

Recent Work in Hierarchical Document Retrieval & HiKEY: Hierarchical Multimodal Retrieval for Open-Domain Document Question Answering

2026. 09.

presenter: Joohyung Yun

PART 1

Motivation



Part 1 Motivation · why trees

Part 2 A Tour of Tree-Based Search · the lineage

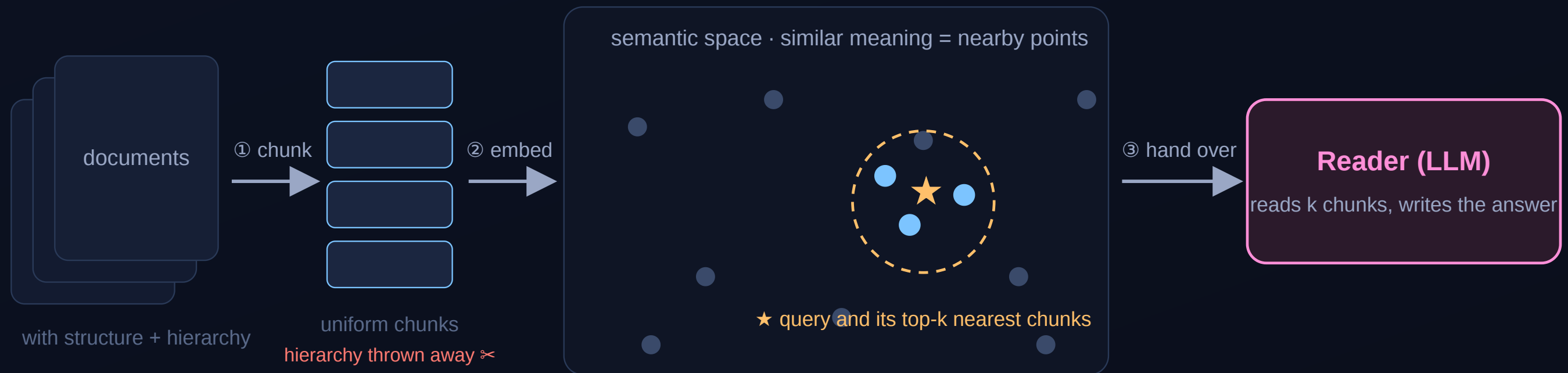
Part 3 HiKEY · deep dive (ACL 2026)

Question Answering over a Document Collection



- **Goal** · find the evidence in a document pile and answer · this example recurs through the deck

Flat Retrieval by Chunking and Similarity Matching



- **A single criterion** · is the chunk semantically close to the question
- Simple and fast · still the most widely used baseline

Failure Modes of Flat Retrieval

① A chunk carries no context

X100 → Maintenance → Battery ✂

"... inspection interval: 6 months ..."

which model? which part?

the severed heading path was its identity

② Evidence is scattered

table: 6 months

§2 caution: halve it
in high heat

they answer only together, yet

worse when the answer spans table,
figure, and text

③ Ambiguity at corpus scale

X100 · battery inspection 6 months

X200 · battery inspection 12 months ?

X300 · battery inspection 3 months

hundreds of docs × hundreds of chunks ·
near-identical sentences compete

- Common cause · structure is discarded and **chunks are judged in isolation**

Searching by the Document Hierarchy

how a person searches

Table of Contents

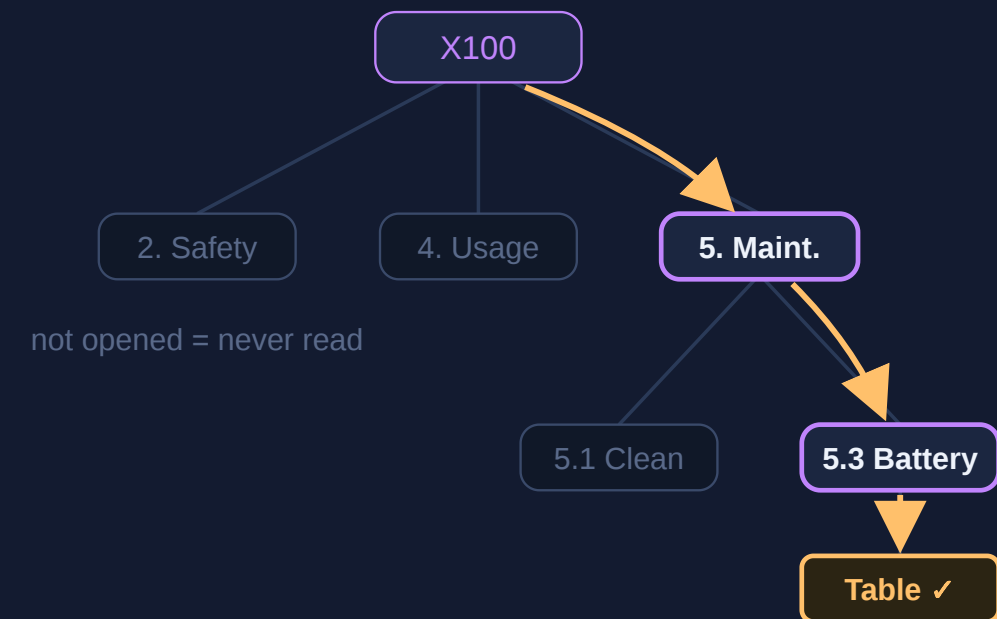
- 1. Introduction
- 2. Safety Precautions
- 3. Installation
- 4. Usage
- 5. Maintenance ←
- 5.3 Battery ←

Table: intervals ← arrival

reads only headings,
rules out whole branches

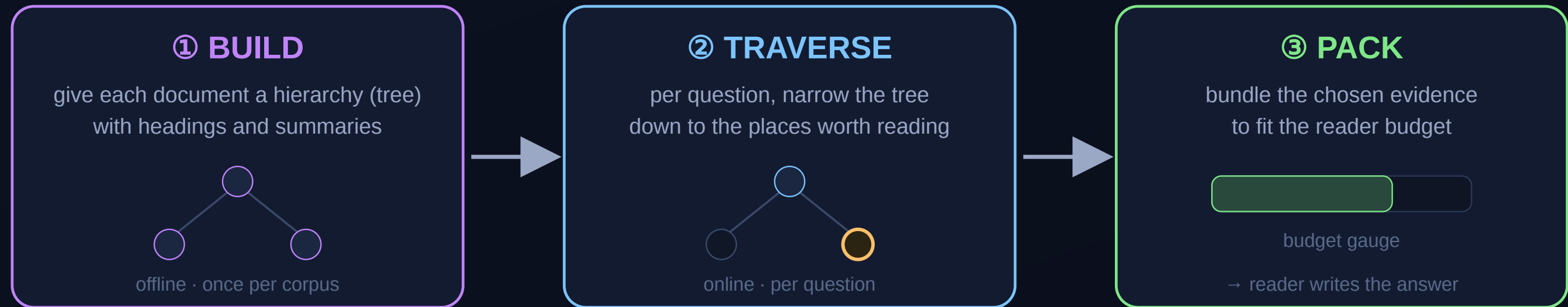
=

the same strategy, mechanized: tree traversal



- Choose branches from headings and summaries alone · reading shrinks sharply
- A ToC is just the document's **hierarchy** · tree-based search = this strategy, mechanized

A Common Frame: Build, Traverse, Pack



every method that follows = **a different way to fill these three boxes**

PART 2

A Tour of Tree-Based Search

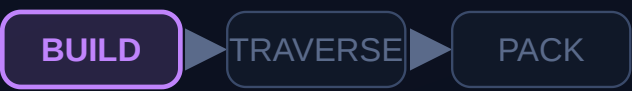


Part 1 Motivation · why trees

Part 2 A Tour of Tree-Based Search · the lineage

Part 3 HiKEY · deep dive (ACL 2026)

