Agentic Verification for Ambiguous Query Disambiguation

Youngwon Lee[1*] **Seung-won Hwang**[1†] **Ruofan Wu**[2*] **Feng Yan**[2]
Danmei Xu[3] **Moutasem Akkad**[3] **Zhewei Yao**[3] **Yuxiong He**[3]
[1]Seoul National University [2]University of Houston [3]Snowflake AI Research

Abstract

We study ambiguous-query disambiguation in retrieval-augmented generation (RAG). Prior Diversify-then-Verify (DtV) pipelines first generate interpretations and then retrieve evidence, often introducing ungrounded queries that cannot be answered from the corpus and requiring costly post-hoc pruning and verification. We propose VERDICT, a novel approach that unifies diversification with verification by integrating retriever relevance and generator answerability feedback early. This not only reduces cascading errors but also enables parallelism. On ASQA, VERDICT improves grounding-aware F_1 by an average of 23% over the strongest baselines across multiple LLM backbones.

1 Introduction

Retrieval-Augmented Generation (RAG; Lewis et al., 2020) is expected to complement large language models (LLMs), trained on static data, when they struggle to provide reliable responses without external retrieval. This challenge is particularly evident in enterprise settings, where queries are asked on domain-specific and potentially evolving corpora inaccessible during LLM training. In such settings, answering a short, vague query requires RAG to disambiguate the query, into interpretations that can be answered from the corpus.

A natural approach is to retrieve a set of passages then generate disambiguations that can be answered from that set as in RAC (Kim et al., 2023), where a single top-k retrieval defines the scope of possible disambiguations. However, in such approaches, disambiguation coverage suffers from bounded recall (V et al., 2025). If the top-k passages do not sufficiently cover different meanings of the query, valid disambiguations may never be considered. Increasing k only partially alleviates this issue: it also introduces more irrelevant passages, and passing a large retrieved set to the generator at once makes it harder for the LLM to produce clean disambiguations.

Diversify-then-Verify (DtV) methods (Min et al., 2020; Cole et al., 2023; Kim et al., 2023; In et al., 2025) attempt to address this limitation by increasing retrieval diversity. They first generate candidate disambiguations with LLMs to diversify, then verify whether each disambiguation is actually supported by retrieved evidence. This can improve recall, but it does so before seeing evidence. As a result, the system cannot know which disambiguations are actually answerable from the corpus. Such unsupported disambiguations may still trigger retrieval and downstream processing, only to be discarded later. For example, in Figure 1(a), the query "What is HP" may refer to Hewlett-Packard, horsepower, or Harry Potter. LLMs trained on general knowledge may generate all three, even when the enterprise corpus lacks any reference to Harry Potter.

To address this limitation, we propose VERDICT (Verified DIversification with Consolidation), which integrates diversification with verification rather than performing them as separate stages. Unlike RAC, VERDICT does not rely on a single retrieved pool followed by one long-context generation step; unlike DtV, it does not generate disambiguations before seeing evidence. Instead, VERDICT first retrieves a high-recall universe for the ambiguous query and then processes each passage independently to determine whether it supports a concrete disambiguated question-answer pair. In this way, retrieval provides coverage over meanings present in the corpus, while passage-wise generation provides answerability. Diversification is therefore grounded at the point of creation, and unsupported disambiguations are pruned before they trigger additional retrieval.

* Work done while at Snowflake.
† Correspondence to: seungwonh@snu.ac.kr.

Findings of the Association for Computational Linguistics: ACL 2026, pages 38781–38796

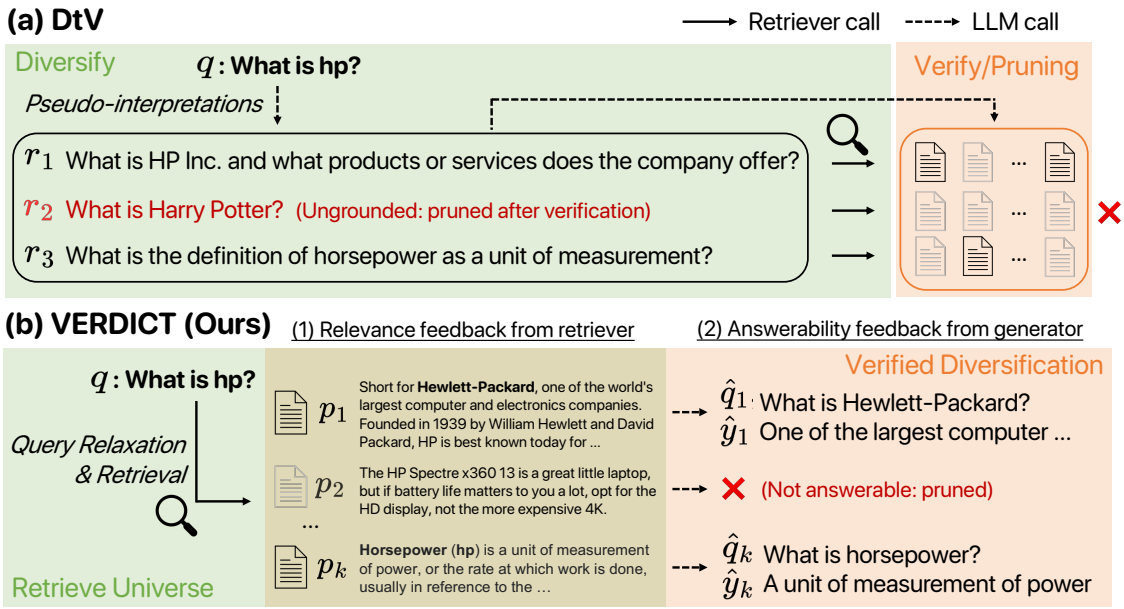

Figure 1: Comparison of (a) DtV (Diversify-then-Verify) and (b) VERDICT (Verified DIversification with ConsolidaTion, ours). Grayed-out icons represent passages that failed verification and were thus pruned; solid arrows represent retriever calls while dashed arrows refer to LLM calls.

Figure 1(b) illustrates that this integrated approach, verified diversification, leverages two complementary signals, namely the relevance feedback from the retriever and the answerability feedback from the generator LLM. Our parallel design also allows us to *consolidate* these two signals across interpretations. We cluster the generated interpretations to mitigate noise from either the retriever or the generator, without invoking additional retriever or generator calls. This has the additional benefit of accounting for the inherent redundancy in the retrieved passages, where multiple passages may support the same interpretation. While previous pipelines rely solely on the generator LLM to avoid generating duplicate interpretations, VERDICT elegantly accounts for this by choosing the single best representative interpretation from each cluster.

Finally, the tight coupling of each interpretation with its supporting passage in VERDICT facilitates the evaluation of grounded metrics, which are in high demand in RAG (Li et al., 2023a; Liu et al., 2023). As grounded metrics such as citation quality and verifiability are increasingly important in RAG, we use both existing and grounded metrics to demonstrate that VERDICT produces disambiguations that are not only diverse and accurate but also well-grounded.

Our key contributions are as follows:

- We introduce VERDICT, a novel framework for ambiguous question handling that unifies diversification and verification.
- We show that VERDICT improves efficiency by eliminating unnecessary retrieval and verification, and by enabling parallel execution. It also enhances effectiveness by mitigating cascading errors typical of sequential pipelines.
- We empirically validate VERDICT on widely adopted ambiguous QA benchmark, ASQA (Stelmakh et al., 2022), with average gain of 23% in F_1 score across different backbone LLMs.[1]
- Our code and evaluation are publicly available.[2]

2 Related Work

This section reviews prior work on the DtV workflow and on treating the retriever as an environment.

