Beyond Semantic Similarity: Task-Aware Anti-Shortcut Example Retrieval for In-Context Learning

Anonymous ACL submission

Abstract

In-context learning (ICL) is highly sensitive to the choice of demonstrations. Most existing retrieval methods select demonstrations mainly by semantic similarity, but this strategy has two important weaknesses: it does not explicitly encode task information, and it often prefers examples with strong lexical overlap, which can encourage shortcut-based predictions. We propose **T**ask-aware **A**nti-shortcut **E**xample **R**etrieval (TAER), a simple two-stage retrieval framework for ICL. TAER first prepends a short task description to the test input and candidate examples, and retrieves task-relevant examples according to task-aware similarity. It then reranks these candidates using shallow-layer surface-feature similarity and keeps the least superficially similar examples, reducing the chance that the model relies on shallow correlations. Experiments on 15 natural language understanding and generation datasets, using LLMs from multiple families and scales, show that TAER consistently outperforms strong retrieval baselines. Additional ablations and robustness analyses further support the value of combining task-aware retrieval with anti-shortcut filtering.

1 Introduction

Large language models (LLMs) exhibit strong few-shot capabilities through in-context learning (ICL), where a model solves a new task from only a small set of input-output examples in the prompt (Brown et al., 2020; Li et al., 2024). Because ICL does not require parameter updates, it has become a practical and widely used adaptation paradigm for LLMs across many natural language processing tasks (Guo et al., 2024; Duan et al., 2024).

Despite its simplicity, ICL is notoriously sensitive to the demonstrations shown to the model (Liu et al., 2022; Guo et al., 2024; Min et al., 2022). As a result, demonstration retrieval has become a central research question. A common strategy is to retrieve examples whose inputs are semantically similar to the test input. This family of methods is attractive because it is simple, efficient, and often better than random selection. However, semantic similarity alone is not always the right signal for ICL.

First, semantic similarity does not explicitly encode the target task. Two inputs may be semantically close while supporting different decision boundaries or output patterns, which means that highly similar examples are not necessarily the most instructive demonstrations for the current task. Second, semantic retrieval often favors strong lexical overlap (Kaster et al., 2021). This is potentially problematic because LLMs are known to exploit superficial cues and shortcuts when they are available (Tang et al., 2023; Wang et al., 2023). If retrieved examples repeatedly share salient surface forms with the test input, the model may over-rely on shallow correlations instead of inferring the intended task pattern.

Throughout this paper, we use the term *anti-shortcut* in an operational rather than causal sense. Concretely, we target a retrieval failure mode in which demonstrations are task-relevant but overly redundant at the lexical or shallow structural level, making them easy to match through superficial cues. We do not claim to characterize all forms of shortcut behavior in LLMs; instead, we introduce a lightweight retrieval-time proxy for reducing shortcut-prone overlap in the final prompt.

To address these issues, we propose **T**ask-aware **A**nti-shortcut **E**xample **R**etrieval (TAER), a simple two-stage retrieval method for ICL. In the first

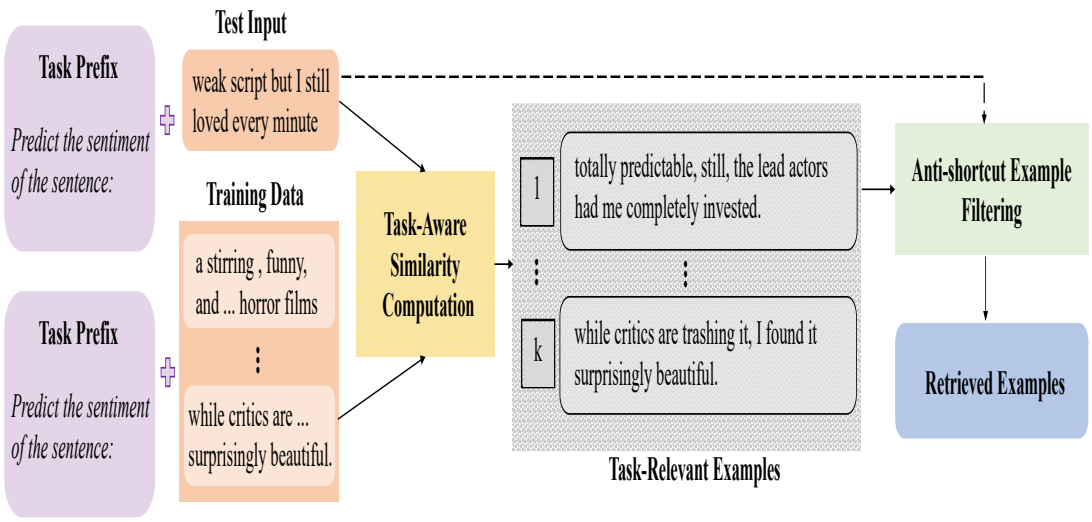

Figure 1: Overall framework of TAER. This process is demonstrated using sentiment analysis task as an example.

stage, we prepend a task-aware prefix (e.g., "Predict the sentiment of the sentence:") to both the test input and each candidate example, and compute task-aware similarity in this task-conditioned space. This stage favors examples that are not only semantically related but also aligned with the desired task behavior. In the second stage, we operate on the top-k task-relevant candidates and prefer examples with low surface-feature similarity to the test input. This filtering step is designed to reduce shortcut opportunities while preserving task relevance.