Two Design Axes from Build and Traverse

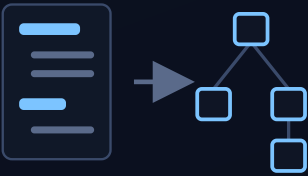


decided at BUILD

Q1 · Where does the hierarchy (tree) come from?

a hierarchy must exist before anyone can traverse it · BUILD has to get one from somewhere

the document's own hierarchy

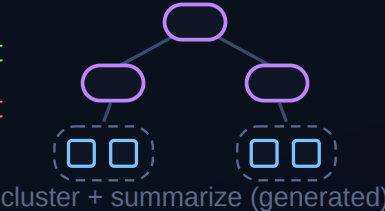


headings become the tree, as-is

- + free · faithful to the author
- only as good as its formatting

machine-built (summaries · learned)

- + works on unstructured text
- generated: cost · can drift

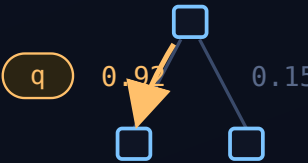


decided at TRAVERSE

Q2 · Who decides where to go next?

at every fork someone must compare the branches with the question and choose

score-based traversal (word overlap · embeddings)

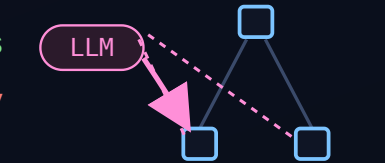


the bigger number wins · no thinking

- + cheap · 0 LLM calls · scales
- matches only, cannot reason

LLM-guided traversal (reads and reasons)

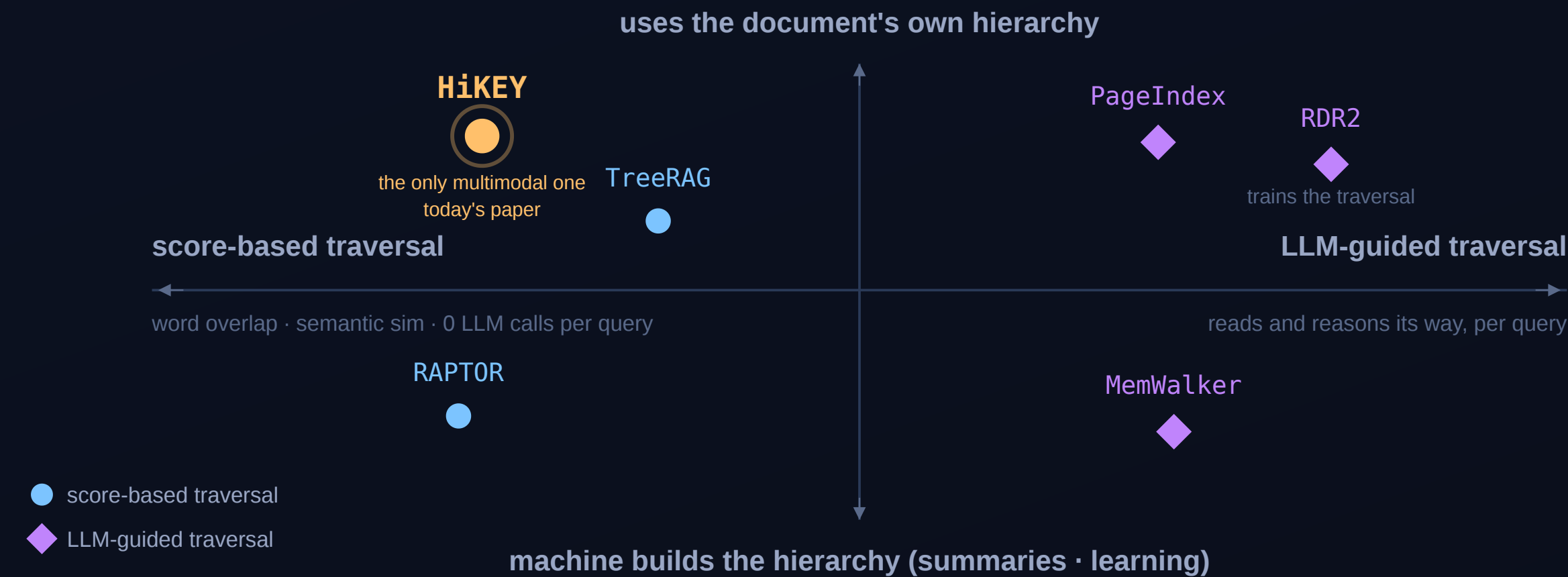
- + reasons · follows references
- calls per query: slow, costly



reads both children, then chooses

each method answers both questions → next slide places all seven on this **2x2 map**

