow M_{\text{inst}}([q_i, c_{a,i}])$
7: end for
8: return $\mathcal{D}$
```

3.2. Context Distillation Objective.

We base our context distillation objective off of the work of Snell et al. (2022).

Given an input query q and applicable natural language context $c_a \subseteq \mathcal{C}$, the original model M_θ produces a response conditioned on both q and c_a. Rather than training on the model's discrete output, we distill the model's conditional distribution into a student model via soft targets, while removing context from the student input.

Concretely, for each pair of synthetic query (q) and applicable context (c_a) we generate, we construct a teacher input

$$c = [q; c_a], \tag{1}$$

and obtain the teacher distribution

$$p_T(y \mid q, c_a) = M_\theta(c). \tag{2}$$

We retain the top-K logits from p_T to form a truncated soft target distribution $\tilde{p}_T$, which preserves fine-grained preference information while remaining computationally efficient. We set $k = 100$ for our experiments following Snell et al. (2022).

The student model M_ϕ is trained on the same query x *without* access to any context c_a, and is optimized to match the teacher distribution:

$$\mathcal{L}_{\text{CD}} = \text{KL}\big(\tilde{p}_T(y \mid q, c_a) \parallel M_\phi(y \mid q)\big). \tag{3}$$

The model's parameters are optimized over this objective. We experimented with alternative options such as LoRA training though saw worse performance.

Models. We focus our experiments on the Qwen3 8B family of models (Qwen Team, 2025). We provide additional ablations with other model families such as Llama 3.1 8B (Llama Team, 2024) and Rnj 1 8B (Vaswani et al., 2025a;b) in Section 4.4.2. We sample rollouts for distillation from the same model we train in all experiments.

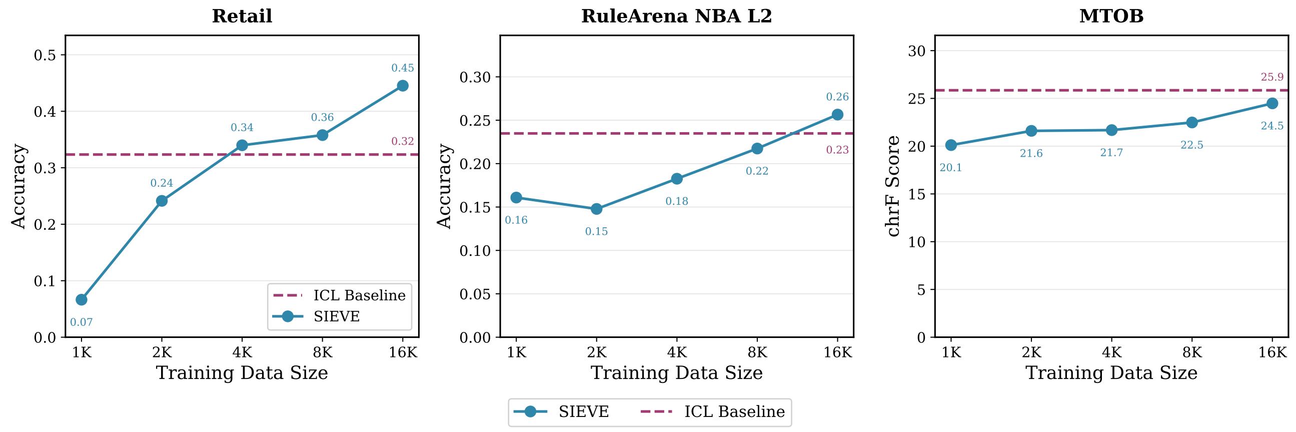

Figure 2. **SIEVE improves with scale while real data input is constant**. Across various domains, SIEVE improves as we scale the amount of data we generate with SIEVE-GEN (using the same fixed three example queries as inputs), approximately matching or exceeding ICL baseline performance when evaluated without any context. All domains use the Qwen3-8B model family with thinking disabled.

4. Evaluation

We show that with SIEVE, natural language context can be efficiently internalized into model weights and reach performance comparable to or surpassing ICL baselines.

4.1. Domains

Because existing benchmarks rarely study adapting from natural language context, we both construct a controlled synthetic domain and modify other benchmarks to fit our setting. We evaluate SIEVE on domains requiring reasoning over context (not just factual recall) that are verifiable. We note that we expect methods such as SIEVE that learn from natural language to work well in, and potentially address a larger gap in, less verifiable domains as well (such as personalization), though for the purposes of controlled study, we stick to verifiable ones.

Retail. We first introduce the Retail domain, a synthetic task carefully constructed to be verifiable and explicitly require the natural language context to solve any instance of the task. This domain further differs from prior studies on parametric learning in requiring the model to reason over the compositionality of its context and selectively apply relevant rules at inference time, rather than simply recall facts.

