# Clarify-Then-Search: A Clarification Benchmark for Deep Search with End-to-End Nugget Restoration

Deqiang Huang*
University of Science and Technology of China
Hefei, Anhui, China
deqianghuang@mail.ustc.edu.cn

Jingbo Zhou†
Frontier Research Department
Baidu Inc.
Beijing, China
zhoujingbo@baidu.com

Xinjiang Lu
Frontier Research Department
Baidu Inc.
Beijing, China
luxinjiang@baidu.com

Tong Xu†
University of Science and Technology of China
Hefei, Anhui, China
tongxu@ustc.edu.cn

Hua Wu
Frontier Research Department
Baidu Inc.
Beijing, China
wu_hua@baidu.com

Enhong Chen†
University of Science and Technology of China
Hefei, Anhui, China
cheneh@ustc.edu.cn

## Abstract

Deep search is brittle on underspecified user queries: missing constraints (e.g., time, location, scope, or definitions) often lead to retrieval drift and incomplete answers. Although LLMs can ask clarifying questions, existing evaluations of clarification for deep search remain limited. We introduce a *clarify-then-search* benchmark to evaluate the ability of LLMs to ask clarifying questions that facilitate deep search. Built on real-world query data from the Baidu search engine, the benchmark contains 518 curated instances that target the scenarios where clarification is most needed: each instance provides an intent query $q^*$ (`fused_query`) and a corresponding underspecified query $q$ (`blurred_query`). We run a deep-search backbone (we use WebDancer in this study) *once* per $q^*$ to archive evidence and construct a *static* golden reference as weighted, evidence-grounded nuggets with traceable source identifiers. At evaluation time, a Clarifier asks $k \in \{1, 2, 3\}$ questions; a closed-book User Answerer replies using only information explicitly stated in $q^*$ (otherwise returning `unknown`); and a closed-book Rewriter produces a rewritten query $\hat{q}$ using only $q$ and the elicited Q&A pairs. WebDancer then executes on $\hat{q}$, and we score end-to-end utility by `restore_score_100` (0–100), a weighted nugget-recall score with partial credit against the static gold.

Across all evaluated models, clarification improves over the no-interaction baseline at $k = 1$, and larger budgets ($k = 2, 3$) yield further gains. Notably, GPT-5.2 achieves the highest mean score at $k = 1$, while ERNIE-4.5-Turbo-128K leads the evaluated open-weight models at $k = 1$ and becomes the overall top-performing model at $k = 3$, surpassing GPT-5.2, Claude-Sonnet-4.5, and Gemini-2.5-Pro. Interaction diagnostics further reveal a consistent failure mode under strict closed-book clarification: many systems over-ask region-only questions that are often unanswerable from the intent and thus elicit `unknown`, producing low-yield interactions despite additional turns. Overall, the benchmark enables leakage-resistant and reproducible evaluation of clarify-then-search pipelines, while supporting fine-grained analyses of question utility, answerability, and clarification budget effects in deep search. All data and code are available on the project page: https://clarify-then-search.github.io/.

## CCS Concepts

• **Computing methodologies** → **Natural language processing**; • **Information systems** → **Information retrieval**; *Question answering*.

## Keywords

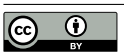
Deep Search Agents, Conversational Search, Clarifying Questions, Query Rewriting, Tool-augmented Retrieval, Nugget-based Evaluation, Evidence-grounded Benchmarks

**ACM Reference Format:**
Deqiang Huang, Jingbo Zhou, Xinjiang Lu, Tong Xu, Hua Wu, and Enhong Chen. 2026. Clarify-Then-Search: A Clarification Benchmark for Deep Search with End-to-End Nugget Restoration. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '26), August 09–13, 2026, Jeju Island, Republic of Korea.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3770855.3817586

## 1 Introduction

Deep search systems are increasingly deployed for complex information needs, yet they remain brittle when user queries are ambiguous or underspecified. In real search logs, users frequently omit critical constraints such as temporal scope, geographic region, entity disambiguation, numeric thresholds, or even the intended definition of a concept. Compared with standard retrieval, underspecification in deep search has larger downstream consequences: an early ambiguity can mislead search planning, waste tool-call budget on low-yield exploration, trigger off-target evidence visits, and propagate into incomplete or incorrect final answers. This motivates a *clarify-then-search* paradigm: ask targeted clarification questions, incorporate user feedback into a rewritten query, and then execute deep search more efficiently and accurately.

*This work was done when the first author was an intern in the Big Model Frontier Research Department of Baidu Inc., under the supervision of Jingbo Zhou.
†Corresponding authors.




*KDD '26, Jeju Island, Republic of Korea*

ACM ISBN 979-8-4007-2259-2/2026/08
https://doi.org/10.1145/3770855.3817586

Despite growing interest in clarification question generation, progress is bottlenecked by evaluation. First, many prior benchmarks focus on question-level properties, such as fluency, plausibility, or topical relevance, which are weak proxies for whether clarification actually improves downstream search utility. Second, standard retrieval-oriented evaluation captures only part of the effect of clarification. In deep search, clarification affects not only document recall, but also planning, evidence browsing, aggregation, and final answer completeness over multi-faceted information needs. Third, end-to-end evaluations can suffer from *intent leakage*: rewriting or retrieval stages may indirectly observe oracle intent, such as the original clear query or ground-truth slots, making it difficult to attribute gains to clarification quality.

As a result, it remains unclear whether stronger clarification behavior translates into measurable improvements under realistic information constraints. To address these issues, our benchmark (i) evaluates the clarification ability of LLMs by downstream utility in an end-to-end clarify→rewrite→deep-search pipeline, (ii) enforces a closed-book protocol to prevent intent leakage, and (iii) uses a static nugget-based gold target to enable reproducible comparisons. In this benchmark, *information gain* is used in an operational sense: a clarification question is useful if it elicits intent-grounded information that can be safely incorporated into the rewrite and ultimately improves downstream nugget restoration.

Based on real-world queries from the Baidu search engine, we introduce a fully open benchmark[1] that evaluates clarification by *end-to-end utility* in a fixed deep-search pipeline. In this study, the pipeline is powered by WebDancer [26], although other deep-search backbones can also be used. Our benchmark contains 518 curated instances that target scenarios where clarification is most needed: each instance includes an underspecified query (`blurred_query`) paired with its underlying intent query (`fused_query`). For each intent, we build a *static* golden reference by running WebDancer once on `fused_query` and extracting evidence-grounded nuggets traceable to retrieved sources. At evaluation time, a Clarifier generates $k$ clarification questions ($k \in \{1, 2, 3\}$), receives constrained answers from a closed-book user simulator, rewrites the query, and runs WebDancer on the rewritten query. We measure end-to-end utility by how well the final output restores the static nuggets, reported as `restore_score_100`, i.e., weighted nugget recall (0–100) with partial credit for `partial` coverage (Section 4.4).

A central design goal is to prevent intent leakage while preserving a realistic interaction pattern for LLM clarification. We therefore adopt a two-phase closed-book protocol. The **User Answerer (UA)** sees only `fused_query` and the current clarification question, and must answer *only what is asked*; if the intent does not explicitly specify the requested attribute, UA outputs `unknown`. The **Rewriter** sees only `blurred_query` and the clarification Q&A pairs, and must rewrite without introducing new entities or constraints; when an answer is `unknown`, it must not be forcefully concretized. This separation ensures that any additional constraint in the rewritten query must come from information actually elicited through clarification, rather than being copied from the hidden intent. It also exposes intermediate interaction signals, such as answerability, multi-turn no-signal probability, and question-type bias, for diagnosis beyond the final score.

Our results show that (i) all Clarifiers improve over the no-interaction baseline at $k = 1$, and (ii) larger clarification budgets generally yield further gains. We further observe that GPT-5.2 achieves the highest mean `restore_score_100` with a single clarification question ($k = 1$). Among the evaluated open-weight models, ERNIE-4.5-Turbo-128K leads at $k = 1$, outperforming Qwen3-235B-A22B-Instruct, Kimi-K2-Instruct, and DeepSeek-V3.2. ERNIE-4.5-Turbo-128K further becomes the overall top-performing model at $k = 3$, surpassing GPT-5.2, Claude-Sonnet-4.5, and Gemini-2.5-Pro.