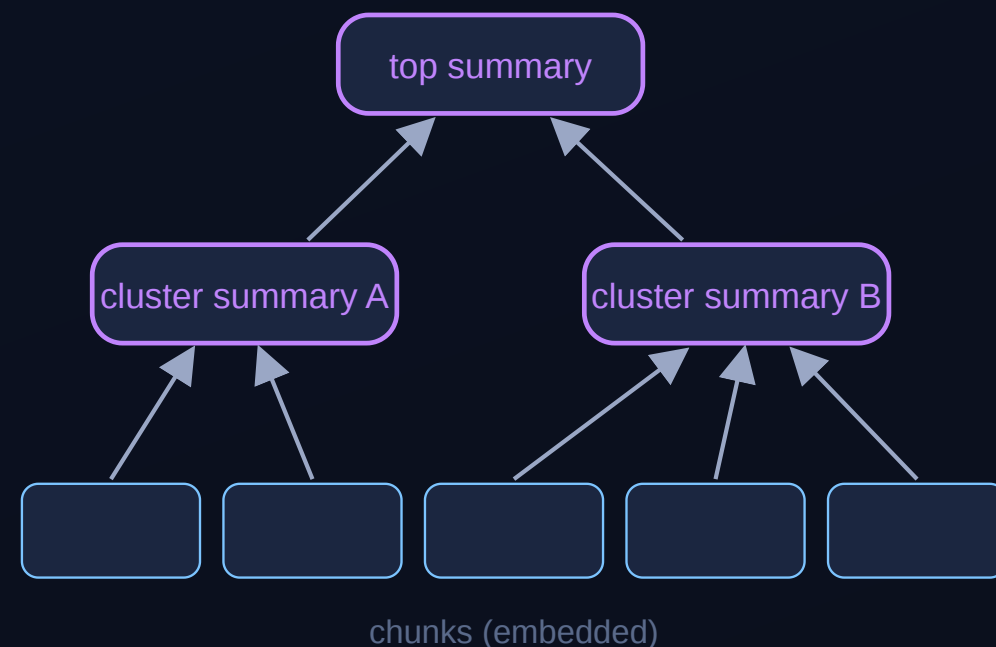
Design Space of Tree-Based Search



RAPTOR: Summary Trees by Recursive Clustering

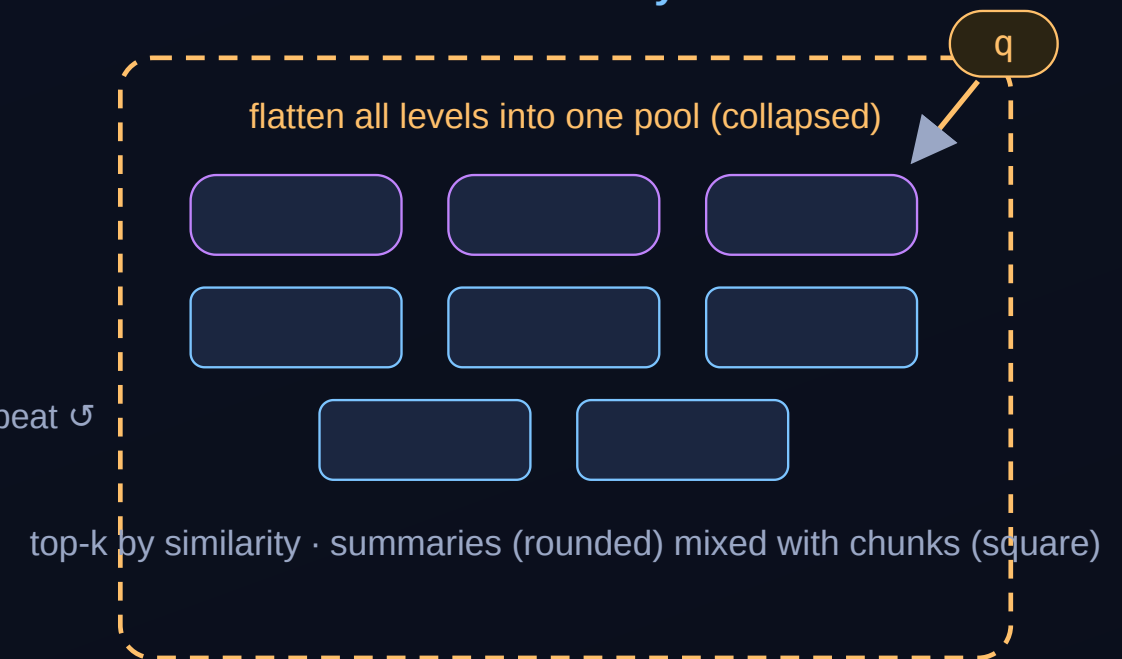
① a tree, no traversal → ② score-based, both ways → ③ LLM-guided instead → ④ traversal learned → ⑤ multimodal

BUILD · the machine writes the hierarchy



cluster similar ones,
LLM summarizes · repeat ↺

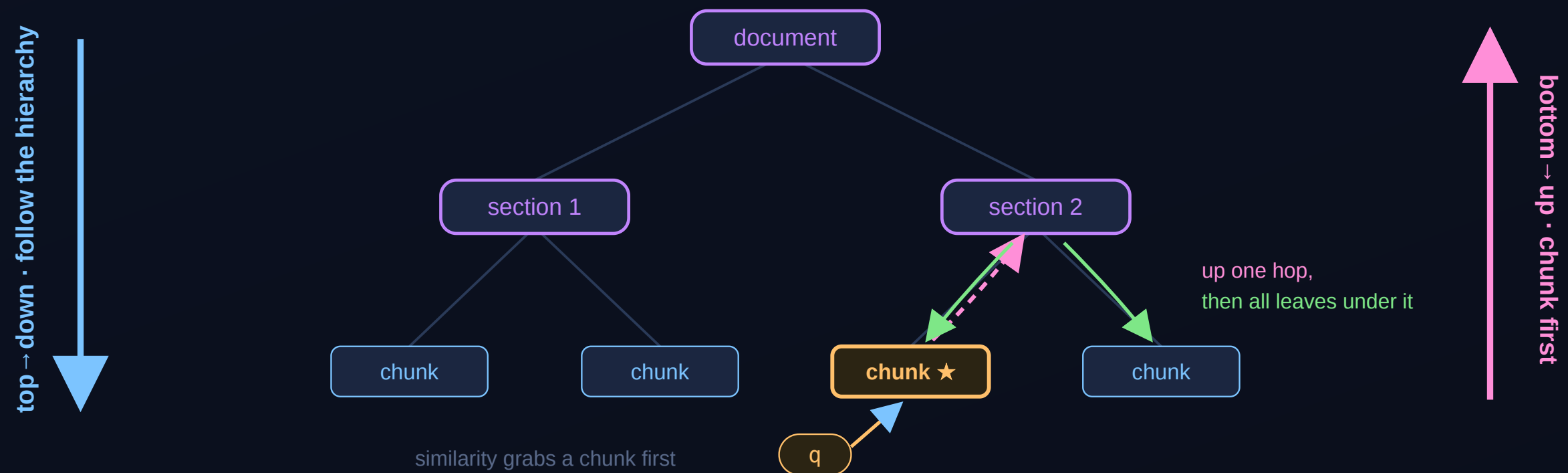
TRAVERSE · it barely traverses



- **The starting point** (ICLR 2024) · summaries at several resolutions become searchable
- Irony · it builds a tree, yet **flat similarity over the pool worked better**

TreeRAG: Bidirectional Tree Traversal

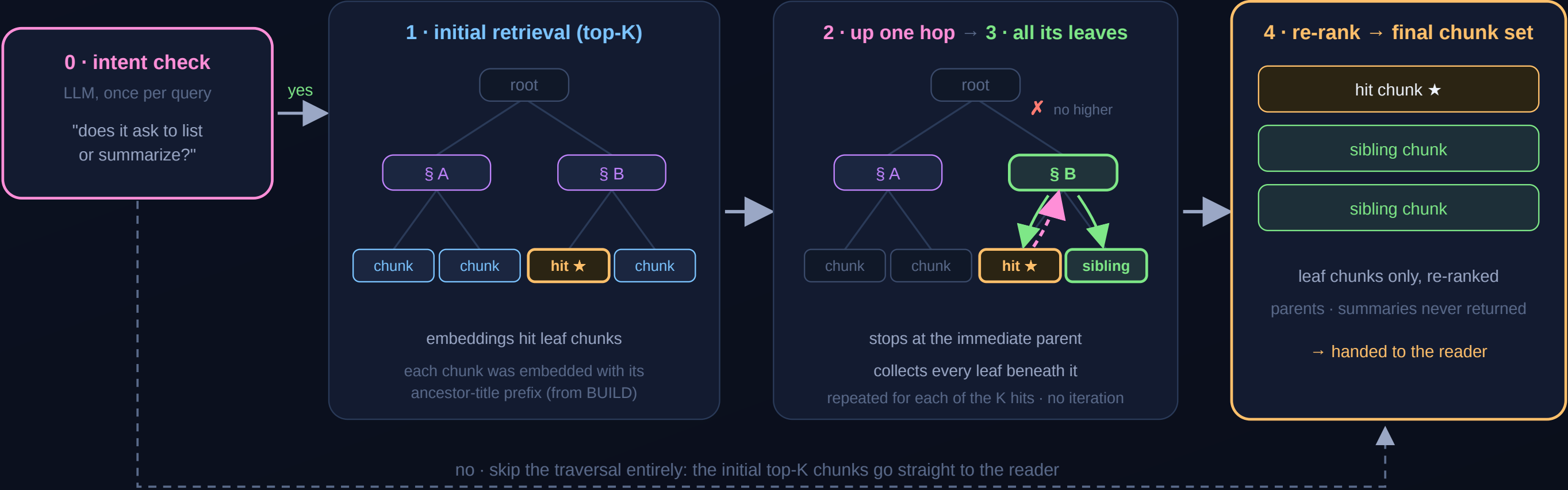
① a tree, no traversal → ② **score-based, both ways** → ③ LLM-guided instead → ④ traversal learned → ⑤ multimodal



- **Two directions** · up from the hit chunk, then down to the leaves under its parent
- Lost context is repaired at **BUILD**: ancestor titles are prefixed into every chunk before embedding