Diversification. The goal of this phase is to turn the given query into a diverse set of interpretations. Min et al. (2021) and Sun et al. (2023) studied an iterative approach by identifying one new intent

[1]The gain is computed against the best baseline for each backbone LLM.

[2]https://github.com/ludaya/verdict

at a time. Gao et al. (2021) generated clarification questions conditioned on generated answers from ambiguous questions, and then answers the clarification questions again.

With LLMs, few-shot in-context learning was used to generate clarifications or query rewrites directly, relying solely on their internal knowledge captured during pretraining (Kim et al., 2023; Ma et al., 2023).

Verification. Errors in retrieval or intent inference may generate question-passage pair that cannot provide the expected answer. Verification aims at pruning such pairs. Shao and Huang (2022) train a verifier to choose correct answers from a list of candidates drafted from each passage without question clarifications. More recent works such as Self-RAG (Asai et al., 2024) and Corrective RAG (Yan et al., 2024) also train a verifier to decide whether the retrieved passages are relevant enough to assist answer generation. To avoid verifier training, LLMs may leverage its parametric knowledge instead to verify (Li et al., 2024). Alternatively, MADAM-RAG (Wang et al., 2025) poses verification as a process of aggregating signals through multi-agent debate, where retrieved documents are assigned to individual LLM agents to reach a consensus.

Our Distinction. Unlike existing DtV approaches that consider verification as a post-hoc step, we combine diversification and verification to avoid inefficiencies and sequential dependencies.

3 Preliminary

3.1 Problem Formulation

Our formulation adheres to the conventional ambiguous question answering setting. Given an ambiguous question q and a passage corpus C, the task is to identify a set of valid interpretations $\mathcal{Q} = \{q_1, \ldots, q_N\}$, where each q_i resolves one plausible meaning of q, and corresponding answers $\mathcal{Y} = \{y_1, \ldots, y_N\}$. Because our focus is retrieval-augmented generation, we treat a grounded model output as a set of triples $\hat{\mathcal{T}} = \{(\hat{q}_j, \hat{y}_j, \hat{p}_j)\}_{j=1}^{M}$, where $\hat{p}_j \in C$ is the evidence passage associated with generated interpretation $\hat{q}_j$ and answer $\hat{y}_j$. We use $\hat{\mathcal{Q}}$ and $\hat{\mathcal{Y}}$ to denote the question and answer projections of $\hat{\mathcal{T}}$, respectively. Human annotations are denoted with tildes, e.g., $\tilde{\mathcal{Q}}$ and $\tilde{\mathcal{Y}}$.

3.2 Baselines: RAC and DIVA

In this section, we formally describe the main baseline method, DIVA (In et al., 2025), the most recent work with state-of-the-art performance which best exemplifies the existing DtV approaches. We start by describing a simpler form, RAC (Kim et al., 2023), which directly prompts an LLM with the passages retrieved with q. The retrieved passages form a *universe* $U_q = \mathrm{TopK}_k(C, q; s)$, where $\mathrm{TopK}_k(C, x; s)$ denotes the top-k passages in corpus C under retriever score $s(x, p)$. Then, RAC generates $(\hat{\mathcal{Q}}, \hat{\mathcal{Y}})$ under instruction I_{G},[3] without diversification or verification. As mentioned earlier, this suffers from (1) lack of verification for passages in U_q, and (2) the conflict between longer input context and promoting coverage of interpretations, both depending on $k = |U_q|$.

DIVA first diversifies the query, or, identifies pseudo-interpretations $\mathcal{R} = \{r_1, r_2, \ldots, r_L\}$ by prompting the LLM with q and instruction I_{P}, without accessing any retrieved knowledge. Each generated pseudo-interpretation $r_i \in \mathcal{R}$ is then used as a search query to retrieve top-k supporting passages, whose union forms its retrieval universe

$$U_{\mathcal{R}} \leftarrow \bigcup_{r \in \mathcal{R}} \mathrm{TopK}_k(C, r; s), \tag{1}$$

where the subscript $\mathcal{R}$ indicates that the universe $U_{\mathcal{R}}$ is derived from the set of pseudo-interpretations $\mathcal{R}$. As pseudo-interpretations can be ungrounded and the resulting $U_{\mathcal{R}}$ can be noisy, verification phase follows, to examine each pseudo-interpretation with universe $U_{\mathcal{R}}$. After this phase, $U_{\mathcal{R}}$ is reduced to a verified subset $U_V = \{p \in U_{\mathcal{R}} \mid \exists_{r \in \mathcal{R}}\, \texttt{Verify}(r, p) = 1\}$, from which disambiguated queries and answers $(\hat{\mathcal{Q}}, \hat{\mathcal{Y}})$ are generated under instruction I_{G}.

We identify the challenges and inefficiencies in DtV line of work as follows:

1. $U_{\mathcal{R}}$: Each pseudo-interpretation $r \in \mathcal{R}$ incurs at least one retriever call, some of which are ungrounded and could introduce noise.

2. U_V: Obtaining a verified subset U_V requires to process all (r, p) pairs, for example, inducing $|\mathcal{R}|$ `Verify` calls with input size of $\mathcal{O}(|U_{\mathcal{R}}|)$ each.[4] Plus, such long-context ver-

[3] For baseline reproduction, we reuse I_{G} from RAC; see Appendix G.

[4] In et al. (2025) *pruned* $U_{\mathcal{R}}$ in advance, though asymptotic cost remains unchanged.

ification requires a powerful LLM, which increases cost and hinders generalizability.

4 Method

We illustrate VERDICT with two distinctions: Verified Diversification (Section 4.1) and Consolidated Feedback (Section 4.2). In Section 4.3, we provide grounded evaluation protocols.

4.1 Verified Diversification

Unlike DtV, a valid interpretation in grounded disambiguation must be both plausible for q and answerable from C. We therefore construct grounded candidate triples $\mathcal{T}_0$ upfront by combining verification with diversification, rather than viewing verification as a separate, post-hoc process. This construction leverages feedback from both the retriever and the generator.

Relevance Feedback. "Relevance feedback from the retriever" in Figure 1 shows how we first retrieve $U = \mathrm{TopK}_k(C, q'; s)$ with a single round of retrieval before generating interpretations, where q' is a relaxed query obtained from the LLM with prompt I_{R}.[5] With relaxed q' and choice of a larger k for retrieval,[6] we further increase the diversity and coverage of the interpretations while keeping the process computationally efficient.

Answerability Feedback. "Answerability feedback from the LLM generator" in Figure 1 prunes interpretations that cannot be answered with evidence. Conditioned on our high-coverage universe U, the LLM generator is prompted to identify interpretations $\hat{\mathcal{Q}}$ and corresponding answers $\hat{\mathcal{Y}}$ from *each* passage.

This ensures the answerability of each disambiguated query $\hat{q}_i$, given the passage $\hat{p}_i$ from which it was derived: For example, in the figure, DtV retains passages like p_2, which mentions HP products but cannot answer the query asking what HP is, whereas our approach prunes such passages through failed execution of generating a question-answer pair.

In contrast, we keep p_1, where a valid question-answer pair can be generated. For each $p_i \in U$, the generator receives only (q, p_i) with prompt I_{E} and either returns a pair $(\hat{q}_i, \hat{y}_i)$ or abstains.[7]

[5]The prompt I_{R} can be found in Figure 5.