However, intermediate diagnostics reveal systematic failure modes, especially a bias toward region questions that often elicit `unknown` under the closed-book constraint, showing that reliably improving deep search through clarification remains challenging. Additional robustness analysis further shows that while absolute scores vary across coverage judges, the main improvement trend remains stable across ERNIE-, GPT-, and Claude-based judges.

Our contributions are summarized as follows:

- **Dataset and Task.** We formalize deep-search clarification as an end-to-end benchmark from underspecified queries to nugget restoration, and release a dataset of 518 curated instances derived from real-world Baidu search queries with standardized evaluation outputs.
- **Leakage-Free Protocol.** We introduce a two-phase closed-book evaluation protocol that prevents rewrite-stage exposure to oracle intent, enabling controlled comparisons of Clarifiers for $k \in \{1, 2, 3\}$.
- **Static Golden Nuggets and Metric.** For each intent query, we archive an evidence-grounded static golden reference with weighted nuggets and score systems by weighted nugget recall with partial credit (`restore_score_100`).
- **Diagnostic Signals.** The benchmark makes intermediate interaction behavior measurable, including UA answerability, multi-turn all-`unknown` rate, and question-type stratification, enabling analyses of why clarification succeeds or fails beyond final scores.
- **Reproducibility Artifacts.** We release the dataset, pipeline scripts, prompts, and per-instance tool traces/snippets needed to reproduce golden construction and evaluation against a fixed target.

## 2 Related Work

**Clarifying Questions in Conversational Search.** Prior work studies how systems ask clarifying questions when an initial query is underspecified. Benchmarks and datasets such as Qulac and MIMICS support modeling clarification behavior and its impact on retrieval [1, 32, 33]. Recent resources further broaden this line with challenge-style clarification datasets and offline/online evaluation protocols for multi-turn clarification [12, 21]. Other work also investigates when and how to ask (or avoid) clarifying questions [10], including the role of negative feedback and interaction signals [4]. These efforts often emphasize question quality or short-horizon retrieval effects. In contrast, our benchmark evaluates clarification by *end-to-end utility* (clarify → rewrite → deep search → nugget restoration) under a controlled protocol.

[1] Project page: https://clarify-then-search.github.io/.

**Query Rewriting for Conversational Search.** Query rewriting converts contextual or ambiguous inputs into explicit standalone queries and is central to conversational search. TREC CAsT operationalizes this setting and provides rewritten queries for evaluation [6]. Learning-based approaches optimize rewriting via supervision or retrieval-oriented objectives, and LLM-based prompting can produce competitive rewrites [27, 31]. Widely used rewriting datasets (e.g., CANARD and QReCC) further support modeling context-to-query reformulation and retrieval-oriented rewriting behavior [2, 8]. Many setups allow the rewriter to access the full original context, which can introduce intent leakage. Our two-phase closed-book design prevents this: the rewriter never observes the intent query, isolating the contribution of clarification signals.

**Tool-Augmented Agents and Deep Search Evaluation.** Tool-augmented agents that interleave reasoning with actions (e.g., web search [11, 17, 26] and web agents [9, 34]) motivate new evaluation paradigms. ReAct formalizes reasoning–acting interleaving [30], while benchmarks such as BrowseComp and API-Bank evaluate browsing or tool-use capabilities [13, 25]. Complementary agent benchmarks evaluate realistic web interaction and general LLM-as-agent competence in interactive environments [16, 29, 35]. Other tool-use benchmarks systematize large tool collections and evaluate whether LLMs can select and invoke tools reliably [20]. Our focus is narrower but practically important: *clarify-then-search* under strict information constraints. We evaluate how well a system elicits and uses clarifying information to improve a query for a fixed deep-search backend (WebDancer), enabling controlled comparisons across Clarifiers.

**Nugget-Based and Evidence-Grounded Evaluation.** Nugget-based protocols score coverage of atomic information units and have been used in QA-style evaluations [14], with pyramid weighting improving robustness [7]. This family of methods is closely related to TREC-style evaluation practices that emphasize coverage of key facts rather than surface-form matching [23]. Recent RAG evaluation work revisits nugget extraction and matching as a fine-grained measure of answer completeness [19]. Our benchmark follows this direction but adds two properties: (i) *static golden nuggets* archived per intent query from a one-time WebDancer run for reproducibility, and (ii) a restoration metric that measures end-to-end clarification utility by quantifying how much of the evidence-grounded golden nugget set is recovered.

**Summary.** Existing work provides foundations for clarification, rewriting, tool-augmented search, and nugget-based evaluation. We integrate these into a leakage-resistant, end-to-end benchmark for deep search, evaluated on a curated set of 518 underspecified queries, where clarification is scored by downstream nugget restoration against static, evidence-grounded gold under a closed-book interaction protocol.

# 3 Task Definition and Closed-Book Protocol

## 3.1 Problem Setup

Our benchmark consists of 518 instances, each defined by an intent query $q^*$ (stored as `fused_query`) and a corresponding underspecified query $q$ (stored as `blurred_query`). Given only $q$, a *Clarifier* produces $k$ clarification questions $C = (c_1, \ldots, c_k)$, where $k \in \{1, 2, 3\}$. A constrained user then provides answers $A = (a_1, \ldots, a_k)$, and a *Rewriter* generates a rewritten query $\hat{q}$ that should be more specific and retrieval-ready. Finally, a fixed deep-search backend, WebDancer, runs on $\hat{q}$ to produce an answer with supporting evidence traces, and performance is evaluated against a static golden reference constructed for $q^*$. Figure 1 summarizes the end-to-end evaluation pipeline.

*Baselines and targets.* We use two fixed reference points. **Gold** corresponds to executing WebDancer once on the intent query $q^*$, archiving the resulting tool traces, and extracting evidence-grounded nuggets as a static reference. **Orig** denotes executing WebDancer directly on the underspecified query $q$ without clarification and rewriting. The benchmark therefore measures whether clarification and closed-book rewriting improve nugget restoration beyond **Orig**, using **Gold** as the fixed evaluation target.

## 3.2 Two-Phase Closed-Book Interaction

To prevent intent leakage in rewriting, we use two closed-book agents: a *User Answerer* and a *Rewriter*. The key idea is to separate the component that can observe the hidden intent from the component that constructs the final query. UA may access $q^*$ only to answer the current clarification question, while the Rewriter never observes $q^*$ and can only use information explicitly elicited through Q&A.

*User Answerer (UA).* UA receives the intent query $q^*$ and a single clarification question $c_i$. It must answer using only information *explicitly* present in $q^*$ and must not introduce new entities, facts, or constraints. If $q^*$ does not specify the requested attribute, UA outputs `unknown`.[2]

*Rewriter.* The Rewriter receives only the blurred query $q$ and the clarification Q&A pairs $\{(c_i, a_i)\}_{i=1}^{k}$, and outputs a single rewritten query $\hat{q}$. Crucially, the Rewriter must not access $q^*$ and must not invent new constraints. When an answer is `unknown`, the Rewriter preserves the original ambiguity for that aspect rather than forcing a concrete value.[3]

*Leakage example.* Consider BQ: "best universities for AI" and OQ: "best universities for AI in Europe for master's study". If the Clarifier asks "Which region are you asking about?", UA may answer "Europe" because this information is explicitly present in OQ. The Rewriter may then produce "best universities in Europe for AI". However, it cannot add "for master's study" unless that constraint is explicitly elicited by a clarification question. Thus, any added constraint in $\hat{q}$ must be grounded in the observed Q&A pairs rather than copied from the hidden intent.

This two-phase protocol ensures that downstream improvements are attributable to (i) the clarification questions asked and (ii) how effectively the Rewriter incorporates the elicited answers, rather than recovering intent directly from $q^*$.

## 3.3 Static Golden Reference

For each intent query $q^*$, we construct a static golden reference by running WebDancer once on $q^*$, archiving its tool traces, and

[2] In our implementation, UA is instantiated as an LLM following a strict no-fabrication policy; the prompt is shown in Appendix A and released with our artifact.
[3] The Rewriter prompt is shown in Appendix A and included in the released artifact.