TreeRAG: Retrieval Procedure

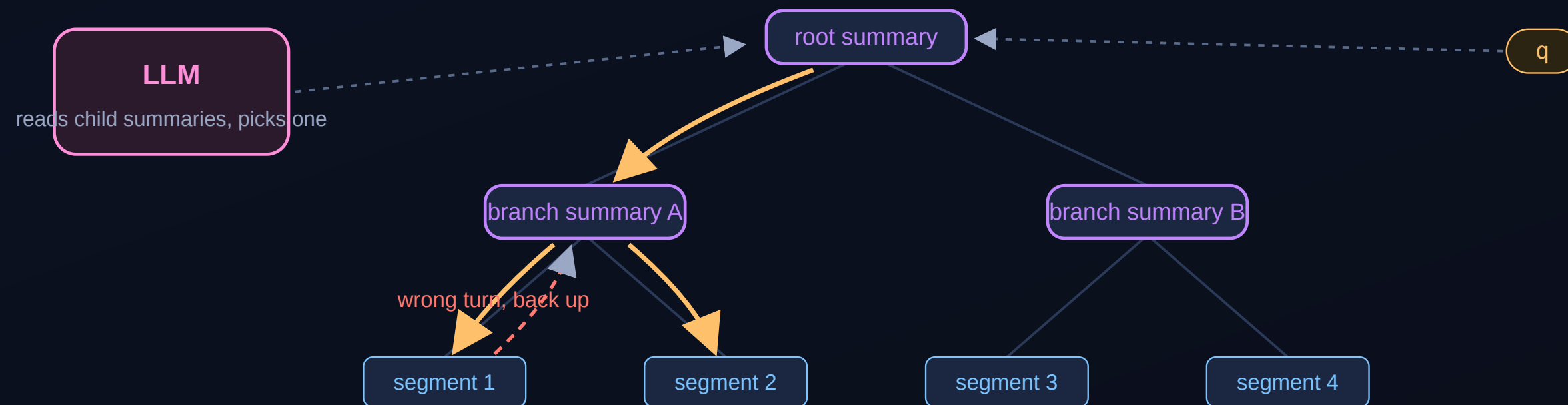
① a tree, no traversal → ② **score-based, both ways** → ③ LLM-guided instead → ④ traversal learned → ⑤ multimodal



a single pass, no iteration: each hit gets **one hop up** + **one leaf expansion**, then re-rank and stop · nothing is revisited

MemWalker: LLM Navigation over Summary Trees

① a tree, no traversal → ② score-based, both ways → ③ **LLM-guided instead** → ④ traversal learned → ⑤ multimodal

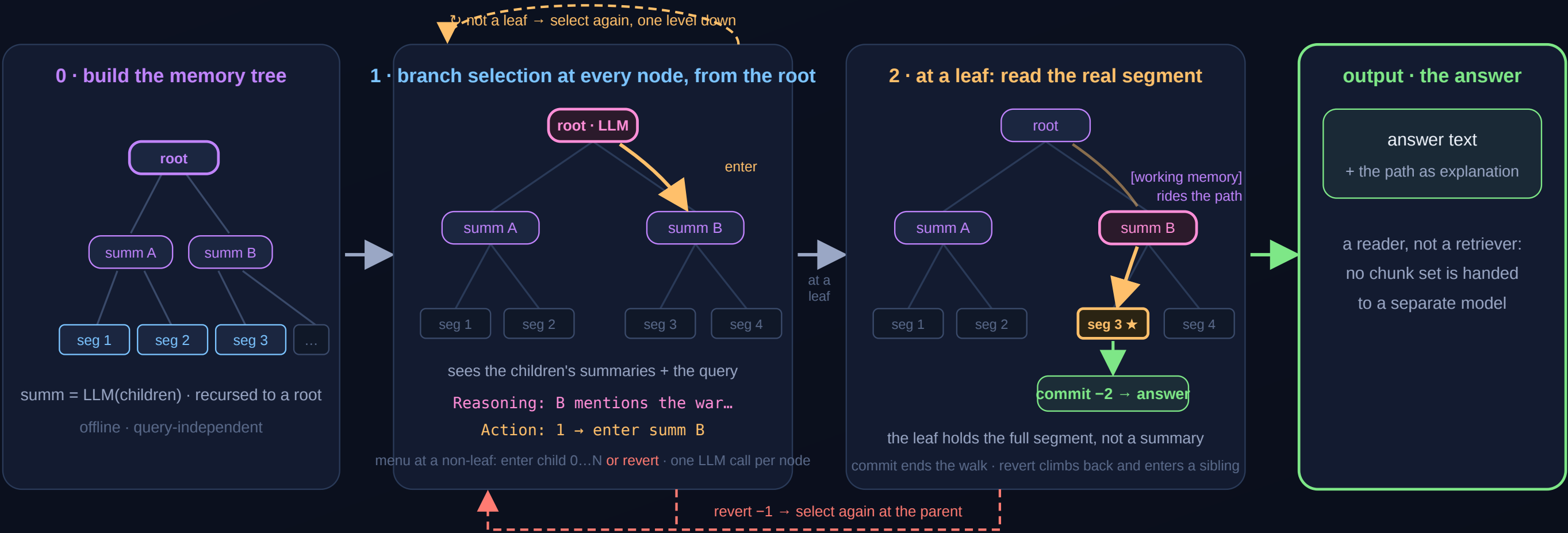


original goal: reading documents longer than the reader's context window

- The traversal goes to an LLM · read summaries, reason, descend · back up on a wrong turn
- The cost · **several LLM calls per question** (one per step)

MemWalker: Navigation Procedure

① a tree, no traversal → ② score-based, both ways → ③ LLM-guided instead → ④ traversal learned → ⑤ multimodal

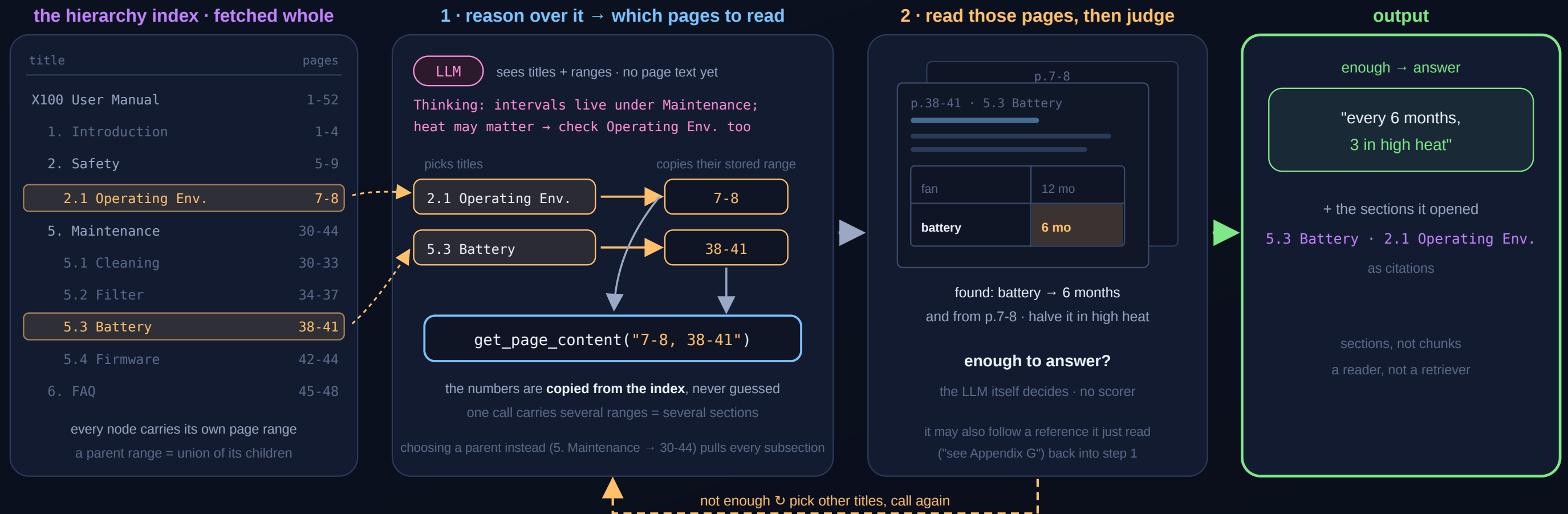


terminates when the LLM commits at a leaf, or after 3 unparsable replies ("no answer") · a single path, no beam

not tool-calling: two zero-shot prompts (branch-selection · leaf) fix an output format · the LLM writes Reasoning / Action as plain text, parsed each step

