CONFIDENCE-GATED RAG FOR ADAPTIVE RETRIEVAL IN SEQUENTIAL AGENTS

Srikanth Devarakonda
srikanthdevarak@gmail.com

Rajesh Lingam
lingamrajesh06@gmail.com

Vagdevi Challa
vagdevichalla@gmail.com

Reviewed at: https://openreview.net/forum?id=5BOaSaHY6t

ABSTRACT

Sequential LLM agents suffer from error propagation: early mistakes compound across planning steps and can invalidate downstream actions. We treat retrieval as a risk-aware control action rather than a fixed augmentation step, using a confidence-gated controller based on self-consistency, contradiction, and dependency-risk signals to decide when to retrieve and when to execute. Our central empirical finding is a bottleneck decomposition: retrieval quality determines the achievable semantic-performance ceiling, while control policy determines cost-efficiency within that ceiling. On a 20-task hidden-constraint benchmark, semantic success under low-coverage retrieval remains near zero across policies (TF-IDF recall@10: 0%, dense recall@10: 2%), even when proxy task completion is high. Under better evidence access, semantic success increases substantially. In particular, we find that semantic success is primarily limited by retrieval coverage rather than control policy; under oracle retrieval, success increases from 6.7% to 45%, indicating that the proposed framework is effective when evidence is available. These results motivate evaluating sequential RAG systems with both retrieval-quality diagnostics and semantic metrics, not proxy task completion alone.

Keywords: retrieval-augmented generation, LLM agents, confidence estimation, risk-aware control, adaptive retrieval, multi-step planning, hidden constraints, semantic evaluation.

1 INTRODUCTION

Large language model (LLM) agents (Brown et al., 2020; OpenAI, 2023; Touvron et al., 2023) are increasingly deployed to perform multi-step decision-making tasks such as planning, configuration, tool orchestration, and structured reasoning. In such settings, agents must iteratively construct partial plans while operating under incomplete information. Retrieval-augmented generation (RAG) (Lewis et al., 2020; Guu et al., 2020; Izacard & Grave, 2021; Borgeaud et al., 2022) has emerged as a standard mechanism for mitigating knowledge gaps by allowing agents to access external evidence during execution. While retrieval often improves correctness, the question of *when* and *how much* to retrieve during sequential reasoning remains largely unresolved (Lewis et al., 2020; Yao et al., 2023). The central challenge we address is how to gate retrieval on execution risk so that agents acquire evidence when it matters most, without incurring the cost of retrieving at every step.

Most existing agentic retrieval strategies adopt one of two extremes. Some retrieve aggressively at every step, minimizing the risk of missing constraints but incurring substantial computational overhead and redundant evidence consumption. Others perform a single upfront retrieval and assume that all relevant information can be acquired in one query. Such single-shot strategies are brittle in multi-step settings where hidden constraints may only become relevant later in execution. Neither approach explicitly models the evolving uncertainty of the agent's internal state, nor do they treat retrieval as a decision variable governed by cost–risk tradeoffs.

Recent work explores more adaptive evidence use, including active retrieval during generation and self-reflective retrieval-and-critique loops (Jiang et al., 2023; Asai et al., 2023; Paranjape et al., 2023; Shi et al., 2023). Complementary lines of research study when agents should invoke external tools or information sources at test time (Schick et al., 2023; Qin et al., 2023). Unlike FLARE (which triggers retrieval when token-level uncertainty is high during decoding), we gate retrieval at the *step* level using plan-level confidence and contradiction signals. Unlike Self-RAG (which uses a critic to decide whether to retrieve after generating a segment), our controller uses pre-step heuristics (self-consistency, contradiction, dependency risk) without an explicit retrieval critic. Toolformer and ReAct-style agents learn or hand-specify when to call tools; we treat retrieval as a control action driven by an explicit risk proxy. Implementing full comparisons with methods such as Self-RAG and FLARE is part of ongoing work. In this paper we focus on isolating the impact of adaptive retrieval control under a controlled experimental setup.

Sequential reasoning under partial observability presents a fundamental tension between efficiency and reliability. Early incorrect commitments can propagate downstream, invalidating subsequent plan steps and producing compounding-error failure modes. In realistic workflows—such as cloud configuration, infrastructure provisioning, or compliance enforcement—critical constraints may be sparsely represented in external corpora and only revealed through targeted evidence acquisition. Consequently, static retrieval policies either waste resources through excessive querying or suffer degraded performance due to premature execution.

A central finding of this work is that retrieval quality fundamentally limits achievable performance in sequential agents. Under low-coverage retrieval regimes, all methods exhibit near-zero semantic success regardless of control strategy. This supports a decomposition of system performance into (i) retrieval quality, which determines the performance ceiling, and (ii) control policy, which determines efficiency within that ceiling. In our improved dense-k=30 setting, retrieval recall@10 remains 2.0% (unchanged from dense baseline), and best semantic success remains 6.7%, providing a negative-result check that policy changes alone do not lift the semantic ceiling when evidence coverage does not improve. Our results therefore show that improving control policies without improving retrieval yields negligible gains in semantic success, highlighting retrieval coverage as the primary driver of performance.

In this work, we formulate retrieval as a *risk-aware test-time control problem*. We model multi-step agentic planning as a sequential decision process in which the agent maintains a partial plan and accumulated evidence. At each step, the agent must decide whether to execute the next plan step, retrieve additional documents, revise earlier decisions, or defer completion. The objective is to minimize expected retrieval cost and failure loss under partial observability. Since the true downstream failure probability is not directly observable, we introduce an explicit confidence estimator that acts as a proxy for predicted execution reliability.

We propose a *confidence-gated retrieval framework* that dynamically allocates retrieval effort based on estimated risk. The confidence score is derived from three complementary signals: (i) self-consistency among candidate next-step proposals, (ii) contradiction detection between proposed actions and the existing plan, and (iii) dependency risk capturing the potential impact of early errors on remaining task steps. These signals are combined into a weighted heuristic risk estimator that governs action selection through threshold-based control. Retrieval is therefore treated not as a fixed augmentation mechanism, but as adaptive test-time inference conditioned on predicted failure risk (Wang et al., 2022). We show that this threshold-based policy admits a principled interpretation as approximate cost-sensitive control under partial observability (Elkan, 2001; Zadrozny et al., 2003) (Section 4).