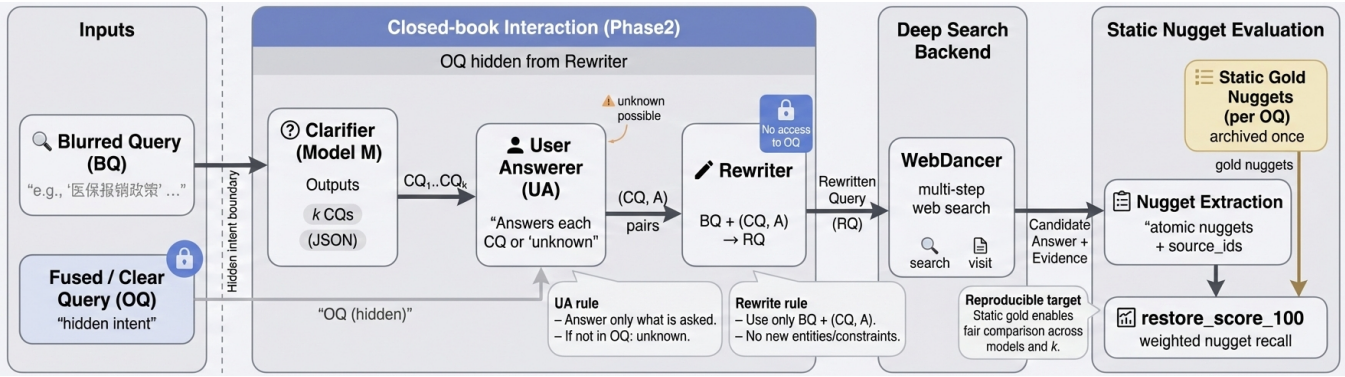



**Figure 1: Phase2 closed-book clarify-then-search pipeline. Given a blurred query (BQ), a Clarifier generates $k \in \{1, 2, 3\}$ clarification questions. A constrained User Answerer (UA) replies using only the hidden intent query (OQ; `fused_query`) and outputs `unknown` when the intent does not specify the requested attribute. A Rewriter then produces a rewritten query (RQ; $\hat{q}$) from BQ and the Q&A pairs without access to OQ. WebDancer runs deep search on RQ, and performance is scored by `restore_score_100` against static golden nuggets archived once per OQ.**

extracting a set of evidence-grounded *golden nuggets*. Each nugget is an atomic, judgeable statement supported by one or more evidence snippets from the archived traces, identified by source IDs. This yields a stable target that is reused across all Clarifiers and all clarification budgets $k$, enabling reproducible comparison of different clarification strategies.

## 4 Dataset

### 4.1 Overview and Released Fields

Our benchmark contains 518 information-seeking queries, deliberately selected to represent underspecified queries where clarification is expected to provide high utility. Each instance consists of an intent query $q^*$ (`fused_query`) and an underspecified query $q$ (`blurred_query`). Under the closed-book protocol (Section 3), a system observes only $q$, generates $k$ clarification questions, receives constrained answers, rewrites to $\hat{q}$, and is evaluated by running a fixed deep-search backend on $\hat{q}$ and scoring nugget restoration against a static reference built from $q^*$.

*Why paired blurring?* Our goal is to construct controlled pairs $(q, q^*)$ where missing constraints are explicit, so that clarification can be evaluated under a leakage-free protocol. Rather than synthesizing arbitrary ambiguity, we start from real user intents and remove key constraints such as time, region, entity scope, and criteria. This paired construction is necessary because $q^*$ serves as the hidden intent for UA and as the source query for static gold construction, while $q$ is the only query visible to the Clarifier and Rewriter.

*Why we do not release "gold" clarification labels.* Unlike datasets that provide canonical clarification questions as supervised targets, we treat clarification questions as *system outputs*. Our evaluation asks: *does the model ask for information that is both answerable under the intent and useful for downstream deep search?* This avoids prescribing a single "correct" clarification and instead measures clarification by end-to-end utility.

*Released artifacts.* We release on the project page: (i) all `fused_query` and `blurred_query` pairs ($n = 518$); (ii) a static golden reference per $q^*$, including weighted nuggets and traceable evidence identifiers; and (iii) code to reproduce one-time golden construction, Phase2 evaluation runs, and scoring via `restore_score_100`. Model-generated clarification questions, UA answers, rewrites, and WebDancer outputs are reproducible using the released runner and prompt templates. All data and code are available on the project page.[4] Figure 2 summarizes dataset construction and the one-time creation of static golden nuggets.

### 4.2 Static Golden Nuggets

For each intent query $q^*$, we construct a static golden reference in two stages.

*Stage 1: one-time WebDancer run on $q^*$.* We execute WebDancer once on $q^*$ and record (i) the final answer and (ii) tool traces from `search` and `visit`. These traces define the evidence pool used to construct the golden reference and are archived for reproducibility.

*Stage 2: evidence-grounded nugget extraction.* We then apply an evidence-grounded nugget extractor that takes as input $q^*$, WebDancer's answer, and the retrieved evidence snippets, and produces a structured `GOLD` JSON. The extractor enforces traceability: each nugget is an atomic, judgeable statement and stores identifiers of supporting evidence snippets. It also records a decomposition of the intent into required sub-questions (`must_cover`), missing evidence requirements (`missing_evidence`), and unsupported claims that appeared in the answer. Each nugget is assigned an integer weight $w_j \in \{1, 2, 3\}$ reflecting its importance for answering $q^*$. The extraction prompt is summarized in Appendix B, and the full prompt and JSON schema are released with our artifact.

This procedure yields a single archived nugget set per query, which is reused across all evaluated systems and all $k$ settings. Because the golden reference is built once and held fixed, system

[4] https://clarify-then-search.github.io/

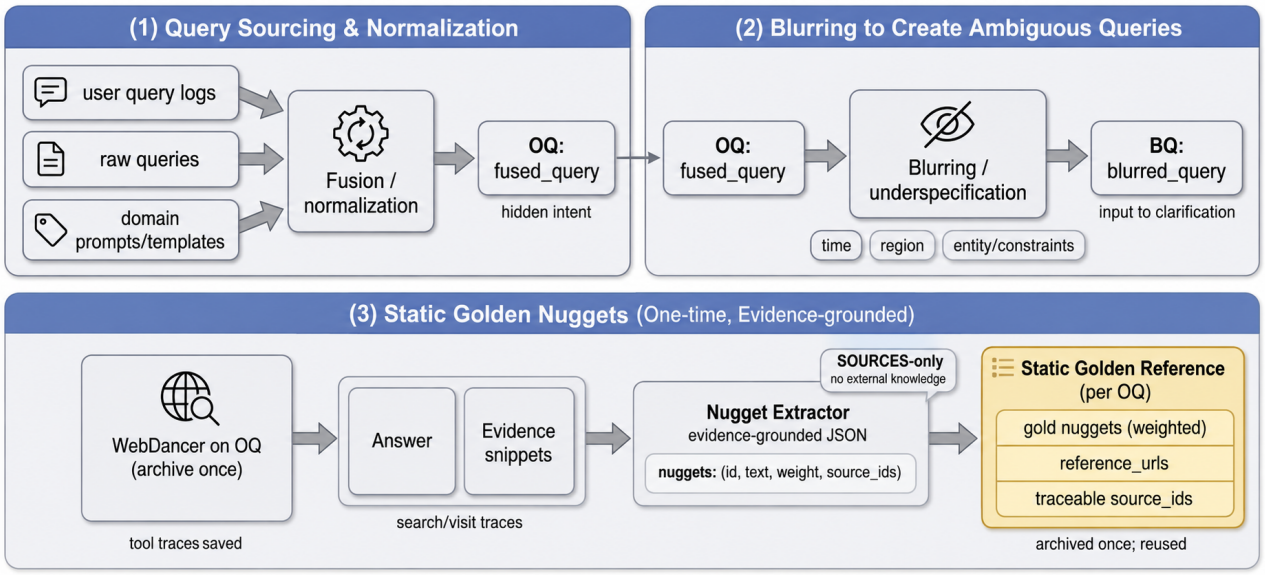



**Figure 2: Dataset construction and static golden nugget creation. We fuse and normalize raw user queries into intent queries (OQ; `fused_query`), then create underspecified benchmark inputs by blurring key constraints to form blurred queries (BQ; `blurred_query`). For each OQ, we run WebDancer once to archive an answer and search/visit evidence traces, and extract evidence-grounded, weighted golden nuggets with traceable source IDs. This static gold is reused to score all systems with `restore_score_100`, enabling reproducible comparisons across models and clarification budgets $k$.**

comparisons are not affected by run-to-run variation in the deep-search backend during gold construction.