PageIndex: Retrieval by Reasoning over the Hierarchy

① a tree, no traversal → ② score-based, both ways → ③ **LLM-guided instead** → ④ traversal learned → ⑤ multimodal



state between rounds = the **tool history already in the prompt** · what it read stays there, there is no note store

- **Reasoning, not similarity** · an LLM matches titles, page numbers are looked up · zero embeddings

RDR2: Routing over Document Structure

① a tree, no traversal → ② score-based, both ways → ③ LLM-guided instead → ④ **traversal learned** → ⑤ multimodal



the document tree as it stands now

0: =Wonder Woman in other media=
2: ==Live action==
▼ 3: ===Theatrical releases===
4: Gal Gadot portrays Diana...
► 5: ===Television===
► 8: ==Animation==
► 10: ==Video games==

every heading is always there · only passages open and close
open now: the passage retrieval landed on, plus its siblings
► collapsed heading ▼ opened heading

one step of the router

★ the router is the only trained part
reads the question + the tree on the left
[EXP] 5 open heading 5, close what was open EXPLORE
[ANS] 4 take passage 4 out, the tree is untouched HARVEST
[REF] -1 not chosen this step – would end this document STOP

an action is {tag, node id} · several per step
only [EXP] rewrites the tree →
only [ANS] writes to the pile below ↓

after [EXP] 5 · the same tree, rewritten

0: =Wonder Woman in other media=
2: ==Live action==
► 3: ===Theatrical releases=== ← closed
▼ 5: ===Television=== ← opened
6: Cathy Lee Crosby portrays...
7: Lynda Carter portrays...
► 8: ==Animation==
► 10: ==Video games==

identical headings · one window of text swapped for another
the tree never grows, so the prompt stays a fixed size

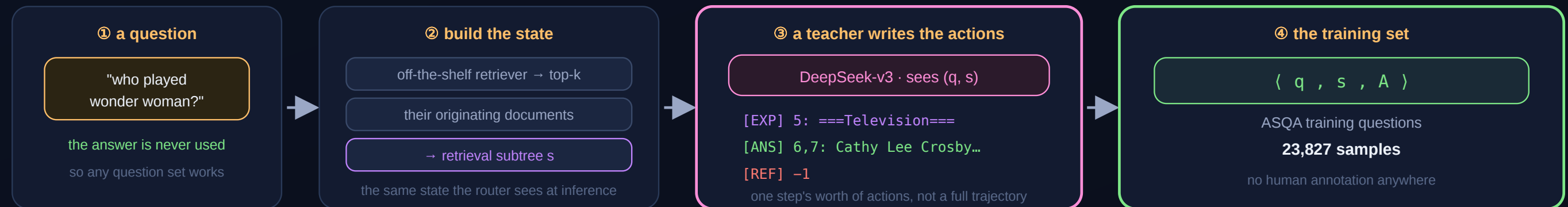


RDR2: Training the Router

① a tree, no traversal → ② score-based, both ways → ③ LLM-guided instead → ④ **traversal learned** → ⑤ multimodal

the obstacle · standard RAG data gives a question and an answer, but **no routing trajectory to imitate**

action curation · manufacture the trajectories from the question alone



trained on single steps, never on whole trajectories

the model learns "what to do in this state"; the iteration is left to inference

the output format carries the node id

"[ACTION] id: text_prefix" – every action is bound to a real node

supervised fine-tuning · plain next-token prediction

$$\mathcal{L} = -\log P(A \mid q, s)$$

cross-entropy on the output tokens only

Llama-3.1-8B-Instruct + LoRA · 3.5 epochs

Comparison of Tree-Based Search Methods

① a tree, no traversal → ② score-based, both ways → ③ LLM-guided instead → ④ traversal learned → ⑤ **multimodal**

Method	Who writes the hierarchy	Who traverses it	LLM calls / query	Table · figure evidence
RAPTOR	machine · pyramid of cluster summaries	no traversal · flat similarity over all levels	0	✗
TreeRAG	the document's own hierarchy	similarity · bidirectional (up + down)	0	✗
MemWalker	machine · segment summaries	LLM reasoning + backtracking	per step	✗
PageIndex	the document's own hierarchy	LLM reasoning (no vectors)	iterative	✗
RDR2	the document's own hierarchy	a trained LLM navigator	calls	✗
HiKEY	the document's own hierarchy (auto-parsed)	scores, two stages · pick docs → pick sections	0 (reader only)	✓ first-class

– One blank remains · documents whose **key evidence sits in tables and figures** → Part 3

PART 3

HiKEY: Hierarchical Multimodal Retrieval



Part 1 Motivation · why trees

Part 2 A Tour of Tree-Based Search · the lineage

Part 3 HiKEY · deep dive (ACL 2026)

Two Bottlenecks in Industrial Document QA

① Can't find it wrong document from the start

Q. X100 battery inspection interval?

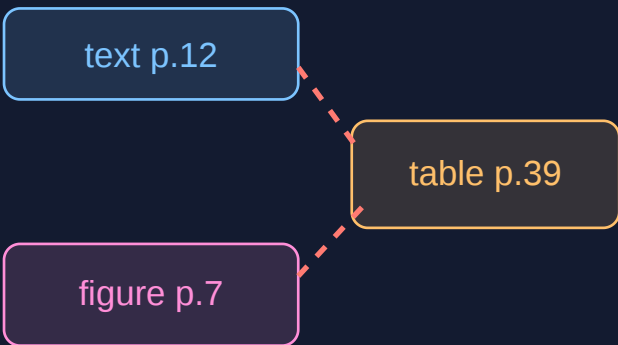


thousands of docs · similar titles · similar senten

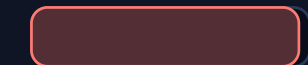
when a question needs several docs, **missing one = failure**

