



International Journal of Research Publication and Reviews

Journal Homepage: www.ijrpr.com ISSN 2582-7421

AIKDAP (AI Knowledge Discovery And Analytics Platform) : A Review of Retrieval-Augmented, Multi-Agent and Provenance-Aware AI Research Assistants

Prof. Ms. Madhuri Bhaisare, Harshita Dayma, Vineet Singh, Thoravi Daf, Divya Bhangre

Department of Artificial Intelligence and Data Science, Priyadarshini College of Engineering, Nagpur, Maharashtra, India

ABSTRACT—

Large language model (LLM) assistants are increasingly used to read papers, analyse data and draft reports, yet most ask the model to stay grounded through a prompt instruction, which is a request rather than a guarantee. This paper reviews the literature on making such assistants verifiable. It surveys work on hallucination, retrieval-augmented generation, two-stage retrieval and reranking, attribution and citation checking, multi-agent orchestration, natural-language data analysis, explainability and provenance, and the reliability engineering of LLM services. From this body of work it derives a five-level taxonomy of grounding enforcement that separates prompt-level requests from guarantees implemented in backend control flow, and it compares representative research assistants against that taxonomy. AIKDAP, a project-scoped multi-agent platform built by the authors, serves as a worked case study: its calibrated relevance gate (threshold -2.0) reached a precision and recall of 0.818 on a 70-pair fixture, and four of five out-of-corpus queries were reduced to zero evidence. The paper closes with open problems: multi-domain calibration, benchmark and human evaluation, the gap between attribution and correctness, latency, and the semantic binding of numbers.

Keywords—retrieval-augmented generation, hallucination, grounding, attribution, multi-agent systems, reranking, explainable AI, provenance, research assistants.

1. Introduction

Students, researchers and analysts rarely finish a task inside a single tool. A literature question may involve a search engine, a document reader, a chat model, a notebook for data analysis and a presentation tool, and every hand-off loses context and leaves no record of how a final claim was produced. Large language model (LLM) assistants promise to collapse this chain, but most are designed to *answer questions* rather than to *complete work*, and their answers are difficult to verify structurally.

The central difficulty is hallucination: fluent text that is unsupported by, or contradicts, its sources [1]. Retrieval-augmented generation (RAG) [2] conditions the model on retrieved passages and reduces the problem, but it does not remove it, because grounding is usually requested in the prompt (“answer only from the context”) rather than enforced by the surrounding software. A prompt instruction is a request, not a guarantee: the model may still cite a document it never saw, and a top-k retriever returns k passages however irrelevant they are. A concrete illustration comes from the platform used as a case study in this paper. The question “What is the capital of Japan?”, asked over a poultry-industry corpus, retrieved five poultry chunks at a cosine similarity of about 0.24. Passing such evidence to a generator invites fabrication.

This paper reviews the literature on making LLM-based research assistants verifiable. Its contributions are:

- a structured survey of the strands of work that bear on the problem: hallucination, retrieval-augmented generation, reranking and score calibration, attribution, agent orchestration, data analysis, provenance and service reliability (Sections 2–9);
- a five-level taxonomy of grounding enforcement that separates prompt-level requests from guarantees implemented in backend control flow, and a comparison of representative tools against it (Section 10);
- a worked case study, the AIKDAP platform developed by the authors, together with the calibration evidence reported for it (Section 11); and
- a consolidated list of open problems (Section 12).

Scope and method.

This is a narrative review, not a systematic one. Sources were chosen for direct relevance to grounding, attribution, orchestration, verifiable analytics, provenance or reliability in LLM-based assistants; for publication in peer-reviewed venues, widely used preprints or maintained technical documentation;

and for their influence on the systems discussed. No formal search protocol was applied, and coverage of the fastest-moving recent work is necessarily partial. Statements about commercial tools reflect their publicly documented behaviour as assessed by the authors and should be re-verified, because vendor features change often.

