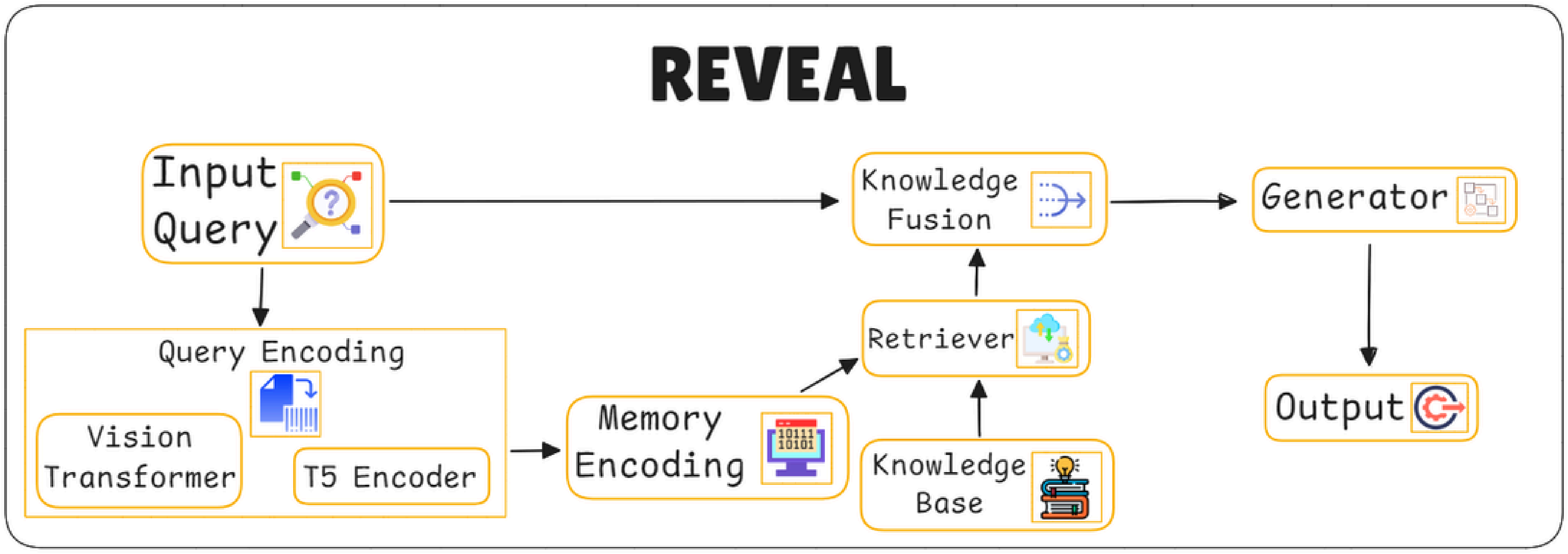
6 Different RAG Techniques

Part 3

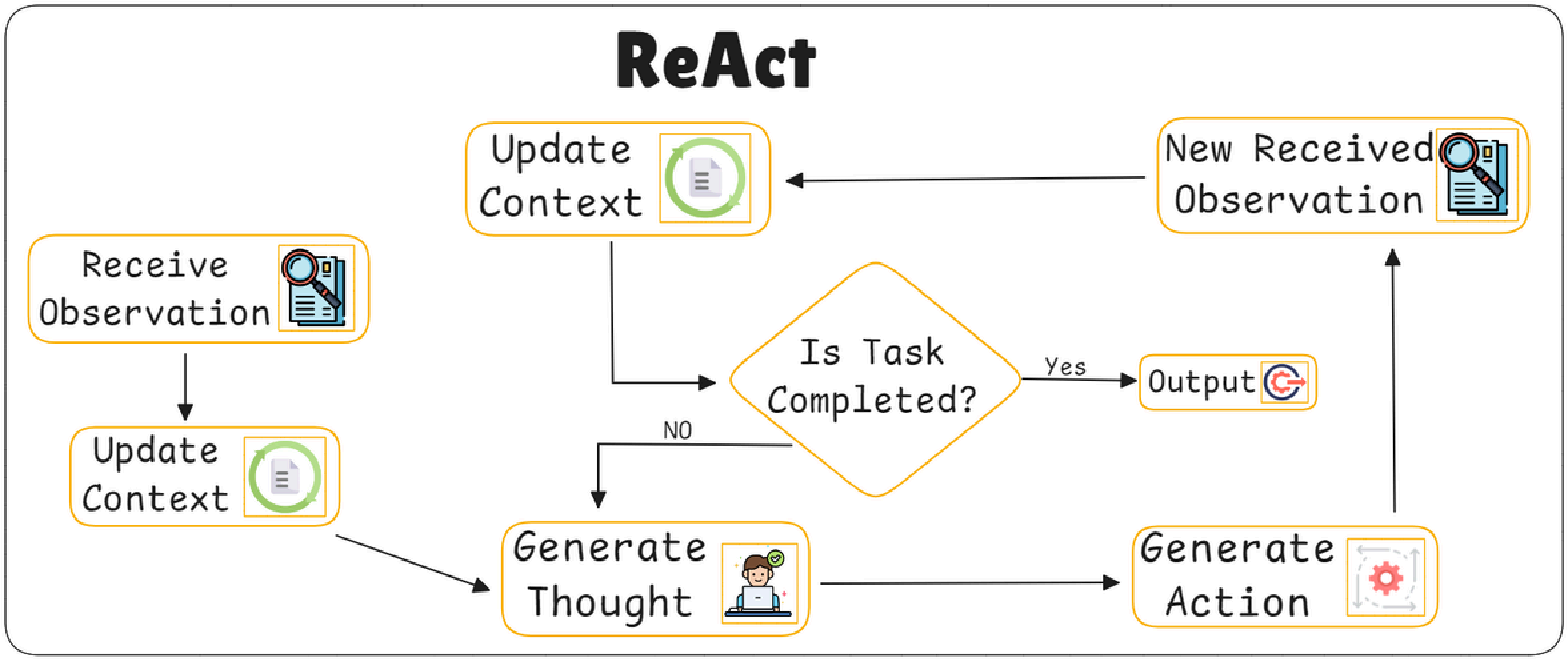
@Bhavishya Pandit

REVEAL: Retrieval-Augmented Visual-Language Model



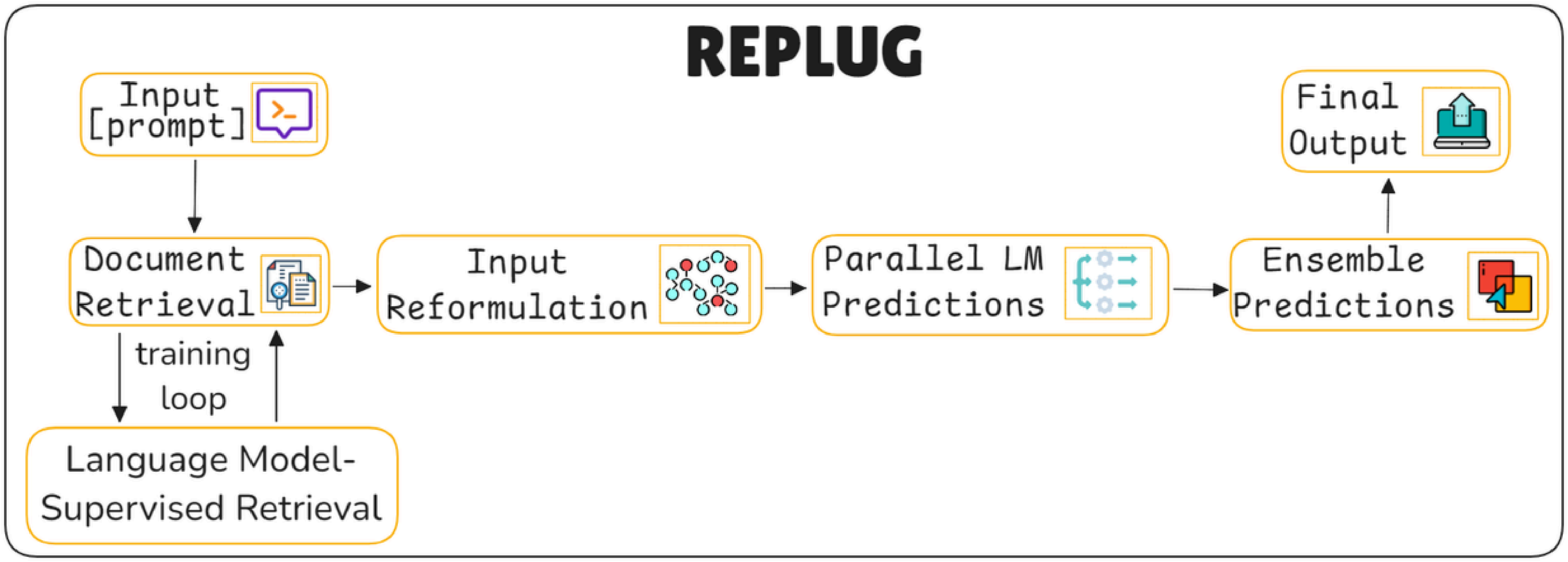
- This technique enhances AI models by **combining reasoning** with task-specific actions and **external knowledge** for decision-making.
- It reduces errors by grounding reasoning in real-world facts, **minimizing inaccuracies & hallucinations**.