[6]k was empirically tuned to 20, when the interpretation coverage plateaus, e.g., yielding identical results to top-100 with a GPT-4o backbone.

[7]The LLM prompt I_{E} can be found in Figure 6.

	Retriever	LLM	
	# calls	# calls	input len.
DtV	$\mathcal{O}(\lvert\mathcal{R}\rvert)$	$\mathcal{O}(\lvert\mathcal{R}\rvert)$	$\times$ $\mathcal{O}(\lvert U_{\mathcal{R}}\rvert)$
VERDICT	$\mathcal{O}(1)$	$\mathcal{O}(\lvert U\rvert)$	$\times$ $\mathcal{O}(1)$

Table 1: Cost comparison between DtV and VERDICT in terms of retriever and LLM calls per question. For LLM calls, the input context length is measured by the number of passages, with the total input size determined by multiplying this length by the number of calls.

The successful outputs form candidate triples $\mathcal{T}_0 = \{(\hat{q}_i, \hat{y}_i, p_i) \mid p_i \in U,\ \hat{q}_i \neq \texttt{null}\}$.

From a practical standpoint, this passage-wise extraction can be executed in parallel, with each LLM call processing a single passage at a time. This minimizes the input sequence length for each call, optimizing both latency and computational overhead while reducing hallucination in less capable models.

Table 1 provides comparison of asymptotic cost of retrieval and LLM calls in VERDICT and DtV. First, while VERDICT makes a single call to the retriever to build U, DtV issues several search queries, proportional to the number of pseudo-interpretations $|\hat{\mathcal{Q}}|$. Next, the number of LLM calls and input size of each call, measured in terms of the number of passages, also shows the efficiency of VERDICT. DtV provides several passages as a list input, increasing the context size and the chance of introducing noise while processing the elongated input. In contrast, VERDICT requests answerability feedback for each passage in U, using only a single passage as a context.

While we assume that each interpretation of q can be answered using a single passage, an assumption inherently linked to the chosen passage granularity, VERDICT can be naturally extended to a multi-hop setting by evaluating small groups of passages (e.g., pairs).

4.2 Consolidated Feedback

While Section 4.1 ensures that each interpretation $\hat{q}_i$ is relevant to q and answerable using its supporting passage $\hat{p}_i$, our design allows us to account for any residual noise arising from either imperfect retrieval or LLM, by consolidating the candidate triples in $\mathcal{T}_0$, *without* requiring additional retrieval or LLM inference.

Specifically, question-answer pairs from generator are projected into a latent space $\mathbb{R}^d$ using an encoder f from the retriever (Figure 2, right box).

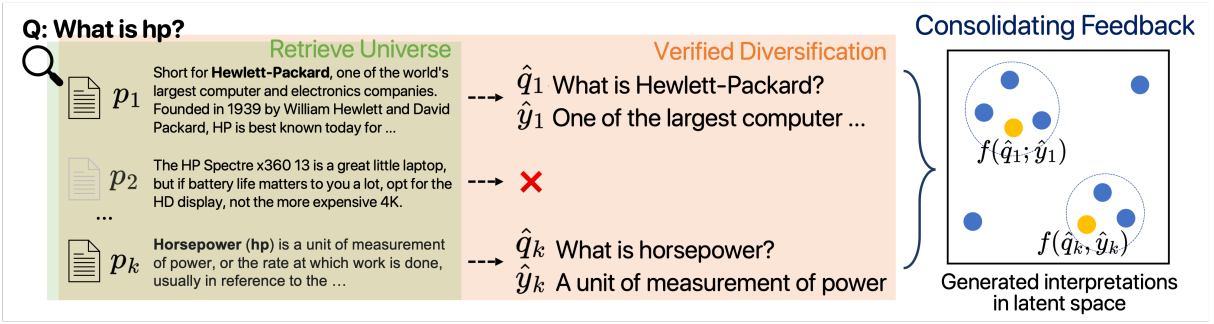

Figure 2: Illustration of the full pipeline of VERDICT: Verified diversification (Section 4.1) followed by consolidation phase (Section 4.2). On the right, yellow and blue dots represent embeddings of generated interpretations and their answers, embedded together after concatenation, while yellow color denotes medoids.

We then cluster these pairs and discard items that do not belong to any dense cluster, i.e., "outliers" in the density-based sense. In our setting, these typically arise from (1) interpretations that are not answerable from the associated passage, or (2) low-faithfulness rewrites loosely related to the original query q; excluding them improves groundedness without adding an external verifier. Instead, multiple generator and retriever feedback signals are consolidated, which aligns with seeking the most consistent outputs (Wang et al., 2023), improving robustness by aggregating multiple LLM outputs.

The final output $\hat{\mathcal{T}}$ contains the medoid triple from each non-outlier cluster. Selecting a medoid question per cluster also eliminates duplicate interpretations arising from the inherent redundancy in the corpus, where multiple passages may convey similar information and thus support identical interpretations. In contrast, both RAC and DIVA defer deduplication to the answer generation stage, leaving it solely to the LLM.

4.3 Grounding in Evaluation

This section discusses existing evaluation metrics, contrasted with grounded evaluation, which is often required in enterprise RAG scenarios.

Existing: Ungrounded Metrics. Current benchmarks (Min et al., 2020; Stelmakh et al., 2022) measure recall by checking whether diverse human-annotated interpretations "match" model-generated ones. Lexical similarity (e.g., BLEU) determines the match between generated ($\hat{q}$) and reference ($\tilde{q}$) interpretations as a binary decision:

$$V(\hat{q}, \tilde{q}) \in \{0, 1\}. \tag{2}$$

We classify this as **ungrounded evaluation**, as it does not assess whether the generation is supported by a passage. Moreover, such reference-based evaluation suffers from bounded coverage (missing grounded interpretations) or includes ungrounded interpretations, thereby providing an unfair boost.

Proposed: Grounded Metrics. Whether an answer can be grounded to the correct passage, known as verifiability or citation quality (Li et al., 2023a; Liu et al., 2023), has been a key evaluation criterion particularly in enterprise RAG settings.

We extend this idea to disambiguation: grounded metrics evaluate not only lexical matches, but also whether the interpretation can be supported by a passage in the corpus. This leads to an extended binary evaluation function factoring in passage:

$$V(\hat{q}, \hat{p}) \in \{0, 1\}, \tag{3}$$

where $\hat{p}$ is the supporting passage for $\hat{q}$, as identified by the model.

With this extended V, grounded precision (**G-precision**) is defined as the ratio of correctly grounded among model-generated interpretations:

$$\text{G-Precision} = \frac{1}{|\hat{\mathcal{Q}}|} \sum_{\hat{q} \in \hat{\mathcal{Q}}} V(\hat{q}, \hat{p}). \tag{4}$$

Unlike conventional recall, encouraging ungrounded diversity, optimizing for G-precision penalizes ungrounded interpretations. Similarly, recall metric can be grounded to evaluate model-generated interpretations against the **grounded gold** set, $\bar{\mathcal{Q}}$, complementing the incomplete human annotations.

$$\text{G-Recall} = \frac{1}{|\bar{\mathcal{Q}}|} \sum_{\bar{q} \in \bar{\mathcal{Q}}} V(\bar{q}, \hat{p}), \tag{5}$$

Compared to ungrounded recall, G-recall ensures that interpretations lacking corpus support, such

as Harry Potter are not rewarded, while grounded interpretations are encouraged. Due to its extended matching mechanism, G-recall is never lower than ungrounded recall, but the increase in the score remains marginal for models that overfit to ungrounded diversity. We further elaborate on the grounded metrics in Appendix A.

