Dual-Confidence Contrastive Decoding for Retrieval-Augmented Generation

Anonymous ACL submission

Abstract

Retrieval-augmented generation (RAG) increasingly requires models to answer questions from multiple retrieved documents, where only some sources are relevant and the retrieved bundle may contain stale, noisy, or conflicting evidence. Existing contrastive decoding methods primarily focus on resolving conflicts between the model's internal memory and the retrieved context. In contrast, we study the complementary problem of intra-context conflict in multi-document RAG. To evaluate this setting, we introduce **DRQA**, a factual-conflict question answering benchmark derived from enterprise deep-research scenarios, where answers are grounded in synthetic enterprise-specific facts that are designed not to be recoverable from the model's internal memory. We further propose **Dual-Confidence Contrastive Decoding** (**DCCD**), a training-free decoding method that combines document-level confidence, which estimates whether a document appears sufficient for answering the question, with token-level confidence, which estimates whether that document supports a confident next-token prediction. DCCD selects positive and negative document-conditioned streams using these dual-confidence signals and scales a document-level contrast by their confidence margin. Across DRQA and standard multi-document QA benchmarks, DCCD achieves the best average performance among full-context and contrastive decoding baselines, with the largest gains on DRQA. These results highlight the importance of source-aware, confidence-gated decoding when retrieved evidence is internally conflicting.

1 Introduction

Retrieval-augmented generation (RAG) has become an essential paradigm for grounding large language models (LLMs) in external knowledge (Gao et al., 2023), enabling them to answer domain-specific and knowledge-intensive queries beyond what is stored in their parameters. Initially developed for open-domain question answering (QA) (Karpukhin et al., 2020; Lewis et al., 2020), RAG has become an integral component of workflows that require resolving information from multiple sources, from open-domain deep-research agents that synthesize evidence across the web (Du et al., 2026; Java et al., 2026) to enterprise systems that retrieve from heterogeneous data such as reports, emails, and chats (Choubey et al., 2025; Abaskohi et al., 2026).

Despite these advancements, a central challenge for RAG systems is the resolution of conflicts among the knowledge sources (Xu et al., 2024). At inference time, the LLM can be overly reliant on its internal memory acquired during pre-training, even when the retrieved context provides newer or more task-specific evidence (Chen et al., 2022; Xie et al., 2024). One popular training-free approach to context-memory conflict is contrastive decoding (CD) (Li et al., 2023), which modulates generation by contrasting model predictions under different conditioning contexts (Shi et al., 2024; Wang et al., 2025a; Khandelwal et al., 2025). Rather than fine-tuning the LLMs, contrastive decoding adjusts token probabilities based on the difference between output distributions with and without the context. However, this formulation treats the retrieved evidence as a single context and does not resolve conflicts within the retrieved context itself.

In multi-document QA, the model needs to decide the relevance of each document to the query and which evidence to use (Wan et al., 2024; Jin et al., 2024b). The retrieved evidence may contain answer-supporting evidence alongside irrelevant distractors, stale information, and plausible but incorrect claims. In this setting, the decoding problem is not only about choosing the next token, but also about choosing the evidence source that should inform that token. We use *confidence* as the complement of *uncertainty* and distinguish two

levels: *document-level confidence*, which estimates whether a retrieved document appears sufficient for answering the question, and *token-level confidence*, which estimates whether a document-conditioned stream yields a sharp local next-token distribution. Token-level confidence has been used to capture how confidently the model predicts under a given context (Malinin and Gales, 2021; Jiang et al., 2023; Fadeeva et al., 2024; Nguyen et al., 2026), but it cannot determine whether the context itself is sufficient to answer the question. Document-level confidence addresses the complementary question of which retrieved source contains answer-bearing evidence (Kadavath et al., 2022; Singh et al., 2023; Wan et al., 2024; Joren et al., 2025), but it does not ensure that the model can generate from that source with confidence. Effective multi-document RAG therefore requires both source selection and generation confidence.

Most existing RAG and conflict-resolution benchmarks do not directly isolate this joint capability. For example, retrieval-augmented QA benchmarks (Joshi et al., 2017; Kwiatkowski et al., 2019; Mallen et al., 2023; Zhang et al., 2024) mainly test whether models can ground answers in retrieved context despite noisy retrievals, while conflict-resolution benchmarks (Lee et al., 2024; Wang et al., 2025b) introduce contradictory evidence through altered public facts or ambiguous entity references, such that resolving the conflict may depend on the model's internal world knowledge rather than on confidence over the retrieved evidence itself. This challenge is especially pronounced in enterprise settings, where relevant facts are often private or organization-specific, and retrieved evidence can be heterogeneous, time-sensitive, and internally conflicting (Choubey et al., 2025; Abaskohi et al., 2026).

Motivated by these gaps, we introduce an enterprise benchmark for multi-document question answering and propose a novel decoding method. To address the *evaluation gap*, we propose **DRQA**, a factual-conflict QA benchmark derived from the recent DRBench (Abaskohi et al., 2026) consisting of synthetic enterprise-specific facts grounded in company profiles, user personas, and deep-research tasks, where the answer cannot be inferred from the model's internal memory. Instead, each answer must be recovered from a retrieved bundle containing correct, temporal, misinformation, and noise documents, requiring models to identify the evidence source that supports the answer. To address the *decoding gap*, we propose Dual-Confidence Contrastive Decoding (**DCCD**), a training-free contrastive decoding method that uses dual-confidence signals: document-level confidence estimates whether a document appears sufficient for answering the question, while token-level confidence estimates whether the model can produce a sharp next-token prediction from that document. DCCD uses these signals to select positive and negative document-conditioned streams and scales the contrastive term by their confidence margin, amplifying high-confidence evidence while suppressing conflicting or unsupported alternatives. Across multiple RAG benchmarks, DCCD achieves the best average performance, with the largest gains on DRQA.

We summarize our contributions as follows. (1) We introduce **DRQA**, a factual-conflict QA benchmark derived from enterprise deep-research scenarios, where each query is paired with a retrieved bundle containing one correct document and distractors that are misinformation, stale temporal, or noise. (2) We propose Dual-Confidence Contrastive Decoding (**DCCD**), a training-free contrastive decoding method that combines document-level answerability confidence with token-level generation confidence to perform confidence-gated document-level contrast. (3) We show that DCCD improves over full-context and contrastive decoding baselines across standard multi-document QA benchmarks, with especially large gains on DRQA, highlighting the importance of source-aware contrast when retrieved evidence is internally conflicting.

2 Related Work

2.1 Retrieval-Augmented Question Answering

Despite their capabilities, LLMs remain prone to hallucination and are limited by the knowledge acquired during training, making it necessary to ground generation in verifiable external evidence. Retrieval-augmented generation (RAG) dynamically incorporates information through dense passage retrieval (Karpukhin et al., 2020), and has been shown to improve open-domain question answering when the model's internal knowledge is insufficient (Lewis et al., 2020). Consequently, models are typically evaluated on open-domain QA benchmarks on their ability to answer factual questions using retrieved passages from public corpora. For example, Izacard and Grave (2021) use real Google queries and Wikipedia answers to eval-

uate the ability of the model to ground responses in retrieved evidence, PopQA (Mallen et al., 2023) further probes the boundary between internal and retrieved knowledge by focusing on long-tail facts for which retrieval is often necessary, while RetrievalQA (Zhang et al., 2024) evaluates adaptive RAG behavior under retrieved contexts of varying usefulness.

Since knowledge conflict can also arise within the retrieved context itself (Xu et al., 2024), conflict-oriented benchmarks have been proposed to evaluate the models' ability to arbitrate among conflicting evidence. However, existing benchmarks are constructed by altering public facts or referencing ambiguous entities (Lee et al., 2024; Wang et al., 2025b). In these settings, models may still be able to resolve the conflict using internal world knowledge (e.g., recognizing the true public fact or the intended entity) rather than by reasoning over the retrieved text itself. In parallel, deep-research benchmarks evaluate broader systems that gather and synthesize evidence across many sources, including open-domain web research and enterprise settings (Du et al., 2026; Java et al., 2026; Abaskohi et al., 2026). Our proposed **DRQA** dataset fills this gap by combining conflict-oriented factual QA with enterprise deep-research scenarios, where the answer is unlikely to be available from internal memory and must instead be recovered by identifying which retrieved document supports the answer.

2.2 Contrastive Decoding

Contrastive decoding (CD) (Li et al., 2023) was originally proposed as a training-free strategy that selects tokens with high expert-model probability and low amateur-model probability, using their probability discrepancy to favor fluent and informative continuations. Context-Aware Decoding (CAD) (Shi et al., 2024) adapts this idea to improve context faithfulness in RAG by contrasting the LLM with and without retrieved context, so that generation follows retrieved evidence rather than prior knowledge. Further work makes the contrastive weight adaptive (Zhao et al., 2024; Kim et al., 2024). For example, ADACAD (Wang et al., 2025a) adjusts the contrastive weight at each decoding step using the Jensen-Shannon divergence (JSD) between contextual and zero-shot distributions, while COCOA (Khandelwal et al., 2025) introduces token-level confidence-aware adaptation to address conflict between contextual and internal knowledge. Most similar to our work is Dynamic Contrastive Decoding (DVD) proposed by Jin et al. (2024a), which targets multi-document QA by contrasting document-conditioned predictions and dynamically increasing the influence of documents estimated to be useful for the current generation step. However, DVD does not explicitly distinguish document-level support from token-level generation confidence.

2.3 Confidence Estimation

Current work on confidence estimation has primarily focused on confidence in the model's output distribution (Geng et al., 2024). Entropy-based methods estimate predictive uncertainty for autoregressive generation (Malinin and Gales, 2021; Jiang et al., 2023), and token-level uncertainty has been used for hallucination detection and fact checking (Fadeeva et al., 2024). Recent probability-only methods motivate lightweight top-k token statistics for uncertainty estimation (Nguyen et al., 2026), where our token-level confidence follows the Dirichlet uncertainty framework of Malinin and Gales (2018) by mapping the top-k logits of each stream to Dirichlet concentration parameters and computing the expected entropy. In parallel, document-level confidence addresses the complementary problem of whether an evidence source is sufficient to answer the question. Prior work has studied the models' ability to self-evaluate whether they know an answer (Kadavath et al., 2022), identify convincing evidence under conflicting sources (Wan et al., 2024), and estimate the sufficiency of sources in retrieved evidence (Joren et al., 2025; Hwang et al., 2025). The closest work to our document-level confidence is context-sufficiency estimation (Joren et al., 2025), which asks whether the provided context contains enough information to answer a question. In this work, our proposed DCCD combines document-level and token-level confidence estimates inside the contrastive decoding process.