The model is tasked with calculating the price of a shopping cart given 30 discount rules that conditionally may apply. These rules are not provided at evaluation time when testing parametric learning methods. We programatically generate 256 evaluation queries as well as the corresponding ground truth price. Accuracy is measured as binary exact accuracy within .01 error. We provide a full example query and the entire list of natural language rules for the Retail domain in Appendix B.1.

RuleArena (NBA). We then turn to existing benchmarks. We take a task from the RuleArena benchmark (Zhou et al., 2025), typically intended as a complex instruction-following benchmark, and apply context distillation methods to internalize the entire corpus of natural language rules a model would normally be provided with in context. This corpus is approximately 20,000 tokens of NBA player trade rules and regulations to determine if a sequence of trades is illegal and if so why. We use the Level 2 setting for evaluation, which comprises the most difficult tasks in the benchmark. Accuracy is measured as exact match on the legality determination and violation identification.

MTOB. Though we focus our work on context that requires reasoning, for the sake of evaluating on very long context settings, we turn to the Machine Translation from One Book (MTOB) benchmark (Tanzer et al., 2024). Here, a model is tasked with translating an extremely low-resource language, Kalamang, for which there exists very limited resources outside of the corpus provided in the benchmark, to English. The grammar book and parallel examples span approximately 50,000 tokens, exceeding typical context windows and requiring $2\times$ RoPE scaling for the ICL baseline. This domain emphasizes memorization more than compositional reasoning, making it a complementary challenge to Retail and RuleArena. For the MTOB domain specifically, we also omit the verification step. Because the model lacks prior knowledge of the target language, synthetic queries can only incorporate concepts present in the seed context. In contrast, for domains where the model has background knowledge, verification is necessary—the model might generate queries that implicitly require additional context beyond the seed. Performance is measured with the chrF metric (Popović, 2015).

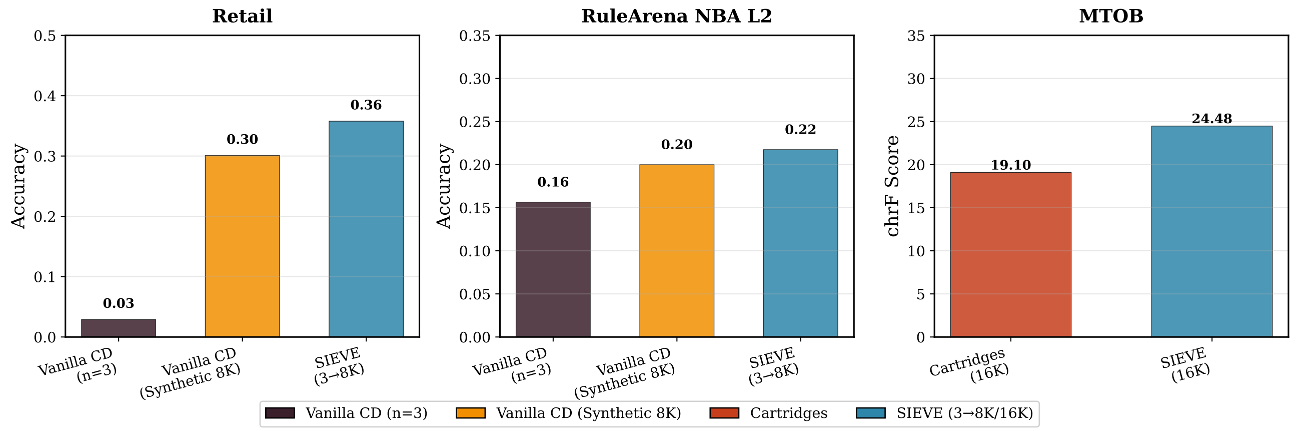

Figure 3. **Comparison to baseline context distillation methods.** We compare SIEVE against vanilla context distillation baselines across domains. V_{CD} (3 seeds) trains on only the three seed query examples with all context. V_{CD-S} (8K) uses our synthetically generated queries but includes all context during rollout generation (no selective filtering). SIEVE generates synthetic data from three seeds to 8/16K scales and outperforms baselines with the same amount of training data in all scenarios.

4.2. SIEVE: Scaling Data

In Figure 2, we evaluate how SIEVE scales with increasing amounts of synthetic data. Critically, all experiments across all domains use only the context and three query examples—no additional expert-curated traces or automated verifiers are required. These three examples mainly serve as formatting examples for the synthetic data generation. From the fixed context and three seeds, we evaluate SIEVE-GEN on up to 16,000 synthetic tuples of queries paired with their applicable context from the broader corpus of natural language context. We train for 2 epochs on the Retail and RuleArena domains and 5 epochs on the MTOB domain, chosen by empirically seeing when loss converged. Further details of training hyperparameters used by our runs of SIEVE are provided in Appendix A.

