Advanced RAG Systems

Week 7: Self-RAG, CRAG, and Agentic Retrieval

PhD Course in Agentic Artificial Intelligence

12-Week Research-Level Course

Learning Objectives

Bloom's Taxonomy Levels Covered

- **Remember**: Define Self-RAG, CRAG, critique tokens (quality markers), relevance scoring
- **Understand**: Explain how self-reflection improves retrieval quality
- **Apply**: Implement a Self-RAG system with critique generation
- **Analyze**: Compare retrieval strategies across different domains
- **Evaluate**: Assess when to use adaptive vs. fixed retrieval
- **Create**: Design a custom agentic RAG pipeline

By end of lecture, you will understand how agents make retrieval decisions.

Limitations of Naive RAG

Fixed Retrieval Problems

- Always retrieves, even when not needed
- No quality check on retrieved documents
- Cannot correct retrieval errors

Common Failure Modes

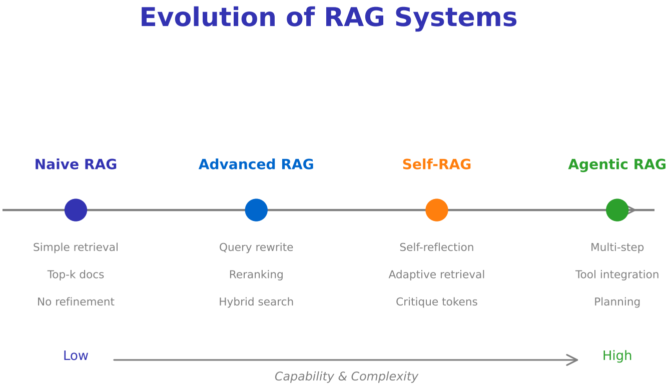
- **Over-retrieval**: Adds noise for simple queries
- **Under-retrieval**: Misses relevant documents
- **Irrelevant context**: Retrieved docs don't match query intent
- **Conflicting sources**: No resolution strategy

The Solution

- Make retrieval adaptive and self-correcting

Advanced RAG adds decision-making to the retrieval process.

Evolution of RAG Systems



Each generation adds more intelligence to the retrieval process.

Self-RAG: Learning to Retrieve

Core Innovation (Asai et al., 2023)

- LLM learns when and what to retrieve through special tokens
- Generates critique tokens to evaluate retrieval quality
- Adaptive: retrieves only when beneficial

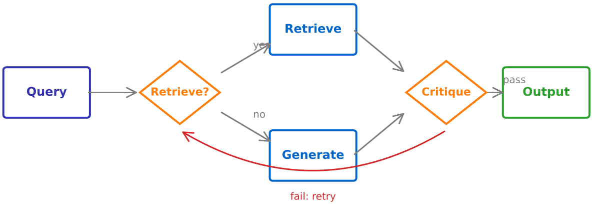
Critique Tokens

- `[Retrieve]`: yes/no – should we retrieve?
- `[IsRel]`: relevant/irrelevant – is document relevant?
- `[IsSup]`: supported/contradicted – does doc support answer?
- `[IsUse]`: useful/not useful – is final output useful?

Self-RAG is trained to generate these tokens alongside normal output.

Self-RAG: Decision Flow

Self-RAG: Adaptive Retrieval



Critique Tokens:

[Retrieve], [IsRel], [IsSup], [IsUse]

The model decides at each step whether to retrieve, generate, or retry.

CRAG: Corrective RAG

Core Innovation (Yan et al., 2024)

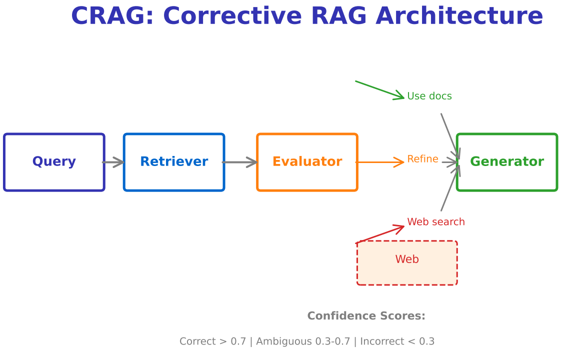
- Evaluate retrieval quality with confidence scores
- Take corrective actions based on evaluation
- Use web search as fallback for low-confidence retrieval

Three-Way Decision

- **Correct** (> 0.7): Use retrieved documents directly
- **Ambiguous** (0.3-0.7): Refine query and re-retrieve
- **Incorrect** (< 0.3): Fall back to web search

CRAG adds a lightweight evaluator to trigger corrective actions.