2. Hallucination: Definitions and Measurement

Transformer-based LLMs [3] generate fluent text, but fluency is not evidence of truth. Surveys of natural-language generation separate *intrinsic* hallucination, which contradicts the source, from *extrinsic* hallucination, which adds content that cannot be verified from the source [1]. A later survey of LLMs extends this into a taxonomy of factuality and faithfulness errors and traces their causes to data, training and decoding [4]. Both kinds matter in research settings, where a fabricated figure or reference can propagate into downstream work.

Several instruments measure the problem. TruthfulQA [5] probes whether models reproduce common falsehoods. SelfCheckGPT [6] detects likely hallucinations without external knowledge by checking consistency across multiple sampled answers. FActScore [7] decomposes long-form text into atomic facts and reports the share supported by a knowledge source.

These instruments estimate how often a model hallucinates, or flag likely errors after the fact. None of them prevents an individual unsupported statement from reaching the user of a deployed assistant. That gap motivates the structural approaches reviewed in the rest of this paper.

3. Retrieval-Augmented Generation

REALM [8] integrated retrieval into language-model pre-training. RAG [2] combined a parametric generator with a dense non-parametric index for knowledge-intensive tasks, and Atlas [9] showed that retrieval-augmented models are competitive in few-shot settings. Dense passage retrieval [10] made embedding-based retrieval the usual first stage. A survey [11] organises the field into naive, advanced and modular RAG, the later forms adding query rewriting, reranking and routing around the retriever.

Three recurring limitations motivate the work reviewed later. First, most pipelines take a fixed top-k, so they cannot express “no relevant evidence exists.” Second, irrelevant or badly placed context can hurt: Liu et al. [12] found that models use information less reliably when it sits in the middle of a long context. Third, in most pipelines grounding stays inside the prompt and the model’s own behaviour. Self-RAG [13] moves part of the decision into the model by training it to retrieve on demand and to critique its output with reflection tokens, which improves behaviour but still leaves the guarantee inside the model.

Reference-free evaluation frameworks such as RAGAs [14] score properties like faithfulness and context relevance, typically with an LLM as judge. They are useful for monitoring but, like the hallucination detectors above, they are probabilistic and run after generation.

4. Retrieval Pipelines, Reranking and Score Calibration

Two-stage retrieval, in which a fast bi-encoder proposes candidates and a more accurate cross-encoder rescores each (query, passage) pair jointly [15], is well established in information retrieval. Late-interaction models sit between the two: ColBERT [16] keeps token-level representations to gain accuracy at moderate cost, and ColBERTv2 [17] adds residual compression to reduce index size. BEIR [18] showed that models that do well in-domain do not always generalise zero-shot, a reminder that any retrieval configuration has to be checked against the target corpus. BGE-M3 [19] provides multilingual, multi-functionality and multi-granularity embeddings and, together with the BGE reranker family, is a common open-source choice.

A cross-encoder emits an unbounded logit, not a probability. A relevance threshold therefore has to be calibrated on labelled data from the deployment instead of assumed. Calibration of neural-network confidence is a well-studied problem [20], but the literature on thresholds for reranker scores in end-user RAG products is thin. Thresholding changes what the system can say: with a gate, the retrieval stage can return *zero* evidence, and the pipeline can route to an explicit “insufficient evidence” outcome instead of generating from noise.

The choice of operating point is a product decision as much as a statistical one, because the two errors have different costs. A false accept feeds irrelevant text to the generator; a false reject turns a question the corpus can answer into a spurious “insufficient evidence.” An F-measure that weights precision heavily can therefore select a threshold that gives up too much recall. Section 11 reports a case where such a threshold was rejected.