## 4.3 Evaluation Inputs and Output Records

At evaluation time, a system produces clarification questions $C$, UA answers $A$, a rewritten query $\hat{q}$, and a WebDancer output $\hat{y}$ with tool traces. We compute `restore_score_100` by comparing $\hat{y}$ against the static golden nuggets for $q^*$ (Section 4.4). We report mean `restore_score_100` as the primary metric, and additionally report distributional summaries including p50, p90, min, and max.

*Implementation note.* In our runner, the candidate answer $\hat{y}$ is extracted from WebDancer's final assistant output by parsing the `<answer>...</answer>` tag when present, with robust fallbacks for partially closed tags. All per-instance tool traces are saved for auditing and reproducibility; compact summaries are included in the released evaluation outputs.

## 4.4 Evaluation Protocol

*4.4.1 Nugget Coverage Judging.* Given a candidate answer $\hat{y}$ and a set of golden nuggets $\mathcal{N} = \{(n_j, w_j)\}_{j=1}^{m}$, we determine nugget-level coverage using a strict LLM-based judge. For each nugget $n_j$, the judge outputs one of three labels: `full`, `partial`, or `none`. `full` indicates that $\hat{y}$ expresses the nugget content with required constraints; `partial` indicates that the core statement is present but a critical qualifier is missing; `none` indicates that the nugget is absent or contradicted.

We use ERNIE-4.5-Turbo as the coverage judge with deterministic decoding (temperature 0). To improve reliability, we apply robust JSON parsing with retries and enforce a strict rubric. The judge prompt is summarized in Appendix B and released with our artifact.

*4.4.2* `restore_score_100`. We define weighted nugget recall with partial credit as

$$\text{Recall}(\hat{y}) = \frac{\sum_{j=1}^{m} w_j \cdot s(\text{cov}(n_j, \hat{y}))}{\sum_{j=1}^{m} w_j}, \tag{1}$$

where $\text{cov}(\cdot) \in \{\texttt{full}, \texttt{partial}, \texttt{none}\}$ is the judge label and

$$s(\texttt{full}) = 1, \qquad s(\texttt{partial}) = 0.5, \qquad s(\texttt{none}) = 0. \tag{2}$$

We report

$$\texttt{restore\_score\_100} = 100 \times \text{Recall}(\hat{y}). \tag{3}$$

The primary benchmark score is the mean `restore_score_100` over evaluated instances, and we additionally report percentile summaries to characterize distributional behavior.

# 5 Experimental Setup

## 5.1 Phase2 Closed-Book Pipeline and Roles

We evaluate clarification utility under the controlled closed-book protocol described in Section 3. The pipeline instantiates three LLM roles: a *Clarifier* that generates clarification questions, a *User Answerer (UA)* that answers under a strict no-fabrication policy given the hidden intent, and a *Rewriter* that produces a rewritten query $\hat{q}$ using only the blurred query and the clarification Q&A pairs. To isolate clarification ability, we vary only the Clarifier while keeping UA, Rewriter, the deep-search backend, nugget construction, and nugget judging fixed.

**Table 1: Model backends used in our benchmark. We vary only the clarification-question generator (Clarifier); UA, Rewriter, nugget extraction, and coverage judging are fixed to ERNIE-4.5-Turbo-128K for controlled comparisons.**

| Component | Model |
|---|---|
| Clarifier (CQ generator) | Qwen3-235B-A22B-Instruct [28]<br>ERNIE-4.5-Turbo-128K [24]<br>Kimi-K2-Instruct [22]<br>DeepSeek-V3.2 [15]<br>GPT-5.2 [18]<br>Claude-Sonnet-4.5 [3]<br>Gemini-2.5-Pro [5] |
| User Answerer (UA) | ERNIE-4.5-Turbo-128K [24] |
| Rewriter | ERNIE-4.5-Turbo-128K [24] |
| Nugget extractor | ERNIE-4.5-Turbo-128K [24] |
| Coverage judge | ERNIE-4.5-Turbo-128K [24] |

Table 1 lists the model backends used for each pipeline component. This controlled design attributes performance differences primarily to (i) which ambiguity dimensions are asked about and (ii) whether those questions elicit answerable and useful information under the closed-book UA policy. The core prompts are shown in Appendices A–B, and the full executable prompts are released with the benchmark artifact.

*Clarifier (varied).* Given a blurred query $q$ (`blurred_query`), the Clarifier outputs $k$ clarification questions $C = (c_1, \ldots, c_k)$, where $k \in \{1, 2, 3\}$. Clarifiers are prompted to derive questions from $q$ only, with no access to the intent query $q^*$, and to ask questions that target a single ambiguity dimension, such as time, region, entity scope, or definition/criterion. We enforce strict output formatting as a JSON array of strings and run all Clarifiers with deterministic decoding.

*User Answerer (UA; fixed).* UA is fixed to ERNIE-4.5-Turbo (Table 1) with temperature 0. For each question $c_i$, UA receives the intent query $q^*$ (`fused_query`) and must output a minimal answer $a_i$ only if the intent explicitly specifies the requested attribute; otherwise it outputs `unknown`. UA is explicitly forbidden from adding any information not present in $q^*$.

*Rewriter (fixed).* The Rewriter is also fixed to ERNIE-4.5-Turbo (Table 1) with temperature 0. It receives only the blurred query $q$ and the clarification Q&A pairs $\{(c_i, a_i)\}_{i=1}^{k}$, and outputs a single rewritten query $\hat{q}$. The Rewriter must not introduce new entities or constraints; when $a_i$ = `unknown`, it must preserve the ambiguity rather than forcing a concrete value.

*Static gold and judging components (fixed).* To ensure controlled comparisons across Clarifiers, we also fix the nugget extractor and coverage judge to ERNIE-4.5-Turbo (Table 1) with deterministic decoding. Golden nuggets are extracted from evidence retrieved by running WebDancer once on $q^*$ (Section 4.2). At evaluation time, nugget coverage is judged with a strict rubric that outputs `full`/`partial`/`none` per nugget (Section 4.4), with robust JSON parsing and retries.

## 5.2 Deep Search Backend: WebDancer

We use WebDancer [26] as the fixed deep-search backend for all evaluations. Given a rewritten query $\hat{q}$, WebDancer interleaves reasoning with tool calls to `search` and `visit`, and produces a final answer $\hat{y}$ along with tool traces. We run WebDancer in batch mode with a fixed tool budget and fixed configuration, and persist per-instance tool traces for auditing and reproducibility. The candidate answer $\hat{y}$ is extracted from WebDancer's final output by parsing the `<answer>. . . </answer>` tag when present, with robust fallbacks for partially closed tags.

Fixing WebDancer is a deliberate choice for controlled evaluation. Our goal is not to claim backend-independent absolute scores, but to compare Clarifiers under the same deep-search environment. This avoids conflating differences in clarification quality with variation from different retrieval or agentic search backends.

## 5.3 Evaluation Set and Reporting

*Evaluation set.* All experiments and analyses in this paper are conducted on our released dataset of 518 instances (Section 4). These instances are selected to represent underspecified queries where clarification is expected to matter most, so we do not further partition them into difficulty subsets in the main paper.

*Budgeted clarification protocol.* For each query and each budget $k \in \{1, 2, 3\}$, we execute: (1) the Clarifier generates $k$ questions $C$; (2) UA generates $k$ answers $A$ under the closed-book constraint; (3) the Rewriter outputs $\hat{q}$ using only $q$ and $\{(c_i, a_i)\}_{i=1}^{k}$; and (4) WebDancer runs on $\hat{q}$ to produce a candidate answer $\hat{y}$. We then compute `restore_score_100` of $\hat{y}$ against the static golden nuggets archived from a one-time WebDancer run on $q^*$ (Section 4.2).