3 DRQA: Deep Research Question Answering Benchmark

We derive **DRQA** from DRBench (Abaskohi et al., 2026), a 100-task enterprise deep-research benchmark spanning three industries—retail, healthcare, and electric vehicles—and ten domains. DRBench contains internal and external supporting facts embedded in generated enterprise documents verified

by human annotators. We use the 568 synthetically generated internal facts, which are designed not to appear on the public web and therefore are not expected to be recoverable from the model's internal memory. Additional construction details and prompts are provided in Appendix A.

3.1 Dataset Construction

Each DRBench task contains internal insights generated from the organization's industry, priorities, customer segments, and business goals. Since we focus on retrieval-augmented question answering rather than full deep-research report generation, we convert each internal insight into a QA pair. For example, the insight "Lee's Market reduced food waste by 8% in Q2 2024, saving $1.2M" becomes the question "What food-waste reduction rate did Lee's Market achieve in Q2 2024?"

For each QA pair, we construct an evidence set with four document types: *correct*, *misinformation*, *temporal*, and *noise*. Correct documents contain the ground-truth answer, misinformation documents contain a plausible but incorrect answer, temporal documents contain an answer that was valid at an earlier document date, and noise documents are topically plausible but do not answer the question. Each document is assigned a document date in ISO 8601 format and an original source type such as email, chat, PDF, DOCX, or slide deck.

DRBench generates insight files through a three-step needle-in-a-haystack procedure (Abaskohi et al., 2026) by first producing a document outline in a randomly assigned source type, and then injecting the ground-truth insight into an appropriate section, before finally filling the remaining sections with contextually plausible but topically irrelevant text. We reuse the DRBench insight and distractor files as correct and noise documents. Since we evaluate model reasoning rather than file extraction, we serialize documents as Markdown and JSONL formats for model input.

Misinformation To construct *misinformation* documents, we prompt the LLM to generate a plausible perturbation of the ground-truth insight and then use the same three-step DRBench document-generation procedure. To make these documents realistic but not explicitly labeled, we add subtle format-specific uncertainty markers, such as conversational hedges in chat or tentative phrasing in reports. These markers reflect naturally occurring epistemic language in enterprise communication, but do not identify the document as incorrect. Thus, models must rely on weak cues, source metadata, and context rather than explicit correctness labels.

Temporal Temporal documents are generated to read as *authoritative at the time of writing*, rather than as retrospective historical summaries. We first sample the `document_date` uniformly between 6 months and 3 years before the ground-truth document date, then prompt the LLM to generate a value that would have been accurate and current as of that date. A second call produces the full document in present tense using the same DRBench document-generation procedure.

We ensure that temporal documents contain no explicit "archived" or "outdated" labels, where `document_date` is the sole staleness signal. The direction of change is unconstrained, since the conflict should be resolved by comparing dates rather than by judging whether a trend is intuitive. When serialized for model input, each document is preceded by its document date and source format.

3.2 Quality Verification

Complementing DRBench's per-file human spot-check protocol, we apply an automated LLM-as-a-judge quality filter to all generated conflict documents. Each document type is checked against a fixed rubric that returns a structured JSON binary verdict. For misinformation documents, we verify that the claimed answer is factually and specifically different from the ground truth while remaining plausible to an uninformed reader. For temporal documents, we verify that the historical answer is distinct from the current ground truth and that the document contains no explicit staleness labels. We manually inspect and fix failures, then rerun the filter until all criteria are met.

Unlike standard open-domain QA benchmarks, where retrieval is usually expected to help, DRQA creates a setting in which retrieval is necessary but not sufficient. The answers are synthetic private facts, and each retrieved bundle contains both the correct answer and plausible contradictions. The evaluation question is therefore not only whether the model benefits from retrieval, but which value it commits to when confronted with conflicting and noisy evidence. DRQA directly evaluates retrieval-grounded conflict resolution, a capability that remains underexplored in prior work.

4 Methodology

In this section, we describe our proposed Dual-Confidence Contrastive Decoding (**DCCD**) with an overview diagram illustrated in Figure 1. Additional derivations and implementation details of DCCD are presented in Appendix B.

4.1 Problem Setup

Let x denote the question and let $\mathcal{D} = \{d_1, \ldots, d_n\}$ be the list of documents returned by the retriever. The goal is to generate an accurate and precise answer to x using $\mathcal{D}$. At each decoding step t, the model produces next-token logits $\mathbf{z}_t$ conditioned on the current prefix $y_{<t}$, and the decoding strategy selects the next token y_t.

4.2 Document-Level Confidence

In multi-document RAG, the retrieved set $\mathcal{D}$ can contain documents with different levels of relevance and evidential value. While retrieval similarity scores provide a useful filter, they are optimized for query-document relevance rather than for whether a document contains enough information to answer the question. Motivated by prior work studying whether the model knows the answer (Kadavath et al., 2022) and the sufficiency of a retrieved document for answering the query x (Wan et al., 2024; Joren et al., 2025), we first estimate a document-level confidence score for each question-document pair (x, d_i).

Specifically, we construct a support-probe prompt $\pi_{\text{sup}}(x, d_i)$ that asks the model whether d_i contains enough information to answer x. The probe is formulated as a binary yes/no question. Let $\mathcal{Y}_+$ be the set of yes verbalizer tokens and $\mathcal{Y}_-$ be the set of no verbalizer tokens, including common case and leading-space variants. We use the model's next-token distribution after the support probe, and define the document-level confidence score q_i as in Equation 1.

$$q_i = \frac{\sum_{v \in \mathcal{Y}_+} p_\theta(v \mid \pi_{\text{sup}}(x, d_i))}{\sum_{v \in \mathcal{Y}_+ \cup \mathcal{Y}_-} p_\theta(v \mid \pi_{\text{sup}}(x, d_i))}. \quad (1)$$

Document-level confidence q_i is computed once before decoding and estimates whether a retrieved document d_i should be considered a candidate evidence source. Importantly, the support probe requires only a single forward pass per document, and adds a one-time preprocessing cost that is amortized over the answer-generation steps.

4.3 Token-Level Confidence

Document-level confidence estimates whether a retrieved document is sufficient for answering the question, but it does not indicate whether the model can use that document to make a confident prediction at a particular decoding step. We therefore compute a token-level confidence score for each document-conditioned generation stream.

At decoding step t, let $\mathbf{z}_t^i \in \mathbb{R}^V$ denote the next-token logits produced by the model conditioned on the question x, document d_i, and the current prefix $y_{<t}$. Following the Dirichlet uncertainty framework of Malinin and Gales (2018), we approximate predictive uncertainty by treating the top-k logits of each document-conditioned stream as evidence for a Dirichlet distribution over next-token categorical distributions. This gives a lightweight, training-free confidence estimate based only on the most relevant next-token alternatives (Nguyen et al., 2026).

Let $\mathcal{T}_{(i,t)}$ be the set of top-k token indices under $\mathbf{z}_t^i$. For each $v \in \mathcal{T}_{i,t}$, we define the Dirichlet concentration parameter:

$$\alpha_{(i,t,v)} = \text{softplus}\big(\mathbf{z}_t^i[v]\big) + 1. \quad (2)$$

Let:

$$S_{(i,t)} = \sum_{v \in \mathcal{T}_{(i,t)}} \alpha_{(i,t,v)}, \quad \bar{p}_{(i,t,v)} = \frac{\alpha_{(i,t,v)}}{S_{(i,t)}} \quad (3)$$

We measure raw top-k uncertainty as the expected entropy of the categorical distribution induced by this Dirichlet posterior:

$$\widetilde{H}_k(\mathbf{z}_t^i) = \sum_{v \in \mathcal{T}_{i,t}} \bar{p}_{i,t,v} \big[\psi(S_{i,t} + 1) - \psi(\alpha_{i,t,v} + 1)\big] \quad (4)$$

where $\psi(u) = \frac{d}{du} \log \Gamma(u)$ is the digamma function. We normalize this value by the maximum entropy over a top-k support,

$$H_k(\mathbf{z}_t^i) = \frac{\widetilde{H}_k(\mathbf{z}_t^i)}{\log k}, \quad (5)$$

and define token-level confidence as

$$c_{(i,t)} = 1 - H_k(\mathbf{z}_t^i). \quad (6)$$

Thus, $c_{(i,t)}$ is high when document d_i yields a sharp next-token prediction under the current prefix, and low when its evidence is spread across many plausible next tokens. Unlike q_i, which is computed once before decoding, $c_{(i,t)}$ is recomputed at every decoding step and captures whether a candidate evidence source supports a clear local continuation.

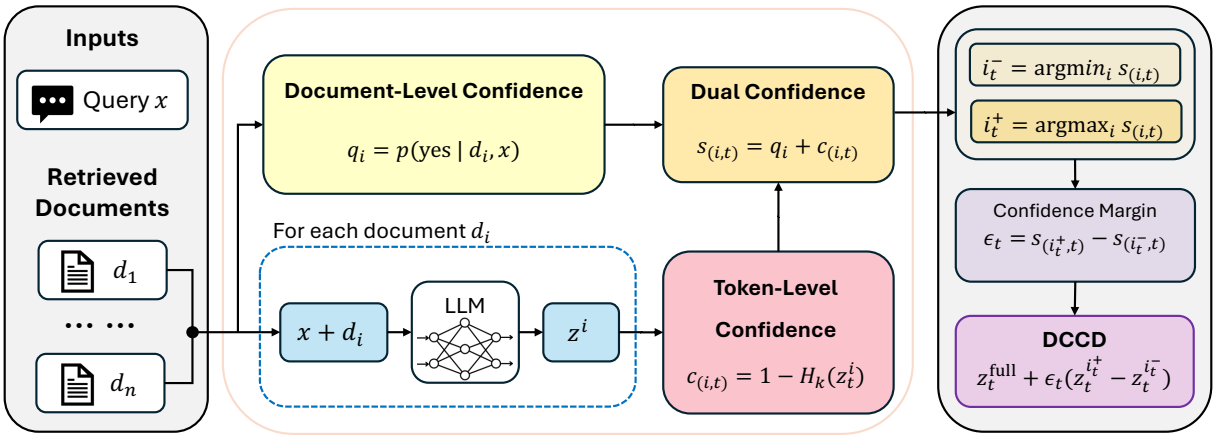

Figure 1: Overview of Dual-Confidence Contrastive Decoding (DCCD). Given a query x and retrieved documents $\{d_1, \ldots, d_n\}$, z^i denotes the next-token logits conditioned on document d_i. DCCD combines document-level (q_i) and token-level ($c_{(i,t)}$) confidence scores to select positive and negative document-conditioned streams at each decoding step. The margin between the selected scores controls the strength of the contrastive term ϵ_t.