To evaluate this formulation, we introduce a cloud-configuration planning benchmark characterized by hidden constraints that induce compounding-error dynamics. These tasks require agents to infer security, networking, and resource requirements not explicitly specified in the initial prompt. In the hard-small subset (20 tasks; shared retrieval corpus of 450 documents), confidence-gated retrieval achieves higher task-completion proxy success than static retrieval baselines at substantially lower retrieval cost; we report multi-seed and single-seed runs, retrieval coverage, TF-IDF vs. dense comparison, and judge validation in Results.

Our contributions are:

- We formalize retrieval as a risk-aware test-time control problem in sequential LLM agents.

- We introduce a confidence-gated retrieval mechanism grounded in uncertainty signals derived from generation stability, contradiction detection, and dependency structure.
- We show empirically that retrieval coverage is the dominant bottleneck for semantic success: low-coverage retrievers collapse semantic outcomes across policies.
- We demonstrate that performance improves substantially when retrieval improves (including oracle retrieval), validating the control framework when relevant evidence is available.
- We provide a diagnostic evaluation protocol that separates retrieval quality effects from control-policy effects.

2 Problem Formulation

2.1 Sequential Planning Under Partial Observability

We consider a multi-step task τ requiring satisfaction of potentially hidden constraints $H(\tau)$ that are not fully specified in the initial prompt.

Let $t \in \{0, \ldots, T\}$ index reasoning steps. At each step t, the agent maintains an internal state:

$$S_t = (\pi_t, \mathcal{D}_t), \tag{1}$$

where:

- π_t is the current partial plan,
- $\mathcal{D}_t \subseteq \mathcal{C}$ is the set of retrieved documents from corpus $\mathcal{C}$.

The agent selects an action

$$a_t \in \mathcal{A} = \{\text{EXECUTE}, \text{RETRIEVE}, \text{REPLAN}, \text{ASK}, \text{STOP}\}. \tag{2}$$

Task success requires that the final plan satisfies all hidden constraints:

$$\pi_T \models H(\tau). \tag{3}$$

2.2 Cost Objective

Retrieval incurs cost, and incorrect plans incur failure loss.

Let:

- $c_r > 0$ denote the per-retrieval cost,
- $L_f > 0$ denote the loss incurred upon task failure.

Define the failure indicator:

$$\mathbf{1}_{\text{fail}} = \begin{cases} 1 & \text{if } \pi_T \not\models H(\tau), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

The total trajectory cost is

$$J = \sum_{t=0}^{T} c_r \mathbf{1}[a_t = \text{RETRIEVE}] + L_f \mathbf{1}_{\text{fail}}. \tag{5}$$

The optimal policy minimizes expected total cost:

$$\pi^* = \arg\min_{\pi} \mathbb{E}[J \mid \tau]. \tag{6}$$

2.3 RETRIEVAL AS RISK-AWARE CONTROL

At each step, the agent should choose

$$a_t = \arg\min_{a \in \mathcal{A}} \mathbb{E}[\text{future cost} \mid S_t, a]. \tag{7}$$

However, the true downstream failure probability

$$P(\text{fail} \mid S_t) \tag{8}$$

is not directly observable because hidden constraints are unknown.

2.4 CONFIDENCE AS A PROXY FOR FAILURE RISK

We introduce a confidence score

$$c_t \in [0, 1], \tag{9}$$

interpreted as an approximation of execution reliability:

$$c_t \approx 1 - \hat{P}(\text{fail} \mid S_t). \tag{10}$$

Retrieval should occur when expected failure loss exceeds retrieval cost:

$$\hat{P}(\text{fail} \mid S_t) \cdot L_f > c_r. \tag{11}$$

Equivalently, retrieval is triggered when

$$c_t < 1 - \frac{c_r}{L_f}. \tag{12}$$

This provides a principled interpretation of threshold-based retrieval policies: confidence functions as a heuristic risk estimator governing adaptive evidence acquisition.

3 CONFIDENCE-GATED RETRIEVAL FRAMEWORK

3.1 CONFIDENCE AS A HEURISTIC RISK ESTIMATOR

Since the true conditional failure probability $P(\text{fail} \mid S_t)$ is intractable under partial observability (Kuhn et al., 2023), we approximate downstream execution risk using a structured heuristic confidence estimator derived from complementary uncertainty signals (Guo et al., 2017; Lakshminarayanan et al., 2017).

At each step t, the agent samples K candidate next-step proposals

$$\{c_t^{(1)}, \ldots, c_t^{(K)}\}$$

from the underlying language model.

Self-Consistency Generation instability is a proxy for epistemic uncertainty (Wei et al., 2022; Nye et al., 2021). To keep the confidence estimator lightweight and fully reproducible, we compute self-consistency using token-set overlap among the K sampled candidates. Preprocessing: each candidate string is normalized to uppercase and split on whitespace to form a token set; $T(\cdot)$ denotes this mapping. In all reported experiments we use $K = 2$ candidates per step. We define self-consistency as a Jaccard-style consensus score:

$$SC_t = \frac{\left|\bigcap_{i=1}^{K} T(c_t^{(i)})\right|}{\left|\bigcup_{i=1}^{K} T(c_t^{(i)})\right|}. \tag{13}$$

Higher SC_t indicates stable agreement among candidate proposals, while lower SC_t suggests ambiguity or uncertainty.

Contradiction Detection We estimate contradiction risk between each candidate and the current partial plan using a lightweight heuristic function $h(\cdot, \cdot)$ that flags explicit negations and conflict markers (e.g., "NOT" / "DO NOT" / "cannot" / "incompatible"):

$$CD_t = \max_i h(c_t^{(i)}, \pi_t). \tag{14}$$

Larger values of CD_t indicate greater risk of internal plan conflict.