TAER is intentionally lightweight: it does not require task-specific model training, complex search procedures, or repeated LLM feedback. Instead, it combines two complementary inductive biases for demonstration retrieval: retrieve examples that are relevant to the task, and avoid examples that are too easy to match through superficial overlap.

Our main contributions are summarized as follows:

- We propose TAER, a simple retrieval-and-reranking framework that first enforces task relevance and then downweights shortcut-prone surface redundancy in the final prompt.
- We formalize an operational view of anti-shortcut retrieval for ICL: among task-relevant candidates, shallow-layer similarity provides a simple and efficient proxy for identifying near-duplicate demonstrations.
- Across 15 NLU and NLG datasets and multiple LLM families, TAER consistently outperforms strong baselines. In particular, on Qwen3-8B, TAER improves over the strongest baseline by 5.35 points in the 5-shot setting.

2 Related Work

In-context example selection. Since GPT-3, a large body of work has studied why some demonstrations are more effective than others for ICL (Brown et al., 2020; Liu et al., 2022). A major line of work retrieves examples according to input similarity. Early approaches rank candidates by semantic similarity computed from pretrained encoders (Liu et al., 2022). Later work augments this idea with representativeness, diversity, or coverage constraints (Su et al., 2022; Kapuriya et al., 2025; Gupta et al., 2023). Other methods use iterative search or evaluation-based strategies to identify stronger demonstrations (Wu et al., 2024; Qin et al., 2024; Fang et al., 2025). While these methods can be effective, many either still depend on semantic similarity as the core signal or require more expensive search procedures.

Beyond semantic similarity. Several recent studies suggest that similarity alone is not sufficient for robust example selection. Some methods adapt retrieval to specific problem structures, such as syntactic similarity for machine translation (Ji et al., 2024) or reasoning-graph similarity for reasoning tasks (Lin et al., 2025). At the same time, prior analyses show that commonly used similarity measures may overemphasize lexical overlap (Kaster et al., 2021), and that LLMs often exploit superficial cues when available (Tang et al., 2023). These

observations motivate retrieval objectives that go beyond plain semantic closeness. TAER follows this direction by combining task-conditioned relevance with an explicit anti-shortcut filtering step, while remaining simple and task-agnostic.

TAER is also intentionally close to lightweight retrieval engineering rather than heavy prompt optimization. Unlike generic diversity-based selection, its second stage is query-specific and is applied only after task-aware retrieval has narrowed the candidate set. This design targets a concrete failure mode, namely misleading near-duplicates among otherwise relevant examples, without requiring additional LLM calls or task-specific training.

Our motivation is also related to recent analyses of how demonstrations influence ICL behavior. For example, prior work shows that demonstrations can provide strong label and format priors even when their labels are corrupted (Min et al., 2022), and other work explicitly studies demonstration shortcuts and attempts to correct them at prediction time rather than retrieval time (Jang et al., 2024). TAER is complementary to this line of work because it intervenes earlier, at demonstration selection time.

3 Method

Figure 1 illustrates TAER. Given a test input q and a candidate pool $\mathcal{T} = \{t_1, \ldots, t_m\}$, TAER returns a final demonstration set of size n in two stages: (1) task-aware retrieval, which identifies task-relevant candidates, and (2) anti-shortcut filtering, which removes candidates that are overly similar at the surface level.

3.1 Task-aware Similarity Computation

The goal of the first stage is to retrieve examples that are aligned with the task behavior required by the test input, rather than merely close in raw semantic space.

Step 1: Task-aware Example Construction. We first define a short task description p such as "Predict the sentiment of the sentence:". We prepend p to the test input and every candidate example:

$$\mathbf{q}_{\text{task}} = [p; q], \qquad \mathbf{t}_i^{\text{task}} = [p; t_i]. \tag{1}$$

This simple construction places both the query and the candidates in a task-conditioned input space.