*Metrics and reported statistics.* We report mean `restore_score_100` as the primary score, and additionally report p50/p90 and min/max to characterize distributional behavior. All prompts, scripts, and configurations required to reproduce the full pipeline are released with the benchmark.

# 6 Results

## 6.1 Main Results: One-Turn Clarification

We first evaluate clarification utility under the closed-book Phase2 protocol with a single clarification question ($k = 1$) on the 518-instance benchmark. Table 2 reports `restore_score_100` for each Clarifier, where `orig` denotes running WebDancer directly on the underspecified query without clarification.

All Clarifiers improve over `orig`, showing that even one clarification can elicit useful intent-grounded information for rewriting and downstream deep search. However, the magnitude of improvement varies across models. Among the evaluated open-weight models, including Qwen, ERNIE, DeepSeek, and Kimi, mean gains are modest (+3.09 to +4.03), and the median remains close to the baseline, indicating that one-turn clarification often fails to recover decisive constraints. In contrast, GPT, Claude and Gemini yield larger gains (+6.44 to +7.04), with GPT achieving the highest mean and median. Beyond the mean, clarification substantially lifts the recoverable tail: p90 increases from 35.3 for `orig` to roughly 46–53 across Clarifiers, suggesting that some instances become much more recoverable once a key ambiguity is resolved.

**Table 2: Benchmark results ($n = 518$) under closed-book Phase2 with $k = 1$. Metric: `restore_score_100`. $\Delta$ is mean difference vs. `orig`.**

| System | mean ($\Delta$) | p50 | p90 |
|---|---|---|---|
| orig | 19.434 | 20.454 | 35.294 |
| Qwen | 22.527 (+3.093) | 19.565 | 47.211 |
| ERNIE | 23.460 (+4.026) | 20.000 | 50.000 |
| DeepSeek | 23.234 (+3.799) | 21.053 | 48.416 |
| Kimi | 22.687 (+3.252) | 20.000 | 46.236 |
| GPT | **26.474 (+7.040)** | **25.000** | 50.000 |
| Claude | 25.911 (+6.477) | 23.333 | **52.996** |
| Gemini | 25.874 (+6.440) | 21.429 | **52.996** |

**Table 3: Mean improvement over `orig` at $k = 1$ with paired bootstrap 95% CIs.**

| System | mean $\Delta$ vs. `orig` | 95% CI |
|---|---|---|
| Qwen | +3.093 | [1.482, 4.729] |
| ERNIE | +4.026 | [2.526, 5.637] |
| DeepSeek | +3.799 | [2.338, 5.357] |
| Kimi | +3.252 | [1.753, 4.762] |
| GPT | **+7.040** | [5.566, 8.578] |
| Claude | +6.477 | [4.890, 8.159] |
| Gemini | +6.440 | [4.902, 8.057] |

*Statistical robustness.* We compute paired bootstrap 95% confidence intervals (CIs) on per-instance score differences against `orig`. As shown in Table 3, all CIs are strictly above zero, confirming that the improvements are not driven by a small number of outliers.

## 6.2 UA Diagnostics: Unknown Rate and Question-Type Bias

Phase2 exposes an intermediate information bottleneck: a clarification question is useful only if it elicits an intent-grounded answer that can be safely incorporated into the rewritten query. We therefore analyze UA responses and question types on the 518 instances.

For each clarification question, UA returns `unknown` when the hidden intent does not explicitly specify the requested attribute. At $k = 1$, `ua_unknown_rate` is also the probability that the interaction yields no grounded signal. Table 4 shows that `ua_unknown_rate` remains high across Clarifiers (0.600–0.701). Thus, in roughly two thirds of instances, one clarification question fails to retrieve any intent-grounded constraint, limiting what the Rewriter can safely add and leaving the rewritten query close to the original underspecified input.

A consistent low-yield failure mode is *region-only* clarification, such as asking for a city or province. Many intents do not specify a location, so such questions frequently elicit `unknown`. Table 5 shows that region-only questions have very high unknown rates (0.750–0.869), while models differ substantially in how often they ask them: `cq_region_rate` ranges from 0.143 for Kimi to 0.498 for Gemini. This illustrates why plausible clarification is not necessarily useful clarification: a question can be natural and topical, yet yield no usable information under the closed-book constraint.

**Table 4: UA diagnostics at $k = 1$ ($n = 518$). `ua_unknown_rate`: fraction of answers labeled `unknown`. `cq_region_rate`: fraction of clarification questions about region/location.**

| Model | ua_unknown_rate | cq_region_rate |
|---|---|---|
| Qwen | 0.620 | 0.195 |
| ERNIE | 0.680 | 0.357 |
| DeepSeek | 0.622 | 0.479 |
| Kimi | 0.666 | 0.143 |
| GPT | 0.618 | 0.351 |
| Claude | 0.600 | 0.326 |
| Gemini | 0.701 | 0.498 |

**Table 5: Region-only clarification as a low-yield failure mode at $k = 1$ ($n = 518$). We report the number of region-only questions and the UA unknown rate conditioned on region-only.**

| Model | #region-only | region-only unknown |
|---|---|---|
| Qwen | 64 | 0.766 |
| ERNIE | 136 | 0.787 |
| DeepSeek | 223 | 0.807 |
| Kimi | 48 | 0.750 |
| GPT | 121 | 0.785 |
| Claude | 148 | 0.824 |
| Gemini | 236 | 0.869 |

## 6.3 Budgeted Clarification: $k = 2$ and $k = 3$

A single-turn interaction may miss the most useful ambiguity dimension. We therefore evaluate larger clarification budgets $k \in \{2, 3\}$ under the same closed-book protocol. Table 6 summarizes the results.

Increasing the budget from $k = 1$ to $k = 2$ yields a clear jump in mean restoration for all models, with gains ranging from +6.16 to +7.64 over `orig`. At $k = 3$, ERNIE achieves the largest improvement (+8.90), while some models show saturation or slight regression relative to $k = 2$. The median also increases with budget, indicating that additional turns make improvements more typical rather than only improving a small tail.

The relative ordering changes with larger budgets. At $k = 1$, GPT achieves the highest mean score, while ERNIE achieves the highest mean among all open-weight models. At $k = 3$, ERNIE becomes the top-performing model. This suggests that larger budgets reward sustained question-selection quality across turns, rather than a single high-impact question.

*Multi-turn unknown rate.* We further analyze how often an entire interaction yields no grounded signal. Table 7 reports `all_unknown_rate` and the number of non-`unknown` answers per instance. Increasing $k$ substantially reduces `all_unknown_rate`: for example, GPT drops from 0.618 at $k = 1$ to 0.375 at $k = 2$ and 0.189 at $k = 3$. This aligns with the restoration gains, suggesting that larger budgets help mainly by reducing no-signal interactions and increasing the chance of eliciting at least one useful constraint.

**Table 6: Benchmark results under Phase2 for $k \in \{2, 3\}$ ($n = 518$). Metric: `restore_score_100`. $\Delta$ is mean difference vs. `orig`.**

| $k$ | System | mean ($\Delta$) | p50 | p90 |
|---|---|---|---|---|
| 2 | orig | 19.434 | 20.454 | 35.294 |
| | Qwen | 26.020 (+6.586) | 23.607 | 52.941 |
| | ERNIE | 26.351 (+6.917) | 23.842 | 52.632 |
| | DeepSeek | 26.628 (+7.194) | 23.747 | 53.259 |
| | Kimi | 25.593 (+6.159) | 22.902 | 50.000 |
| | GPT | **27.073 (+7.639)** | **25.000** | 53.487 |
| | Claude | 26.203 (+6.769) | 23.509 | 53.654 |
| | Gemini | 26.904 (+7.470) | **25.000** | **54.545** |
| 3 | orig | 19.434 | 20.454 | 35.294 |
| | Qwen | 26.235 (+6.801) | 22.997 | 51.744 |
| | ERNIE | **28.339 (+8.904)** | **26.087** | **57.143** |
| | DeepSeek | 26.486 (+7.052) | 25.000 | 52.424 |
| | Kimi | 26.568 (+7.134) | 23.509 | 56.310 |
| | GPT | 26.297 (+6.863) | 23.385 | 54.407 |
| | Claude | 27.029 (+7.594) | 25.000 | 53.329 |
| | Gemini | 27.126 (+7.691) | 24.194 | 52.628 |