4.4 Dual-Confidence Contrastive Decoding

At each decoding step, DCCD combines document-level and token-level confidence to choose the document-conditioned streams that define the contrastive direction. For each document d_i, we define the dual-confidence score as the sum of the two terms:

$$s_{(i,t)} = q_i + c_{(i,t)}. \tag{7}$$

Because both terms are normalized to $[0, 1]$, this unweighted sum provides a simple hyperparameter-free aggregation of source sufficiency and local generation confidence, where weighted extensions are left as future work.

From Equation 7, we select the positive and negative document-conditioned streams based on the highest and lowest dual-confidence scores at each decoding step:

$$i_t^+ = \arg\max_i s_{(i,t)}, \qquad i_t^- = \arg\min_i s_{(i,t)}. \tag{8}$$

We use the margin between the positive and negative scores to control the strength of the contrastive term:

$$\epsilon_t = s_{(i_t^+,t)} - s_{(i_t^-,t)}. \tag{9}$$

Finally, let $\mathbf{z}_t^{\text{full}}$ denote the next-token logits conditioned on the question with the full retrieved document set $\mathcal{D}$ and the current output prefix $y_{<t}$. Let $\mathbf{z}_t^i$ denote the next-token logits conditioned on the question, document d_i, and the same prefix. The decoding rule is then:

$$\mathbf{z}_t = \mathbf{z}_t^{\text{full}} + \epsilon_t \left(\mathbf{z}_t^{i_t^+} - \mathbf{z}_t^{i_t^-} \right). \tag{10}$$

This rule anchors generation in the full retrieved context by adding a document-level contrast that amplifies evidence from the most confident source and suppresses evidence from the least confident source at each decoding step.

5 Experiments

5.1 Settings

Datasets In addition to DRQA, we evaluate on well-known multi-document question answering benchmarks, including Natural Questions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017), using the preprocessing of Izacard and Grave (2021). We also evaluate on the more recent PopQA (Mallen et al., 2023) and RetrievalQA (Zhang et al., 2024). For DRQA, since the ground-truth answer is expressed as free-form text, we use LLM-as-a-judge to evaluate semantic correctness against the ground-truth insight. For all other datasets, we use normalized exact string match (`str-em`) following prior work.

Baselines We compare DCCD against standard full-context decoding (full) and contrastive decoding baselines. In particular, we include the standard CAD (Shi et al., 2024), DVD (Jin et al., 2024a), AdaCAD (Wang et al., 2025a) and CoCoA (Khandelwal et al., 2025). We also include the zero-shot baseline without any addition evidence for reference. Additional details and hyperparameters setting of all decoding methods are described in Appendix C.

Method	DRQA		NQ		TriviaQA		RetrievalQA		PopQA	
	@5	@10	@5	@10	@5	@10	@5	@10	@5	@10
`Qwen3.5-2B`										
zero-shot	0.00		15.62		32.21		47.83		18.44	
full	11.44	7.39	51.75	52.44	69.59	71.33	65.35	66.00	51.35	53.32
CAD	10.92	7.39	51.30	52.22	69.75	71.56	65.24	66.14	51.30	52.61
ADACAD	10.92	7.92	51.05	52.05	69.33	71.28	65.06	66.03	51.11	52.57
DVD	12.68	6.87	50.25	50.30	69.57	71.13	64.56	65.21	50.41	52.31
COCOA	7.75	4.75	45.40	45.29	66.47	67.71	61.62	61.87	49.29	49.94
DCCD (ours)	**16.73**	**12.15**	**53.10**	**53.24**	**70.04**	**72.01**	**66.25**	**66.61**	**51.75**	**53.34**
`Qwen3.5-9B`										
zero-shot	0.00		29.56		57.76		50.02		26.06	
full	17.78	10.56	54.99	57.23	72.47	74.30	65.67	66.10	50.93	52.10
CAD	17.78	10.21	55.21	57.56	72.39	74.31	65.42	66.28	50.57	51.56
ADACAD	16.73	10.74	55.04	57.70	72.46	74.35	65.21	66.03	50.50	51.49
DVD	16.90	10.92	54.32	55.57	71.00	72.44	65.35	64.99	50.36	50.60
COCOA	14.61	10.74	50.83	51.61	71.35	73.50	64.34	64.81	51.16	52.21
DCCD (ours)	**21.48**	**13.03**	**55.68**	**58.42**	**73.08**	**75.10**	**66.61**	**66.79**	**51.42**	**53.29**
`Phi-3-medium` (14B)										
zero-shot	0.00		39.11		69.11		49.08		37.22	
full	14.96	10.92	57.89	59.56	75.17	76.66	66.21	67.18	50.90	51.86
CAD	15.49	10.56	58.06	**59.78**	75.06	76.60	66.10	67.00	50.74	51.68
ADACAD	14.61	10.74	57.95	59.72	75.04	76.58	66.18	66.61	50.93	51.56
DVD	12.85	7.75	57.56	58.81	75.06	76.10	65.46	66.54	50.18	50.39
COCOA	12.85	10.56	55.10	54.79	**75.97**	76.52	66.00	66.54	50.71	51.16
DCCD (ours)	**20.25**	**12.32**	**58.31**	59.56	75.47	**77.06**	**67.86**	**67.94**	**51.39**	**52.14**

Table 1: Experiment results for DCCD and baseline decoding strategies with `Qwen3.5-2B`, `Qwen3.5-9B` and `Phi-3-medium` (14B). We report the `str-em` accuracy for NQ, TriviaQA, RetrievalQA, PopQA, and LLM-as-a-judge accuracy for DRQA for top-5 (@5) and top-10 (@10) documents provided by the respective datasets. The best result for each setting is **bolded**.

Models Motivated by recent trend on deploying Small Language Models (SLMs) for deep-research agents (Zheng et al., 2025; Wan et al., 2025), we evaluate all decoding strategies on instruction-tuned SLMs of small to medium sizes. Specifically, we use `Qwen3.5-2B`, `Qwen3.5-9B` (Team, 2026), and `Phi-3-medium` (14B) (Abdin et al., 2024) provided by Hugging Face's `transformers` library (Wolf et al., 2020). For all models, we use the same retrieval inputs and the default chat template used for instruction-training.

5.2 Results

As shown in Table 1, DCCD achieves the best average performance across nearly all settings. The gains are consistent across model sizes and datasets, and are especially pronounced on DRQA. We attribute these improvements to the dual-confidence design, where document-level confidence helps identify documents that appear sufficient for answering the question, while token-level confidence identifies document-conditioned streams that support sharp local predictions. These two signals address complementary failure modes. A document may appear answerable but still yield an uncertain next-token distribution, while a document may yield a sharp prediction despite supporting an incorrect or misleading answer.

On DRQA, zero-shot accuracy is 0% for all evaluated models, which is consistent with the benchmark design. Results on DRQA also show substantially lower accuracies than the Wikipedia-style datasets, as the retrieved evidence contains the correct document alongside noise, misinformation, and temporally stale evidence. We further observe an accuracy drop from @5 to @10, since each example contains only one current ground-truth document, additional distractors increase the chance that decoding follows a misleading source. In this conflict-heavy setting, DCCD achieves larger gains over baselines than on NQ, PopQA, RetrievalQA, and TriviaQA. This trend is consistent with the

Ablation	DRQA	NQ	TQA	RetQA	PopQA
@5					
DCCD	16.73	53.10	70.04	66.25	51.75
Fixed Gate	−0.02	−1.33	+0.22	−0.18	−0.78
Token-Only	−6.34	−1.44	−0.42	−0.15	−0.26
Doc-Only	+0.17	−0.44	+0.07	+0.18	−0.31
Random	−3.88	−1.22	−0.16	+0.11	−0.21
@10					
DCCD	12.15	53.24	72.01	66.61	53.34
Fixed Gate	−1.06	−0.33	+0.10	−0.54	−0.25
Token-Only	−4.58	−0.80	−0.23	−0.51	−0.40
Doc-Only	−0.23	−0.10	−0.20	−0.18	−0.12
Random	−3.70	−0.72	−0.05	−0.25	−0.34

Table 2: Ablation results with `Qwen3.5-2B` using the top-5 (@5) and top-10 (@10) documents. Performance differences are reported relative to DCCD, where green indicates improvement and red indicates degradation.

dual-confidence design, which favors document-conditioned streams that are both answerable and locally confident while contrasting against unsupported, stale, or misleading alternatives.

5.3 Ablation Study

We ablate the main components of DCCD to understand where the performance gain come from. Specifically, we run `Qwen3.5-2B` with @5 and @10 on all 5 datasets, and consider the following four settings. (1) We set gate $\epsilon_t = 1$, removing the adaptive gate during decoding (**Fixed Gate**). (2) We remove the document-level confidence probe, and select contrastive pairs with only token-level confidence (**Token-Only**). (3) We remove the token-level confidence probe, and select contrastive pairs with only document-level confidence (**Doc-Only**). (4) We randomly select the positive and negative by uniformly sampling from the retrieved documents (**Random**).

The ablation results in Table 2 show that DCCD benefits most from combining document-level and token-level confidence for source selection. Removing the document-level confidence probe and relying only on token-level confidence produces the largest degradation, especially on DRQA, with drops of -6.34 at @5 and -4.58 at @10. This suggests that token-level sharpness alone is insufficient in conflict-heavy settings, where a misleading document can still yield a confident local prediction. Randomly selecting the positive and negative documents also hurts performance in most settings, confirming that confidence-based source selection is important for identifying useful contrastive directions.

By contrast, using document-level confidence alone is comparatively stable, and even slightly improves over full DCCD in a few @5 settings. This indicates that document-level answerability is the dominant signal for resolving intra-context conflict, especially on DRQA, where the model must identify the document containing the current ground-truth answer. However, Doc-Only becomes consistently worse at @10, suggesting that token-level confidence provides a useful complementary signal as the retrieved bundle becomes more distractor-heavy. The fixed-gate ablation is mixed, setting $\epsilon_t = 1.0$ slightly improves TQA, but hurts most other settings, including larger drops on NQ and PopQA at @5 and on DRQA at @10. This suggests that always applying the same contrast strength can occasionally help, but the adaptive margin is more robust across datasets and retrieval depths.