5 Results

In this section, we empirically validate VERDICT ensures human-level diversity, while balancing with grounding accuracy.

5.1 Experimental Settings

5.1.1 Evaluation Metrics

To evaluate diversity, we report the average number of generated interpretations per query $|\hat{\mathcal{Q}}|$, the proportion of queries with sufficient diversity, denoted Sufficient%[8] and the (ungrounded) recall computed against human interpretations without grounding.

For grounded metrics, we use G-precision and G-recall in Section 4.3, and G-F_1 score, defined as the harmonic mean of precision and recall, balancing the objectives of both.

5.1.2 Evaluation Datasets

We consider the ASQA benchmark (Stelmakh et al., 2022) as our main target dataset for evaluation. It is built upon the AmbigNQ benchmark (Min et al., 2020) which provides human-annotated interpretations $\tilde{\mathcal{Q}}$, along with corresponding answers $\tilde{\mathcal{Y}}$ for each ambiguous question q. Each interpretation $\tilde{q}_i$ is optionally attached with a supporting passage $\tilde{p}_i$ from the corpus C, Wikipedia. We utilize its validation split, which consists of 948 examples.

5.1.3 Baselines and Implementation Details

We compare VERDICT with RAC (Kim et al., 2023) and DIVA (In et al., 2025) illustrated in Section 3.2, the latter serving as the representative for DtV frameworks. We also evaluate pseudo-interpretations in DIVA as a separate baseline, to study how LLM diversification leads to ungrounded interpretations. This variant, denoted as "DIVA−Verification" in Table 2, also serves as a lightweight, efficiency-oriented baseline.

The retrieval system comprises `arctic-embed` (Merrick et al., 2024)[9] based first-phase retrieval and second-phase passage reranking using `gte-Qwen2` (Li et al., 2023b).[10]

For the clustering algorithm in the consolidation phase described in Section 4.2, we used HDBSCAN (Campello et al., 2013), a hierarchical density-based clustering algorithm. As the encoder f for obtaining embeddings, we reused the same `gte-Qwen2` embedding model used in retrieval. For decoding, greedy decoding was used to obtain deterministic responses from LLMs. More details can be found in Appendix C.

5.2 Results

We now evaluate whether VERDICT delivers diverse yet grounded interpretations efficiently and why its design reduces cascading errors. We organize the analysis around five research questions:

- RQ1: Is VERDICT diverse?
- RQ2: Does diversity balance with grounding?
- RQ3: What are the individual contributions of VERDICT's components, and how are they tunable?
- RQ4: Is VERDICT efficient?
- RQ5: Does VERDICT generalize across datasets, domains and tasks?

VERDICT ensures human-level diversity. Table 2 reports scores of VERDICT, human-annotation, RAC (Kim et al., 2023), and DtV (In et al., 2025), in terms of diversity metrics '$|\hat{\mathcal{Q}}|$', 'Sufficient%', and 'Recall' (Section 5.1.1).

Across all three, VERDICT's output is comparable to human annotations and remains consistent across model sizes. In contrast, diversity of interpretations from DIVA and RAC tend to be highly affected by model choice or size. For example, with the smaller model, 8B, the baseline output becomes noticeably less diverse, e.g., Sufficient% is dropped to <1%. This robust performance validates that VERDICT generalizes better across models, a direct benefit of its per-passage context isolation, which avoids the long-context reasoning that smaller models struggle with.

While Table 2 shows that DIVA achieves higher ungrounded recall, it is often inflated by generating plausible but ungrounded interpretations, such as "Harry Potter" for the query "What is HP" even

[8] We deduplicate the generated interpretations with the prompt I_{D} in Figure 9, and count unique interpretations.

[9] `Snowflake/snowflake-arctic-embed-m-v2.0`

[10] `Alibaba-NLP/gte-Qwen2-7B-instruct`

Method	Existing–Ungrounded			Grounded		
	$\lvert\hat{\mathcal{Q}}\rvert$	Sufficient%	Recall	G-Precision	G-Recall	G-F_1
LLaMA 3.1 8B						
DIVA (In et al., 2025)	1.36	0.21	21.36	76.78	25.02	37.74
−Verification	2.26	1.37	**40.24**	52.75	44.43	48.23
RAC (Kim et al., 2023)	0.93	8.09	11.40	**84.27**	17.43	28.89
VERDICT (Ours)	**3.78**	**23.76**	36.76	61.51	**54.77**	**57.94**
LLaMA 3.3 70B						
DIVA	2.06	5.60	32.44	69.89	36.55	48.00
−Verification	3.68	34.73	**47.23**	46.25	52.16	49.03
RAC	**3.78**	**57.81**	45.27	76.71	50.29	60.75
VERDICT	3.70	24.21	36.66	**81.50**	**58.04**	**67.80**
GPT-4o						
DIVA	1.73	3.80	25.57	72.34	29.38	41.79
−Verification	3.07	19.51	**45.39**	51.35	49.85	50.59
RAC	2.12	15.42	23.43	84.79	36.57	51.10
VERDICT	**3.16**	**21.49**	37.41	**92.82**	**57.25**	**70.82**
Human interpretations $\tilde{\mathcal{Q}}$	3.36	22.02	·	65.47	·	·

Table 2: Evaluation of diversity and correctness of generated interpretations on ASQA validation set, with both ungrounded and grounded metrics. "DIVA−Verification" refers to the pseudo-interpretations $\mathcal{R}$ generated before retrieval and verification.

when the corpus has no relevant documents. VERDICT intentionally prunes such interpretations early; this deliberate trade-off is precisely what leads to its superior performance on grounded metrics, as discussed next.

VERDICT ensures grounded diversity. VERDICT's strategy of early, localized verification prevents ungrounded interpretations from ever entering the pipeline. As shown in Table 2, this results in VERDICT achieving the highest G-precision, of 93% with GPT-4o and 81% with LLaMA 70B as the backbone LLM, while obtaining diverse enough interpretations at the same time. In contrast, G-precision shows that 35% of human-annotated interpretations and up to 53% of DtV interpretations are not grounded in the corpus. As a result, baselines fall behind grounded interpretations from VERDICT in terms of G-recall as well.

This gap between ungrounded and grounded scores indicates *cascading errors* in the baselines: ungrounded pseudo-interpretations proceed to retrieval and late verification, inflate ungrounded recall, and then depress grounded precision/recall once support is required. By deciding answerability per passage before any long-context reasoning and pruning duplicates in consolidation, VERDICT truncates these branches rather than letting them propagate.

These results reinforce our earlier discussion in Section 4.3: DtV, despite achieving higher ungrounded recall, suffers from the lowest G-precision across all backbone LLMs, highlighting its tendency to generate ungrounded interpretations. In contrast, both RAC and VERDICT produce more accurately grounded interpretations, with VERDICT further benefiting from both retriever and generator feedback. The larger gap between ungrounded recall and G-recall for our method, ultimately boosting grounded recall to about 58%, also shows VERDICT's interpretations are well-supported by their corresponding passages.