② Can't assemble it evidence fragments even when found

even inside one document



reader budget

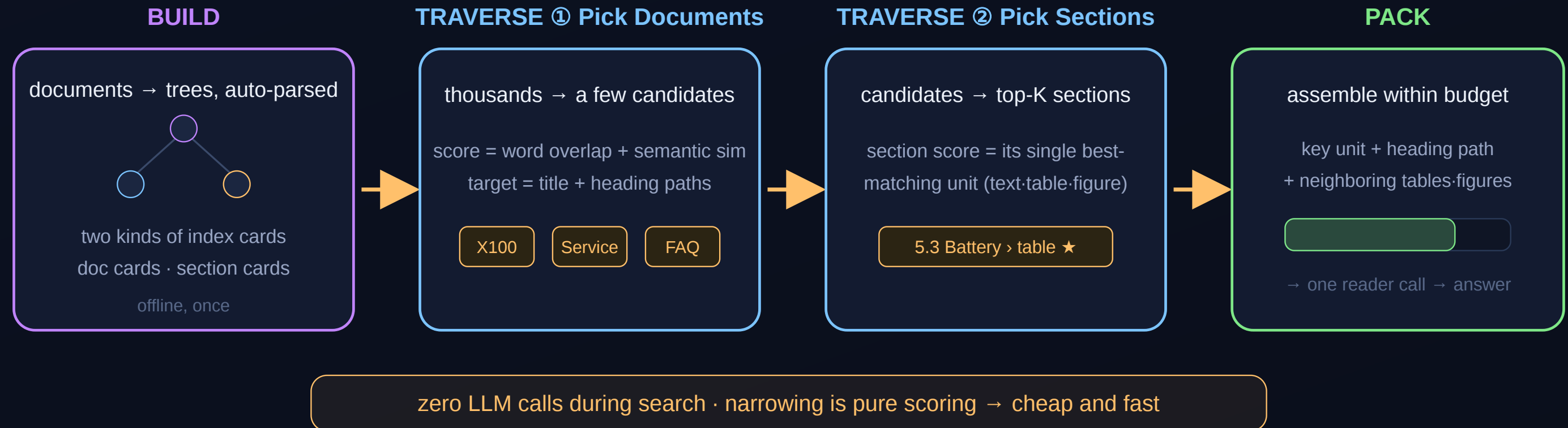


all of it overflows,
cutting loses evidence

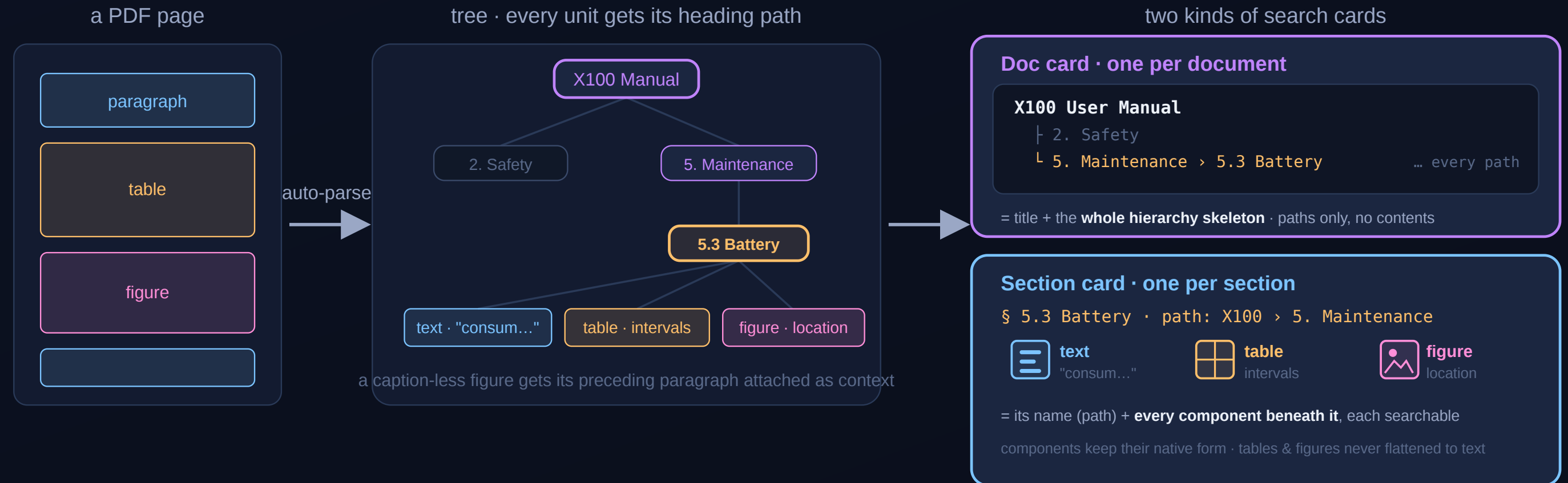
industrial docs = long PDFs · **tables and figures carry the evidence** · budget is finite

- HiKEY's diagnosis · both are symptoms of **search that ignores document structure**

Overview of HiKEY

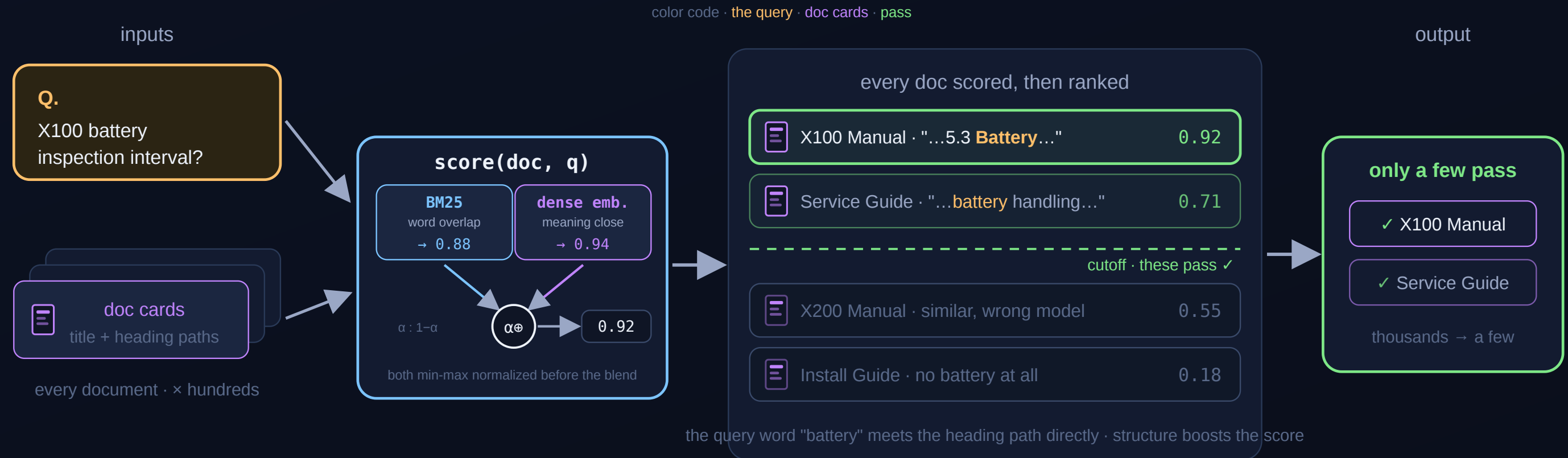


Building Document Trees and Index Cards



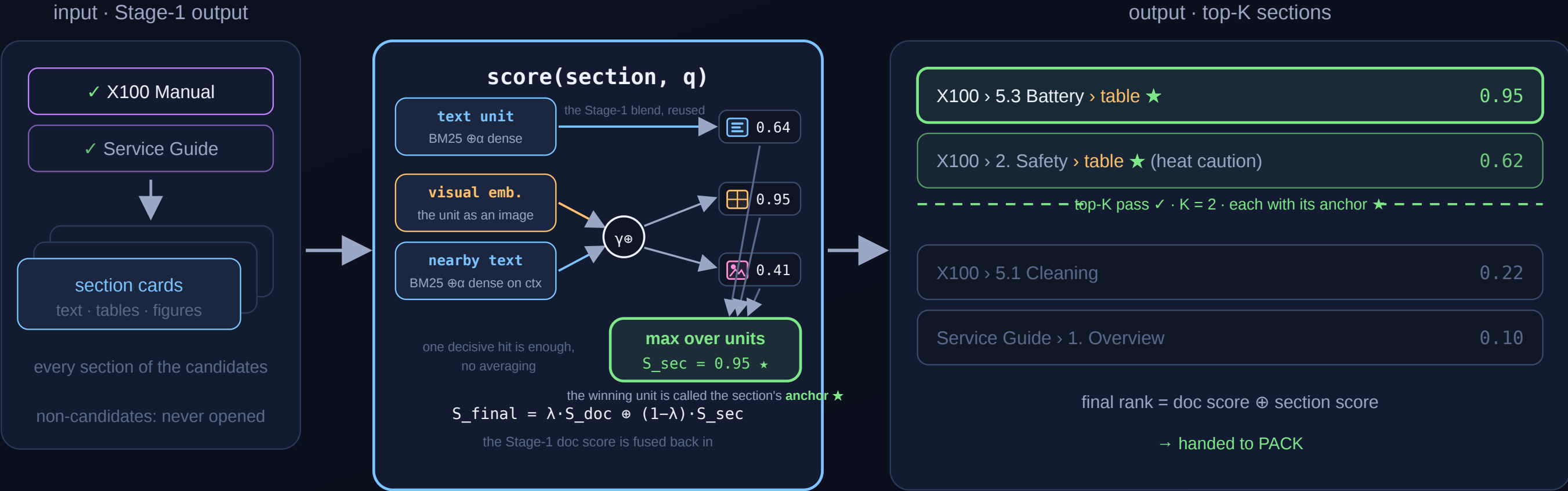
- **Structure becomes a first-class search signal** · heading paths live inside the cards
- **Tables and figures become first-class targets** · the blank in the lineage, filled