6 Conclusion and Future Work

In this work, we propose Dual-Confidence Contrastive Decoding (**DCCD**), a training-free contrastive decoding method that combines document and token-level confidence to select positive and negative evidence sources during generation, and introduce **DRQA**, a factual-conflict QA benchmark derived from enterprise deep-research scenarios. Across DRQA and standard RAG benchmarks, DCCD consistently improves over prior decoding baselines, with the largest gains on DRQA. These results suggest that source-aware, confidence-guided contrast is especially useful when retrieved evidence is internally inconsistent.

For future work, improving the efficiency of DCCD is an important next step, since the method requires document-conditioned forward passes during decoding. Promising directions include more aggressive caching, document pruning, and lightweight confidence estimation. Second, DRQA currently focuses on factual-conflict QA in enterprise-style settings, extending the benchmark to longer-form answers, richer source metadata, and additional domains would further test retrieval-grounded conflict resolution. Finally, our confidence signals are simple and training-free. Learning or calibrating document-level and token-level confidence estimates may further improve robustness when retrieved evidence is ambiguous or insufficient.

Limitations

Similar to other document-aware contrastive decoding methods, DCCD introduces additional inference cost. At each decoding step, the method requires the full-context logits and document-conditioned logits for the retrieved documents, while the document-level confidence probe is computed once per question–document pair. This cost scales linearly with the number of retrieved documents, which has the same computational complexity as DVD (Jin et al., 2024a). In practice, our experiments use top-5 and top-10 retrieval settings, which are common in multi-document QA evaluation, but more efficient implementations such as batching document-conditioned forward passes, caching document states, or applying DCCD only at answer-critical decoding steps could reduce runtime overhead.

DRQA is designed to isolate retrieval-grounded conflict resolution in enterprise-style multi-document QA, where the answer is unavailable from internal memory and must be recovered from retrieved evidence. This controlled design is useful for evaluating the specific capability targeted by DCCD, but it is not intended to cover all forms of enterprise RAG. In particular, our current benchmark focuses on factoid question answering over text representations of enterprise documents, rather than full long-form report generation, interactive tool use, or multi-modal file understanding. These settings are closely related to deep-research workflows and could provide valuable extensions of DRQA.

Ethical considerations

DRQA is constructed from synthetic enterprise-style documents and is intended for research on retrieval-grounded conflict resolution, not for representing real companies, employees, or operational records. Because the benchmark includes deliberately generated misinformation and outdated evidence, the dataset should not be used as a factual knowledge source. Its intended use is to evaluate whether RAG systems can identify reliable evidence when retrieved documents conflict. The generated documents use synthetic personas, organizations, and facts. We have verified that the dataset does not contain real private enterprise records or intentionally identifying information. The benchmark is intended for research use only.

References

Amirhossein Abaskohi, Tianyi Chen, Miguel Muñoz-Mármol, Curtis Fox, Amrutha Varshini Ramesh, Étienne Marcotte, Xing Han Lù, Nicolas Chapados, Spandana Gella, Christopher Pal, Alexandre Drouin, and Issam H. Laradji. 2026. DRBench: A realistic benchmark for enterprise deep research. In *The Fourteenth International Conference on Learning Representations*.

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, and 1 others. 2024. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*.

Anthropic. 2025. Introducing claude opus 4.5. https://www.anthropic.com/news/claude-opus-4-5.

Hung-Ting Chen, Michael Zhang, and Eunsol Choi. 2022. Rich knowledge sources bring complex knowledge conflicts: Recalibrating models to reflect conflicting evidence. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2292–2307, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Prafulla Kumar Choubey, Xiangyu Peng, Shilpa Bhagavath, Kung-Hsiang Huang, Caiming Xiong, and Chien-Sheng Wu. 2025. Benchmarking deep search over heterogeneous enterprise data. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 501–517, Suzhou (China). Association for Computational Linguistics.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Licheng Zhang, Xiaorui Wang, and Zhendong Mao. 2026. Deepresearch bench: A comprehensive benchmark for deep research agents. In *The Fourteenth International Conference on Learning Representations*.

Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. 2024. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9367–9385, Bangkok, Thailand. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, Haofen Wang, and 1 others. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1):32.

Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. 2024. A survey of confidence estimation and calibration in large

language models. 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 6577–6595, Mexico City, Mexico. Association for Computational Linguistics.

Jeongyeon Hwang, Junyoung Park, Hyejin Park, Dongwoo Kim, Sangdon Park, and Jungseul Ok. 2025. Retrieval-augmented generation with estimation of source reliability. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 34279–34303, Suzhou, China. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Abhinav Java, Ashmit Khandelwal, Sukruta Prakash Midigeshi, Aaron Halfaker, Amit Deshpande, Navin Goyal, Ankur Gupta, Nagarajan Natarajan, and Amit Sharma. 2026. Characterizing deep research: A benchmark and formal definition. In *The Fourteenth International Conference on Learning Representations*.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore. Association for Computational Linguistics.

Jing Jin, Houfeng Wang, Hao Zhang, Xiaoguang Li, and Zhijiang Guo. 2024a. DVD: Dynamic contrastive decoding for knowledge amplification in multi-document question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4624–4637, Miami, Florida, USA. Association for Computational Linguistics.

Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, Xiaojian Jiang, Jiexin Xu, Li Qiuxia, and Jun Zhao. 2024b. Tug-of-war between knowledge: Exploring and resolving knowledge conflicts in retrieval-augmented language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 16867–16878, Torino, Italia. ELRA and ICCL.

Hailey Joren, Jianyi Zhang, Chun-Sung Ferng, Da-Cheng Juan, Ankur Taly, and Cyrus Rashtchian. 2025. Sufficient context: A new lens on retrieval augmented generation systems. In *International Conference on Learning Representations*, volume 2025, pages 20310–20334.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, and 1 others. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Anant Khandelwal, Manish Gupta, and Puneet Agrawal. 2025. CoCoA: Confidence- and context-aware adaptive decoding for resolving knowledge conflicts in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 6835–6855, Suzhou, China. Association for Computational Linguistics.

Youna Kim, Hyuhng Joon Kim, Cheonbok Park, Choonghyun Park, Hyunsoo Cho, Junyeob Kim, Kang Min Yoo, Sang-goo Lee, and Taeuk Kim. 2024. Adaptive contrastive decoding in retrieval-augmented generation for handling noisy contexts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2421–2431, Miami, Florida, USA. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Yoonsang Lee, Xi Ye, and Eunsol Choi. 2024. Ambigdocs: Reasoning across documents on different entities under the same name. In *First Conference on Language Modeling*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Hein-

rich 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*, volume 33, pages 9459–9474. Curran Associates, Inc.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12286–12312, Toronto, Canada. Association for Computational Linguistics.

Andrey Malinin and Mark Gales. 2018. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Andrey Malinin and Mark Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Manh Nguyen, Sunil Gupta, and Hung Le. 2026. Probabilities are all you need: A probability-only approach to uncertainty estimation in large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(38):32546–32554.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. 2024. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 783–791, Mexico City, Mexico. Association for Computational Linguistics.

Chandan Singh, John Morris, Alexander Rush, Jianfeng Gao, and Yuntian Deng. 2023. Tree prompting: Efficient task adaptation without fine-tuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6253–6267, Singapore. Association for Computational Linguistics.

Qwen Team. 2026. Qwen3.5: Accelerating productivity with native multimodal agents.

Alexander Wan, Eric Wallace, and Dan Klein. 2024. What evidence do language models find convincing? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7468–7484, Bangkok, Thailand. Association for Computational Linguistics.

Yi Wan, Jiuqi Wang, Liam Li, Jinsong Liu, Ruihao Zhu, and Zheqing Zhu. 2025. Pokeeresearch: Effective deep research via reinforcement learning from ai feedback and robust reasoning scaffold. *arXiv preprint arXiv:2510.15862*.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. 2025a. AdaCAD: Adaptively decoding to balance conflicts between contextual and parametric knowledge. 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 11636–11652, Albuquerque, New Mexico. Association for Computational Linguistics.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. 2024. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *The Twelfth International Conference on Learning Representations*.

Rongwu Xu, Zehan Qi, Zhijiang Guo, Cunxiang Wang, Hongru Wang, Yue Zhang, and Wei Xu. 2024. Knowledge conflicts for LLMs: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8541–8565, Miami, Florida, USA. Association for Computational Linguistics.

Zihan Zhang, Meng Fang, and Ling Chen. 2024. RetrievalQA: Assessing adaptive retrieval-augmented generation for short-form open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6963–6975, Bangkok, Thailand. Association for Computational Linguistics.

Zheng Zhao, Emilio Monti, Jens Lehmann, and Haytham Assem. 2024. Enhancing contextual understanding in large language models through contrastive decoding. 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 4225–4237, Mexico City, Mexico. Association for Computational Linguistics.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025. DeepResearcher: Scaling deep research via reinforcement learning in real-world environments. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 414–431, Suzhou, China. Association for Computational Linguistics.

A DRQA Construction Details

This section describes the procedure used to construct DRQA from DRBench. For each selected DRBench internal insight, we construct one DRQA example. Each example consists of a question, a ground-truth answer, and a retrieved document bundle. The question is derived from the internal insight and asks for a specific enterprise-contextualized value, such as a percentage, dollar amount, date, count, or named entity. The ground-truth answer is the original DRBench internal insight. For example, an insight such as "Lee's Market reduced food waste by 8% in Q2 2024, saving $1.2M" can be converted into the question "What food-waste reduction rate did Lee's Market achieve in Q2 2024?".

A.1 DRBench Background

DRBench (Abaskohi et al., 2026) is an enterprise deep-research benchmark designed to evaluate agents on complex, open-ended research tasks. Unlike standard question answering benchmarks (Kwiatkowski et al., 2019; Izacard and Grave, 2021; Mallen et al., 2023; Zhang et al., 2024), which usually ask for a short factual answer from a public corpus, DRBench tasks require an agent to synthesize information across public and private enterprise sources and produce a structured research report. Each task is grounded in an enterprise scenario that specifies a company context, a user persona, and a broad deep-research question. The retrieval environment includes heterogeneous sources such as public web pages, productivity documents, cloud files, emails, and chat conversations.