Clustering	$\lvert\hat{\mathcal{Q}}\rvert$	G-Precision	G-Recall
Parameter			
Default	**3.70**	81.50	**58.04**
Conservative	2.41	**82.40**	50.72
Embedding			
$f(\hat{q};\hat{y})$	**3.70**	**81.50**	**58.04**
$f(\hat{q})$	3.65	78.21	57.27

Table 3: Clustering strategies and performance of VERDICT with 70B generator. $f(\hat{q};\hat{y})$ denotes embedding the concatenated interpretation-answer pair, while $f(\hat{q})$ embeds the interpretation alone. Shaded results are based on default setting, and were copied from Table 2.

	Redundancy (%, ↓)
VERDICT	**10.21**
VERDICT − clustering	21.94

Table 4: Redundant interpretations are effectively pruned during clustering.

Variant	$\lvert\hat{\mathcal{Q}}\rvert$	G-Precision	G-Recall
VERDICT (full)	3.16	**92.82**	**57.25**
Retriever-only	**3.59**	68.35	55.08
Generator-only	1.15	91.10	24.36

Table 5: Ablation study of retriever and generator feedback in VERDICT. Detailed setting can be found in Appendix C.

Finally, G-F_1 scores, measuring the balance of G-precision and G-recall, show a clear performance gap between VERDICT and the baselines in pursuing both accurately grounded and diverse interpretations. For instance, while RAC achieves higher G-precision with the 8B backbone, it fails to ensure sufficient diversity, i.e., average interpretation per question is lower than 1. This leads to significantly lower G-recall and, consequently, G-F_1 score, reflecting its failure to provide meaningful interpretations of the original ambiguous question.

Component-wise contributions. To answer RQ3, we analyze the contributions of VERDICT's core components: The two feedback signals in verified diversification, and consolidation.

First, to isolate the impact of relevance and answerability feedback, we analyze the effect of ablating each signal. A retriever-only variant sees its G-precision drop sharply from 93% to 68%, confirming that many retrieved passages are relevant but not sufficient to answer a specific interpretation. Conversely, a generator-only variant suffers a catastrophic drop in diversity ($\lvert\hat{\mathcal{Q}}\rvert$ falls from 3.16 to 1.15) and G-recall (from 57% to 24%), as the generator has no grounded context to work from. These results confirm that both feedback signals are critical and work synergistically to achieve diverse and grounded disambiguation.

Second, we analyze the consolidation phase. Table 4 shows that consolidation effectively deduplicates interpretations. In addition, Table 3 also shows this step provides tunable control over the final output. Using a more conservative clustering setting increases precision at the cost of recall, a knob not available in other methods where it is solely up to the LLM to decide the number of resulting interpretations. The same table also shows that the performance of VERDICT is robust against the choice of embedding function f, yielding consistent performance. This demonstrates that while the joint feedback is the primary driver of performance, the consolidation phase acts as a crucial final step for refining, denoising, and controlling the output.

	Latency (s, ↓)
VERDICT	**2.12**
DIVA	5.34

Table 6: Average end-to-end latency of VERDICT and DIVA.

	DIVA	RAC	VERDICT
Answer Recall (%)	26.1	27.4	**35.8**

Table 7: Answer recall on AmbigDocs (validation split).

Finally, we compare our consolidation phase with multi-agent debate in Appendix D. Results further show that our clustering-based consolidation is preferable to a simplified multi-agent method, yielding higher performance and substantially lower latency.

Parallel processing improves efficiency. Turning to RQ4, Table 6 shows that the end-to-end latency of VERDICT is substantially lower than that of DtV. The reduction comes from structuring diversification and verification to operate independently for each retrieved passage, which removes sequential dependencies and enables concurrent LLM calls. In contrast, DIVA's late, long-context verification introduces longer serialized steps. A visual comparison of the two pipelines and a detailed latency breakdown are provided in Appendix B.

VERDICT generalizes to entity-level ambiguity. To answer RQ5, we first evaluate VERDICT on a different disambiguation benchmark, AmbigDocs (Lee et al., 2024), which primarily targets entity-level disambiguation, where the task is to resolve which specific referent is meant. Our main target benchmark, ASQA, on the other hand, emphasizes generating distinct plausible interpretations by completing underspecified intent. Despite this difference in ambiguity type, our results show that VERDICT generalizes well; it achieves higher answer recall on the AmbigDocs validation split than both DIVA and RAC as shown in Table 7, indicating that our verified diversification mechanism is beneficial for resolving entity ambiguity as well.

We note that the AmbigDocs metric, answer recall, measures the recall of ground-truth answers (entities) that can be derived from some Wikipedia article. As correct entity recovery requires identifying corpus-grounded answers, this result provides complementary evidence that improved grounding

	$\lvert\hat{\mathcal{Q}}\rvert$	Recall	G-Precision
DIVA	1.53	73.3	78.6
− Verification	**2.97**	66.7	30.2
VERDICT	1.87	**86.7**	**90.1**

Table 8: Performance on private, enterprise benchmark. The (ungrounded) recall is computed against the original query before rewriting during the dataset preparation process.

translates into better answer quality.

VERDICT also generalizes to proprietary enterprise data. To reflect enterprise usage, we evaluate VERDICT and baseline methods on a private, proprietary corporate IT-document corpus (215 internal PDFs) with 60 human-authored query-answer pairs and supporting passages. The document collection comprises internal, IT-related corporate PDFs, which were first converted to plain text. Then, human experts annotated 60 queries, each paired with ground-truth answers and supporting documents. In order to repurpose this benchmark for the ambiguous question answering task, the queries are rewritten to introduce greater ambiguity, where the original query is regarded as the human-annotated interpretation in ASQA. Query types include, but are not limited to, software configuration, compliance/policy and internal troubleshooting.

As shown in Table 8, VERDICT also outperforms the baselines in this enterprise-specific setting, where the private document corpus naturally constrains unsupported disambiguations.

VERDICT extends to multi-hop QA. Finally, we show that the design of VERDICT is not bound to single-hop QA tasks. Due to the page limit, we present how VERDICT can be extended to handle multi-hop QA tasks in Appendix E.

5.3 Error Analysis

In this section, we examine the types of errors arising from the retriever and the generator in VERDICT. Figure 3 shows the categorization $(\hat{q}, \hat{y})$ pairs generated from VERDICT. Specifically, an LLM judge decided whether $\hat{q}$ is relevant to the original question q, and whether $\hat{q}$ is answerable from $\hat{p}$.

First, the top-left corner of each plot in Figure 3 illustrates cases where the retrieved passage is both relevant and answerable. The 8B model correctly answers 92% of these instances, while stronger

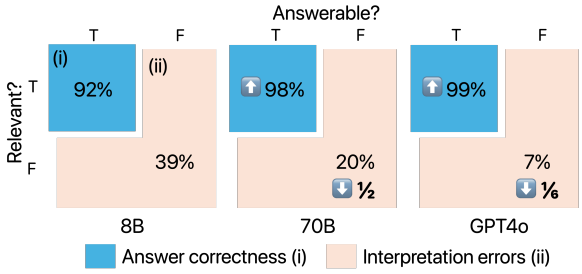

Figure 3: Analysis on accuracy of generated $\hat{q}$'s and $\hat{y}$'s from VERDICT. (i, answer correctness) Models easily derive correct $\hat{q}, \hat{y}$ given an answerable passage. (ii, interpretation error rate) Impact of model scale is more critical in discerning unanswerable passages.

models achieve even higher accuracy, reaching 98% and 99%, respectively.

Second, the remaining L-shaped sections depict scenarios where the passage $\hat{p}$ is either irrelevant or unanswerable. Here, the numbers represent the interpretation error rate, where the model should have identified that no interpretations can be derived from the passage but failed to do so. While smaller models struggle more, these errors are substantially mitigated by scaling model size.