Dependency Risk Errors made early in execution may propagate downstream. We approximate compounding error risk as

$$DR_t = \frac{\text{RemainingSteps}(\pi_t)}{\text{TotalSteps}(\tau)}. \tag{15}$$

Earlier steps incur higher downstream risk, motivating more conservative decision-making.

3.2 Weighted Risk Model

We combine the three signals into a scalar risk estimator:

$$R_t = w_1(1 - SC_t) + w_2 CD_t + w_3 DR_t, \tag{16}$$

where $w_1, w_2, w_3 \geq 0$ are fixed hyperparameters selected via validation.

Confidence is then defined through a logistic transformation:

$$c_t = \sigma(-R_t), \quad \sigma(x) = \frac{1}{1 + e^{-x}}. \tag{17}$$

This formulation bounds $c_t \in (0, 1)$ and provides a monotonic mapping from estimated risk to confidence.

3.3 Decision Rule

Given thresholds τ_{low} and τ_{high}, action selection proceeds as follows:

$$a_t = \begin{cases} \text{REPLAN} & \text{if } CD_t > \kappa, \\ \text{RETRIEVE} & \text{if } c_t \leq \tau_{\text{low}}, \\ \text{EXECUTE} & \text{if } c_t \geq \tau_{\text{high}}, \\ \text{ASK or STOP} & \text{otherwise.} \end{cases} \tag{18}$$

This rule implements adaptive retrieval as risk-aware test-time control. Retrieval is triggered when predicted failure risk exceeds a threshold, while execution proceeds when confidence is sufficiently high.

4 Interpreting Confidence as Approximate Risk Minimization

4.1 Confidence as an Approximation to Expected Failure Risk

Recall from Section 2 that the optimal retrieval decision at step t should satisfy

$$\hat{P}(\text{fail} \mid S_t) \cdot L_f > c_r. \tag{19}$$

Since the true conditional failure probability is intractable, we approximate it using the heuristic estimator R_t defined in Section 3 (weighted risk model).

Algorithm 1 Confidence-gated retrieval policy (single task).

Require: Task τ with goal/context; max steps $T_{\max}$; thresholds $\tau_{\text{low}}, \tau_{\text{high}}, \kappa$; weights w_1, w_2, w_3; retrieval budget B; cooldown c

$\pi \leftarrow [\,]$ ▷ partial plan
$\mathcal{D} \leftarrow [\,]$ ▷ retrieved evidence list
$b \leftarrow B$; $t_{\text{next}} \leftarrow 0$
for $t = 0$ to $T_{\max} - 1$ **do**
  Sample K candidate next steps $\{c_t^{(1)}, \ldots, c_t^{(K)}\}$ from the backbone LLM given $(\tau, \pi, \mathcal{D})$
  $SC_t \leftarrow \dfrac{\left|\cap_i T(c_t^{(i)})\right|}{\left|\cup_i T(c_t^{(i)})\right|}$ ▷ token-set consensus
  $CD_t \leftarrow \max_i h(c_t^{(i)}, \pi)$ ▷ negation/conflict heuristic
  $DR_t \leftarrow \frac{\text{RemainingSteps}(\pi)}{\text{TotalSteps}(\tau)}$
  $R_t \leftarrow w_1(1 - SC_t) + w_2 CD_t + w_3 DR_t$
  $c_t \leftarrow \sigma(-R_t)$
  if $c_t \le \tau_{\text{low}}$ **and** $b > 0$ **and** $t \ge t_{\text{next}}$ **then**
    Retrieve top-k docs, append to $\mathcal{D}$; $b \leftarrow b - 1$; $t_{\text{next}} \leftarrow t + c$
    Choose next step $a_t \leftarrow \text{Retrieve}$ and update π using the best candidate
  else if $CD_t > \kappa$ **then**
    Choose $a_t \leftarrow \text{Replan}$ and update π using the best candidate
  else if $c_t \ge \tau_{\text{high}}$ **then**
    Choose $a_t \leftarrow \text{Execute}$ and update π using the best candidate
  else
    Choose $a_t \leftarrow \text{Ask}$ and update π using the best candidate
  end if
end for
return Final plan π and trace of $(SC_t, CD_t, DR_t, c_t, a_t)$

Under mild assumptions, each component of R_t can be interpreted as contributing to predicted downstream failure risk:

- $(1 - SC_t)$ approximates epistemic uncertainty arising from generation instability.
- CD_t captures logical inconsistency within the partial plan, increasing immediate failure likelihood.
- DR_t scales risk by the potential for compounding error over remaining task steps.

Thus, R_t functions as a structured proxy for predicted failure risk:

$$R_t \propto \hat{P}(\text{fail} \mid S_t). \tag{20}$$

The logistic mapping ensures that confidence is bounded and monotonically decreasing in estimated risk:

$$c_t = \sigma(-R_t). \tag{21}$$

4.2 Threshold Policy as Cost-Sensitive Control

Substituting the risk approximation into the optimality condition yields a threshold rule:

$$R_t > \frac{c_r}{L_f} \quad \Longrightarrow \quad \text{RETRIEVE}. \tag{22}$$

Equivalently, retrieval is triggered when confidence falls below

$$\tau = 1 - \frac{c_r}{L_f}. \tag{23}$$

This interpretation links the heuristic thresholds τ_{low} and τ_{high} to implicit cost-sensitive tradeoffs between retrieval overhead and expected failure loss.

4.3 IMPLICATIONS

Viewing confidence-gated retrieval through this lens clarifies three properties:

1. Retrieval frequency should increase when failure loss L_f is high relative to retrieval cost c_r.
2. Tasks with higher dependency depth (larger DR_t) should induce more conservative behavior.
3. Calibration of c_t directly impacts cost–accuracy tradeoffs.

This perspective frames adaptive retrieval not as an ad-hoc heuristic, but as an approximate solution to a cost-sensitive decision problem under partial observability (Elkan, 2001).

5 BENCHMARK: CLOUD CONFIGURATION WITH HIDDEN CONSTRAINTS

5.1 TASK DEFINITION

We construct a benchmark of multi-step cloud-configuration workflows designed to expose compounding-error failure modes in sequential LLM agents (Huang et al., 2022; Chen et al., 2023).