Step 2: Task-aware Similarity Computation and Selection. We encode $\mathbf{q}_{\text{task}}$ and each $\mathbf{t}_i^{\text{task}}$ with a pretrained encoder (RoBERTa (Liu et al., 2019) in our implementation) and use the `[CLS]` hidden state as the sequence representation:

$$\mathbf{h}_q = f(\mathbf{q}_{\text{task}}), \qquad \mathbf{h}_i = f(\mathbf{t}_i^{\text{task}}). \tag{2}$$

We then compute task-aware similarity with cosine similarity:

$$s_i^{\text{task}} = \cos(\mathbf{h}_q, \mathbf{h}_i). \tag{3}$$

Finally, we keep the top-k candidates with the highest s_i^{task} values. In our implementation, $k = 2n$, where n is the number of final demonstrations.

3.2 Anti-shortcut Example Filtering

The second stage operates on the top-k task-relevant candidates and aims to remove examples that are too easy to match through superficial overlap.

Operationally, we treat *shortcut-prone redundancy* as high similarity in shallow encoder representations among candidates that have already passed the task-aware stage. This is a practical proxy rather than a complete definition of shortcut behavior, but it directly targets the near-duplicate retrieval failure mode emphasized in our motivation.

Step 1: Superficial Feature Representation. We compute surface-oriented representations from the original inputs, *without* task prefixes. Following prior work showing that lower transformer layers capture shallow linguistic information (Tenney et al., 2019), we use the first three layers of the same pretrained encoder and extract the hidden representation from the third layer:

$$\mathbf{s}_q = g(q), \qquad \mathbf{s}_i = g(t_i), \tag{4}$$

where $g(\cdot)$ denotes the shallow encoder.

Step 2: Anti-shortcut Selection. For each candidate in the top-k set, we compute a surface-similarity score:

$$s_i^{\text{surface}} = \cos(\mathbf{s}_q, \mathbf{s}_i). \tag{5}$$

We then select the final n examples with the *lowest* surface similarity scores. Formally, if $\mathcal{E}_k$ is the top-k set returned by the first stage, the final demonstration set is

$$\mathcal{D}_n = \underset{\substack{\mathcal{D} \subseteq \mathcal{E}_k \\ |\mathcal{D}| = n}}{\arg\min} \sum_{t_i \in \mathcal{D}} s_i^{\text{surface}}. \tag{6}$$

This design encourages the final demonstrations to remain task-relevant while reducing opportunities for shortcut-based matching.

Table 1: Experimental results on 15 datasets. "NLU Avg." and "NLG Avg." represent the average performance on NLU datasets and NLG datasets, respectively. "Avg." denotes the average across both categories. Zero-shot results are reported in the 3-shot setting. † indicates statistical significance with $p < 0.05$ compared with Top-k and DICL.

Model	Method	3-shot			5-shot		
		NLU Avg.	NLG Avg.	Avg.	NLU Avg.	NLG Avg.	Avg.
Qwen3-8B	Zero-shot	62.45	52.88	57.66	-	-	-
	Random	66.26	55.21	60.73	66.83	55.56	61.20
	Top-k	67.82	55.89	61.85	68.71	56.89	62.80
	DICL	69.13	56.21	62.67	69.47	57.01	63.24
	TAER (ours)	**74.26**†	**60.73**†	**67.50**†	**75.54**†	**61.64**†	**68.59**†
Qwen3-32B	Zero-shot	76.31	67.34	71.83	-	-	-
	Random	78.21	68.64	73.42	78.67	69.21	73.94
	Top-k	79.15	69.01	74.08	79.83	69.22	74.53
	DICL	79.86	69.23	74.55	80.07	70.33	75.20
	TAER (ours)	**82.31**†	**71.38**†	**76.84**†	**82.94**†	**72.10**†	**77.52**†
Qwen2.5-72B-Instruct	Zero-shot	73.31	64.26	68.78	-	-	-
	Random	75.65	65.89	70.77	75.57	66.21	70.89
	Top-k	76.84	66.14	71.49	76.97	66.83	71.90
	DICL	77.34	67.26	72.30	77.85	68.02	72.94
	TAER (ours)	**79.65**†	**69.83**†	**74.74**†	**80.23**†	**70.64**†	**75.44**†
Llama-3.3-70B-Instruct	Zero-shot	74.64	65.67	70.16	-	-	-
	Random	76.63	67.04	71.84	76.80	67.59	72.19
	Top-k	77.92	67.89	72.91	77.74	68.23	72.98
	DICL	78.62	68.42	73.52	79.04	69.58	74.31
	TAER (ours)	**80.91**†	**69.88**†	**75.39**†	**81.59**†	**70.44**†	**76.02**†

4 Experiments

4.1 Settings

Datasets. We evaluate on 15 datasets spanning natural language understanding (NLU) and natural language generation (NLG). The NLU portion contains all 9 GLUE datasets (Wang et al., 2019); the NLG portion contains CNN/DM, XSum, NYT, WebNLG, WCEP, and ToTTo. We follow the standard evaluation metric for each dataset. Higher scores are better in all cases.

