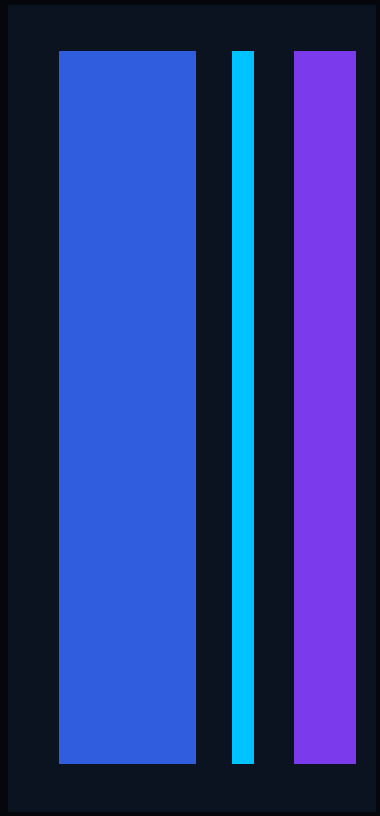
KDD Lecture Tutorial

Retrieval-Augmented Generation (RAG) — From Modular to Agentic Systems

Xin Luna Dong, Sanat Sharma, Kai Sun, Jiaqi Wang, Yinglong Xia, Xiao Yang, Scott
Wen-tau Yih @ Meta
Franklin Zhang @ UW CSE

KDD Tutorial, 8/2026



This talk does not represent the company's point of view

01-01

Introduction (10 min)

How knowledgeable and factual are LLMs?
What is hallucination and where does it come from?
What is the ultimate goal of information providing?

What is a Large Language Model (LLM)?

- A neural network trained on massive text data (web, books, code)
  - Examples: GPT-4, Claude, Gemini, LLaMA
- Core mechanism: predict the next token ("word")
  - Knowledge is stored implicitly in billions of parameters
  - Surprisingly capable: translation, QA, reasoning, coding

📖 Next-Token Prediction

Given tokens $x_1, x_2, \ldots, x_{t-1}$, predict x_t for every position in the sequence.

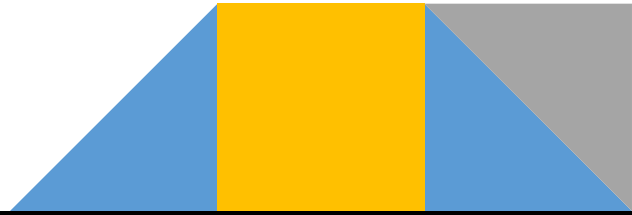
$$\mathcal{L} = -\sum_{t} \log P(x_t \mid x_1, x_2, \ldots, x_{t-1})$$

🔑 Key Point

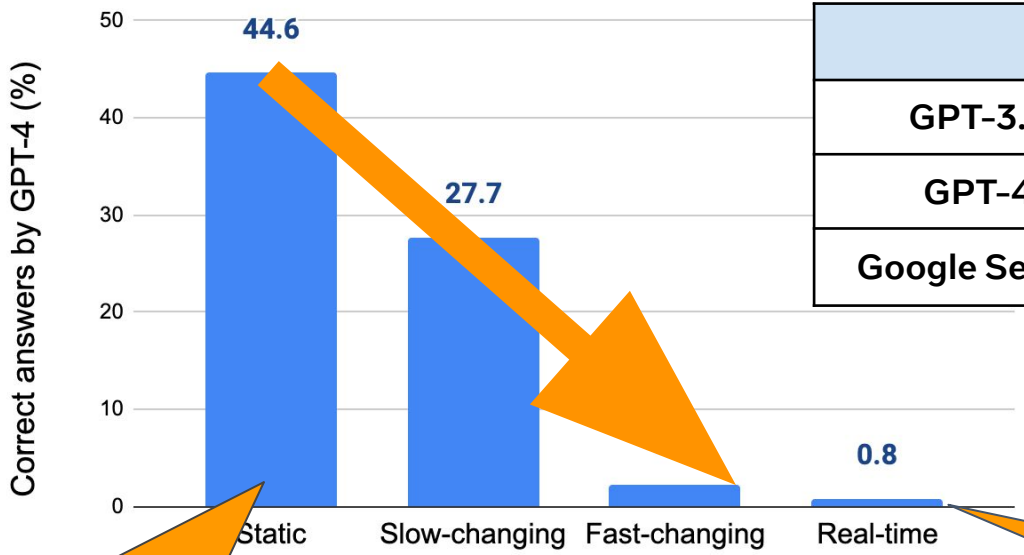
LLMs learn **statistical patterns** from text — they do *NOT* "understand" *truth*. They're trained to produce plausible text, not necessarily *accurate text*.

How Knowledgeable Are LLMs?

- How ***reliable*** are LLMs in answering factual questions?
- Do LLMs perform ***equally well*** across different types of factual knowledge?
- What are ***key factors*** that affect LLM factuality?



LLM QA for Facts of Different Dynamisms



	Never	Slow	Fast
GPT-3.5	59%	15%	4%
GPT-4	64%	4%	12%
Google Search	**68%**	**46%**	**32%**

Low accuracy even for static facts

Very low accuracy for fast changing facts

Vu, Iyyer, Wang, Constant, Wei, Wei, Tar, Sung, Zhou, Le, Luong. FreshLLMs: Refreshing LLMs with Search Engine Augmentation. arXiv, 2023.
Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, et al. CRAG–Comprehensive RAG Benchmark. arXiv, 2024.

Simple Static Questions

Answer the following questions in as few words as possible. Say "unsure" if you don't know.

Question: What is the capital of China?
Answer: Beijing

Question: What is the captical of Wernythedia?
Answer: unsure

Question: {QUESTION}
Answer:

Head-to-Tail Benchmark

	IMDb		Goodreads	DBpedia
	Title	Person	Book	-
Head	767 (0.01)	34,903 (0.48)	3,150 (2.31)	103,564 (1.30)
Torso	4,113 (0.05)	87,645 (1.21)	7,304 (5.35)	1,255,113 (15.77)
Tail	7,536,482 (99.94)	7,111,496 (98.31)	126,134 (92.35)	6,600,206 (82.93)

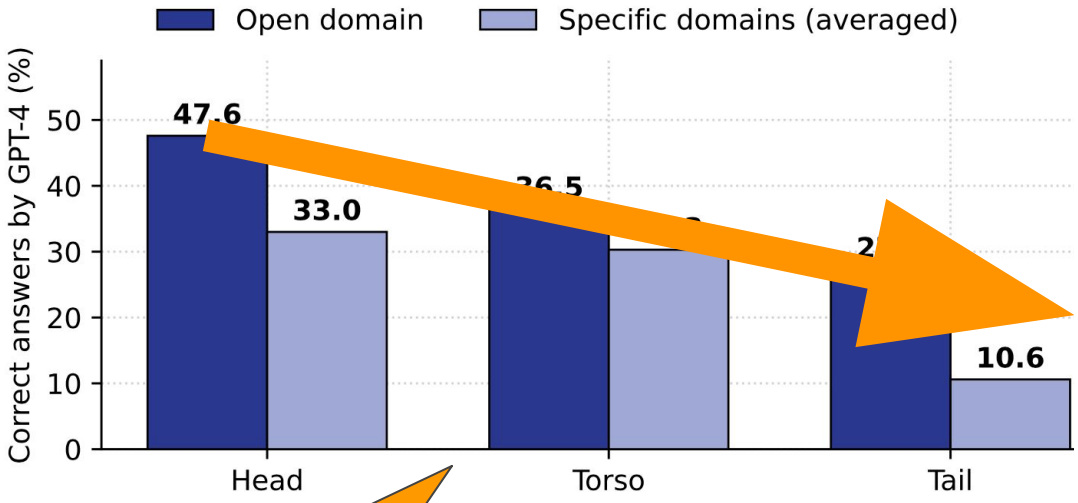
	MAG			DBLP
	Article	Conference	Journal	Scholar
	1,827,710 (0.70)	257 (1.63)	225 (0.46)	79,521 (2.44)
	9,386,034 (3.60)	965 (6.12)	1,266 (2.58)	500,778 (15.36)
	249,311,539 (95.70)	14,550 (92.25)	47,546 (96.96)	2,680,704 (82.20)

When counted by popularity, majority of entities are long-tail

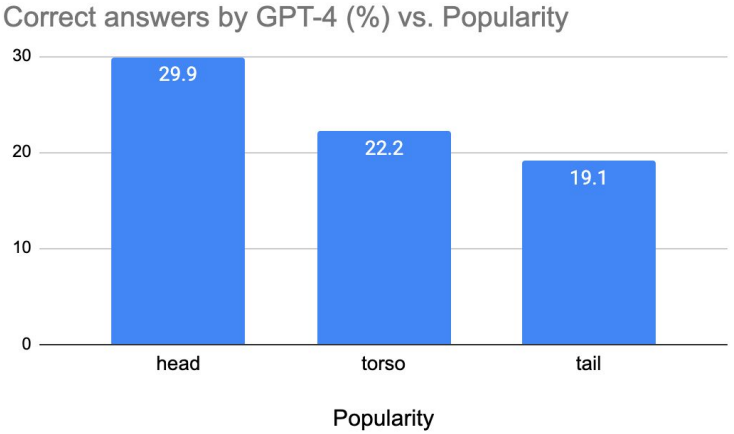
Sun et al. Head-to-Tail: How Knowledgeable are Large Language Models (LLM)?A.K.A. Will LLMs Replace Knowledge Graphs? NAACL, 2024.

LLM QA for Entities of Different Popularities

Simple questions



General questions



Low accuracy even for popular entities

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, et al. CRAG–Comprehensive RAG Benchmark. arXiv, 2024.

Sun et al. Head-to-Tail: How Knowledgeable are Large Language Models (LLM)?A.K.A. Will LLMs Replace Knowledge Graphs? NAACL, 2024.

What Is Hallucination?

AI-generated content that is ***factually incorrect, fabricated, or not grounded in its input,*** but presented with ***full confidence***.

❌ Fabricated Fact

"The Eiffel Tower was designed by Gustave Eiffel in 1823."

Wrong year (1889), presented confidently

❌ Invented Citation

"According to Smith et al. (2023) in Nature..."

Paper doesn't exist — entirely fabricated

❌ Plausible Nonsense

"A 2019 Stanford study found that writing down your goals makes you 42% more likely to achieve them.

Sounds right but whole story is made up

Intrinsic vs. Extrinsic Hallucination

Extrinsic Hallucination

Output cannot be verified from input — may or may not be true

Example:

Source: bio of a scientist
Model adds: "She also enjoyed painting"

→ Not in source. Could be true or false.

Also called: factual fabrication
Related concept: factuality

Intrinsic Hallucination

Output contradicts the provided input/source

Example:

Source: "The company reported $5M revenue"
Model output: "Revenue reached $8M"

→ Directly contradicts the given source

Also called: unfaithfulness
Related concept: faithfulness

Knowledge Conflicts: Parametric vs. Contextual

What happens when the model's internal knowledge disagrees with retrieved evidence?

🧠 Parametric Knowledge

What the model learned during training:

"The capital of Myanmar is Yangon"

(outdated — was true before 2006)

📄 Contextual Knowledge

What the retrieved document says:

"The capital was moved to Naypyidaw in 2006"

⚡ The Conflict

Model must decide which to trust.

Larger models are paradoxically **MORE likely to override correct context** with wrong parametric knowledge.

(ContextFocus, 2026) (Yigit-Sert, 2026)

Why Hallucination Matters?

🏥 Healthcare

AI suggests wrong drug interactions or fabricates medical studies

⚖️ Legal

ChatGPT invented 6 fake legal cases cited in a real court filing (2023)

💰 Finance

Incorrect financial data or fabricated company reports affect decisions

🎓 Education

Students receive plausible but wrong explanations of concepts

📊 The Scale of the Problem

- SimpleQA factuality benchmark: even for frontier models (GPT-4, Claude) hallucination rate could go up to 60%
- CRAG benchmark: the best industry SOTA RAG system still had 18% hallucination rate
- Hallucination rates increase for rare/long-tail knowledge — exactly where help is needed most

(SimpleQA, 2024) (CRAG, 2024)

Root Causes of Hallucinations

Training Data Issues

- Errors, biases, and contradictions in training data
- Long-tail facts underrepresented
- Positional biases from auto-regressive training

Decoding & Sampling

- Random sampling (temperature > 0) introduces randomness
- Top-k/top-p can select wrong tokens
- Beam search can amplify low-probability errors

Architecture Limitations

- Finite context window limits evidence
- Attention mechanism can miss relevant info
- Knowledge stored distributedly, hard to pinpoint

Alignment Tax

- Fine-tuning for helpfulness can degrade factuality
- RLHF optimizes for human preference, not truth
- Models learn to be confidently wrong

Ultimate Goals for Information Providing

Factuality

Is the output true about the world?

"Paris is in France" ✅
"Paris is in Germany" ❌

Scope: World knowledge

Faithfulness

Is the output consistent with the given source?

Source says revenue = $5M
Output says $5M ✅ / $8M ❌

Scope: Source grounding

Truthfulness

Does the model express only what it knows?

"I'm not sure about this" ✅
vs. confident fabrication ❌

Scope: Honesty & calibration

RAG & Factuality

Two complementary approaches to the same problem: making LLMs factually reliable

🎯 The Problem:
LLMs hallucinate confidently

🟠 Factuality: Internal Improvement

- Improve knowledge internalization at training time
- Detect hallucinations via internal states
- Calibrate confidence & know what you don't know
- Verify outputs against decomposed claims

Closed-book QA—Less Successful

🔵 RAG: External Grounding

- Retrieve relevant evidence at inference time
- Ground generation in external sources
- Enable source attribution & traceability
- Keep knowledge up-to-date without retraining

Open-book QA—Dominant Solution

01-02

A Brief Introduction to Factuality Landscape (15 min)

How can we internalize knowledgeable into LLMs?
How can we suppress hallucinations in generation?
Do we really need RAG or can we just improve closed-book QA?

How Knowledge Gets Into LLMs?

During pre-training, LLMs compress vast amounts of text into their parameters. But this process is far from perfect.

📚 Massive Data

Trillions of tokens from web, books, code, Wikipedia, etc.

🔄 Next-Token Prediction

Model learns patterns by predicting what comes next

$$\mathcal{L} = -\sum_{t} \log P(x_t \mid x_1, x_2, \ldots, x_{t-1})$$

💾 Implicit Storage

Knowledge is distributed across billions of parameters

❓ Imperfect Recall

Can access some facts but not others — even if seen in training

The Frequency Effect: What Gets Learned?

Facts that appear frequently in training data are learned well.
Facts that appear rarely need exponentially more parameters to memorize.

Head (Common)

"Obama was the 44th US president"

Appears in millions of docs
→ Learned reliably ✅

Torso (Medium)

"Canberra is the capital of Australia"

Appears in thousands of docs
→ Sometimes recalled ⚠️

Tail (Rare)

"Guðjón Samúelsson designed Hallgrímskirkja church"

Appears in a handful of docs
→ Often hallucinated ❌

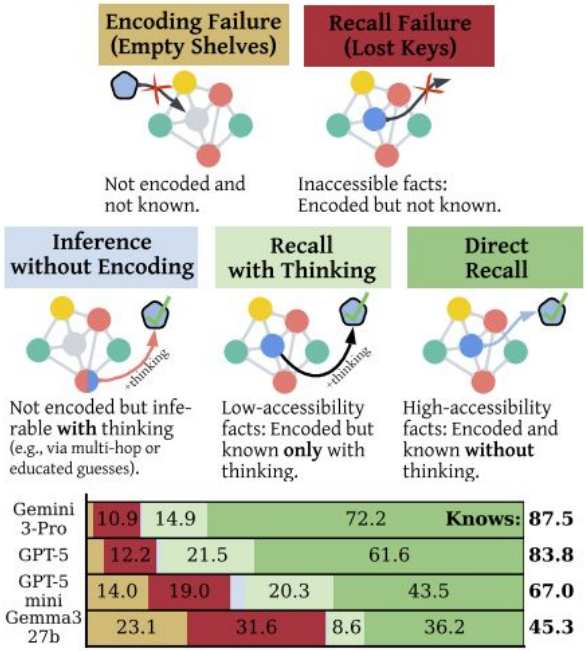
Empty Shelves or Lost Keys? —The Storage vs. Recall Gap

The 'Lost Keys' Problem

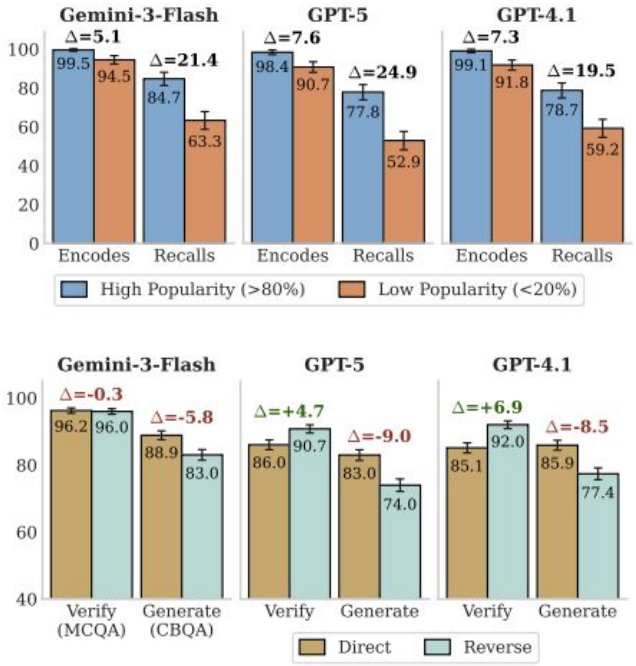
Encoding ≠ Accessibility

Chain-of-Thought Helps

Where Does QA Fail?



Where Does Recall Fail?



Key discovery: The problem isn't that LLMs lack knowledge — it's that they can't access what they already have. *(On Wiki-based benchmark)*

Calderon, Ben-David, Gekhman, Ofek, Yona. Empty Shelves or Lost Keys? Recall Is the Bottleneck for Parametric Factuality. arXiv, 2026.

Two Pillars of Factuality Research

Knowledge Encoding

🧠 Knowledge Internalization

Acquiring and storing factual knowledge at ***training*** time

- Pre-training data curation
- Post-training alignment for knowledge enrichment
- Knowledge editing (update specific facts)
- Understanding internal knowledge mechanisms

Knowledge Recall

🛡️ Hallucination Suppression

Detecting and reducing incorrect outputs at ***inference*** time

- Internal state probing & activation steering
- Contrastive decoding strategies
- Verification pipelines
  (decompose → check → aggregate)
- Confidence-based uncertainty quantification

Two Pillars of Factuality Research

Knowledge Encoding

🧠 Knowledge Internalization

Acquiring and storing factual knowledge at ***training*** time

- Pre-training data curation
- Post-training alignment for knowledge enrichment
- Knowledge editing (update specific facts)
- Understanding internal knowledge mechanisms

Knowledge Recall

🛡️ Hallucination Suppression

Detecting and reducing incorrect outputs at ***inference*** time

- Internal state probing & activation steering
- Contrastive decoding strategies
- Verification pipelines
  (decompose → check → aggregate)
- Confidence-based uncertainty quantification

Knowledge Internalization I. Knowledge Update

LLMs are trained on static snapshots. When facts change, retraining costs millions. Can we surgically edit individual facts instead?

📅 Before 2021

"Who is the Chancellor of Germany?"

→ "Angela Merkel"

Correct at training time

🔄 Reality Changed

Olaf Scholz became Chancellor in Dec 2021

Model's answer is now wrong

🎯 Goal: Surgical Edit

Update this ONE fact without retraining the entire model

- Locates the specific FFN layer storing a fact
- Applies a rank-one weight update: $W + \Delta W$
- Overwrites a single factual association

Examples: ROME (2023), MEMIT (2023)

Preserve all other knowledge (ROME, 2023) (MEMIT, 2023)

MOSE: Stable Editing After Thousands of Updates

🏆 **MOSE: Multiplicative Orthogonal Sequential Editing (2026)**

Adding a correction matrix
($W' = W + \Delta W$)

⇒ Rotating the weight matrix using an orthogonal transformation ($W' = R * W$)

Preserving numerical stability even after 4,000+ sequential edits.
+12% improvement over additive baselines.

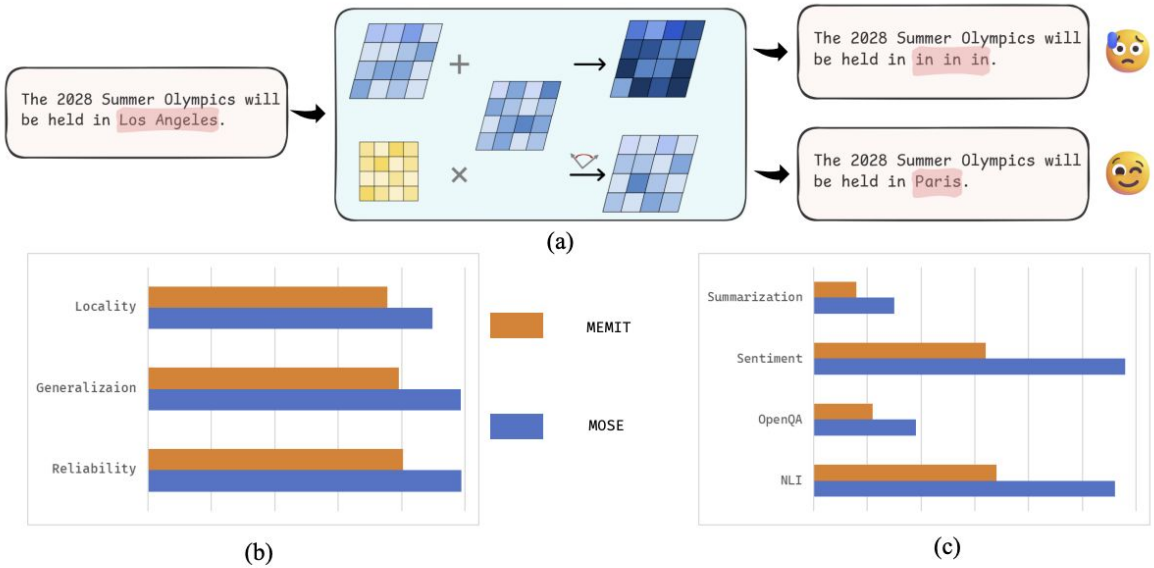


Figure 1: (a) Comparison between the previous methods and MOSE. The previous methods performed updates by adding an update matrix, whereas the MOSE employs left-multiplication by an orthogonal update matrix. (b) Comparison of editing performance after additive-based editing and after multiplicative-based editing by MOSE. (c) Comparison of general downstream task performance before editing, after additive-based editing, and after multiplicative-based editing by MOSE.

H.X. Xu, J.Y. Ma, Z. Peng, Y. Sun, Z.H. Ling, J.C. Gu. Multiplicative Orthogonal Sequential Editing for Language Models. arXiv, 2026.

MOSE: Stable Editing After Thousands of Updates

🏆 MOSE: Multiplicative Orthogonal Sequential Editing (2026)

Instead of adding a correction matrix (W’ = W + ΔW), MOSE rotates the weight matrix using an orthogonal transformation (W' = R * W) — preserving numerical stability even after 4,000+ sequential edits. +12% improvement over additive baselines.

(MOSE, 2026)

🔧 When to Edit

- Small number of high-value corrections
- Facts that rarely change once corrected
- No external retrieval infrastructure available
- Latency-critical applications (no retrieval overhead)

🔍 When to Use RAG Instead

- Rapidly changing knowledge (news, stocks, etc.)
- Thousands of facts need updating
- Need for source attribution and traceability
- When interconnected facts must stay consistent

Knowledge Update: The Cascading Failures Problem

Knowledge editing looks promising in isolation — but breaks down in realistic conditions.

🔗 Logical Consistency

Editing 'Germany's Chancellor = Scholz' doesn't automatically update 'What party does the Chancellor belong to?' ROME achieves >90% on target but <30% on implied facts.

↩️ Reversal Curse

Teaching 'A is B' doesn't teach 'B is A.' If you edit 'The Chancellor is Scholz,' asking 'What role does Scholz hold?' may still fail.

📉 Sequential Degradation

Applying many edits over time causes additive errors. Accuracy and general capabilities degrade. After 100+ edits, models become unreliable.

🗣️ Context Fragility

Edited knowledge reverts when the model encounters conversational context related to the original fact. Edits work in isolation but fail in dialogue.

Knowledge Internalization II: Post-training Alignment

Standard alignment (SFT + RLHF) optimizes for helpfulness — but this can actively degrade factuality by forcing models to generate facts beyond their knowledge boundaries.

😊 Helpfulness Pressure

RLHF rewards detailed, confident responses.
Models learn to fabricate rather than say "I don't know."

📉 Factuality Degradation

Fine-tuning on data beyond the model's knowledge boundary creates hallucinations.
SFT can teach models to confidently lie.

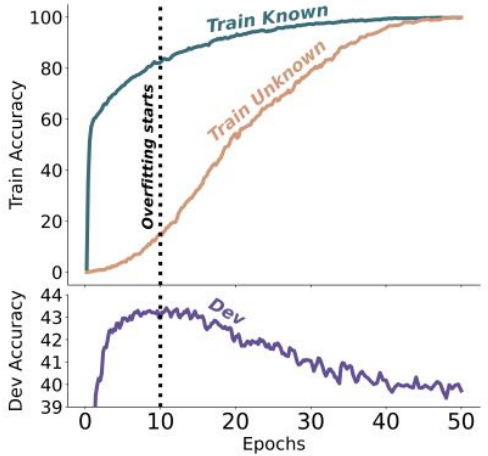


Figure 1: Train and development accuracies as a function of the fine-tuning duration, when fine-tuning on 50% Known and 50% Unknown examples. Unknown examples are fitted substantially slower than Known. The best development performance is obtained when the LLM fits the majority of the Known training examples but only few of the Unknown ones. From this point, fitting Unknown examples reduces the performance.

The Alignment Tax: When RLHF Hurts Factuality

Gekhman et al. Does Fine-Tuning LLMs on New Knowledge Encourage Hallucinations? arXiv, 2024.

Two Pillars of Factuality Research

Knowledge Encoding

🧠 Knowledge Internalization

Acquiring and storing factual knowledge at ***training*** time

- Pre-training data curation
- Post-training alignment for knowledge enrichment
- Knowledge editing (update specific facts)
- Understanding internal knowledge mechanisms

Knowledge Recall

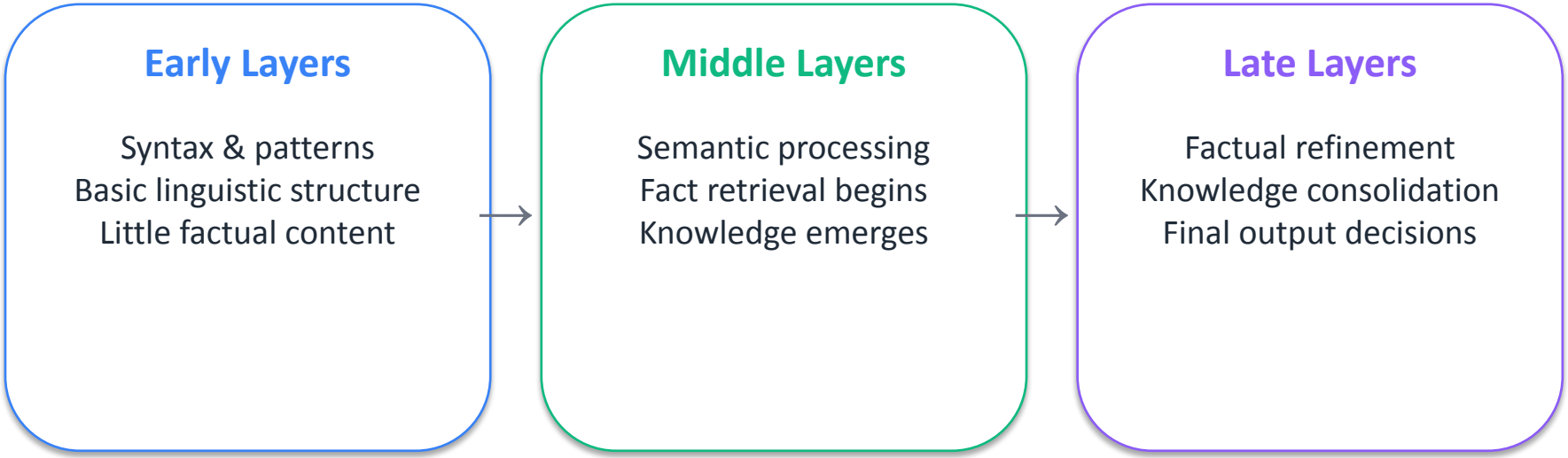
🛡️ Hallucination Suppression

Detecting and reducing incorrect outputs at ***inference*** time

- Internal state probing & activation steering
- Contrastive decoding strategies
- Verification pipelines
  (decompose → check → aggregate)
- Confidence-based uncertainty quantification

Hallucination Suppression I. Internal Parameter Intervention

Key idea: LLMs often 'know' correct answers internally but fail to express them. By tapping into internal signals, we can detect and correct errors without external knowledge or expensive fine-tuning.



💡 Key insight: Factual knowledge emerges progressively across layers. The contrast between layers reveals what the model truly knows.

Probing Internal States

🏆 DoLa: Decoding by Contrasting Layers (2023)

Subtract early-layer token probabilities from final-layer probabilities. This cancels non-factual noise (linguistic patterns) and amplifies factual knowledge that emerges across layers. Training-free — no fine-tuning needed.

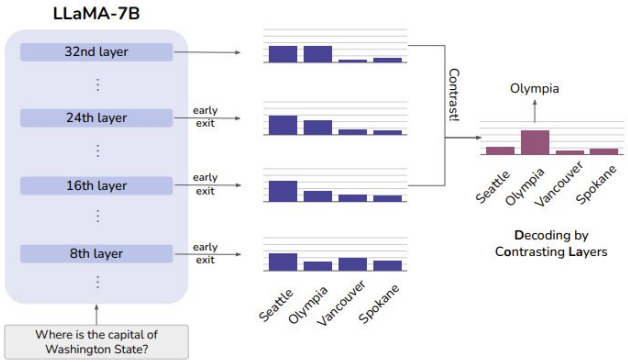


Figure 1: Illustration of an LLM progressively incorporates factual information along layers. While the next-word probabilities of "*Seattle*" remain similar throughout different layers, the probabilities of the correct answer "*Olympia*" gradually increase from lower to higher layers. DoLa uses this fact to decode by contrasting the difference between layers to sharpen an LLM's probability towards factually correct outputs.

TruthfulQA

+12-17%

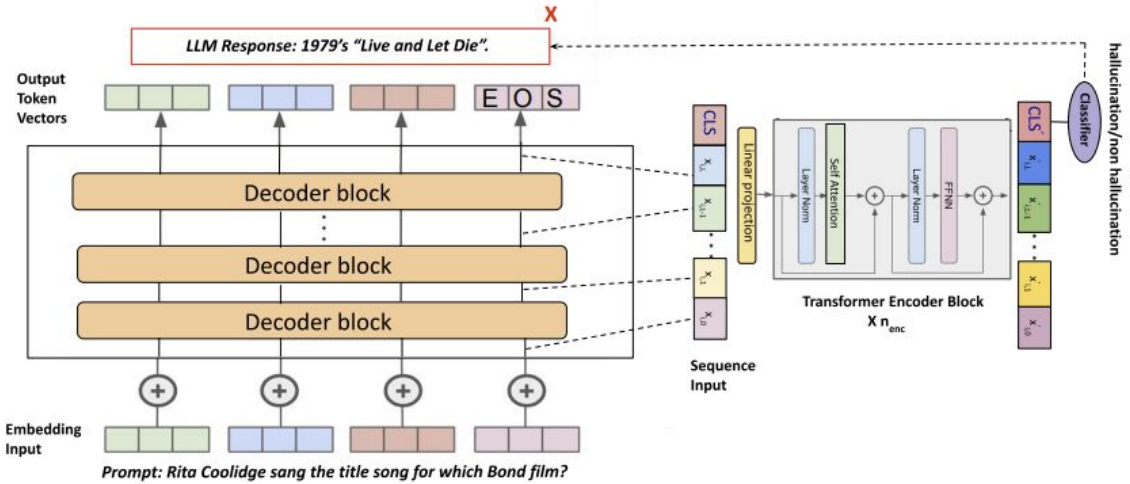
Absolute improvement across LLaMA models

Chuang, Xie, Luo, Kim, Glass, He. DoLa: Decoding by Contrasting Layers Improves Factuality in Large Language Models. arXiv, 2024.

Probing Attention Patterns

🏆 CLAP: Attention Pattern Analysis (2025)

Processing the entire residual stream of an LLM as a sequence using an attention mechanism, enabling fine-grained detection and reliable mitigation.



WikiData, NQ, TQA

+6.5%

AUC improvement
on Out-Of-Distribution tasks

Suresh, Aljundi, Nkisi-Orji, Wiratunga. Cross-Layer Attention Probing for Fine-Grained Hallucination Detection. TrustAI, 2025.

Key Insight: Internal Signals Predict Errors Early

💡 Hallucinations are linearly separable in hidden state space

Simple linear probes on hidden states detect hallucinations as effectively as expensive multi-sample methods — at near-zero additional cost. The model 'knows' it's about to hallucinate before the error appears in the output.

Limitations

📏 Short-Form Ceiling

Most methods are validated on short-answer benchmarks. In long-form generation—where hallucination errors compound across sentences—performance degrades unpredictably.

🐌 Latency Overhead

Multi-layer probing, attention-graph topology analysis, and contrastive forward passes each add non-trivial compute, which can be prohibitive for latency-sensitive applications.

Hallucination Suppression II. DPO for Factuality Preference

Key idea: Sample multiple responses, automatically rank them by factual accuracy, and use DPO to teach the model to prefer its own more truthful generations.

1 Sample

Generate multiple responses for each prompt

2 Score

Auto-score factuality via retrieval or self-confidence

3 Rank

Create preference pairs: factual > non-factual

4 Train

DPO training on factuality-ranked pairs

🔑 FactTune (2023): Reduced factual errors by 58% on biography generation using automated retrieval-based scoring — no human labels needed.

Fine-Grained Alignment: Sentence-Level & Claim-Level

Response-level DPO treats mixed-quality responses as uniformly good or bad. Fine-grained methods target errors precisely.

📐 FactAlign (2024)

Extends KTO to sentence-level optimization (fKTO), scoring each sentence independently for factual accuracy.

Result: **+13.5%** Factual F1 improvement on LongFact while maintaining helpfulness.

🎯 Mask-DPO (2025)

Applies sentence-level masking to DPO loss — only penalizes the specific sentences that are wrong, not the whole response.

Result: 8B model surpasses 70B model in factuality by precisely targeting errors within mixed-quality responses.

Factuality-Aware Reinforcement Learning

Key idea: Go beyond preference pairs: reward truthful reasoning steps during RL training.

🎲 TruthRL (2025)

Ternary reward structure:
✅ Correct answer → positive reward
🤷 Abstention → small negative reward
❌ Hallucination → large negative reward

Makes abstention mathematically preferable to guessing.

Result: 28.9% hallucination reduction across benchmarks.

🧠 KnowRL (2025)

Incorporates factuality rewards into GRPO:

- Per-step factuality verification during training
- Models learn to reason correctly, not just answer
- External KB-based reward signal

Result: 20.3% reduction in incorrect rate on SimpleQA while preserving reasoning on GPQA.

Key Finding: Small Models Can Beat Giants

8B > 70B

An 8B parameter model with Mask-DPO surpasses a 70B model in factuality by precisely targeting sentence-level errors during training.

(Mask-DPO, 2025)

Limitations

🔑 Knowledge-Base Bottleneck

Each atomic fact is verified against an external KB, so only works where comprehensive, up-to-date KBs exist, challenging for fast-moving domains (e.g., recent events, niche science).

🎯 Correct-Answer Laundering

Outcome-based RL can still reward a reasoning chain that contains fabricated intermediate steps, once the final answer is correct.
Per-step factuality rewards may also leave subtle mid-chain hallucinations unpenalized when verification coverage is incomplete.

Hallucination Suppression III. Verification-Based Methods

Key idea: Break long-form text into atomic claims, verify each one independently, then aggregate results to score overall factuality.