We compare our method with scale to an **in-context learning baseline** where the natural language context is provided directly in context at inference time. This is typically the gold baseline in traditional context distillation works, and what we aim to approximately match or potentially even surpass with our approach.

We consistently find across all three domains that performance improves as synthetic data scales, and trends suggest that scaling data generation further would continue to yield gains. Notably, SIEVE roughly matches or outperforms ICL baselines at the scales we evaluate, with this effect most prominent in shorter-context settings (Retail). The performance further emphasizes that our method's data generation process allows for a trained model to compose different pieces of internalized context together at inference time, improving over ICL by 37.7%.

For RuleArena and MTOB, the longer context nature (20-50K tokens requiring RoPE scaling) makes for more challenging tasks, yet SIEVE still shows consistent improvement with scale and with 16K data points can match ICL performance without relying on any context.

These results demonstrate that sample-efficient parametric learning from natural language is achievable: with only three query examples, our method can internalize complex reasoning knowledge and match ICL performance without requiring context at inference time.

4.3. Comparison to Baseline Context Distillation Methods

We evaluate the effectiveness of SIEVE in contrast to alternative context distillation approaches and show that it can outperform vanilla context distillation methods as well as methods designed for longer contexts. Results are shown in Figure 3.

4.3.1. BASELINES:

Vanilla Context Distillation (V_{CD}): Traditional context distillation methods, such as proposed in Snell et al. (2022); Bhargava et al. (2024), do not leverage synthetic data towards sample-efficient learning. Thus we provide a result if we were to train on the same three seed query examples our method receives (and all context). Since vanilla context distillation does not perform synthetic data generation, this baseline operates under severe data constraints and is meant to show the importance of sample-efficient approaches. We train till loss converges.

Vanilla Context Distillation w/ Synthetic Queries (V_{CD-S}): To create a fairer comparison, we provide vanilla context distillation with access to our synthetically generated queries (up to 8K examples). However, unlike SIEVE, this baseline includes *all* context in context during rollout generation, without filtering to only applicable context. As a result, this also serves as an ablation to the filtering portion of our method.

Cartridges: For MTOB, the 50K token corpus exceeds the model's 32K context window, making standard context distillation infeasible. We instead compare against Cartridges (Eyuboglu et al., 2025), a method explicitly designed for long-context parametric memorization into KV caches, matching the amount of data to our method and using their recommended training hyperparameters. We do not evaluate Cartridges on Retail or RuleArena, as it is designed for factual recall from long documents rather than compositional reasoning over rules and instructions. Preliminary experiments suggested improvements over no-context baselines were marginal.

Results. On Retail, V_{CD} with only three seed examples achieves just 3% accuracy—insufficient data for the model to learn 30 compositional discount rules. When provided with 8K synthetic queries (V_{CD-S}), performance improves dramatically to 30%, but SIEVE still outperforms at 36% by filtering to only applicable context during training. This demonstrates that **selective applicability is critical**: even with abundant synthetic queries, including all context indiscriminately yields worse results than our targeted approach.

On RuleArena (NBA), we observe similar trends: SIEVE achieves a 10% lift over V_{CD-S} that uses our synthetic data and 37.5% over a context distillation approach that only uses the three seeds.

For MTOB, vanilla context distillation is infeasible due to the 50K token corpus exceeding context limits. Instead, Cartridges achieves 19.10 chrF score—outperforming having no internalized knowledge (15.88)—while SIEVE reaches 24.48 (both using 16K data points, though Cartridges targets the KV cache while SIEVE targets the model weights). This represents a substantial improvement, though both methods fall short of the ICL baseline due to the difficulty of the long-context memorization task.

Oracle Query Experiment. To further isolate the importance of selective context filtering, we conduct an additional experiment on Retail where we programmatically generate queries from our ground truth pipeline rather than using synthetically generated queries — thus perfectly matching the train and test distributions. We provide these oracle queries to vanilla context distillation, which includes all context in context during rollout generation. Results in Table 1 show that even with perfect queries, vanilla context distillation achieves only 27.11%, while SIEVE with synthetic queries achieves 33.98%. This result underscores that selective applicability—pairing queries with only their relevant context—is potentially more important than query quality alone.

Table 1. **Oracle access to programmatic queries on Retail.** Comparison under perfect access to programmatically generated queries, where the baseline method is trained on these ground truth (non-synthetic) queries and full context in the rollouts. Our selective context distillation outperforms vanilla context distillation even in this unfavorable setting where we use synthetic data. Both methods use 4096 examples for training.