Traversal Stage 1: Selecting Candidate Documents

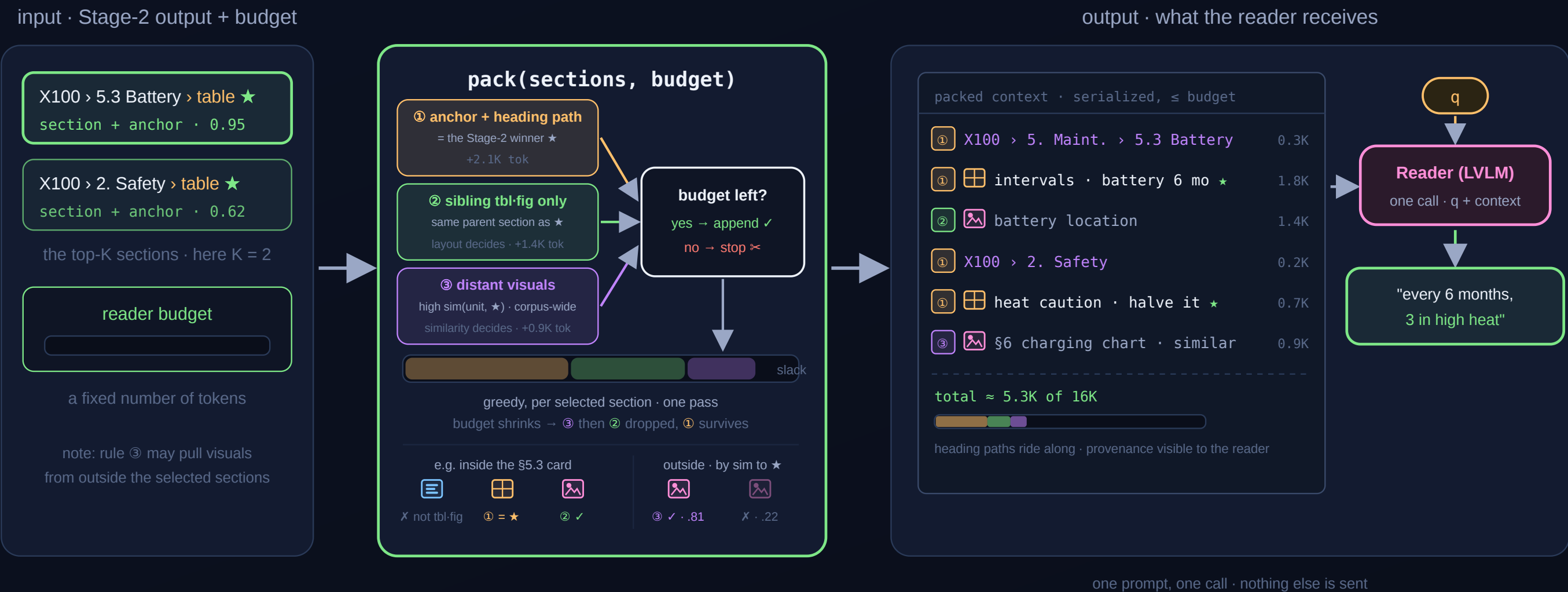


- Thousands → a few · the cheapest computation prunes the largest space

Traversal Stage 2: Scoring Sections



Packing Evidence under a Reader Budget



HiKEY versus the Surveyed Methods

Their bet	What they claimed	What HiKEY does instead
RAPTOR · generated summary pyramid	multi-level summaries make retrieval smarter	uses the hierarchy the document already has · nothing generated on the retrieval path
TreeRAG · bidirectional traversal	traversing upward recovers the lost context	context is restored at PACK : ancestor headings ship with the evidence, no traversal needed
MemWalker · PageIndex · LLM navigation	reasoning picks the right branch	same coarse-to-fine narrowing by plain scores · zero LLM calls per query
RDR2 · learned routing	training makes the traversal accurate	no training · works out of the box on a new corpus
left open by all of them	tables and figures as evidence	first-class multimodal units + budget-aware packing · the genuinely new part

Every ingredient looks simple because it is borrowed. The contribution is **where the pieces sit**: structure enters at BUILD and PACK instead of a smarter TRAVERSAL, and the whole system runs on **one reader call**. The experiments have to show the combination wins.

PART 3

HiKEY: Experimental Results



Part 1 Motivation · why trees

Part 2 A Tour of Tree-Based Search · the lineage

Part 3 HiKEY · deep dive (ACL 2026)

Experimental Setup

Benchmarks · corpus-level: one joint index

M3DocVQA

multi-page PDFs · evidence in **text, tables, figures** · multi-hop

FRAMES-PDF

Wikipedia questions re-rendered as PDFs · one query needs **several documents**

Competitors · 14 baselines

text chunks

page / length / structural chunking · LumberChunker · MultiDocFusion ...

text graph

RAPTOR (from Part 2) · HopRAG

page-level MM

M3DocRAG · VDocRAG

MM graph

MoLoRAG · SimpleDoc

Metrics

retrieval

Recall · MRR · Hit · **All** (every needed doc found · miss one = fail)