1
Decompose

Break output into atomic facts: "Curie studied at the Sorbonne"

2
Retrieve

Search for evidence for each claim: Google, KGs, documents

3
Verify

Check each claim against evidence: supported / not supported / unknown

4
Aggregate

Compute factuality score: # supported / # total claims

Key Verification Methods: From Research to Practice

FacTool (2023)

Multi-tool verification: search, code, math, KGs

Generalizes across multiple domains

SAFE (2024)

LLM agents + Google Search verify with superhuman accuracy

20x cheaper than human annotation

RefChecker (2024)

Claim-triplet granularity with 3-way classification

+26% over FacTool in human correlation

FActScore (2024)

Trained on carefully constructed synthetic verification examples.

770M model matches GPT-4 fact-checking at 400x lower cost

📏 **Limitations:** Reliance on external data

Hallucination Suppression IV. Confidence and Calibration

Key idea: Can models distinguish what they know from what they're guessing? If so, we can build systems that abstain rather than hallucinate.

✅ Calibrated Model

"The capital of France is Paris."
Confidence: 98%

→ High confidence, correct answer

⚠️ Overconfident Model

"Shakespeare was born in 1562."
Confidence: 95%

→ High confidence, wrong answer (1564)

🤷 Ideal Uncertainty

"I'm not sure when the Treaty of Tordesillas was revised."

→ Model knows its limits

Two Approaches: Verbalized vs. Token Probabilities

📊 Token Probabilities

Use the model's actual output token log-probabilities

- Direct access to model's internal uncertainty
- Predictive entropy over token sequences
- Often poorly calibrated after fine-tuning
- Conflates meaning uncertainty with wording uncertainty
- ⚠️ High confidence ≠ correct

💬 Verbalized Confidence

Ask the model: "How confident are you? (0-100%)"

- Works with black-box APIs (no logits needed)
- GPT-3 can learn well-calibrated verbal scores
- **P(True)**: ask model to predict its own correctness
- Larger models are better calibrated
- ⚠️ Systematic overconfidence after RLHF

Signal 1—Semantic Entropy: Uncertainty Over Meaning

Key idea: Don't measure uncertainty over words — measure it over meanings. If all sampled responses say the same thing differently, the model is confident. (Semantic Entropy)

1

Sample Responses

Generate 5-10 different answers to the same query

2

Cluster by Meaning

Group semantically equivalent answers using NLI

3

Compute Entropy

High entropy = many clusters = model is uncertain

4

Flag or Abstain

High semantic entropy → likely hallucination → abstain

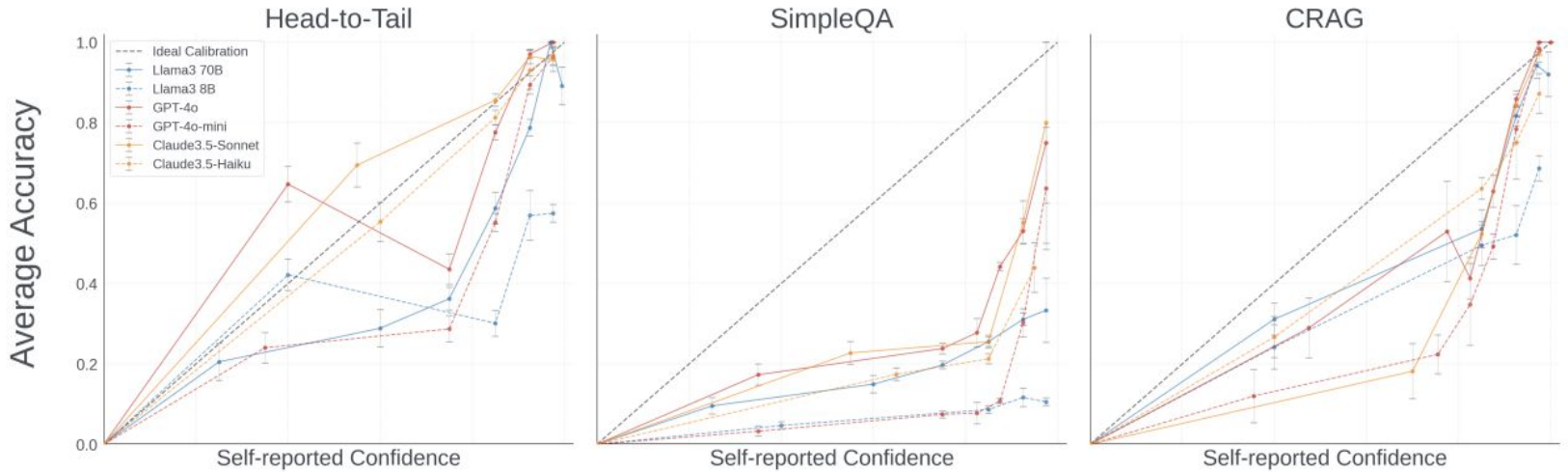
⚠️ Limitation: Requires 5-20 generations per query → expensive. SEP probes can predict semantic entropy from a single pass at near-zero cost.

Signal 2—Model Reported Confidence

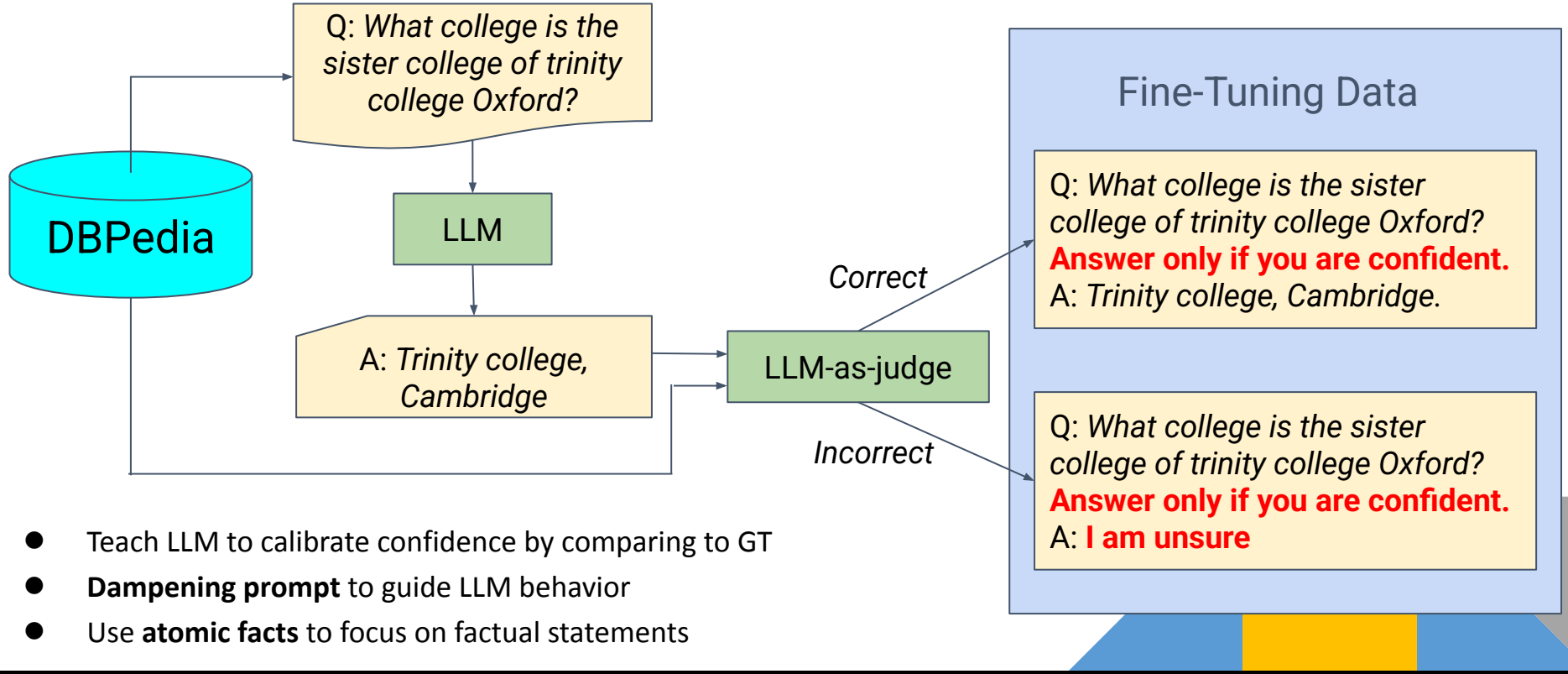
🚨 Overconfident Hallucinations (CHOKE)

Models can hallucinate with high certainty on questions they demonstrably know — defeating ALL uncertainty-based methods that assume low confidence = error. — Fundamental limit

Experiments on Three Factuality Benchmarks



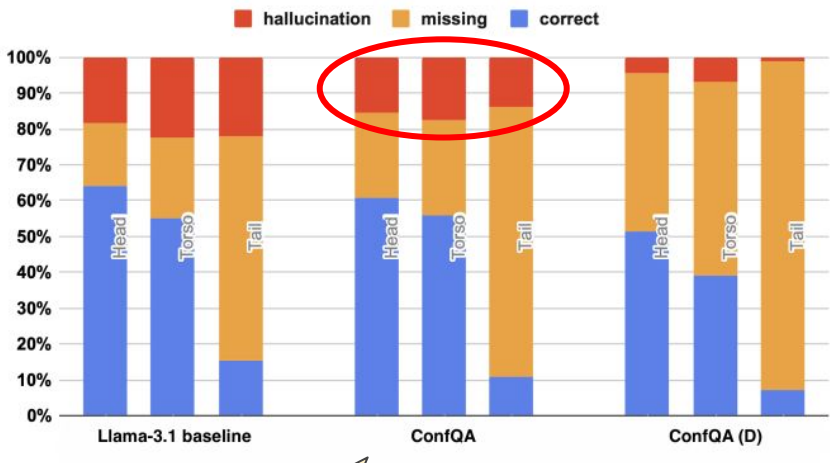
Can We Teach LLMs to Refrain from Hallucinating?



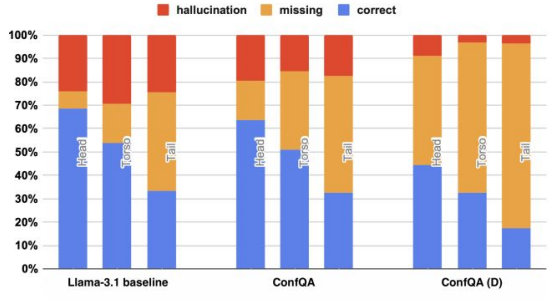
- Teach LLM to calibrate confidence by comparing to GT
- **Dampening prompt** to guide LLM behavior
- Use **atomic facts** to focus on factual statements

Yin Huang, Yifan Ethan Xu, Kai Sun, Vera Yan, Alicia Sun, et al. ConfRAG: Confidence-Guided Retrieval-Augmented Generation. arXiv, 6/2025

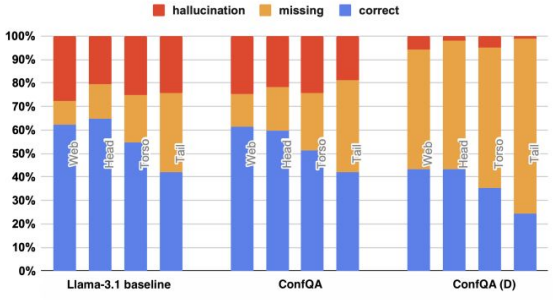
Can We Teach LLMs to Refrain from Hallucinating?



) IMDB



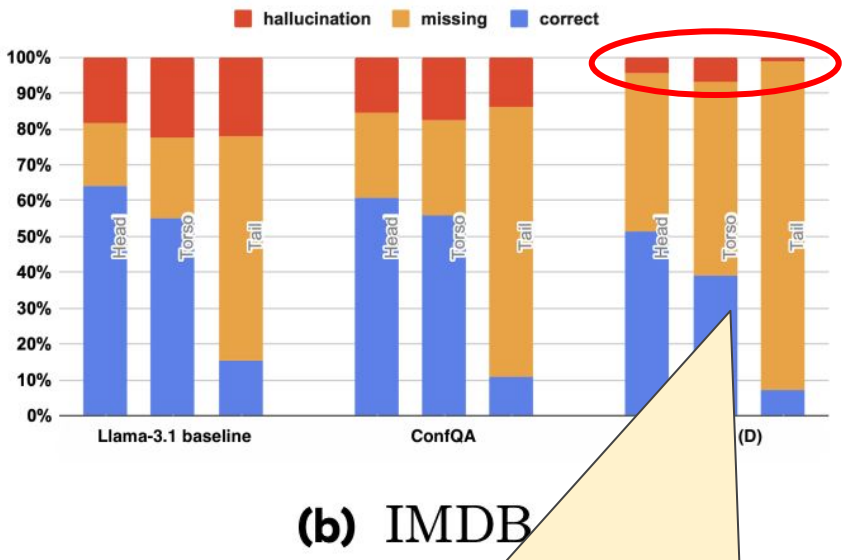
(a) DBPedia



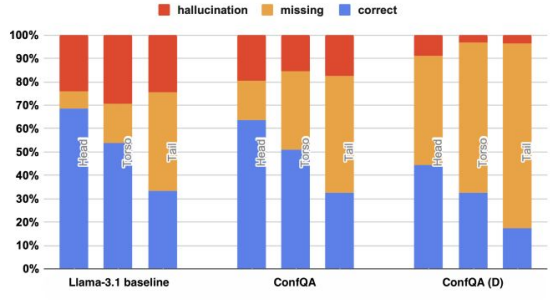
(c) CRAG

1. Our fine-tuned model has similar correct% and mild hallucination% reduction

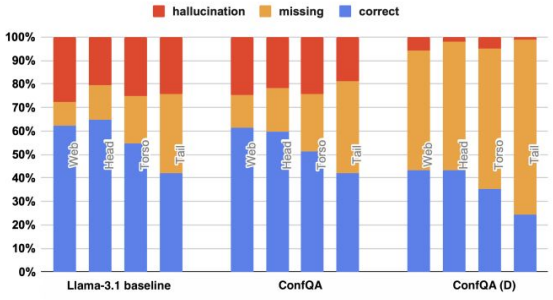
Can We Teach LLMs to Refrain from Hallucinating?



(b) IMDB



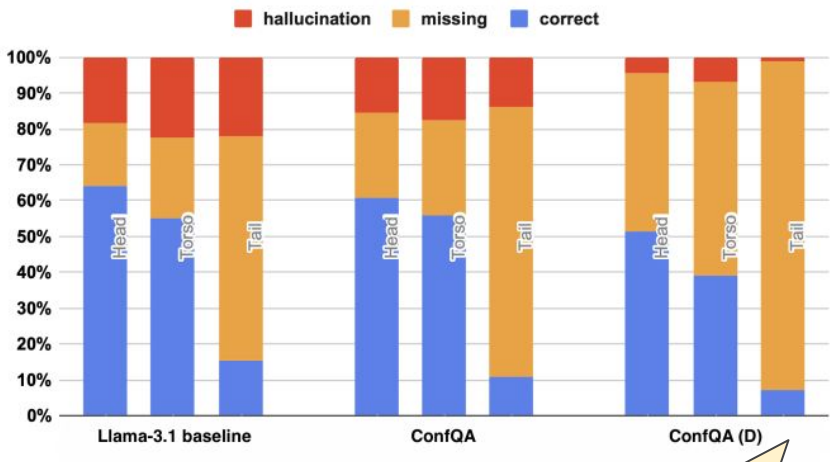
(a) DBPedia



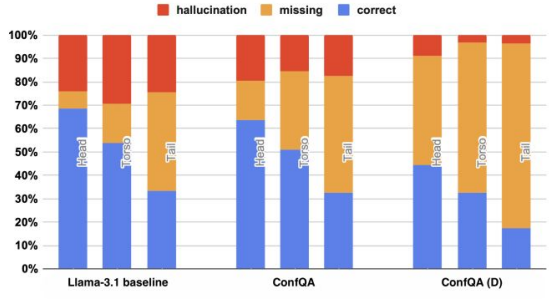
(c) CRAG

2. Dampener is critical in training. At inference, with the dampener prompt, hallucination reduced to below 5%

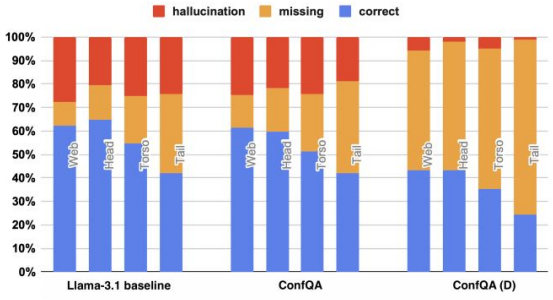
Can We Teach LLMs to Refrain from Hallucinating?



(b) IMDB



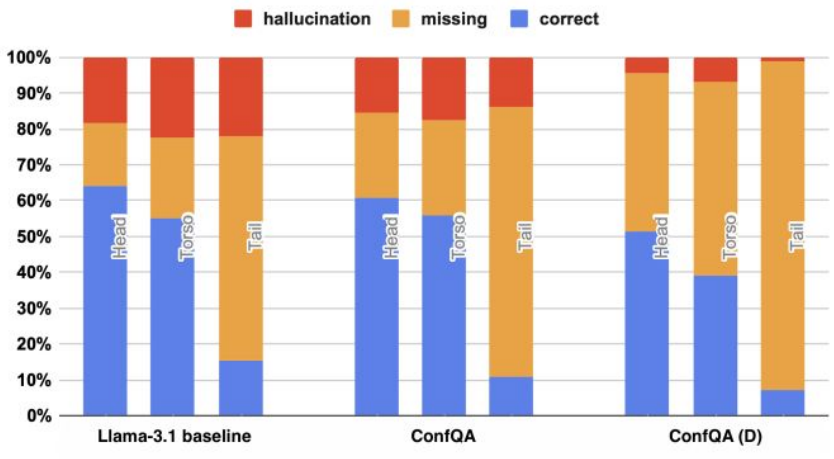
(a) DBPedia



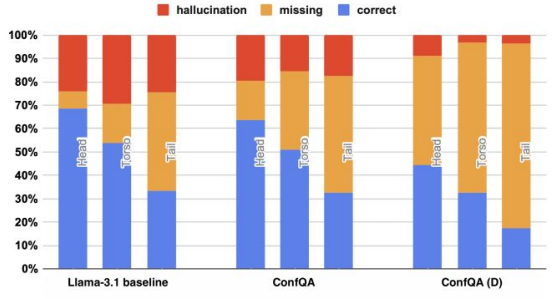
(c) CRAG

3. Suppress more for long-tail facts.

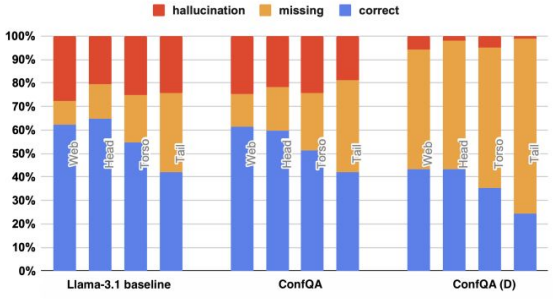
Can We Teach LLMs to Refrain from Hallucinating?



(b) IMDB



(a) DBPedia



(c) CRAG

4. Transfer well from DBPedia to IMDb. However, training data from MMLU (w. non-factual questions) significantly lower accuracy

Can We Teach LLMs to Refrain from Hallucinating?

Model	Long Fact				Alpaca Fact				Biography			
	Prec	Rec	F1	Miss	Prec	Rec	F1	Miss	Prec	Rec	F1	Miss
Llama3.1	64.5	65.4	64.3	0	62.3	71.0	63.8	0	35.4	40.3	37.1	0
RAG (Llama3.1) (Yu et al., 2022)	71.7	74.6	72.7	0	65.8	74.3	66.0	0	44.9	48.1	43.8	0
ConfQA	67.0	67.7	66.7	0.8	62.2	71.1	63.8	0.4	42.0	46.5	42.6	12.6

Table 4 ConfQA improves precision and recall for long-form answer generation.

5. Transfer well to long-form answers w. higher quality, and no regression on other tasks

6. Feeding GT only will teach LLMs to hallucinate

Model	MMLU (5-shot)	MMLU-Pro
Llama3.1	82.7	66.3
ConfQA	82.8	65.4

Table 5 ConfQA does not regress on MMLU.

The Benchmarks—Why Factuality Evaluation is Hard

🎭 Fluency Masks Errors

Hallucinations are grammatically perfect and plausible. Standard ROUGE/BLEU scores miss them entirely because most tokens are correct.

🌊 Open-Ended Outputs

Long-form responses contain dozens of claims. A single wrong date buried in 500 correct words is hard to find.

📊 Scale Problem

Human evaluation is expensive (~$1-5/response) and slow. Automatic metrics often correlate poorly with human judgments.

🔄 Moving Target

World knowledge changes. Benchmarks become stale. Models may memorize evaluation data (contamination).

Key Factuality Benchmarks

TruthfulQA

Tests if models repeat common misconceptions

MC1/MC2 accuracy
817 questions

SimpleQA

Short-form factual QA
adversarial to GPT-4

Frontier models <40%
OpenAI benchmark

FActScore

Atomic fact precision for biographies

% supported claims
Wikipedia-grounded

HALOGEN

10K+ prompts, novel taxonomy of causes

Failed recall vs. fabrication types

HaluBench

Multi-domain hallucination evaluation

RAG, summarization, knowledge QA

Atomic Evaluation: Breaking Down Claims

Key insight: Decomposing text into atomic claims before verification consistently outperforms holistic response-level evaluation across all domains.

❌ Holistic Evaluation

"Rate the factuality of this paragraph."

Problems:
- Misses subtle errors in correct text
- Inconsistent across annotators
- ROUGE/BLEU scores don't measure facts
- One wrong date in 500 words = high score

✅ Atomic Claim Evaluation

Break into: "Curie won Nobel in Physics in 1903."
"She studied at the Sorbonne." etc.

Advantages:
- Each claim independently verifiable
- Consistent, reproducible scoring
- Fine-grained error localization
- FActScore, SAFE, VeriScore standard

The Problem with Benchmarks

Static benchmarks are fragile — models may memorize evaluation data or be gamed.

📋 Data Contamination

Training data may include benchmark questions.
Models score well on memorized answers, not generalization.
Even GPT-4 shows contamination on some benchmarks.

🎯 Metric Gaming

Automated factuality metrics can be gamed by superficial edits.
Adversarial rewriting reduces GPT-4o detector AUC by 17.5 points.
Metrics rely on shallow heuristics, not deep understanding.

⏰ Temporal Decay

World facts change. Benchmark answers become stale.
DyKnow: even GPT-4 gives ~15-20% outdated answers.
Dynamic benchmarks needed for temporal knowledge.

Human vs. Automatic Evaluation: SAFE Surpasses Humans

🏆 SAFE: LLM-based verification achieves superhuman accuracy at 20x lower cost

LLM agents with search access verify facts more accurately than human annotators. 72% of disagreements with humans were resolved in SAFE's favor. This is the decompose-and-search paradigm at scale.

(SAFE, 2024)

👤 Human Annotators

Expensive (~$1-5/claim)
Slow (minutes/claim)
Inconsistent across annotators
Limited by annotator knowledge

🤖 SAFE (LLM Agents)

Cheap (~$0.05/claim)
Fast (seconds/claim)
Consistent and reproducible
Access to search engines

📊 Key Findings

SAFE > humans on accuracy
72% of disagreements favor SAFE
20x cost reduction
Scales to millions of claims

Recap & Takeaways for Factuality Landscape

What we covered

Knowledge Encoding

🧠 **Knowledge Internalization**

1. Knowledge update
2. Post-training alignment

Knowledge Recall

🛡️ **Hallucination Suppression**

1. Internal Parameter Intervention
2. DPO for Factuality Preference
3. Verification-based
4. Confidence and Calibration

Takeaways

1. Hallucination is because of both **empty shelves** (not internalized) and **lost keys** (recall failure)
2. Many successes, but insufficient for factuality

   → **RAG (external knowledge) is the rescue**

01-03

RAG at a Glance (10 min)

What is RAG?
What are its unique advantages?
Where are we for RAG?

Why RAG? The Motivations

📅 Outdated Knowledge

LLM parameters are frozen at training time. They can't know about events after their training cutoff date.

💭 Hallucination

Without grounding, LLMs confidently fabricate facts. RAG provides evidence to anchor responses.

🔒 No Source Attribution

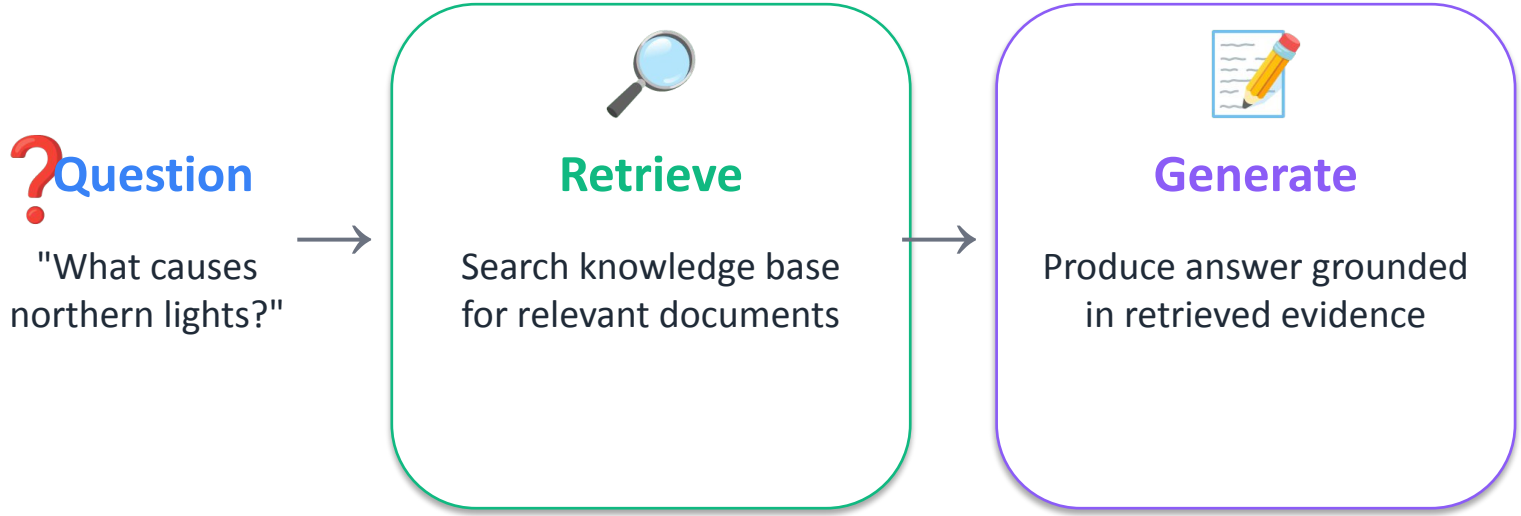
Parametric-only models can't cite sources. RAG enables transparency through traceable evidence.

💰 Expensive Retraining

Updating knowledge via retraining costs millions. RAG updates knowledge by simply updating the document store.

What Is RAG? The Core Idea

RAG augments language models by retrieving relevant external information from web, knowledge graphs, and documents to ground responses in factual evidence.



Prompting vs. Fine-tuning vs. RAG

	Prompting	Fine-tuning	RAG
Knowledge Update	Limited by context ⚠️	Needs retraining ❌	Instant ✅
Cost	Very low ✅	High (GPU hours) ❌	Low (index docs) ✅
Accuracy	Variable	High (for known tasks)	High (with good retrieval)
Attribution	No ❌	No ❌	Yes — cites sources ✅
Best For	Simple tasks, few-shot examples	Behavioral changes, style/format	Dynamic knowledge, fact-heavy tasks

RAG: Key Research Findings—

"A small model with the right retriever beats a giant model flying blind."

11B + retrieval > 540B alone

Atlas (2022): a small retrieval-augmented model outperforms a 540B parametric model on few-shot tasks

(Atlas, 2022)

Retrieval quality > model size

Switching embedding models swings accuracy by 17-34% — invest in retrieval, not just bigger generators

25x fewer parameters, same quality

RETRO (2022): retrieval from 2-trillion-token database matches GPT-3 with 25x fewer parameters

(RETRO, 2022)

Three RAG Paradigms

🔗 Modular RAG

Most common today

Distinct, swappable stages: trigger → rewrite → retrieve → post-process → generate. Each component can be independently improved.

🕸 Graph RAG

Rising rapidly (2024-25)

Builds knowledge graphs from documents. Enables multi-hop reasoning and relationship-aware retrieval that flat text retrieval cannot provide.

🤖 Agentic RAG

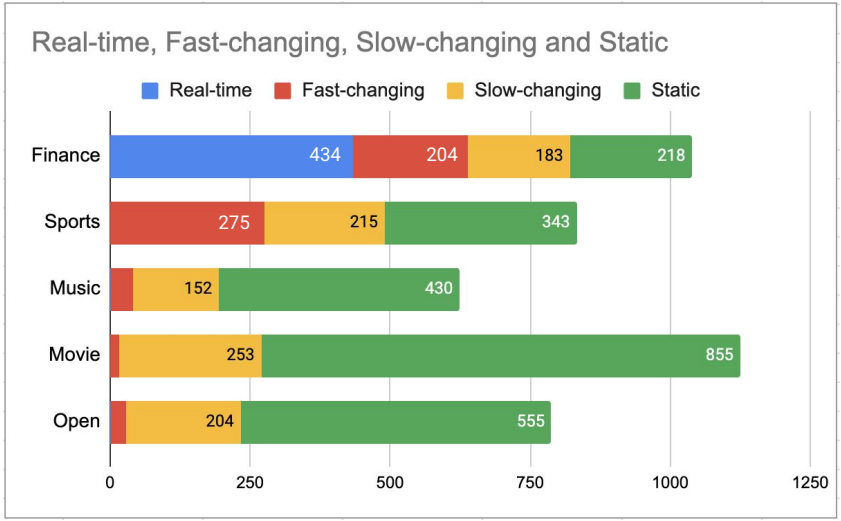
Cutting-edge frontier

Autonomous systems that dynamically decide when, what, and how to retrieve. Interleaves retrieval with reasoning via RL or self-reflection.

Where Are We in This Journey?
—A Quantitative Answer

Rich and *Insightful* Question-Answer Set

- 4400+ QA pairs from 5 domains (Finance, Sports, Music, Movie, Encyclopedia)
- Questions for *static*, *slow-changing*, *fast-changing*, and *real-time* information
- Questions for *head*, *torso*, and *tail* entities
- *Simple-fact* questions and *complex* questions



Total	Simple	Simple w. Cond	Set	Comparison	Aggregation	Multi-hop	Post-processing	False Premise
4409	1205	689	403	546	489	382	180	525

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, et al. CRAG–Comprehensive RAG Benchmark. NeurIPS, 2024.

Accessible Retrieval Content

- 220K webpages: 50 webpages for each question from BraveAPI web search
- Mock KG: 2.6M entities, 1:30 signal-to-noise ratio
- Mock APIs: 38 mock APIs

Reliable Tasks and Evaluation

- Task 1: Answer generation over top-5 web search results—**Answer Summarization**
- Task 2: + Mock-KG Search API—**Structured Search**, **Answer Selection**
- Task 3: + 50 web search results—**Search Ranking**

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, et al. CRAG–Comprehensive RAG Benchmark. NeurIPS, 2024.

LLM-Only vs. Straightforward RAGs

Domain	Solution	Accuracy	Hallucination	Missing	Factuality =Accuracy-Halluci.
Llama-3 (70B)	LLM Only	32%	29%	39%	3%
	Straightforward Task 3	41%	32%	28%	9%
GPT4 Turbo	LLM Only	34%	14%	53%	20%
	Straightforward Task 1	36%	28%	36%	8%
	Straightforward Task 2	41%	25%	34%	16%
	Straightforward Task 3	44%	30%	26%	14%

RAG may improve accuracy

KG w. crisper retrieval results helps

RAG may also bring more hallucinations

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, et al. CRAG–Comprehensive RAG Benchmark. NeurIPS, 2024.

KDD Cups Winning Solutions

Tasks	Auto-eval		Manual-eval	
	GPT-4	Llama 3	KDDCup Winning team	
Task 1	8%	5%	29%	30%
Task 2	16%	8%	30%	32%
Task 3	13%	9%	31%	36%

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, et al. CRAG–Comprehensive RAG Benchmark. NeurIPS, 2024.
Xiao Yang, Kai Sun, et al. KDD Cup CRAG competition: Systems, Findings and Learnings. IEEE Data Engineering Bulletin, 2024.

State-of-the-Art Industry Solutions (2024)

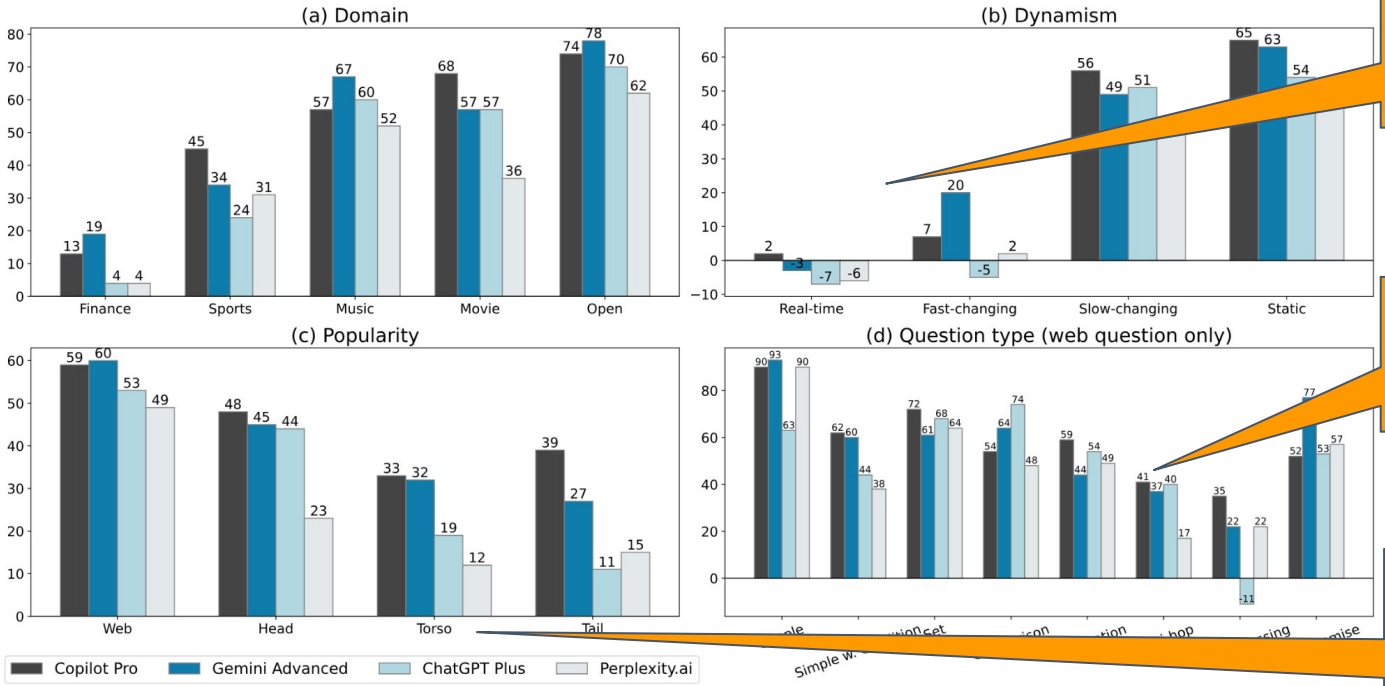
System	Perfect	Acceptable	Incorrect	Missing	Factuality	Latency (s)
Copilot Pro	63%	12%	18%	8%	**50.4%**	11.6
Gemini Advanced	61%	10%	17%	13%	*49.5%*	5.2
ChatGPT Plus (4o)	60%	13%	25%	2%	42%	6.2
Meta Wearables	53%	10%	16%	22%	41%	**3.4**
Perplexity.ai	56%	9%	25%	10%	35%	*4.6*
Notes: 1. Manual annotations. 2. Retrieval by the SOTA solutions themselves.						

Perfect < 63%. Still a big gap to fill

Different latency-quality tradeoffs

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, et al. CRAG–Comprehensive RAG Benchmark. NeurIPS, 2024.