Baselines. We compare TAER with four representative baselines:

(1) **Zero-shot**, which uses only the task instruction;

(2) **Random**, which samples demonstrations uniformly from the training set;

(3) **Top-k** (Liu et al., 2022), which retrieves demonstrations with the highest semantic similarity to the test input; and

(4) **DICL** (Kapuriya et al., 2025), a strong retrieval baseline that combines local relevance with diversity-aware selection.

ICL setting. Following prior work (Liu et al., 2022), we study both 3-shot and 5-shot ICL. For TAER, we first retrieve the top-k task-relevant candidates and then keep the final n examples after anti-shortcut filtering, with $k = 2n$.

Models and implementation. We evaluate four LLMs from different families and scales: Qwen3-8B, Qwen3-32B, Qwen2.5-72B-Instruct, and Llama-3.3-70B-Instruct. For retrieval, we use RoBERTa to compute both task-aware and shallow-layer surface representations. All experiments are run with PyTorch 2.6.0 and vLLM 0.8.5 on Ubuntu 22.04.2 using 8 NVIDIA H100 80G GPUs.

Prompt construction and candidate pool. For each benchmark, we use the same task instruction and label verbalization across all baselines so that the only changing factor is demonstration selection. Candidate examples are drawn from the task-specific training split. For classification tasks, each candidate is represented by its input and gold label; for generation tasks, the candidate additionally includes the target summary or structured output. This unified construction keeps the comparison between Random, Top-k, DICL, and TAER fair across heterogeneous tasks.

Hyperparameters. Unless otherwise stated, the final number of demonstrations is $n \in \{3, 5\}$, and

the first-stage task-aware retriever keeps $k = 2n$ candidates for the anti-shortcut reranking stage. Cosine similarity is used in both stages. We compute the task-aware representation from the final hidden layer of the encoder, and the surface-oriented representation from a shallow layer, following the intuition that lower layers emphasize lexical and syntactic information. All retrieval hyperparameters are selected once on development tasks and then reused across the full evaluation suite.

4.2 Main Results

Table 1 reports the main results. We make three observations.

First, all ICL-based methods outperform zero-shot prompting, and the gains are especially large for smaller models. This confirms that high-quality demonstrations remain a key factor in practical ICL.

Second, retrieval-based methods consistently outperform random selection, which further highlights the importance of choosing informative examples rather than relying on arbitrary prompts.

Third, TAER achieves the best overall performance across all four LLMs and both shot settings. The gains are consistent on both NLU and NLG averages, which suggests that the method is not tied to a single task type. In particular, for Qwen3-8B in the 5-shot setting, TAER improves over the strongest baseline by **5.35** points. These results support our claim that combining task-aware retrieval with anti-shortcut filtering produces better demonstrations than semantic retrieval alone.

The gains are also remarkably stable across model scale. For larger models such as Qwen2.5-72B-Instruct and Llama-3.3-70B-Instruct, the absolute margin over strong retrieval baselines is smaller than for Qwen3-8B, but the ranking remains unchanged. This is an important observation because it indicates that TAER is not merely compensating for weak base models. Instead, it improves the quality of the retrieved evidence itself, which remains useful even when the downstream LLM already has strong general reasoning and instruction-following ability.

Another notable trend is that the improvement is visible on both NLU and NLG subsets. This suggests that TAER is not over-specialized to discrete-label classification. The first stage mainly improves task relevance, while the second stage removes highly overlapping demonstrations that may encourage brittle copying behavior. These two biases

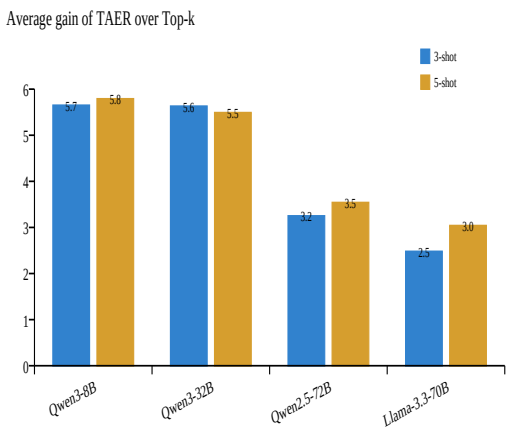

Figure 2: Average gain of TAER over Top-k across the four evaluated LLMs. Improvements remain positive in both 3-shot and 5-shot settings, with the largest margins on smaller models.