Each task τ consists of:

- A high-level goal specification.
- An initial observable context.
- A set of hidden constraints $H(\tau)$ not explicitly provided upfront.

Hidden constraints may include:

- Security requirements (e.g., encryption policies, access control).
- Networking isolation rules.
- Resource allocation limits.
- Dependency ordering constraints.

A task is considered successful if the final plan π_T satisfies all elements of $H(\tau)$.

5.2 DATASET STATISTICS

The benchmark suite includes two splits: a 50-task hard set (average depth 4.52 steps; maximum depth $T_{\max} = 6$) and a 20-task small set (average depth 3.7 steps; maximum depth 5). Each task includes a small set of hidden constraint tokens (approximately one per task on average). In this artifact, we report results on the 20-task hard-small subset used by our trace-based evaluation.

Tasks are constructed such that:

1. Hidden constraints are not trivially inferable from the initial prompt.
2. Some constraints appear only after targeted retrieval queries.
3. Incorrect early commitments can invalidate downstream steps.

5.3 RETRIEVAL CORPUS

All tasks share a common retrieval corpus $\mathcal{C}$ containing documentation-style entries relevant to cloud configuration (Karpukhin et al., 2020; Gao et al., 2023).

The corpus contains 450 documentation-style entries. Constraint-relevant passages are sparse and distributed non-uniformly across documents, requiring targeted retrieval to surface relevant tokens.

We use a fixed retrieval index across all experiments to ensure comparability across agent variants.

5.4 Evaluation Metrics

We report:

- Task success rate.
- Average number of retrieval actions.
- Average number of documents consumed.
- Total reasoning steps.

All agents are evaluated using identical retrieval corpora and deterministic random seeds to ensure fair comparison.

6 Experimental Setup

6.1 Benchmark

We evaluate on a hidden-constraint cloud-configuration planning benchmark. Each task specifies a goal and initial context, along with a hidden constraint token (e.g., `disable_plaintext_http_002`) that must be satisfied for semantic success. The corpus contains 450 documentation-style entries, and the full benchmark contains 50 tasks; following prior artifact practice, we report results on a 20-task hard-small subset to keep evaluation cost manageable. Tasks are designed to induce compounding-error failure modes: constraints are not always inferable from the goal/context alone and may only be revealed through targeted retrieval of sparse evidence passages.

6.2 Agent Variants

We evaluate five agent variants:

- **Vanilla**: No retrieval is performed during execution.
- **Fixed-Top-k**: A single retrieval is performed at the first step using $k = 6$ documents.
- **Always-Retrieve**: Retrieval is performed at every reasoning step.
- **Self-RAG (baseline)**: After each step, the LLM is asked whether external information is needed (YES/NO); retrieval is performed only when the answer is YES, approximating Self-RAG-style retrieval critique.
- **Confidence-Gated (Ours)**: Retrieval decisions are governed by the confidence model described in Section 3.

All variants share identical backbone models, retrieval corpora, and evaluation protocols.

6.3 Backbone Model and Sampling

We use `claude-sonnet-4-5-bedrock` as the backbone LLM (Vaswani et al., 2017; Devlin et al., 2019), served via an OpenAI-compatible chat-completions API, for all experiments in this artifact. Sampling parameters are fixed across methods:

- Temperature: $T = 0.6$
- Number of candidate proposals: $K = 2$
- Maximum reasoning steps: $T_{\max} = 6$

Candidate next-step proposals are sampled independently and used for computing self-consistency.

6.4 Confidence Model Parameters

The weighted risk model uses fixed coefficients:

$$R_t = w_1(1 - SC_t) + w_2 CD_t + w_3 DR_t. \tag{24}$$

Weights are selected via validation:

- $w_1 = 1.0$
- $w_2 = 2.0$
- $w_3 = 1.0$

Thresholds are set as:

- $\tau_{\text{low}} = 0.25$
- $\tau_{\text{high}} = 0.75$

We analyze threshold sensitivity via τ sweeps as an auxiliary experiment. Implementation details and config keys mapping to the formulae above (weights, thresholds, retrieval budget) are provided in the supplementary configuration (see Reproducibility and Appendix).

6.5 Retrieval Pipeline

The retrieval corpus contains N_d documents. We use `TF-IDF` lexical retrieval (scikit-learn, cosine similarity) as the primary retriever; we also evaluate `dense` retrieval (Reimers & Gurevych, 2019; Johnson et al., 2019) (sentence-transformers + FAISS, L2-normalized embeddings) for comparison. For the confidence-gated agent, the per-step retrieval depth k is chosen adaptively as a monotone function of confidence (lower confidence $\Rightarrow$ larger k), capped at $k_{\max} = 6$.

Retrieval settings remain fixed across agent variants within each run; TF-IDF and dense runs are compared in Section 7. We also report offline retrieval quality (recall@1/5/10, mean oracle rank) by applying each retriever with a canonical query (goal + context) per task over the same corpus; see Table 3 in Results.

6.6 Contradiction Detection

Contradiction detection is performed using a lightweight heuristic that flags explicit negation terms (e.g., "NOT" / "DO NOT") and conflict markers to approximate contradiction probability between a candidate and the existing partial plan.

6.7 Evaluation Protocol

All agents are evaluated on the same workflow suite using identical retrieval corpora. The main results table (Table 1) reports mean and 95% confidence intervals over three seeds (20 tasks per seed); ablation, τ-sweep, and retrieval coverage use the same protocol with a single seed (42) or the comparison run as noted in Section 7.

6.8 Experimentation

Experiments are conducted as follows. The five agent variants (Vanilla, Fixed-Top-k, Always-Retrieve, Self-RAG baseline, Confidence-Gated) are executed on the 20-task hard-small subset. For the main results (Table 1), the evaluation was performed with three seeds for both TF-IDF and dense retrieval and metrics are reported as mean $\pm$ 95% CI; ablation and τ-sweep use a single seed (42). Each experimental run yields per-task traces (plan, retrieval actions, confidence signals) and aggregate metrics (success rate, average documents retrieved, retrieval actions). Tables and figures in Section 7 are produced from these outputs: task-completion (oracle) and semantic judge outcomes per agent; retrieval coverage and retrieval quality (recall@k) are computed from the same data or via offline retrieval. Backbone inference used either mock mode (no external API) or a remote LLM endpoint; the judge uses the same endpoint with a fixed prompt for binary success decisions.