5. Attribution and Citation Verification

Attributed question answering [21] proposed evaluating whether answers are supported by identified sources. ALCE [22] built a benchmark for generating text with citations, measuring fluency, correctness and citation quality, and found that strong models still fall short of complete citation support. A human evaluation of generative search engines [23] reported that only about half of the generated sentences were fully supported by their citations, and that a substantial share of citations did not support the sentences they were attached to.

Most attribution checking is post hoc and model-based: another model, often an entailment classifier or an LLM judge, decides whether a cited passage supports a statement. This is flexible but probabilistic, and it adds cost and latency. A complementary and much weaker, but fully deterministic, check treats citation validity as *set membership*: a citation is valid only if its identifier belongs to the set of evidence actually placed in the prompt, and invalid

identifiers are dropped and recorded rather than repaired. This guarantees that every stored citation points to real, supplied evidence. It does not prove that the evidence entails the claim. The two checks address different failure modes and are best seen as layers.

6. LLM Agents and Orchestration

ReAct [24] interleaves reasoning with tool actions, Toolformer [25] trains a model to decide when to call external APIs, and Reflexion [26] uses verbal feedback across attempts. Surveys [27] catalogue planning, memory and tool-use designs. In multi-agent frameworks, AutoGen [28] lets agents converse to solve tasks, and MetaGPT [29] assigns roles under standardised operating procedures.

Conversational coordination is flexible, but the resulting transcripts are hard to audit and to reproduce. LangGraph [30] takes the opposite approach and models a workflow as an explicit state graph with typed state, conditional edges and reducers. The trade-off is flexibility for determinism. For trust-critical products the second is often preferable, because the topology can be inspected and every transition logged. Principles for auditable orchestration, as implemented in the case-study platform of Section 11, include:

- the planner decides and the graph orchestrates; routing functions, not agents, choose the next node;
- agents never call each other; each node returns a partial update to a typed shared state;
- accumulating channels (documents, messages) use reducers, and all other keys are last-write-wins;
- compiled graphs are stateless, with database sessions and strategies injected per invocation, so one graph can serve concurrent runs;
- every node is wrapped by common instrumentation that times it and applies a declared failure policy; and
- when a non-critical node fails, the deliverable must state that the source was unavailable.

7. Natural-Language Data Analysis

Text-to-SQL research, benchmarked on datasets such as Spider [31], and program-aided reasoning, in which a model writes code that an interpreter runs [32], move arithmetic out of the model and so improve numeric reliability. Letting a model generate executable code nevertheless raises a safety question, since the code runs with the host's privileges, and a verification question, since a reader cannot easily confirm that the code does what the narrative claims.

A more conservative design lets the model see only the dataset schema and emit a schema-validated *analysis plan* (filters, group keys, aggregations, time periods). Each plan field maps to one explicit pandas operation, with no eval and no strings converted to code. The model then narrates only the result rows, and a validator extracts every number from the narrative, including k/M/B and percentage forms, and matches it against the result table within a relative tolerance (1% in the case study). Unmatched numbers are flagged to the user. The limitation is that such a check confirms that a stated value *appears* in the results, not that it is attached to the right quantity.

8. Explainability and Provenance for Agentic Systems

Research on explainable AI [33] concentrates mostly on explaining individual model predictions. For agentic systems, a more practical explanation is *process provenance*: what was planned, which agent ran, with what inputs, and how long it took. The W3C PROV data model [34] offers a general vocabulary of entities, activities and agents for such records.

A key design decision is whether the explanation is generated or recorded. A narrative that the model writes about its own process is another generation and can itself be wrong. A trace written to database tables by the orchestration code is evidence. In the case-study platform, run records, per-node step records and the inter-agent message transcript are persisted. Grounding status is a first-class column, so “every run that could not be answered” is a plain query, and a reconciliation job marks runs stale after 1,200 s so that a crashed worker never leaves a run pending forever. The user interface renders the same data as a workflow timeline.