We also discuss the stability of the LLM judge in Appendix F.

6 Conclusion

We challenge the conventional DtV workflow for grounding ambiguous questions in RAG and propose VERDICT, which more efficiently and effectively grounds diversification in retrieval. We enhance the existing protocol to not only evaluate ungrounded recall, but also assess whether interpretations can be properly grounded. Evaluations using both ungrounded and grounded metrics show that VERDICT significantly improves grounding while maintaining human-level diversity.

Limitations

First, while our consolidation reduces noisy feedback, we have not considered extreme cases where the retriever or generator performs unreasonably poor or adversarially. Such failures can corrupt the initial retrieval; an adaptive mechanism that trades retrieval breadth (via additional calls) against efficiency could mitigate this, but designing an agent that adjusts its behavior online based on feedback quality remains open.

Second, this study focuses on grounded disambiguation for single-turn, text-only retrieval.

While results on multi-hop QA using HotpotQA (Appendix E) suggest that our method generalizes beyond the evaluated setting, other retrieval-augmented scenarios, such as conversational or multimodal RAG, introduce additional requirements or adjustments that remain to be explored in future work.

Finally, our grounded metrics instantiate $V(\cdot, \cdot)$ with an LLM judge and therefore inherit standard LLM-as-a-judge limitations (e.g., prompt sensitivity). Because each evaluation example is presented as a simple task containing only a single interpretation and a single passage, the context is small, which substantially reduces these risks, as empirically validated in Appendix F.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Ricardo J. G. B. Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining, 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14-17, 2013, Proceedings, Part II*, volume 7819 of *Lecture Notes in Computer Science*, pages 160–172. Springer.

Jeremy Cole, Michael Zhang, Daniel Gillick, Julian Eisenschlos, Bhuwan Dhingra, and Jacob Eisenstein. 2023. Selectively answering ambiguous questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 530–543, Singapore. Association for Computational Linguistics.

Yifan Gao, Henghui Zhu, Patrick Ng, Cicero Nogueira dos Santos, Zhiguo Wang, Feng Nan, Dejiao Zhang, Ramesh Nallapati, Andrew O. Arnold, and Bing Xiang. 2021. Answering ambiguous questions through generative evidence fusion and round-trip prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3263–3276, Online. Association for Computational Linguistics.

Yeonjun In, Sungchul Kim, Ryan A. Rossi, Mehrab Tanjim, Tong Yu, Ritwik Sinha, and Chanyoung Park. 2025. Diversify-verify-adapt: Efficient and robust retrieval-augmented ambiguous question answering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1212–1233, Albuquerque, New Mexico. Association for Computational Linguistics.

Gangwoo Kim, Sungdong Kim, Byeongguk Jeon, Joonsuk Park, and Jaewoo Kang. 2023. Tree of clarifications: Answering ambiguous questions with retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 996–1009, Singapore. Association for Computational Linguistics.

Yoonsang Lee, Xi Ye, and Eunsol Choi. 2024. Ambigdocs: Reasoning across documents on different entities under the same name. *CoRR*, abs/2404.12447.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Dongfang Li, Zetian Sun, Xinshuo Hu, Zhenyu Liu, Ziyang Chen, Baotian Hu, Aiguo Wu, and Min Zhang. 2023a. A survey of large language models attribution. *Preprint*, arXiv:2311.03731.

Xiaonan Li, Changtai Zhu, Linyang Li, Zhangyue Yin, Tianxiang Sun, and Xipeng Qiu. 2024. LLatrieval: LLM-verified retrieval for verifiable generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5453–5471, Mexico City, Mexico. Association for Computational Linguistics.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023b. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Nelson Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating verifiability in generative search engines. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7001–7025, Singapore. Association for Computational Linguistics.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

Luke Merrick, Danmei Xu, Gaurav Nuti, and Daniel Campos. 2024. Arctic-embed: Scalable, efficient, and accurate text embedding models. *Preprint*, arXiv:2405.05374.

Sewon Min, Kenton Lee, Ming-Wei Chang, Kristina Toutanova, and Hannaneh Hajishirzi. 2021. Joint passage ranking for diverse multi-answer retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6997–7008, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797, Online. Association for Computational Linguistics.

Zhihong Shao and Minlie Huang. 2022. Answering open-domain multi-answer questions via a recall-then-verify framework. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1825–1838, Dublin, Ireland. Association for Computational Linguistics.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8273–8288, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Weiwei Sun, Hengyi Cai, Hongshen Chen, Pengjie Ren, Zhumin Chen, Maarten de Rijke, and Zhaochun Ren. 2023. Answering ambiguous questions via iterative prompting. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7669–7683, Toronto, Canada. Association for Computational Linguistics.

Venktesh V, Mandeep Rathee, and Avishek Anand. 2025. SUNAR: Semantic uncertainty based neighborhood aware retrieval for complex QA. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5818–5835, Albuquerque, New Mexico. Association for Computational Linguistics.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. 2025. Retrieval-augmented generation with conflicting evidence. In *Proceedings of the Second Conference on Language Modeling*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective retrieval augmented generation. *Preprint*, arXiv:2401.15884.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

A Grounding Evaluation

Here, we explain in more detail how our evaluation protocol identified matching pairs for computing precision and recall. We begin by reviewing the setting of previous works and how original precision/recall has been computed in ambiguous question answering evaluation.

Ungrounded Precision/Recall To obtain precision, prior work determines whether each generated interpretation is correct by comparing it against the human interpretations $\tilde{\mathcal{Q}}$. If the generated interpretation matches one of the human interpretations, it is considered correct; otherwise, it is considered incorrect. This can be formulated as

$$\text{Precision} = \frac{1}{|\hat{\mathcal{Q}}|} \sum_{\hat{q} \in \hat{\mathcal{Q}}} \texttt{Match}\left(\hat{q}, \tilde{\mathcal{Q}}\right), \quad (6)$$

where $\texttt{Match}(\cdot, \cdot)$ denotes whether two questions are matched, or, the extended match

$$\texttt{Match}(\hat{q}, \tilde{\mathcal{Q}}) = \mathbb{1}\left(\exists_{\tilde{q} \in \tilde{\mathcal{Q}}} \texttt{Match}\left(\hat{q}, \tilde{q}\right)\right), \quad (7)$$

compared against the set of interpretations $\tilde{\mathcal{Q}}$. Following the notation from the main text, from this point we denote such matching function as V, for the sake of notational simplicity.

Before LLM-as-a-judge was widely adopted, V was often instantiated with measures such as lexical-overlap based scores, such as BLEU, exceeding some threshold τ or not,

$$V\left(\hat{q}, \tilde{q}\right) = \mathbb{1}\left(\texttt{BLEU}\left(\hat{q}, \tilde{q}\right) > \tau\right). \quad (8)$$

While such pairwise match can be easily replaced with querying an LLM judge, it would incur $|\hat{\mathcal{Q}}| \times |\tilde{\mathcal{Q}}|$ LLM calls; for the sake of efficiency, we let the LLM judge to directly determine the match against the set of human interpretations as follows:

$$V\left(\hat{q}, \tilde{\mathcal{Q}}\right) = \texttt{LLM}\left(\hat{q}, \tilde{\mathcal{Q}}; I_{\text{M}}\right). \quad (9)$$

Similarly, recall, the proportion of ground-truth interpretations successfully generated, is defined as

$$\text{Recall} = \frac{1}{|\tilde{\mathcal{Q}}|} \sum_{\tilde{q} \in \tilde{\mathcal{Q}}} V\left(\tilde{q}, \hat{\mathcal{Q}}\right), \quad (10)$$

where whether each $\tilde{q}$ has been covered or not is determined with an LLM in the same way as Eq. 9.