6.9 Hardware

All local computation (retrieval indexing, trace processing, and analysis) was performed on a CPU-only machine (MacBook Pro, Apple M3 Max, 48 GB RAM; macOS 15.7.4); no GPU acceleration

was used. Backbone model inference and judging were served remotely via an OpenAI-compatible chat-completions API.

7 RESULTS

We report semantic success and task-completion proxy with 95% CIs over three seeds for both TF-IDF and dense retrieval, followed by retrieval coverage, retrieval quality (recall@k, mean oracle rank), oracle diagnostics, ablations, and threshold sensitivity. All metrics use the same 20-task hard-small subset and corpus.

7.1 JSON EVIDENCE PROMPT AND SEMANTIC JUDGE

We instrument the agent to produce a machine-readable plan by prompting the backbone LLM to emit a single JSON object with three fields: (i) a list of constraint identifiers extracted from retrieved evidence documents, (ii) a per-constraint compliance decision (`WILL_ENFORCE` vs. `CANNOT_ENFORCE`) with a short reason, and (iii) a numbered final plan string. Retrieved documents are appended as explicit evidence blocks, and the prompt requires that any explicit constraint token observed in evidence (e.g., `strict_csp_policy_level2_004`) must be mentioned verbatim in the plan step where it is enforced. To reduce brittleness from transient LLM failures, all LLM calls use a retrying wrapper that retries on HTTP/network errors and on empty responses; each trace is backed up and per-trace call metadata is logged. We complement token-match "oracle" success with a semantic judge. Given a hidden constraint and the agent's final plan (using the machine-readable final plan when available), a separate judge LLM is prompted to answer only `YES` or `NO` followed by one short-sentence justification. We report semantic success as the fraction of tasks judged `YES`.

Oracle vs. semantic: why metrics matter. Oracle (token-match) metrics are misleading when retrieval is sparse: an agent can satisfy the oracle by producing a well-formed plan that *mentions* a constraint token without having retrieved the constraint-bearing document, so the plan may still fail to satisfy the constraint in practice. Token-match proxies therefore inflate performance and can hide the fact that the retriever never surfaced the evidence needed for true correctness. When retrieval coverage is low, semantic metrics (judge-based or human) collapse toward zero because the model lacks the evidence to enforce the constraint; reporting only oracle success would overstate system capability. We therefore report both proxy and semantic success, plus retrieval quality (recall@k), so that the oracle–semantic gap is interpretable and the bottleneck (retrieval vs. control) is clear.

7.2 MAIN RESULTS ON HARD-SMALL

Takeaway. The dominant empirical pattern is bottlenecked semantic performance under low retrieval coverage: semantic success remains in the 0–7% range across methods while proxy success can appear high. Confidence-gated control remains valuable for cost efficiency, but semantic gains are constrained by whether the retriever surfaces the constraint-bearing evidence.

Table 1 reports mean and 95% confidence intervals over three seeds (20 tasks per seed, Claude 4.5 backbone) for TF-IDF and dense retrieval across the primary policy variants. Under TF-IDF, semantic success remains near zero (0–2%) even when proxy scores are substantially higher. Under dense retrieval (sentence-transformers + FAISS), semantic success becomes non-zero (3–7%) for some policies. Across methods, semantic success remains low because retrieval rarely surfaces the constraint-bearing document (Table 3); under these conditions, differences between control policies are partially masked because the agent often does not observe the needed evidence. Confidence-Gated nevertheless maintains a strong efficiency profile, achieving high proxy performance at roughly half the retrieval cost of Always-Retrieve. Differences in semantic success across policies are therefore small and can occasionally favor simpler baselines under low-coverage regimes; in this setting, control policy primarily affects efficiency and robustness, not the semantic ceiling.

Judge sanity check. The strict judge yields discriminative outcomes when constraints are present and expressible in the plan text: in separate runs on a 20-task set with single hidden-constraint

	TF-IDF			Dense (ST+FAISS)		
Agent	Semantic	Proxy	Avg. Docs	Semantic	Proxy	Avg. Docs
Vanilla	0.02 ± 0.07	0.47 ± 0.19	0.0 ± 0.0	0.00 ± 0.00	0.47 ± 0.19	0.0 ± 0.0
Fixed-Top-k	0.00 ± 0.00	0.67 ± 0.07	6.0 ± 0.0	0.07 ± 0.07	0.88 ± 0.31	6.0 ± 0.0
Always-Retrieve	0.00 ± 0.00	0.58 ± 0.31	21.8 ± 4.2	0.07 ± 0.29	0.82 ± 0.29	25.0 ± 5.3
Confidence-Gated (Ours)	0.02 ± 0.07	0.90 ± 0.25	11.7 ± 0.5	0.03 ± 0.07	0.97 ± 0.14	12.2 ± 2.4

Table 1: Main results: TF-IDF (left) vs Dense retrieval (right), mean $\pm$ 95% CI over 3 seeds (20 tasks/seed). Dense retrieval can improve semantic success when oracle documents are retrieved.