9. Reliability Engineering for LLM Services

Assistants that depend on external model providers inherit their failure modes. The circuit-breaker pattern from general distributed-systems practice [35] applies directly, yet academic literature on LLM gateway behaviour is thin and most guidance is engineering documentation. One incident from the case study shows why error classification matters: a free-tier provider returned a rate-limit response carrying a retry delay of 35 s on a quota that reset daily, so any retry loop that honoured the hint could never succeed. Table 1 summarises the resulting policy.

TABLE 1. LLM gateway error classes and behaviour (case-study platform)

Error class	Behaviour	Cooldown
Per-minute rate limit	Retry with jittered exponential backoff (1–8 s, at most 2 retries)	60 s
Day-scale quota	No retry; go straight to the fallback	3600 s
Balance exhausted	No retry; use the fallback	—
Terminal (bad credentials or request)	Stop the whole chain	300 s

Requests pass through a fallback chain (for example two keys of one provider, then two other providers). Provider health is stored in a shared store (Redis) so that the API process and the workers share one breaker, and every response records the number of attempts, whether a fallback was used and which model actually answered. Degradation is therefore reported rather than hidden.

10. A Taxonomy of Grounding Enforcement

Synthesising the strands above, we distinguish five levels according to *where* grounding is enforced. The levels are cumulative and descriptive; they are a reading aid proposed in this review, not an empirical scale of quality.

TABLE 2. Five levels of grounding enforcement

Level	What is enforced, and where	Typical residual failure	Representative work
L0	Nothing: the model is asked to be accurate (prompt only)	Fluent, unsupported claims	Plain LLM generation [3]
L1	Retrieval plus a prompt instruction to use only the context; fixed top-k	Irrelevant evidence is always supplied; the model may assert or cite beyond it	Baseline RAG [2, 11]
L2	L1 plus reranking and a calibrated relevance gate in code	Threshold is corpus-specific; some recall is lost	Two-stage retrieval [15, 20]
L3	L2 plus deterministic citation validation (stored citations are a subset of supplied evidence)	Valid citation does not prove entailment	Attribution checking [22]
L4	L3 plus backend-computed grounding status and a persisted, queryable execution trace	Attribution is not correctness; added infrastructure	Provenance [34]; AIKDAP (Section 11)

Table 3 compares widely used research assistants on seven criteria derived from the taxonomy. The entries are the authors' assessment of publicly documented behaviour at the time of writing the project report and are not the result of independent testing.

TABLE 3. Comparison with existing tools (assessment of publicly documented behaviour)

Criterion	ChatGPT	Perplexity	NotebookLM	Elicit / SciSpace	AIKDAP
Grounded in user documents	partial	no	yes	yes	yes
Citation validity enforced in code	no	no	no	no	yes
Calibrated zero-evidence outcome	rarely	no	rarely	rarely	yes

Criterion	ChatGPT	Perplexity	NotebookLM	Elicit / SciSpace	AIKDAP
Provenance to chunk and raw scores	no	link	passage	passage	yes
Persisted execution trace	no	no	no	no	yes
Verified dataset analytics	code interpreter	no	no	no	yes
Unit of memory	chat	query	notebook	query	project

On these criteria, none of the other tools is documented as enforcing citation validity in code or as keeping an execution trace that the user can query. In this assessment no existing tool combines project-scoped memory, multi-agent completion of whole workflows and grounding enforced structurally rather than by prompt. The platform described next was designed to occupy that gap at level L4.

11. Case Study: The AIKDAP Platform

11.1 Architecture

AIKDAP (AI Knowledge Discovery and Analytics Platform) is an AI work operating system organised around *projects*. A user gives one task; intent analysis and a planner agent break it down; a LangGraph orchestrator distributes it to specialised agents; and a validation service checks the result before an explainable, traceable deliverable is returned. The backend is built with FastAPI and organised into 14 feature modules, with Celery workers on a Redis broker for ingestion, research, analytics and report tasks. Four LangGraph orchestrators implement the research, business-analytics, report and build-plan workflows. All model calls pass through a single LLM gateway (Section 9). Persistence uses PostgreSQL 17 with pgvector, Redis and ChromaDB, and the interface is a React and TypeScript single-page application. Memory is per project, and every generated artifact becomes a persistent asset.