SOTA Industry Solutions on Diff Dimensions



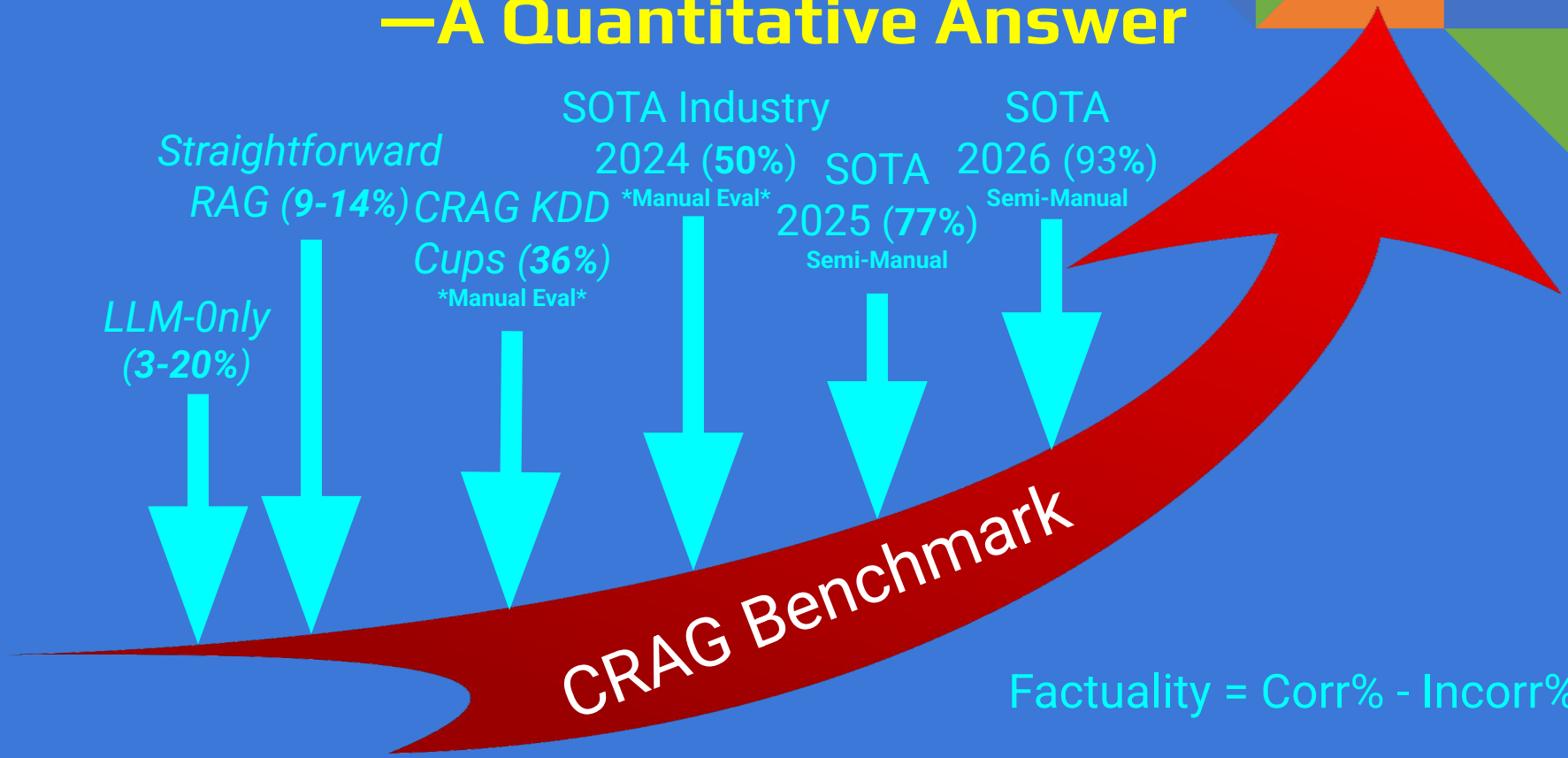
Improvements needed for **fast-changing** facts

Improvements needed for **complex** questions

Improvements needed for **torso/tail** questions

Figure 3: SOTA systems human-eval scores (in percentage) across different dimensions.

Where Are We in This Journey? —A Quantitative Answer



Xiao Yang, Yifan Ethan Xu, Kai Sun, Jiaqi Wang, et al. KDD Cup CRAG Competition: Systems, Findings, and Learning. IEEE Bulletin, 2024.

01-04

Tutorial Structure (5 min)

What is covered in the tutorial?
How did AI help?

Tutorial Outline



1. Introduction (40 min): Motivation, Factuality landscape, RAG at a glance

2. RAG Deepdive (110 min)
    - Modularized RAG (50 min)

    - Graph-based RAG (30 min)

    Break
    - Agentic RAG (30 min)

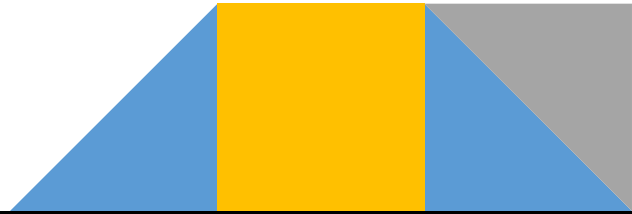
3. Advanced Topics (40 min)
    - Deep research (20 min)

    - Multi-modal RAG (20 min)

4. Conclusions and future directions (15 min)

Section Structure

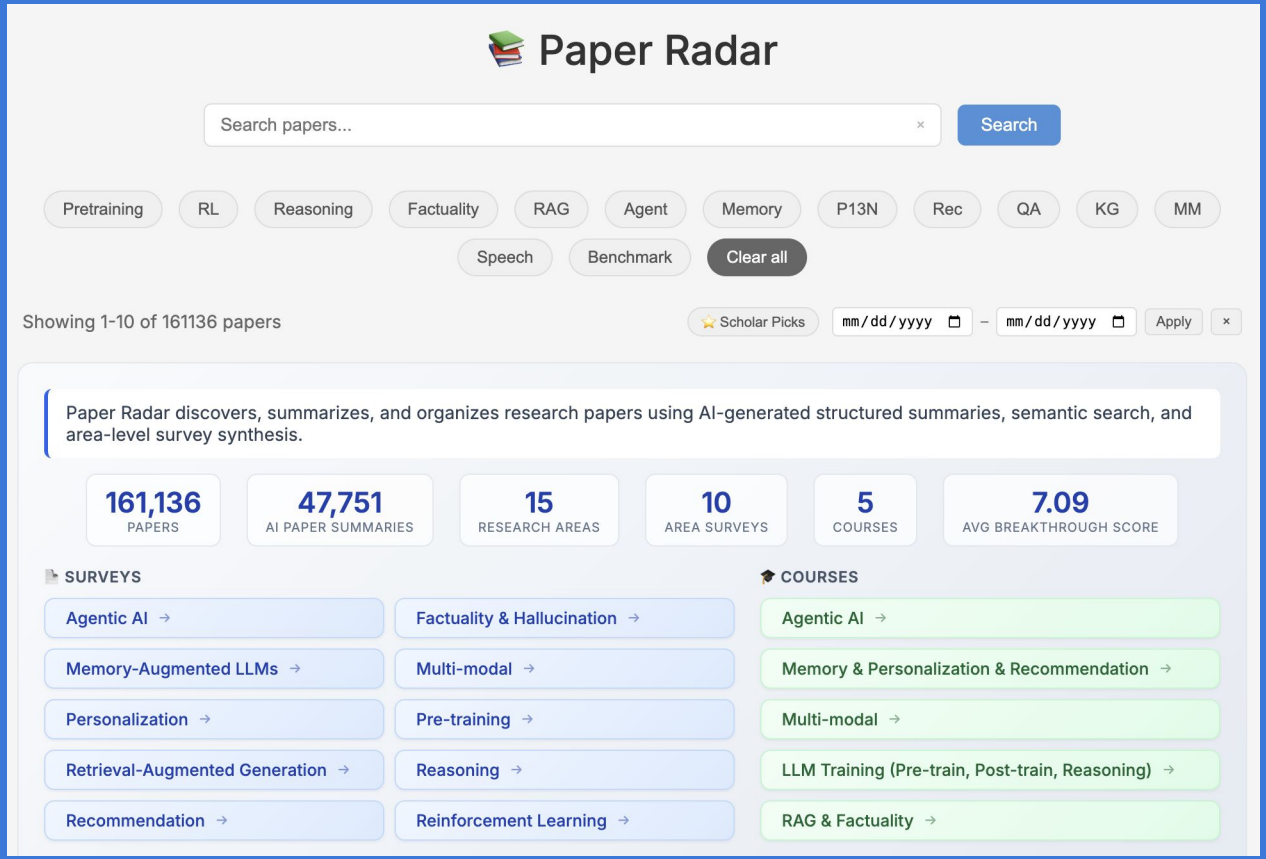
1. Problem framing
   What is the problem definition?
2. Why it matters
   Where do naive solutions fall short? What are challenges and opportunities?
3. Solution landscape
   What has been explored to solve the problem?
4. Canonical method deep-dive
   What are details regarding architecture, training / inference strategy, experimental results for a canonical method?
5. Recap & Future Work
   What are future research directions to improve this solution?



Running Examples Throughout the Tutorial

1. Simple question
   What is the battery capacity of Tesla Model 3??
2. Complex question
   Which has a longer driving range — the Tesla Model 3 or the Lucid Air?
3. Deep Research question
   I'm considering buying an electric vehicle in 2026.
   Which model is the best choice, and why?
4. Multi-modal question
   [In view: EV in the parking lot]. How far can it go on a full charge?

AI-Facilitated Tutorial Preparation



- **630K** views since March
- **~4K** daily views
- **~500** daily views on surveys and courses

AI ResearchMate

consider a scenario where I record what user sees every day, like taking a pic every 4 sec. Now I'd like to support QA on such memory. What are solutions? tips? challenges?

📊 **Sparse Memory QA** — sub-topic under **Memory-Augmented LLMs**
📊 **Dense Memory QA** — sub-topic under **Memory-Augmented LLMs**
See sub-topics on the left panel 👈

Similar sub-areas:

- LLM-enhanced Recommendation (in **Recommendation**)
- Memory Recall (in **Memory-Augmented LLMs**)
- Conversational Personalization (in **Personalization**)
- Search and Adaptive Compute (in **Reasoning**)
- Leaderboard / Benchmarks / SOTA (in **Personalization**)

Recent papers:

- VideoMLA: Low-Rank Latent KV Cache for Minute-Scale Autoregressive Video Diffusion
- Unlocking the Working Memory of Large Language Models for Latent Reasoning
- On Language Generation in the Limit with Bounded Memory

📍 Sparse Memory QA 📍 Dense Memory QA

tl;dr: For QA over continuous photo streams (e.g., every 4 sec), the leading solutions combine **multi-signal retrieval** (time, location, semantics) with **offline metadata augmentation** and **recursive question decomposition**; key challenges include near-duplicate filtering, vague temporal reasoning, and scalability.

Solutions

- **Pensieve (Multi-Signal Retrieval)**: Pre-augments each captured image offline with OCR text, captions, and timestamps, then retrieves using multiple explicit signals—date matching, location proximity, and semantic similarity—rather than embeddings alone. Achieves up to 14% accuracy improvement over standard multimodal RAG on MemoryQA.
- **ReQAP (Recursive Question Decomposition)**: Recursively breaks complex questions into a tree of RETRIEVE → EXTRACT → AGGREGATE operators, handling analytical queries (e.g., "how many times did I...") over 100K+ token heterogeneous archives. Supports distillation to small on-device models for privacy.
- **End-to-End Memory Networks (MemN2N)**: Performs multiple hops of soft attention over memory slots, progressively narrowing which memories are relevant—useful when the answer depends on combining two sparse fragments.
- **Entities as Experts / MoWE**: Sparse entity-specific memory slots activated on demand, matching models 10× larger with far fewer FLOPs.

Tips

- **Offline metadata augmentation is critical**: Run OCR, captioning, and timestamp extraction at ingest time so retrieval can operate over rich text signals rather than raw pixels.
- **Use multi-signal scoring, not just semantic similarity**: Temporal recency, date matching, and location matching dramatically improve recall in dense streams.
- **Train with noise injection**: Injecting irrelevant but semantically similar memories during training makes the answer generator robust to retrieval noise.
- **Fixed vocabulary routing > learned routing**: For knowledge-intensive tasks, deterministic routing to expert memory slots outperforms dynamic gating.
- **Distill to on-device models**: Enables private personal QA without sending data to the cloud.

Challenges

- **Near-duplicate overload**: Hundreds of visually similar frames (e.g., same coffee shop daily) flood retrieval; deduplication across modalities is essential.
- **Vague temporal/spatial references**: Queries like "last Tuesday" or "near the mall" require specialized parsing beyond embedding similarity.
- **Aggregation queries**: Counting, comparing, or summarizing across many events cannot be solved by retrieve-then-generate alone—requires structured decomposition.
- **Scalability**: Offline augmentation pipelines (OCR, captioning) may not keep pace with a 4-second capture rate in real time; incremental/streaming processing is needed.
- **Entity coverage**: Sparse entity methods fail on novel entities unseen at training time.
- **Evaluation gaps**: Current benchmarks (MemoryQA, PerQA) are narrow; generalization to open-ended conversational settings remains unproven.

📍 Sparse Memory QA 📍 Dense Memory QA

AI ResearchMate

tell me some research directions for this

tl;dr: Research directions for memory QA on continuous visual recordings span adaptive retrieval, real-time augmentation, on-device inference, unified benchmarks, and hybrid analytical pipelines.

Based on the current state and known gaps in both Sparse and Dense Memory QA, here are promising research directions:

- **Incremental/streaming memory augmentation**: Current systems like Pensieve require batch pre-processing (OCR, captioning, metadata extraction) of all memories offline. A key direction is building pipelines that process new memories as they arrive in real time, essential for continuously growing photo streams.
- **Adaptive multi-hop termination**: Multi-hop attention (as in MemN2N) scales linearly with hop count. Developing mechanisms that dynamically decide when sufficient evidence has been gathered would improve efficiency for large memory stores.
- **Dynamic entity discovery**: Sparse entity memory methods require pre-defined entity vocabularies and cannot handle novel entities. Research into expanding entity embeddings incrementally during inference would address ever-changing personal contexts.
- **On-device distillation for private QA**: Distilling capable memory QA models to small on-device models enables private personal QA without sending data to cloud services—critical for privacy-sensitive visual lifelogs.
- **Hybrid retrieval-extraction-aggregation pipelines**: Flat retrieval cannot handle counting, comparison, or temporal aggregation queries. Recursive decomposition approaches (like ReQAP) that combine RETRIEVE, EXTRACT, and COUNT operators in a unified pipeline are an active frontier.
- **Multi-signal retrieval beyond semantics**: Temporal recency, date matching, and location proximity are essential signals for dense personal memories where semantic similarity alone retrieves too many near-duplicates.
- **Unified cross-benchmark evaluation**: Each current method is tested on its own benchmark (MemoryQA, PerQA, bAbI, TriviaQA). Developing standardized suites spanning multimodal recall, conversational, and complex analytical queries remains an open need.

📍 Sparse Memory QA 📍 Dense Memory QA

regarding this: Hybrid retrieval-extraction-aggregation pipelines: Flat retrieval cannot handle counting, comparison, or temporal aggregation queries. Recursive decomposition approaches (like ReQAP) that combine RETRIEVE, EXTRACT, and COUNT operators in a unified pipeline are an active frontier. Propose some concrete solutions

tl;dr: Concrete solutions for hybrid retrieval-extraction-aggregation pipelines combine recursive question decomposition (à la ReQAP) with multi-signal retrieval and multi-hop reasoning to handle counting, comparison, and temporal aggregation over dense memory stores.

Based on the methods and insights in this area, here are actionable solution directions:

- **Recursive operator-tree pipeline**: Decompose complex queries into a tree of typed operators—RETRIEVE (find candidate memories), EXTRACT (parse structured fields from unstructured text/images), and COUNT/COMPARE/AGGREGATE (perform the final computation). Each node operates independently, enabling parallelism and composability for queries like "How many times did I visit Starbucks last month?"
- **Multi-signal pre-filtering before aggregation**: Before feeding candidates into an aggregation operator, score them with explicit temporal (date matching, recency), spatial (location proximity), and semantic signals—as Pensieve does—to prune near-duplicates and irrelevant entries early, reducing noise for downstream counting or comparison.
- **Multi-hop iterative refinement**: Apply MemN2N-style multiple attention hops over the retrieved memory pool so the system can progressively narrow candidates (e.g., first hop finds "coffee" memories, second hop filters to "Starbucks," third hop restricts to the target month) before the aggregation step executes.
- **Cascade pruning with cached intermediates**: Index and cache intermediate decomposition results (e.g., pre-extracted entities per memory, pre-resolved temporal anchors) so that repeated or similar queries avoid recomputation—addressing the latency challenge of recursive decomposition over 100K+ token archives.
- **Distill to lightweight on-device models**: Once the operator tree is trained at scale, distill the decomposition logic and small extractors to compact models that run locally, enabling private personal QA without sending raw memories to the cloud.

Two Surveys from 1882 Papers

RAG and Factuality for LLMs

https://papers.lunadong.com/rag
https://papers.lunadong.com/factuality

A 4-Week Course on

RAG and Factuality for LLMs

https://papers.lunadong.com/courses/rag_and_factuality/

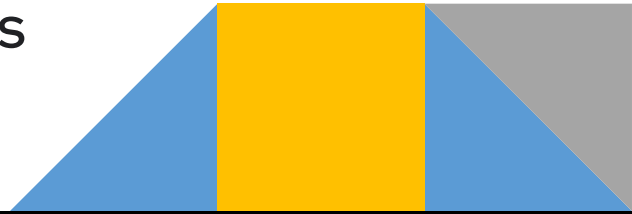
How Did We Generate the Surveys?

- Download Papers from arXiv & Semantic Search

- Generate AI Summaries for each paper
  – Lots of Prompt Engineering

- Generate AI area taxonomy
  – By ourselves, or by AI reading surveys

- Generate AI area surveys & courses
  – Lots of Prompt Engineering



Tutorial Outline



1. Introduction (40 min): Motivation, Factuality landscape, RAG at a glance

2. RAG Deepdive (110 min)
    - Modularized RAG (50 min)

    - Graph-based RAG (30 min)

    Break
    - Agentic RAG (30 min)

3. Advanced Topics (40 min)
    - Deep research (20 min)

    - Multi-modal RAG (20 min)

4. Conclusions and future directions (15 min)

02-01

Modularized RAG (50 min)

One question - five design decisions

Which EV should I buy in 2026?

Trigger	Rewrite	Retrieve	Post-processing	Generate
Need retrieval?	Search intent?	Evidence source?	Final context?	Synthesis?

What can go wrong?

A bad answer can originate from different modules, so diagnosis must be stage-specific.

Wrong trigger

retrieve too often, or miss retrieval and use stale / ambiguous facts

Wrong rewrite

retriever never sees the right intent

Wrong retrieval

good query, weak evidence

Wrong post-processing

right evidence, wrong final context

Wrong Answer

Right context, poor answer generation, hallucation

Modular RAG as a system interface

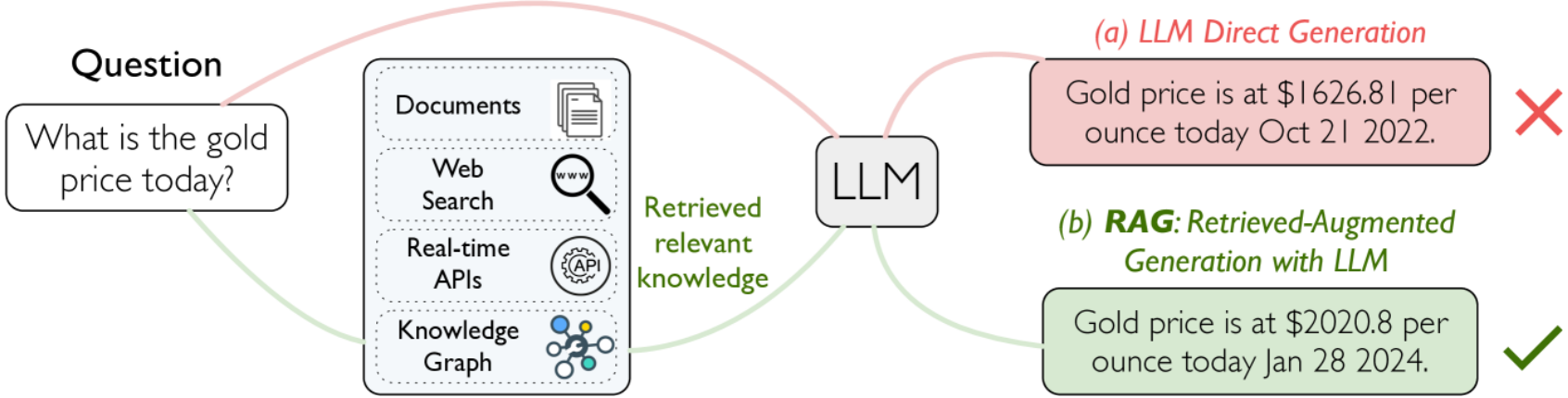
The same user query can be viewed as a sequence of transformations from natural language to evidence and then to an answer.

Module	Question it answers	Typical outputs
Trigger	Should I retrieve at all?	yes/no, route, uncertainty
Rewrite	How do I expose the retrieval intent?	subqueries, filters, query plan
Retrieve	Where do I search, and what do I pull back?	documents, tables, APIs, images
Post-process	Which evidence should survive?	ranked, filtered, compressed context
Generate	How do I synthesize the answer?	natural language answer, report, citation

Source: Modular RAG survey

Why modularity matters

CRAG frames RAG as a multi-source pipeline; each stage creates a distinct failure mode.



Why modularity helps

Documents, web search, real-time APIs, and knowledge graphs can fail differently—through stale sources, weak retrieval, or unsupported synthesis.

Practical implication

Evaluate each stage separately: whether to retrieve, how to rewrite, what to rank, and how to shape evidence.

Source: CRAG (Yang et al., 2024)

02.01.01

RAG Triggering

When should Retriever be triggered?

RAG Triggering

Problem definition: should the model retrieve?
Input: query + context + model signals → Output: retrieve or answer directly
Accuracy, calibration, low latency/cost, and robustness to changing facts

Motivation 1 — Unknown knowledge boundary

“Model 3 battery capacity?” may need retrieval for trim/year; summarizing supplied text does not.

Motivation 2 — Poor calibration

Models can be confidently wrong, so self-reported uncertainty is a weak retrieval gate.

Motivation 3 — Cost–quality trade-off

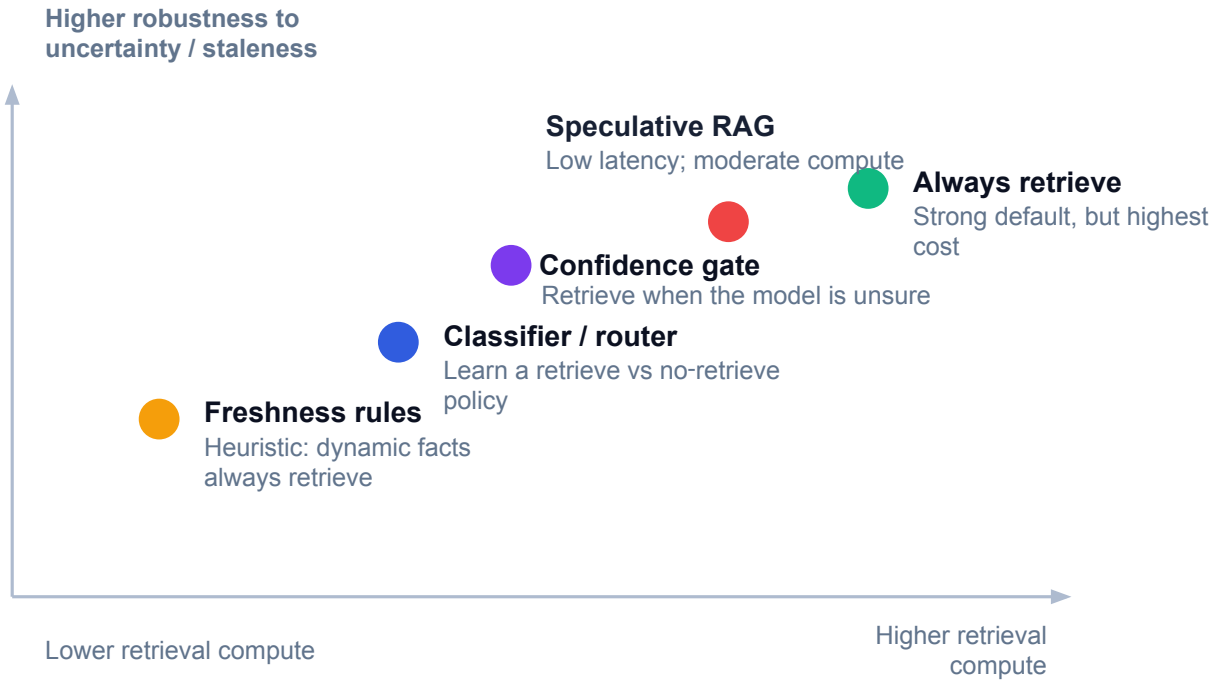
Always retrieving adds latency and noise; skipping retrieval risks stale or unsupported claims.

Motivation 4 — Distribution shift

A trigger must generalize across domains and changing facts without per-domain tuning.

How triggering strategies differ

The design space is not just “retrieve or not”. Different policies trade off cost, robustness, and engineering complexity.



Source: ConfRAG (2025); RAG triggering literature

Three ways to decide whether retrieval is needed

Corpus-Grounded Signals

Estimate whether needed knowledge appeared in pre-training using entity frequency and co-occurrence. Retrieve when corpus support is weak.

QuCo-RAG

Calibrated Model Uncertainty

Train a confidence estimator to predict whether the model can answer correctly without evidence. Retrieve below a learned threshold.

ConfRAG

Self-Routing

Summarizer model decides itself whether it needs to search or not from parametric memory.

Self-Routing RAG

QuCo-RAG — Intuition: what the corpus reveals

Quantifying Uncertainty from the Pre-training Corpus for Dynamic RAG
Min, Zhang, Wu, Cheng — Findings of ACL 2026

The Problem

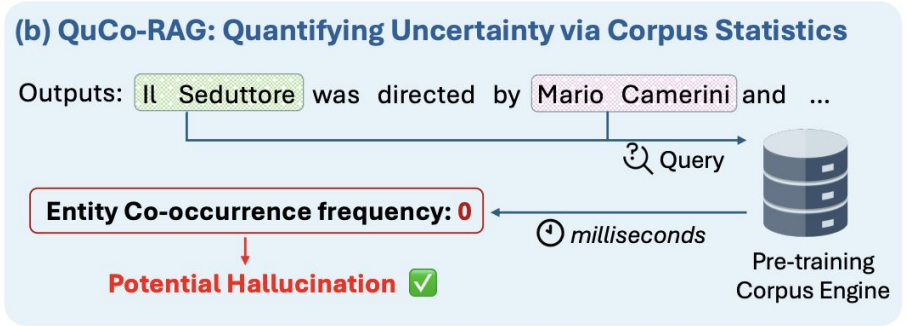
LLM-internal signals (logits, entropy) are unreliable for deciding when to retrieve. Models are poorly calibrated and often show high confidence even when wrong.

The Gap

Prior methods rely on model-internal uncertainty signals. No method uses objective, external evidence about what the model has actually seen during pre-training.

Key Idea

Quantify Uncertainty via Pretraining Corpus. Use pre-training corpus statistics (entity frequency, co-occurrence) via **Infini-gram** to ground retrieval decisions.



QuCo-RAG — Method

Two-stage corpus-grounded retrieval triggering

Stage 1: Pre-Generation

Extract entities from the question → query frequency in the pre-training corpus via Infini-gram (~4T tokens).

If avg frequency < 10^3, trigger retrieval — the model likely lacks this knowledge.

Stage 2: Runtime Verification

Extract knowledge triplets using a 0.5B extractor (89.9% F1). Check entity co-occurrence in the corpus.

If co-occurrence < 1, trigger retrieval for that claim.

Key Properties

- **Model-agnostic:** works with OLMo, Llama, Qwen models as long as you have access to PT corpus
- **Low Latency:** Millisecond-latency corpus lookups via Infini-gram
- **Efficiency:** Distilled 0.5B triplet extractor (Qwen2.5-0.5B, 40K GPT-4o-mini examples)

Example — "What is the battery capacity of Tesla Model 3?" Stage 1 retrieves for the exact trim/year; Stage 2 verifies the capacity claim.

QuCo-RAG — Results

Outperforms GPT-5 built-in web search by 5–9 EM points

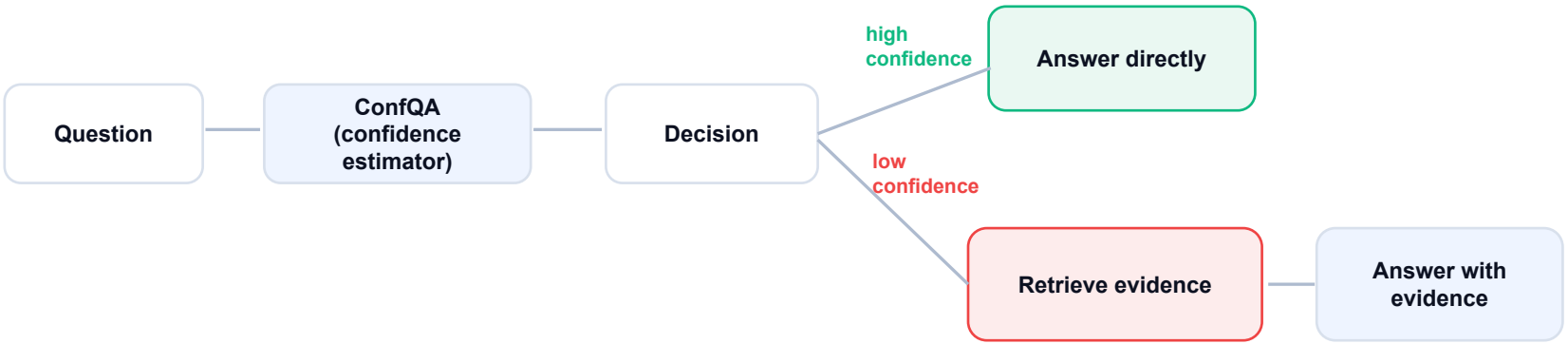
Model	Method	2Wiki EM	HotpotQA EM	Avg Tokens	LLM Calls
OLMo-2-7B	Best baseline	25.3	29.7	—	—
OLMo-2-7B	QuCo-RAG	**32.7 (+7.4)**	**35.3 (+5.6)**	87	1.84
Qwen2.5-32B	FS-RAG	35.9	—	—	—
Qwen2.5-32B	QuCo-RAG	50.0 (+14.1)	—	—	—
GPT-5	Web Search	48.3	19.8	—	—
GPT-5	QuCo-RAG	**59.7 (+11.4)**	**48.4 (+28.6)**	—	—

Lowest token consumption (87 avg) and fewest LLM calls (1.84) among all dynamic RAG methods.

Drawbacks - Need access to PT corpus, not as effective as PT corpuses expand (eg latest SOTA models have >50T pretraining tokens).
Triplet extraction model has overhead and might not be able to identify out of domain tokens.

Deep dive: ConfRAG

Idea: train the system to abstain or route to retrieval when confidence is low, instead of treating retrieval as a default.



Training intuition

Teach the model to say “I am not confident” on cases where internal knowledge is insufficient, then trigger retrieval only on those cases.

Why it works

Confidence is used as a routing signal. The system can save cost on easy cases while still improving factuality when the question is ambiguous or dynamic.

Drawback

External model guiding whether to search or not, good for smaller LLMs, sota models can be trained directly to answer or search instead.

Source: ConfRAG (Meta, 2025)

Triggering examples

The retrieval decision becomes more obvious as the task moves from lookup to comparison to research and multimodality.

Example	Trigger?	Why
Battery capacity	**Yes**	Simple question, but trim/year ambiguity and stale specs make retrieval useful.
Model 3 vs Lucid Air	**Yes**	Comparison requires normalized external facts for both entities.
Best EV in 2026	**Yes**	Needs broad, multi-source research; parametric memory is not enough.
Which year was Tesla Created	**No**	Can be answered through parametric knowledge

Triggering is easiest to explain on lookup questions, but it matters even more once the task becomes comparative, multi-hop, or multimodal.

RAG Triggering — Open Problems

What remains unsolved?

1. Threshold Sensitivity

Domain-specific tuning needed for binary retrieve/don't-retrieve decisions. No universal threshold works across domains.

2. Corpus Access Dependency

Corpus-grounded methods (QuCo-RAG) require access to trillion-token pre-training corpora — not always available.

3. Knowledge Currency

Static corpus statistics can't capture time-sensitive information that has changed since pre-training.

4. Evaluation Gap

Academic benchmarks (PopQA, SimpleQA) don't reflect enterprise retrieval workflows or production constraints.

Emerging Directions

Increasingly as models get smarter, single LLM decides whether to trigger RAG and then summarizer or answer directly. However this requires good prompt tuning and finetuning.

Practical takeaways — RAG triggering

A useful gate minimizes unsupported answers without making retrieval the default for every request.

Bias against missed retrieval

Optimize asymmetric error costs

Route on evidence need

Use task type, freshness, corpus support, and answerability—not raw model confidence alone.

Confidence is only one signal

Evaluate end to end

Report answer quality together with retrieval rate, latency, and cost. A router helps only when downstream evidence improves the answer.

Router + retriever + generator

02.01.02

Query Rewriting

What query should be formulated?

Query rewriting: from a user question to an evidence plan

User question

"Which has longer driving range - Tesla Model 3 or Lucid Air?"

Rewrite moves

- decompose into two entity-specific queries
- add trim/year filters
- normalize the attribute ("EPA range")

Better query set

1) "Tesla Model 3 EPA range 2026"
2) "Lucid Air EPA range 2026"
3) filters: trim/year, miles, official specs

What changed? which entities to look up, which attribute to compare, and which filters are required to keep the comparison fair.

Source: EAR (2023); Task-aware retrieval (2022)

Query Rewriting - Key Challenges

Key Challenges

Transforming user queries to better capture intent and bridge vocabulary gaps.

Vocabulary Mismatch

User queries and relevant documents often use different terminology.

Query Ambiguity

Single queries may have multiple valid interpretations requiring diverse reformulations.

Diversity vs. Relevance

More query variants can increase recall but also introduce redundancy and noise.

Latency Overhead

Multiple retrieval passes for rewritten queries add unacceptable latency in production.

Three Approaches

Multi-Query & Rank Fusion

Generate multiple query variants, refine and fuse results.

BlendFilter, EAR

Feedback-Driven Optimization

Extract LLM's knowledge, identify gaps, target retrieval at missing info.

ERRR, GroGU

RL-Aligned Expansion

Use retrieval/generation quality as reward signal for query expansion.

CoAugRetriever

CoAugRetriever — Problem

Query-Document Co-Augmentation via Reinforcement Learning

Liu, Li, Shi, Ding, Su, Zhou — arXiv 2025

The Challenge

Query expansion misses document-side gaps, document augmentation yields little benefit, and separate training creates misaligned representations.

The Gap

Prior methods optimize either queries or documents in isolation. No method jointly trains an LLM to augment both sides with a shared RL objective.

Key Idea

Train a single LLM (Qwen2.5-7B) via RL to jointly augment both sides using NDCG as reward for better representation alignment H(Q,D).

Impact: A 7B jointly-trained model surpasses 72B zero-shot augmentation across all datasets.