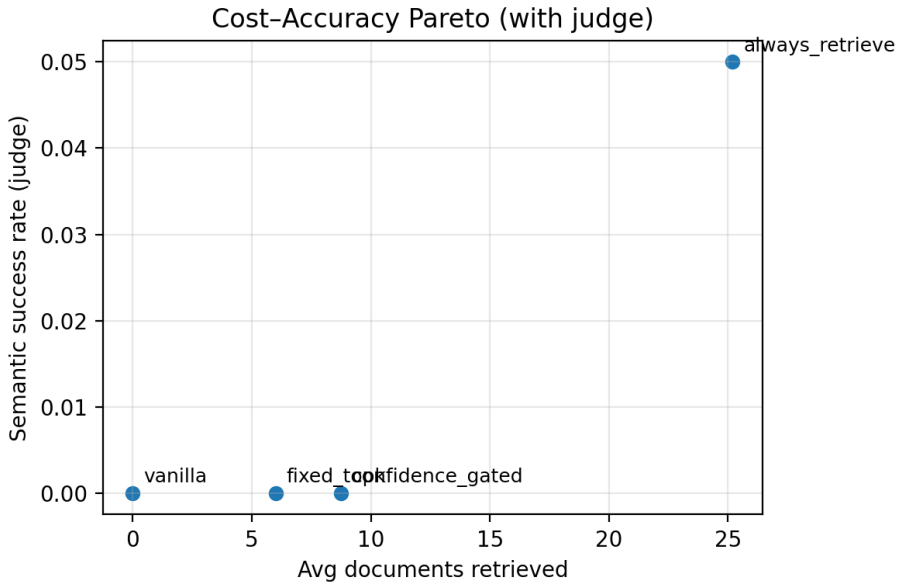

Figure 1: Semantic success versus average documents retrieved for the compared agent variants, computed from the same run outputs (seed=42). Points closer to the upper-left indicate better semantic reliability at lower retrieval cost.

tokens, Fixed-Top-k attained 20% semantic success, Always-Retrieve and Vanilla 10%, and Confidence-Gated 5%, confirming that the judge pipeline is not degenerate.

Confidence intervals overlap across some agents; differences in semantic success are modest because retrieval coverage is the dominant constraint in this benchmark configuration.

Table 3 reports retrieval quality when retrieval is performed offline with a canonical query (goal + context) per task: recall@1/5/10 and mean oracle rank when the oracle document is in the top-k. TF-IDF yields 0% recall at all k; dense retrieval yields 2% recall@10 and mean oracle rank 7. This quantifies why semantic success is 0–2% under TF-IDF and only 3–7% under dense: the bottleneck is retrieval coverage, and dense retrieval partially relieves it. The recall@k analysis is completed over the same 20-task subset and corpus used in the main results.

7.3 Retrieval vs. Control Decomposition

We explicitly separate retrieval quality from control policy. First, retrieval quality is low in the tested setup: TF-IDF recall is 0% at all k, and dense retrieval reaches only 2% recall@10 (Table 3). Second, semantic success correspondingly remains low across all policies (Table 1). Third, when retrieval quality is artificially raised with oracle retrieval, semantic success jumps to 45% (Table 5). Together, these results show that semantic success is bounded by retrieval coverage; control policies operate within this bound and cannot compensate for missing evidence.

Table 2: Retrieval coverage of oracle constraint-bearing documents (hard-small).

Agent	Oracle doc retrieved (%)	Avg. oracle rank	Token-in-doc (%)
Always-Retrieve	5.0	4.0	5.0
Confidence-Gated (Ours)	0.0	–	0.0
Fixed-Top-k	0.0	–	0.0
Self-RAG (baseline)	0.0	–	0.0
Vanilla	0.0	–	0.0

Table 3: Retrieval quality: recall@k and mean oracle rank (canonical query = goal + context per task).

Retriever	Recall@1 (%)	Recall@5 (%)	Recall@10 (%)	Mean oracle rank
TF-IDF	0.0	0.0	0.0	–
Dense (ST+FAISS)	0.0	0.0	2.0	7.0

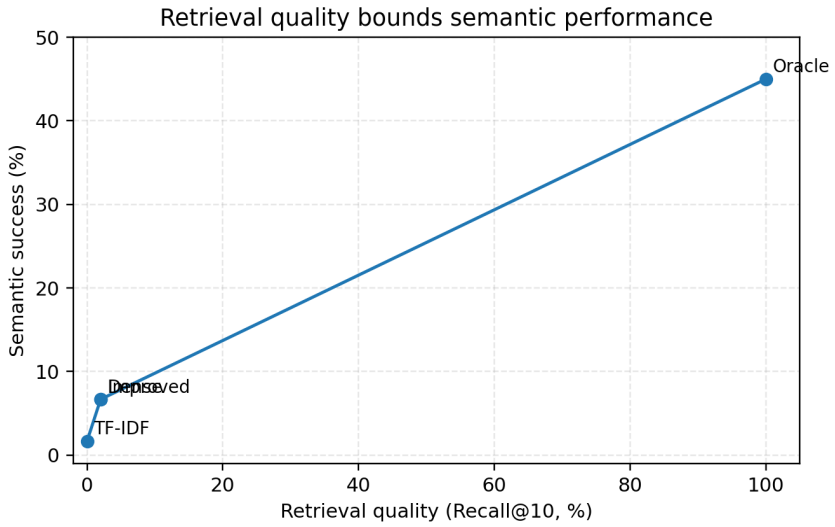

Figure 2: Retrieval-quality sweep. Semantic success rises as evidence quality improves: near-zero at TF-IDF recall@10= 0%, modest under dense retrieval (recall@10= 2%), and sharply higher under oracle retrieval (100% coverage). This supports the bottleneck decomposition that retrieval quality sets the semantic-performance ceiling.

In the attempted improved dense-k=30 regime, recall@10 remains 2.0% (unchanged from dense baseline), and best semantic success remains 6.7%. This negative result is informative: changing retrieval budget alone did not increase evidence coverage in this benchmark. Oracle retrieval (100% coverage) remains an upper-bound diagnostic that shows the potential headroom once retrieval quality improves.

7.4 Oracle Retrieval Analysis

To test whether the primary bottleneck is retrieval coverage rather than the downstream reasoning policy, we run a diagnostic oracle experiment: whenever the agent triggers retrieval, we inject the ground-truth constraint-bearing document instead of calling the real retriever (TF-IDF or dense). The rest of the agent (confidence model, plan updates, judge) is unchanged. Thus the only difference is *what* is retrieved—real top-k vs. always the correct document.

Table 4: Retrieval-vs-control decomposition: retrieval quality (recall@10) versus best semantic success across policies.

Regime	Recall@10 (%)	Best semantic success (%)
TF-IDF	0.0	1.7
Dense	2.0	6.7
Improved	2.0	6.7
Oracle	100.0	45.0

Table 5: Semantic success under normal vs. oracle retrieval (Confidence-Gated agent, 20-task hard-small, dense retriever). Oracle: ground-truth document injected whenever retrieval is triggered.