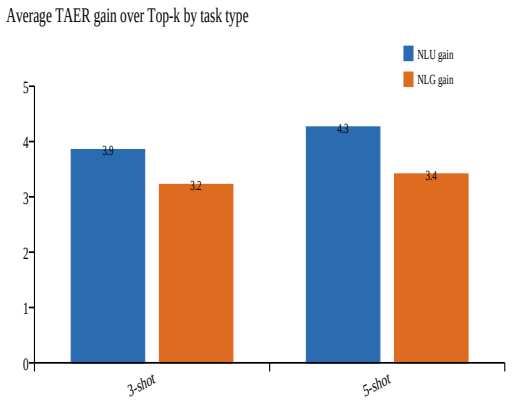

Figure 3: Average TAER gain over Top-k on NLU and NLG subsets. The gains are visible on both task families, with slightly larger margins on NLU.

are beneficial not only when the output is a class label, but also when the output requires generation conditioned on examples, such as summarization or data-to-text tasks.

Figure 2 summarizes the improvement over the standard Top-k retriever across model scales. The largest gains appear on Qwen3-8B and Qwen3-32B, where the downstream model is strong enough to benefit from better demonstrations but still vulnerable to shortcut-heavy retrieval. The gains remain positive even for the strongest evaluated models, indicating that TAER improves the prompt evidence itself rather than merely compensating for weak decoders.

Figure 3 further separates the improvements by task family. TAER improves both NLU and NLG, which is important because these two settings stress

Table 2: Per-model gains of TAER over Top-k on NLU and NLG subsets.

Model	ΔNLU-3	ΔNLG-3	ΔNLU-5	ΔNLG-5
Qwen3-8B	6.44	4.84	6.83	4.75
Qwen3-32B	3.16	2.37	3.11	2.88
Qwen2.5-72B	2.81	3.69	3.26	3.81
Llama-3.3-70B	2.99	1.99	3.85	2.21

different aspects of demonstration usefulness. The somewhat larger gains on NLU suggest that reducing shortcut-friendly overlap is especially helpful when label decisions are sensitive to fine-grained distinctions, while the consistent gains on NLG indicate that task-aware retrieval still improves generative prompts.

The per-model breakdown in Table 2 also shows that the balance between NLU and NLG gains varies by model family. Qwen3 models benefit more on NLU, while Qwen2.5-72B shows slightly larger gains on NLG. This suggests that TAER interacts with the inductive biases of the downstream LLM rather than producing a single fixed effect pattern.

Table 3 makes the improvement pattern more explicit. TAER delivers its largest absolute gains on Qwen3-8B, but the effect is not limited to smaller models. Even on 70B-scale systems, the method still produces consistent gains over both Top-k and DICL, which suggests that the retrieval stage remains a meaningful bottleneck even when the decoder itself is already very capable.

4.3 Ablation Study

We conduct ablations to isolate the effect of each component. Table 4 shows that removing the task-aware prefix substantially degrades performance in both the 3-shot and 5-shot settings, which confirms that task conditioning improves the quality of retrieved examples. Removing the anti-shortcut stage also leads to clear drops, indicating that the second-stage reranking contributes independently beyond task-aware retrieval. Together, these results show that TAER benefits from both components rather than from either one in isolation.

The ablation trends are also informative from a mechanism perspective. Removing the task-aware prefix hurts more consistently on NLG datasets, where the retriever must match not only topical content but also output behavior under the task instruction. By contrast, removing anti-shortcut filtering tends to harm both NLU and NLG, suggesting that superficial redundancy is a broad failure mode rather than a dataset-specific artifact. The two modules therefore solve different problems: the first aligns the retriever with task intent, while the second prevents the final prompt from collapsing into a set of near-duplicates.

Additional analyses on task-prefix variants and grouped scale/shot effects are deferred to Appendix A to keep the main paper focused on the core retrieval decomposition.

5 Analysis

5.1 Does TAER target shortcut-prone overlap?

The core motivation of TAER is that better demonstrations should be both task-relevant and less vulnerable to shortcut matching. In our manual inspection, standard semantic retrieval often returns examples with highly overlapping topical words, entity names, or sentiment-bearing expressions. Such examples can be useful when the task truly depends on these surface cues, but they can also encourage brittle copying behavior when the target output depends on a subtler task-specific pattern. TAER reduces this risk by separating the retrieval problem into two steps: relevance is enforced first, and surface redundancy is controlled only after a candidate set has already been restricted to task-relevant examples.

Beyond these aggregate signals, we expect the clearest gains to appear on instances where the nearest semantic neighbors share many topical words with the test input but express a different output pattern. In such cases, a purely similarity-based retriever can surface demonstrations that look close at the lexical level yet provide misleading supervision. TAER is designed to avoid exactly this failure mode: the task-aware stage keeps examples that remain relevant under the task instruction, and the second stage favors candidates whose usefulness does not come from superficial string overlap alone.

This interpretation should be read narrowly. TAER does not claim to detect shortcut reasoning in a fully general sense; rather, it controls a concrete retrieval artifact that is plausibly shortcut-inducing in practice. We view this narrower framing as a strength of the method: it keeps the scientific claim specific, testable, and compatible with lightweight deployment constraints.