CoAugRetriever — Method

Bidirectional RL framework for query-document co-augmentation

Training Framework

1. Composite Sampling

Each batch = q queries × d_pos relevant × d_neg irrelevant

2. Within-Batch Reward

NDCG computed over sampled rollout combinations

3. Advantage: Centering-only (x - mean)

Differential scaling:
Queries: 1.0 | Relevant docs: 0.2 | Irrelevant docs: 0.1

Retrieval

Sparse

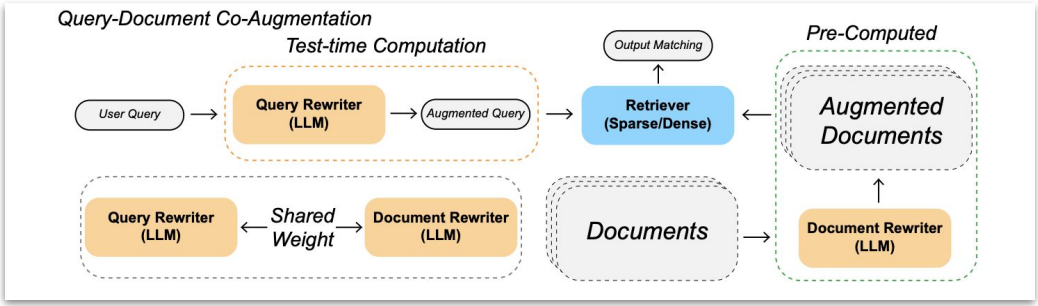
Word-level expansions and keywords

Dense

Sentence-level expansions concatenated with originals

Training Details

- Qwen2.5-7B, batch 512, max 300 steps, 8× A100 80GB
- Removed KL/Entropy loss — Reward-only learning



CoAugRetriever — Results

Joint RL outperforms separate training and 10× larger models

Setting	NFCorpus BM25	SciFact BM25	FiQA BM25	NFCorpus BGE	SciFact BGE
Base retriever	0.343	0.691	0.254	0.368	0.738
RL-Q (query only)	0.381	—	—	0.379	—
RL-D (doc only)	0.372	—	—	0.373	—
RL-QD (joint, ours)	**0.403**	**0.748**	**0.328**	**0.384**	**0.753**

NDCG@10 — Joint RL-QD consistently outperforms separately trained RL-Q + RL-D models.

Drawbacks

1) Model can overfit to a dataset on which it was trained with.
2) Expensive to run RL training on large datasets
3) Access to document set may not be present always.

Results show tendency to overfit on Representative Training Corpus

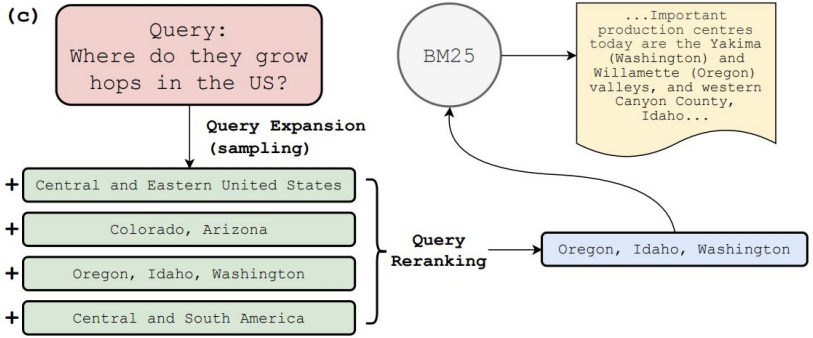
Settings		NFCorpus	SciFact	FiQA-2018
Base Retriever	BM25	0.343	0.691	0.254
	BGE-base-en-v1.5	0.368	0.738	0.391
Qwen2.5-7B	BM25	0.363	0.696	0.258
	BGE-base-en-v1.5	0.371	0.746	0.383
Ours-NF	BM25	**0.403**	0.741	0.272
	BGE-base-en-v1.5	**0.384**	0.743	0.371
Ours-SF	BM25	0.378	**0.748**	0.268
	BGE-base-en-v1.5	0.378	**0.753**	0.376
Ours-FQ	BM25	0.386	0.744	**0.328**
	BGE-base-en-v1.5	0.369	0.743	**0.395**

Deep dive: Expand, Rerank, Retrieve (EAR)

EAR separates two intuitions: generate diverse candidate rewrites, then estimate which rewrites are actually useful for retrieval.

Problem: How to Provide the best results to final LLM summarizer quickly. With BM25 instead of dense retrievers (DPR)

Key idea Train reranker to predict rank-orders of the gold passages when issuing expanded queries



2 Types of Rankers: 1) Retrieval Independent (RI) -> q + e_i

2) Retrieval Dependent (RD) -> Sees top1 passage for each query expansion before reranking

Train Ranker using Contrastive Loss.

$$\mathcal{L}_{\text{Rank}} = \sum_{\substack{i,j \in [1,n] \\ r_i < r_j}} \max(0, \mathcal{M}(q, e_i) - \mathcal{M}(q, e_j) + (r_j - r_i) \cdot \alpha)$$

Source: EAR (Meta, 2023)

EAR Results

EAR improves retrieval on both Natural Questions and TriviaQA: reranking expanded queries strengthens lexical retrieval, and fusion with DPR gives the best overall recall.

Model	Natural Questions			TriviaQA		
	Top-5	Top-20	Top-100	Top-5	Top-20	Top-100
Dense Retrieval						
DPR	68.3	80.1	86.1	72.7	80.2	84.8
Lexical Retrieval						
BM25	43.8	62.9	78.3	67.7	77.3	83.9
GAR	60.8	73.9	84.7	71.8	79.5	85.3
SEAL	61.3	76.2	86.3	-	-	-
Liu et al. (2022)	63.9	76.8	**86.7**	72.3	80.1	85.8
EAR-RI	63.2	76.4	85.9	73.4	80.8	85.9
EAR-RD	**69.3**	**78.6**	86.5	**77.6**	**82.1**	**86.4**
GAR *best query*	*81.9*	*88.1*	*92.0*	*85.0*	*88.1*	*90.1*
Fusion (Dense + Lexical) Retrieval						
BM25 + DPR	69.7	81.2	88.2	71.5	79.7	85.0
GAR + DPR	72.3	**83.1**	88.9	75.7	82.2	86.3
Liu et al. (2022) + DPR	72.7	83.0	89.1	76.1	82.5	86.4
EAR-RI + DPR	71.1	82.5	89.1	76.4	83.0	87.0
EAR-RD + DPR	**74.2**	**83.1**	**89.3**	**79.0**	**83.7**	**87.3**

Lexical gains

EAR-RD is the strongest lexical model in the table, reaching 69.3 NQ Top-5 and 77.6 TriviaQA Top-5.

Best overall

EAR-RD + DPR is best overall: 74.2 / 83.1 / 89.3 on NQ and 79.0 / 83.7 / 87.3 on TriviaQA.

Drawbacks and Takeaways

1) Query Expansion uses Parametric memory of model which may be wrong, eg Tesla trim 2021 vs Tesla trim 2026

2) Larger LLMs are more robust to noise and dense retrieval is smaller cost to pay for generality.

Rewriting playbook for the EV examples

Each example suggests a different rewrite move. Showing the move is often more intuitive than naming the method family.

Battery capacity — **Clarify trim / year**

“Tesla Model 3 Long Range battery capacity kWh 2026”

Range comparison — **Decompose into comparable lookups**

“Tesla Model 3 EPA range 2026”
“Lucid Air EPA range 2026”

Best EV in 2026 — **Generate a multi-query research plan**

price • range • charging • reliability • incentives • safety

EV image — **Entity first, factual query second**

identify make/model/trim → ask for EPA rated range

Query Rewriting — Open Problems

What remains unsolved?

1. Parametric Knowledge Error

Model may use parametric knowledge for expansion which may be wrong

2. Redundancy and Noise

Near-duplicate passages from multiple query variants waste context window and Noise can degrade generation quality.

Emerging Directions

Joint query-document augmentation via RL (CoAugRetriever) yields larger gains than optimizing either side alone.

Practical Directions

Start with Single query rewrite, expand to multi-query if needed to increase recall. Use Strong Summarizer model to handle noise which comes with increased recall.

02.01.03

Retrieval

How can a retrieval system be improved to provide the most relevant and accurate information?

Retrieval

Key Challenges

Retrieving relevant information — the critical RAG bottleneck. Retriever choice swings performance 17-34 points.

Precision-Recall Balance

Too many documents introduce noise, too few miss evidence. Inverted-U accuracy curve.

Query Type Diversity

Must handle single-hop, multi-hop, and domain-specific queries with different strategies.

Scalability

Sub-second latency across billions of documents while maintaining quality.

Adversarial Robustness

Corpus poisoning with as few as 10 passages achieves 98% attack success.

BadRAG: Identifying Vulnerabilities in Retrieval Augmented Generation of LLMS, Xue et.al

Two Approaches

Retrieval is moving toward richer models, selective execution, and multimodal evidence.

Dense Retrieval & Joint Pre-training

End-to-end retriever+LM training; documents as latent variables.

DRAMA, Atlas, GritLM

Foundation-Model Retrieval

Adapt LLM backbones for task-aware embedding and reranking.

Qwen3-Embedding, GritLM

Deep dive: Qwen3 embedding + reranking

A single foundation-model family serves both stages of a modern retrieval pipeline.

Task + query

“Compare official EPA range”
Model 3 vs Lucid Air

→

Qwen3 embedder

Causal LLM; [EOS] hidden state → vector

→

ANN retrieval

Cosine search over precomputed document vectors

→

Rerank

Instruction + query + doc
→ P(yes)

Training recipe

Synthetic Training Data

Stage 1 - ~150M weak-supervision pairs
Stage 2- ~12M high-quality pairs for supervised fine-tuning

Why it matters

The same backbone family supports task instructions, 32K inputs, configurable embedding dimensions, and pointwise reranking.

Source: Qwen3 Embedding (Zhang et al., 2025)

Embedding vs Reranking

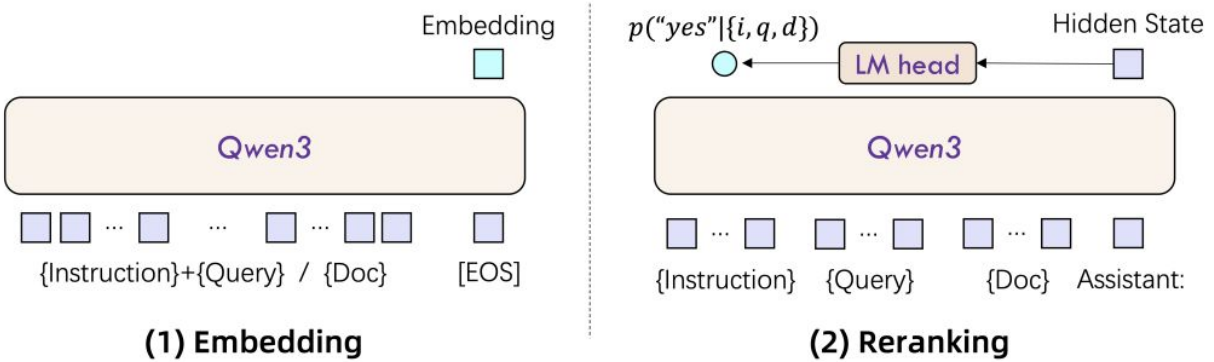


Figure 1: Model architecture of Qwen3-Embedding (left) and Qwen3-Reranker (right).

Qwen3 reranking saturates around 4B

All models rerank the same top-100 from Qwen3-Embedding-0.6B; scores are retrieval-specific.
MTEB-R - Massive Text Embedding Benchmark, MMTEB-R (Multilingual)

Model	Size	Mean (Task)	Mean (Type)	Bitext Mining	Class-ification	Clus-tering	Inst. Retrieval	Multilabel Class.	Pair Class.	Rerank	Retrieval	STS
Selected Open-Source Models												
NV-Embed-v2	7B	56.29	49.58	57.84	57.29	40.80	1.04	18.63	78.94	63.82	56.72	71.10
GritLM-7B	7B	60.92	53.74	70.53	61.83	49.75	3.45	22.77	79.94	63.78	58.31	73.33
BGE-M3	0.6B	59.56	52.18	79.11	60.35	40.88	-3.11	20.1	80.76	62.79	54.60	74.12
multilingual-e5-large-instruct	0.6B	63.22	55.08	80.13	64.94	50.75	-0.40	22.91	80.86	62.61	57.12	76.81
gte-Qwen2-1.5B-instruct	1.5B	59.45	52.69	62.51	58.32	52.05	0.74	24.02	81.58	62.58	60.78	71.61
gte-Qwen2-7b-Instruct	7B	62.51	55.93	73.92	61.55	52.77	4.94	25.48	85.13	65.55	60.08	73.98
Commercial APIs												
text-embedding-3-large	-	58.93	51.41	62.17	60.27	46.89	-2.68	22.03	79.17	63.89	59.27	71.68
Cohere-embed-multilingual-v3.0	-	61.12	53.23	70.50	62.95	46.89	-1.89	22.74	79.88	64.07	59.16	74.80
Gemini Embedding	-	68.37	59.59	79.28	71.82	54.59	5.18	**29.16**	83.63	65.58	67.71	79.40
Qwen3 Embedding Models												
Qwen3-Embedding-0.6B	0.6B	64.33	56.00	72.22	66.83	52.33	5.09	24.59	80.83	61.41	64.64	76.17
Qwen3-Embedding-4B	4B	69.45	60.86	79.36	72.33	57.15	**11.56**	26.77	85.05	65.08	69.60	80.86
Qwen3-Embedding-8B	8B	**70.58**	**61.69**	**80.89**	**74.00**	**57.65**	10.06	28.66	**86.40**	**65.63**	**70.88**	**81.08**

Deep dive: DRAMA retriever training

Large models can be used to generate diverse training signals that improve smaller dense retrievers.

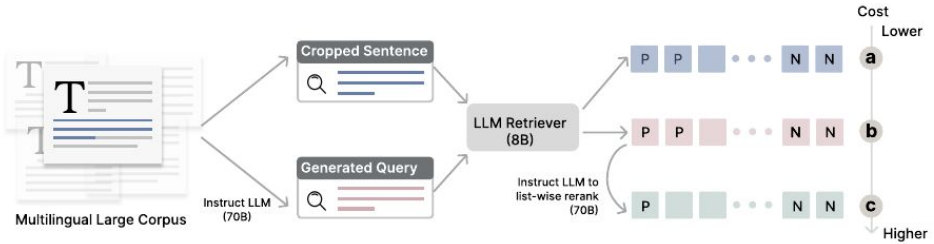


Figure 1: Methods to create data augmentation for smaller retriever with LLMs: (a) Using cropped sentences as queries, selecting the top-ranked documents from top-k retrieval as positives and the remaining as hard negatives.

M1 - Cropped Sentences (psuedo query) -> Retrieve docs -> topk retrieval, topn = +ve, [k-n,k] = -ve
M2 - Given document, generate synth queries using LLM -> similar retrieval as M1
M3 - M2 + ask LLM to rerank results

Core idea

Use LLM-generated diverse augmentations to create a richer training signal for a compact dense retriever.

Efficiency lever

Use large models to synthesize diverse supervision, then serve the smaller retriever. This keeps retrieval quality and serving cost decoupled.

Training signal

Retriever quality is not only about architecture. Data construction and training signals are major levers.

Source: DRAMA (Meta, 2025)

Deep dive: DRAMA retriever training

On BEIR, MIRACL, and multilingual MTEB subsets, DRAMA improves compact retrievers across scales: the 1B model is best overall in the table, while the 0.1B and 0.3B variants lead several smaller-size baselines.

Method	Non-Emb. Param.	Repre. Dim.	Contra. Pretrain.	Data Aug.	Multi. Lang.	English	Multilingual			
						BEIR (13)	MIRACL (18)	MTEB-FR (5)	MTEB-ZH (8)	MTEB-DE (4)
BM25	-	-	×	×	✓	43.7	38.5	-	-	-
Contriever	86M	768	✓	×	×	47.5	-	-	-	-
DRAGON	86M	768	×	✓	×	50.2	-	-	-	-
E5-v2-base	86M	768	✓	×	×	51.9	-	-	-	-
bge-base-en-v1.5	86M	768	✓	×	×	55.0	-	-	-	-
mE5-base	86M	768	✓	×	✓	50.2	60.1	45.4	61.6	49.2
mGTE-Dense	113M	768	✓	×	✓	54.3	62.1	50.6	**72.0**	49.1
ArcticEmb-v2-M	113M	768	✓	×	✓	56.9	59.2	53.7	55.7	55.0
DRAMA$_{0.1B}$	113M	768	×	✓	✓	56.9	70.4	52.1	61.7	55.1
E5-large-v2	303M	1024	✓	×	×	52.1	-	-	-	-
bge-large-en-v1.5	303M	1024	✓	×	×	56.1	-	-	-	-
mE5-large	303M	1024	✓	×	✓	52.9	65.4	47.7	63.7	50.4
mE5-Inst	303M	1024	✓	✓	✓	54.1	66.0	49.9	64.2	52.5
M3-BGE-Dense	303M	1024	✓	×	✓	50.0	69.2	48.6	65.6	50.4
ArcticEmb-v2-L	303M	1024	✓	×	✓	57.2	64.9	54.5	63.6	**55.9**
DRAMA$_{0.3B}$	265M	1024	×	✓	✓	58.0	71.4	54.8	63.0	55.6
Gecko	1B	768	✓	✓	✓	58.0	56.2	-	-	-
DRAMA$_{1B}$	1B	2048	×	✓	✓	**59.1**	**71.7**	**57.6**	63.7	56.2
DRAMA$_{1B}$ (768d)	1B	768	×	✓	✓	58.5	70.9	56.5	62.8	55.8
MistralE5	7B	4096	×	✓	✓	59.0	62.2	-	-	-

Small model win

DRAMA 0.1B ties the best BEIR score in the ~100M group (56.9) and leads MIRACL with 70.4.

Best overall shown

DRAMA 1B is best on BEIR, MIRACL, MTEB-FR, and MTEB-DE, while staying competitive on Chinese retrieval.

Takeaway: better training data can outweigh raw model size for retrieval.

Retrieval — Open Problems

What remains unsolved?

1. Retrieval Latency

Dense retrieval nearly doubles TTFT (495ms→965ms). 100M chunks degrade throughput 20×. Hybrid pre-filtering helps.

2. Adversarial Poisoning

90%+ attack success in black-box settings. As few as 10 poisoned passages compromise 98% of targeted queries.

3. Domain Adaptation

Wikipedia-pretrained retrievers fail on specialized domains (legal, medical, code). Joint training with async index refresh needed.

4. Lost-in-the-Middle

More passages often hurts — models attend to beginning and end, missing middle evidence. Reordering partially helps.

Emerging Directions and Practical Takeaways

1) 11B retrieval-augmented models matching 540B parametric-only — retrieval is more cost-effective than scaling parameters.

2) Strong LLMs -> Retriever Backbones

02.01.04

Post Processing

Refine the initial retrieval results before passing to the LLM

Post Processing

Key Challenges

Refining initial retrieval results before passing to the LLM for generation.

Scalability

Sophisticated reranking adds latency; LLM-based listwise rerankers are slow for large sets.

Compression vs. Info Loss

Aggressive pruning saves cost but risks losing multi-hop evidence chains.

Adversarial Robustness

Poisoned documents can bypass simple filters and contaminate the generator.

Three Approaches

Three approaches to post processing

Re-ranking

LLM-based listwise re-ranking; 335M models outperform 20× larger ones.

RankZephyr, InfoGain-RAG

Context Filtering

Evaluate document quality; drop if poor quality

Chain-of-Note

Compression & Chunking

Aggressive pruning (50-80%) can improve speed and accuracy.

SmartChunk

InfoGain-RAG: The Utility Gap

Wang, Liang, Shao et al. — EMNLP 2025

THE PROBLEM

Relevance ≠ Utility

Standard rerankers score by relevance. However, topically relevant documents can be misleading, redundant, or destabilizing for the generator.

THE GAP

Missing Measurement

No current reranking method directly measures a document's actual contribution to producing the correct answer.

Key Innovation: Document Information Gain (DIG)

$$DIG = P(answer|query,doc) - P(answer|query)$$

Train a 335M reranker on DIG labels — 20× smaller than GTE-7B, yet outperforms it.

InfoGain-RAG — Method

Utility-based reranking: score by how much documents help the generator

DIG COMPUTATION

$$DIG(d \mid x) = P(y \mid x, d) - P(y \mid x)$$

- Positive DIG: Helpful document
- Near-zero: Neutral / Irrelevant
- Negative: Misleading / Harmful

INFERENCE

Retrieval & Filtering Pipeline

Reranks top-100 Contriever results to top-4 docs. Filtering threshold 0.2, min 2 docs retained.

Sliding window smoothing (α=0.6) used for first k=3 tokens.

RERANKER TRAINING

Architecture: RoBERTa-large (335M) — 20× smaller than GTE-7B

Loss Function: Multi-task L = 0.75·L_CE + 0.25·L_Margin (Circle Loss)

Data: 110K TriviaQA queries, labeled by Qwen2.5-7B → 88K samples

✅ **DIG labels are LLM-agnostic — Qwen vs LLaMA yield consistent reranker performance**

InfoGain-RAG — Results

+17.9% EM over naive RAG with a 335M model outperforming 7B rerankers

Generator	Naive RAG	BGE-550M	GTE-7B	InfoGain (335M)
Qwen2.5-72B (TriviaQA)	59.9	70.6	73.4	76.3
Qwen2.5-72B (NaturalQA)	40.3	44.9	53.9	58.1
GPT-4o (NaturalQA)	37.5	41.6	53.1	57.2
DeepSeek-R1 (FM2)	77.1	78.9	80.3	83.8

EM accuracy — InfoGain-RAG (335M) outperforms proprietary GTE-7B (20× larger) across every model and dataset.

ABLATION

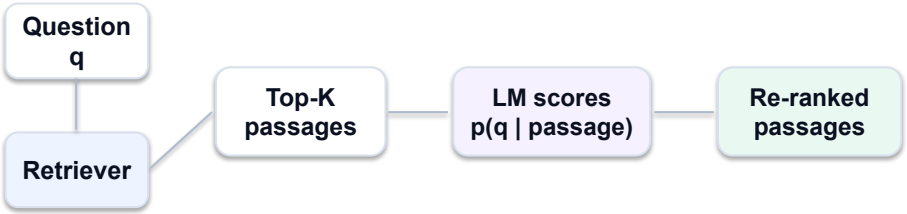
Loss	Qwen2.5-7B	Qwen2.5-72B
CE only	67.6	73.0
Margin only	68.2	71.4
Multi-task (ours)	71.8	76.8

Key Takeaways:

- Multi-task loss: +4.2 over CE-only
- Document filtering: +2.7 consistently
- 335M beats all 7B rerankers
- DIG labels transfer across LLMs

Deep dive: zero-shot question generation reranking

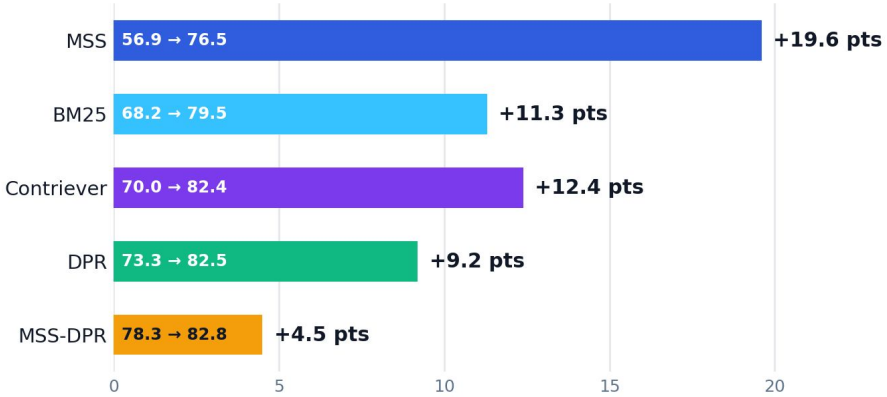
One way to judge a passage is to ask: if I saw this passage, could I generate the question back from it?



Score each candidate by how likely an LM is to generate the original question from the passage.

UPR improves top-20 retrieval accuracy

Absolute gain after zero-shot QG reranking; averages across QA datasets.



Mechanism

UPR scores each passage by the likelihood of generating the original question from that passage.

Result pattern

Consistent top-20 gains across sparse, dense, unsupervised, and supervised retrievers.

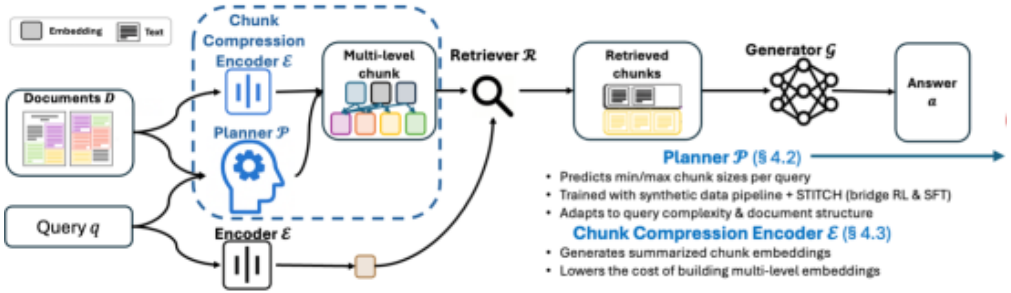
Takeaway

Post-processing can be a strong quality layer even when the first-stage retriever is unchanged.

Source: Improving Passage Retrieval with Zero-Shot Question Generation (2022)

SmartChunk: query-aware chunking

The query chooses the granularity; a lightweight compressor makes multi-level retrieval affordable.



Query + document metadata → predict [level_min, level_max]
Retrieve only those levels from a multi-scale hierarchy
Generate from fine text and compressed high-level chunks

Query-aware planner

Predicts the smallest and largest chunk levels needed from the query and document metadata.

Compression encoder

Aggregates fine-chunk embeddings into high-level semantic embeddings, avoiding repeated LLM summary generation.

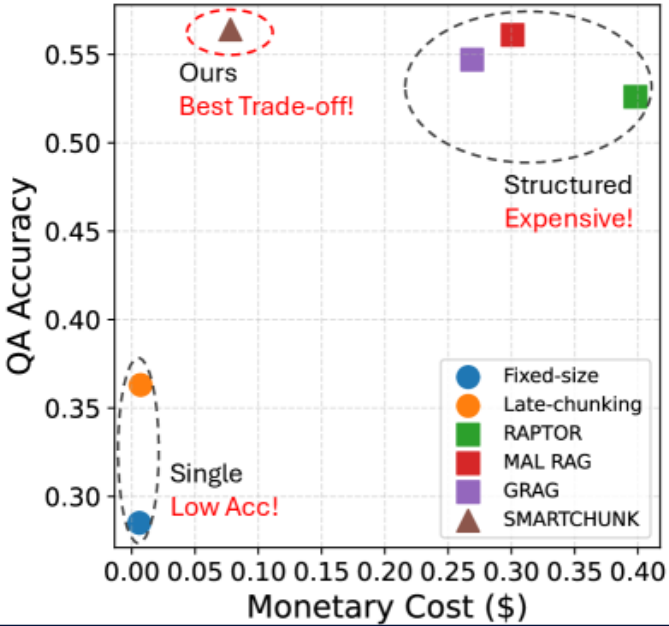
Why adapt granularity?

Fine chunks preserve details; larger chunks preserve broader context. The planner mixes them per query.

Source: SmartChunk Retrieval

SmartChunk selects the accuracy–cost frontier

SmartChunk matches MAL-RAG answer quality at 74% lower per-query monetary cost.



Comparable quality

QA accuracy: 0.564 vs 0.561 for MAL-RAG. Retrieval recall: 0.829 vs 0.842.

Lower recurring cost

$0.078 vs $0.301 per query (−74%); latency is also lower: 3.62s vs 4.14s.

Takeaway: retain structured-RAG quality without paying its recurring summarization cost.

Post-process.

Evidence shaping for the EV examples

Post-processing decides what evidence actually reaches the generator — and in what form.

Example	After post-processing
Battery capacity	Choose the official spec sheet or a high-confidence source; remove forum chatter and duplicate snippets.
Range comparison	Keep only comparable range facts; normalize units and trim/year; discard marketing copy that mixes trims.
EV buying question	Cluster evidence by dimension (price, range, charging, reliability, incentives) and keep the strongest item from each cluster.
Image + range	After the entity is identified, keep only evidence tied to that exact make/model/trim.

Post-processing is where “good retrieval” becomes usable evidence. The goal is not just top-k relevance, but a context set that is clean and fit for synthesis.

Post Processing — Open Problems

What remains unsolved?

1. Latency Overhead

LLM-based rerankers are slow for large candidate sets. Single-token reranking (FIRST) cuts 40%;

2. Compression Information Loss

Multi-hop evidence chains get lost during aggressive pruning. Current compression methods are optimized for single-hop QA.

3. Evaluation Generalization

Factoid QA benchmarks don't capture open-ended, multi-turn, or long-form generation quality in production settings.

4. Adversarial Vulnerability

Poisoned documents bypass standard filters. Specialized detection (GMTP) achieves >99% filtering but adds complexity.

Emerging Directions and Practical Takeaways

Proposition-level indexing (atomic self-contained facts) consistently outperforms passage-level

Finer Granularity extraction by pruning - training free pruning techniques can reduce 40-60% retrieved “fluff”

LLM-based reranking is very effective but is latency heavy.

02.01.05

Answer Generation

Ensure the LLM producing accurate and faithful answers when given noisy retrieved context

Answer Generation

Primary Goal

Producing accurate, faithful answers even when the retrieved context is noisy, irrelevant, contradictory, or incomplete.

Knowledge Conflicts

Deciding whether to trust retrieved context or parametric memory when they disagree.

Noise Sensitivity

Irrelevant or adversarial passages can corrupt generation, especially hard negatives.

Evidence Aggregation

Synthesizing coherent answers from multiple passages without losing details.

Efficiency

Computational expense of multi-document contexts vs. response latency.

Knowledge Conflict Resolution

Advanced decoding and attention techniques to reconcile model memory with external context at inference time.

Token-level RAG Switching

Dynamically selects between parametric and retrieved outputs per token based on divergence signals.

Examples: CoCoA, Tok-RAG

Entropy-Based Decoding

Detects conflict through generation uncertainty signals and adjusts decoding without retraining.

Example: CLeHe contrastive decoding

Credibility-aware Attention

Modifies attention weights to prioritize trustworthy passages while suppressing unreliable ones.

Example: OpenDecoder

CoCoA — Method

Confidence- and Context-Aware Adaptive Decoding for Knowledge Conflicts in LLMs

Khandelwal, Gupta, Agrawal — **EMNLP 2025**

Problem

When retrieved context contradicts the LLM's parametric memory, standard decoding blindly follows one source. Prior contrastive methods (CAD, AdaCAD) use JSD, which saturates on peaked distributions and can't distinguish high- vs. low-conflict tokens.

Key Idea

At each token, measure ***how much*** the context changes the model's mind, and blend accordingly.

Training-free — works at inference time on any LLM

CoCoA — Method

Confidence- and Context-Aware Adaptive Decoding for Knowledge Conflicts in LLMs
Khandelwal, Gupta, Agrawal — **EMNLP 2025**

Step	What It Computes	Intuition
① Rényi Divergence (α=0.5)	Gap between parametric vs. context distributions	*Detects whether a conflict exists (sensitive to tail events unlike JSD)*
② Entropy Gap	H(parametric) − H(contextual)	*Measures how much context reduces uncertainty*
③ Contextual Peakedness	Margin between top-2 context token probs	*Measures how confident the context signal is*
④ Adaptive Gate λ_t	$\sigma(z \cdot \log m_t + \log((1-s_t) / s_t))$	*High conflict + confident context → trust context*
⑤ Power Interpolation	$p\_ctx^{\lambda_t} \cdot p\_param^{1-\lambda_t}$	*Smooth per-token blend (not a hard switch)*

CoCoA - Result

Consistent gains across 4 models, 6 QA benchmarks, summarization, and long-form QA

Model	Greedy	ADACAD	**CoCoA**	Δ
Llama3-70B	66.5	68.7	**77.9**	+9.2
Llama3-8B	60.4	62.5	**71.7**	+9.2
Llama2-13B	56.3	59.0	**65.5**	+6.5
Mistral-7B	57.7	60.2	**64.9**	+4.7

Across NQ, NQ-SWAP, TriviaQA, PopQA, HotpotQA, TabMWP

Ablation (Llama3-8B, NQ-SWAP)

Full CoCoA 79.2

- Rényi → KL divergence 76.7 (−2.5)
- Remove divergence entirely 70.8 (−8.4)
- Remove entropy gap 68.9 (−10.3)
- Remove peakedness 65.2 (−14.0)
- Fixed λ=0.5 (no gating) 62.5 (−16.7)

Greedy baseline 47.8 (−31.4)

Key Takeaway: CoCoA excels on high-conflict inputs (NQ-SWAP: **80.8** vs AdaCAD's 62.8, **+18 pts**) while preserving low-conflict performance (**86.6** vs 86.4).

Noise-Resilient Generation

How can RAG systems produce accurate answers when retrieved context is noisy, irrelevant, or adversarial?

Isolate-then-Aggregate

Process each retrieved passage independently, then aggregate via voting or consensus — preventing cross-contamination from adversarial docs.

Examples: RobustRAG, IGP

Distractor-Aware Training

Fine-tune with intentionally noisy retrieval contexts, teaching models to extract signal from noise and cite evidence via chain-of-thought.

Example: RAFT, ATM

Preference Alignment

Use DPO/RPO to align generation toward context-faithful answers, teaching the model to prefer grounded responses over hallucinated ones.

Example: Context-DPO, RPO

RAFT — Problem

Adapting Language Models to Domain-Specific RAG
Zhang, Patil, Ganesh, Gonzalez — **arXiv 2024**

The Gap

Standard fine-tuning never sees retrieved documents. Standard RAG never trains with retrieval noise. Neither teaches the model what to ignore.

Training vs. Test Mismatch

Fine-tuning (DSF)	Q	→	A	trains without docs				
RAG at test time	Q	+	D*	+	D noise	→	A	never trained with noise
RAFT	Q	+	D*	+	D noise	→	CoT + A	✓trained with noise

Key Idea: Train under realistic retrieval conditions — mix oracle documents with distractors, forcing the model to learn what to ignore. Like studying for an open-book exam with irrelevant handouts mixed in.

RAFT — Method

Two ingredients: distractor-mixed training data + chain-of-thought answers

TRAINING DATA RECIPE

P% Q + D* + $D_2 \ldots D_k$ → CoT + A*

(1-P)% Q + $D_1 \ldots D_k$ → CoT + A*

Default: P ≈ 80%, k = 4 distractors.
Omitting oracle in (1-P)% forces the model to memorize domain knowledge as fallback.

CHAIN-OF-THOUGHT SUPERVISION

1. GPT-4 generates reasoning chains from Q + context + verified answer

2. Must cite evidence verbatim:

`##begin_quote##` … `##end_quote##`

3. Produces interpretable output:

```
##Reason: {reason} ##Answer: {answer}
```

RAFT — Results

LLaMA2-7B fine-tuned with RAFT outperforms GPT-3.5+RAG on 4 out of 5 benchmarks

+35%
HotpotQA
vs LLaMA2+RAG

+31%
HuggingFace Hub
vs DSF+RAG

+76%
Torch Hub
vs LLaMA2+RAG

METHOD	PUBMED	HOTPOTQA	HF HUB	TORCH HUB	TF HUB
LLaMA2 + RAG	58.8	0.03	26.4	8.6	43.1
DSF + RAG	71.6	4.4	42.6	82.8	60.3
GPT-3.5 + RAG	71.6	41.5	29.1	60.2	65.6
RAFT (7B)	**73.3**	**35.3**	**74.0**	**85.0**	**86.9**

RAFT uses LLaMA2-7B — 25× smaller than GPT-3.5, yet outperforms it on 4/5 datasets

Evidence Fusion and Compression

How should multiple retrieved passages be combined for generation — and how much can we throw away?

Multi-Passage Fusion

Encode passages independently, then fuse at decoder time — avoiding quadratic cross-attention over concatenated context.

Examples: FiD, REPLUG

Chunked Cross-Attention

Integrate retrieval into the transformer architecture itself — attending to retrieved chunks at intermediate layers rather than prepending to input.

Example: RETRO, In-Context RALM

Soft Compression

Compress retrieved evidence into compact representations — soft tokens, summary vectors, or pruned KV caches — before feeding to the generator.

Example: COCOM, OSCAR

COCOM — Problem

Context Embeddings for Efficient Answer Generation in RAG

Rau, Wang, Déjean, Clinchant — **arXiv 2024**

THE COST OF FULL-TEXT RAG

Standard RAG concatenates all retrieved passages as raw text into the prompt. With 5 passages of 128 tokens each, that's 640 extra tokens — causing quadratic attention cost and high latency.

Standard RAG

5 passages × 128 tokens

→ 640 tokens prepended to prompt

→ Full attention over everything

25,031 GFLOPs

per generated token

COCOM (128× compression)

5 passages × 128 tokens

→ 5 context embeddings

→ Attend over 5 vectors only

1,138 GFLOPs

per generated token — 22× cheaper

Key Idea: Compress each retrieved passage into a small number of continuous embedding vectors. The LLM generates answers by attending over these compact representations instead of raw text.