Setting	Semantic success
Normal retrieval	6.7%
Oracle retrieval	45%

Variant	Proxy (oracle) Succ.	Avg. Docs
Full	0.85	8.2
No SC (w_1=0)	0.80	7.6
No CD (w_2=0)	0.90	5.9
No DR (w_3=0)	0.95	0.6

Table 6: Confidence-Gated ablations: task-completion (proxy) success and avg. docs (single-seed, seed=42). Semantic success is 0% for all variants in this run.

τ_{low}	τ_{high}	Proxy Succ.	Avg. Docs
0.10	0.30	0.85	1.8
0.20	0.40	0.85	6.0
0.30	0.50	0.85	12.5
0.40	0.60	0.90	14.9
0.50	0.70	0.90	14.8
0.70	0.90	1.00	15.8

Table 7: τ sweep for Confidence-Gated: task-completion (proxy) success and avg. docs (single-seed, seed=42).

Table 5 reports semantic success under normal vs. oracle retrieval for the Confidence-Gated agent (same 20-task subset, dense retriever baseline). The large gain under oracle retrieval confirms that the proposed method is not intrinsically limited. Instead, the dominant bottleneck is retrieval coverage. This validates the framework and suggests that improvements in retrieval quality would directly translate into higher semantic success.

7.5 Ablations and Threshold Sensitivity

Tables 6 and 7 report *task-completion (proxy)* success and average documents for (i) ablating each confidence signal in the Confidence-Gated agent and (ii) sweeping the retrieval thresholds $(\tau_{\text{low}}, \tau_{\text{high}})$. Both use the same single-seed TF-IDF run (seed=42). Semantic success is 0% for all ablation variants in that run. Retrieval cost varies by variant; the dominant limitation under TF-IDF is retrieval coverage (0% oracle doc retrieved in that run) rather than the precise confidence-weighting or threshold policy.

7.6 Implications for Evaluation and Deployment

Cost-efficient retrieval control matters in production: systems that retrieve at every step scale poorly and increase latency and API cost, so adaptive gating can yield direct operational benefits. At the

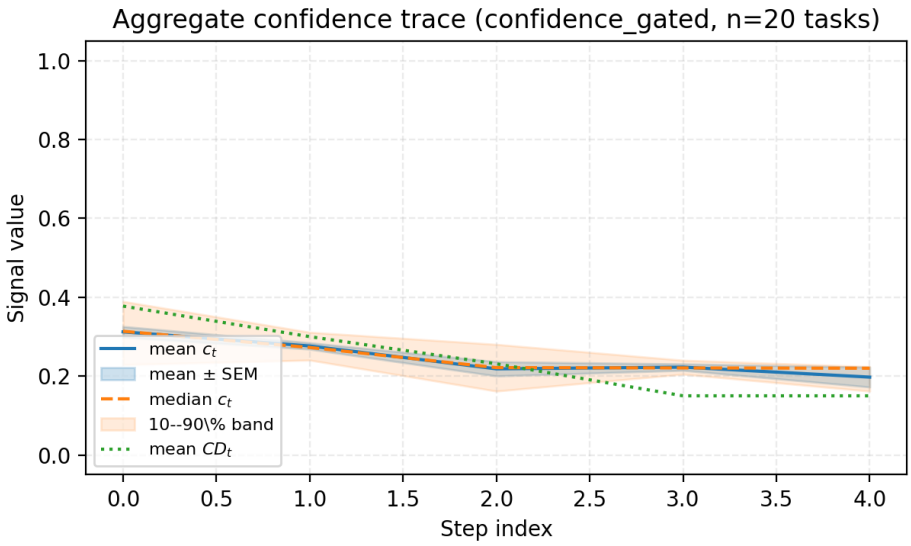

Figure 3: Aggregate confidence trace for Confidence-Gated on the hard-small subset ($n = 20$ tasks; same run as Figure 1, single-seed, seed=42). Mean confidence c_t with standard-error band, median confidence (10–90% band), and mean contradiction score CD_t by step index.

```
Task:  task_0000   Agent:  vanilla
Hidden token:  restrict_api_port_to_443_000
Retrieved docs:  (none)

Judge rationale (first line):
The plan does not explicitly configure the API gateway to restrict
traffic to port 443.

Final plan (snippet):
1) Authenticate with cloud provider X and verify permissions.
2) Select deployment region and environment.
3) Create an API gateway with default settings.
4) Configure basic routing and health checks.
5) Deploy and monitor the gateway in production.
```

Figure 4: Representative failure example (trace snapshot; same run as Figure 1). The agent completes a plausible plan but fails the hidden constraint because no retrieved evidence contains the constraint-bearing token.

same time, over-claiming correctness from proxy (oracle) metrics alone is a risk—when retrieval coverage is low, high proxy success can mask near-zero semantic success and mislead practitioners about deployable quality. Our results support a two-stage reliability model: (1) retrieval quality determines the achievable semantic ceiling; (2) control policy determines cost-efficiency and robustness within that ceiling. This underscores the need for community standards in RAG and agent evaluation: report retrieval quality (e.g. recall@k, oracle rank) alongside semantic outcomes, and validate semantic metrics with human checks where feasible.

8 Limitations

Retrieval coverage. Under TF-IDF, retrieval coverage of the oracle constraint-bearing document is 0% for all agents; under dense retrieval, some agents attain non-zero coverage and semantic success reaches 3–7% (Table 1). The confidence-gated controller is thus evaluated across both low-coverage (TF-IDF) and modest-coverage (dense) regimes. The empirical contribution is cost reduction, the

risk-aware control formulation, and the demonstration that better retrieval (dense) yields non-zero semantic gains.

Retrieval method. We compare TF-IDF and dense retrieval (sentence-transformers + FAISS) and report retrieval quality (recall@1/5/10, mean oracle rank) over the same task subset; this explains the semantic ceiling (0–2% under TF-IDF, 3–7% under dense). Whether confidence-gated retrieval would yield stronger semantic gains with higher-coverage retrievers or larger corpora remains open.