Each DRBench task contains supporting facts, referred to as *insights*, that are relevant to answering the deep-research question. These insights include both external facts that may be available on the public web and internal facts that are generated as private enterprise knowledge. The internal facts are embedded in generated enterprise documents using a "needle-in-a-haystack construction". Specifically, a realistic document is first outlined in a target modality, the insight is inserted into an appropriate section, and the remaining content is filled with contextually plausible but topically irrelevant text. This construction creates realistic enterprise files in which the answer-bearing evidence is embedded among surrounding noise.

DRQA reuses DRBench as scenario scaffolding rather than as a long-form report-generation benchmark. In particular, we take the synthetically generated internal insights from DRBench, which are not expected to be recoverable from the model's internal memory, and convert them into short-answer multi-document question answering tasks. This allows us to isolate the intermediate RAG capability we study in this paper: given a question and a bundle of retrieved documents, the model must identify which document contains the most reliable evidence and commit to the correct answer.

A.2 Document Generation Prompts

We list the prompt templates used during DRQA construction. Each template is shown as a Python f-string as passed to the LLM. Unless otherwise stated, prompts were run with `anthropic/claude-opus-4-5` (Anthropic, 2025) through OpenRouter[1].

Wrong-answer generation for misinformation documents This prompt produces the perturbed value inserted into each misinformation document. It is called once per misinformation document before the format-specific document generator. The document date is passed to the prompt so that generated dates or time periods remain coherent with the document timestamp.

```
Given this correct answer, generate a WRONG but plausible
↪  alternative.

CORRECT ANSWER: {fact.answer}
QUESTION: {fact.question}

RULES:
- If the answer contains a number/percentage, change it
↪  significantly (e.g., 8% -> 15% or 3%)
- If it contains a date, shift it by 6-18 months in either
↪  direction
```

[1] https://openrouter.ai

```
- The document containing this answer is dated {doc_date}. Any
↪  dates or time periods in the wrong answer MUST be
↪  compatible with the document date (i.e., the document
↪  should not claim knowledge of events that occur after its
↪  timestamp).
- Keep the same format/type as the original
- Make it plausible but clearly different

Return JSON:
{"wrong_answer": "The wrong answer", "reason": "Brief
↪  explanation of what was changed"}
```

Historical-answer generation for temporal documents This prompt produces the historical value inserted into a temporal document. The document date is sampled first and passed to the LLM, ensuring that the generated value is coherent with the document's timestamp.

```
You are writing a fact for a document dated {doc_date}.
The document will present this fact as its CURRENT finding --
↪  not as
a historical reference.

CURRENT (most recent) answer: {fact.answer}
QUESTION: {fact.question}

The document that will contain this answer is dated {doc_date}.
Generate a value that would have been accurate and considered
current as of that date.

STRICT RULES:
- The time period in your answer MUST differ from the time
↪  period
  in the current answer.
  e.g., if current says "Q2 2024", your answer must say "Q1
  ↪  2023" or
  "Q4 2022", NOT "Q2 2024".
- Change the numeric value OR the referenced date/period --
↪  both if
  possible.
- Do NOT use the phrase "as of [same date as current answer]".
- The value can be higher or lower than current -- both
↪  directions
  are valid.

Return JSON:
{"temporal_answer": "The value as it stood on {doc_date}",
 "reason": "Brief note on what period this covers"}
```

Additional context for misinformation documents For misinformation documents, we append a format-specific uncertainty context to the document-generation prompt. This context encourages the generated document to express the incorrect claim with subtle unreliability markers rather than obvious disclaimers. Two phrases are sampled uniformly without replacement from the pool corresponding to the document format, shown in Table 3.

```
Document date: {formatted_date}. This contains INCORRECT
↪  information
with subtle uncertainty markers. {phrase_1} {phrase_2} Keep
uncertainty natural and subtle, not obvious.
```

Additional context for temporal documents For temporal documents, we append a format-specific context block that makes the document read as authoritative at the time of writing. The document date is the only staleness signal. Two phrases are sampled uniformly from the format-specific pool in Table 4.

```
Document date: {formatted_date}. Write this as the current
↪  state of
affairs as of the document date above -- the author believes
↪  this
information is accurate right now. Do NOT reference it as
↪  historical,
archived, or outdated. Do NOT use phrases like 'back in',
'previously', or 'as of [earlier date]'. The date field is the
↪  only
staleness signal; the content must read as authoritative for
↪  its
time. {phrase_1} {phrase_2}
```

Format-specific document generators After the wrong-answer or temporal-answer prompt produces a perturbed answer, we call a format-specific generator to produce the final document. The default `word_limit` values are 300 for email, 300 for chat, and 500 for Markdown. JSONL outputs are parsed line by line. Markdown outputs are stripped of surrounding Markdown code fences when present.

Topic Variable Prompt:

```
Topic: {category}
Insight: {question} {answer}
Question: {question}
Answer: {answer}
Context: {additional_context}
```

Email Generator Prompt:

```
You are an email expert who creates professional emails with
↪  proper
structure and formatting.
Create a JSONL file that contains a series of email messages in
↪  the
mailboxes of persona: {persona.first_name} {persona.last_name},
role: {persona.role}, department: {persona.department},
responsibilities: {persona.responsibilities}.
The emails need to cover the following information:

{topic}

The emails should:
1. Be realistic email conversations that mention the answer
↪  naturally to the question
2. Follow this exact format for each email:
   {
       "type": "email",
       "id": "email_001",
       "from": "sender@company.com",
       "from_name": "Sender Name",
       "to": ["recipient@company.com"],
       "cc": [],
       "subject": "Email Subject",
       "date": "{formatted_date}T09:00:00-05:00"
       "body": "Email body content...",
       "folder": "inbox",
       "read": false,
       "attachments": []
   }
3. Have the persona as the sender or one of the recipients
4. Be approximately {word_limit} words in total across all
↪  emails
5. Use realistic names, email addresses, and subjects based on
↪  the topic
6. Use sequential email IDs
7. Use realistic recent dates
8. Use folders from: inbox, sent, drafts, spam, trash
9. Mark some emails as read and some as unread

Consider generating:
- Emails between team members discussing the topic
- Emails from management asking for specific information
- Internal announcements or updates

Wrap the email content in a JSONL code block.
```

Format	Phrase pool; two sampled per misinformation document
email	Use casual hedging such as “I think” or “from what I understand”; mention hearing the information from a colleague or in passing; use phrases such as “if I remember correctly” or “I believe”; frame the message as informal knowledge sharing rather than official communication; include a casual tone suggesting secondhand information.
chat	Use casual language such as “pretty sure” or “I think so”; frame the claim as something heard from someone else; include conversational uncertainty such as “might be” or “could be”; use informal phrasing that suggests incomplete knowledge; add hedging such as “afaik” or “last I heard”.
md	Include qualifying language such as “preliminary findings” or “initial assessment”; use phrases such as “based on available information” or “current understanding”; frame conclusions tentatively with “appears to” or “suggests that”; include a note that information is subject to verification; use conditional language such as “may” or “could” for key claims.

Table 3: Format-specific uncertainty phrase pools for misinformation documents. Two phrases are sampled uniformly without replacement for each generated misinformation document.

Format	Phrase pool; two sampled per temporal document
md	Write the document as if it were current at the time of writing; use present tense to describe the information as valid; make the document read as authoritative for its time period; include natural date references that match the document date.
email	Write the email naturally as if sent on that date; have the sender convey the information confidently; use normal business communication style for the time; include a natural email signature with the date context.
chat	Write the chat as a natural conversation from that time; have participants discuss the topic as current information; use casual workplace-chat language; include natural timestamp references.

Table 4: Format-specific context phrases for temporal documents. Two phrases are sampled uniformly per temporal document.

```
Company Context:
- {company_info_key_1}: {company_info_value_1}
- ...

Return only the complete email content in JSONL format.
```

Chat Generator Prompt

```
You are a MatterMost chat expert who creates professional chats
↪  with
proper structure, tables, and formatting. Create a JSONL file
↪  that
contains a series of chat messages in channels. The messages
↪  need to
cover the following information:

{topic}

The chat should:
1. Be a discussion between reasonable users on the topic and
↪  mention
   the answer to the question inherently
2. Contain realistic messages to teams/channels
3. Contain all necessary team, channel, and user settings
↪  before posts
4. Be approximately {word_limit} words in total
5. Use realistic names for people, teams, and channels
6. Contain a version object at the beginning of the file
7. Order objects as: version, teams, channels, users, posts
8. Include user credential information and team membership

Consider generating:
- Chats between customers/partners and support/sales/product
↪  teams
- Chats between team members
- Chats between employees and management

Wrap the chat content in a JSONL code block.

Company Context:
- {company_info_key_1}: {company_info_value_1}
- ...

Return only the complete chat content in JSONL format.
```

Markdown Document Generator Prompt:

```
You are a markdown expert who creates professional documents
↪  with
proper structure, tables, and formatting. Create a complete
↪  markdown
document given the following requirements.

{topic}

The document should:
1. Use proper markdown headings
2. Be approximately {word_limit} words in total
3. Include at least one table with relevant data
4. Use proper markdown formatting for lists, code blocks,
↪  blockquotes,
   and links where appropriate
5. Use professional terminology and relevant examples
6. Use realistic names for people
7. Use a random name for the document author
8. Include a title and author at the beginning
9. Never include file paths

Company Context:
- {company_info_key_1}: {company_info_value_1}
- ...

Return only the complete markdown content.
```

A.3 LLM-as-Judge Verification Prompts

Each generated misinformation, temporal, and noise document is independently checked by an LLM-as-judge using a fixed type-specific rubric. The judge returns structured JSON containing a binary verdict, a one-sentence rationale, and a scalar confidence in $[0, 1]$. The judge receives the question, the ground-truth answer, the claimed answer when applicable, the document date when applicable, and the first 2000 characters of the document text. Correct documents copied from DRBench

source files are auto-passed because they were already validated in DRBench; applying a 2000-character truncation to long source documents can otherwise create false negatives when the answer appears later in the file.