COCOM — Method

A single model compresses and generates; context embeddings are question-independent and pre-computable

PIPELINE

Compression

Each passage is mapped to k embedding vectors via the LLM's last hidden layer using special ⟨CTX⟩ tokens. Passages are compressed independently — enabling offline pre-computation.

4×	16×	128×
128 → 32 embeddings	128 → 8 embeddings	128 → 1 embedding

Training (Two Pre-training Tasks)

Auto-encoding: Recover original tokens from compressed embeddings — teaches faithful compression.

Language modeling: Compress prefix, generate continuation — teaches useful representation for generation.

$$L = -\Sigma \log P(x_t \mid \varepsilon, x_1, \ldots, x_{t-1})$$

✅ **Question-independent compression — index once, query many times**

COCOM — Results

Retains most of full-RAG accuracy at a fraction of the cost

22×
fewer GFLOPs
at 128× compression

94%
of full-RAG accuracy
at 4× compression

87%
of full-RAG accuracy
at 128× compression

METHOD	COMPRESSION	NQ	TRIVIAQA	HOTPOTQA	POPQA	AVG
LLM (no context)	—	35.9	70.8	26.4	19.9	41.6
xRAG (Mixtral-8×7B)	128×	26.5	74.4	23.9	31.8	37.2
COCOM	128×	**51.1**	**83.5**	**37.8**	**39.1**	**54.0**
COCOM	4×	**55.4**	**85.9**	**43.0**	**47.4**	**58.5**
RAG (no compression)	—	59.7	88.3	50.0	51.4	62.3

Exact Match (%) — all methods use Mistral-7B decoder with SPLADE-v3 retrieval, 5 passages

Answer Generation — Open Problems

What remains unsolved despite rapid progress?

1. Models Can't Say "I Don't Know"

GPT-4o achieved only **31% true-positive** on deflection (2024). Unreliable refusal persists across frontier models.

2. Relevance ≠ Generation Utility

High retrieval relevance can **negatively correlate** with answer quality. Conflicts destabilize generation.

3. Knowledge Conflict Fragility

GPT-4 preferred its own context **88% of the time** even when wrong. CoCoA and Context-DPO emerged to solve this.

4. Evaluation Misalignment

GPT-4o reaches only **60% on factuality-with-grounding**. Specialized RAG (QuCo-RAG) beats GPT-5 built-in search.

Emerging Directions

Utility-based scoring: Rank documents by generator help, not just similarity (e.g., InfoGain-RAG +17.9% EM).

Noise robustness improves with scale: gap shrinks **59.6% → 16.9% (Llama-2 → Llama-3)**, but abstention and grounding require targeted research.

Tutorial Outline



1. Introduction (40 min): Motivation, Factuality landscape, RAG at a glance

2. RAG Deepdive (110 min)
    - Modularized RAG (50 min)

    - Graph-based RAG (30 min)

    Break
    - Agentic RAG (30 min)

3. Advanced Topics (40 min)
    - Deep research (20 min)

    - Multi-modal RAG (20 min)

4. Conclusions and future directions (15 min)

02.02

Retrieval-Augmented Generation with Graphs GraphRAG for Personalization & Recommendation

Deep dive into GraphRAG pipeline, retrievers, organizers, generators, and how it folds into Agentic memory and harnesses for next-generation recommender systems.

Graph-based RAG pipelines in General

WHAT

Methods constructing knowledge graphs from text to leverage structural relations (triples, hierarchies, hypergraphs) for improved RAG reasoning.

WHY

Standard RAG misses entity relationships and fails at multi-hop reasoning, cross-doc synthesis, and connecting dispersed facts.

BASELINE

Chunk-based vector retrieval using cosine similarity and direct concatenation of text segments as context.

KEY CHALLENGES

- Multi-hop reasoning requires structural navigation that flat vectors cannot provide.
- KG construction is noisy/expensive, often introducing downstream hallucinations.
- Balancing precision vs. coverage: traversal can add noise or miss critical context.
- Scaling to large corpora while maintaining real-time inference speed.

https://papers.lunadong.com/area/rag#graph_based_rag_pipeline

Where this tutorial sits: the graph-based RAG landscape

Hybrid KG–Text retrieval · 35 papers

KG traversal & text retrieval reinforce each other.

ToG-2.0 · KERAG · EWEK-QA

Construction & indexing · 30 papers

Hypergraphs, hierarchies, schema-guided extraction.

HyperGraphRAG · Youtu-GraphRAG

Community & hierarchical · 20 papers

Retrieve coherent entity clusters in one step.

ArchRAG · CommunityKG-RAG

GNN & neural retrieval · 20 papers

GNN scoring over KG neighborhood: cheap & fast.

GNN-RAG · GNN-Ret · Amar

Temporal & event-aware · 15 papers

Time constraints & event order inside the graph.

TimeR4 · EventRAG · E2RAG

Benchmarks & evaluation · 20 papers

Multi-hop, temporal, incomplete-KG, multimodal.

CRAG · KGQAGen · BRINK · STaRK

State of the field in one line. KG and text are complementary — combining them beats either alone — yet most KG-RAG still relies on lookup, not reasoning (20–60% accuracy drop when answer links are removed). Everything in Part II maps onto these six sub-areas.

ROADMAP

A 30-minute tour, organized by the GraphRAG pipeline

Part I · Foundations (≈ 6 min)

- Motivation & running example (EV recommendation)
- From RAG → GraphRAG: three differences
- The 5-component master pipeline

Part II · Pipeline deep-dive (≈ 14 min)

- ① Query processor ② Retriever
- ③ Organizer ④ Generator
- ⑤ Graph data sources for recommendation

Part III · Canonical method (≈ 3 min)

- KERAG (EMNLP 2025) end-to-end
- Walk-through on the EV running example

Part IV · Bridges to emerging topics (≈ 5 min)

- Agent Memory · Tool Harnesses · Context Engineering
- Challenges, open problems, takeaways

Prerequisite assumption. Earlier talks in this RAG session have covered basic RAG (retriever / generator / chunking). This tutorial deliberately concentrates on what **graphs** add — and what they make harder.

Why personalization & recommendation need graphs

Running example (used throughout)

"I'm considering buying an electric vehicle in 2026. Which model is the best choice, and why?"

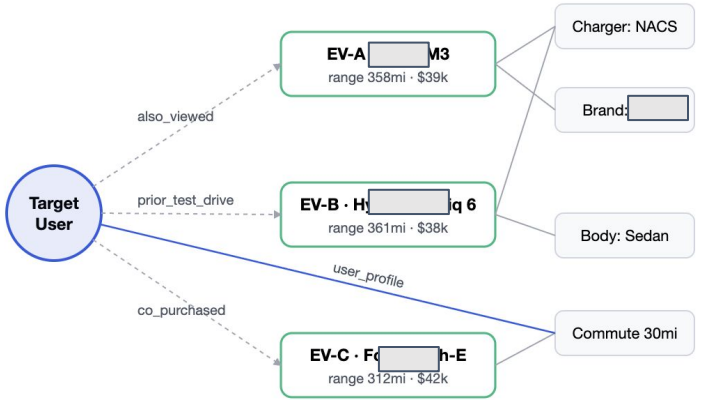
Expected answer: Structured report w/ 2~3 top EVs, compared on key dimensions, tailored to the user.

Naïve RAG fails because…

- Dense retrievers pull independent review snippets — **no** comparison structure.
- "Best" depends on user's prior interactions, budget, etc. — **none** of which the retriever sees.
- Multi-hop reasoning (range × charger × commute) — **invisible** to chunk-level retrieval.

Graphs already underpin recommendation

The EV recommendation graph connects:



Edges encode **structural** signals (co-view, co-purchase, brand affinity, commute geometry). Personalization quality depends on traversing them.

Han et al. 2025 §2.2, §6.1 · Wu et al. STaRK 2024 · Wang et al. CARES 2024

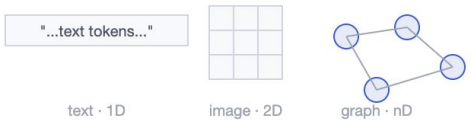
Three fundamental differences that force a dedicated pipeline

Δ1 · FORMAT

Unified vs. Diverse-Formatted

RAG: text = 1-D tokens; image = 2-D grid.

Graphs: nodes, edges, attributes, hierarchies, k-D geometry.



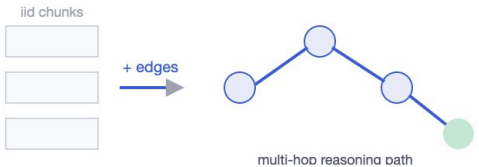
Δ2 · DEPENDENCE

Independent vs. Interdependent

RAG chunks are iid in a vector DB.

Graph nodes depend on neighbors

as edges are first-class.



Δ3 · TRANSFER

Domain Invariant vs. Domain-Specific

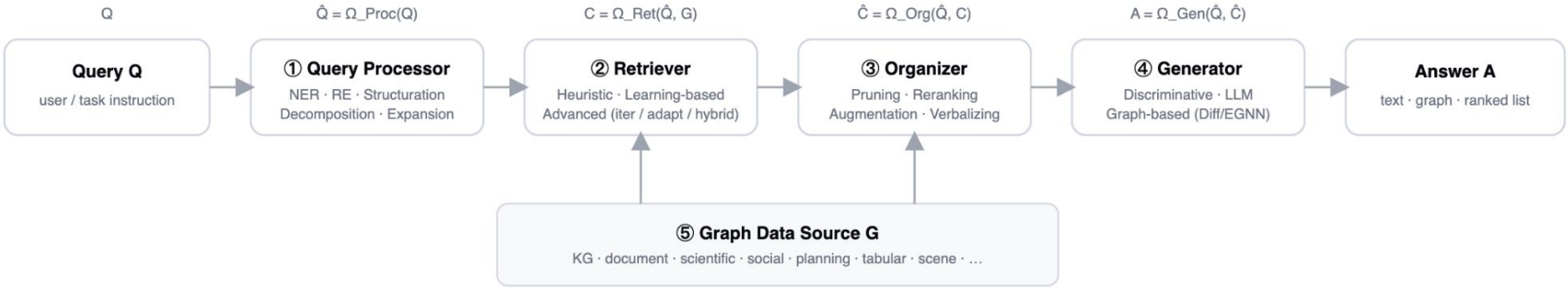
Text shares tokens; images share textures → scaling laws.

Graph relations domain-specific

Implication: Every component of RAG — query processor, retriever, organizer, generator, data source — needs a graph-aware redesign.

SLIDE 3 · THE MASTER PIPELINE

The five components of GraphRAG (reused as our backbone)



Why this matters. Every slide in Part II highlights one box of this pipeline, so you always know "where we are."

For recommendation, the same five boxes recur, but ⑤ is dominated by user–item, item–item, and metadata–attribute graphs (Slides 16–18).

Han et al. 2025, Fig. 3 (re-rendered)

Three relational rationales behind every recommender

PROXIMITY

"Birds of a feather"

Nodes close in the social/interaction graph share preferences. Used by collaborative filtering, GNN recsys, friend-of-friend retrieval.

McPherson 2001 · DeepWalk · LightGCN · CARES

ROLE

Structural twins

Nodes with similar local topology behave similarly — useful for cold-start ("act like a user with similar interaction shape").

Role2Vec · GraphWave · Wang et al. 2024

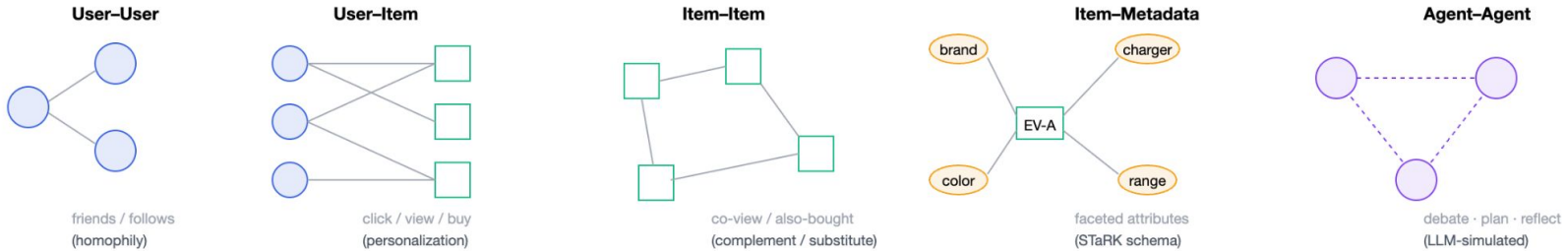
PERSONALIZATION

Idiosyncratic interactions

Each user's interaction (click, view, cart, purchase, review) is unique. GraphRAG retrieves per-user evidence instead of means.

ONCE (Liu '23) · STaRK (Wu '24)

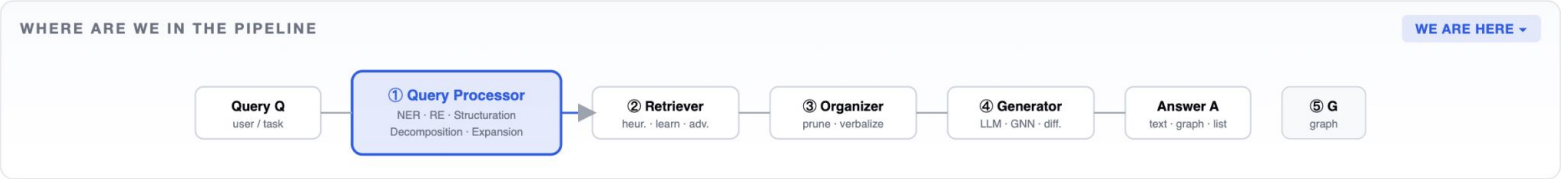
Graph types in the EV running example



PART II · SECTION 1

① QUERY PROCESSOR

From natural-language queries to graph-executable intent



Problem framing

Input: User query Q in NL (with optional context, persona, image).

Output: A processed query Q that exposes entities, relations, sub-queries, and structural intent.

Success: Entities/relations grounded to G · reasoning hops decomposed · negligible cascade error.

Han et al. 2025 §2.3 — NER · RE · Structuration · Decomposition · Expansion

Concrete failure — EV example. Plain BM25 sees only "*best electric vehicle 2026*", pulls a generic listicle, and loses three signals at once: the **target user** (no personalization), the implicit "**NACS charger**" constraint (no relation), and the 2-hop "**commute → range → model**" chain (no decomposition).

Why query processing differs from RAG

Entity Recog.: mention in text → grounded **node** in G

Relation Ext.: textual → matched **edge** in G

Structuration: SQL/SPARQL → **GQL** (Cypher, GraphQL)

Decomposition: independent subqueries → **logically linked**

Expansion: semantic synonyms → **relational** neighbors

Han et al. 2025, Table 2

All five techniques in one trace on the EV query

Q = "I'm considering buying an electric vehicle in 2026. Which model is the best choice, and why?"

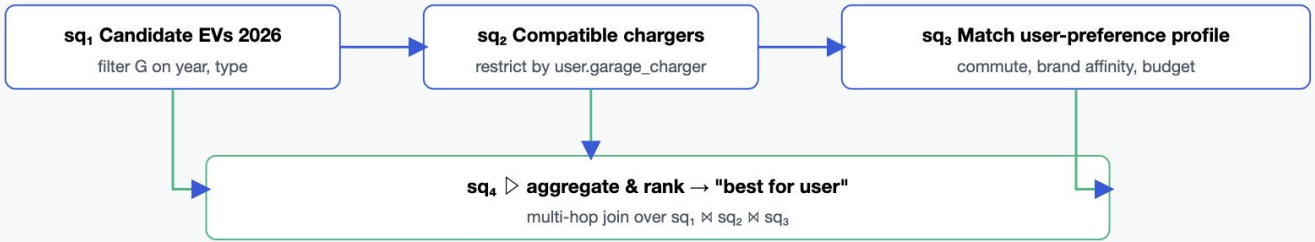
① NER
{user_2026, electric_vehicle, year:2026, model:* }
EntityLinker → KG/product nodes [395,464]

② Relation Extraction
"buy" → user[purchase]→item
"best for" → user[prefers]→item
edge-vector match in G [119,200,272]

③ Structuration (GQL / Cypher)
MATCH (u:User{id:$me}) →
[r:interacted] → (e:EV)
WHERE e.year >= 2026 RETURN e, r
StructGPT-style interface [181,186]

⑤ Query Expansion (relational)
+ neighbors of user_2026: prior_test_drive, zip_code, garage_charger=NACS
Xia et al. 2024 · Wang et al. 2023 (KEQING)

④ Query Decomposition (as a *linked* sub-query graph — Park et al. 2024)



The processed query $\mathcal{Q}$ is a (mini) *graph* over sub-queries — not a flat string.

② RETRIEVER

Find the right nodes, edges, paths, subgraphs — not just chunks

Problem framing

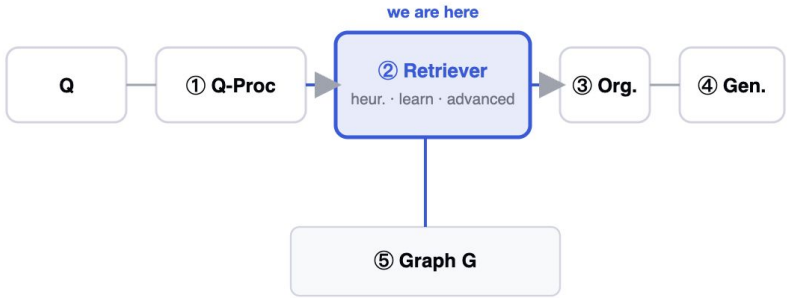
Input. Processed query Q, graph G = (V, E, X).

Output. C ⊂ G: a multi-format payload (nodes, edges, paths, subgraphs) satisfying four GraphRAG workflows: text→text, text→graph, graph→text, graph→graph.

Success criteria. Recall of answer-bearing facts · structural fidelity · compute/latency budget · cold-start handling.

Han et al. 2025 §2.4 · vs. RAG: BM25/TF-IDF + dense → can't reason on edges.

Concrete failure (EV). Dense retrieval brings 50 reviews about EVs, but never traverses user–[prior_test_drive]→T[illegible]3–[charger]→NACS. Result: the LLM cannot explain "why for me."



Four I/O workflows of GraphRAG retrievers

TEXT → TEXT | TEXT → GRAPH

GRAPH → TEXT | GRAPH → GRAPH

RAG mostly lives in box 1. GraphRAG inhabits all four.

Taxonomy of GraphRAG retrievers

Heuristic-based
rules · linear time · no training

- **Entity Linking** [443, 347, 428]
- **Relational Match** [119, 200]
- **Graph Traversal** BFS/DFS [492, 530, 308]
- **Graph Kernel** RW · WL [357, 403]
- **+ Domain Expertise**

Learning-based
embedding similarity in vector space $\mathbb{R}^d$

- **Shallow Embedding** DeepWalk · Node2Vec · Role2Vec
- **Deep Embedding (GNN)** GCN/GAT · query-cond. MP
- **Query-conditioned GNN retriever (GNN-RAG)**
  message passing per-query → top-k candidate
  entities → shortest path = context (Mavromatis 2024)
  Conditional GNN: only seeds initialized (Liu et al. 2024)

Advanced Strategies
complex queries · multi-aspect intent

- **Integrated (Neural+Symbolic)** [272, 443, 289]
- **Iterative** ToG · KGP · StructGPT
- **Adaptive (depth predict)** [134, 455]
- **Agent-based** KG-Agent · KnowledGPT
- **PageRank-guided**

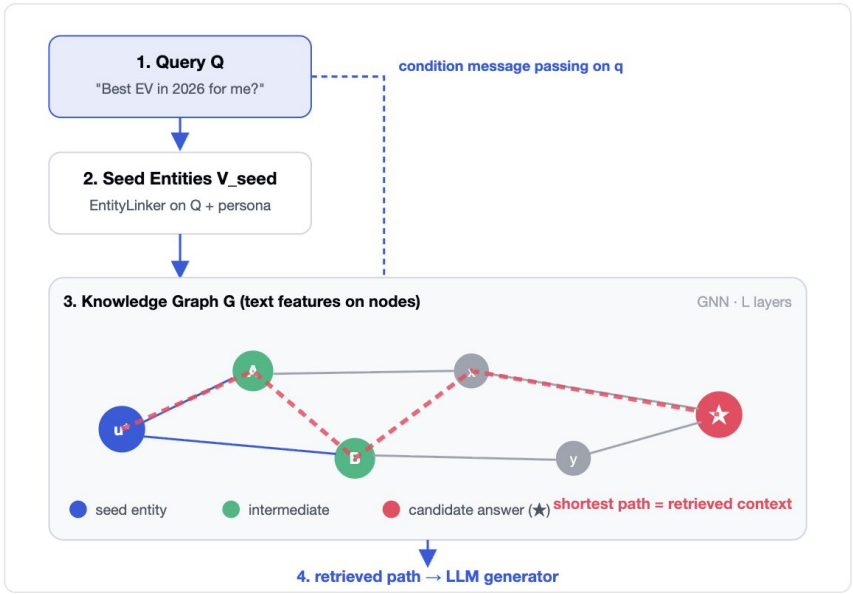
Choosing axis (recsys lens)

⬅ low latency / cold-start / sparse signal — complex reasoning / personalization / explanation ➡

entity-linking · ID-based — **GNN-RAG · collaborative-filter** — **Iterative / Agent / ToG / KEQING** — **KERAG (next deep dive)**

GNN-RAG — query-conditioned message passing for KG-grounded answers

Architecture



Why it works

$$x_i^{(l)} = \gamma_\theta\left(x_i^{(l-1)} \oplus \sum_{j \in N(i)} \varphi_\theta\left(x_i^{(l-1)}, x_j^{(l-1)}, e_{ij}, \boldsymbol{q} \right) \right)$$

message-passing aggregator γ & per-edge weighting φ, both *conditioned* on query **q**

- Every message is **conditioned on the query embedding** q → only query-relevant neighbors up-weighted.
- Trained as **node classification**: label=1 for gold answer entity.
- At inference: threshold answer probability; retrieve shortest path from seed to candidate as evidence.
- Lift over BM25 on WebQSP: **≈ +7 Hits@1** (Mavromatis 2024) REANO ablation shows query-attention carries most of the gain.

Why it's a fit for recommendation

- Same machinery is graph-based collaborative filtering — but conditioned on an NL query.
- Cold-start user? Initialize from persona text; GNN still propagates structurally.
- Inductive: handles newly-added EV models without retraining.

Mavromatis & Karypis 2024 · Liu et al. 2024 · REANO 2024

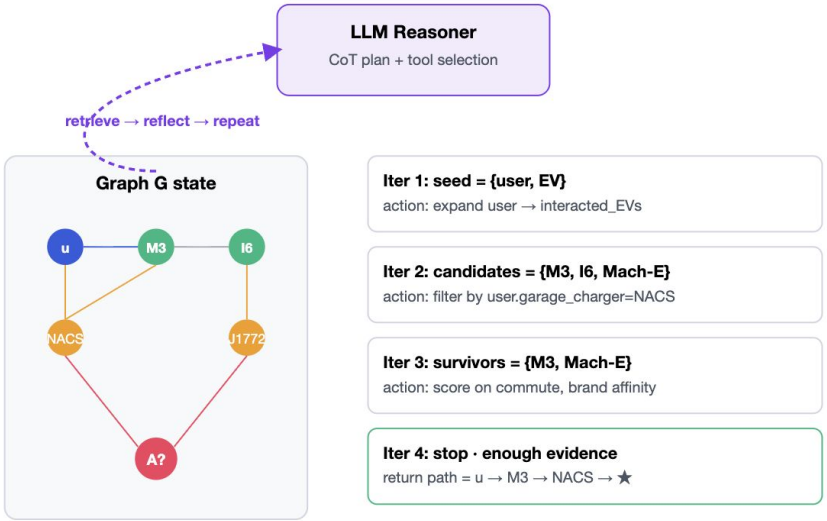
Iterative & agentic retrieval: think → retrieve → think again

Three exemplars at a glance

Method	Loop	Win on EV task
ToG (Sun '24)	seed → beam-expand until "enough"	walks user→test_drive→ model in 3 hops
KGP (Wang '24)	generate evidence ↔ pick neighbor	handles "why" with path
StructGPT (Jiang '23)	LLM calls graph interfaces	safe over million-edge KG
KG-Agent (Jiang '24)	fine-tuned LLM emits tool calls	auditable for compliance

For personalization. Iterative agents can interleave retrieval with user-clarifying questions — turning a one-shot rec into a conversational recommendation [Friedman 2023; He et al. 2024].

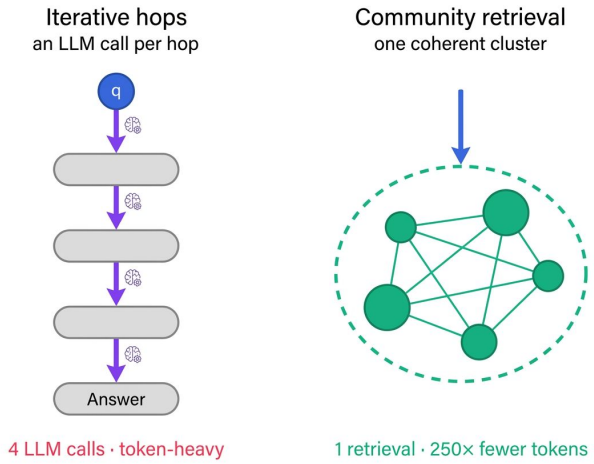
ToG-style loop on EV running example



Three more retriever families worth stealing

Family	Core idea	Evidence
Hybrid KG-Text (ToG-2.0, KERAG)	KG guides text retrieval; text prunes KG paths	SOTA on 6/7 knowledge-intensive datasets
Community / Hierarchical (ArchRAG)	Retrieve a coherent cluster in one step; hierarchical index	250x fewer tokens, +10% acc vs GraphRAG · **6–15×** faster
Neurobiological (HippoRAG 2)	Dense context + sparse entity index, mimicking hippocampal memory	+7.7 pts associativity vs standard RAG

Node-by-node vs. community retrieval



Recsys lens. Community retrieval = pull an item's whole owner-cluster (co-bought / co-viewed peers) in one hop; hierarchical indexing keeps that at ms latency over a 10^6-item catalog.
Note: KERAG (our canonical method #2) is itself a hybrid KG-text retriever.

③ ORGANIZER

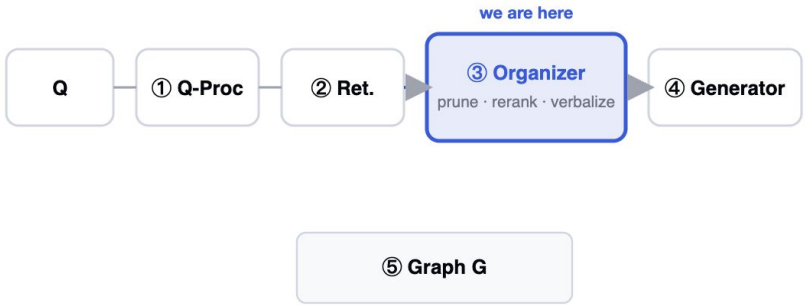
Refining raw subgraphs into LLM-digestible context

Problem framing

Input. Retrieved subgraph/paths/triples C + processed query Q.

Output. Refined context Ĉ — compact, ranked, well-typed, structure-aware text (or graph+text fusion).

Success criteria. Token budget respected, signal-to-noise ↑, long-context attention bias avoided ("lost in the middle"), structure preserved.



Concrete failure (EV). A 2-hop expansion easily yields 800+ triples. Without organization, the LLM *hallucinates* a 4th EV that doesn't exist or latches onto an irrelevant review.

Four organizer modes

GRAPH PRUNING · RERANKER · GRAPH AUGMENTATION · VERBALIZING

The four organizer techniques — and how they show up in recsys

PRUNING

Trim the subgraph

- **Semantic**: QA-GNN, KnowledgeNavigator
- **Syntactic**: dep-parse filter (Su ’23)
- **Structural**: PageRank in RoK
- **Dynamic**: JointLK, DHLK
- **Prize-collecting Steiner:** G-Retriever

RERANKER

Reorder for LLM bias

- Cross-encoder triple rerank (Li ’24)
- GNN-based passage rerank (Yu ’24)
- Time-ordered paths (Liao ’24, GenTKG)
- MMR diversity (KG-Rank)

AUGMENTATION

Enrich the graph

- **Structure**: noun-phrase nodes (GraphQA), query-node connector
- **Feature**: ONCE summarizes/profiles, LLM-Rec / KAR rewrite features
- Diffusion-based edge add (Tang ’24)

VERBALIZING

Graph → text for LLM

- **Linear**: tuple/template (Guo ’23)
- **Model**: graph-to-text transformer (Koncel-K. ’19, Ribeiro ’21)
- **Summarization**: EFSum, CoTKR

Verbalization template (Guo et al. 2024)

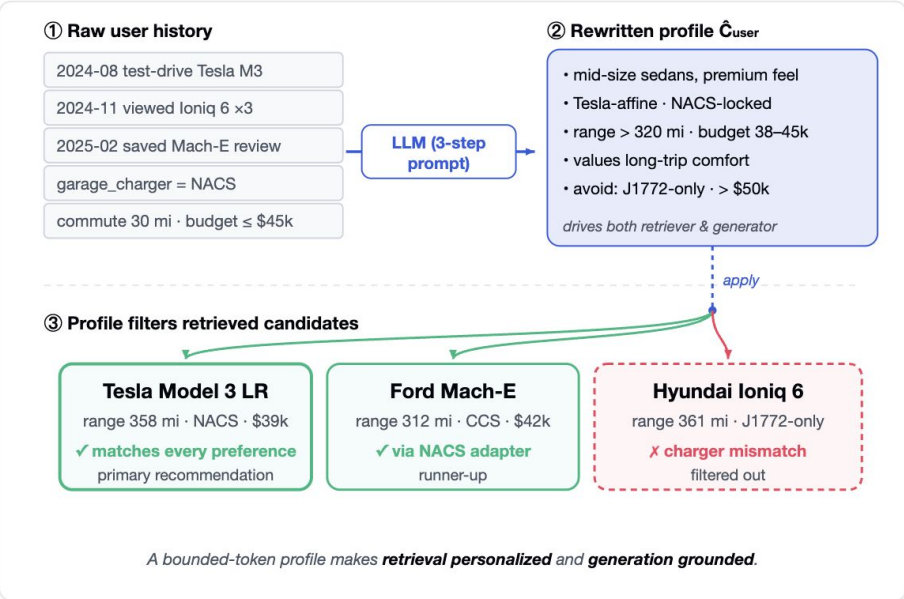
(h, r, t) → "The {r} of {h} is/are: {t}"

Choice of template materially changes downstream accuracy — Wang ’24 (KEQING) shows LLM-verbalizers beat template for ChatGPT, but the reverse holds for LLaMA-13B.

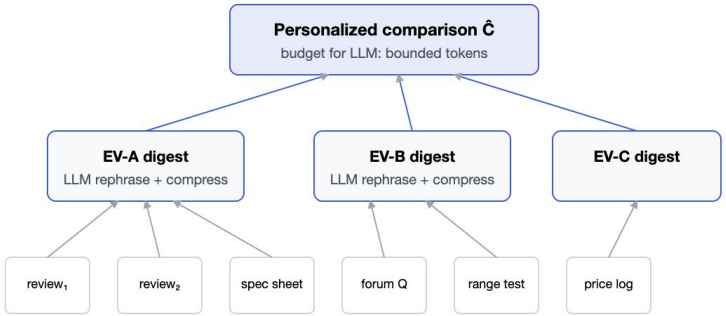
Position-bias fix. Studies show LLM accuracy sags for middle-of-prompt content. **Reranking + pruning are not optional** in multi-hop GraphRAG — they translate into 8–15 pt accuracy on multi-hop QA benchmarks (HotpotQA, 2WikiMultiHop).

Profile summarization & hierarchical aggregation

Profile summarization (Wang & Lim '24; KAR)



Hierarchical aggregation (Du et al. 2024, CARES)



Effect. Receptive field grows linearly in *levels* while prompt size stays bounded — feasible for product graphs with $>10^6$ neighbors.

④ GENERATOR

Producing the answer — text, ranked items, or a new graph

Problem framing

Input. Refined context $\hat{C}$ + processed query Q.

Output. Answer A — structured report (EV), ranked list (recsys), entity (KG-QA), or generated graph (molecule design).

Success criteria. Faithfulness to retrieved facts · structural fidelity (path / subgraph preserved) · controllability of personalization.

Concrete failure (EV). LLM generates a beautiful answer recommending an EV that doesn't exist in 2026. Grounding to retrieved item nodes is essential — the recsys "candidate-set" trick.

Three generator families

- **Discriminative** GCN/GAT/HGNN – classification/ regression
- **LLM-based** verbalize / embed-fuse / pos-fuse
- **Graph-based** diffusion · EGNN · RetMol

How the graph actually enters the LLM

Three integration patterns

A. Verbalize

subgraph → "X is_a Y; …"

+ simple · loses geometry · token-heavy

B. Embedding fusion

GNN h → proj → [h] ⊕ text tok.

G-Retriever, LLaGA, LINKGPT, GraphLLM

C. Positional / structural encoding fusion

Add graph-position tokens to LLM input (GIMLET, LPFormer pairwise enc.)

Preserves **which node is which** — critical for path-based reasoning answers.

Often combined with LoRA fine-tuning on graph-task instructions.

D. Graph-based generator

RetMol / IRDIF: **diffusion** + EGNN with SE(3)-equivariance for molecule generation

QA-GNN / GraphQA: GNN classifier produces the answer entity

Recsys: GNN-based **scoring head** on top of retrieved candidate set

G-Retriever (He et al. 2024) — a recipe for recsys

1. Score node/edge relevance to query.
2. Solve Prize-Collecting Steiner Tree to get a small, connected, on-budget subgraph.
3. Encode with GNN; project to LLM token space; prepend to prompt.
4. LLM (frozen) generates answer, can cite nodes by id.

Why we like it for personalization:

- **Explainable**: the Steiner tree IS the explanation.
- **Bounded**: budget = prompt cost.
- **Pluggable**: keep your favorite recommender as relevance scorer.
- **Compositional**: easy to wire into agent loops.

Grounding trick. Constrain the LLM to only recommend EVs whose ids appear in the retrieved tree — zero hallucinated SKUs.

⑤ GRAPH DATA SOURCES

The "fuel" — designing the right graph is half the job

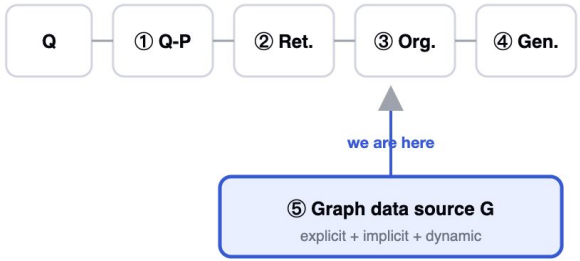
Problem framing

Input. Raw logs, catalogs, documents, profiles, telemetry.

Output. A graph $G=(V,E,X,R)$ with chosen node/edge granularity, attributes, and update cadence.

Success criteria. Aligns with dominant relational rationale (proximity/role/personalization) and serves all downstream tasks within budget.

Concrete failure (EV). Building a graph that only encodes also_bought misses the charger constraint. A graph that encodes everything blows up retrieval latency. Design is a choice.



Construction styles

- **Explicit**: KG triples, citations, hyperlinks, purchases
- **Implicit:** co-occurrence, k-NN over embeddings, dep-parse
- LLM-based: entity+relation extraction (MS GraphRAG)

Five interaction graphs that fuel personalized GraphRAG

Graph type	Edge meaning	Why curation matters	Representative work
User–User	follow, friend, comment	natural; homophily; can leak privacy	Wang '24, He '24
User–Item (bipartite)	click, view, cart, purchase, rate	multi-edge types encode intent strength	STaRK, Amazon-Review
Item–Item	also-viewed, also-bought, complement	manually extracted; sparsity at long tail	CARES (Wang '24)
Item–Metadata	has_brand, has_category, has_attribute	schema-driven, query-decomp-friendly	STaRK-Amazon, ONCE
Agent–Agent	LLM-simulated debate / collaboration	synthetic; fresh frontier	Generative Agents (Park '23)

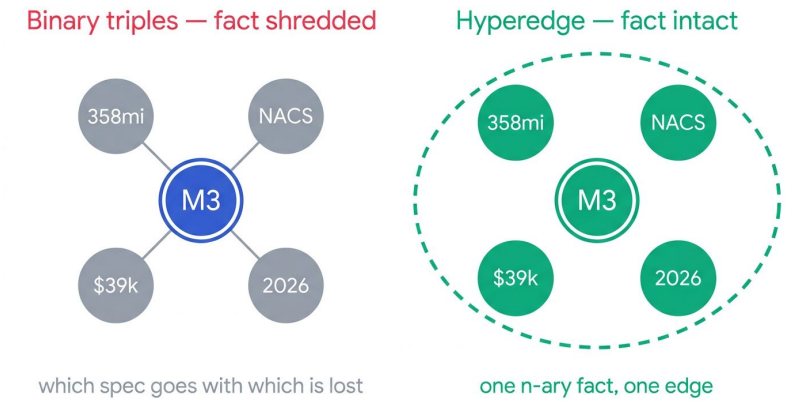
Hybrid by default. Production recommenders fuse user–item (collab. filtering), item–metadata (cold-start), and item–item (complement/substitute).