Understanding (Qwen 3.5 4B), embedding (BGE-M3) and reranking (BGE-Reranker-v2-m3) run locally, while synthesis can use OpenAI-compatible cloud providers behind the fallback chain. Document ingestion is deterministic: no model recovers text. A PDF is sent to Tesseract OCR [36] when more than half of its pages are empty, and chunking is done per page, sheet or slide so that a chunk never crosses a page boundary (size 1000, overlap 100).

11.2 The grounding chain

Research queries pass through four stages. Stage 0 reformulates the query with a schema-validated LLM call. Stage 1 retrieves up to 20 ownership-filtered candidates with BGE-M3 and pgvector. Stage 2 reranks them with a cross-encoder and keeps the top 5. Stage 3 applies the relevance gate at a threshold of -2.0 , which may leave zero evidence. Retrieval and rerank scores are stored side by side on their native scales and are never fused or normalised.

Synthesis enforces three guarantees in code. Only retrieved evidence is sent, and the model receives no tools, database access or history. Only supplied evidence can be cited: every returned citation identifier is checked against those placed in the prompt, and unknown identifiers are dropped and recorded. The model does not grade itself: the grounding status is computed from the validated citation set. When evidence is insufficient, the graph may try web research or query OpenAlex [37] for suggested papers that the user can add to the project before re-running the question. The analytics module applies the same principle to numbers, as described in Section 7.

11.3 Calibration evidence

The gate was calibrated on the real stack, with nothing mocked. A corpus of 7 chunks and 10 queries (5 in-corpus and 5 out-of-corpus) produced 70 (query, chunk) pairs, of which 11 were relevant and 59 irrelevant. Labels were assigned from document content and never from the score being measured. Relevant pairs had a mean reranker score of $+2.296$ (median $+3.262$) and irrelevant pairs a mean of -8.156 (median -10.072), but the distributions overlap: the lowest relevant score was -10.759 and the highest irrelevant score $+2.133$. Table 4 and Fig. 1 show the threshold sweep.

TABLE 4. Relevance-gate threshold sweep on the 70-pair calibration fixture

Threshold	TP	FN	FP	TN	Precision	Recall	FPR
-4.0	9	2	12	47	0.429	0.818	0.203
-3.0	9	2	8	51	0.529	0.818	0.136

Threshold	TP	FN	FP	TN	Precision	Recall	FPR
-2.0	9	2	2	57	0.818	0.818	0.034
-1.0	8	3	2	57	0.800	0.727	0.034
0.0	6	5	2	57	0.750	0.545	0.034
+2.5	6	5	0	59	1.000	0.545	0.017