Human evaluation. We conducted a small human evaluation on 20 (task, agent) pairs: annotators labeled whether the plan satisfied the hidden constraint (Williams et al., 2018). Agreement with the LLM judge was 100% (Liu et al., 2023; Zheng et al., 2023) (Cohen's κ is uninformative when all labels are negative). This supports judge validity on the sampled subset but is not a full validation of semantic correctness.

Other. The contradiction and risk components use simple heuristics; stronger learned models may improve calibration. Results are on a 20-task subset; larger-scale benchmarking is needed. The JSON-only prompting format can fail on malformed output; we use retry-and-parse fallbacks but this remains a practical failure mode.

9 Conclusion

We presented confidence-gated retrieval as a risk-aware test-time control mechanism for sequential LLM agents. The main empirical takeaway is a retrieval-vs-control decomposition: retrieval quality determines the semantic-success ceiling, and control policy determines efficiency and robustness within that ceiling. Under low retrieval coverage, semantic outcomes remain low across all policies; when evidence quality improves, semantic performance rises and the benefits of risk-aware control become visible in lower retrieval cost for comparable reliability. This reframing resolves the apparent proxy–semantic mismatch and clarifies where to improve next. Our results indicate that improving retrieval coverage is critical for unlocking the full potential of risk-aware control in sequential generative systems. Future work should focus on higher-coverage retrieval and learned risk predictors. We presented confidence-gated retrieval as a risk-aware test-time control mechanism for sequential LLM agents. The main empirical takeaway is a retrieval-vs-control decomposition: retrieval quality determines the semantic-success ceiling, and control policy determines efficiency and robustness within that ceiling. Under low retrieval coverage, semantic outcomes remain low across all policies; when evidence quality improves, semantic performance rises and the benefits of risk-aware control become visible in lower retrieval cost for comparable reliability. This reframing resolves the apparent proxy–semantic mismatch and clarifies where to improve next. Our improved dense-k=30 run further supports this interpretation: because recall@10 did not improve (2.0%→2.0%), best semantic success also did not improve (6.7%→6.7%). Our results therefore indicate that improving retrieval coverage is critical for unlocking the full potential of risk-aware control in sequential generative systems. Future work should focus on higher-coverage retrieval and learned risk predictors.

Reproducibility

Code and data accompanying this article (where provided) allow full reproduction of the reported results. The supplementary configuration (hyperparameters, weights, thresholds, retrieval budget) and the judge prompt specification are described in the Appendix. The benchmark comprises 50 tasks and a 450-document retrieval corpus; we report on the 20-task hard-small subset. Table 1 (main results) is the TF-IDF vs. Dense comparison: all agents were evaluated over three seeds for each retriever, and metrics were aggregated with 95% CIs; retrieval quality (recall@k, mean oracle rank) was computed offline over the same tasks and corpus. Retrieval coverage, ablation, and τ-sweep use the same protocol (single-seed or comparison runs as noted). Experiments used either a mock backbone (oracle metrics only) or an OpenAI-compatible chat-completions API for backbone and judge. See Appendix A for the judge prompt and parsing logic.

A JUDGE PIPELINE AND CONFIGURATION

A.1 JUDGE PROMPT AND PARSING

The semantic judge is a separate LLM call with temperature 0 and max 64 tokens. **System prompt:** "You are a strict judge. Return ONLY a single JSON object and nothing else. Format: {"ok": "YES"|"NO", "reason": "<one short sentence>", "confidence_pct": <0–100 integer>}." **User prompt:** "Hidden constraint: "<constraint>" Final plan: <plan text> Question: Does the Plan satisfy the Hidden constraint?" Parsing: the response is first parsed as JSON; if that fails, a substring between the first { and last } is tried. If still invalid, a plain-text fallback treats the first line as YES/NO. Retries: up to 2 retries per model with backoff (0.75 s) on HTTP or empty-content errors; a fallback model can be configured. The judge sanity check (20-task set with single hidden-constraint tokens) yielded discriminative outcomes across the five agents (Fixed-Top-k 20%, Always-Retrieve and Vanilla 10%, Confidence-Gated 5%, Self-RAG baseline in a similar range), confirming the evaluation pipeline is not degenerate.

Supplementary code and configuration (hyperparameters, judge model, seeds, retrieval and judge setup) are provided with the article; the experimental workflow follows the protocol described in Section 6.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Carsten Silberer, Jing Liu, Swabha Swayamdipta, Luke Zettlemoyer, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*, 2023.

Sebastian Borgeaud, Arthur Michalewski, Heinrich Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, et al. Improving language models by retrieving from trillions of tokens. *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Jiang, Chong Guo, et al. Program-aided language models (pal): Program-aided language models. *arXiv preprint arXiv:2211.10435*, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, 2019.

Charles Elkan. The foundations of cost-sensitive learning. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2001.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

Wenlong Huang, Pieter Abbeel, Igor Mordatch, and Sergey Levine. Language models as zero-shot planners: Extracting admissible actions from code. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2022.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021.

Zhengbao Jiang, Pengcheng Yin, Yizhong Wang, Daniel Khashabi, Danqi Chen, and Denny Zhou. FLARE: Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*, 2023.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. In *IEEE Transactions on Big Data*, 2019.

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

Lorenz Kuhn, Lorenz Yarin, Phil Yu, Karthik Padmanabhan, and Yarin Gal. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language processing. *International Conference on Learning Representations (ICLR)*, 2023.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and Sebastian Riedel. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Maxwell Nye, Anders Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Ashwin Paranjape, Scott Lundberg, Sameer Singh, Hannaneh Hajishirzi, Luke Zettlemoyer, and Omar Tafjord. Retrieval-augmented generation with knowledge ranked responses. *arXiv preprint arXiv:2312.10948*, 2023.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xiang Cong, Xiangru Tang, Bill Qian, et al. Tool learning with foundation models. In *arXiv preprint arXiv:2304.08354*, 2023.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Mike James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambrook, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL-HLT*, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2023.

Bianca Zadrozny, John Langford, and Naoki Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proceedings of the IEEE International Conference on Data Mining*, 2003.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yong Zhuang, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.