METHOD	MEAN ACCURACY (%)
VANILLA CD (ORACLE QUERIES)	27.11
SIEVE	**33.98**

Table 2. **Multiple rollouts vs. scaling distinct queries on Retail.** We find that there is a tradeoff between generating multiple rollouts per query versus scaling the number of distinct queries, holding total generations constant, potentially indicating another axis for putting more compute in towards improved performance when data diversity is saturated.

SETTING	DISTINCT QUERIES	ACCURACY (%)
512×8	512	30.23
4096×1	4096	**33.98**
1024×8	1024	**37.97**
8192×1	8192	35.78

4.4. Ablations

4.4.1. MULTIPLE ROLLOUTS VS. SCALING DATA

Recent work has suggested that generating multiple rollouts per query can improve distillation performance without requiring additional distinct data (Guha et al., 2025). To investigate this tradeoff in our setting, we compare increasing the number of rollouts per query to scaling the number of distinct queries, while holding the total number of teacher generations approximately constant. Specifically, we compare two controlled settings: (i) generating 8 rollouts per query at smaller scales (512×8 and 1024×8), and (ii) scaling the number of distinct queries to matched totals (4096×1 and 8192×1). Results are shown in Table 2.

We observe a tradeoff between scaling the number of distinct queries and increasing the number of rollouts per query. At smaller scales, increasing query diversity is more effective: 4096×1 outperforms 512×8, indicating that additional distinct queries provide greater benefit than repeated sampling of the same inputs. At larger scales, however, 1024×8 slightly outperforms 8192×1, suggesting that once sufficient query diversity has been achieved, multiple rollouts can provide complementary gains. This indicates a regime in which stochastic sampling of high-quality queries becomes increasingly beneficial.

Overall, these results suggest that scaling distinct queries is critical in low-data regimes, while multiple rollouts become a useful secondary mechanism for improved performance after query diversity saturates. The optimal tradeoff between query diversity and rollout count therefore appears to be domain- and scale-dependent, and should be determined empirically.

4.4.2. Alternative Model Families

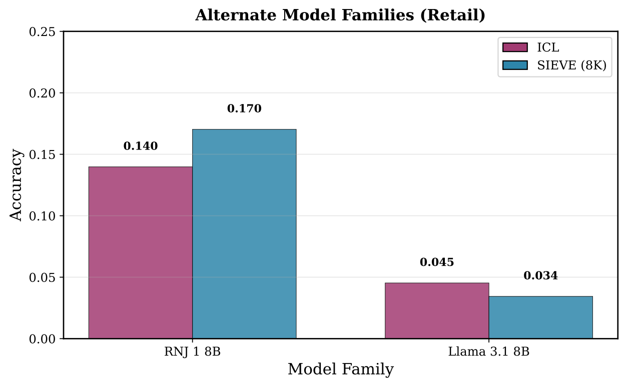

Figure 4. **SIEVE generalizes across model families.** We evaluate SIEVE on the Retail domain using alternative model families: Llama 3.1 8B and Rnj 1 8B. Results demonstrate that SIEVE consistently improves model performance across diverse architectures (8K training examples).

To verify that our method generalizes beyond the Qwen3 model family, we replicate experiments on the Retail domain using Llama 3.1 8B (Llama Team, 2024) and Rnj 1 8B (Vaswani et al., 2025a;b). Results are shown in Figure 4.

For Rnj 1 8B, SIEVE achieves 17.03% accuracy with 8K training examples, exceeding the ICL baseline of 13.98%. This demonstrates that sample-efficient parametric learning from natural language context successfully generalizes to alternative model architectures, with the trained model outperforming ICL without requiring context at inference.

However, for Llama 3.1 8B, we observe a different pattern. The ICL baseline achieves only 4.53% accuracy, and SIEVE training yields 3.44%—slightly below the ICL baseline. This reveals an important limitation: *the base model must have sufficient capability for the method to be effective*. The Llama 3.1 8B model performs extremely poorly on Retail even with context in context, suggesting it lacks the foundational reasoning abilities required for this compositional rule-application task. When the base model is too weak, it struggles both to generate high-quality synthetic data through SIEVE-GEN and to internalize the resulting training signal effectively.

This ablation highlights that SIEVE requires a model with reasonable capabilities on the target domain. For models that can leverage context in context (like Qwen3 and Rnj), our method successfully internalizes that context into weights. For models that cannot effectively use context even when provided (like Llama 3.1 8B on this task), sample-efficient parametric learning remains challenging.

Future work could explore whether curriculum learning or progressive training strategies might enable weaker models to benefit from this approach. Another line of work might also consider decoupling the synthetic data generation models from the model that is trained, stepping away from the self-distillation paradigm we focus on in this work.