**Table 7: Interaction-level UA unknown diagnostics. `all_unknown`: all $k$ answers are `unknown`. `known_cnt`: number of non-unknown answers.**

| $k$ / Model | all_unknown | known_cnt=1 | known_cnt=2 | known_cnt=3 |
|---|---|---|---|---|
| | | $k = 1$ | | |
| Qwen | 0.620 | 0.380 | – | – |
| ERNIE | 0.680 | 0.320 | – | – |
| DeepSeek | 0.622 | 0.378 | – | – |
| Kimi | 0.666 | 0.334 | – | – |
| GPT | 0.618 | 0.382 | – | – |
| Claude | 0.600 | 0.400 | – | – |
| Gemini | 0.701 | 0.299 | – | – |
| | | $k = 2$ | | |
| Qwen | 0.446 | 0.398 | 0.156 | – |
| ERNIE | 0.498 | 0.390 | 0.112 | – |
| DeepSeek | 0.394 | 0.452 | 0.154 | – |
| Kimi | 0.510 | 0.407 | 0.083 | – |
| GPT | **0.375** | 0.458 | 0.168 | – |
| Claude | 0.452 | 0.409 | 0.139 | – |
| Gemini | 0.523 | 0.380 | 0.097 | – |
| | | $k = 3$ | | |
| Qwen | 0.417 | 0.417 | 0.139 | 0.027 |
| ERNIE | 0.394 | 0.403 | 0.164 | 0.039 |
| DeepSeek | 0.251 | 0.396 | 0.270 | 0.083 |
| Kimi | 0.421 | 0.427 | 0.137 | 0.015 |
| GPT | **0.189** | 0.432 | 0.315 | 0.064 |
| Claude | 0.295 | 0.438 | 0.226 | 0.041 |
| Gemini | 0.413 | 0.398 | 0.164 | 0.025 |

# 7 Analysis

## 7.1 Information Gain is the Main Bottleneck

A key advantage of our benchmark is that it makes intermediate interaction behavior measurable, rather than judging only the final deep-search answer. Under the closed-book protocol, we can inspect both what clarification questions a model asks and whether the hidden intent supports a grounded answer. This allows us to distinguish two factors that are often conflated in interactive search: whether a question sounds reasonable, and whether it actually yields usable information for rewriting.

The results show that information gain is often the bottleneck. At $k = 1$, UA `unknown` rates remain high across models (0.600–0.701; Table 4), meaning that many single-turn clarifications elicit no intent-grounded constraint. When the only answer is `unknown`, the Rewriter has little grounded content to incorporate and must proceed with residual ambiguity. This helps explain why one-turn clarification reliably improves over `orig` but still leaves substantial room for improvement.

Multi-turn statistics further support this interpretation. As shown in Table 7, increasing the clarification budget reduces the probability that an entire interaction yields no grounded signal. For example, GPT's `all_unknown` rate drops from 0.618 at $k = 1$ to 0.375 at $k = 2$ and 0.189 at $k = 3$. This aligns with the restoration gains in Table 6: additional turns help mainly by increasing the chance of eliciting at least one usable constraint. However, the marginal gain depends on whether later questions target orthogonal and answerable ambiguity dimensions, rather than repeatedly asking about attributes absent from the intent.

## 7.2 Question-Type Bias is a Measurable Failure Mode

Our diagnostics reveal a concrete failure mode: plausible-sounding clarification is not necessarily useful clarification. A prominent example is region-only questioning. At $k = 1$, models differ substantially in their tendency to ask region/location questions: `cq_region_rate` ranges from 0.143 for Kimi to 0.498 for Gemini (Table 4). However, region-only questions are consistently low-yield under our closed-book protocol. As shown in Table 5, their UA unknown rates are very high across all models (0.750–0.869).

This low-yield pattern persists beyond one-turn clarification. Table 8 summarizes the full turn-level diagnostics, with the complete turn-level region-only statistics provided in Appendix C: across larger budgets, region-only unknown rates remain high across models and turns. This indicates that repeatedly asking for location does not reliably increase information gain when the hidden intent does not contain location information.

This pattern matters because a region question can appear natural and relevant even when the hidden intent does not specify any location. In such cases, the answer becomes `unknown`, and the Rewriter is not allowed to add a concrete region. Thus, region bias can waste clarification budget without improving the rewritten query. By exposing this behavior, the benchmark supports diagnostics beyond surface fluency or topical relevance and identifies which question types are likely to produce low information gain.

**Table 8: Region-only clarification remains low-yield across clarification budgets. We report the range of UA unknown rates for region-only questions across models and turns.**

| Setting | Region-only unknown range | Observation |
|---|---|---|
| $k = 1$ | 0.750–0.869 | consistently high |
| $k = 2$ | 0.800–0.917 | remains high across turns |
| $k = 3$ | 0.692–1.000 | often still unanswerable |

**Table 9: Judge-backbone robustness at $k = 1$ on the benchmark set. Absolute scores vary across judges, but the improvement over orig and the relative ordering remain stable.**

| Judge | orig | ERNIE-$k1$ | GPT-$k1$ | Ordering |
|---|---|---|---|---|
| ERNIE | 19.43 | 23.46 | 26.47 | GPT > ERNIE > orig |
| GPT | 27.74 | 29.54 | 29.79 | GPT > ERNIE > orig |
| Claude | 32.19 | 33.80 | 34.62 | GPT > ERNIE > orig |

## 7.3 Budgeted Clarification as a Controlled Intervention

Varying the clarification budget $k$ provides a controlled way to test whether performance is limited by one-shot question selection or by deeper limitations of the pipeline. The restoration results show a clear budget effect: moving from $k = 1$ to $k = 2$ increases mean improvements for all models, and $k = 3$ further improves several systems (Table 6). At the same time, the gains are not purely monotonic for every model, suggesting that extra turns are useful only when they elicit additional grounded constraints.

The relative ordering also changes with budget. GPT performs best at $k = 1$ and $k = 2$, while ERNIE becomes strongest at $k = 3$. This suggests that different budgets reward different clarification abilities: $k = 1$ emphasizes single-shot targeting of a high-value ambiguity, whereas larger budgets reward sustained question-selection quality across turns. Therefore, evaluating only one clarification budget can be misleading for understanding model behavior.

These observations also motivate an ask-or-not decision. A practical clarify-then-search system should ask only when the expected information gain is positive. Signals such as all_unknown rate and question-type-specific unknown rate provide measurable proxies for learning or evaluating such gating policies. For example, high predicted answerability may justify asking, while low-yield question types such as region-only clarification may suggest abstaining or switching to retrieval-first exploration.

## 7.4 Robustness to Supporting and Judging Backbones

Our main experiments fix UA, Rewriter, nugget extraction, and coverage judging to ensure controlled comparisons. To examine whether the main conclusions depend on a single ERNIE-based stack, we further conduct a backbone-substitution analysis for the supporting components and coverage judge.

For the end-to-end judge, replacing the ERNIE-based coverage judge changes the absolute score scale, but preserves the main ordering. As shown in Table 9, under ERNIE-, GPT-, and Claude-based judges, both ERNIE-$k1$ and GPT-$k1$ outperform orig, and the ordering GPT > ERNIE > orig remains stable. This suggests that the main end-to-end conclusion is not an artifact of one particular coverage judge.

For the intermediate stages, UA shows moderate backbone sensitivity, mainly in the overall unknown rate and answer style. However, pairwise known/unknown agreement remains relatively high across supporting backbones, indicating that different UA backbones often agree at the coarse answerability level. In contrast, Rewriter protocol behavior is highly stable: unknown-preservation rates remain above 0.98, and unsupported concretization, such as adding a new year or region when the answer is unknown, is near zero. Overall, these results support our design choice of fixing supporting components for controlled comparison while showing that the primary findings are robust to reasonable backbone substitutions. Detailed supporting-backbone stability statistics are provided in Appendix C.