Grounded Precision/Recall Our grounded evaluation essentially replaces $V(\cdot, \cdot)$ with new matching mechanism that also counts grounding. For grounding the precision metric, we directly verify if the supporting passage $\hat{p}$ provided along with $\hat{q}$ can answer $\hat{q}$. Thus, Eq. 6 is rewritten to consider the 'match' between $\hat{q}$ and $\hat{p}$ as

$$\text{Grounded Precision} = \frac{1}{|\hat{\mathcal{Q}}|} \sum_{\hat{q} \in \hat{\mathcal{Q}}} V\left(\hat{q}, \hat{p}\right), \quad (11)$$

where we replace $\hat{p}$ with retrieved passages from the corresponding retrieval pool, proxy to the whole passages C, if such supporting passage $\hat{p}$ is not available, for example for the case of pseudo-interpretations which are obtained independently of retrieval. Implementation-wise, V is realized with an LLM judge as

$$V(\hat{q}, \hat{p}) = \texttt{LLM}\left(\hat{q}, \hat{p}; I_{\text{V}}\right). \quad (12)$$

With the same rationale, we redefine recall as a grounded metric

$$\text{Grounded Recall} = \frac{1}{|\bar{\mathcal{Q}}|} \sum_{\bar{q} \in \bar{\mathcal{Q}}} V\left(\bar{q}, \hat{p}\right), \quad (13)$$

where we also accommodate models that do not provide $\hat{p}$ for each $\hat{q}$ with $\tilde{\mathcal{P}}$, as for precision.

In Eq. 10, the ground-truth interpretation has been approximated by human interpretations $\tilde{\mathcal{Q}}$. To increase the recall of this proxy, we add verified model prediction $V(\hat{q}, \hat{p}) = 1$.

$$\bar{\mathcal{Q}} = \left\{\hat{q} \in \left(\hat{\mathcal{Q}} \cup \tilde{\mathcal{Q}}\right) \middle| V\left(\hat{q}, \hat{p}\right)\right\}. \quad (14)$$

Eq. 13 shows how $\bar{\mathcal{Q}}$ complements the human interpretations $\tilde{\mathcal{Q}}$.

The prompts used to instruct the judge LLM, I_{M} for finding a match in a list of questions and I_{V} for verifying a passage can be found in Appendix G.

B E2E Pipeline of VERDICT and DIVA

Figure 4 describes how an ambiguous question is processed by VERDICT and DIVA (In et al., 2025), a representative of DtV (Diversify-then-Verify) workflow. DtV involves more sequential steps, verifying the relevance of passages post hoc while in VERDICT verification is integrated into diversification, as shown on the left.

Table 9 illustrates how the difference in design complexity translates into a difference in efficiency, providing a detailed breakdown of the

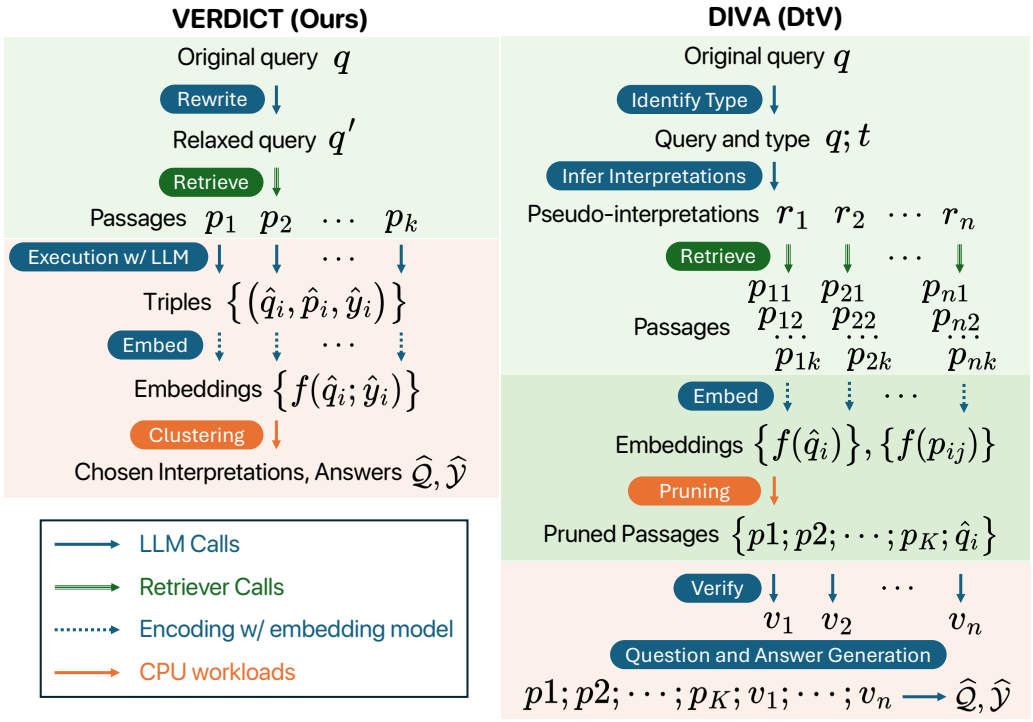

Figure 4: Comparison of end-to-end workflow of VERDICT and DtV (DIVA; In et al., 2025) for handling ambiguous question answering task. Vertical arrangement denotes sequential dependency, while calls that can run in parallel are placed at the same horizontal level.

Component	Latency (s, ↓)
VERDICT	
query rewriting	0.15
retrieval	0.5
generation	1.24
embed + clustering	0.2
DtV	
pseudo-interpretations	0.44
retrieval	0.58
embed + prune	0.25
verify	0.2
generation	∼3.8

Table 9: Latency per component of VERDICT and DtV pipelines.

average latency for both systems. The first step of DIVA, "pseudo-interpretations," corresponds to "DIVA−Verification" in Table 2, which confirms that it is indeed a lightweight baseline in isolation. At the same time, its poor grounding shows that the sequential nature of DtV pipelines leads to unnecessary retrieval, pruning, and verification.

C More on Implementation Details

Here, we provide additional implementation details.

For main experiments, VERDICT (ours) and RAC use top-20 passages, retrieved with relaxed question q' as the search query. For DtV, we use the same encoder, Qwen, for pruning the passages and then fix the verifier LLM to GPT-4o: The verifier had to use the most expensive model among those used in our evaluation, as less capable models struggled to provide reliable results for long-context inputs, severely damaging their performance. Following the original paper's setting, we retrieved top-5 passages for each pseudo-interpretation, then finally selected top-5 among them, after pruning.

For the ablation study provided in Table 5, since VERDICT cannot completely remove either retriever or generator feedback, we consider a proxy of such ablated version where either feedback signal is substantially weakened. Specifically, we weaken the answerability feedback by deferring to a small model (8B) when large (GPT-4o) and small models disagree on the answerability. For retriever feedback, we randomize the second-phase retrieval to suppress retriever signals.

D Comparison with Multi-Agent Method

MADAM-RAG (Wang et al., 2025) treats the processing of each retrieved document as a document-level LLM agent and then uses debate to aggregate

Method	$\lvert\hat{Q}\rvert$	G-Precision	G-Recall
VERDICT	3.70	**81.5**	**58.0**
MADAM-RAG, single round	4.02	76.6	57.8

Table 10: Comparison with an LLM aggregation variant inspired by MADAM-RAG on ASQA using the LLaMA 3.3 70B backbone.