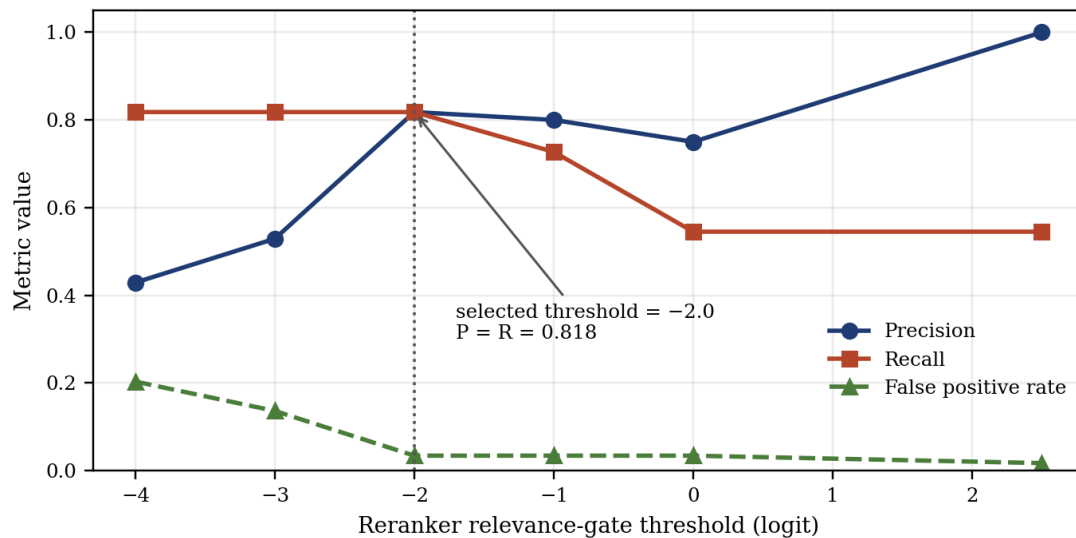


Fig. 1. Precision, recall and false positive rate against the gate threshold (data from Table 4).

At -2.0, precision rises from 0.529 (at -3.0) to 0.818 while recall stays at its maximum of 0.818. All five in-corpus queries keep at least one genuinely relevant chunk with no false accepts, and four of the five out-of-corpus queries fall to zero evidence, so no synthesis call is made for them. The threshold that maximises F0.5 (+2.25, precision 1.0, recall 0.545) was rejected because it would convert a question the corpus does answer into a false “insufficient evidence.” For a research assistant, answering real questions was judged more important than an aggregate metric.

Further tests inject model outputs that cite identifiers which were never supplied. In every case the identifier is dropped and recorded, and a response that claims grounding without valid citations is stored as insufficient evidence. For analytics, a narrative value that is absent from the result table is flagged, while values within 1% relative tolerance (for example “1.23M” against 1,234,567.89) are accepted.

11.4 Implementation scale and deployment observations

The implementation comprises about 31,300 lines of backend Python, about 1,160 automated backend test functions, about 65 REST endpoints under /api/v1, 14 feature modules, four LangGraph orchestrators and about 27,400 lines of frontend code across 185 TypeScript/TSX files. In a single deployment with six projects and twelve documents, a physics question asked inside the Machine Learning project was marked *Un sourced* and took 11.8 s, whereas two grounded runs carried 10 and 4 citations and took 31.0 s and 29.0 s. The interface labels the unsourced reply as general knowledge and not as a finding from the user’s documents. These are single observations from one deployment and not benchmarks.

11.5 What the case study does and does not show

The case study shows that the mechanisms of levels L2 to L4 can be built and operated together, and that a calibrated gate can separate in-corpus from out-of-corpus queries on a small fixture. It does not show superiority over a baseline RAG system, answer correctness, or acceptable latency at scale. The threshold was chosen and evaluated on the same 70 pairs, so the reported precision and recall are calibration figures and not out-of-sample estimates; no inter-annotator agreement study was run for the relevance labels; and the fixture covers one domain. The threshold of -2.0 should not be assumed to transfer to other corpora, models or hardware.

12. Open Challenges and Research Directions

Table 5 consolidates the open problems that emerge from the review and the case study.

TABLE 5. Open challenges and directions

Challenge	Current state	Direction
Calibration at scale	Threshold calibrated on 70 pairs from one domain and tied to the deployment	Multi-domain fixtures with thousands of labelled pairs; automatic recalibration when the corpus changes
Benchmark and human evaluation	No head-to-head comparison with a baseline RAG system and no human evaluation	Evaluate on attribution benchmarks such as ALCE [22] against baselines, with human raters
Attribution versus correctness	Set membership and grounding status measure attribution, not truth	Add entailment-level verification as a second layer
Latency and throughput	Retrieval agents run sequentially; latency not benchmarked	Parallel fan-out using the existing reducers; load testing
Semantic binding of numbers	The check shows a number appears in the results, not what it describes	Bind narrated numbers to named columns or quantities
Safe experiment execution	Experiment plans are derived but never executed	Sandboxed execution behind an explicit review step
Privacy	Cloud providers receive synthesis prompts that contain the selected evidence	Local synthesis options and prompt redaction