- The method produces clear, human-like task-solving steps, **increasing transparency**.
- REVEAL delivers **strong performance across** tasks with fewer training examples, **improving efficiency** and **adaptability**.
- Its flexibility allows for interactive adjustments, making models **more controllable** and **responsive** in real-world applications.

REACT: Retrieval-Enhanced Action generation



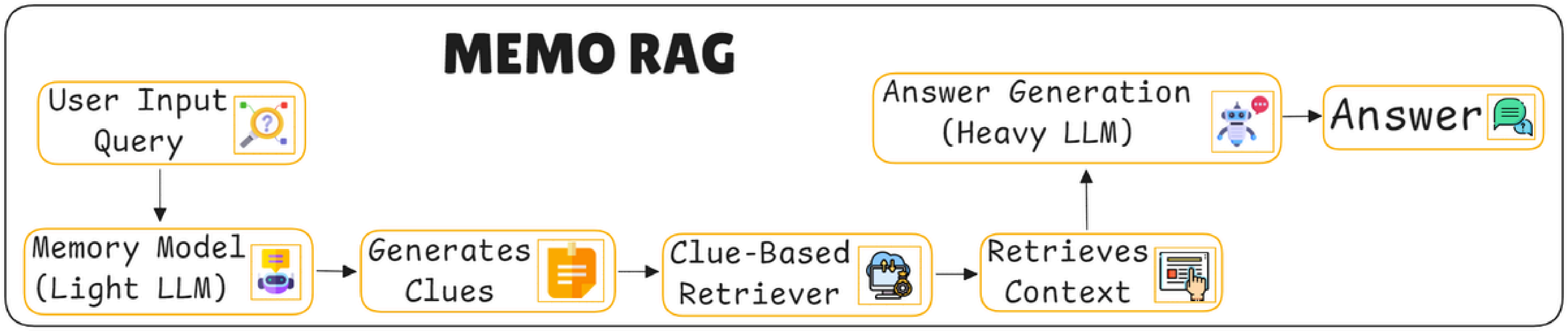
- The ReAct technique **combines reasoning & action**, starting with the model receiving an observation from its environment.
- It updates its context with past actions and thoughts to **maintain situational awareness**.
- The model generates a thought that guides its next action, ensuring **decisions are logical and task-aligned**.
- After executing the action, new feedback helps refine its understanding.
- This blend of reasoning and action r**educes errors, adapts to real-time changes**, and **leads to more transparent, reliable decisions**.