Table 3: Average gains of TAER over Top-k and DICL on the overall score. Positive values indicate improvements.

Model	ΔTop-k (3-shot)	ΔDICL (3-shot)	ΔTop-k (5-shot)	ΔDICL (5-shot)
Qwen3-8B	5.65	4.83	5.79	5.35
Qwen3-32B	2.76	2.29	2.99	2.32
Qwen2.5-72B-Instruct	3.25	2.44	3.54	2.50
Llama-3.3-70B-Instruct	2.48	1.87	3.04	1.71

Table 4: Ablation study. "Without task-aware prefix" removes task conditioning and reduces the first stage to standard semantic retrieval. "Without anti-shortcut filtering" directly uses the task-relevant examples returned by the first stage. Scores are averaged across the four LLMs.

Method	**3-shot Avg.**	**5-shot Avg.**
TAER	**73.62**	**74.39**
w/o task-aware prefix	71.58	72.96
w/o anti-shortcut filtering	72.58	72.89

5.2 Qualitative Case Analysis

A representative example arises in sentiment analysis when the test sentence and a candidate demonstration both contain strong opinion words but differ in compositional structure, such as negation or contrastive conjunctions. A semantic retriever may rank the candidate highly because of lexical and topical similarity, even though its label pattern is misleading for the test instance. In contrast, TAER is more likely to retain demonstrations that are aligned with the task-conditioned decision boundary while filtering out those whose apparent relevance comes mostly from surface overlap.

We observe a similar pattern in generation tasks. In summarization and data-to-text benchmarks, purely similarity-based retrieval often favors candidates that share named entities or salient content words with the input. While this can improve factual alignment in some cases, it can also bias the model toward copying local lexical material instead of learning the intended structural correspondence between the input and output. TAER mitigates this effect by preferring examples that remain informative without being near-duplicates at the surface level.

5.3 Discussion

TAER can be viewed as a simple decomposition of the retrieval problem: the first stage asks *which examples are relevant to the task*, while the second asks *which of those examples are least likely to trigger shortcut behavior*. This decomposition is useful because it separates two objectives that are often entangled in a single similarity score. Our results suggest that explicitly balancing them leads to better demonstrations across both understanding and generation tasks.

This decomposition also makes TAER easy to extend. For example, the task-aware stage can be replaced by stronger instruction-aware encoders, and the anti-shortcut stage can incorporate alternative surface-bias measures such as token overlap, syntactic similarity, or shallow n-gram features. The current formulation is intentionally lightweight, but the framework is general enough to serve as a plug-in reranking module for more sophisticated retrievers.

Another practical implication is that TAER offers a favorable trade-off between retrieval quality and engineering complexity. Methods that directly optimize prompts through search or iterative LLM feedback can be effective, but they introduce additional latency and make the retrieval pipeline harder to maintain. TAER instead injects two simple biases into a conventional retrieval stack, which preserves efficiency while still producing meaningful gains. This makes it attractive in production settings where prompt construction must remain deterministic and fast.

The comparison in Figure 4 is also revealing because DICL is already a diversity-aware retrieval baseline. TAER still improves over DICL on every tested model, which suggests that the contribution of anti-shortcut filtering is not reducible to generic diversity alone. Instead, explicitly discouraging surface-level redundancy after task-aware retrieval appears to offer an additional and complementary benefit.

5.4 Retrieval Cost Analysis

TAER also remains lightweight from a systems perspective. Relative to Top-k retrieval, the first stage has the same dominant cost: computing task-aware representations for the query and candidate pool. The second stage only reranks the top-k candidates, where $k = 2n$ in our implementation and n is at

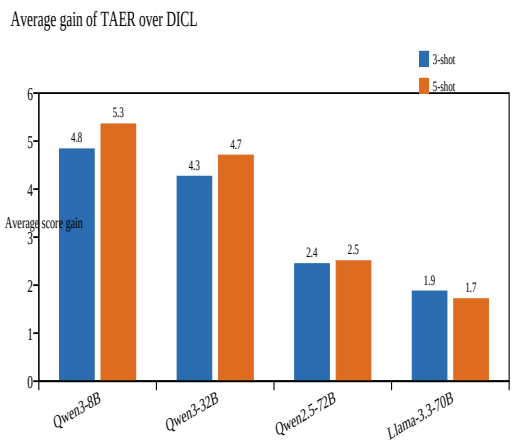

Figure 4: Average gain of TAER over DICL. The margins are smaller than those over Top-k, but remain positive across all model families and both shot settings.

most the final demonstration count. Since $k \ll m$ for a candidate pool of size m, this reranking step introduces only a small additional overhead compared with full-pool retrieval.