CRAG: Architecture


CRAG: Corrective RAG Architecture
Use docs
Query
Retriever
Evaluator
Refine
Generator
Web search
Web
Confidence Scores:
Correct > 0.7 | Ambiguous 0.3-0.7 | Incorrect < 0.3

The evaluator routes to different actions based on confidence.

Agentic RAG

Beyond Single-Step Retrieval

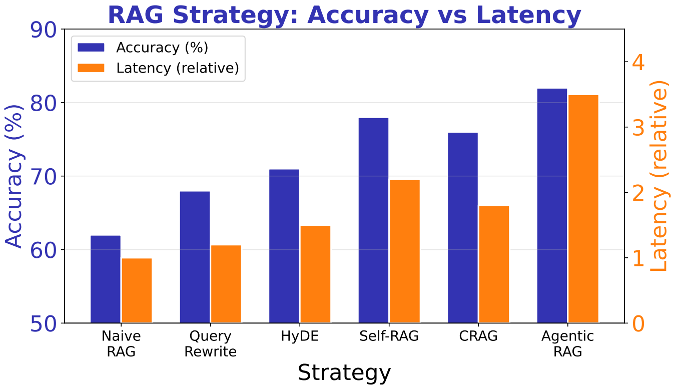
- Agent decides retrieval strategy dynamically
- Multi-step: search, summarize, verify, refine
- Tool integration: calculators, APIs, databases

Key Capabilities

- **Query decomposition**: Break complex queries into sub-queries
- **Source triangulation**: Cross-reference multiple sources
- **Iterative refinement**: Multiple retrieval rounds
- **Citation generation**: Track provenance (source origin)

Agentic RAG = RAG + Planning + Tool Use

Retrieval Strategy Comparison



Higher accuracy comes with increased latency – choose based on use case.

Implementation Patterns

Query Enhancement

- Query rewriting with LLM
- HyDE (Hypothetical Document Embeddings): generate fake answer to embed
- Multi-query: Generate query variations

Retrieval Enhancement

- Hybrid search: Dense (embedding) + sparse (keyword) retrieval
- Reranking: Cross-encoder scoring
- MMR (Maximal Marginal Relevance): diversify results

Generation Enhancement

- Citation injection
- Answer verification
- Confidence calibration

Combine patterns based on accuracy/latency requirements.

RAPTOR: Hierarchical Retrieval

Recursive Abstractive Processing (Sarthi et al., 2024)

- Build hierarchical document tree through summarization
- Cluster documents, summarize clusters, repeat
- Retrieve from multiple abstraction levels

Key Benefits

- Captures both detail and high-level themes
- Handles long documents naturally
- Provides multi-granularity context

Use Cases

- Long document QA
- Multi-document synthesis
- Thematic analysis

RAPTOR trades indexing cost for better retrieval on complex queries.

Required Readings

This Week

- Asai et al. (2023). "Self-RAG: Learning to Retrieve, Generate, and Critique." arXiv:2310.11511
- Yan et al. (2024). "Corrective Retrieval Augmented Generation." arXiv:2401.15884

Supplementary

- Gao et al. (2024). "Retrieval-Augmented Generation for Large Language Models: A Survey." arXiv:2312.10997
- Sarthi et al. (2024). "RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval." arXiv:2401.18059

Start with Self-RAG – it introduces the core concepts of adaptive retrieval.

Summary and Key Takeaways

Key Concepts

- **Self-RAG**: LLM decides when to retrieve with critique tokens
- **CRAG**: Corrective actions based on confidence scores
- **Agentic RAG**: Full agent loop for complex retrieval

Design Principles

- Make retrieval adaptive, not fixed
- Add evaluation/critique steps
- Enable corrective actions

Next Week

- GraphRAG and Knowledge Integration

Advanced RAG = Adaptive Retrieval + Self-Correction + Quality Critique