5. Conclusion & Future Work

Our work introduces a method that achieves sample-efficient parametric learning from natural language. Through SIEVE and its novel synthetic data generation method SIEVE-GEN, we show that models can internalize complex reasoning knowledge using a very small number of seed query examples. Our method outperforms prior context distillation methods in sample-efficient settings. SIEVE further matches or exceeds the performance of in-context learning without requiring context at inference time. Our key insight is that for many tasks, not all context applies to every query. By decomposing the context and constructing synthetic data that only use applicable parts of it, we sample higher quality rollouts that outperform past context distillation methods, including when given access to more data or higher-quality (non-synthetic) oracle queries.

We find that parametric learning can be practical across multiple domains requiring compositional reasoning (Retail, RuleArena) and long-context memorization (MTOB). Crucially, this can be achieved without the complexities that traditionally make parametric learning challenging: many examples of input queries, large supervised datasets of expert-curated traces, or verifiers. This opens new possibilities for continual learning systems that improve persistently from natural language context in real-world settings.

Future Directions. Sample-efficient parametric learning from natural language enables several promising research directions. First, this paradigm allows for learning algorithms that adapt from rich, non-scalar context in sample-efficient ways moving beyond simple preference scores to leverage the full expressiveness of natural language. Second, expanding the suite of learning methods remains important: exploring architectural modifications (e.g., specialized memory layers), alternative training objectives beyond context distillation, and methods for managing interference between updates could further improve sample efficiency and scalability. Third, developing more challenging benchmarks that require deeper reasoning, longer-term memory, and compositional generalization will be critical for advancing parametric learning methods. Finally, studying how these sample-efficient learning methods can enable truly continual learning—where models iteratively improve from ongoing feedback—represents an exciting frontier for making LLMs more adaptive and specialized over time.

Acknowledgements

We would like to thank Alan Zhu, Andrew Qin, and Joseph E. Gonzalez for many insightful discussions about this work.

Sky Computing Lab is supported by gifts from Accenture, Amazon, AMD, Anyscale, Broadcom, Google, IBM, Intel, Intesa Sanpaolo, Lambda, Lightspeed, Mibura, NVIDIA, Samsung SDS, and SAP. We would additionally like to acknowledge Databricks for their generous compute support.

This material is based upon work supported by the National Science Foundation Grant No. CCF-2019844. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Bhargava, A., Witkowski, C., Detkov, A., and Thomson, M. Prompt baking, 2024. URL https://arxiv.org/abs/2409.13697.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Chen, A., Scheurer, J., Korbak, T., Campos, J. A., Chan, J. S., Bowman, S. R., Cho, K., and Perez, E. Improving code generation by training with natural language feedback, 2024. URL https://arxiv.org/abs/2303.16749.

Eyuboglu, S., Ehrlich, R., Arora, S., Guha, N., Zinsley, D., Liu, E., Tennien, W., Rudra, A., Zou, J., Mirhoseini, A., and Re, C. Cartridges: Lightweight and general-purpose long context representations via self-study, 2025. URL https://arxiv.org/abs/2506.06266.

Guha, E., Marten, R., Keh, S., Raoof, N., Smyrnis, G., Bansal, H., Nezhurina, M., Mercat, J., Vu, T., Sprague, Z., Suvarna, A., Feuer, B., Chen, L., Khan, Z., Frankel, E., Grover, S., Choi, C., Muennighoff, N., Su, S., Zhao, W., Yang, J., Pimpalgaonkar, S., Sharma, K., Ji, C. C.-J., Deng, Y., Pratt, S., Ramanujan, V., Saad-Falcon, J., Li, J., Dave, A., Albalak, A., Arora, K., Wulfe, B., Hegde, C., Durrett, G., Oh, S., Bansal, M., Gabriel, S., Grover, A., Chang, K.-W., Shankar, V., Gokaslan, A., Merrill, M. A., Hashimoto, T., Choi, Y., Jitsev, J., Heckel, R., Sathiamoorthy, M., Dimakis, A. G., and Schmidt, L. Openthoughts: Data recipes for reasoning models, 2025. URL https://arxiv.org/abs/2506.04178.

Hübotter, J., Lübeck, F., Behric, L., Baumann, A., Bagatella, M., Marta, D., Hakimi, I., Shenfeld, I., Buening, T. K., Guestrin, C., and Krause, A. Reinforcement learning via self-distillation, 2026. URL https://arxiv.org/abs/2601.20802.

Kujanpää, K., Marttinen, P., Valpola, H., and Ilin, A. Efficient knowledge injection in llms via self-distillation, 2025. URL https://arxiv.org/abs/2412.14964.

Lin, J., Berges, V.-P., Chen, X., Yih, W.-T., Ghosh, G., and Oğuz, B. Learning facts at scale with active reading, 2025. URL https://arxiv.org/abs/2508.09494.