## 8 Conclusion

We introduced Clarify-Then-Search, a benchmark for evaluating whether clarification improves deep search under realistic information constraints. The benchmark uses a two-phase closed-book protocol and static evidence-grounded golden nuggets to measure end-to-end nugget restoration after clarification, rewriting, and deep search. Our experiments show that clarification consistently improves over the no-interaction baseline, and that larger clarification budgets generally yield stronger restoration, with ERNIE-4.5-Turbo-128K becoming the overall top-performing model at $k = 3$. At the same time, diagnostic analyses reveal that many clarification turns still elicit unknown or low-yield answers, especially for region-only questions, highlighting the need for future systems to better decide when to ask, what to ask, and how to use clarification signals for robust deep search.

## Limitations

First, our closed-book protocol uses an LLM-based UA to produce constrained answers from the hidden intent query. Although this design prevents intent leakage to the Rewriter, the exact unknown policy can affect measured utility. Second, our end-to-end evaluation fixes WebDancer as the deep-search backend. The benchmark is therefore best interpreted as a controlled evaluation of clarification utility under a shared deep-search pipeline, rather than a backend-agnostic ranking of all possible search systems. Third, golden nugget construction and coverage judging rely on LLMs with strict rubrics and deterministic decoding; borderline semantic matches may still introduce judging noise.

## Acknowledgments

This research was supported in part by the National Natural Science Foundation of China under Grant No. 92370204, the National Science and Technology Major Project of China under Grant No. 2023ZD0121104, and the Anhui Natural Science Foundation under Grant No. 2508085ZD006.

## References


[1] Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval*. 475–484.

[2] Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2021. Open-domain question answering goes conversational via question rewriting. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 520–534.

[3] Anthropic. 2025. Introducing Claude Sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5. Accessed: 2026-02-07.

[4] Keping Bi, Qingyao Ai, and W Bruce Croft. 2021. Asking clarifying questions based on negative feedback in conversational search. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. 157–166.

[5] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261* (2025).

[6] Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. 2020. TREC CAsT 2019: The conversational assistance track overview. *arXiv preprint arXiv:2003.13624* (2020).

[7] Hoa Trang Dang and Jimmy Lin. 2007. Different structures for evaluating answers to complex questions: Pyramids won't topple, and neither will human assessors. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. 768–775.

[8] Ahmed Elgohary, Denis Peskov, and Jordan Boyd-Graber. 2019. Can you unpack that? learning to rewrite questions-in-context. *Can You Unpack That? Learning to Rewrite Questions-in-Context* (2019).

[9] Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. Webvoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 6864–6890.

[10] Deqiang Huang, Xinjiang Lu, Jingbo Zhou, Nijia Lu, Fuxin Li, Bo Hong, Chuanming Zhang, Tong Xu, and Enhong Chen. 2026. Uncertainty-Aware Planning for Disambiguating User Intent in Interactive LLM Agents: Application to Baidu Maps. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

[11] Lisheng Huang, Yichen Liu, Jinhao Jiang, Rongxiang Zhang, Jiahao Yan, Junyi Li, and Wayne Xin Zhao. 2025. ManuSearch: Democratizing Deep Search in Large Language Models with a Transparent and Open Multi-Agent Framework. In *Findings of the Association for Computational Linguistics: EMNLP 2025*. 2403–2417.

[12] Vaibhav Kumar and Alan W Black. 2020. ClarQ: A large-scale and diverse dataset for clarification question generation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*. 7296–7301.

[13] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. Api-bank: A comprehensive benchmark for tool-augmented llms. *arXiv preprint arXiv:2304.08244*.

[14] Jimmy Lin and Dina Demner-Fushman. 2006. Methods for automatically evaluating answers to complex questions. *Information Retrieval* 9, 5 (2006), 565–587.

[15] Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. 2025. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556* (2025).

[16] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688* (2023).

[17] Pengfei Luo, Jingbo Zhou, Tong Xu, Yuan Xia, Linli Xu, and Enhong Chen. 2025. Imagescope: Unifying language-guided image retrieval via large multimodal model collective reasoning. In *Proceedings of the ACM on Web Conference 2025*. 1666–1682.

[18] OpenAI. 2025. Introducing GPT-5.2. https://openai.com/index/introducing-gpt-5-2/. Accessed: 2026-02-07.

[19] Ronak Pradeep, Nandan Thakur, Shivani Upadhyay, Daniel Campos, Nick Craswell, and Jimmy Lin. 2024. Initial nugget evaluation results for the trec 2024 rag track with the autonuggetizer framework. *arXiv preprint arXiv:2411.09607* (2024).

[20] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789* (2023).

[21] Leila Tavakoli, Johanne R Trippas, Hamed Zamani, Falk Scholer, and Mark Sanderson. 2022. Mimics-duo: Offline & online evaluation of search clarification. (2022), 3198–3208.

[22] Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534* (2025).

[23] Ellen M Voorhees and L Buckland. 2003. Overview of the TREC 2003 Question Answering Track.. In *TREC*, Vol. 2003. 54–68.

[24] Haifeng Wang, Hua Wu, Tian Wu, Yu Sun, Jing Liu, Dianhai Yu, Yanjun Ma, Jingzhou He, Zhongjun He, Dou Hong, et al. 2026. ERNIE 5.0 Technical Report. *arXiv preprint arXiv:2602.04705* (2026).

[25] Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516* (2025).

[26] Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhenglin Wang, Zhengwei Tao, Ding-Chu Zhang, Zekun Xi, Robert Tang, et al. 2026. Webdancer: Towards autonomous information seeking agency. *Advances in Neural Information Processing Systems* 38, 120957–120985.

[27] Zeqiu Wu, Yi Luan, Hannah Rashkin, David Reitter, Hannaneh Hajishirzi, Mari Ostendorf, and Gaurav Singh Tomar. 2022. Conqrr: Conversational query rewriting for retrieval with reinforcement learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 10000–10014.

[28] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388* (2025).

[29] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems* 35 (2022), 20744–20757.

[30] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

[31] Fanghua Ye, Meng Fang, Shenghui Li, and Emine Yilmaz. 2023. Enhancing conversational search: Large language model-aided informative query rewriting. *arXiv preprint arXiv:2310.09716*.

[32] Hamed Zamani, Susan Dumais, Nick Craswell, Paul Bennett, and Gord Lueck. 2020. Generating clarifying questions for information retrieval. In *Proceedings of the web conference 2020*. 418–428.

[33] Hamed Zamani, Gord Lueck, Everest Chen, Rodolfo Quispe, Flint Luu, and Nick Craswell. 2020. Mimics: A large-scale data collection for search clarification. In *Proceedings of the 29th ACM international conference on information & knowledge management*. 3189–3196.

[34] Le Zhang, Yixiong Xiao, Xinjiang Lu, Jingjia Cao, Yusai Zhao, Jingbo Zhou, Lang An, Zikan Feng, Wanxiang Sha, Yu Shi, et al. 2026. OmegaUse: Building a General-Purpose GUI Agent for Autonomous Task Execution. *arXiv preprint arXiv:2601.20380* (2026).

[35] Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. 2024. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, Vol. 2024. 15585–15606.

# A Prompt Templates

For reproducibility, we include the core prompt templates used in our closed-book clarify-then-search protocol. Prompts A–C correspond to the Clarifier, User Answerer, and Rewriter, which define the interaction protocol. Prompts D–E summarize the evidence-grounded nugget generation and coverage judging procedures. The full executable prompts, JSON schemas, and scripts are released on the project page.

**Prompt A: Clarifier**

# **Role.** You are a *search clarification question generator.*

# **Input.** You are given a search-box query: `BLURRED_QUERY`. The query may be underspecified in time, location, target scope, criteria/definition, or other constraints.

# **Task.** (1) Identify key missing or ambiguous constraints only from `BLURRED_QUERY`. Do not introduce new entities, facts, or constraints not implied by the query.

(2) Select the top-`K` questions with the highest expected information gain. Generate exactly `K` short clarification questions, ordered from highest to lowest priority.

(3) Each question must focus on one aspect only, such as time range, geographic region, entity disambiguation, scope, or evaluation criteria. Questions must not be redundant or heavily overlapping.

(4) Do not rewrite or answer the query. Output only the clarification questions.