This complexity profile is practically important. More aggressive prompt-search methods can improve ICL, but they often require repeated LLM calls or iterative prompt evaluation. TAER instead keeps all expensive operations in the encoder-based retrieval stage and restricts the anti-shortcut decision to a small candidate subset. As a result, it offers a good trade-off between retrieval quality and deployment simplicity, especially in pipelines where prompts must be built online for many user queries. We also observe that the gains are typically larger on smaller models and remain strong in the 5-shot setting, which is consistent with the intuition that redundant demonstrations become more harmful as prompt budget grows. Detailed grouped results are reported in Appendix A.

5.5 Boundary Cases

TAER is not expected to help equally on every instance. When the target task truly depends on strong lexical anchoring, such as copy-heavy generation or slot filling with very stable surface forms, aggressively discouraging overlap may remove demonstrations that are genuinely useful. Conversely, when the candidate pool is already small or homogeneous, the benefit of anti-shortcut filtering can be limited because there is little meaningful surface-level variation to exploit. These boundary cases suggest that future work could adapt the strength of the second stage dynamically based on the diversity of the retrieved candidate set.

6 Conclusion

In this paper, we presented TAER, a novel, efficient, and theoretically grounded two-stage retrieval framework specifically engineered for optimal in-context example selection. Under this proposed paradigm, TAER first retrieves highly task-relevant demonstrations within a task-conditioned representation space, and subsequently dynamically filters out candidate examples that exhibit excessive, confounding surface-level similarity to the target test input. This methodical dual-stage design effectively and concurrently addresses two pervasive weaknesses inherent in standard semantic retrieval methodologies: the structural insufficiency of generalized task awareness and the chronic over-reliance on superficial, distractor-heavy lexical overlap. Rigorous evaluations spanning 15 diverse datasets and encompassing multiple prominent large language model families thoroughly demonstrate that TAER yields consistent, statistically significant performance enhancements over several highly competitive baselines.

More broadly, our empirical achievements and architectural design principles offer a fresh perspective that reframes the community's collective approach to contextual data management in downstream scenarios. By isolating task relevance from surface-level alignments, our framework establishes a more reliable, precise, and traceable pathway for contextual evidence generation. Ultimately, we hope this established framework serves as a foundational and practically useful direction for subsequent research dedicated to constructing more robust, controllable, and sample-efficient demonstration retrieval methods for advanced in-context learning applications.

7 Limitations

TAER has several limitations. First, it still relies on an externally specified task description, and although our prefix-robustness results are encouraging, performance may vary when task descriptions are ambiguous or poorly written. Second, the anti-shortcut stage is motivated by shallow-layer similarity as a proxy for surface overlap; this proxy is intuitive but not a perfect measurement of shortcut behavior. Third, TAER introduces an additional reranking step compared with standard Top-k retrieval, which increases retrieval-time computation.

Finally, while the method performs consistently across 15 datasets, we have not yet tested it on multimodal ICL or on tasks with highly structured outputs such as code generation.

Another limitation is that our current analysis is still aggregate. Although the average gains are consistent, we do not yet provide large-scale instance-level evidence showing exactly when TAER replaces misleading near-duplicates and when it helps for other reasons such as broader task alignment. We summarize promising diagnostic directions in Appendix D.

There are also broader deployment considerations. A retrieval method that downweights superficially similar examples can improve robustness, but it may also under-serve tasks where lexical anchoring is genuinely necessary, especially in domains with fixed templates or highly formulaic outputs. Future work should therefore study calibrated variants of TAER that expose controllable bias strength, so practitioners can tune the trade-off between diversity, faithfulness, and task-specific copying requirements.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, and Amanda Askell. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Hanyu Duan, Yixuan Tang, Yi Yang, Ahmed Abbasi, and Kar Yan Tam. 2024. Exploring the relationship between in-context learning and instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3197–3210.

Liancheng Fang, Aiwei Liu, Hengrui Zhang, Henry Peng Zou, Weizhi Zhang, and Philip S. Yu. 2025. TABGEN-ICL: Residual-aware in-context example selection for tabular data generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20027–20041.

Qi Guo, Leiyu Wang, Yidong Wang, Wei Ye, and Shikun Zhang. 2024. What makes a good order of examples in in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14892–14904.

Shivanshu Gupta, Matt Gardner, and Sameer Singh. 2023. Coverage-based example selection for in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13924–13950.

Joonwon Jang, Sung Yang, Taehun Lee, and Minjoon Shin. 2024. Rectifying demonstration shortcuts in in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

Baijun Ji, Xiangyu Duan, Zhenyu Qiu, Tong Zhang, Junhui Li, Hao Yang, and Min Zhang. 2024. Submodular-based in-context example selection for LLM-based machine translation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15398–15409.