Correct-Document Rubric Prompt:

```
You are quality-checking a "correct" document in a factual QA
conflict benchmark.

Question: {question}
Ground-truth answer: {gt_answer}
Document excerpt:
{text}

Does this document explicitly state or clearly entail the
↪  ground-truth answer?

Respond with valid JSON only:
{
  "verdict": "PASS" or "FAIL",
  "reason": "<one sentence>",
  "confidence": <float 0.0-1.0>
}

PASS: the document contains the ground-truth answer verbatim or
↪  as an unambiguous paraphrase.
FAIL: the ground-truth answer cannot be inferred from this
↪  document alone.
```

Misinformation-Document Rubric Prompt:

```
You are quality-checking a "misinformation" document in a
↪  factual
QA conflict benchmark.

Question: {question}
Ground-truth answer (correct): {gt_answer}
Claimed answer in this document: {claimed_answer}
Document excerpt:
{text}

Verify two properties:
(A) DISTINCT -- the claimed answer is factually and
↪  specifically different from the ground-truth answer.
(B) PLAUSIBLE -- the claimed answer sounds credible to a naive
↪  reader and is not obviously absurd.
(C) NOT_LABELLED -- the document does not explicitly identify
↪  the claim as wrong, false, fabricated, or intentionally
↪  misleading.

Respond with valid JSON only:
{
  "verdict": "PASS" or "FAIL",
  "distinct": <true/false>,
  "plausible": <true/false>,
  "reason": "<one sentence>",
  "confidence": <float 0.0-1.0>
}

PASS requires distinct=true AND plausible=true.
```

Temporal-Document Rubric Prompt:

```
You are quality-checking a "temporal conflict" document in a
↪  factual
QA conflict benchmark.

Question: {question}
Current ground-truth answer: {gt_answer}
Historical answer in this document: {claimed_answer}
Document date: {document_date}
Document excerpt:
{text}

Verify two properties:
(A) DISTINCT -- the historical answer differs meaningfully from
↪  the
    current ground-truth answer.
(B) NOT_LABELLED -- the document does NOT contain explicit
↪  "archived",
    "outdated", or "superseded" labels. The document date is
    ↪  the only
    staleness signal; the text should read as authoritative for
    ↪  its time.

Respond with valid JSON only:
{
  "verdict": "PASS" or "FAIL",
  "distinct": <true/false>,
  "not_labelled": <true/false>,
  "reason": "<one sentence>",
  "confidence": <float 0.0-1.0>
}

PASS requires distinct=true AND not_labelled=true.
```

Noise-Document Rubric Prompt:

```
You are quality-checking a "noise" document in a factual QA
↪  conflict
benchmark.

Question: {question}
Ground-Truth answer (for reference): {gt_answer}
Document excerpt:
{text}

Does this document NOT answer the target question? That is, is
↪  it
genuinely about a different topic and does not contain or
↪  clearly imply the ground-truth answer?

Respond with valid JSON only:
{
  "verdict": "PASS" or "FAIL",
  "reason": "<one sentence>",
  "confidence": <float 0.0-1.0>
}

FAIL if the document contains the ground-truth answer or makes
↪  a strongly similar claim.
PASS if the document is about a different topic entirely.
```

A.4 PPTX Markdown Reconstruction

Since DRBench PPTX files do not include an exported text representation, we avoid using an LLM for deterministic reformatting and render the structured `file_dict.json` metadata as Markdown.

```
def render_pptx_as_markdown(fd: dict) -> str:
    lines = [f"# {fd.get('file_title', 'Presentation')}", ""]
    intro = fd.get("introduction", "").strip()
    if intro:
        lines += ["## Introduction", "", intro, ""]
    for section in fd.get("subsections", []):
        heading = section.get("heading", "").strip()
        content = section.get("content", "").strip()
        if heading:
            lines += [f"## {heading}", ""]
        if content:
            lines += [content, ""]
    conclusion = fd.get("conclusion", "").strip()
    if conclusion:
        lines += ["## Conclusion", "", conclusion, ""]
    return "\n".join(lines)
```

This reconstruction preserves the text fields but does not attempt to preserve slide layout or visual formatting.

A.5 Final Dataset

Table 5 summarizes the two released DRQA bundle configurations. Both configurations are derived from the same source examples. The @10 configuration adds one additional misinformation document, one additional temporal document, and three additional noise documents relative to @5.

	@5	@10
Correct	1	1
Misinformation	1	2
Temporal	1	2
Noise	2	5
Total documents per example	**5**	**10**
Examples in release	568	568
Examples at exact-k bundle size	568	564

Table 5: DRQA bundle configurations. All 568 examples in both configurations have the listed numbers of correct, misinformation, and temporal documents. For @10, four examples have fewer than five noise documents because the corresponding DRBench tasks have insufficient distinct distractors after deduplication or repeated noise-rubric rejection.

B DCCD Implementation Details

This appendix provides implementation details for Dual-Confidence Contrastive Decoding (**DCCD**), complementing the method description in §4.

B.1 Support Probe for Document-Level Confidence

For each question–document pair (x, d_i), we compute a document-level confidence score q_i before decoding. The goal of q_i is to estimate whether document d_i contains enough information to answer the question, independent of the current generation prefix.

Prompt Template For each retrieved document d_i, we use the following confidence probe:

```
{document}

Question: {question}
Does this document contain enough information to answer the
↪  question? (yes/no)
Answer:
```

Yes/No Verbalizers Let $\mathcal{Y}_+$ denote the set of yes verbalizer token ids and $\mathcal{Y}_-$ denote the set of no verbalizer token ids. In practice, we construct these sets by tokenizing common case and leading-space variants:

$$\{\texttt{yes}, \texttt{Yes}, \texttt{YES}, \ \texttt{yes}, \ \texttt{Yes}, \ \texttt{YES}\}$$

for yes, and analogously for no. We keep only variants that are represented as a single token by the tokenizer. Leading-space variants are included because the BPE (Sennrich et al., 2016) for `Qwen3.5` and the SentencePiece tokenizers (Kudo and Richardson, 2018) encode the first content token of an assistant response with an initial whitespace marker.

Score Computation We compute the document-level confidence score by re-normalizing the probability mass assigned to yes and no verbalizers. In practice, we compute the same quantity in logit space for numerical stability. Let $\mathbf{z}_i^{\text{sup}}$ be the final-position logits produced by the support probe for document d_i. We compute

$$\ell_i^+ = \text{logsumexp}_{v \in \mathcal{Y}_+} \mathbf{z}_i^{\text{sup}}[v], \tag{11}$$

$$\ell_i^- = \text{logsumexp}_{v \in \mathcal{Y}_-} \mathbf{z}_i^{\text{sup}}[v], \tag{12}$$

$$q_i = \sigma(\ell_i^+ - \ell_i^-), \tag{13}$$

where σ is the logistic sigmoid. This is equivalent to the yes/no renormalization in Eq. 1. All document probes for a query are batched into a single forward pass, so the support-probe stage is computed once before decoding.

B.2 Token-Level Confidence

At decoding step t, let $\mathbf{z}_t^i \in \mathbb{R}^V$ denote the next-token logits conditioned on the question x, document d_i, and the current output prefix $y_{<t}$. Let $\mathcal{T}_{i,t}$ be the set of top-k token indices under $\mathbf{z}_t^i$. We compute token-level confidence by first mapping these top-k logits to positive evidence values and then measuring the expected entropy of the induced distribution.

Top-k Evidence Following the Dirichlet uncertainty framework of Malinin and Gales (2018), we convert each selected logit into a positive concentration parameter for a lightweight Dirichlet approximation:

$$\alpha_{(i,t,v)} = \text{softplus}\big(\mathbf{z}_t^i[v]\big) + 1, \qquad v \in \mathcal{T}_{(i,t)}. \tag{14}$$

The softplus transformation keeps the evidence positive without exponentiating logits, and the $+1$ shift ensures that each concentration parameter is at least one. We define the total concentration and the corresponding Dirichlet mean as

$$S_{(i,t)} = \sum_{v \in \mathcal{T}_{i,t}} \alpha_{(i,t,v)}, \qquad \bar{p}_{(i,t,v)} = \frac{\alpha_{(i,t,v)}}{S_{(i,t)}}. \tag{15}$$

Expected Entropy We take the top-k next-token probabilities (Nguyen et al., 2026) and treat them as a categorical distribution sampled from a Dirichlet posterior:

$$\boldsymbol{\pi}_t^i \sim \text{Dir}\big(\boldsymbol{\alpha}_t^i\big). \tag{16}$$

The raw token-level uncertainty is the expected entropy of this sampled categorical distribution:

$$\widetilde{H}_k(\mathbf{z}_t^i) = \mathbb{E}_{\boldsymbol{\pi}_t^i \sim \text{Dir}(\boldsymbol{\alpha}_t^i)} \left[H(\boldsymbol{\pi}_t^i) \right]. \tag{17}$$

This expectation has the closed form

$$\widetilde{H}_k(\mathbf{z}_t^i) = \sum_{v \in \mathcal{T}_{(i,t)}} \bar{p}_{(i,t,v)} \left[\psi(S_{(i,t)} + 1) - \psi(\alpha_{(i,t,v)} + 1)\right], \tag{18}$$

where ψ is the digamma function:

$$\psi(u) = \frac{d}{du} \log \Gamma(u) = \frac{\Gamma'(u)}{\Gamma(u)}. \tag{19}$$

The uncertainty $\widetilde{H}_k$ is small when the document-conditioned evidence is concentrated on a few next-token candidates, and larger when the evidence is spread across many candidates.

Normalized Confidence To make the score comparable across documents and bounded in $[0, 1]$, we normalize by the maximum entropy over a top-k support:

$$H_k(\mathbf{z}_t^i) = \frac{\widetilde{H}_k(\mathbf{z}_t^i)}{\log k}. \tag{20}$$

The token-level confidence used in the main decoding rule is then

$$c_{(i,t)} = 1 - H_k(\mathbf{z}_t^i). \tag{21}$$

Thus, $c_{(i,t)}$ is high when document d_i yields a sharp next-token prediction at step t, and low when the model remains uncertain about how to continue from that document. Unlike the document-level confidence q_i, which is computed once per question–document pair, $c_{(i,t)}$ is recomputed at every decoding step.

B.3 Pseudocode

Algorithm 1 summarizes the complete procedure for one query.

Algorithm 1 DCCD decoding for one query.