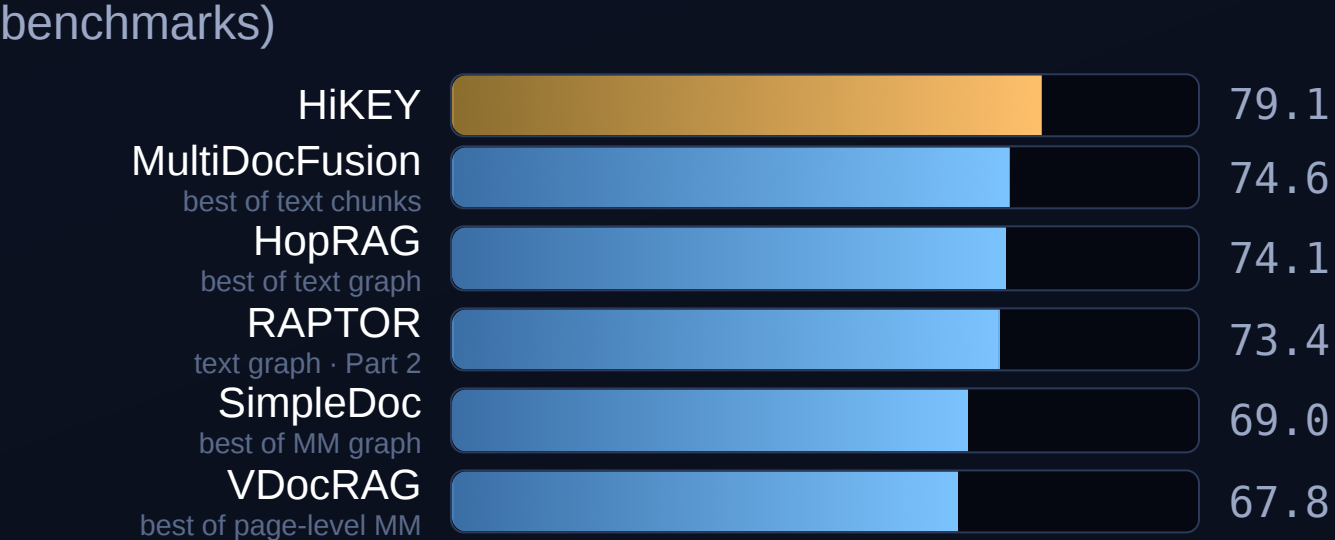
QA

EM (exact match) · ANLS · ROUGE-L · METEOR

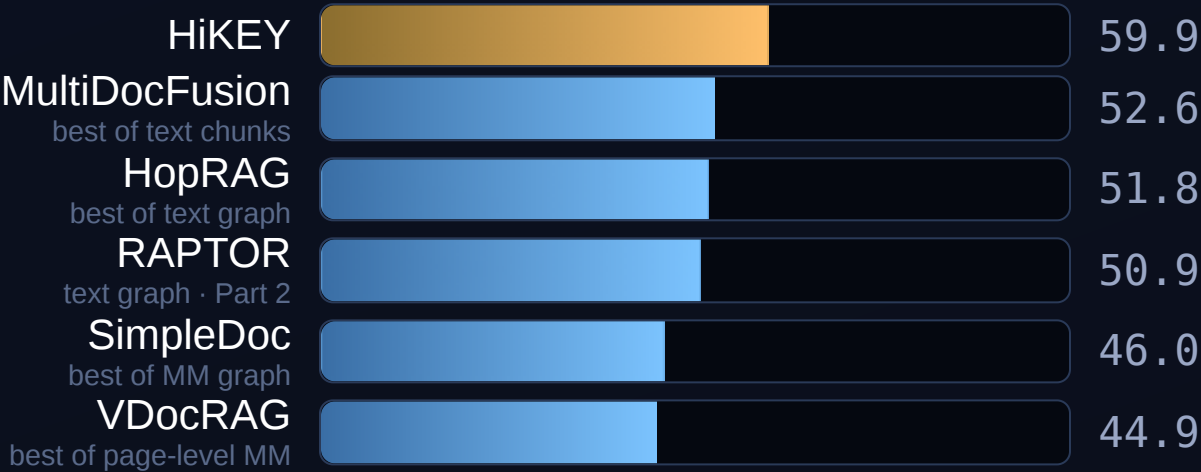
Fair control · every method gets the **same reader** (Qwen2.5-VL 7B), the **same 16K token budget**, and the **same encoders** (keyword + text embedding + visual embedding) → a difference between rows is a difference of method, not of budget or encoder

Document-Level Retrieval Accuracy

Did it find the right documents · avg recall (mean of two benchmarks)



Did it find all of them · All (miss one = fail)



one bar per family: the strongest of the 14 baselines (+ RAPTOR from Part 2) · full 14-row tables are in the paper

On the multi-document benchmark (FRAMES), All improves by **+11.4pt** · hierarchy-path routing misses fewer of the scattered documents

Token-Budget Robustness and End-to-End QA

Does it survive a tighter budget document recall@10

Budget	HiKEY	MultiDocFusion	page methods
0.5K tokens	78.8	74.3	cannot adapt (whole pages)
1K tokens	79.1	74.6	
2K tokens	81.3	76.8	

the **Pack stage** at work · still best at 1/32 of the full budget

Do the answers improve EM · exact match, mean of two benchmarks

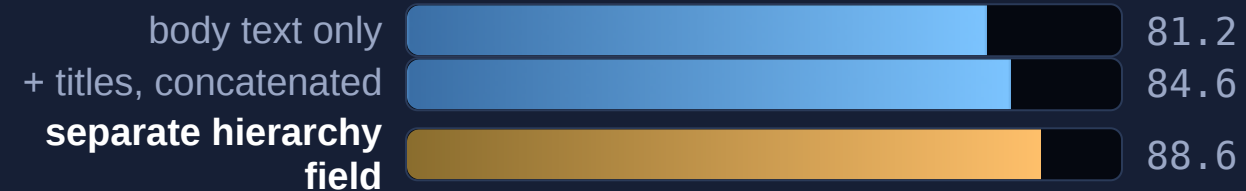


largest gaps on **table/figure-grounded questions** and **multi-hop (3+) questions**

Ablation Study

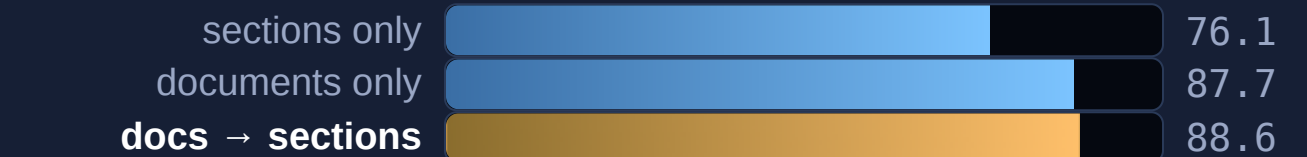
document recall (Avg R@10) when one design choice is removed · bar length = the value on a 0–100 axis

hierarchy in the index · does the Doc card need it?



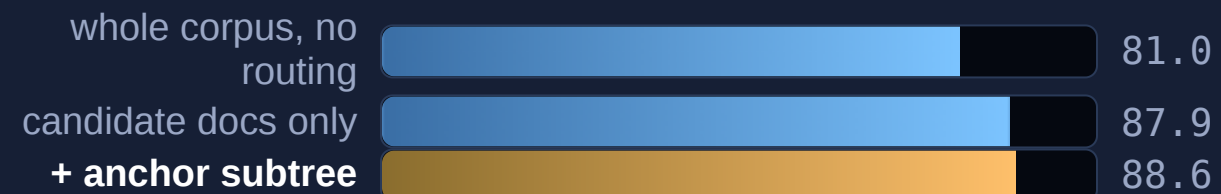
hierarchy as an explicit signal: **+7.4** over no hierarchy

two-stage routing · is coarse-to-fine needed?



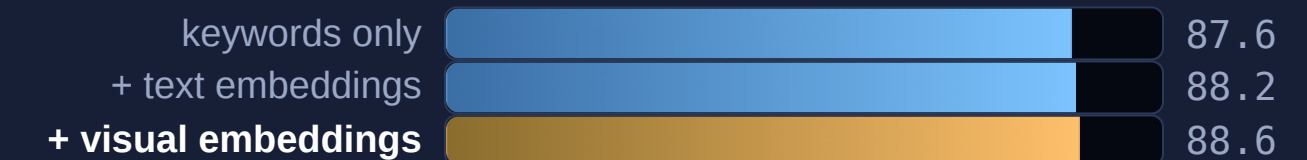
skipping the document stage collapses recall (−12.5)

search scope · does narrowing help at all?



routing itself is worth **+7.6**

score signals · does each one earn its place?



each signal adds a little · the visual one completes it

No latency or cost experiment in the paper · the efficiency case rests on **zero query-time LLM calls** (by design) and the **budget robustness** of the previous slide; offline DHP indexing overhead is acknowledged as a limitation, without numbers

Takeaways

The hierarchy is a free signal

Documents already carry the structure that should steer search. This is why the lineage moved from chunk similarity to hierarchy traversal.

Narrow, then bundle

HiKEY's answer: narrow in two stages, documents → sections (zero LLM calls), then fill the budget core-first. A combination tuned to the industrial setting: thousands of docs, tables and figures, a finite reader budget.

The open question (discussion)

Whichever direction the traversal takes, and however it is trained, it happens **over headings and summaries**. A fact they fail to carry is invisible to every method here. Who breaks this limit?

One line · document-QA search has become a **hierarchy-traversal problem**, and HiKEY is where that traversal converges for multimodal industrial documents: cheap and robust