REPLUG: Retrieval Plugin



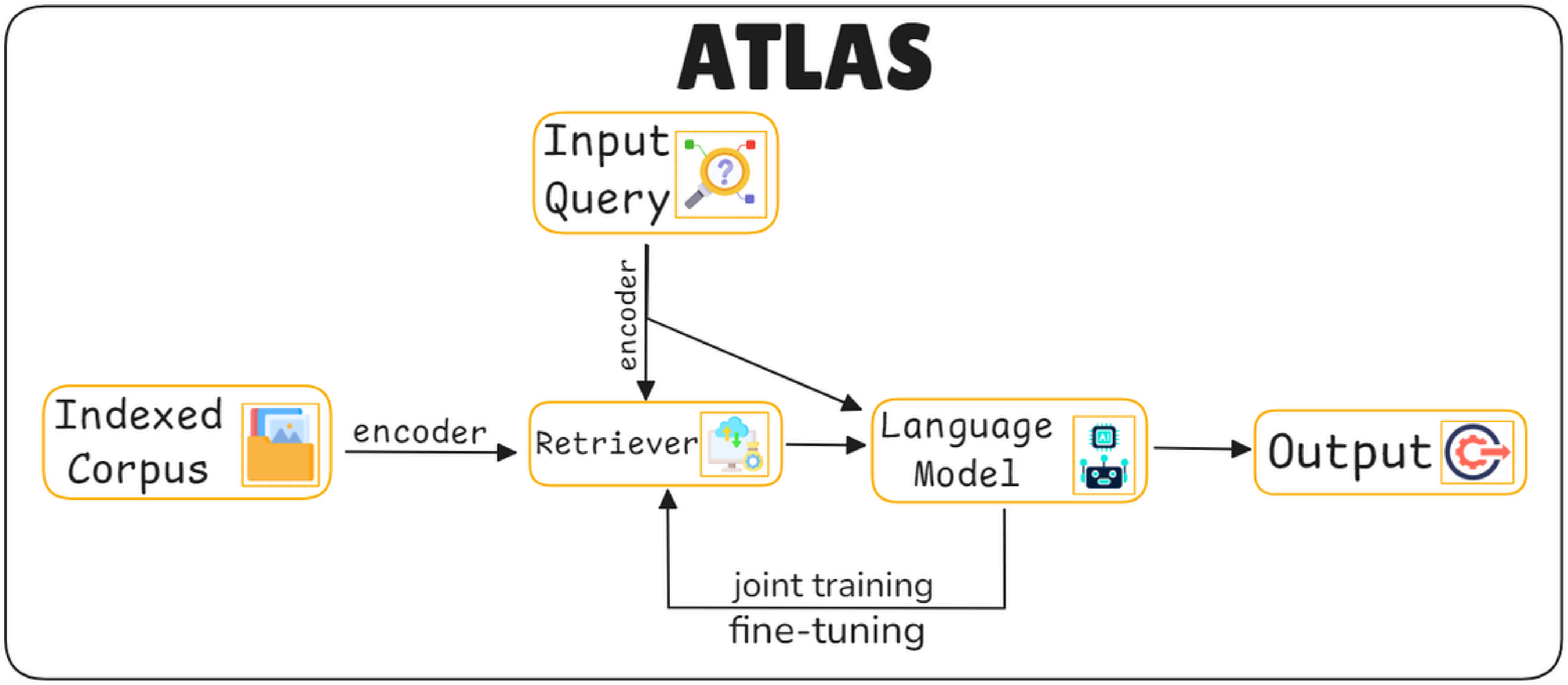
- REPLUG **enhances LLMs** by retrieving relevant external documents to improve predictions.
- It treats the language model as a fixed "**black box**", prepending retrieved information to the input.
- This flexible design can be easily applied to existing models without modifying them.
- By integrating external knowledge, REPLUG **reduces errors** like hallucinations and expands the model's understanding of niche information.
- The retrieval component can be fine-tuned using feedback from the language model, **improving alignment with the model's needs**.