Require: Question x, retrieved documents $\mathcal{D} = \{d_1, \ldots, d_n\}$, model p_θ, tokenizer, max new tokens $T_{\max}$, top-k value k

// Stage 1: document-level confidence
for $i = 1, \ldots, n$ **do**
  Construct support probe prompt $\pi_{\text{sup}}(x, d_i)$
end for
Run a batched forward pass over all support probes
for $i = 1, \ldots, n$ **do**
  Compute q_i using Eq. (13)
end for
// Stage 2: decoding
Initialize generated prefix $y_{<1} = \emptyset$
for $t = 1, \ldots, T_{\max}$ **do**
  Compute full-context logits $\mathbf{z}_t^{\text{full}}$
  for $i = 1, \ldots, n$ **do**
    Compute document-conditioned logits $\mathbf{z}_t^i$
    Compute $c_{(i,t)}$ using Eqs. (14)–(21)
    $s_{(i,t)} \leftarrow q_i + c_{(i,t)}$
  end for
  $i_t^+ \leftarrow \arg\max_i s_{(i,t)}$
  $i_t^- \leftarrow \arg\min_i s_{(i,t)}$
  $\epsilon_t \leftarrow s_{(i_t^+,t)} - s_{(i_t^-,t)}$
  $\mathbf{z}_t^{DCCD} \leftarrow \mathbf{z}_t^{\text{full}} + \epsilon_t \left(\mathbf{z}_t^{i_t^+} - \mathbf{z}_t^{i_t^-}\right)$
  Select y_t from $\text{softmax}(\mathbf{z}_t^{DCCD})$
  Append y_t to the generated prefix
  if y_t is an end-of-sequence token **then**
    break
  end if
end for
return decoded answer

B.4 Computational Cost

The document-level support probe is computed once per query. All n document probes can be batched into a single forward pass, producing the n values of q_i. This cost is amortized over the entire generation.

At each decoding step, DCCD requires one full-context forward stream and n single-document forward streams. Thus, for n retrieved documents, the decoding stage uses $n + 1$ streams per token. The token-level confidence computation adds only $O(nk)$ operations for top-k extraction and digamma evaluations, which is negligible compared with the model forward passes for the values of k used in our experiments.

The implementation can use the model's standard KV cache for each stream. With caching, each decoding step only extends the existing streams by one token. The memory footprint scales linearly with the number of streams, and the runtime scales approximately linearly with $n + 1$ relative to full-context greedy decoding, plus the one-time support-probe pass.

C Experimental Details

This appendix provides additional experimental details that describe the datasets, baselines, models, and headline evaluation protocol. Here, we record the implementation-level choices needed to repro-

duce the experiments, including dataset formatting, prompting, decoding configurations, and hyperparameters.

C.1 Dataset Details

Table 6 summarizes the datasets used in our experiments. Natural Questions (NQ) and TriviaQA (TQA) use the preprocessing of Izacard and Grave (2021), which provides DPR-retrieved passages with title and text fields. PopQA uses the original release from Mallen et al. (2023). RetrievalQA follows Zhang et al. (2024) and is converted into the same document-list format. DRQA is our factual-conflict benchmark derived from DRBench, which consists of 568 internal enterprise facts.

Dataset	# Examples	Avg @5	Avg @10
NQ	3,610	526	1,042
TQA	11,313	527	1,041
PopQA	4,267	521	1,036
RetrievalQA	2,785	443	1,003
DRQA	568	2,039	4,002

Table 6: Datasets statistics used in our experiments. We include the average number of words of the input for the @5 and @10 settings.

C.2 Prompt Templates

All methods use the same answer-generation prompt format. We adopt the same prompt template from prior work (Shi et al., 2024; Jin et al., 2024a) with the zero-shot and contextualized prompts displayed below.

Zero-shot Prompt:

```
Question: {question}
Answer:
```

Contextualized Prompt:

```
Write a high-quality concise answer for the given question
↪ using only the provided search results (some of which might
↪ be irrelevant).

{search_results}

Question: {question}
Answer:
```

All prompts are wrapped with the default chat template provided by the corresponding Hugging Face tokenizer. We use the chat template both for answer generation and for the document-level support probe in DCCD, so confidence estimation and final decoding are performed under the same instruction-following format. The per-document prompts follow the same template as the contextualized prompt following Jin et al. (2024a).

C.3 Baseline Decoding Configurations

We use greedy decoding with temperature 0 and a maximum generation length of 60 new tokens for all methods. To reduce artifacts from recall-oriented string-match metrics, we generate and evaluate a single-sentence answer. Since `strem` rewards the presence of a normalized gold string anywhere in the prediction, long generations that copy retrieved passages or list multiple candidate answers can be artificially rewarded even when the model does not clearly commit to the correct answer. Single-sentence generation makes the comparison across decoding methods more meaningful by forcing each method to produce a concise prediction. Unless otherwise stated, hyperparameters are fixed across all datasets and models. We describe the hyperparameter settings for all decoding strategies used in our experiments and summarize in Table 7.

Zero-shot Decoding The zero-shot baseline generates from the question alone, without any retrieved documents. It is included to estimate how much of each dataset can be answered from parametric memory alone.

Full-Context Decoding The full-context baseline decodes directly from the logits conditioned on the question and the full retrieved document set:

$$\mathbf{z}_t = \mathbf{z}_t^{\text{full}}.$$

CAD Context-Aware Decoding (CAD) contrasts full-context logits against no-context logits:

$$\mathbf{z}_t^{\text{CAD}} = (1+\alpha)\mathbf{z}_t^{\text{full}} - \alpha \mathbf{z}_t^{\text{none}}.$$

We use $\alpha = 0.2$ based on the results reported by Shi et al. (2024).

AdaCAD Adaptive Context Aware Decoding (AdaCAD) replaces the fixed CAD weight with a token-level adaptive weight based on the Jensen-Shannon divergence between the full-context and no-context distributions:

$$\alpha_t = \text{JSD}\left(p_t^{\text{full}} \| p_t^{\text{none}}\right),$$

$$\mathbf{z}_t^{\text{AdaCAD}} = (1+\alpha_t)\mathbf{z}_t^{\text{full}} - \alpha_t \mathbf{z}_t^{\text{none}}.$$

We set the minimum weight floor to 0.0 following Wang et al. (2025a).

DVD Dynamic Contrastive Decoding (DVD) is a document-aware contrastive decoding baseline for multi-document QA. We use the static-weight configuration with $\beta = 0.25$, $\gamma = 0.2$, and document top-$k = 10$. In early experiments, we find that dynamic β performs poorly, so we followed the suggested configuration from Jin et al. (2024a). The method selects document-conditioned streams according to its entropy-based criterion and applies a document-level contrast together with a context-prior contrast.

CoCoA Confidence- and Context-Aware Adaptive Decoding (CoCoA) uses a confidence-aware interpolation between no-context and full-context predictions. We use the paper-faithful configuration with Rényi divergence order $\alpha = 0.5$, entropy-gap weight $\gamma = 1.0$, peakedness amplifier $z = 5.0$, and numerical stability constants $\delta = 10^{-8}$ and $\varepsilon = 10^{-12}$ following Khandelwal et al. (2025).

DCCD Our proposed Dual-Confidence Contrastive Decoding (DCCD) has the advantage of being relatively hyperparameter-free, with the only hyperparameter being the top-k logits used for computing token-level confidence, which we set to $k = 10$ throughout all our experiments.

Method	Hyperparameter	Value
All methods	Temperature	0
All methods	Decoding strategy	greedy
All methods	Max new tokens	60
CAD	α	0.2
AdaCAD	JSD floor	0.0
DVD	β	0.25
DVD	γ	0.2
DVD	Document top-k	10
CoCoA	Rényi order	0.5
CoCoA	Entropy-gap weight	1.0
CoCoA	Peakedness amplifier	5.0
CoCoA	Numerical floors	$\delta = 10^{-8}, \varepsilon = 10^{-12}$
DCCD	top-k	10

Table 7: Main decoding hyperparameters used in the experiments. Hyperparameters are fixed across datasets and models.

C.4 Models and Inference Backend

We evaluate the three instruction-tuned language models through Hugging Face `transformers` (Wolf et al., 2020), namely, `Qwen3.5-2B`[2] `Qwen3.5-9B`[3], and `Phi-3-medium`[4]. All models are loaded with their default tokenizers and chat templates. We use bfloat16 inference when supported. Tokenizer padding is set to the left so the final-position logits are aligned across batched streams.

For contrastive decoding methods, each decoding step may require multiple forward streams. CAD, AdaCAD, and CoCoA use no-context and full-context streams. DVD and DCCD additionally use document-conditioned streams. For DCCD, the support probe is computed once before decoding, while the full-context and document-conditioned logits are computed during each decoding step. Streams are batched whenever possible.

C.5 Evaluation Metrics

Normalized string exact match. For NQ, TriviaQA, PopQA, and RetrievalQA, we use normalized string exact match, denoted `strem`. We first extract the generated answer sentence and then normalize both the prediction and gold answers using the standard QA normalization procedure: lowercasing, removing punctuation, removing articles, and collapsing whitespace. A prediction is counted as correct if any normalized gold answer matches the normalized prediction by exact or substring match, following common retrieval-augmented QA evaluation practice.

LLM-as-a-judge for DRQA. For DRQA, the reference answer is a free-form enterprise insight rather than a short canonical span. We therefore evaluate semantic correctness with an LLM judge. The judge receives the question, the reference answer, and the model prediction, and returns a binary correctness label. A prediction is marked correct if it conveys the same answer as the reference at the required level of specificity. Numerical and date answers must match the reference detail; refusals are marked incorrect unless the reference answer itself indicates that the information is unavailable.

We use deterministic judge decoding with temperature 0. Judge outputs are parsed as JSON with a boolean correctness field. If the judge response cannot be parsed, the example is conservatively marked incorrect. Below is the prompt template for our judge using `anthropic/claude-opus-4-5` (Anthropic, 2025) through OpenRouter API[5].

```
You are a strict but fair grader for short-answer
question answering. Decide whether the model answer
```

[2] https://huggingface.co/Qwen/Qwen3.5-2B
[3] https://huggingface.co/Qwen/Qwen3.5-9B
[4] https://huggingface.co/microsoft/Phi-3-medium-128k-instruct
[5] https://openrouter.ai

```
matches the reference answer.

Question:
{question}

Reference answer:
{reference_answer}

Model answer:
{prediction}

Mark the model answer correct if and only if it
conveys the meaning of the reference answer.

Rules:
1. Surface form does not matter if the meaning is
   equivalent.
2. Numerical and date answers must match the
   reference at the level of detail provided.
3. A refusal or "not specified" answer is incorrect
   unless the reference answer also says the
   information is unavailable.
4. Extra context is acceptable only if the answer
   clearly commits to the correct value.
5. If the answer gives a wrong primary value but
   mentions the correct value only incidentally,
   mark it incorrect.

Return exactly one JSON object:
{"correct": true_or_false, "reason": "<short reason>"}
```

C.6 Hardware and Runtime

All experiments can be run on a single NVIDIA H100 80GB GPU. Runtime depends on model size, dataset size, retrieval depth, and decoding method. Full-context decoding is the cheapest setting because it uses a single generation stream. CAD, ADACAD, and COCOA require additional no-context streams. DVD and DCCD require document-conditioned streams. We report the average wall clock runtime in Table 8.