Two of these deserve emphasis. The first is evaluation: structural guarantees are easy to state and test, but their value to users can only be established against baselines and with human judgement. The second is the distinction between attribution and correctness. A system that never cites unsupplied evidence is more trustworthy than one that does, yet it can still faithfully repeat an error present in its own corpus. Combining deterministic structural checks with model-based entailment checks, and recording both in the execution trace, appears to be a promising direction.

13. Conclusion

Hallucination detection, RAG, reranking, attribution benchmarks, agent frameworks and provenance models have each advanced considerably, but they are rarely combined into assistants whose trustworthiness is a property of the software and not of the prompt. This review has organised that literature, proposed a five-level taxonomy of grounding enforcement and compared representative tools against it. The AIKDAP case study illustrates level L4 in practice: a calibrated relevance gate, deterministic citation validation, a backend-computed grounding status, verified analytics numbers and a persisted execution trace. Its evidence is small-scale and preliminary, and the open problems of Table 5, above all multi-domain calibration and rigorous evaluation, define the work needed before such systems can be considered validated rather than merely demonstrated.

References

- [1] Z. Ji et al., "Survey of hallucination in natural language generation," *ACM Computing Surveys*, vol. 55, no. 12, 2023.
- [2] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [3] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [4] L. Huang et al., "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," arXiv:2311.05232, 2023.
- [5] S. Lin, J. Hilton and O. Evans, "TruthfulQA: Measuring how models mimic human falsehoods," in *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022.
- [6] P. Manakul, A. Liusie and M. J. F. Gales, "SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models," in *Proc. EMNLP*, 2023.
- [7] S. Min et al., "FactScore: Fine-grained atomic evaluation of factual precision in long form text generation," in *Proc. EMNLP*, 2023.

-
- [8] K. Guu et al., "REALM: Retrieval-augmented language model pre-training," in *Proc. International Conference on Machine Learning (ICML)*, 2020.
- [9] G. Izacard et al., "Atlas: Few-shot learning with retrieval augmented language models," *Journal of Machine Learning Research*, vol. 24, 2023.
- [10] V. Karpukhin et al., "Dense passage retrieval for open-domain question answering," in *Proc. EMNLP*, 2020.
- [11] Y. Gao et al., "Retrieval-augmented generation for large language models: A survey," arXiv:2312.10997, 2023.
- [12] N. F. Liu et al., "Lost in the middle: How language models use long contexts," *Transactions of the Association for Computational Linguistics*, vol. 12, 2024.
- [13] A. Asai et al., "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," in *Proc. International Conference on Learning Representations (ICLR)*, 2024.
- [14] S. Es et al., "RAGAs: Automated evaluation of retrieval augmented generation," in *Proc. EACL (System Demonstrations)*, 2024.
- [15] R. Nogueira and K. Cho, "Passage re-ranking with BERT," arXiv:1901.04085, 2019.
- [16] O. Khattab and M. Zaharia, "ColBERT: Efficient and effective passage search via contextualized late interaction over BERT," in *Proc. ACM SIGIR*, 2020.
- [17] K. Santhanam et al., "ColBERTv2: Effective and efficient retrieval via lightweight late interaction," in *Proc. NAACL*, 2022.
- [18] N. Thakur et al., "BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models," in *NeurIPS Datasets and Benchmarks Track*, 2021.
- [19] J. Chen et al., "BGE M3-Embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," arXiv:2402.03216, 2024.
- [20] C. Guo, G. Pleiss, Y. Sun and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. ICML*, 2017.