MEMO RAG: Memory-Augmented RAG



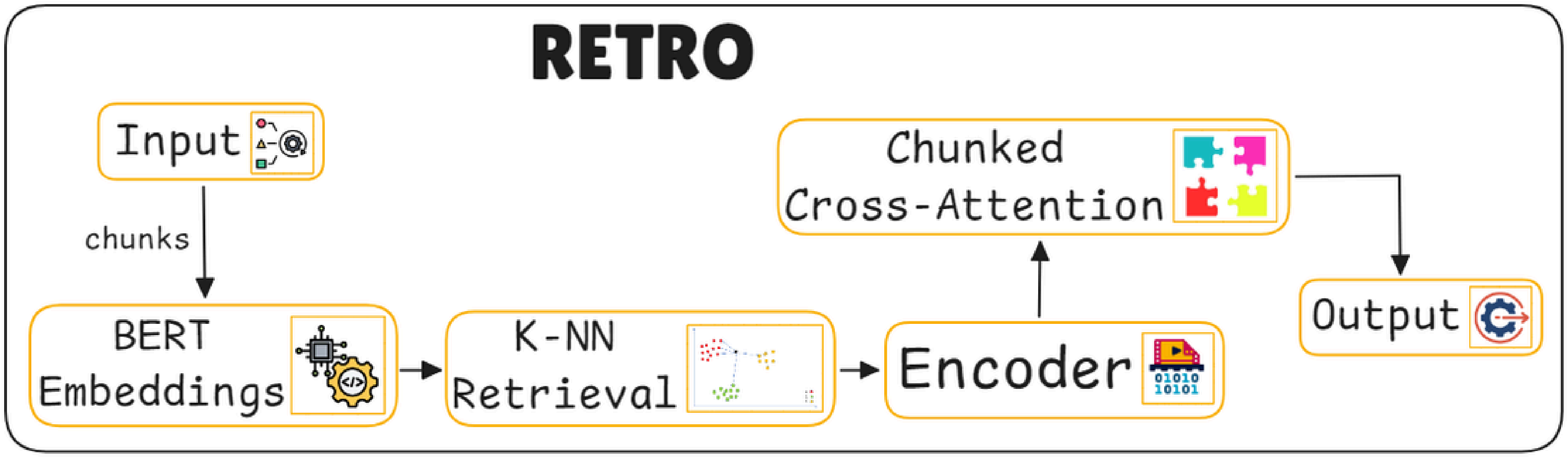
- Memo RAG **combines memory and retrieval** to handle complex queries.
- A memory model generates draft answers that guide the search for external information.
- The retriever then gathers relevant data from databases, which a more powerful language model uses to create a comprehensive final answer.
- This method helps Memo RAG **manage ambiguous queries** and **efficiently process large amounts of information** across various tasks.