Dynamic graphs. Real catalogs update hourly. Open challenge: **maintain** embeddings and traversal indices incrementally without breaking explainability.

EV product-graph schema (toy):

How you build the graph decides what it can answer

Hypergraphs preserve n-ary facts



HyperGraphRAG keeps (M3, 358mi, NACS, $39k, 2026) as a single hyperedge instead of shredding it into triples — **+5.9 F1** over binary GraphRAG across five domains.

Construction quality is the real bottleneck

- **The problem.** LLM-extracted entities & relations are noisy and expensive; hallucinated nodes propagate downstream through the whole pipeline.
- **Schema-guided extraction** (Youtu-GraphRAG) constrains entity types so spurious nodes never enter — reports ~90% token-cost savings via agentic, schema-aware construction.
- **Post-generation verification** (DO-RAG) cross-checks the answer against graph evidence, reaching ~1.0 contextual recall.
- **KG→text linearization matters:** template & edge-direction choices alone swing accuracy up to 10 points.

Recsys lens. A product spec is inherently n-ary; a fixed schema (Item · Spec · Charger · Price · Year) keeps the catalog KG clean, makes the EV recommendation faithfully grounded, and bounds construction noise.

Benchmarks & toolkits — what to grab on Monday morning

Recommendation benchmarks

- **STaRK-Amazon** — semi-structured QA on products
- **Amazon-Review** ('14, '18) — with also_viewed/bought
- **MovieLens-100k/1M** — w/ demographics
- **Netflix Prize** — crawled multi-modal
- **Diginetica / Yoochoose** — session graphs
- **Yelp** — biz + reviews + check-ins

Knowledge / Doc graph corpora

- **Freebase / WikiData** — general KGs
- **ConceptNet** — commonsense
- **HotpotQA / 2WikiMultiHop** — multi-hop QA
- **WebQSP, CWQ** — KG-QA standards
- **STaRK-Prime** — biomedical KG
- **HuggingGPT / RestBench** — tool graphs

Toolkits

- **microsoft/graphrag** — LLM-built KG + community
- **LangChain / LlamaIndex** — graph stores + retrievers
- **RecBole** — 100+ recsys algorithms, 43 datasets
- **PyG / DGL** — GNN training
- **PyTorch Frame, RelBench** — tabular graphs
- **X-Developer / Reddit API** — live social graphs

For the EV case study: STaRK-Amazon (schema-rich) ⊕ custom EV catalog, microsoft/graphrag for community indexing, RecBole for candidate generator, G-Retriever for LLM head.

Evaluation pitfalls. Recall@k and Hits@k under-credit GraphRAG. Also use: path-faithfulness, structural recall, and personalization lift (A/B between user-conditioned and population-default).

What the benchmarks actually say about GraphRAG

What works

- **KG + text > either alone** — consistent across benchmarks.
- **Small beats big:** 7B GNN-RAG matches GPT-4, 9x fewer tokens.
- **Communities cut cost:** 200-250x token reduction.
- **Hypergraphs:** +5-7 F1 on complex queries.

Where it still breaks

- **Lookup != reasoning** — BRINK: 20-60% drop when links removed.
- **Broken yardsticks:** KGQA avg only 57% factual accuracy.
- **Hard ceiling:** CRAG best ~63%; KDD-Cup top ~36%.
- **Retriever > LLM:** embedder swap = ~17.5 pt swing.

WebQSP — KG-QA leaderboard

Method	Score / note	Year
RPO-RAG (Llama-3.1-8B)	89.9% Hits@1	2026
GNN-RAG (7B)	matches GPT-4, 9x fewer tok	2025
Think-on-Graph 2.0	SOTA on 6/7 benchmarks	2024

End-to-end difficulty (CRAG)

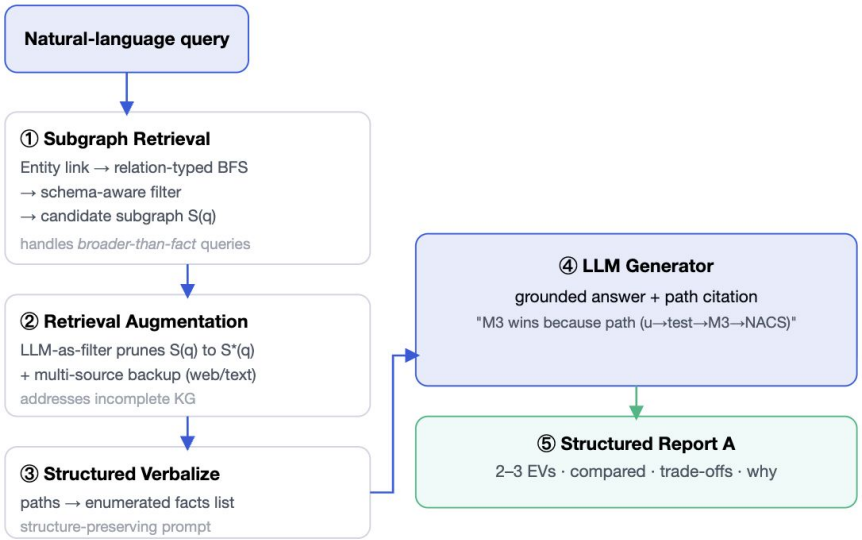
System	Result
SOTA RAG systems	63% truthfulness (best case)
KDD Cup 24 top systems	~36% task completion
KERAG (our method #2)	52.9% CRAG truthfulness

Caution for recsys. Recall@k flatters GraphRAG. Re-test with answer-links removed and report path-faithfulness.

BRINK · CRAG · WebQSP leaderboard · RPO-RAG (26) · GNN-RAG (25)

KERAG — Knowledge-Enhanced RAG (EMNLP 2025)

Architecture (Sun et al. 2025)



Why KERAG matters for personalization

- **Beyond fact lookup.** Targets broader questions where retrieved subgraphs — not single facts — are the unit of evidence.
- **LLM-as-filter.** Treats the LLM as a learned ranker over the retrieved subgraph.
- **Robustness** to KG incompleteness via multi-source backup.
- **Strong empirical wins** on advanced KGQA benchmarks: outperforms GraphRAG/HyKGE/GoT and even GPT-4o on broader-than-fact queries.

Reading recommendation: Sun, Sun, Xu, Yang, Dong, Tang, Chen. "KERAG." EMNLP 2025. arXiv:2509.04716

Take-away. For our EV task, KERAG provides the missing piece: a principled **retrieve-filter-verbalize** loop that scales subgraph reasoning to messy real-world catalogs.

HippoRAG — Neurobiologically inspired long-term memory for LLMs

HippoRAG acts as long-term memory system by creating a knowledge graph (hippocampal index) and using Personalized PageRank to simulate pattern completion, enabling single-step multi-hop retrieval.

Why HippoRAG matters

- Important real-world tasks like scientific review require **connecting information** across documents.
- Current iterative retrieval methods (like IRCoT) are **expensive and slow** due to multiple LLM calls.
- **LLMs lack** a continuously updating long-term memory system capable of efficient knowledge integration.

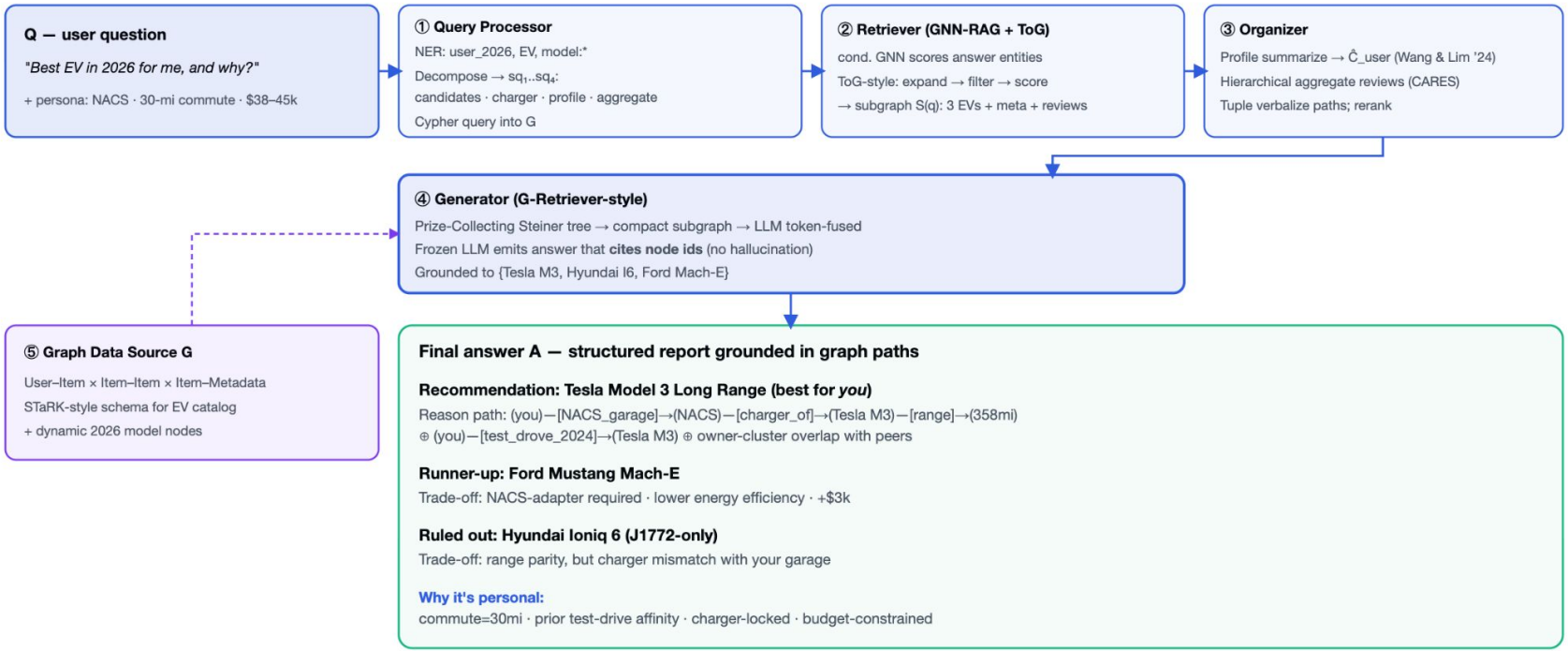
Neurobiologically-inspired Graph Retrieval

- Mimics the hippocampal indexing theory: The LLM (neocortex) processes passages into a Knowledge Graph (index).
- Uses Personalized PageRank (PPR) on this graph to simulate 'pattern completion,' activating relevant subgraphs based on query concepts.
- Achieves multi-hop reasoning in a single retrieval step by exploring graph paths rather than iterative LLM prompting.

(C
Yu Su, NeurIPS (2024)

SLIDE 20 · RUNNING EXAMPLE · END-TO-END

All 25 minutes condensed: one pipeline solving the EV question



GraphRAG ↔ Agent Memory

Modern agents (AutoGen, MemGPT, Generative Agents, A-MEM, HippoRAG) store experience over time. The graph view of memory has emerged as the dominant abstraction.

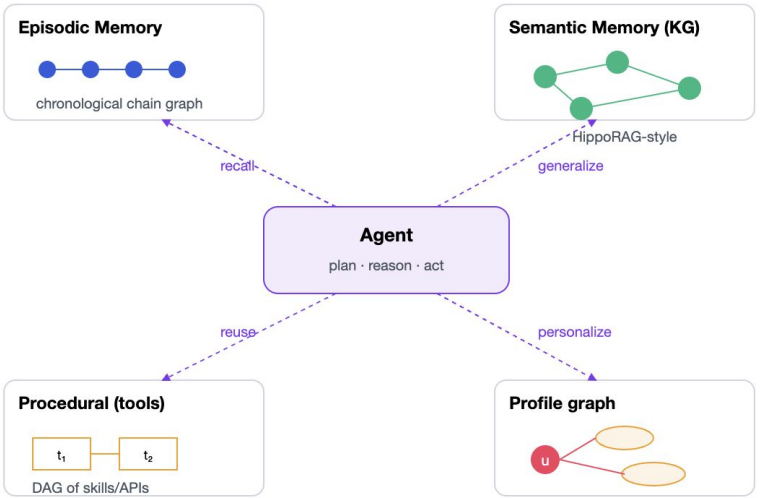
Memory as a typed graph

Memory layer	Graph realization	Why GraphRAG?
Episodic	event nodes + temporal edges	"recall what user told you 3 weeks ago"
Semantic	concept KG (HippoRAG)	generalize across episodes
Procedural	tool / skill graph	reuse plans
Profile	user attribute graph	personalization “north star”

Recsys connection. A long-running EV assistant remembers test-drives, charger constraints, budget evolution; GraphRAG over agent memory keeps recommendations personally consistent.

Agent Memory Architecture

HippoRAG (Gutiérrez '24) · A-MEM (Xu '25) · MemGPT (Packer '23) · Generative Agents (Park '23)



GraphRAG ↔ Harnesses & Tool Graphs

A modern "agent harness" is a graph problem

HuggingGPT, ControlLLM, MetaGPT, ToolBench, RestBench — every harness implicitly maintains a tool/API graph with dependency edges.

Dependency types:

- **Resource dependency**— output type of tool A = input type of tool B
- **Temporal dependency** — search must precede summarize
- **Causal** — only execute booking once payment confirmed
- **Analogy** — reuse old plans for new goals

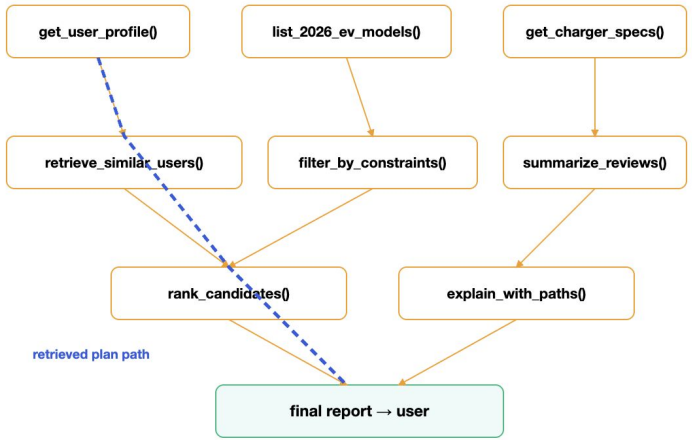
Retrieval techniques are the same as KG-QA:

- Tree-search retrieval (A* — Zhuang ’24)
- Monte-Carlo Tree Search (Gao ’24)
- Thought-Propagation Retrieval — analogical reuse (Yu ’24)
- Iterative LLM-as-router (HuggingGPT)

Recsys angle. A personalized harness can choose different tools per user — your EV assistant invokes get_charger_price() for Tesla owners but get_adapter_kit() for CCS owners.

Tool-graph retrieval for EV question

The retrieved plan path through the tool graph mirrors KG reasoning paths.



GraphRAG ↔ Context Engineering

From "prompt engineering" → "context engineering"

The frontier work (Anthropic, OpenAI, DeepMind 2025) reframes the problem as **filling a finite context window with the right structured evidence**, not crafting clever prose

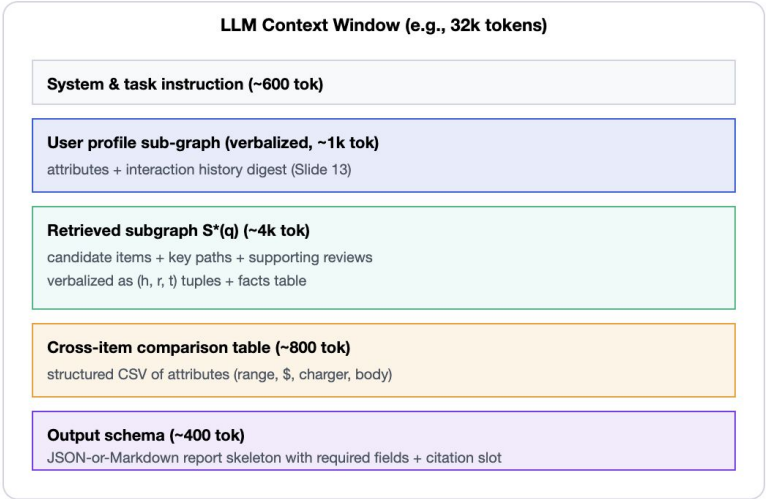
GraphRAG is the most principled context-engineering machinery:

- **Structure-aware compression.** Hierarchical aggregation ≡ context-budget allocator.
- **Typed evidence.** Subgraph slots (user, item, attribute, path) — separate channels resist confusion.
- **Citation by id.** Steiner-tree generators attach node ids → context attribution.
- **Adaptive depth.** Predict-then-traverse (KnowledgeNavigator) ≡ adaptive context budget.

Implication. Context engineering for recsys = graph engineering. Investing in the right graph schema pays off twice — at retrieval and at prompt time.

Anatomy of a GraphRAG-engineered context window

Each slot is filled by a different GraphRAG sub-pipeline. The total context is a typed, structured payload, not a soup of chunks.



Where RAG is heading — four currents to watch

① Agentic & RL-optimized

RL agents discover retrieval strategies that beat hand-designed heuristics. Process supervision is 18× more data-efficient; 7-8B models match GPT-4o on multi-hop.

ReSearch · ReasonRAG · CoRAG +36.5%

② Adaptive retrieval — when to look

Decide retrieval from corpus statistics, not raw logits. QuCo-RAG beats GPT-5 web search by 5–9 EM; ConfRAG cuts hallucination 20-40% → <5% and −30% lookups.

QuCo-RAG · ConfRAG · Adaptive-RAG

③ Multimodal graph RAG

Retrieve page images directly — skip lossy text extraction. VisRAG +20–40%; LILaC layered component graph +15.75% MRR@10; MRAMG first benchmark.

VisRAG · LILaC · MRAMG

④ Security & robustness

10 poisoned passages → 98% attack success (BadRAG); a single emoticon hijacks retrieval (EmoRAG, F1 > 0.92). Counter-intuitively, larger models are more vulnerable.

BadRAG · EmoRAG · WARD watermark

Why this matters for graph personalization. Each current is a lever on our pipeline: agentic/RL learns the retrieve-reason loop rather than hand-coding it; adaptive retrieval decides when the user graph is even needed; multimodal graphs fold product images into the catalog; and security becomes first-class once recommendations are grounded in a poisonable graph.

SLIDE 24 · WHERE RESEARCH MUST GO

Challenges & open problems — your KDD paper, perhaps

Graph construction

- Granularity choice (word↔entity↔doc)
- Multi-modal nodes
- **Dynamic graphs** & incremental indexing
- LLM-built KGs: validating quality

Retriever

- Neural↔symbolic harmonization
- Knowledge-conflict reconciliation
- Accuracy/diversity/novelty trade-offs
- Adaptive depth

Organizer

- Completeness↔conciseness
- Optimal verbalization format
- Multi-modal alignment
- Augmentation w/o noise

Generator

- Native graph-token LLMs
- Structural encodings
- Bound hallucination by candidate set
- Faithfulness audits

GraphRAG as a system

- Scalability (10^9 edges, ms latency)
- RLHF + retrieval co-training
- Multi-hop uncertainty calibration
- Privacy in homophilous graphs
- Robustness to structural attacks

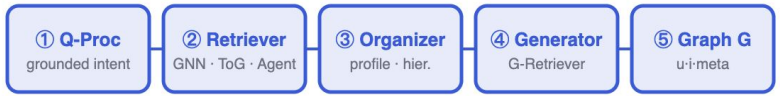
Evaluation & new apps

- Component-level + end-to-end
- Path-faithfulness, structural recall
- Personalization lift as 1st-class metric
- Code, cybersecurity, robotics

A grand challenge for recsys. A unified GraphRAG that (i) ingests live user logs, (ii) reasons over a fast-changing product catalog, (iii) explains every recommendation via grounded paths, and (iv) calibrates its uncertainty to the user's risk tolerance — open along every axis above.

Recap & five takeaways

What we covered (the 5-component lens)



Connections to emerging topics:

- **Agent Memory** — Episodic + Semantic + Procedural + Profile, all graphs.
- **Harnesses** — Tool-DAGs retrieved like KG paths.
- **Context Engineering** —Typed, structured prompts assembled from sub-graphs.

Five takeaways

1. **The 5-component pipeline** is your mental scaffold for any GraphRAG project.
2. **Graphs are golden for personalization** — proximity, role, and personalization rationales, each map to a recsys problem.
3. **Mix neural & symbolic.** GNN-RAG, KEQING, ToG, KERAG all win by integration.
4. **Organizer is not optional** — profile summarization & hierarchical aggregation define real-world feasibility.
5. **Future = GraphRAG as system.** Memory, harnesses, context engineering converge to a single graph substrate..

Closing thought. Personalization is fundamentally about the edges that connect you to the world. GraphRAG is the first retrieval paradigm to put those edges front-and-center.

Survey: github.com/Graph-RAG/GraphRAG · KERAG: arxiv.org/abs/2509.04716

Tutorial Outline



1. Introduction (40 min): Motivation, Factuality landscape, RAG at a glance

2. RAG Deepdive (110 min)
    - Modularized RAG (50 min)

    - Graph-based RAG (30 min)

    Break
    - Agentic RAG (30 min)

3. Advanced Topics (40 min)
    - Deep research (20 min)

    - Multi-modal RAG (20 min)

4. Conclusions and future directions (15 min)

02.03

Agentic RAG

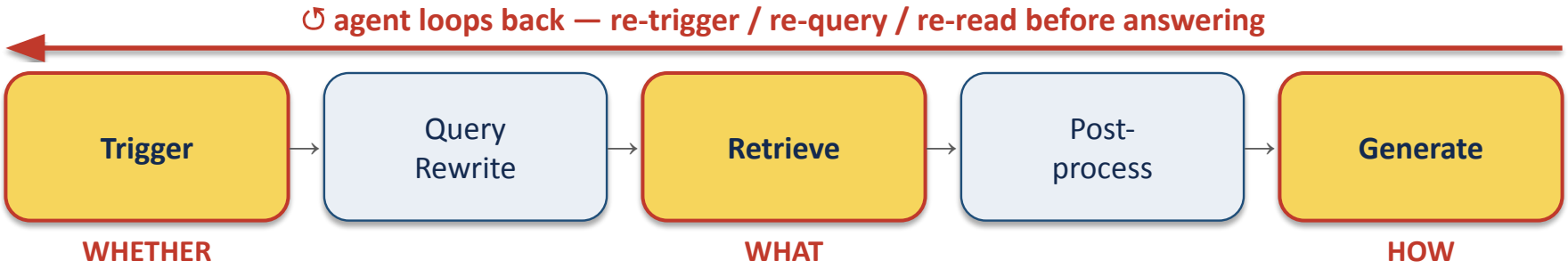
When to retrieve, what to fetch, and how to use the results — as one learned decision?

Agentic RAG

Turn RAG from a fixed pipeline into a learned, dynamic decision: whether to retrieve, what to retrieve, and how to leverage the retrieved results.

Problem framing

RAG as a dynamic decision process



- **Input: a (possibly complex) question. Output: a grounded, cited answer.**
- **Agentic RAG turns the fixed pipeline into a LOOP that decides, at inference:**
  - WHETHER to retrieve
  - WHAT to retrieve
  - HOW to leverage the results
- **Success: higher factuality and acceptable latency on multi-hop / dynamic questions.**

Why it matters

where single-pass RAG breaks

- **Running example (continues from 2.1/2.2):**
  - "Which has a longer driving range — Tesla Model 3 or Lucid Air?"
- **Single-pass / modular RAG struggles:**
  - one retrieval round rarely returns BOTH entities' range specs
  - no step to compare the two numbers → partial or wrong answer
  - over-retrieves on easy parts / under-retrieves on hard parts
- **Need:**

  decompose → retrieve per sub-question → reason → (re-)retrieve → compare.

From modular pipelines to agentic policies

▸ **Paradigm shift: hand-crafted pipelines → learned retrieval policies.**

	Modular RAG (2.1/2.2)	Agentic RAG (2.3)
Control	fixed, hand-crafted pipeline	learned, dynamic policy
Retrieval	often single pass, always-on	decide whether / what / how, iteratively
Reasoning	read-then-generate	interleaved multi-step reasoning
Self-check	little / none	reflection, self-critique, correction
Structure	linear	loop + planning (± multi-agent)

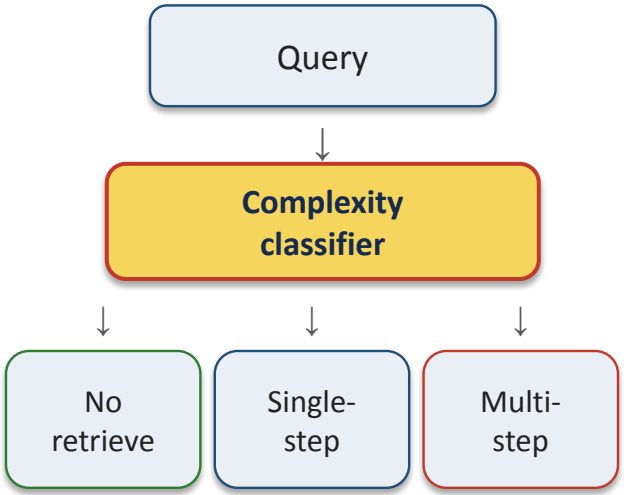
Surveys: [Fan 2024] KDD'24:2405.06211 · [Ni 2025] 2502.06872 · [Singh 2025] 2501.09136 · RAG-Reasoning survey:2507.09477

Four approaches to agentic RAG

- **One exemplar per family (next slides).**

Family	Key decision	Exemplar
(A) Adaptive Triggering	WHETHER to retrieve	Adaptive-RAG
(B) Iterative Retrieve–Reason	WHAT, step by step	IRCoT
(C) Learned RL Policies	learn when/how to search	Search-R1
(D) Planning & Scaling	how to use results / plan	REAP

(A) Adaptive Triggering: Deciding whether to retrieve



▸ **Exemplar: Adaptive-RAG [Jeong 2024] — a complexity classifier routes each**

– query to no-retrieval / single-step / multi-step (accuracy vs. cost).

✓ makes the whether/how-much decision explicit and cheap

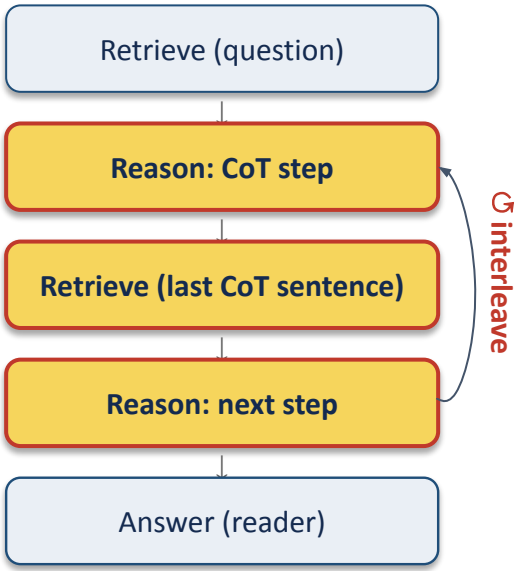
– Also: Mallen'23; Ni'24; Self-Routing RAG [Wu'25]; FLARE [Jiang'23]

🚗 **EV example: skip retrieval for "what is an EV?"; trigger it for the Model 3 vs Lucid Air range (long-tail specs).**

💡 **Key insight:** Retrieval is a cost — spend it only when the model is likely wrong (long-tail or low-confidence).

[Jeong 2024] NAACL:2403.14403 · [Mallen 2023] ACL:2212.10511 · [Ni 2024] ACL-F:2402.11457 · [Wu 2025] 2504.01018 · [Jiang 2023] EMNLP:2305.06983

(B) Iterative Retrieve–Reason: Deciding what to retrieve, step by step



▸ **Exemplar: IRCoT [Trivedi 2023] — retrieve first (with the question), then**

– interleave: reason one CoT step → retrieve using that sentence → repeat.

✓ training-free; stop at "answer is:", then a reader answers over collected docs

– Also: DRAGIN [Su'24]; TRACE [Fang'24]; Self-RAG [Asai'24]; Corrective RAG [Yan'24]

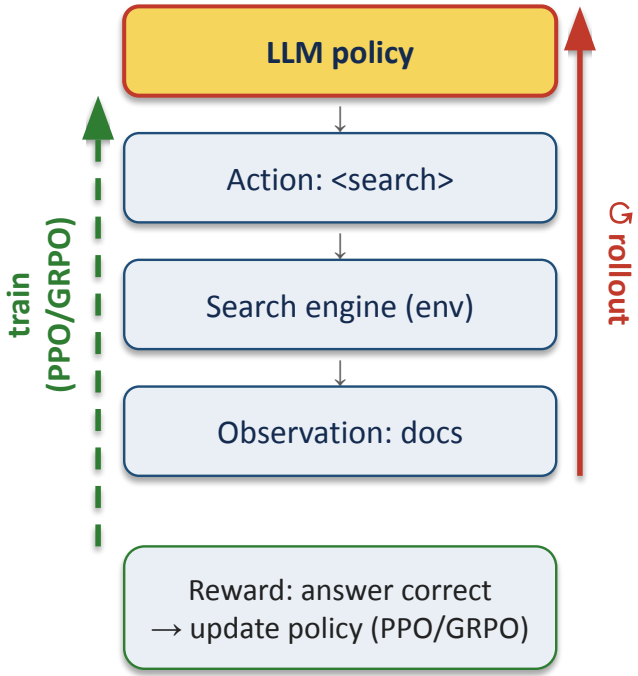
🚗 **EV example: reason "need Model 3 range" → retrieve it → reason "now Lucid Air range" → retrieve → compare (reasoning picks each next query).**

💡 **Key insight:** Let the reasoning chain choose the next query — retrieval and reasoning improve each other.

[Trivedi 2023] ACL:2212.10509 · [Su 2024] ACL:2403.10081 · [Fang 2024] EMNLP-F:2406.11460 · [Asai 2024] ICLR:2310.11511 · [Yan 2024] 2401.15884

(C) Learned Retrieval Policies: Learning when & how to search (RL)

Family C



▸ **Exemplar: Search-R1 [Jin 2025] — RL teaches the model to emit search**

– calls interleaved with reasoning; reward = answer correctness.

✓ the reference recipe for the 2025 RL-search wave

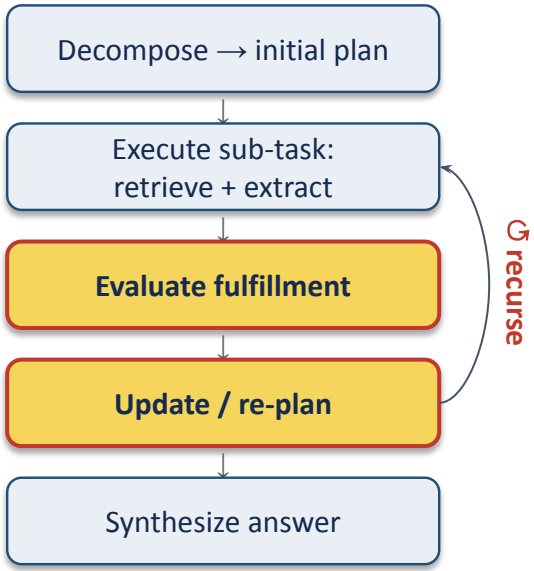
– Also: ReSearch [Chen'25]; DeepRetrieval [Jiang'25]; Search-o1; Co-MARL; DecEx-RAG

🚗 **EV example: the policy learns to issue both range queries and stop once it can compare.**

💡 **Key insight:** Don't script search — reward the final answer and let the search policy emerge.

[Jin 2025] COLM:2503.09516 · [Chen 2025] ReSearch:2503.19470 · [Jiang 2025] DeepRetrieval:2503.00223 · [Li 2025] Search-o1:2501.05366 · Co-MARL:2501.15228

(D) Planning & Scaling: Structuring multi-step reasoning



▸ **Exemplar: REAP [Zhu 2025] — decompose → execute → evaluate → re-plan:**

– a recursive plan↔evaluate loop for multi-hop QA; the plan mutates each cycle.

– Also: ReasonIR [Shao'25] — retriever trained for reasoning; StructRAG [Li'24]; Tree-of-Thoughts [Yao'23]; MEM1 [Zhou'25]

🚗 **EV example: look up each range; if a spec is missing, re-plan & search again; then compare.**

💡 **Key insight:** For hard questions, plan and structure the evidence first; retrieval follows the plan.

[Zhu 2025] REAP:2511.09966 · [Li 2024] StructRAG ICLR:2410.08815 · [Yao 2023] ToT:2305.10601 · [Zhou 2025] MEM1:2506.15841 · [Shao 2025] ReasonIR:2504.20595

The four agentic RAG approaches, side by side

ONE PASS: Trigger → Retrieve → Generate

Family	TRIGGER *whether*	RETRIEVE *what*	GENERATE *how*	LOOP *run again?*
Adapt ONE step of the pass				
(A) Adaptive Triggering Adaptive-RAG	**classifier routes: none / single / multi**	standard	standard	**STATIC** *depth fixed up front (0/1/many)*
(B) Iterative Retrieve–Reason IRCoT	always	**reasoning picks the next query**	standard	**DYNAMIC** *interleave reason + retrieve*
Replace the CONTROLLER of the whole loop — C learns it, D plans it				
(C) Learned RL Policies Search-R1	**a single search POLICY (implicit in the model) LEARNED offline by RL · reward = answer correct**			**DYNAMIC** *multi-turn search*
(D) Planning & Scaling REAP	**an explicit, written PLAN (decompose → sub-tasks) REVISED at inference · evaluate → re-plan · no training**			**DYNAMIC** *plan-driven recurse*

Running example

iterative retrieval & self-check

▸ **One retrieval per sub-question; grade each passage (relevant? supported?) and re-query if ambiguous.**

Sub-question	Retrieved evidence	Self-check
Range, Tesla Model 3	~363 mi (Long Range RWD, EPA; AWD 346)	relevant ✓ / supported ✓ — trim matters
Range, Lucid Air	~512 mi (Grand Touring, EPA)	relevant ✓ / supported ✓
Ambiguity?	trims differ (AWD vs RWD; wheels)	re-query with the specific trim

Running example

compare & answer

- **Reason / integrate: 512 mi (Lucid Air) > 363 mi (Tesla Model 3).**
- **Answer (grounded + attributed):**
  - "The Lucid Air has the longer range — ~512 mi (Grand Touring) vs ~363 mi
    · (Model 3 Long Range RWD), per EPA estimates."
    ✓ agentic RAG added: decomposition · per-fact retrieval · self-verification · comparison
- **Next: two canonical methods that learn this loop — Self-RAG (self-check) & Search-R1 (learned search).**

Self-RAG — Problem & Key Idea (1/3)

Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection

Asai et al. — ICLR 2024 · arXiv 2310.11511

The Challenge

Grounding a long-form answer means knowing WHEN to retrieve and whether each claim is actually supported by evidence.

The Gap

Standard RAG retrieves on every query (wasteful) and never checks that its own output is supported — no self-correction.