# **Output.** Output a single JSON array with exactly `K` Chinese colloquial clarification question strings. Do not output explanations, prefixes, suffixes, or code fences.

**Prompt B: User Answerer**

# **Role.** You are acting as the *real user*. The user's true intent, i.e., the clear query, is provided to you privately.

# **Inputs.** The system provides: `clear_query` and one `clarification_question`.

# **Task.** (1) Answer the clarification question only using information explicitly stated in `clear_query`. Do not fabricate or add any new facts or constraints.

(2) Answer only what the clarification question asks. Do not provide additional information beyond the asked aspect.

(3) If the requested information is not explicitly specified in `clear_query`, output exactly: `unknown`.

(4) Keep the output extremely brief: a single short sentence or phrase. No explanations, reasoning, or bullet points.

# **Output.** Output only the user answer text. If the answer is not specified in `clear_query`, output exactly `unknown`.

**Prompt C: Rewriter**

# **Role.** You are acting as the *search user*. You only remember the original blurred query and the clarification Q&A pairs. You do not have access to the true intent query.

# **Inputs.** The system provides: `blurred_query` and `qa_pairs`, where each pair contains a `clarification_question` and a `user_answer`.

# **Task.** (1) Rewrite the blurred query into a clearer, more specific, retrieval-ready single query using only `blurred_query` and `qa_pairs`.

(2) Do not introduce new entities, facts, or constraints not present in the inputs.

(3) If a `user_answer` is `unknown`, do not force that aspect to be specific. Keep it underspecified as in the original query.

(4) Output only the final rewritten query as a single sentence. No explanations, steps, or multiple candidates.

# **Output.** Output exactly one rewritten query sentence in plain text.

**Figure 3:** Core closed-book interaction prompts. Prompt A generates clarification questions, Prompt B answers them under the hidden intent without adding extra information, and Prompt C rewrites the query without access to the hidden intent.

# B Evaluation Prompt Summary

**Prompt D: Evidence-Grounded Golden Reference Generator**

# **Role.** You are a *golden reference* generator for retrieval-augmented evaluation.

# **Inputs.** The input contains: `QUERY`, a model `ANSWER`, and retrieved `SOURCES`, where each source contains identifiers and evidence snippets.

# **Goal.** Using only `SOURCES`, produce a traceable and scorable `GOLD` JSON.

# **Mandatory Rules.** (1) Use only `SOURCES` as factual grounding. Do not introduce external knowledge or common sense.

(2) Decompose `QUERY` into sub-questions. Only evidence-supported sub-questions may be included in `must_cover`; unsupported requirements must be placed in `missing_evidence`.

(3) Nuggets must be atomic and judgeable factual statements. Each nugget must include one or more `source_ids`.

(4) Compare `ANSWER` against `SOURCES` and list unsupported factual claims in `unsupported_claims`.

(5) Output strict JSON only.

# **Weighting Rule.** `3`: core essential; `2`: important information; `1`: supplemental information.

# **Main GOLD fields.** `query_intent`, `nuggets`, `reference_urls`, `unsupported_claims`, and `notes`. Each nugget stores `id`, `text`, `weight`, and `source_ids`.

**Prompt E: Nugget Coverage Judge**

# **Role.** You are a rigorous *coverage judge* for nugget-based evaluation.

# **Inputs.** The input contains: `QUERY`, `GOLD_NUGGETS`, and `CANDIDATE_ANSWER`.

# **Task.** For each golden nugget, decide whether `CANDIDATE_ANSWER` covers the nugget's meaning. Judging must be strict and based only on the candidate answer.

# **Labels.** `full`: semantically consistent and includes all critical details.

`partial`: mentions the core idea but misses a critical qualifier, object, mechanism, or condition.

`none`: not mentioned, contradicted, or only irrelevant content is present.

# **Rules.** Do not award credit using external knowledge or what should be true. Judge only what is explicitly present in `CANDIDATE_ANSWER`. If the answer contradicts a nugget, label it `none`.

# **Output.** Output strict JSON parseable by `json.loads`: `{"results":[{"id":"N1","coverage":"full"}, ...]}`. Do not output any additional text.

**Figure 4:** Evaluation prompts. Prompt D constructs evidence-grounded static golden nuggets, while Prompt E judges nugget coverage using `full`/`partial`/`none` labels. Full executable prompts and JSON schemas are released with the artifact.

## C Additional Diagnostics

This appendix provides additional diagnostics omitted from the main text due to space limits. Table 10 gives the full turn-level region-only statistics behind the compact summary in Table 8. Table 11 summarizes supporting-backbone stability for the UA and Rewriter components.

**Table 10: Turn-level UA `unknown` rate conditioned on region-only questions on the benchmark set ($n = 518$).**

| $k$ | (Model, turn) | #region-only | region-only unknown |
|---|---|---|---|
| 1 | (Qwen, 1) | 64 | 0.766 |
| | (ERNIE, 1) | 136 | 0.787 |
| | (DeepSeek, 1) | 223 | 0.807 |
| | (Kimi, 1) | 48 | 0.750 |
| | (GPT, 1) | 121 | 0.785 |
| | (Claude, 1) | 148 | 0.824 |
| | (Gemini, 1) | 236 | 0.869 |
| 2 | (Qwen, 1) | 71 | 0.831 |
| | (Qwen, 2) | 24 | 0.917 |
| | (ERNIE, 1) | 54 | 0.852 |
| | (ERNIE, 2) | 38 | 0.868 |
| | (DeepSeek, 1) | 239 | 0.841 |
| | (DeepSeek, 2) | 61 | 0.885 |
| | (Kimi, 1) | 90 | 0.822 |
| | (Kimi, 2) | 23 | 0.870 |
| | (GPT, 1) | 108 | 0.787 |
| | (GPT, 2) | 24 | 0.833 |
| | (Claude, 1) | 198 | 0.838 |
| | (Claude, 2) | 22 | 0.818 |
| | (Gemini, 1) | 222 | 0.887 |
| | (Gemini, 2) | 25 | 0.800 |
| 3 | (Qwen, 1) | 52 | 0.769 |
| | (Qwen, 2) | 35 | 1.000 |
| | (Qwen, 3) | 12 | 0.917 |
| | (ERNIE, 1) | 67 | 0.881 |
| | (ERNIE, 2) | 51 | 0.882 |
| | (ERNIE, 3) | 33 | 0.879 |
| | (DeepSeek, 1) | 231 | 0.879 |
| | (DeepSeek, 2) | 74 | 0.824 |
| | (DeepSeek, 3) | 6 | 1.000 |
| | (Kimi, 1) | 94 | 0.830 |
| | (Kimi, 2) | 33 | 0.788 |
| | (Kimi, 3) | 13 | 0.692 |
| | (GPT, 1) | 113 | 0.779 |
| | (GPT, 2) | 27 | 0.778 |
| | (GPT, 3) | 9 | 0.889 |
| | (Claude, 1) | 203 | 0.852 |
| | (Claude, 2) | 25 | 0.920 |
| | (Claude, 3) | 15 | 1.000 |
| | (Gemini, 1) | 212 | 0.877 |
| | (Gemini, 2) | 19 | 1.000 |
| | (Gemini, 3) | 7 | 1.000 |

**Table 11: Supporting-backbone stability summary. UA agreement measures whether two supporting backbones produce the same known/unknown status; rewrite similarity is normalized text similarity between rewritten queries; unknown preservation measures whether rewrites avoid concretizing unknown answers.**

| Pair | UA agree | Rewrite sim. | Unknown preserv. |
|---|---|---|---|
| GPT-ERNIE vs. GPT-GPT | 0.865 | 0.771 | 0.988 |
| GPT-ERNIE vs. GPT-Claude | 0.874 | 0.709 | 0.987 |
| GPT-GPT vs. GPT-Claude | 0.881 | 0.726 | 0.990 |
| ERNIE-ERNIE vs. ERNIE-GPT | 0.885 | 0.798 | 0.989 |
| ERNIE-ERNIE vs. ERNIE-Claude | 0.880 | 0.745 | 0.990 |
| ERNIE-GPT vs. ERNIE-Claude | 0.891 | 0.756 | 0.994 |