ATLAS: Attention-based retrieval Augmented Sequence generation

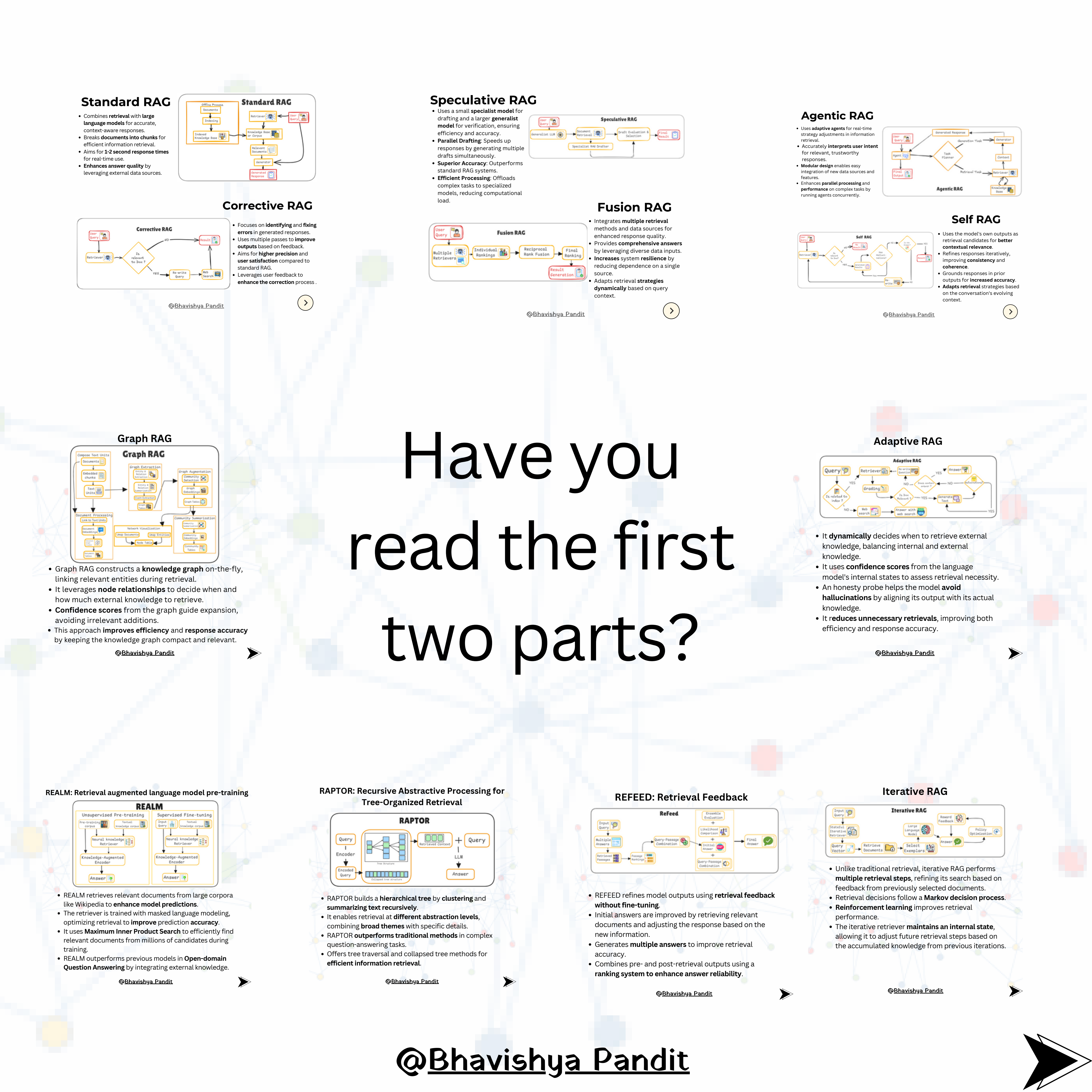


- ATLAS improves language models by retrieving external documents to **boost accuracy** in tasks like question answering.
- It uses a **dual-encoder retriever** to search large text corpora and find the top-K relevant documents for a query.
- These documents are processed by a **Fusion-in-Decoder model**, integrating query and document data to generate the final response.
- With fewer parameters, it **reduces reliance on memorization**, using dynamic document retrieval instead.
- The **document index can be updated without retraining**, keeping its current and effective for knowledge-intensive tasks

RETRO: Retrieval-Enhanced Transformer



- RETRO splits input text into smaller chunks and retrieves relevant information from a large text database.
- Using pre-trained BERT embeddings, it pulls in similar chunks from external data to **enrich context**.
- By integrating these chunks through chunked cross-attention, it **improves predictions** without significantly increasing model size.
- This approach enables better access to external knowledge, enhancing tasks like question answering and text generation.
- It achieves **greater efficiency**, **handling large amounts of information without the heavy computational demands** of larger models.