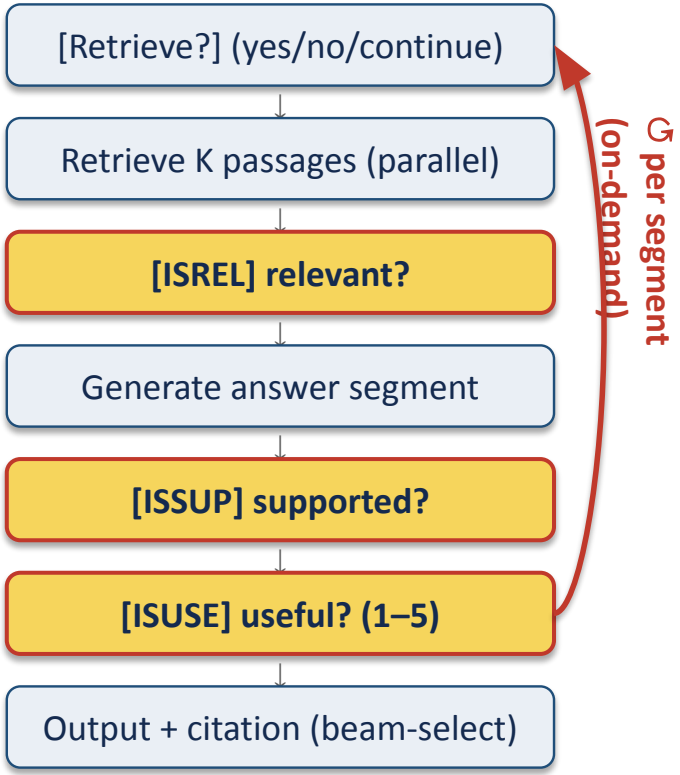
💡 Key Idea

Train ONE LM to emit reflection tokens that trigger retrieval on demand and critique relevance / support / usefulness.

Impact: Adaptive retrieval + verifiable self-critique in a single model — beats ChatGPT on PopQA/PubHealth; improves citation accuracy.

Self-RAG — Method

reflection tokens + segment beam search (2/3)



[Asai 2024] Self-RAG, ICLR 2024, arXiv:2310.11511

- **Training (no RL):**
  - – a critic LM (distilled from GPT-4) labels a corpus with reflection tokens
  - – the generator is fine-tuned over text + reflection tokens (next-token)
- **Inference:**
  - – segment-level beam search uses reflection-token probabilities
  - – to re-rank / re-retrieve — controllable at decode time

Self-RAG — Results

concrete gains, driven by self-critique (3/3)

- **Concrete gains vs. best-performing LLMs with proprietary data / retrieval-augmented baselines.**

Benchmark	Self-RAG vs. best-performing baseline
PopQA (acc)	55.8 vs Ret-Llama2-chat 51.8 (13B)
PubHealth (acc)	74.5 vs ChatGPT 70.1 (13B)
Bio generation (FactScore)	81.2 vs Ret-Llama2-chat 79.9 (7B)
ASQA (citation precision)	70.3 vs Ret-ChatGPT 65.1 (13B)
Ablation: remove self-critique	ASQA correctness 32.1 → 18.1

Key Takeaway: the self-critique mechanism drives the citation & long-form gains — not retrieval alone.

[Asai 2024] Self-RAG, ICLR 2024, arXiv:2310.11511 (Table 2 & ablation).

Search-R1 — Problem & Key Idea (1/3)

Search-R1: Training LLMs to Reason and Leverage Search Engines with Reinforcement Learning

Jin et al. — COLM 2025 · arXiv 2503.09516

The Challenge

Multi-hop questions need the model to decide WHEN to search, WHAT to query, and WHEN to stop — a sequential decision problem.

The Gap

Prompt-engineered search agents are brittle; supervised trajectories are rigid & expensive — neither learns a robust policy.

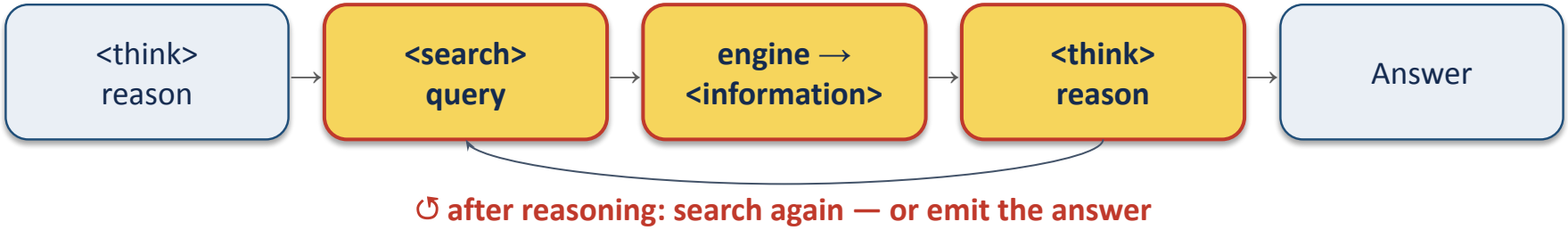
💡 Key Idea

Use RL so the LLM searches mid-reasoning, rewarded only by final-answer correctness (retrieved-token masking stabilizes training).

Impact: An emergent, query-adaptive search policy — +26% average EM over RAG baselines (Qwen2.5-7B).

Search-R1 — Method

think / search / information loop (2/3)



- **RL setup:**
  - outcome reward (final-answer correctness) via PPO/GRPO; no process supervision
  - retrieved-token masking — train the policy to USE retrieved text, not imitate it
- **Search becomes a LEARNED skill, not a fixed pipeline step.**

[Jin 2025] Search-R1, COLM 2025, arXiv:2503.09516

Search-R1 — Results

learned search beats scripted retrieval (3/3)

- **RL with only outcome reward; Exact Match averaged over 7 QA datasets.**

Aspect	Finding
Avg EM gain vs. best RAG baselines	+24% (Qwen2.5-7B), +20% (3B)
Avg EM, 7 QA sets (7B)	43.1 vs best baseline 34.8
Ablation: retrieved-token masking	avg EM 43.1 → 34.3 without it

Key Takeaway: retrieved-token masking improves EM and stabilizes RL; search behavior is emergent.

[Jin 2025] Search-R1, COLM 2025, arXiv:2503.09516 (Tables 2 & 4).

Self-RAG vs. Search-R1

inference-time control vs. learned policy

▸ **Two ends of the agentic-RAG spectrum.**

Dimension	Self-RAG	Search-R1
Learning	supervised fine-tuning (no RL)	reinforcement learning (PPO/GRPO)
Signal	distilled reflection-token labels	final-answer outcome reward
Control	decode-time reflection tokens	emergent, learned search policy
Strength	controllability + attributability	flexible search at scale

Key Takeaway: self-reflection when you want control & citations; RL policies for emergent search at scale.

Closing highlight — Stream RAG

Team highlight

agentic RAG made real-time

Standard:
listen ⟶ [pause] retrieve ⟶ answer

→

Stream RAG:
speak + STREAM tool calls concurrently

- **Stream RAG [Arora 2026] — instant & accurate spoken dialogue with streaming tool usage.**
- **Emit/stream tool calls as the model speaks, instead of pausing for a full cycle.**

✓ agentic RAG made practical for real-time, on-device / wearable assistants

Key Takeaway: the next frontier is efficient agentic RAG under latency budgets, not just accuracy.

[Arora 2026] Stream RAG, ICML 2026, arXiv:2510.02044

Closing highlight — Stream RAG

Team highlight

real-time spoken dialogue (1/2): the problem & the idea

Standard:
listen → [pause] retrieve → answer

→

Stream RAG:
speak + STREAM tool calls concurrently

- **Standard "retrieve-after-endpoint": wait for the user to finish → then retrieve → seconds of delay that break natural turn-taking.**
- **Stream RAG [Arora 2026]: issue tool queries during the user's speech — masking API latency while keeping quality.**

✓ agentic RAG made practical for real-time, on-device / wearable assistants

[Arora 2026] Stream RAG, ICML 2026, arXiv:2510.02044

Closing highlight — Stream RAG

Team highlight

real-time spoken dialogue (2/2): how it works & results

▸ Model-Triggered Stream RAG: a learned policy reads speech in chunks and decides each step — fire a new query or “NO QUERY” — from partial input.
▸ A new query cancels the previous call (one thread at a time) → low overhead, on-device friendly; post-training teaches when to call & how to speak results.

Results — AudioCRAG (built from CRAG), vs. cascade & WavRAG baselines:

Setting	EM	First-token latency
Cascade (Whisper→Qwen2.5→VITS)	36.6	1.53 s
WavRAG (GPT-4o)	40.1	0.56 s
Stream RAG (Qwen-OMNI)	**55.8**	**−0.68 s †**

† first token generated before the user finishes speaking.

Key Takeaway: the next frontier is efficient agentic RAG under latency budgets, not just accuracy.

[Arora 2026] Stream RAG, ICML 2026, arXiv:2510.02044

Agentic RAG — Open Problems

what remains unsolved?

- **Latency & cost: every loop step adds an LLM call + retrieval round-trip.**
- **Error compounding: a wrong sub-query or bad passage cascades across steps.**
- **Evaluation: multi-hop / dynamic benchmarks — MultiHop-RAG, FRAMES, CRAG [Yang 2024].**
- **Reliability on dynamic / fast-changing facts remains hard.**

[Yang 2024] CRAG NeurIPS:2406.04744 · MultiHop-RAG:2401.15391 · FRAMES NAACL'25:2409.12941

Agentic RAG — Recap & 3 takeaways

1. Agentic RAG = RAG as a learned, dynamic decision: whether / what to retrieve, and how to use the results.
2. Two levers: inference-time self-reflection (Self-RAG) vs. RL-learned search policy (Search-R1).
3. The open frontier is efficiency — strong answers under latency/cost budgets (e.g., Stream RAG).

Recommendation: start with adaptive triggering + iterative retrieve–reason; add RL search policies for reasoning-heavy, high-value queries.

Tutorial Outline



1. Introduction (40 min): Motivation, Factuality landscape, RAG at a glance

2. RAG Deepdive (110 min)
    - Modularized RAG (50 min)

    - Graph-based RAG (30 min)

    Break
    - Agentic RAG (30 min)

3. Advanced Topics (40 min)
    - Deep research (20 min)

    - Multi-modal RAG (20 min)

4. Conclusions and future directions (15 min)

03.01

Deep Research

How do we answer open-ended questions with a structured, cited report?

Deep Research

A long-horizon agent that plans, browses the web, reasons over evidence, and writes a structured, cited report.

Problem framing

what is Deep Research?

- **From a single fact, to a reasoned answer, to a full report.**

	Shallow QA	Agentic RAG (2.3)	Deep Research (3.1)
Question	simple, factual	specific, multi-hop	open-ended, exploratory
Horizon	one shot	seconds	minutes–hours
Process	retrieve → read	retrieve–reason loop	plan → browse → synthesize → write
Output	a short answer	a grounded answer	a structured, cited report

- **Success: coverage, faithful synthesis, correct attributions, useful structure.**

Why it matters

open-ended questions break QA-style RAG

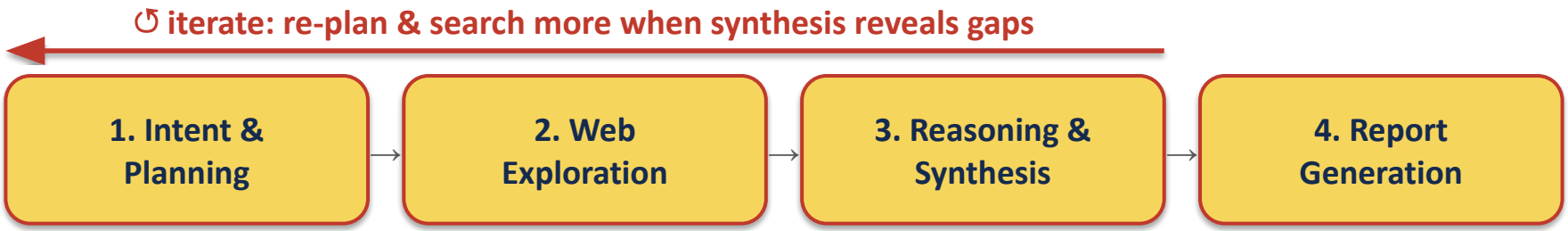
- **Running example (deep-research tier):**
  - "I'm considering buying an EV in 2026. Which model is best, and why?"
- **Single-pass or even agentic-QA RAG returns a thin answer; the task needs:**
  - breadth — survey many candidate models
  - structure — compare across dimensions (range, price, charging, reliability)
  - synthesis — weigh trade-offs, tailor a recommendation
  - faithful citations for every claim

Landscape

commercial deep research went mainstream

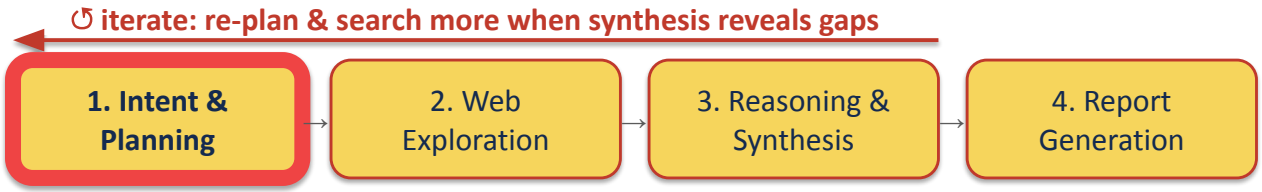
- **Deep research is now a mainstream product capability:**
  - OpenAI Deep Research · Gemini Deep Research · Claude Research · Perplexity
- **All are closed products — we see the shape, not the internals:**
  - common recipe: plan → iterative web search / tool calls → reason → cited report
  - heavy test-time compute traded for depth & coverage
- **They differ mainly in UX, underlying model, and time/compute budget —**
  - not in the core recipe → next, we formalize that recipe as a 4-stage pipeline.
- **Mechanisms come from open & academic systems (studied within each stage).**

The deep research pipeline



- **Four stages — the first three iterate until the report is complete:**
  - 1. Intent & planning — understand the goal, draft a plan
  - 2. Web exploration — iterative search & browsing across many sources
  - 3. Reasoning & synthesis — aggregate, resolve conflicts, ground claims
  - 4. Report generation — long-form, structured, cited output

Stage 1 — Intent clarification & planning

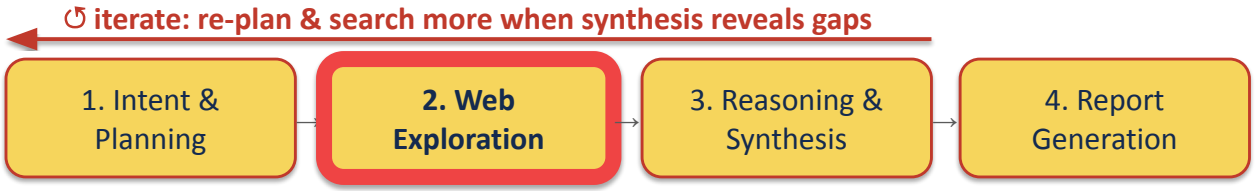


- **Turn an open-ended ask into an executable research plan:**
  - clarify scope / constraints (budget, region, use case)
  - decompose into sub-questions & dimensions to investigate
  - draft an outline / plan that later stages fill in
- **Methods & challenges:**
  - REAP [Zhu 2025] — recursive evaluation + adaptive planning (carried from 2.3)
  - long-horizon credit assignment: which early step caused a late error?

🚗 **EV example: clarify budget/region/use → plan dimensions: range, price, charging, reliability.**

[Zhu 2025] REAP, arXiv:2511.09966

Stage 2 — Web exploration

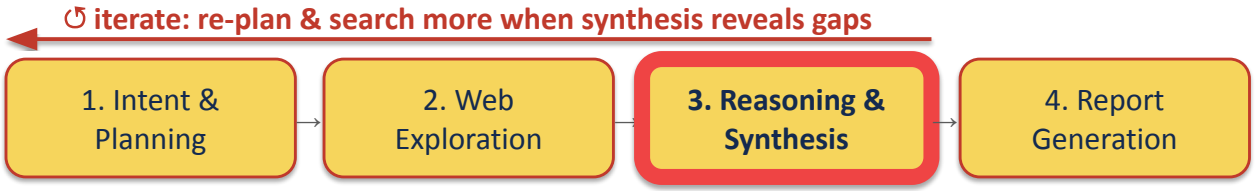


- **Iteratively search, browse, and gather evidence across many sources:**
  - WebGPT [Nakano 2022] — browser-assisted QA (precursor)
  - Search-o1 [Li 2025] — agentic search inside a large reasoning model
  - Search-R1 [Jin 2025] — RL-learned search policy (carried from 2.3)
- **Open-source: GPT-Researcher; HF Open Deep Research (smolagents).**
- **Long trajectory: MEM1 [Zhou 2025] — memory + reasoning for long-horizon agents.**

🚗 **EV example: run per-model searches (Model 3, Lucid Air, Ioniq 6) for specs, reviews, recalls — keep sources.**

[Nakano 2022] WebGPT:2112.09332 · [Li 2025] Search-o1:2501.05366 · [Jin 2025] Search-R1:2503.09516 · [Zhou 2025] MEM1:2506.15841

Stage 3 — Reasoning & synthesis

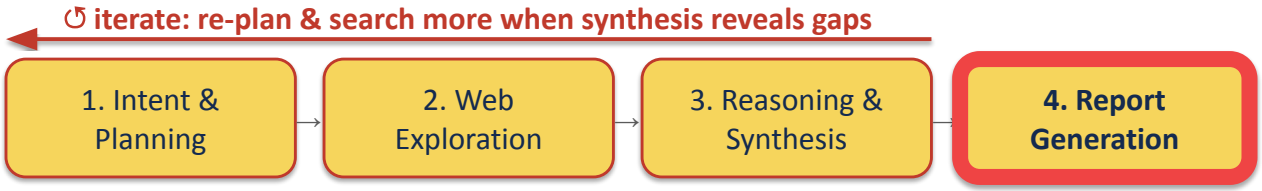


- **Combine evidence from many sources into a coherent picture:**
  - aggregate & deduplicate findings across dozens of sources
  - resolve knowledge conflicts (which source to trust, recency)
  - track which sources support each claim — evidence→claim links [Asai 2024]
- **Challenge: faithful multi-document reasoning at scale —**
  - structured synthesis, not just concatenating retrieved text.

🚗 **EV example: reconcile conflicting range numbers across sources; build a per-model comparison.**

[Asai 2024] Reliable, Adaptable, and Attributable LMs with Retrieval, arXiv:2403.03187

Stage 4 — Report generation



- **Turn synthesized evidence into a structured, long-form deliverable:**
  - plan-then-write: outline → sections → prose
  - STORM / Co-STORM [Shao 2024; Jiang 2024] — perspective-driven outline-then-write
  - render the Stage-3 evidence links as inline citations
    - ✓ verify each citation actually supports its sentence (post-hoc check)
- **Challenge: coherence + faithfulness over long output —**
  - avoid confident-but-unsupported claims.
    - 🚗 **EV example: write "top 2–3 picks + trade-offs" with a citation for every spec.**

STORM NAACL'24:2402.14207 · Co-STORM EMNLP'24:2408.15232

Running example

synthesize & report

▸ **Stage 3 → comparison table; Stage 4 → 2–3 picks tailored to the user, with trade-offs & citations.**

Model	Range (EPA comb.)	MSRP	Charging	Note
Tesla Model 3 LR	363 mi (RWD) / 346 (AWD)	~$44k	Supercharger 250 kW	best charging network
Lucid Air GT	512 mi	~$115k	900V, up to 300 kW	longest range, premium
Hyundai Ioniq 6 LR	342 mi	~$43k	800V, ~235 kW	10–80% ≈ 18 min

Key Takeaway: Sample recommendation (all cited): range-first → Lucid Air; value & fast-charging → Model 3 or Ioniq 6 — delivered as a structured report, not a one-liner.

OpenScholar — Problem & Key Idea (1/3)

OpenScholar: Synthesizing Scientific Literature with Retrieval-Augmented Language Models

Asai et al. — Nature 2026 · arXiv 2411.14199

The Challenge

Scientific questions require synthesizing across MILLIONS of papers with faithful, checkable citations.

The Gap

General LLMs hallucinate references (78–90% of the time) and miss recent work; closed systems aren't reproducible.

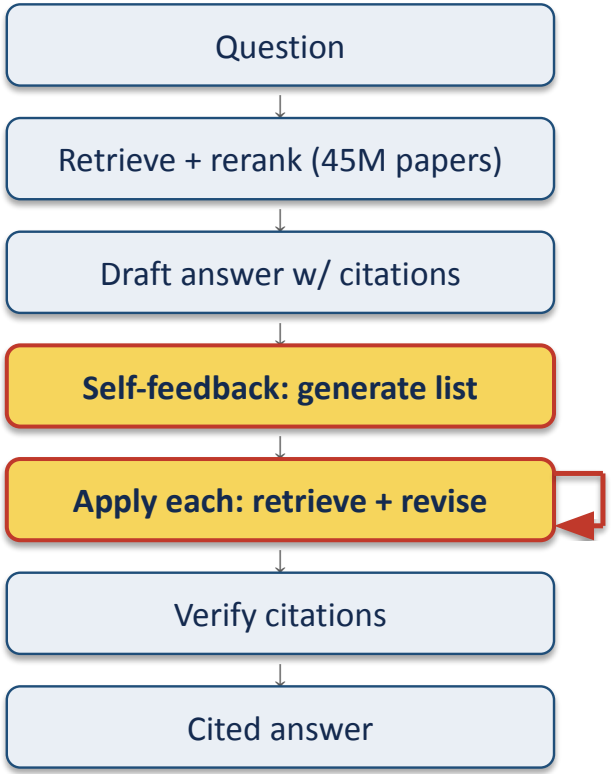
💡 Key Idea

Retrieve over 45M+ papers, then iteratively self-improve the draft with feedback, citing retrieved sources inline.

Impact: An open 8B model matches/beats GPT-4o on grounded synthesis — citations on par with human experts.

OpenScholar — Method

retrieve → draft → self-feedback → revise (2/3)



- **Components:**
  - datastore of 45M papers + trained bi-encoder retriever & cross-encoder reranker
  - self-feedback: generate a feedback list (≤3) ONCE, then apply each — retrieve + revise
  - final step verifies citations (post-hoc insertion if unsupported)
- **Maps onto pipeline stages 2–4 (explore → synthesize → cite).**

Asai et al., Nature 650, 857–863 (2026); arXiv:2411.14199.

OpenScholar — Results

a reproducible deep-research recipe (3/3)

▸ **ScholarQABench: literature-grounded long-form QA (2,967 expert queries).**

Evaluation	Finding
Correctness	OpenScholar-8B beats GPT-4o by 6.1%, PaperQA2 by 5.5%
Correctness (GPT-4o back end)	OpenScholar-GPT-4o beats GPT-4o by 12%
Citation accuracy	GPT-4o hallucinates cites 78–90%; OpenScholar on par w/ experts
Expert win-rate vs human	OpenScholar-GPT4o 70%, OpenScholar-8B 51%, GPT-4o 32%
Ablation (largest effect)	removing the reranker → biggest drop in both

Key Takeaway: an open 8B model matches/beats GPT-4o on grounded synthesis — citations on par with experts.

[Asai et al.] OpenScholar, Nature 650:857–863 (2026); arXiv:2411.14199

Evaluating deep research

- **Hard to score: report quality, coverage, citation faithfulness.**

Benchmark	What it tests
GAIA	general assistant tasks (tools + reasoning)
BrowseComp	hard, realistic web-browsing tasks
Humanity's Last Exam	frontier knowledge & reasoning
DeepResearch Bench / ResearchRubrics	report quality & rubric-scored deep research
ScholarQABench	literature-grounded long-form answers

GAIA ICLR'24:2311.12983 · BrowseComp:2504.12516 · HLE:2501.14249 · DeepResearch Bench:2506.11763 · ResearchRubrics:2511.07685 · survey:2512.02038

Deep Research — Open Problems

what remains unsolved?

- **Citation faithfulness & hallucination in long reports — confident-but-unsupported claims.**
- **Planning & long-horizon credit assignment — hard to learn what mattered.**
- **Exploration cost & latency — minutes–hours, many tool calls.**
- **Evaluation — open-ended outputs are hard to score reliably.**

Deep Research — Recap & 3 takeaways

1. Deep Research = a long-horizon agent that produces a structured, CITED report — not a one-line answer.
2. A four-stage pipeline: plan → explore → synthesize → report; each stage is its own research problem.
3. Faithful citation and long-horizon planning are the hardest open problems.

Recommendation: for open-ended / report tasks, use a plan-then-explore agent with explicit citation checking (OpenScholar-style self-feedback).

Tutorial Outline



1. Introduction (40 min): Motivation, Factuality landscape, RAG at a glance

2. RAG Deepdive (110 min)
    - Modularized RAG (50 min)

    - Graph-based RAG (30 min)

    Break
    - Agentic RAG (30 min)

3. Advanced Topics (40 min)
    - Deep research (20 min)

    - Multi-modal RAG (20 min)

4. Conclusions and future directions (15 min)

03.02

Multi-modal RAG

The Paradigm Shift: Transitioning from "Text-to-Text" to "Any-to-Any" Retrieval and Generation

MM-RAG: Retrieval Beyond Text

Multi-modal RAG (MM-RAG) extends RAG by retrieving external knowledge represented in **multiple modalities**—such as text, images, video, audio, tables, charts, and document layout—and integrating that evidence into a model's reasoning or generation process.

MM-RAG Interfaces

A Running Example

FOCUS OF THIS TUTORIAL

An **image** and text query in; **multimodal evidence** retrieved; a text answer out.

Q: How far can it go on a full charge?

The pixels anchor the question.
The answer lives outside the pixels.

1 **Ground the visual target**
Which object or region matters?

2 **Resolve missing identity**
Which attributes affect the answer but may not be visible?

3 **Formulate and route search**
Convert the visual observation into explicit retrieval queries;
Retrieval can be both **image-based** and **text-based**

4 **Verify and reconcile evidence**
Are the retrieved evidence consistent with the query?

5 **Generate grounded answer**
With supporting evidence

Why MM-RAG? The Motivations

1. Converting Everything to Text Is Lossy

Pseudo MM-RAG

Forces multimodal data into text via OCR/Captioning. Destroys spatial layouts and causes irrecoverable information loss.

- Parse → OCR → Text Retrieval
- Significant fidelity degradation

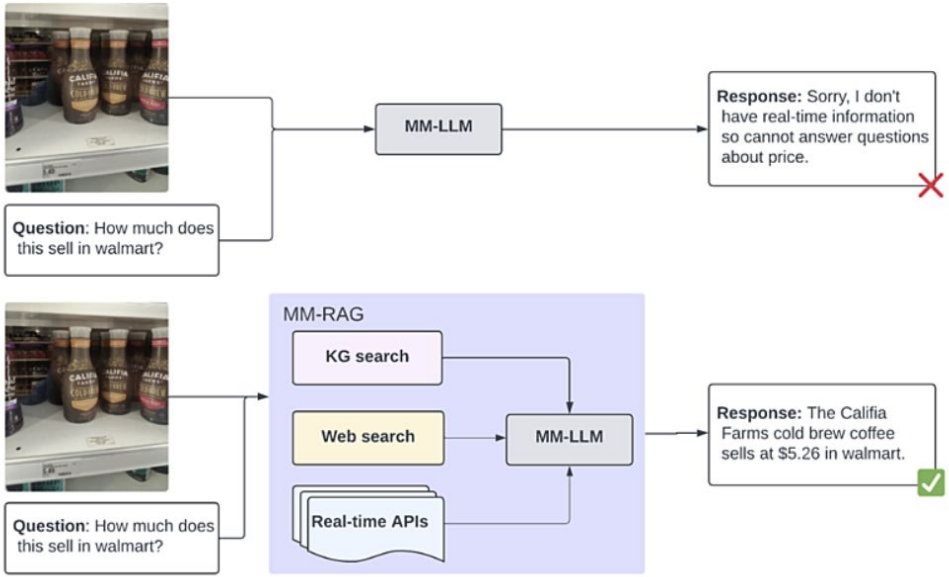
True MM-RAG

Maintains original multimodal representations. Leverages VLM encoding for direct feature search.

- VLM Encoding → MM Retrieval
- Preserves original layout/data fidelity

Why MM-RAG? The Motivations

2. Grounding Multimodal LLMs to Reduce Hallucinations



Why MM-RAG? The Motivations

3. Retrieval: Enabling Cross-Modal Alignment and Search

4. Query: Understanding Inherently Multimodal Queries

MM-RAG Applications

perception + knowledge + memory → assistance

Wearable AI Assistant

Understand what I'm looking at

Recognize objects, landmarks, or products and retrieve facts, prices, or instructions.

Visual Memory

Help me remember my past

Search personal photos, videos, audio, and context to answer episodic memory questions.

Personalized Recommendation

Recommend what I should do next

Combine the current scene, live local information, and user preferences for travel or shopping.

Task Guidance

Guide me through a task

Retrieve manuals, demonstrations, or past experience and align the next step to the current scene.

How Do We Measure MM-RAG Quality?

CRAG-MM is the first comprehensive benchmark designed to evaluate Multi-Modal Retrieval-Augmented Generation (MM-RAG) systems, focusing on Wearables AI.

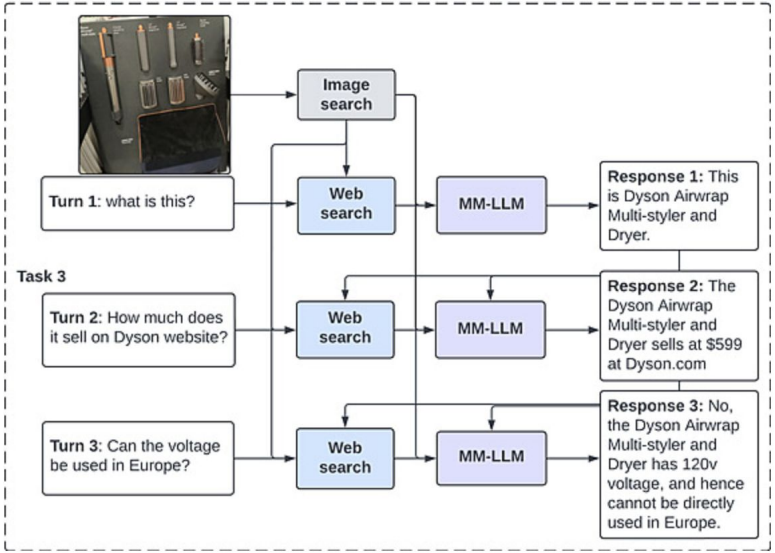
6.5K
Single-turn image-QA triples

2K
Vision-based multi-turn conversations

13
Domains

7.9K
Images, 79% egocentric

Image & Web Corpus
68K images, 26K entities & 800K URLs, 2.7M chunks



An Example of Multi-turn QA in CRAG-MM

Presentation: 8/11 Tue 1:30 PM, Dataset & Benchmark Track

Wang and Yang et al. CRAG-MM: Multi-modal Multi-turn Comprehensive RAG Benchmark, KDD 2026

How to Measure MM-RAG Quality?

CRAG-MM is a Comprehensive RAG benchmark for **Multi-modal Multi-turn** conversations. It features:

- 6.5k single-turn (image, question, answer) triplets
- 2k multi-turn conversations
- 7.9k images, 79% **egocentric**
- **89% factual queries requiring external info**
- **Image corpus** with 68k images, 26k entities, 93% Image entity coverage
- **Web corpus** with 800k URLs, 2.7M chunks, 89% estimated recall
- Optional search APIs

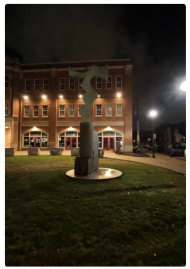


Image quality: Low light
Domain: Local
Query type: Reasoning
Question: Could a guest tour inside this museum in December?
Answer: No, the Indianapolis Firefighters Museum is only open on weekdays and Saturdays from April to October.

Performance of Best SOTA industry solutions on CRAG-MM

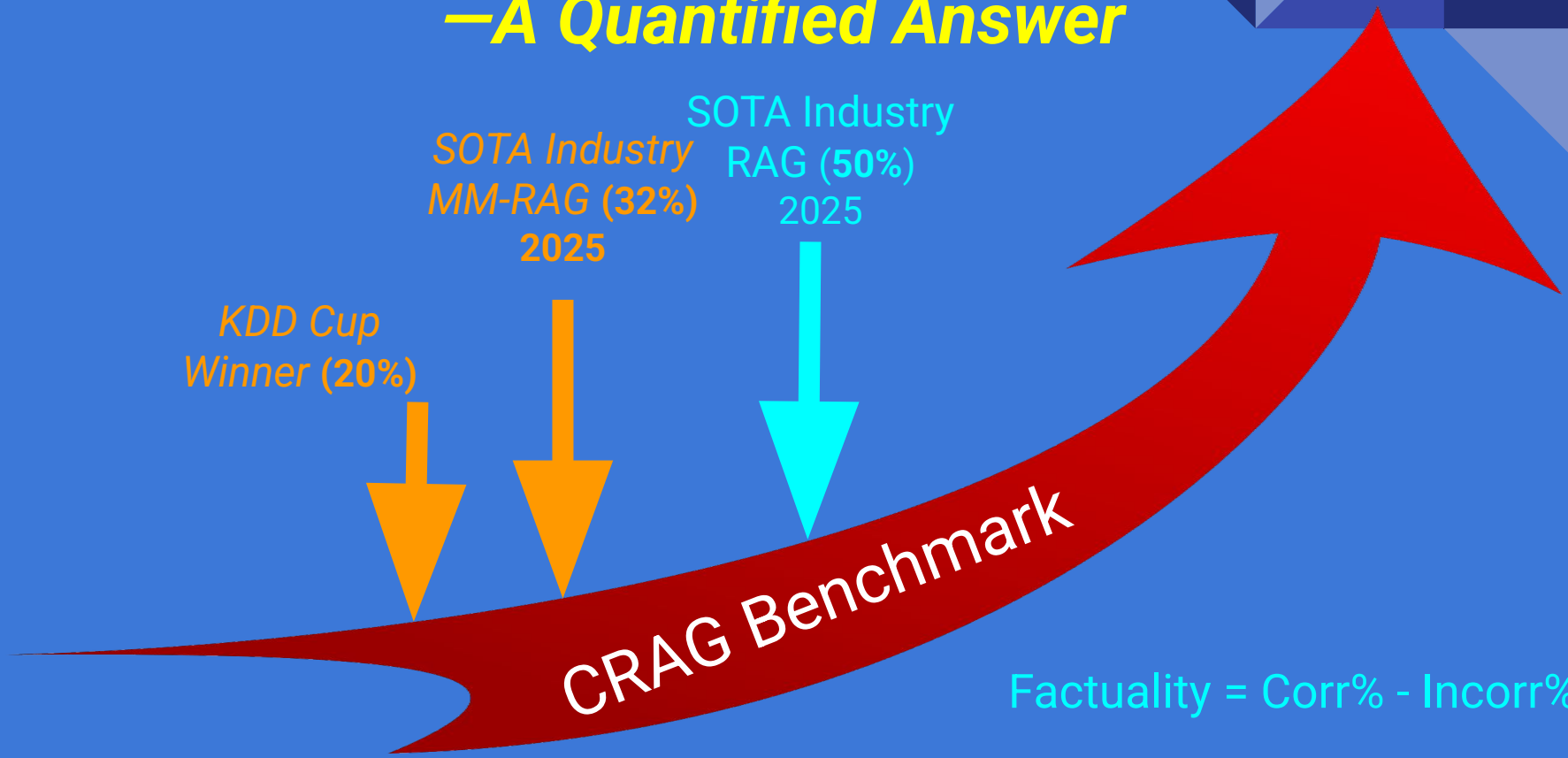
63%	32%	70%	45%
Single-turn Correct	Single-turn Factuality	Multi-turn Correct	Multi-turn Factuality

Factuality = Correct % - Incorrect %

The bottleneck spans perception, retrieval, evidence use, and hallucination control.

Wang and Yang et al. CRAG-MM: Multi-modal Multi-turn Comprehensive RAG Benchmark, KDD 2026

Where Are We in This Journey? —A Quantified Answer



Wang and Yang et al. CRAG-MM: Multi-modal Multi-turn Comprehensive RAG Benchmark, KDD 2026

MM-RAG Landscape: What is Unique?

1

Trigger

Should I retrieve?

Multi-modal Queries

→

2

Visual Grounding & Rewrite

Locate the relevant entity or region; Modality-aware search queries

→

3

Retrieve

Find relevant multi-modal / heterogeneous candidates

→

4

Post-process

Re-rank & Filter

Compression & fusion of multi-modal evidence

→

5

Generate

Produce the Answer

Multi-modal inputs and outputs; attribution becomes spatial and temporal

0

Knowledge Substrate and Indexing

Turn multimodal corpora into linked, retrievable evidence units.

MM-RAG Challenges

Visual Grounding & Query Formulation

What in the image should drive retrieval?