Janak Kapuriya, Manit Kaushik, Debasis Ganguly, and Sumit Bhatia. 2025. Exploring the role of diversity in example selection for in-context learning. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2962–2966.

Marvin Kaster, Wei Zhao, and Steffen Eger. 2021. Global explainability of BERT-based evaluation metrics by disentangling along linguistic factors. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8912–8925.

Yichuan Li, Xiyao Ma, Sixing Lu, Kyumin Lee, Xiaohu Liu, and Chenlei Guo. 2024. MEND: Meta demonstration distillation for efficient and effective in-context learning. In *The Twelfth International Conference on Learning Representations*.

Yukang Lin, Bingchen Zhong, Shuoran Jiang, Joanna Siebert, and Qingcai Chen. 2025. Reasoning graph enhanced exemplars retrieval for in-context learning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9737–9759.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Chengwei Qin, Aston Zhang, Chen Chen, Anirudh Dagar, and Wenming Ye. 2024. In-context learning with iterative demonstration selection. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7441–7455.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, and Noah A. Smith. 2022. Selective annotation makes language models better few-shot learners. In *The Eleventh International Conference on Learning Representations*.

Ruixiang Tang, Dehan Kong, and Longtao Huang. 2023. Large language models can be lazy learners: Analyze shortcuts in in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4645–4657.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *The Seventh International Conference on Learning Representations*.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9840–9855.

Yiquan Wu, Anlai Zhou, Yuhang Liu, Yifei Liu, Adam Jatowt, Weiming Lu, Jun Xiao, and Kun Kuang. 2024. Chain-of-quizzes: Pedagogy-inspired example selection in in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10136–10142.

A Additional Experimental Results

Table 5 reports robustness to several task-prefix variants, and Table 6 summarizes grouped gains by model scale and shot count.

Table 5: Impact of different task-prefix designs. "Generated prefix" averages over five LLM-generated task descriptions. "Template prefix" uses a fixed template of the form "Your task is {task name}". "Task suffix" appends task information after the input instead of prepending it.

Method	3-shot Avg.	5-shot Avg.
TAER	73.62	**74.39**
Generated prefix	**73.71**	74.28
Template prefix	73.26	74.13
Task suffix	73.58	74.26

Table 6: Average improvement of TAER grouped by model scale. T and D denote gains over Top-k and DICL, respectively. "Small" averages Qwen3-8B and Qwen3-32B, while "Large" averages Qwen2.5-72B-Instruct and Llama-3.3-70B-Instruct.

Group	T-3	D-3	T-5	D-5
Small	4.21	3.56	4.39	3.84
Large	2.87	2.16	3.29	2.11

B Implementation Summary

- **Task-aware retrieval.** Prepend a short task description to the query and each candidate, encode them with the final-layer `[CLS]` representation, and rank candidates by cosine similarity.
- **Anti-shortcut reranking.** Restrict to the top-k task-relevant candidates, represent original inputs with a shallow encoder layer, and keep the n candidates with the lowest cosine similarity.
- **Candidate pool.** Use the task-specific training split for each benchmark.
- **Demonstration budget.** Set the final demonstration count to $n \in \{3, 5\}$ and the first-stage retrieval depth to $k = 2n$.
- **Demonstration format.** For classification tasks, demonstrations contain the input and label verbalization; for generation tasks, they contain the input and target text.
- **Controlled factors.** All baselines share the same task instructions, label verbalizations, candidate pool, and downstream decoding setup.

C Example Task Prefixes

Table 7: Representative task-prefix templates used to construct task-aware retrieval inputs.

Task family	Prefix template
Sentiment / classification	"Predict the sentiment of the sentence:"
Paraphrase / similarity	"Determine the semantic relation between the two texts:"
Natural language inference	"Predict whether the premise entails the hypothesis:"
Summarization	"Write a concise summary of the article:"
Data-to-text generation	"Generate a fluent description from the structured input:"

D Boundary Cases and Diagnostic Directions

- **Tasks requiring lexical anchoring.** Downweighting overlap may remove genuinely useful demonstrations. A useful follow-up diagnostic is to compare win rates after stratifying by lexical-overlap level.
- **Homogeneous candidate pools.** The second stage has limited room to find non-redundant alternatives. A useful diagnostic is to examine the similarity distribution within the retrieved top-k set.
- **Near-duplicate-heavy pools.** TAER should help most when redundancy is common among otherwise relevant neighbors. This can be tested by measuring how often TAER replaces a high-overlap candidate from the task-aware top-k set.
- **Larger prompt budgets.** Gains may increase because extra slots are easier to fill with redundant examples. Replacement patterns can therefore be compared across 3-shot and 5-shot settings.
- **Stronger downstream LLMs.** Absolute gains may shrink, but improved demonstration quality can still matter. Grouped analyses by model scale, as in Table 6, provide one simple check.