	Full-Context		CAD / ADACAD / COCOA		DVD / DCCD	
Dataset	@5	@10	@5	@10	@5	@10
DRQA	5	8	8	13	20	38
RetrievalQA	12	20	20	35	45	73
NQ	15	25	25	45	60	108
PopQA	18	30	30	55	70	130
TriviaQA	45	80	80	145	180	340

Table 8: Representative wall-clock runtime for each dataset, retrieval depth, and decoding-method family on a single NVIDIA H100 80GB GPU. Runtime is reported for `Qwen3.5-2B` in rounded minutes.

For $n$retrieved documents, the per-token cost of DCCD scales approximately linearly with $n + 1$: one stream for the full retrieved context and n streams for individual documents. The token-level confidence computation is inexpensive relative to the model forward pass because it only requires top-k extraction and digamma evaluations over a small number of tokens. The document-level support probe is computed once per question–document pair and is amortized over the generated answer.

D Qualitative Examples

We present representative qualitative examples for `Qwen3.5-2B` with top-10 retrieved documents. Each example shows the first sentence generated by each decoding method. Predictions are truncated to 220 characters when necessary. Correct predictions are shaded green and incorrect predictions are shaded red.

For NQ, TriviaQA, RetrievalQA, and PopQA, correctness follows the first-sentence `str-em` metric described in subsection C.5. For DRQA, correctness is determined by the LLM-as-a-judge metric. To avoid `str-em` false positives from long generations that enumerate multiple candidate answers, we additionally filter qualitative examples to predictions of at most 240 characters that commit to a single value when the gold answer is numeric and contain no refusal phrase such as *"no information"* or *"cannot determine"*.

D.1 DRQA

Q (qid 194): *How did Jason Wong respond to a simulated phishing incident in March 2025?*
Gold: In March 2025, Jason Wong coordinated with the IT helpdesk and quickly contained a simulated phishing attack, reducing average internal response time to under 25 minutes.

Method	**Prediction**
full	"Based on the provided documents, there is a discrepancy regarding the specific date of the incident."
CAD	"Based on the provided documents, there is a discrepancy regarding the specific month of the incident:"
ADACAD	"Based on the provided documents, there is a discrepancy regarding the specific month of the incident:"
DVD	"Based on the provided documents, there is a discrepancy regarding the specific month of the incident and the outcome:"
COCOA	"Based on the provided search results, there is a contradiction regarding the specific details of Jason Wong's response to a simulated phishing incident in March 2025."
DCCD	"Jason Wong responded to the simulated phishing incident in March 2025 by coordinating with the IT helpdesk to contain the threat expeditiously."

D.2 Natural Questions

Q (qid 29): *in which sea pearl is found in india*
Gold: the Indian Ocean

Method	Prediction
full	"Based on the provided search results, the sea where pearls are found in India is the Andaman Sea."
CAD	"Based on the provided search results, India's major mineral resources include tin deposits off the coasts of Malaysia and Thailand, which are located in the Andaman Sea. Additionally, the Andaman Sea is noted..."
ADACAD	"Based on the provided search results, India's major mineral resources include tin deposits off the coasts of Malaysia and Thailand, which are located in the Andaman Sea. Additionally, the Andaman Sea is noted..."
DVD	"Based on the provided search results, the sea pearl found in India is the String of Pearls."
COCOA	"The sea!"
DCCD	"Based on the provided search results, the sea where pearls are found in India is the Indian Ocean."

Q (qid 1491): *who wrote it's gonna take a lot of love*
Gold: Neil Young

Method	Prediction
full	"Based on the provided search results, there is no mention of a song titled "It's gonna take a lot of love" or its author."
CAD	"Based on the provided search results, there is no document that states who wrote the song "It's Gonna Take a Lot of Love.""
ADACAD	"Based on the provided search results, there is no document that explicitly states who wrote the song "It's Gonna Take a Lot of Love.""
DVD	"Based on the provided search results, there is no document that states who wrote the song "It's Gonna Take a Lot of Love.""
COCOA	"The song "It's Gonna Take a Lot of Love" was written by Linda Ronstadt and Lotta Love."
DCCD	"The song "Lotta Love" was written and recorded by Neil Young."

Q (qid 2783): *who is the all time leading scorer in ncaa tournament history*
Gold: Pete Maravich

Method	Prediction
full	"Based on the provided search results, the all-time leading scorer in NCAA tournament history is Gary Gait, who holds the record for the most goals in the NCAA tournament with 16 goals. This record was previous..."
CAD	"Based on the provided search results, Phil Latreille is the all-time leading scorer in NCAA Division I history with 4,045 points. He holds the record for both career points and the most goals in a single seaso..."
ADACAD	"Based on the provided search results, the all-time leading scorer in NCAA tournament history is Gary Gait, who holds the record for the most goals in the NCAA tournament with 16 goals. This record was previous..."
DVD	"Based on the provided search results, Don MacLean is the all-time leading scorer in NCAA tournament history, having scored 2,608 points. This record was broken by him in 2013, surpassing the previous record ho..."
COCOA	"The all-time leading scorer in NCAA tournament history is Matt Ward."
DCCD	"Based on the provided search results, the answer is Pete Maravich."

Q (qid 2792): *where do they film young and the restless*
Gold: CBS Television City

Method	Prediction
full	"The series "The Young and the Restless" was originally filmed in Los Angeles, California, with production taking place there while the show aired in New York."
CAD	"The show "The Young and the Restless" was originally filmed in Los Angeles, California, with production taking place there while writers worked from Chicago."
ADACAD	"The show "The Young and the Restless" was originally filmed in Los Angeles, California, with production taking place there while writers worked from Chicago."
DVD	"Based on the provided search results, the soap opera "The Young and the Restless" was filmed in Los Angeles, California."
COCOA	"The 2017 film *Young and the Restless* was filmed in Los Angeles, California."
DCCD	"They film "The Young and the Restless" at CBS Television City, specifically studios 41 and 43 in Hollywood."

Q (qid 1600): *where does stevia in the raw come from*
Gold: the plant species *Stevia rebaudiana*

Method	Prediction
full	"Stevia In The Raw is a zero-calorie sweetener developed by Cumberland Packing Corporation."
CAD	"Stevia in the Raw is a stevia-based low-calorie sweetener developed by Cumberland Packing Corporation."
ADACAD	"Stevia in the Raw is a stevia-based low-calorie sweetener developed by PureVia, a product created jointly by PepsiCo and Whole Earth Sweetener Company."
DVD	"Stevia in the Raw is a sweetener developed by Cumberland Packing Corporation that is made from stevia leaf extract, erythritol, and natural flavors."
COCOA	"Stevia In The Raw is a dietary supplement that!"
DCCD	"Stevia in the Raw comes from the leaves of the plant species *Stevia rebaudiana*, native to Brazil and Paraguay."

D.3 TriviaQA

Q (qid 7277): *The name of a knot on a lasso is also the name of a Japanese motor manufacturer. What is the name?*
Gold: HONDA

Method	Prediction
full	"The name is Miyata."
CAD	"The name is Weds, or WEDS."
ADACAD	"Miyata"
DVD	"The name is Weds, or WEDS."
COCOA	"The name is !WORK."
DCCD	"The name is Honda."

Q (qid 8593): *In imperial measure, how many gills are there in a gallon?*
Gold: 32

Method	Prediction
full	"There are 16 imperial gills in an imperial gallon."
CAD	"There are 16 imperial gills in an imperial gallon."
ADACAD	"There are 16 imperial gills in an imperial gallon."
DVD	"There are 160 imperial gills in an imperial gallon."
COCOA	"In the imperial system, there are 16! (sixteen) gills in one gallon."
DCCD	"In the imperial system, a gallon is equal to 32 gills."

Q (qid 4705): *How many US states border Lake Superior?*
Gold: three

Method	Prediction
full	"Based on the provided documents, two U.S. states border Lake Superior: Minnesota and Wisconsin."
CAD	"Based on the provided documents, two U.S. states border Lake Superior: Michigan and Wisconsin."
ADACAD	"Based on the provided documents, two U.S. states border Lake Superior: Minnesota and Wisconsin."
DVD	"Based on the provided search results, two U.S. states border Lake Superior: Minnesota and Wisconsin."
COCOA	"There are 4 US states that border Lake Superior:"
DCCD	"Three U.S. states border Lake Superior: Minnesota, Wisconsin, and Michigan."

Q (qid 5136): *In which "Star Wars" episode does Anakin Skywalker transform into the black-suited Darth Vader?*
Gold: *Star Wars: Episode III – Revenge of the Sith*

Method	Prediction
full	"Anakin Skywalker transforms into the black-suited Darth Vader in the "Star Wars" episode "Star Wars: Episode IV – A New Hope," which was released in 1977."
CAD	"Based on the provided search results, Anakin Skywalker transforms into the black-suited Darth Vader in the animated series "Star Wars: Clone Wars.""
ADACAD	"Based on the provided search results, Anakin Skywalker transforms into the black-suited Darth Vader in the animated series "Star Wars: Clone Wars.""
DVD	"Based on the provided search results, the specific episode where Anakin Skywalker transforms into Darth Vader is not explicitly named as a single title in the text."
COCOA	"![](https://![](https://![](https://![]..."
DCCD	"Anakin Skywalker transforms into the black-suited Darth Vader in "Star Wars: Episode III – Revenge of the Sith.""

Q (qid 8453): *What most-rejected 1974 bestseller describes the 17-day journey of the author, who calls himself Phaedrus, and his son Chris from Minnesota to California?*
Gold: *Zen and the Art of Motorcycle Maintenance* by Robert M. Pirsig

Method	Prediction
full	"The bestseller is *Zen and the Art of Motorcycle Maintenance* by Robert Pirsig."
CAD	"The 1974 bestseller that describes the 17-day motorcycle journey from Minnesota to Northern California, authored by Richard Pirsig and fictionalized as an autobiography of Phaedrus and his son Chris, is..."
ADACAD	"Based on the provided search results, there is no bestseller that describes an 17-day journey from Minnesota to California by an author named Phaedrus."
DVD	"The bestseller is *Zen and the Art of Motorcycle Maintenance* by Robert Pirsig."
COCOA	"The book you are referring to is *The Road to California*!"
DCCD	"The bestseller is *Zen and the Art of Motorcycle Maintenance* by Robert M. Pirsig."