The entity or region is often implicit in pixels rather than named in the question.

Cross-modal Retrieval & Ranking

Which evidence corresponds to the depicted entity?

The query is image+text, while evidence may be passages, reference images, image–text pairs, or KG entries.

Multimodal Evidence Integration & Reasoning

How should the model apply retrieved facts to the visual referent?

The answer must keep visual–entity–fact correspondence despite noisy or conflicting evidence.

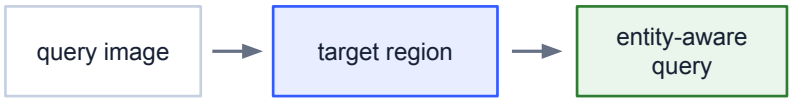
Challenge 1: Visual Grounding & Query Formulation

Core question

Which object, region, or entity in the image should be searched — and how should it become a retrieval query?

Why hard

- Question text may say “this”, “the animal”, or “the object on the left”.
- Background clutter and look-alike entities skew visual similarity.
- Fine-grained cues disappear when the image is reduced to a caption.



Approach

1 Full-image caption or global embedding

+ cheap; no detector; preserves scene context
– distractors dominate; captions miss identity cues

2 Explicit entity / region pipeline

+ interpretable; strong for long-tail entities
– detector and entity-candidate errors cascade

3 Adaptive query-side region selection

+ learns when to crop to improve ranking
– extra inference; cannot fix missing candidates

Deep-dive — Region-R1: make the visual query adaptive

Visual Grounding & Query Formulation

Problem

Standard MM re-rankers process the full query image as a global embedding, so irrelevant visual content can pull the top evidence toward the wrong entity.

Method sketch

Reward = rank improvement over baseline re-ranking metrics

Improvement

E-VQA conditional Recall@1
EchoSight 0.75 → Region-R1 0.90
+20% relative

InfoSeek conditional Recall@1
OMGM 0.75 → Region-R1 0.81
+8% relative

Why it helps: the crop removes distractions while keeping the same candidate set, so gains come from better **query-side grounding** rather than a bigger retrieval corpus.

Hu et al. Region-R1: Reinforcing Query-Side Region Cropping for Multi-Modal Re-Ranking, arXiv 2026

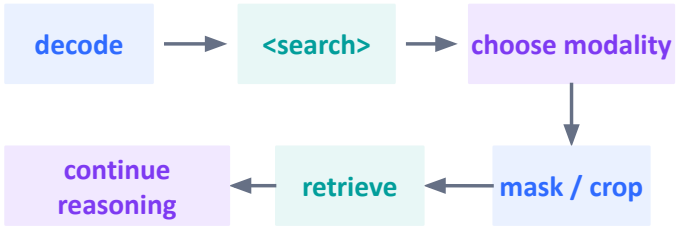
Deep-dive — PixSearch: pixel-grounded retrieval

Visual Grounding & Query Formulation

Problem

Prior MM-RAG often uses full-image search, or modular detector / segmenter / captioner pipelines; this can add noise, translation errors, and unnecessary latency.

Method sketch



Improvement

On CRAG-MM vs. whole-image

+16% Truthfulness

+6% Accuracy

Why it helps:

- Search is a model policy: emits <search> during decoding; can make multiple searches as reasoning unfolds
- Region queries are pixel-grounded

Kim et al. Pixel-Grounded Retrieval for Knowledgeable Large Multimodal Models, arXiv 2026

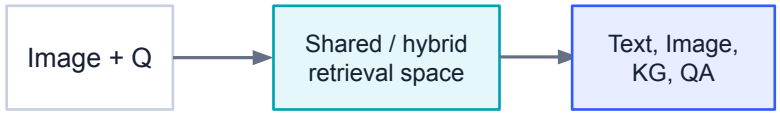
Challenge 2: Cross-modal Retrieval & Ranking

Core question

Which external item corresponds to the visual query and contains the fact needed to answer?

Why hard

- Query is image + question; candidates may be purely textual or multimodal.
- Visually similar entities can have different facts; same entity may appear under varied depictions.
- Scores from text, visual, and multimodal retrievers are not naturally comparable.



Approach

1 Text-proxy retrieval

+ mature BM25/DPR stack; exact-match search
– depends on caption/OCR/entity conversion quality

2 Explicit entity / region pipeline

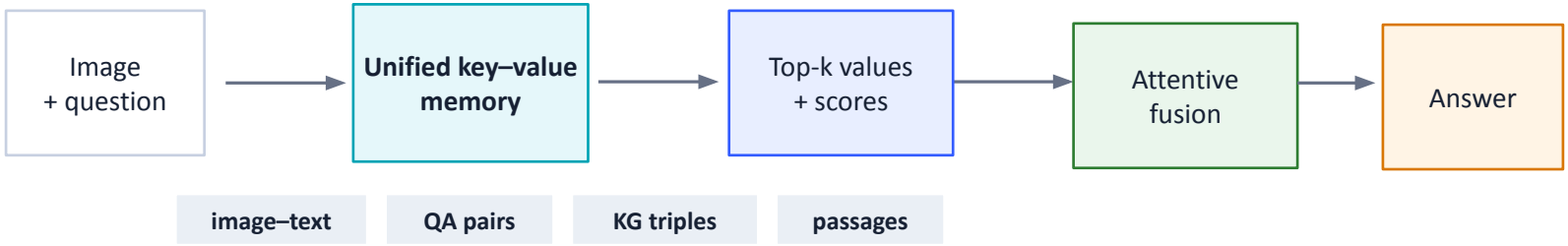
+ stores heterogeneous visual and textual knowledge
– global retrieval can miss fine-grained correspondence

3 Adaptive query-side region selection

+ captures local visual-question-evidence matches
– higher storage/compute; needs hard negatives or reranking

Deep-dive — REVEAL: retrieve from multi-source multimodal memory

Cross-modal Retrieval & Ranking



Key design: retrieval scores are injected into the fusion layer as attention priors, making **retrieved entries influence generation** rather than simply being concatenated.

OK-VQA accuracy

RA-VQA 54.5 → REVEAL 59.1

+4.6 accuracy

Why it helps: retrieval is over multimodal knowledge items, while **fusion is learned jointly with generation** — better matching of image-conditioned queries to answer-useful evidence.

Hu et al. REVEAL: Retrieval-Augmented Visual-Language Pre-Training with Multi-Source Multimodal Knowledge Memory, arXiv 2023

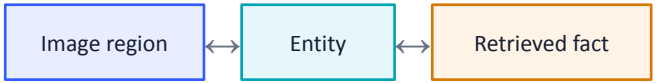
Challenge 3: Multimodal Evidence Integration & Reasoning

Core question

How should the model combine the query image, retrieved knowledge, and its parametric knowledge without losing visual–entity–fact grounding?

Why hard

- A retrieved statement can be true but apply to the wrong visually similar entity.
- Retrieved text can suppress useful visual evidence or override correct parametric knowledge.
- Concatenation gives source access, not visual applicability or conflict resolution.



Approach

1 Prompt-level concatenation

+ training-free; preserves top-k evidence
– no explicit visual applicability check

2 Learned fusion / reflection

+ weights evidence and can decide when retrieval helps
– may still conflate relevance with consistency

3 Visual-semantic arbitration

+ checks conflicts using image-conditioned evidence
– extra VLM calls; conflict analysis can be imperfect

Deep-dive — CC-VQA: resolve conflicts with visual-semantic correlation

Multimodal Evidence Integration & Reasoning

The MM-specific check is not just "is the passage relevant?" but "is this statement visually applicable to the entity in the image?"

E-VQA accuracy

31.2 → 36.1

+4.9 over vanilla RAG

InfoSeek accuracy

41.8 → 45.1

+3.3 over vanilla RAG

Harmful RAG effects

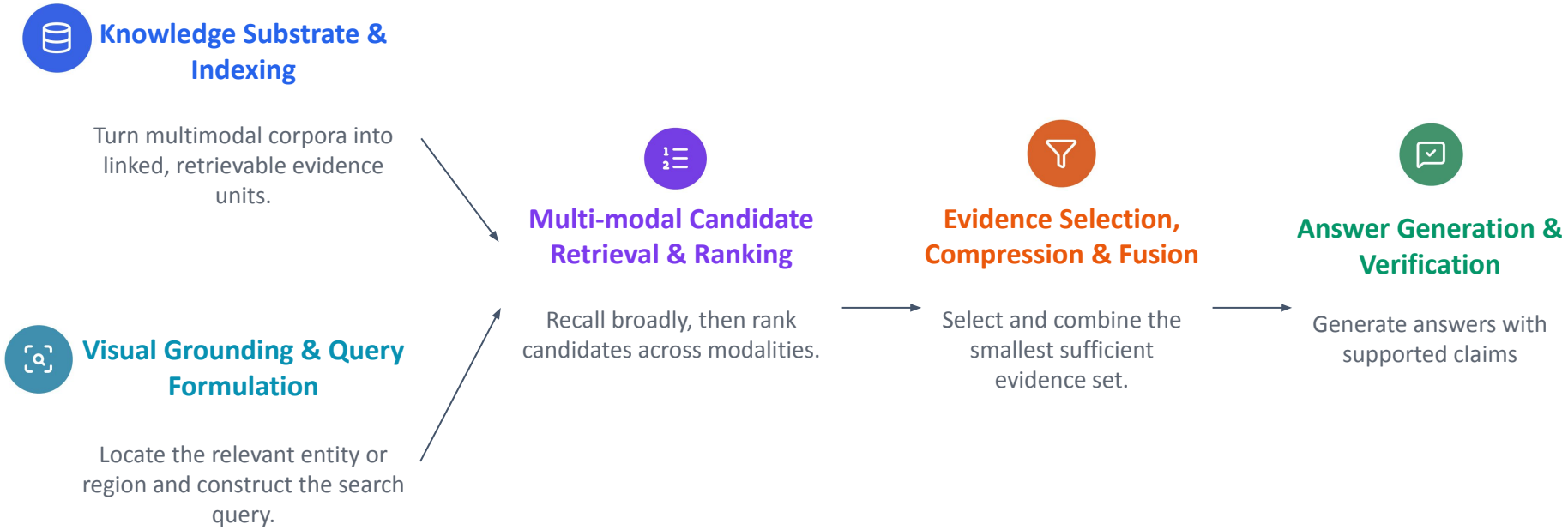
10.53% → 7.69%

less over-trusting bad retrieval

Why it helps: CC-VQA uses visual features as the reference point for conflict analysis, then compresses low-correlation statements and biases decoding toward visually consistent evidence.

Hong et al., CC-VQA: Conflict and Correlation-Aware Method for Mitigating Knowledge Conflict in Knowledge-Based Visual Question Answering, arXiv 2026

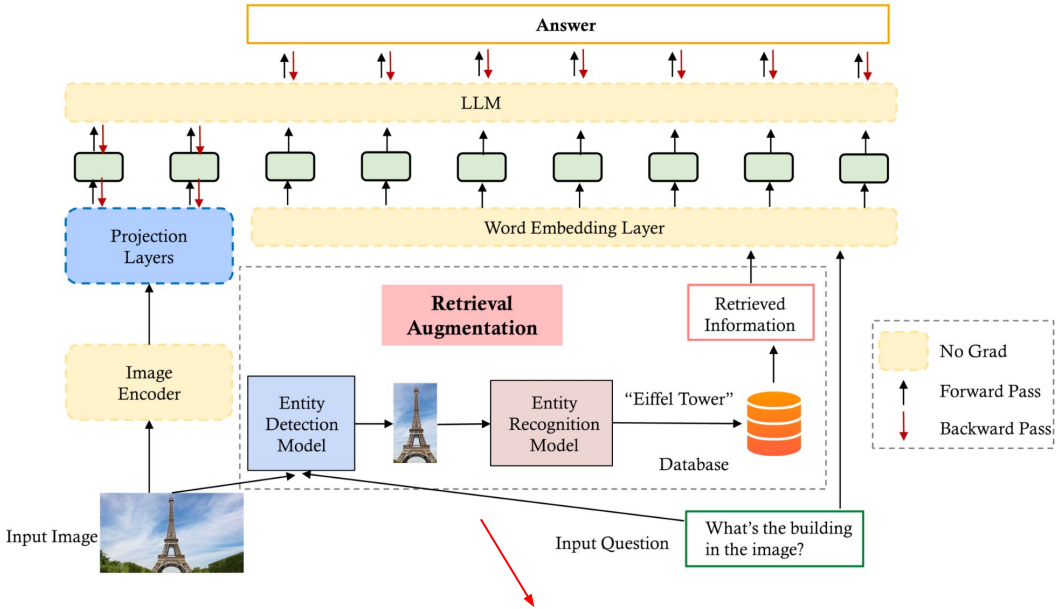
MM-RAG Landscape: What is Unique?



Corpus + grounded query → ranked candidates → compact evidence pack → verified, cited answer

Deep-dive: Modular Visual Entity Grounding

Visual Grounding & Query Formulation



		Accuracy ↑	Hallucination ↓
Head	w/o RA	24.4	75.6
	w/ RA	27.1	72.9
	Δ (100%)	11.1 % ↑	3.6 % ↓
Torso	w/o RA	19.1	80.9
	w/ RA	22.7	77.3
	Δ (100%)	18.8 % ↑	4.4 % ↓
Tail	w/o RA	6.8	93.2
	w/ RA	12.6	87.4
	Δ (100%)	85.3 % ↑	6.2 % ↓

Strength: effective for long-tail entities
Limitation: cascaded errors from incorrect region or entity match; less suitable for multi-object relations

language-guided region extraction before image-based entity recognition.

Qiu et al. SnapNTell: Enhancing Entity-Centric Visual Question Answering with Retrieval Augmented Multimodal LLM, arXiv 2024

MM-RAG Future Opportunities

Grounded Agentic Retrieval

Can the system decide whether to search, what in the image to search for, which modality or tool to use, and when to refine or stop?

Utility-aware and Verifiable Evidence Pipelines

Can the system select the minimum sufficient evidence, resolve conflicts, preserve provenance, and prove that each answer claim depends on that evidence?

Persistent, Personalized, Temporal Multimodal Memory

Can the system safely maintain and retrieve an evolving visual history with time, correction, forgetting, privacy, and user control?

MM-RAG — Recap & Takeaways

- "Any-to-any" modality retrieval and generation
- Unique challenges: visual grounding & query formulation, cross-modal retrieval and ranking, and multimodal evidence integration & reasoning
- Industry SOTA systems only achieved 32% factuality on CRAG-MM benchmark

Takeaways:

- Grounding determines the retrieval ceiling
- Optimize for useful evidence, not just similar evidence
- Next generation will be agentica, verifiable and stateful

04

Conclusions & Future Directions

Tutorial Recap



1. Introduction (40 min): Motivation, Factuality landscape, RAG at a glance

2. RAG Deepdive (110 min)
   - Modularized RAG (50 min)

   - Graph-based RAG (30 min)

   Break
   - Agentic RAG (30 min)

3. Advanced Topics (40 min)
   - Deep research (20 min)

   - Multi-modal RAG (20 min)

4. Conclusions and future directions (15 min)

Recap & Takeaways for Factuality Landscape

What we covered

Knowledge Encoding

🧠 **Knowledge Internalization**

1. Knowledge update
2. Post-training alignment

Knowledge Recall

🛡️ **Hallucination Suppression**

1. Internal Parameter Intervention
2. DPO for Factuality Preference
3. Verification-based
4. Confidence and Calibration

Takeaways

1. Hallucination is because of both **empty shelves** (not internalized) and **lost keys** (recall failure)
2. Many successes, but Insufficient for factuality

   → **RAG (external knowledge) is the rescue**

Modular RAG: Recap & Takeaways

Which EV should I buy in 2026?

Trigger	Rewrite	Retrieve	Post-processing	Generate
Need retrieval?	Search intent?	Evidence source?	Final context?	Synthesis?

What can go wrong?

A bad answer can originate from different modules, so diagnosis must be stage-specific.

Wrong trigger

retrieve too often, or miss retrieval and use stale / ambiguous facts

Wrong rewrite

retriever never sees the right intent

Wrong retrieval

good query, weak evidence

Wrong post-processing

right evidence, wrong final context

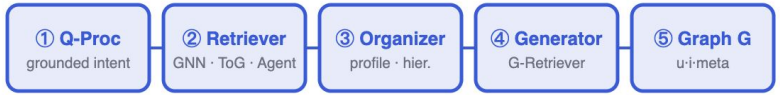
Wrong Answer

Right context, poor answer generation, hallucation

Examples used throughout the deck

Graph-based RAG — Recap & five takeaways

What we covered (the 5-component lens)



Connections to emerging topics:

- **Agent Memory** — Episodic + Semantic + Procedural + Profile, all graphs.
- **Harnesses** — Tool-DAGs retrieved like KG paths.
- **Context Engineering** —Typed, structured prompts assembled from sub-graphs.

Five takeaways

1. **The 5-component pipeline** is your mental scaffold for any GraphRAG project.
2. **Graphs are golden for personalization** — proximity, role, and personalization rationales, each map to a recsys problem.
3. **Mix neural & symbolic.** GNN-RAG, KEQING, ToG, KERAG all win by integration.
4. **Organizer is not optional** — profile summarization & hierarchical aggregation define real-world feasibility.
5. **Future = GraphRAG as system.** Memory, harnesses, context engineering converge to a single graph substrate..

Closing thought. Personalization is fundamentally about the edges that connect you to the world. GraphRAG is the first retrieval paradigm to put those edges front-and-center.

Survey: github.com/Graph-RAG/GraphRAG · KERAG: arxiv.org/abs/2509.04716

Agentic RAG — Recap & 3 takeaways

1. Agentic RAG = RAG as a learned, dynamic decision: whether / what to retrieve, and how to use the results.
2. Two levers: inference-time self-reflection (Self-RAG) vs. RL-learned search policy (Search-R1).
3. The open frontier is efficiency — strong answers under latency/cost budgets (e.g., Stream RAG).

Recommendation: start with adaptive triggering + iterative retrieve–reason; add RL search policies for reasoning-heavy, high-value queries.

Deep Research — Recap & 3 takeaways

1. Deep Research = a long-horizon agent that produces a structured, CITED report — not a one-line answer.
2. A four-stage pipeline: plan → explore → synthesize → report; each stage is its own research problem.
3. Faithful citation and long-horizon planning are the hardest open problems.

Recommendation: for open-ended / report tasks, use a plan-then-explore agent with explicit citation checking (OpenScholar-style self-feedback).

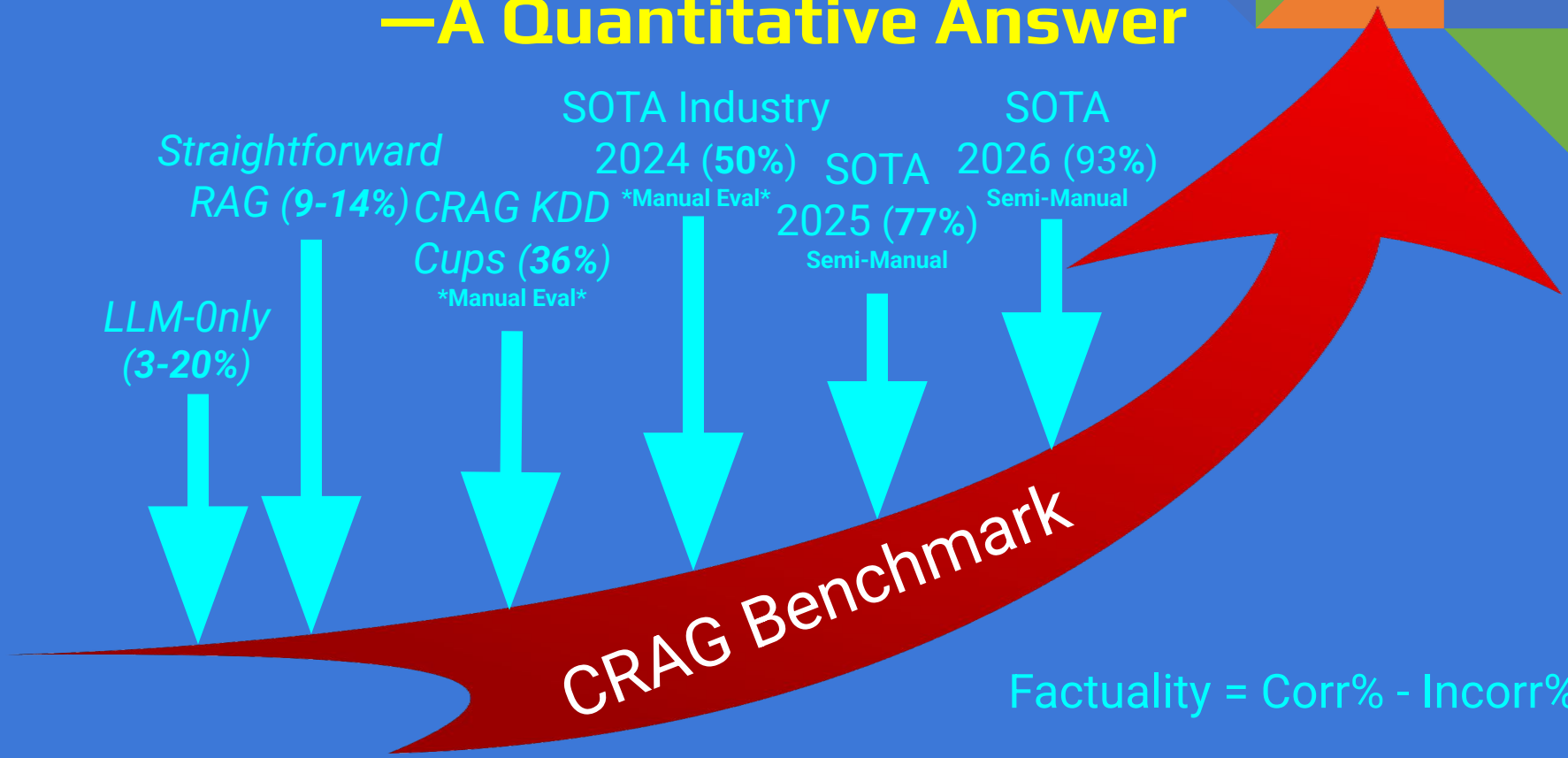
MM-RAG — Recap & Takeaways

- "any-to-any" modality retrieval and generation
- Unique challenges: visual grounding & query formulation, cross-modal retrieval and ranking, and multimodal evidence integration & reasoning
- Industry SOTA systems only achieved 32% factuality on CRAG-MM benchmark

Takeaways:

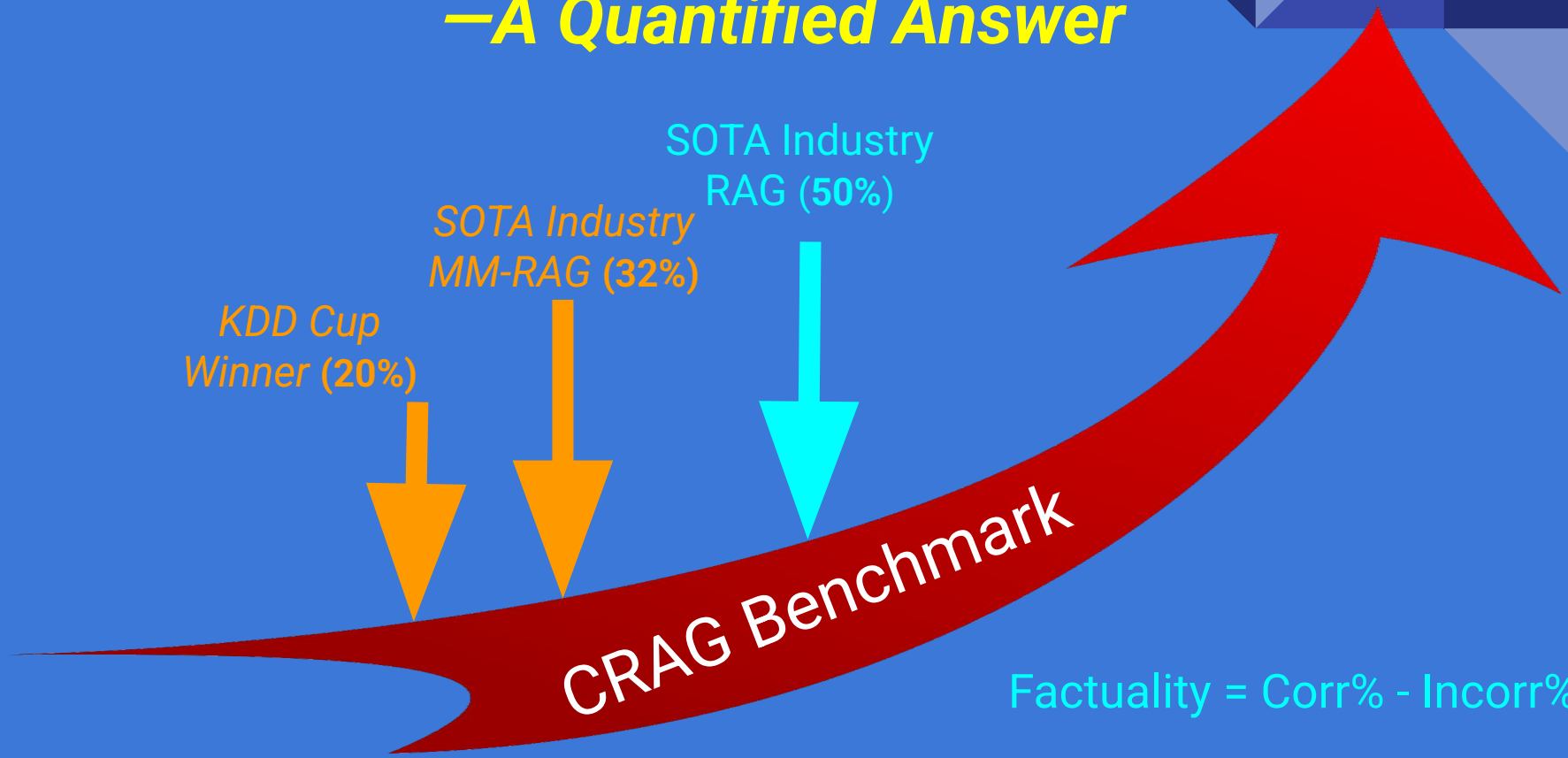
- Grounding determines the retrieval ceiling
- Optimize for useful evidence, not just similar evidence
- Next generation will be agentica, verifiable and stateful

Where Are We in This Journey? —A Quantitative Answer



Xiao Yang, Yifan Ethan Xu, Kai Sun, Jiaqi Wang, et al. KDD Cup CRAG Competition: Systems, Findings, and Learning. IEEE Bulletin, 2024.

Where Are We in This Journey? —A Quantified Answer



Wang and Yang et al. CRAG-MM: Multi-modal Multi-turn Comprehensive RAG Benchmark, KDD 2026

The Evolution and Opportunities

The RAG Evolution

Generation 1
Naive RAG

Simple dense retrieval & generation.
Prone to hallucinations and strict context limits.

Generation 2
Modular & Graph RAG

Integration of query rewriting, hybrid retrieval, re-ranking, and structured knowledge grounding.

Generation 3
Agentic & Any-to-Any

Autonomous tool usage, multi-modal alignment at the pixel level, and deep research loops across heterogeneous data.

Six Opportunities Expose Different RAG Failure Modes

01 LONG TAIL

Tail entities lose signal;

Long-tail retrieval must improve the bottom decile

02 BENCHMARKS

Collect, freeze, test, and rotate;

Benchmarks must move faster than memorization

03 ADVERSARY

Protect ingest-to-act with layered defenses.

04 EFFICIENCY

Replace unbounded loops with a budgeted router;

Spend inference only when evidence is worth it

05 CONFLICT

Separate claim relevance from source trust;

Resolve with recency, authority, support, independence, and uncertainty.

06 MULTILINGUAL

Evaluate query × evidence × answer language.

Future Directions

Deep Research / Agentic RAG

Learn how to acquire, organize, and synthesize evidence over long, adaptive trajectories

Multimodal, Structure-aware RAG

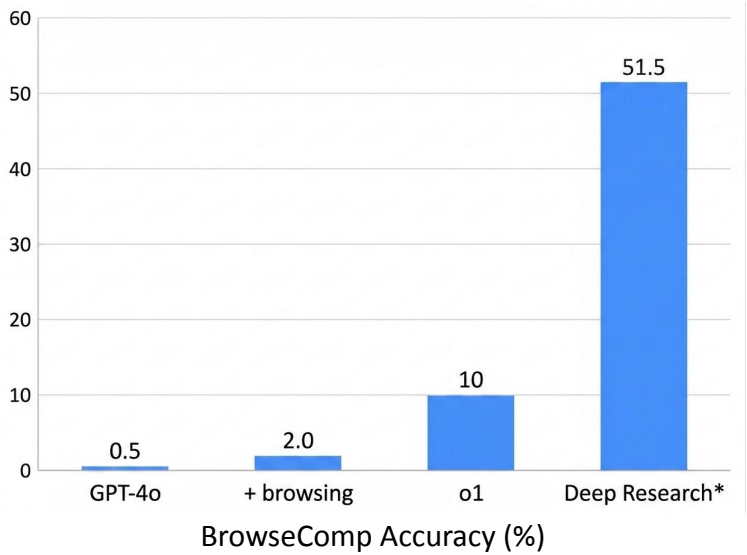
Retrieve and reason over the correct region, table cell, page hierarchy, video interval, or cross-modal evidence set—not just text chunks or whole images.

Verifiable, Uncertainty-aware, and Secure RAG

Make every material claim traceable; detect insufficient, stale, conflicting, or malicious evidence; and abstain or escalate when warranted.

Why RAG Is Becoming Agentic

Static RAG retrieves context once. Complex tasks require the system to manage an evolving evidence-acquisition process.



A browser does not decide what to search next

GPT-4o + browsing reaches 1.9%; the specialized Deep Research system reaches 51.5%.

Many failures start with the wrong search path

Across 64 trials, 14% of tasks were never solved—yet most found evidence after being given the answer.

BrowseComp: a benchmark for browsing agents, OpenAI, 2025

Retrieve the Exact Evidence—Not Just the Document

Unresolved: Choose the right type and granularity; detect conflicts; prove the model used the evidence; control index cost

EVIDENCE CAN LIVE AT MANY LEVELS

- **Corpus**
- **Document**
- **Page / scene**
- **Section / Table / figure**
- **Region / Cell / Frame / Clip**

The question decides the right zoom level.

Design choice	Text-first RAG	Structure-aware multimodal RAG
How it is stored	OCR text split into fixed chunks	Keeps layout, hierarchy, space, and time
What is retrieved	A chunk, page, or whole image	Document → page → region, cell, or time span
How it is chosen	Top-k items scored separately	Pieces that add different, useful facts
How it is cited	Points to a document	Points to a region, cell, or timestamp

Multimodal, Structure-aware RAG

Method	What is different	Reported result	Still unsolved
HiKEY	Uses document hierarchy to route from broad sections to exact evidence	Up to +12.9% recall; +6.8% QA	A wrong early route can hide the right evidence
GranuRAG	Retrieves visual elements directly and links the answer back to them	Up to +29.2% over six baselines	A small element can lose page or document context
Region-R1	Learns when to use the full query image or crop one useful region	Up to +20% conditional Recall@1	The useful region may become clear only after search
Utility selection	Chooses evidence by expected information gain—not similarity alone	Consistent gains with lower compute	Value depends on the model; choosing complementary pieces remains open

Verifiable, Uncertainty-aware, and Secure RAG

Failure	What studies find	What the system should do	Still unresolved
Missing evidence	Answering known questions and rejecting unsupported ones require different skills	Estimate what is missing; search again, add a caveat, or ask a human	No one setting works best for both
Stale evidence	Old information can mislead even when current evidence is present	Track when facts happened, were published, retrieved, and remained valid	Dates can be absent or deceptive
Poisoned evidence	Modular, multimodal, conversational, and agentic RAG can all be attacked	Keep data separate from instructions; check origin, diversity, anomalies, and support	Attackers adapt; stricter defenses can reduce usefulness

Where Integration Breaks for Multi-modal Deep Research

PAPERSCOPE

Papers + text and visuals + multi-step research

>2,000

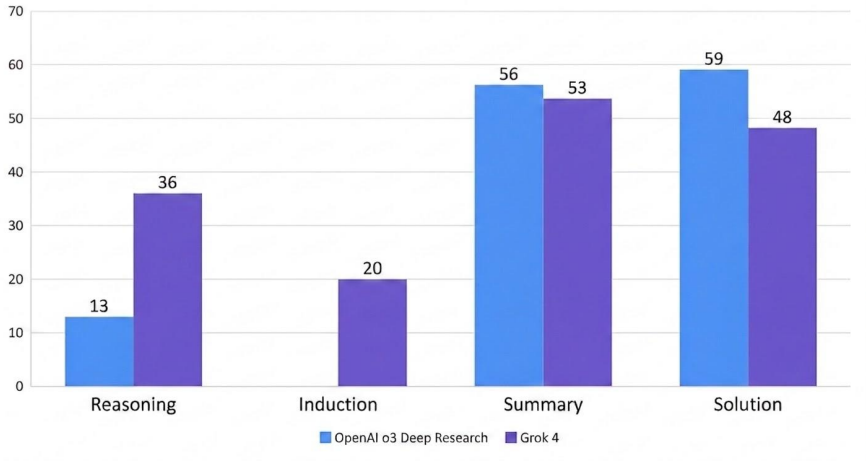
AI papers in a structured graph

2,400

questions across 11 sub-tasks

The Gap

Systems summarize much better than they discover patterns or reason across sources and visuals.



Xiong et al. PaperScope: A Multi-Modal Multi-Document Benchmark for Agentic Deep Research Across Massive Scientific Papers, arXiv 2026

Long-tail retrieval must improve the bottom decile

RAG is most valuable for tail entities, dynamic facts, specialist domains, and minority-language names.

Where systems fail

- Dense encoders blur rare names
- ANN error grows in sparse tail regions
- Aliases and ambiguity defeat one representation

Best first intervention

- Fuse lexical and dense retrieval
- Add entity-conditioned reranking
- Report Recall@20 by popularity decile

Benchmarks must move faster than memorization

Collect + freeze

Snapshot versioned evidence before it enters the test.

Test claim support

Score memory ≠ evidence, claim support, and answer-or-abstain behavior.

Rotate hidden sets

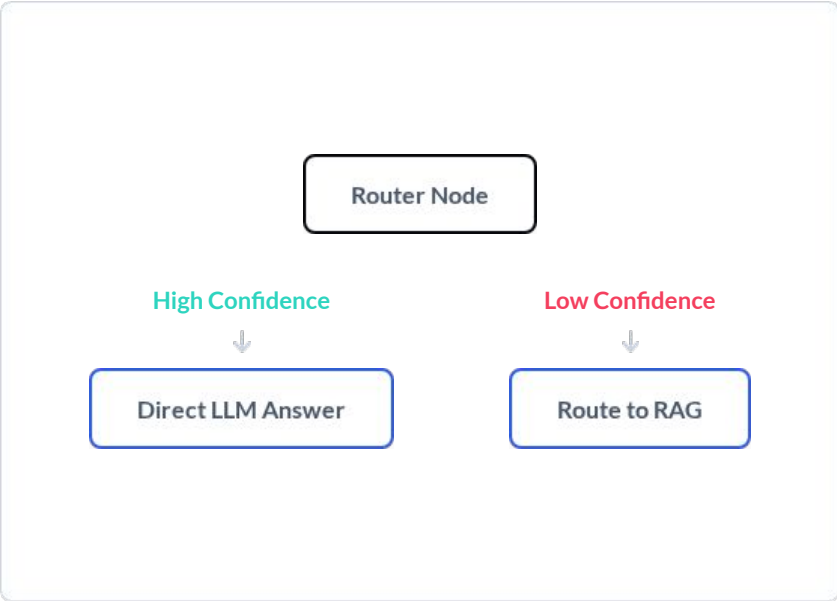
Refresh private slices; recalibrate judges with expert review.

VERSIONED EVIDENCE

CLAIM SUPPORT

LIVING TESTS

PRACTICAL TIPS: DYNAMIC ROUTING



The Anti-Pattern: Always-On Retrieval

Blindly retrieving for every query inflates latency and degrades reasoning by introducing adversarial or irrelevant context.

Adaptive Gating (TARG)

Uses base model confidence to decide when to retrieve, cutting compute costs drastically.

Self-Route Architecture

Routes queries dynamically to either a RAG pipeline or a Long-Context LLM based on the model's self-reflection.

Thank you!

Q & A