Method	$\lvert\hat{Q}\rvert$	G-Precision
VERDICT (ours)	**2.98**	**91.27**
DtV	2.25	77.43

Table 11: Multi-hop results on HotpotQA shows VERDICT is not confined to single-hop settings.

and to reconcile their outputs, using an aggregator agent. In our setting, the closest design question is whether the passage-level candidates from verified diversification should likewise be consolidated by an LLM aggregator. To isolate this choice, we use the same candidate triples generated by VERDICT, but replace our embedding-based clustering with an LLM-based aggregation call that deduplicates candidates and outputs the final set. As shown in Table 10, this aggregation variant slightly degrades G-precision while yielding nearly identical G-recall. It also incurs higher consolidation latency: our embedding and clustering step takes about 0.2 seconds, whereas the LLM-based consolidation call takes about 2 seconds. Full multi-round debate, as used in MADAM-RAG, would further increase this cost.

E Extension to Multi-Hop QA

Here, we examine how VERDICT naturally extends to multi-hop question answering. We focus on HotpotQA (Yang et al., 2018), a two-hop benchmark.

VERDICT keeps the efficiency of parallel single-passage contexts with early answerability checks, but adapts the context construction. During query relaxation, the LLM produces two relaxed sub-queries, q'_1 and q'_2, one per hop. We retrieve the top-k passages for each, yielding sets U_{q_1} and U_{q_2}. We then form the Cartesian product $U_{q_1} \times U_{q_2}$, i.e., all ordered pairs (p_1, p_2) with $p_1 \in U_{q'_1}$ and $p_2 \in U_{q'_2}$. For each pair, we concatenate the two passages to create a single context that plugs directly into the original VERDICT pipeline; the verified diversification and consolidation phases proceed unchanged. This construction implicitly instantiates a concrete two-hop decomposition for each candidate interpretation while preserving our parallel evaluation.

	run 1	run 2	run 3	Avg.
G-Precision	92.82	92.51	92.90	92.74

Table 12: Repeated evaluation of VERDICT with the GPT-4o backbone.

The extended VERDICT handles multi-hop queries by jointly considering two passages at a time. It shows that the breadth-depth trade-off can also be controlled by adjusting the granularity of each passage itself; larger or bundled passages naturally enable broader multi-hop coverage without altering the framework.

We report $\lvert\hat{Q}\rvert$ and G-Precision in Table 11, where VERDICT outperforms the baseline again. Ungrounded metrics are not reported due to the lack of human-annotated interpretation sets in HotpotQA; grounded recall is omitted because it evaluates to 1 by construction in this setup.

F Stability of LLM judge

In this section, we show that variability of the LLM judge is empirically small enough to justify the comparisons in Section 5.2. We repeat the evaluation process for the GPT-4o backbone, to validate whether virtually identical results are produced, and we found that the maximum difference across runs was only 0.4 percentage points, as shown in Table 12.

G LLM Prompts

The instruction prompts used to instantiate the various components of our method are as follows. For reproducing baselines, we reused their prompts, I_{P} for generating pseudo-interpretations (In et al., 2025) or I_{G} for generating a list of interpretations and answers at once (Kim et al., 2023), which can be found in their respective papers.

Figure 5 shows the prompt for query relaxation, and Figure 6 shows the prompt used to generate interpretation and answer in VERDICT. Figure 7 and 8 present prompts used for evaluation, as discussed in detail in Appendix A. Figure 9 also shows the prompt used for deduplication during evaluation.

Query Relaxation Prompt

Convert the given natural language question into a broad retrieval query. Keep only the essential entities and core concepts, making the query as open as possible.

Input fields are:
Question: {`original query` (q)}

Output fields are:
Rewritten Question: {`relaxed query` (q')}

Figure 5: Prompt I_{R} for obtaining the relaxed query q'.

Prompt for Verified Diversification

Given an ambiguous query and one of the passages from retrieval results, provide a disambiguated query which can be answered by the passage. Try to infer the user's intent with the ambiguous query and think of possible concrete, non-ambiguous rewritten questions. If you cannot find any of them, which can be answered by the provided document, simply abstain by replying with 'null'. You should provide at most one subquestion, the most relevant one you can think of.

Here are the rules to follow when generating the question and answer:
1. The generated question must be a disambiguation of the original ambiguous query.
2. The question should be fully answerable from information present in given passage. Even if the passage is relevant to the original ambiguous query, if it is not self-contained, abstain by responding with 'null'.
3. Make sure the question is clear and unambiguous, while clarifying the intent of the original ambiguous question.
4. Phrases like 'based on the provided context', 'according to the passage', etc., are not allowed to appear in the question. Similarly, questions such as "What is not mentioned about something in the passage?" are not acceptable.
5. When addressing questions tied to a specific moment, provide the clearest possible time reference. Avoid ambiguous questions such as "Which country has won the most recent World Cup?" since the answer varies depending on when the question is asked.
6. The answer must be specifically based on the information provided in the passage. Your prior knowledge should not intervene in answering the identified clarification question.

Input fields are:
Question: {`ambiguous question` (q)}
Passage: {`passage` (p)}

Output fields are:
Interpretation: {`generated interpretation` ($\hat{q}$)}
Answer: {`generated answer` ($\hat{y}$)}

Figure 6: Prompt I_{E} for obtaining interpretation $\hat{q}$ and answer $\hat{y}$ with answerability feedback from the LLM.

Evaluation Prompt for Ungrounded Precision/Recall

Given a list of generated disambiguated subquestions that clarify the intent of an ambiguous question, compare them with the list of predefined subquestions and determine how many have been successfully identified. You should return a binary label, Yes or No, for each subquestion indicating whether it was covered or not.

Input fields are:
Question: {`ambiguous question` (q)}
Generated Disambiguations: {`generated interpretations` ($\hat{\mathcal{Q}}$)}
Ground-truth Disambiguations: {`human-annotated interpretations` ($\tilde{\mathcal{Q}}$)}

Output fields are:
Decisions: {`match` ($V(\hat{q}, \tilde{q})$'s)}

Figure 7: Prompt I_{M} for determining matches between $\hat{q}$'s and $\tilde{q}$'s.

Evaluation Prompt for Verification

Given a question, an answer and an associated passage, decide if the passage can support the answer, providing enough evidence to reach the answer given the question. Your answer should be either Yes or No.

Input fields are:
Question: {`interpretation` ($\hat{q}$)}
Passage: {`passage` ($\hat{p}$)}

Output fields are:
Decision: {`match` ($V(\hat{q}, \hat{p})$)}

Figure 8: Prompt I_{V} for determining a match between $\hat{q}$ and $\hat{p}$.

Evaluation Prompt for Deduplication

Given a list of subquestions, which are derived disambiguations of an ambiguous query, remove nearly identical duplicates and leave only distinct ones. You should provide a list of the remaining subquestions, one at a line.

Input fields are:
Ambiguous Question: {`ambiguous question` (q)}
List of Disambiguated Subquestions: {`interpretations` ($\hat{\mathcal{Q}}$ or $\tilde{\mathcal{Q}}$)}

Output fields are:
List of Unique Subquestions: {`deduplicated interpretations` }

Figure 9: Prompt I_{D} for removing (near-)duplicates in a list of interpretations for an ambiguous question.