Standard RAG
Combines retrieval with large language models for accurate, context-aware responses.
Breaks documents into chunks for efficient information retrieval.
Aims for 1-2 second response times for real-time use.
Enhances answer quality by leveraging external data sources.
Corrective RAG
Focuses on identifying and fixing errors in generated responses.
Uses multiple passes to improve outputs based on feedback.
Aims for higher precision and user satisfaction compared to standard RAG.
Leverages user feedback to enhance the correction process.
@Bhavishya Pandit
Speculative RAG
Uses a small specialist model for drafting and a larger generalist model for verification, ensuring efficiency and accuracy.
Parallel Drafting: Speeds up responses by generating multiple drafts simultaneously.
Superior Accuracy: Outperforms standard RAG systems.
Efficient Processing: Offloads complex tasks to specialized models, reducing computational load.
Fusion RAG
Integrates multiple retrieval methods and data sources for enhanced response quality.
Provides comprehensive answers by leveraging diverse data inputs.
Increases system resilience by reducing dependence on a single source.
Adapts retrieval strategies dynamically based on query context.
@Bhavishya Pandit
Agentic RAG
Uses adaptive agents for real-time strategy adjustments in information retrieval.
Accurately interprets user intent for relevant, trustworthy responses.
Modular design enables easy integration of new data sources and features.
Enhances parallel processing and performance on complex tasks by running agents concurrently.
Self RAG
Uses the model's own outputs as retrieval candidates for better contextual relevance.
Refines responses iteratively, improving consistency and coherence.
Grounds responses in prior outputs for increased accuracy.
Adapts retrieval strategies based on the conversation's evolving context.
@Bhavishya Pandit
Graph RAG
Graph RAG constructs a knowledge graph on-the-fly, linking relevant entities during retrieval.
It leverages node relationships to decide when and how much external knowledge to retrieve.
Confidence scores from the graph guide expansion, avoiding irrelevant additions.
This approach improves efficiency and response accuracy by keeping the knowledge graph compact and relevant.
@Bhavishya Pandit
Have you read the first two parts?
Adaptive RAG
It dynamically decides when to retrieve external knowledge, balancing internal and external knowledge.
It uses confidence scores from the language model's internal states to assess retrieval necessity.
An honesty probe helps the model avoid hallucinations by aligning its output with its actual knowledge.
It reduces unnecessary retrievals, improving both efficiency and response accuracy.
@Bhavishya Pandit
REALM: Retrieval augmented language model pre-training
REALM retrieves relevant documents from large corpora like Wikipedia to enhance model predictions.
The retriever is trained with masked language modeling, optimizing retrieval to improve prediction accuracy.
It uses Maximum Inner Product Search to efficiently find relevant documents from millions of candidates during training.
REALM outperforms previous models in Open-domain Question Answering by integrating external knowledge.
@Bhavishya Pandit
RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval
RAPTOR builds a hierarchical tree by clustering and summarizing text recursively.
It enables retrieval at different abstraction levels, combining broad themes with specific details.
RAPTOR outperforms traditional methods in complex question-answering tasks.
Offers tree traversal and collapsed tree methods for efficient information retrieval.
@Bhavishya Pandit
REFEED: Retrieval Feedback
REFEED refines model outputs using retrieval feedback without fine-tuning.
Initial answers are improved by retrieving relevant documents and adjusting the response based on the new information.
Generates multiple answers to improve retrieval accuracy.
Combines pre- and post-retrieval outputs using a ranking system to enhance answer reliability.
@Bhavishya Pandit
Iterative RAG
Unlike traditional retrieval, iterative RAG performs multiple retrieval steps, refining its search based on feedback from previously selected documents.
Retrieval decisions follow a Markov decision process.
Reinforcement learning improves retrieval performance.
The iterative retriever maintains an internal state, allowing it to adjust future retrieval steps based on the accumulated knowledge from previous iterations.
@Bhavishya Pandit
@Bhavishya Pandit

Bhavishya Pandit

Share your thoughts

Save for later

Like this post