Liu, J., Qiu, Z., Li, Z., Dai, Q., Yu, W., Zhu, J., Hu, M., Yang, M., Chua, T.-S., and King, I. A survey of personalized large language models: Progress and future directions, 2025. URL https://arxiv.org/abs/2502.11528.

Llama Team. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Popović, M. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the tenth workshop on statistical machine translation*, pp. 392–395, 2015.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., and Chadha, A. A systematic survey of prompt engineering in large language models: Techniques and applications, 2025. URL https://arxiv.org/abs/2402.07927.

Scheurer, J., Campos, J. A., Chan, J. S., Chen, A., Cho, K., and Perez, E. Training language models with language feedback, 2022. URL https://arxiv.org/abs/2204.14146.

Shenfeld, I., Damani, M., Hübotter, J., and Agrawal, P. Self-distillation enables continual learning, 2026. URL https://arxiv.org/abs/2601.19897.

Snell, C., Klein, D., and Zhong, R. Learning by distilling context, 2022. URL https://arxiv.org/abs/2209.15189.

Tanzer, G., Suzgun, M., Visser, E., Jurafsky, D., and Melas-Kyriazi, L. A benchmark for learning to translate a new language from one grammar book, 2024. URL https://arxiv.org/abs/2309.16575.

Upadhayayaya, R., Osti, M. R., Smith, Z., and Kottmyer, C. Efficient llm context distillation, 2025. URL https://arxiv.org/abs/2409.01930.

Vaswani, A., Callahan, M., Chaluvaraju, A., Gordić, A., Gupta, D., Jain, Y., Mansingka, D., Monk, P., Nguyen, K., Parmar, M., Pust, M., Romanski, T., Rushton, P., Shehper, A., Shivaprasad, D., Singla, S., Smith, K., Srivastava, S., Thomas, A., Tripathy, A., Vanjani, Y., Velingker, A., and Essential AI. Rnj-1, 2025a. URL https://huggingface.co/EssentialAI/rnj-1. base model release.

Vaswani, A., Callahan, M., Chaluvaraju, A., Gordić, A., Gupta, D., Jain, Y., Mansingka, D., Monk, P., Nguyen, K., Parmar, M., Pust, M., Romanski, T., Rushton, P., Shehper, A., Shivaprasad, D., Singla, S., Smith, K., Srivastava, S., Thomas, A., Tripathy, A., Vanjani, Y., Velingker, A., and Essential AI. Rnj-1-Instruct, 2025b. URL https://huggingface.co/EssentialAI/rnj-1-instruct. Instruction-tuned model release.

Xu, T., Hu, Z., Chen, L., and Li, B. Sa-mdkif: A scalable and adaptable medical domain knowledge injection framework for large language models, 2024. URL https://arxiv.org/abs/2402.00474.

Yang, Z., Band, N., Li, S., Candès, E., and Hashimoto, T. Synthetic continued pretraining, 2024. URL https://arxiv.org/abs/2409.07431.

Zhou, R., Hua, W., Pan, L., Cheng, S., Wu, X., Yu, E., and Wang, W. Y. RuleArena: A benchmark for rule-guided reasoning with LLMs in real-world scenarios, 2025. URL https://arxiv.org/abs/2412.08972.

Zhu, A., Asawa, P., Davis, J. Q., Chen, L., Hanin, B., Stoica, I., Gonzalez, J. E., and Zaharia, M. Bare: Leveraging base language models for few-shot synthetic data generation, 2025. URL https://arxiv.org/abs/2502.01697.

A. Training Details

We run all training experiments unless otherwise noted on a single node of 8xH100s. For reproducibility, we provide key non-default training parameters used in Table 3 and Table 4. Number of epochs were chosen empirically per domain based on loss convergence.

Table 3. Training Hyperparameters

Hyperparameter	**Value**
Learning Rate	1×10^{-5}
Batch Size (per device)	1
Gradient Accumulation	8
Effective Batch Size	64
Temperature (τ)	1.0
Warmup Steps	50
Top-K Tokens	100
Max Sequence Length	16,384
Optimizer	AdamW
DeepSpeed Config	ZeRO-3
Number of GPUs	8

Table 4. Dataset-Specific Hyperparameters

Hyperparameter	**Retail**	**NBA**	**MTOB**
Epochs	2	2	5

B. Extended Domain Discussion

We provide further information regarding the domains we evaluate on in the paper here.

B.1. Retail

Retail Domain. The Retail domain involves calculating final prices after applying complex discount rules to shopping carts. Each query is programmatically generated with a subset of the following components:

- **Customer Types:** 6 types (student, senior citizen, veteran, employee, teacher, regular)
- **Product Categories:** 8 categories (electronics, clothing, books, food, home, sports, beauty, health)
- **Promo Codes:** 7 promotional codes (SAVE20, WELCOME10, STUDENT15, HOLIDAY30, NEWBIE5, TEACHER10, BULK10)
- **Membership Tiers:** 3 tiers (bronze: 1+ years, silver: 3+ years, gold: 5+ years)

Each query presents a customer profile (type, membership years), shopping cart (1-5 items with categories, prices, quantities), and optional promo code. The model must calculate the final price after applying all applicable discount rules, which may stack multiplicatively or apply to specific categories. Ground truth is computed programmatically using the rule engine.

Retail Example Query

Calculate the final price for the following customer purchase after applying all applicable discount rules.

Customer Profile:
- Type: senior
- Membership years: 4

Shopping Cart:
- Shoes (apparel): $85.00 x 2
- Jacket (apparel): $60.00 x 1
- Coffee Maker (home): $45.00 x 1

Promo code: None

IMPORTANT: Apply discounts in this exact order to the running total:
1. Category-specific percentage discounts (apply only the highest discount per category to each category's subtotal)
2. Total purchase percentage discounts (apply only the highest total discount to the remaining amount after step 1)
3. Fixed amount discounts (subtract from the remaining amount after step 2, sum all applicable fixed discounts)

Note: Each discount applies to the current running total, not the original price.

Retail Rules

Discount Rules:
- If customer is a student AND total spend is at least \$50, apply 10% discount to total purchase
- If customer is a senior citizen AND total spend is at least \$50, apply 15% discount to total purchase
- If customer is a employee AND total spend is at least \$50, apply 20% discount to total purchase
- If customer is a teacher AND total spend is at least \$50, apply 10% discount to total purchase
- If customer is a student AND cart contains electronics, apply 15% discount on electronics items only
- If customer is a student AND cart contains books, apply 15% discount on books items only
- If customer is a senior citizen AND cart contains food, apply 20% discount on food items only
- If customer is a veteran AND cart contains electronics, apply 5% discount on electronics items only
- If customer is a employee AND cart contains home, apply 25% discount on home items only
- If customer is a teacher AND cart contains books, apply 15% discount on books items only
- If promo code is ’SAVE20’ AND total spend is at least \$100, apply 20% discount to total purchase
- If promo code is ’WELCOME10’, apply \$10 fixed discount
- If customer is a student AND promo code is ’STUDENT15’, apply 15% discount to total purchase
- If promo code is ’HOLIDAY30’ AND total spend is at least \$150, apply 30% discount to total purchase
- If promo code is ’NEWBIE5’, apply \$5 fixed discount
- If customer is a teacher AND promo code is ’TEACHER10’, apply 10% discount to total purchase
- If promo code is ’BULK10’ AND total spend is at least \$200, apply 10% discount to total purchase
- If total spend is at least \$150, apply 5% discount to total purchase
- If total spend is at least \$200, apply 10% discount to total purchase
- If cart contains electronics items AND total electronics spend is \$300 or greater, apply 10% discount on electronics items only
- If cart contains clothing items AND total clothing spend is \$75 or greater, apply 10% discount on clothing items only
- If cart contains books items AND total books spend is \$45 or greater, apply 10% discount on books items only
- If cart contains food items AND total food spend is \$30 or greater, apply 10% discount on food items only
- If cart contains home items AND total home spend is \$60 or greater, apply 10% discount on home items only
- If cart contains sports items AND total sports spend is \$90 or greater, apply 10% discount on sports items only
- If cart contains beauty items AND total beauty spend is \$52 or greater, apply 10% discount on beauty items only
- If cart contains health items AND total health spend is \$75 or greater, apply 10% discount on health items only
- If customer has been a member for 1 or more years, apply 5% discount to total purchase
- If customer has been a member for 3 or more years, apply 10% discount to total purchase
- If customer has been a member for 5 or more years, apply 15% discount to total purchase

Some of the example context units that the natural language context may decompose into for this domain include:

```
["If customer is a student AND total spend is at least $50, apply 10% discount to total
   purchase",
 "If customer is a senior citizen AND total spend is at least $50, apply 15% discount to
    total purchase",
 "If customer is an employee AND total spend is at least $50, apply 20% discount to total
    purchase",
 ...]
```

B.2. RuleArena (NBA)

Quoting the original description of this domain from the benchmark: ”It requires LLMs to determine whether one or more specified transactions are allowed. The regulations are extracted from the 2023 NBA Collective Bargaining Agreements (CBA) and excerpt from the NBA Constitution and ByLaws. Complexity arises from the numerous factors influencing transaction eligibility, including the player’s contract value, salary-matching constraints, and the specific transaction date. LLMs must accurately identify and apply the relevant rules from the agreement to determine whether a given transaction can proceed” (Zhou et al., 2025).

NBA Example Query

QUESTION:
Team Situations:
Team A has a team salary of $130,000,000.
Team B has a team salary of $135,000,000.
Team C has a team salary of $98,000,000.

Player Situations: Player A was the 33th second-round pick of Team B in 2022 NBA draft when he was 20 years old.
Player A signed a 2-year contract with Team B providing the minimum salary.
Player B was the 44th second-round pick of Team C in 2015 NBA draft when he was 22 years old.
Player B signed a 3-year contract with Team B providing annual salary $31,000,000, 5% increase per year in 2021 Cap Year.
Player C was the 21th first-round pick of Team C in 2016 NBA draft when he was 22 years old.
Player C signed a 2-year contract with Team B providing annual salary $7,000,000, 5% increase per year in 2022 Cap Year.
Player D was the 15th first-round pick of Team D in 2017 NBA draft when he was 22 years old.
Player D signed a 3-year contract with Team D providing annual salary $10,000,000, 5% increase per year in 2021 Cap Year.

Operations: A. Team B provides a qualifying offer for Player A.
B. Team A provides Player A with an offer sheet - a 2-year contract providing annual salary $10,000,000 in the first Salary Cap Year (2024-2025), 5% increase per year for the first two Salary Cap Years.
C. Team B matches the offer by team A.
D. Team B signs a 2-year contract with Player D providing annual salary $5,000,000 in the first Salary Cap Year (2024-2025), 5% increase per year.
E. Team C signs a 2-year contract with Player C providing annual salary $7,000,000 in the first Salary Cap Year (2024-2025), 5% increase per year.

The full set of rules for this domain is too large to include in the paper, thus we direct readers to the official repository here: `https://github.com/SkyRiver-2000/RuleArena/blob/main/nba/reference_rules.txt`.

Some of the example context units that the natural language context may decompose into for this domain include:

```
["the Team shall be prohibited from trading (either conditionally or unconditionally) its
    first round draft pick in the first NBA Draft that occurs following the seventh
    Season that follows the Season occurring within such Salary Cap Year;", "A Team may
    use the Taxpayer Mid-Level Salary Exception to sign one (1) or more Player Contracts
    during each Salary Cap Year not to exceed two (2) Seasons in length, that, in the
    aggregate, provide for Salaries and Unlikely Bonuses in the first Salary Cap Year
    totaling up to the amounts set forth below, provided that the Team's Team Salary
    immediately following the Team's use of such Exception exceeds the First Apron
    Level:", "Player Contracts signed pursuant to the Mid-Level Salary Exception for Room
    Teams may provide for annual increases and decreases in Salary and Unlikely Bonuses
    in accordance with Section 5(a)(1) above.", ...]
```

B.3. MTOB

The Machine Translation from One Book (MTOB) benchmark (Tanzer et al., 2024) tests a model's ability to translate between English and Kalamang, an extremely low-resource language for which very few resources exist on the internet (essentially a single grammar book) that they curate into the data provided in this benchmark.

We make use of their medium sized version of this grammar book for our natural language context in this task, which comprises of approximately 50K tokens that include a grammar table and numerous word mappings. We also include a corpus of approximately 375 paired English-Kalamang sentences. We use RoPE scaling to extend the model's context window for ICL baseline evaluations.

We refrain from including example queries and decompositions of the natural language corpus here as it is both too long but more importantly not recommended to prevent leakage in the benchmark. At a high level though, in this setting the context units include both groups of word translations, individual grammar rules, and example sentences.

C. Synthetic Data Prompts

Below we provide the general prompts used for all domains in SIEVE (that fit within context).

C.1. Decomposition Prompt

Decomposition Prompt

Break down the following feedback/guidelines/knowledge into atomic, independent items.

Each atomic item should:
1. Express a single, self-contained rule, fact, definition, or example
2. Be evaluable independently (can determine if it applies without needing other items)
3. Preserve the exact meaning and wording from the original

Content:
{chunk}

Output each atomic item separated by "###" on its own line.
For items with sub-bullets or multiple lines, include all lines as part of that item. Do NOT number or label items. Do not add explanations or commentary.

Example format:
First item content here
###
Second item content here
###
Third item content here

Do not group multiple concepts together. Each item should be atomic.

C.2. Seed Context Selection Prompt (Base Model)

Seed Context Selection Prompt

Task: Select 3-5 guidelines from the natural language feedback that could apply together to a single scenario/question.

Guidelines:
{feedback}
{examples section}
Selected guidelines:
-

C.3. Query Generation Prompt

Query Generation Prompt

Generate a realistic question where the following feedback/guidelines/knowledge would apply:

{selected feedback}

Instructions:
1. Create a specific question where the information applies, similar to the format of the examples below
2. Make it realistic
3. Include all necessary details
4. Output ONLY the question, nothing else

{examples section}

Question: