

Sustainable Local RAG Architectures for Energy Governance: A Green AI Approach via Document Slice Contextualization

Martin Tibor Sándor

University of Debrecen

Róbert Lakatos

`lakatos.robert@inf.unideb.hu`

University of Debrecen

Research Article

Keywords: Retrieval-Augmented Generation, Green AI, Small Language Models, Edge Inference, Energy Governance, Contextual Retrieval

Posted Date: October 21st, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11435653/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Sustainable Local RAG Architectures for Energy Governance: A Green AI Approach via Document Slice Contextualization

Martin Tibor Sándor^{1*} and Róbert Lakatos²

^{1*}Doctoral School of Informatics, Faculty of Informatics, University of
Debrecen, Debrecen, Hungary.

²Department of Data Science and Visualization, Faculty of Informatics,
University of Debrecen, Debrecen, Hungary.

*Corresponding author(s). E-mail(s): sandor.martin@inf.unideb.hu;
Contributing authors: lakatos.robert@inf.unideb.hu;

Abstract

Background: Retrieval-augmented generation pipelines that contextualize every chunk against its whole document follow a “Red AI” pattern: quadratic self-attention and a linearly growing key-value cache push index-time inference beyond the 8 GB of a consumer GPU, while energy-sector auditing and policy work, bound by data-sovereignty rules, must run locally.

Results: We formalize Document Slice contextualization, which conditions each chunk summary on a bounded window of neighboring chunks, and add a two-tier routing agent that selects the radius for each chunk using a regex gate and a single-token forward pass of the same 4B-parameter model. On 25 scientific papers (382 chunks, 250 multi-hop questions) the router chose a mean radius of 1.37, skipped 32 % of the summaries and reached Recall@20 of 0.952 against 0.954 for full-document contextualization, with higher MRR, while cutting tracked generation energy from 106.7 to 13.2 Wh (-87.6%), emissions from 21.8 to 2.8 g CO₂eq and generation time from 54.9 to 8.8 min. In four European Commission policy documents, the fixed-radius variant remained within 0.6 points of the baseline, at 71.8 % lower energy.

Conclusions: Bounded, routed contextualization offers a PEP-AI pathway (Precise, Explainable, Provable) to decentralized compliance auditing: reproducible metrics, inspectable per-chunk decisions, and per-run energy ledgers.

Keywords: Retrieval-Augmented Generation, Green AI, Small Language Models, Edge Inference, Energy Governance, Contextual Retrieval

1 Background

Large language models (LLMs) have changed how organizations search, summarize, and reason over documents [1, 2], but the capability is concentrated where data-center-scale compute is affordable, resulting in a “compute divide” [3–6] between high and low resource institutions. Schwartz et al. called this culture “Red AI” and contrasted it with “Green AI”, which reports efficiency as a first-class result [7]; the energy, carbon and water footprints of large models [8–13] and frameworks for reporting them [14–17] are well documented.

For most organizations, the practical answer to the compute divide is to rent data-center capacity; for some, it is not an option. Energy governance cannot externalize these costs to a cloud provider. The recast Energy Efficiency Directive [18], the Energy Union governance framework [19]. The Commission’s plan for digitalizing the energy system [20] produces long, structured documents that regulators and operators must query precisely and in a defensible manner. In contrast, the General Data Protection Regulation [21], and the Artificial Intelligence Act [22] impose data-sovereignty, transparency, and record-keeping obligations on automated decision support [23]. On-premises retrieval on commodity hardware is therefore a compliance property, and it has to fit the 8 GB of a typical workstation GPU [24].

The standard way of querying such documents with a language model is to retrieve first and generate second. Retrieval-augmented generation (RAG) grounds model answers in retrieved text [25–27]; production systems combine dense retrieval [28] with

BM25 [29, 30] and late-interaction re-ranking [31–34]. Their weakest link is chunking: a passage cut from its document loses the referents that make it retrievable [35–38], layout-aware segmentation only partly repairs this [39, 40], and context-aware representations have therefore been proposed at embedding time [41] and, most influentially, at index time, by prepending a model-written summary of the chunk’s role before embedding [42].

The repair is bought with compute, however, and its price is set by how much context each chunk is given. Industrial index-time contextualization sends the whole document with every chunk: $(n + 1)L$ prompt tokens for a document of n chunks and L tokens, and, since self-attention is quadratic in sequence length [43–45], attention work grows with $\sum_i (L + \ell_i)^2$. The key–value cache grows linearly with the context window [46–48], and once weights plus cache exceed physical VRAM the runtime pages to host memory over PCIe, where throughput collapses [49]; efficient kernels [50], sparse attention [51] and quantization [52] lower the constants, not the dependence on L [53–55]. Long contexts are also used badly: models retrieve information best from the beginning and end of a prompt and degrade for the middle [56], an architectural effect [57–59] that persists in embedding models [60] and multimodal retrieval [61], so a summarizer handed a whole paper to describe one paragraph pays quadratically for context it largely ignores. Giving each chunk less but better-chosen context is the obvious remedy, and several designs move in that direction without settling on how much context is enough. Sentence-window and parent–child retrieval [62, 63], summary trees [64] and graph summarization [65] add neighboring or hierarchical context with inputs that are hard to bound, and adaptive systems decide per query how much to retrieve [66, 67] or which model to call [68, 69]; missing is a per-chunk decision, taken at index time, of how much neighboring context a chunk needs, under a hard bound that keeps every prompt inside a fixed memory envelope.

139 This article extends our conference study of Document Slice contextualization [70]
 140 in that direction with (i) a formal definition of the bounded, boundary-clamped slice
 141 and its index-time cost; (ii) a two-tier routing agent that assigns each chunk a radius
 142 $k \in \{0, 2\}$ with a deterministic table gate and a single-token forward pass of the same
 143 4-billion-parameter model that writes the summaries; (iii) a complexity and memory
 144 model of why the full-document baseline leaves the 8 GB envelope; (iv) an evaluation
 145 on 250 multi-hop questions in which the routed pipeline reaches Recall@20 of 0.952
 146 against 0.954 for the baseline at 12.4% of its tracked generation energy; (v) a cross-
 147 domain check on European Commission policy documents; and (vi) a measurement
 148 protocol with a noise-floor analysis and per-run energy ledgers. Section 2 presents
 149 the method and its cost model, Section 3 presents the results, Section 4 presents the
 150 implications, and Section 5 concludes. The complete architecture, the frozen chunk
 151 and question files, the per-run energy ledger, the routing and retrieval records, and the
 152 scripts that recompute each reported value and draw each figure are openly available
 153 in the project repository [71], archived at Zenodo; the Declarations state which source
 154 documents cannot be redistributed.

167 2 Methods

168 We first describe the two corpora and question sets on which every measurement is
 169 taken (Sect. 2.1). The remaining subsections then follow a document through the
 170 pipeline. The pipeline (Fig. 1) runs on a single workstation with a single 8 GB GPU.

171 A PDF is parsed into layout-aware chunks (Sect. 2.2), a routing agent decides
 172 per chunk how much neighboring context, if any, the summarizer sees (Sects. 2.3–
 173 2.4), and the summary is prepended to the chunk, embedded and indexed for hybrid
 174 retrieval (Sect. 2.5). The only GPU-resident model during context generation is
 175 `gemma3:4b-it-qat` [72], a quantization-aware-trained 4-billion-parameter model served
 176 by Ollama [73], which acts as a router and a summarizer.

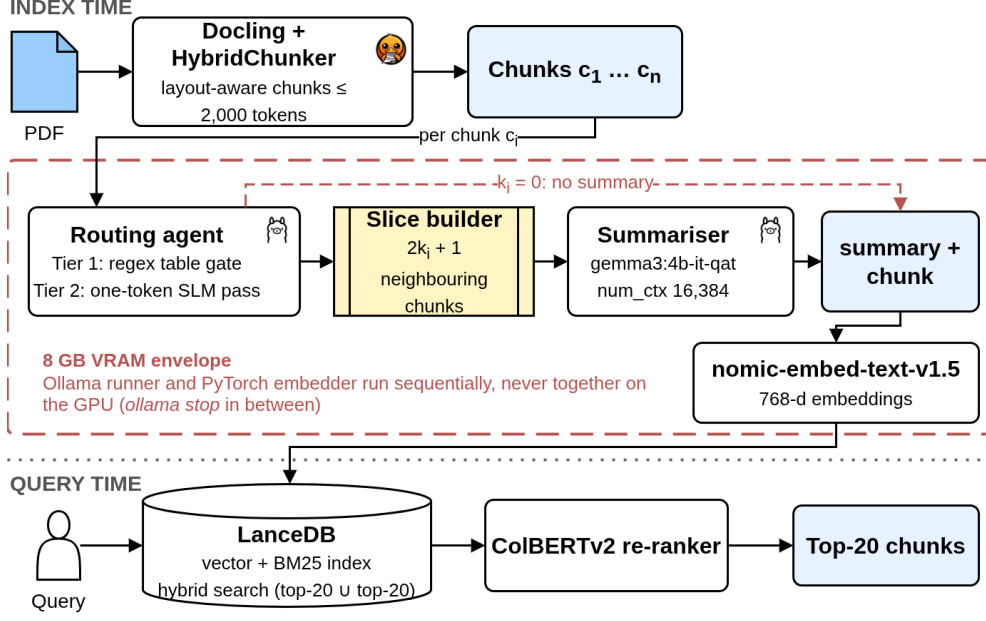


Fig. 1 End-to-end on-premise architecture on one 8 GB consumer GPU: router and summarizer share one resident `gemma3:4b-it-qat` runner (16 384-token window), the embedder runs after the runner is stopped, and chunks routed to $k = 0$ bypass the summarizer.

2.1 Datasets

The primary corpus consists of 25 open-access papers from IEEE and Nature-portfolio journals across engineering, physics, computing, and the life sciences, chosen for their heavy use of tables, formulas, and figure references. Following the LLM-as-a-judge paradigm [74] we built a “silver standard” question set [75]: Gemini 3 Pro was given the chunk JSON of one paper at a time (chunks as produced by the parser of Sect. 2.2) and instructed to write exactly ten high-complexity question-answer pairs grounded only in the provided text, each requiring at least two distinct chunk identifiers (abridged prompt in the conference paper [70]), and a validator rejects questions with fewer than two supporting chunks or with unknown identifiers: 250 questions with 530 gold references over 226 distinct chunks (Table 1). The gold labels come from a large proprietary model and the summaries from a 4-billion-parameter open model, so a shared preference for particular chunks is unlikely. To test the method inside the target genre of regulatory

Table 1 Evaluation corpora: the scientific silver-standard benchmark and the European Commission policy set.

Quantity	Scientific benchmark	EC policy set
Documents	25 (IEEE, Nature portfolio)	4 (COM(2016) 269, COM(2021) 249, SWD(2013) 56, IAP methodology 2023)
Chunks	382	141 (14 / 32 / 35 / 60 per document)
Characters (total) / median chunk length	1 056 773 / 2 090	259 985 / 1 206
Chunk length, tokens (mean / median; tokenizer of Sect. 2.2)	663.6 / 450	446.8 / 246
Questions	250 (10 per document)	80 (10 / 20 / 20 / 30 per document)
Supporting chunks per question	2–4 (mean 2.12; 223 / 24 / 3 questions with 2 / 3 / 4)	2–3 (mean 2.05; 76 / 4)
Gold references	530	164
Distinct gold chunks (share of corpus)	226 (59.5 %)	94 (66.7 %)
Question generator / manual validation	Gemini 3 Pro / —	Gemini 3 Pro / 20 % random subset

text, we use four public European Commission documents on tobacco-control policy (COM(2016) 269, COM(2021) 249, SWD(2013) 56 and the 2023 methodology report of the Independent Advisory Panel on characterising flavours), chunked and questioned with the same protocol (80 questions, 164 gold references; a random 20 % subset was checked manually); they share the genre of energy-governance documents: directive articles, member-state implementation tables, methodology sections and annexes. The exemplars of the routing agent (Sect. 2.4) come from six additional papers that are not in either benchmark, so no text the router has seen as an example can be retrieved as an answer.

2.2 Layout-aware document parsing

Definition 1 (Chunk sequence) Let τ be the tokenizer of the embedding model and $\ell_{\max}$ a token budget. A document is the ordered chunk sequence

$$D = (c_1, \dots, c_n), \quad \ell_j = |\tau(c_j)| \leq \ell_{\max} = 2000, \quad L = \sum_{j=1}^n \ell_j, \quad \text{id}(c_j) = \text{SHA256}(\tilde{c}_j), \quad (1)$$

where $\tilde{c}_j$ is the cleaned chunk text; the content hash gives a chunk the same identifier in every experimental condition.

The sequence of Definition 1 is produced as follows. Segmentation uses Docling [39, 76]: layout analysis, table-structure recognition and OCR per page, then the HybridChunker [77] splits along structural elements and merges adjacent elements of a section while the token count under the `nomic-embed-text-v1.5` tokenizer stays below $\ell_{\max} = 2000$, so that a table, its caption and the paragraph discussing it tend to stay together; the chunks keep the reading order of the document, which is the order the slices of Sect. 2.3 rely on. Chunk text is normalized (Unicode NFKD, ASCII dashes and quotes, URLs removed, whitespace collapsed) before hashing (Fig. 2; Table 2). Two properties matter later: a recognized table is serialized as one line of “<row>, <column> = <value>.” cells, so tables are detectable by a regular expression (the Tier-1 gate of the router, Sect. 2.4), and layout artifacts survive (formula placeholders in 51 of the 382 chunks, glyph codes in 93).

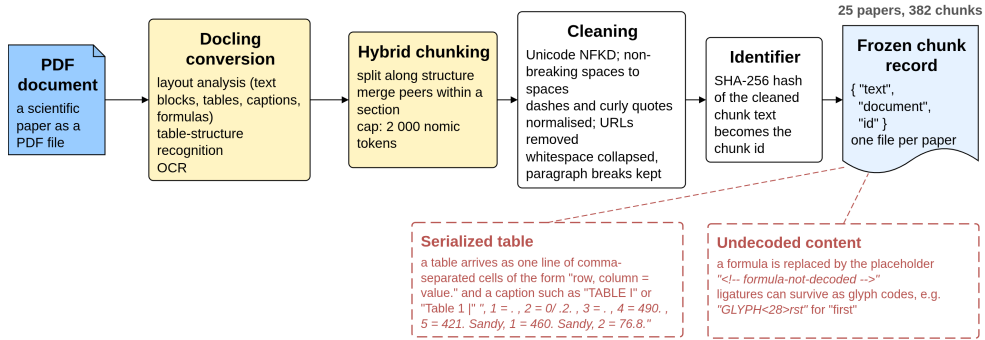


Fig. 2 Layout-aware parsing and chunking: layout analysis, structure-aware segmentation, token-bounded merging of peers within a section, cleaning and hashing. The chunks keep the reading order of the document and form the sequence $D = (c_1, \dots, c_n)$ of Definition 1 that the slices of Sect. 2.3 index; the serialized tables and undecoded content in the call-outs are what the routing agent of Sect. 2.4 detects and has to cope with.

Table 2 Structural parsing metadata and chunk statistics of the scientific corpus (Docling HybridChunker output).

Quantity	Value	Note
Documents / chunks	25 / 382	380 distinct identifiers ¹
Chunks per document (min/median/max)	8 / 14 / 33	token budget 2000, merge peers
Chunk length, characters (min/median/mean/max)	4 / 2 090 / 2 766 / 10 260	total 1 056 773
Chunk length, tokens (min/median/mean/max)	1 / 450 / 666 / 1 999	total 253 488; 4.17 characters per token
Chunks per token band (≤ 250 / 251–500 / 501–1 000 / 1 001–1 500 / 1 501–2 000)	119 / 88 / 78 / 37 / 60	25 chunks at $\geq 1 900$
Document length, tokens (min/median/max)	$\approx 4.3k$ / 10.4k / 18.3k	17 760–76 161 characters
Chunks with formula placeholders / glyph codes	51 / 93	layout artifacts
Chunks carrying a serialized table (Tier-1 gate)	42	cells or caption pattern

¹The 43-character chunk “The authors declare no competing interests.” occurs in three papers and has one identifier; none of the three is a gold chunk.

2.3 Bounded document slices

Definition 2 (Document slice) For a chunk $c_i \in D$ and a context radius $k \in \mathbb{N}_0$ the document slice is the set of chunks within k positions of c_i ,

$$\text{Slice}(c_i, k) = \{c_j \in D \mid \max(1, i - k) \leq j \leq \min(n, i + k)\}. \quad (2)$$

Near a document boundary, where the ideal slice would underflow the start or overflow the end of the document, the implementation compensates the truncated side of Eq. (2) on the other side (Fig. 3): with $u_i = \max(0, k - i + 1)$ chunks missing before c_i and $o_i = \max(0, i + k - n)$ after it,

$$W_i(k) = \{j : \max(1, i - k - o_i) \leq j \leq \min(n, i + k + u_i)\}, \quad |W_i(k)| = \min(2k + 1, n), \quad (3)$$

and the target stays central except at the edges, away from the attention-degraded middle of long prompts [56]. The window is all the context the summarizer is shown.

The summarizer receives a fixed system prompt (Table 3), the window text under the header “FULL DOCUMENT:” (kept verbatim from the baseline, so that only the content differs between conditions) and the chunk under “CHUNK:”; its one- or two-sentence answer s_i is prepended to the chunk before embedding. The prompt length of one call is

$$P_i(k) = s + \sum_{j \in W_i(k)} \ell_j + \ell_i \quad \text{versus} \quad P_i^{\text{full}} = s + L + \ell_i \quad (4)$$

for the full-document design, with $s = 164$ tokens of system prompt, so under $\ell_{\max} = 2000$ every slice prompt is bounded by $s + (2k + 2)\ell_{\max}$ (12 190 tokens for $k = 2$, 16 190 for $k = 3$) regardless of the document: a fixed 16 384-token context window suffices for the whole corpus, whereas the full-document baseline needed a 30 000-token window (Sect. 2.6). Note that $k = 0$ in the static ablation (Sect. 3.1) still calls the summarizer with the chunk as its only context, whereas a radius of 0 assigned by the router means no summarizer call at all.

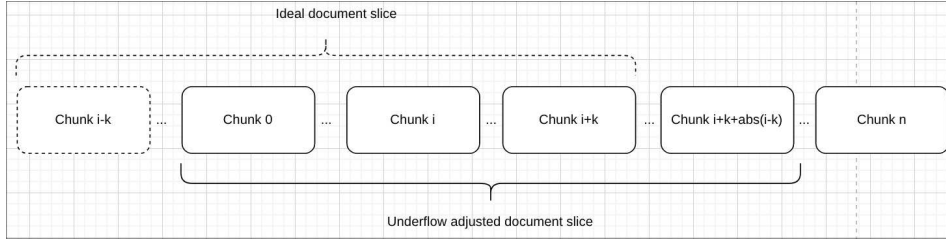


Fig. 3 Boundary handling of the slice window. Underflow case shown; overflow at the end of a document is the mirror image: when the ideal slice of radius k around chunk i would reach before the first chunk, the missing positions are appended after c_{i+k} , so the realized slice still contains $\min(2k + 1, n)$ chunks and the target chunk remains as central as the boundary allows.

2.4 Dynamic radius routing

Our conference ablation [70] showed that $k = 2$ and $k = 3$ are the best fixed radii, but a fixed radius wastes work twice over: chunks that already describe themselves (abstracts, conclusions, reference lists, author and funding statements, layout debris)

415 gain nothing from a summary, and chunks carrying a serialized table are summarized
 416 *worse* when the window is wide, because the model then describes the surrounding
 417 prose or degenerates into one of its own prompt examples. A routing agent $\mathcal{R}$ therefore
 418 chooses the radius per chunk in two cheap tiers, before any summary is written. Tier 1
 419 is a deterministic table gate defined by string matching. Let $\Lambda(c) = (\lambda_1, \dots, \lambda_m)$ be
 420 the lines of a chunk, i.e. c split at its line feeds, and $|w|$ the length of a string w .
 421 For a set of strings X let $\nu_X(w)$ be the number of substrings of w that belong to X ,
 422 counted by a leftmost, non-overlapping scan of w , with ν_u for $X = \{u\}$, and write $\sigma_ =$
 423 for the three-character separator “ = ” (blank, equals sign, blank). Let C be the set
 424 of strings matched by the cell expression of Table 3: the tails “, <col> = <val>.”
 425 of Docling’s serialized cells “<row>, <col> = <val>.”, each cell from the comma
 426 after its row label onward (the row label is unconstrained and may be empty, as in
 427 header cells, so it is not part of the match). Captions are tested by matching predicates:
 428 for an expression r of Table 3, $\tau_r^{\text{pre}}(\lambda) = 1$ if r matches a prefix of the line λ and
 429 $\tau_r^{\text{sub}}(\lambda) = 1$ if r matches some substring of λ , and 0 otherwise. The set of caption tests
 430 is $T = \{\tau_{r_{\text{cap}}}^{\text{pre}}, \tau_{r_{\text{I}}}^{\text{sub}}\}$, where r_{cap} matches the caption heads “TABLE I”, “Table 1:” or
 431 “Table 1 |” together with their leading blanks, which must therefore begin a line, and
 432 r_{I} matches the caption head “Table 1 |” of tables whose caption Docling has glued to
 433 the first cell, which may therefore occur anywhere in a line. With the Iverson bracket
 434 $[P] \in \{0, 1\}$, the gate is

$$449 \quad g_T(c) = \left[\exists \lambda \in \Lambda(c) : (\nu_C(\lambda) \geq 3 \wedge |\lambda| \leq 100 \nu_{\sigma_ =}(\lambda)) \vee \exists \tau \in T : \tau(\lambda) = 1 \right], \quad (5)$$

450 that is, a chunk is a table chunk if one of its lines carries at least three serialized cells
 451 (three cell tails in C) at a density of at most 100 characters per separator, or passes a
 452 caption test, a caption at the start of the line or a pipe caption anywhere in it; table
 453 chunks, $g_T(c) = 1$, receive $k_T = 2$ without a model call. Tier 2 classifies every other
 454
 455
 456
 457
 458
 459
 460

chunk with one forward pass of the resident model emitting a single token, from the router system prompt π_r (Table 3), ten in-context exemplars $\mathcal{X}$ and the chunk clipped to its first 1 500 and last 500 characters; the agent takes the argmax of the top-20 next-token log-probabilities over the two answer letters (fallbacks: the emitted token, then $k_{\text{default}} = 2$), with $\kappa(\text{S}) = 0$ (self-describing, no summarizer call) and $\kappa(\text{N}) = 2$ (needs its neighbors). In the fixed-radius control runs of Sect. 3.1 the agent is bypassed and every chunk receives the same radius $k_{\text{fix}} \in \{0, \dots, 4\}$; the full-document baseline uses no slice at all and is not a case of the policy. The policy, and the two statistics reported for every run, are

$$\mathcal{R}(c_i) = \begin{cases} k_{\text{fix}} & \text{(control runs),} \\ k_T = 2 & \text{if } g_T(c_i) = 1 \text{ (Tier 1),} \\ \kappa(\arg \max_{y \in \{\text{S}, \text{N}\}} \log p_\theta(y \mid \pi_r, \mathcal{X}, \text{clip}(c_i))) & \text{(Tier 2),} \end{cases} \quad (6)$$

$$\bar{k} = \frac{1}{n} \sum_{i=1}^n k_i, \quad N_{\text{calls}} = |\{i : k_i > 0\}|. \quad (7)$$

One forward pass takes about 0.16 s on the RTX 4060M (median 155 ms; the exemplars form a cached prefix of roughly 2 000 tokens), so the corpus is routed in about a minute, and nothing is reloaded between the routing and summarizing phases. With exemplars from six papers outside both benchmarks (109 chunks), the router agrees with a 119-chunk hand-labeled set on 82 of the 99 chunks that reach Tier 2, and the gate fires on all 20 gold table chunks with no false positives. The server’s logits vary slightly with the state of its prefix cache, so chunks whose two letters lie within about 0.1–0.3 nats can flip between runs; the agent logs the letter log-probabilities so that such near-ties can be audited, the “explainable” half of the pathway of Sect. 4.

Table 3 Prompt templates of the summarizer and of the routing agent, and the routing decision criteria.

Component	Specification
Summarizer	<code>gemma3:4b-it-qat</code> served by Ollama; <code>temperature 0, num_ctx 16384</code> (SYSTEM block and <code>PARAMETER</code> lines of <code>Models/chunker_full.doc.Modelfile</code> , passed per call so that the base model stays resident)
Summarizer system prompt	789 characters (164 tokens): return only a short, objective summary of the chunk’s relevance, meaning, or contribution to the document as a whole, no preamble, no phrases such as “This chunk is...”; three one-sentence examples (verbatim in <code>Models/chunker_full.doc.Modelfile</code>)
Summarizer messages	user 1: “FULL DOCUMENT:” + newline + the chunks of $W_i(k)$ joined by single spaces; user 2: “CHUNK:” + newline + c_i ; embedded text s_i + blank line + c_i for $k_i > 0$, c_i alone for $k_i = 0$ (routed)
Router	same resident runner and <code>num_ctx</code> ; <code>temperature 0, num_predict 1, top_p 1, logprobs with top_logprobs 20</code> ; decision: argmax of the letter log-probabilities, $\kappa(S) = 0$, $\kappa(N) = 2$; fallback: parse the emitted letter, else $k_{\text{default}} = 2$
Router system prompt	“Decide whether the passage describes itself”: S – self-describing (abstract; conclusion or summary of the whole paper, even when full of numbers; non-content matter such as reference lists, acknowledgements, funding, author biographies, competing interests, metadata, licence text, layout garbage); N – needs its neighbours (any other passage); “Reply with the letter only” (verbatim: <code>BINARY_SYSTEM_PROMPT</code> in the repository)
Router user message	ten exemplar blocks “Passage: x / Answer: S N” (from six held-out papers: abstract, reference-list excerpt, conclusion, acknowledgement $\rightarrow$ S; formula block, related work, caption fragment, results, device description, experimental setup $\rightarrow$ N), then “Passage: $\text{clip}(c_i)$ / Answer:” with the query clipped to its first 1 500 and last 500 characters
Tier-1 gate, Eq. (5) (Python <code>re</code> syntax)	cell set C , strings matched by $[\text{^,=}\backslash\text{n}\{1,80\}] = [\text{^,=}\backslash\text{n}\{0,80\}]\backslash\text{S}\backslash\text{.}(\text{?: } \$)$ r_{cap} , prefix test $\tau^{\text{pre}}: \text{^}[\backslash\text{t}](\text{?:TABLE Table})[\backslash\text{t}](\text{?:}\backslash\text{d}\{1,2\} \text{[IVXLC]\{1,5\}})\backslash\text{b}$ $(\text{?:}[\backslash\text{t}](\text{?:}\backslash\text{.} \text{[}\backslash\text{t}\text{]}\backslash\text{S} \text{[}\backslash\text{t}\text{]}+\text{[A-Z][A-Za-z]} \text{[}\backslash\text{t}\text{]}*\$)$ r_l , substring test $\tau^{\text{sub}}: \text{^}\backslash\text{bTable}[\backslash\text{t}]+\text{d}\{1,2\}[\backslash\text{t}]*\backslash\text{l}$ ≥ 3 cells of C on one line with ≤ 100 characters per “ = ”, or a caption test $\Rightarrow k_T = 2$

2.5 Context-enriched embedding and hybrid retrieval

The enriched text is embedded with `nomic-embed-text-v1.5` [78], a 768-dimensional open-weight encoder with a long native context and a competitive MTEB score [79] that runs unquantized on the same GPU, through LanceDB’s embedding registry [80] in fp32 with mean pooling, without task prefixes or normalization, identically in every condition. LanceDB stores the enriched text with its vector, a BM25 index, the raw chunk, the summary, the content hash, and the routing decision. Retrieval is hybrid [33, 34]: the 20 nearest chunks by exact L2 search and the 20 best BM25 matches

[29, 30] are re-scored by a ColBERTv2 re-ranker [31, 32] and cut to the top 20. Every system, including the full-document baseline and the no-context control, is indexed and queried with exactly this procedure, so the only experimental variable is the embedded text. The Ollama runner and the PyTorch models cannot share the GPU without spilling, so the phases run in sequence (routing and summarization, `ollama stop`, embedding and indexing, retrieval); Table 4 lists the environment.

Table 4 Experimental hardware and software environment and embedding configuration.

Item	Specification
CPU / system memory	Intel Core i7-14650HX, 24 threads / 31.06 GB
GPU	NVIDIA GeForce RTX 4060 Laptop GPU, 8 GB VRAM; memory bandwidth 256 GB/s; peak use 7 695 MiB over the full pipeline; resident summarizer runner 4.54 GiB (16 384-token window) and 4.82 GiB (30 000), all layers on the device (Sect. 3.6)
OS / Python/server	Linux (zen kernels 6.18–7.2); Python 3.13 (spring and summer 2026 runs) and 3.12 (router study); Ollama 0.17.5 (March 2026 runs), 0.32.1 (July policy-corpus runs), 0.34.0 and 0.34.3 (September router runs), 0.35.1 (memory measurements); one resident runner
Summarizer and router model	<code>gemma3:4b-it-qat</code> (Gemma 3, 4B parameters, quantization-aware trained int4 weights [52, 72]); 34 layers, $n_g = 5$ global and $n_s = 29$ sliding-window (1 024 tokens), $n_{kv} = 4$ key-value heads of width $d_h = 256$, fp16 cache ($b = 2$)
Context window (<code>num_ctx</code>)	16 384 tokens for the slice and routed pipelines (KV cache 494 MiB); 30 000 tokens for the full-document baseline (764 MiB)
Embedding model	<code>nomic-ai/nomic-embed-text-v1.5</code> , 768-d, mean pooling, fp32 on CUDA via the LanceDB HuggingFace registry
Vector store / search	LanceDB 0.26.1; hybrid = exact-L2 vector top-20 $\cup$ BM25 top-20 $\rightarrow$ ColBERTv2 (<code>colbert-ir/colbertv2.0</code>) re-rank $\rightarrow$ 20
Parsing / chunking	docling 2.68.0, docling-core 2.59.0 (HybridChunker, RapidOCR); transformers 4.57.5; torch 2.9.1
Energy tracking	CodeCarbon 3.2.3 (spring/summer 2026 rows) and 3.3.1 (September 2026 rows)

2.6 Computational complexity and memory analysis

Index-time contextualization issues one summarizer call per chunk, so a document costs the sum of Eq. (4). With $\bar{\ell} = L/n$ and the system prompt neglected, the prompt-token

599 volumes of the three designs are

600

601

$$602 \quad C_{\text{full}} = \sum_{i=1}^n (L + \ell_i) = (n+1)L, \quad C_k = \sum_{i=1}^n \left(\sum_{j \in W_i(k)} \ell_j + \ell_i \right) \leq n(2k+2) \ell_{\max},$$

603

$$604 \quad C_{\text{dyn}} = \sum_{i: k_i > 0} \left(\sum_{j \in W_i(k_i)} \ell_j + \ell_i \right) + \sum_{i \notin \mathcal{T}} \rho_i, \quad (8)$$

605

606

607 where $\mathcal{T} = \{i : g_T(c_i) = 1\}$ is the set of table-gated chunks (which bypass the Tier-2

608

609 router and thus incur zero routing tokens), and ρ_i the router prompt (about 2000

610

611 tokens, mostly a cached prefix): the full-document volume is quadratic in n , the bounded

612

613 volume linear with a constant $2k+2$ that the router lowers further by skipping chunks.

614

615 Attention widens the gap, since a forward pass over P tokens costs approximately [45]

616

$$617 \quad F(P) \approx 2N_{\text{par}} P + 2n_L d_{\text{attn}} P^2 \quad (9)$$

618

619

620 (N_{par} non-embedding parameters, n_L layers, attention width d_{attn}), so a document's

621

622 attention work is $\propto \sum_i P_i^2$.

623

624

625 **Proposition 1** (Asymptotic index-time cost) *Let P_i^{full} and $P_i(k)$ be the per-call prompt*

626

627 *lengths of Eq. (4) with the system prompt neglected ($s = 0$); they are the arguments P of*

628

629 *Eq. (9) for the two designs. Assume $\ell_j \leq \ell_{\max}$ for every chunk and $n \geq 2k+1$. Then*

630

$$631 \quad n^3 \bar{\ell}^2 \leq \sum_{i=1}^n (P_i^{\text{full}})^2 \leq 4n^3 \bar{\ell}^2 \quad \text{and} \quad n \bar{\ell}^2 \leq \sum_{i=1}^n P_i(k)^2 \leq n(2k+2)^2 \ell_{\max}^2.$$

632

633

634 *Hence the attention work of the full-document design is $\Theta(n^3 \bar{\ell}^2)$, that of the slice design is*

635

636 *$O(n \ell_{\max}^2)$ for fixed k , and their ratio is at least $(\bar{\ell}/\ell_{\max})^2 n^2/(2k+2)^2$; the prompt-token*

637

638 *volumes of Eq. (8) differ by at least $(\bar{\ell}/\ell_{\max})(n+1)/(2k+2)$. For a token-bounded chunker*

639

640 *that merges peers up to its budget, $\bar{\ell}/\ell_{\max}$ is a constant (0.33 on this corpus), so the ratios*

641

642 *are $\Theta(n^2)$ and $\Theta(n)$.*

643

644

Proof Full document: $P_i^{\text{full}} = L + \ell_i$ with $0 \leq \ell_i \leq L$, so $L^2 \leq (L + \ell_i)^2 \leq 4L^2$ for every i ; summing over the n chunks and substituting $L = n\bar{\ell}$ gives the first chain. Slice: by Eq. (3) the window $W_i(k)$ holds exactly $2k + 1$ chunks when $n \geq 2k + 1$, each of at most $\ell_{\max}$ tokens, so $P_i(k) = \sum_{j \in W_i(k)} \ell_j + \ell_i \leq (2k + 2)\ell_{\max}$, which gives the upper bound. For the lower bound, $i \in W_i(k)$ gives $P_i(k) \geq \ell_i$, so $\sum_i P_i(k)^2 \geq \sum_i \ell_i^2 \geq (\sum_i \ell_i)^2 / n = n\bar{\ell}^2$, the last step being the Cauchy–Schwarz inequality $\langle \mathbf{L}, \mathbf{1} \rangle^2 \leq \|\mathbf{L}\|^2 \|\mathbf{1}\|^2$ for the vectors $\mathbf{L} = (\ell_1, \dots, \ell_n)$ and $\mathbf{1} = (1, \dots, 1)$, i.e. $(\sum_i \ell_i)^2 \leq n \sum_i \ell_i^2$. Each ratio follows by dividing the lower bound of one design by the upper bound of the other; for the token volumes use $C_{\text{full}} = (n + 1)n\bar{\ell}$ and $C_k \leq n(2k + 2)\ell_{\max}$ from Eq. (8). $\square$

On the corpus the bounds are loose but the ordering holds (Fig. 4): the full-document design sends 18.73 M characters (≈ 4.5 M tokens) to the summarizer, fixed $k = 1, \dots, 4$ send 4.25, 6.36, 8.43 and 10.32 M, and the routed pipeline 4.39 M (23.4% of the baseline) in 261 instead of 382 calls; $\sum_i P_i^2$ is 4.6 times smaller for $k = 3$ and 10.8 times smaller for the routed pipeline than for the baseline, and the measured generation-time ratio of 6.3 (Sect. 3.2) lies between this quadratic proxy and the linear one (4.3).

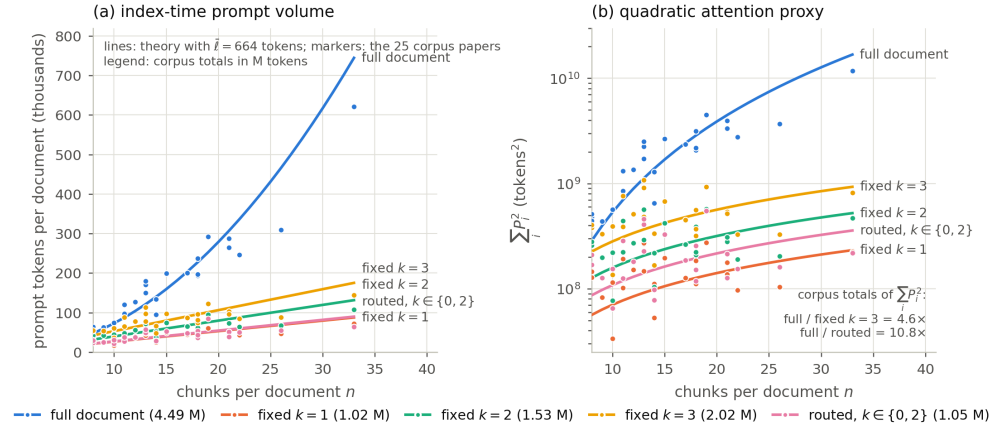


Fig. 4 Theoretical index-time cost: full-document contextualization grows quadratically in the number of chunks in prompt tokens and cubically in attention work, bounded slices linearly with a constant set by the radius, which the router lowers further by skipping self-describing chunks.

Memory decides whether the arithmetic can be done on the device at all: the GPU must hold the quantized weights M_w , the activations M_{act} , and the KV cache, which

691 for P tokens occupies

$$692 \quad M_{KV}(P) = 2 n_L n_{kv} d_h b P \quad (10)$$

694 bytes (n_{kv} key-value heads of width d_h , b bytes per element, and the factor of 2
695 accounts for storing both keys and values) [46, 48], so the run stays on the device only if

$$696 \quad M_w + M_{KV}(P_{\text{ctx}}) + M_{\text{act}} \leq V_{\text{GPU}} \iff P_{\text{ctx}} \leq P^* = \frac{V_{\text{GPU}} - M_w - M_{\text{act}}}{2 n_L n_{kv} d_h b}, \quad (11)$$

701
702 where P_{ctx} is the window reserved at load time, not the current prompt length. The
703 bounded slice fixes $P_{\text{ctx}} = 16\,384$ for every document ($k \leq 3$); the longest paper alone is
704 about 18 300 tokens, so the full-document design needs $P_{\text{ctx}} \geq s + L_{\text{max}} + \ell_{\text{max}} \approx 20\,400$
705 tokens and was run with a 30 000-token window. What the wider window costs depends
706 on the cache architecture. For a dense cache, Eq. (10) with the constants of an 8B
707 model of 32 layers and 8 key-value heads of width 128 gives 3.7 GiB at 30 000 and
708 2.0 GiB at 16 384 tokens in fp16, so next to about 4.6 GiB of int4 weights the wider
709 window no longer fits an 8 GiB device while the bounded one does. The model used
710 here is a more favorable case. Gemma 3 interleaves $n_s = 29$ local layers, whose cache
711 is capped at the sliding window of 1 024 tokens, with $n_g = 5$ global layers, so only the
712 global share grows with the window:
713
714
715
716
717
718
719
720

$$721 \quad M_{KV}(P_{\text{ctx}}) = 2 n_{kv} d_h b (n_g P_{\text{ctx}} + n_s \tilde{w}), \quad (12)$$

722
723 with $\tilde{w} = 1\,536$ cells (the 1,024-token sliding window plus a 512-token query/window
724 plus batch). The server allocates 764 MiB at 30 000 and 494 MiB at 16 384 tokens,
725 and the resident runner measures 4.82 against 4.54 GiB on an otherwise empty device,
726 every layer on the GPU in both cases, under the server release of the March runs as
727 well as the current one. For this model, the reserved window therefore does not cross
728
729
730
731
732
733
734
735
736

P^* on an 8 GiB device by itself; what crosses it is the memory that other pipeline components hold at the same time, which Sect. 3.6 measures.

2.7 Evaluation metrics and statistical protocol

For a question $q \in Q$ let G_q be its gold chunks, R_q^K the top- K retrieved list and $\text{rank}_q(c)$ the 1-based position of c in it; for $K \in \{5, 10, 15, 20\}$ we report

$$\text{Recall@}K = \frac{1}{|Q|} \sum_{q \in Q} \frac{|R_q^K \cap G_q|}{|G_q|}, \quad Q_K^+ = \{q \in Q : R_q^K \cap G_q \neq \emptyset\}, \quad (13)$$

$$\text{MRR@}K = \frac{1}{|Q_K^+|} \sum_{q \in Q_K^+} \frac{1}{\min_{c \in R_q^K \cap G_q} \text{rank}_q(c)}, \quad (14)$$

$$\text{MAR@}K = \frac{1}{|Q_K^+|} \sum_{q \in Q_K^+} \frac{1}{|R_q^K \cap G_q|} \sum_{c \in R_q^K \cap G_q} \text{rank}_q(c). \quad (15)$$

Recall is primary because every question needs at least two chunks; MRR [81] reflects the first relevant chunk; MAR, our exploratory measure of attentional accessibility, is the mean position of *all* retrieved gold chunks and tells whether the evidence sits where the positional bias of Sect. 1 is least harmful. Ranking metrics are averaged over Q_K^+ (questions without a hit are excluded rather than scored zero; 233, 246, 247, and 248 of 250 for the routed pipeline), and gold sets are matched by content hash. Recall@20 differences are tested per question with a 99% BCa bootstrap interval of the paired difference [82, 83] (SciPy [84], 9999 resamples, fixed seed) and an exact McNemar test [85, 86] on the event “all gold chunks in the top 20”, reporting the discordant counts b (only the compared system succeeds) and c (only the baseline succeeds). Because the summarizer is stochastic even at temperature 0, we also measure the noise floor of a single run by regenerating one configuration (Sect. 3.3): about 0.7 points of Recall@20 and 2 points of MRR@20, below which gaps are treated as ties.

2.8 Continuous energy and carbon accounting

Every generation run is wrapped in a CodeCarbon tracker [15, 87] that samples GPU, CPU, and memory power once per second and integrates it over the run; emissions follow through the grid factor:

$$E_{\text{total}} = \int_0^T (P_{\text{GPU}}(t) + P_{\text{CPU}}(t) + P_{\text{DRAM}}(t)) dt \approx \sum_{t=1}^{\lceil T/\Delta t \rceil} (P_{\text{GPU}}^{(t)} + P_{\text{CPU}}^{(t)} + P_{\text{DRAM}}^{(t)}) \Delta t, \quad (16)$$

$$\text{Emissions} = E_{\text{total}} \times \text{CI}_{\text{grid}} \times \text{PUE}, \quad (17)$$

with $\Delta t = 1$ s, where CI_{grid} is the operational carbon intensity of the regional grid in g CO₂eq per kWh and the power-usage effectiveness is 1 for a workstation. GPU power is read from NVML; CPU power came from RAPL counters for the spring-2026 runs and from a load-based estimate for the later runs; memory power is a constant estimate (Table 5). Energy is therefore the primary efficiency metric, and GPU energy, measured consistently throughout, is reported alongside the total as the component comparable without qualification. Durations come in two scopes: the *tracked generation scope*, the interval wrapped by the tracker (routing and summarization over pre-chunked documents, without PDF parsing, embedding or indexing), to which all energy figures refer; and the *end-to-end preprocessing scope*, including PDF conversion, to which the conference paper’s headline times (71 min 34 s baseline, 29 min 09 s fixed $k = 3$) and the routed pipeline’s 10 min 30 s measured for this article (Sect. 3.2) belong.

3 Results

All metrics are those of Sect. 2.7 on the corpora of Sect. 2.1; times and energies refer to the tracked generation scope unless stated otherwise. Sect. 3.1 first reports the fixed-radius sweep that motivated the router, Sect. 3.2 the central comparison of the full-document baseline, the static slice, and the routed pipeline in accuracy, time,

Table 5 Power-measurement coefficients, per-run power draw, and grid emission factors.

Quantity	Value
Sampling interval / tracking mode / PUE	1 s / whole machine / 1.0
GPU power	measured via NVML (all runs)
CPU power	RAPL counters (runs of 5 March 2026); load-based estimate with the tracker’s TDP fallback (“4 W per thread $\times$ 24 threads = 96 W”) from 19 July 2026 on constant 20.0 W estimate (31 GB installed)
Memory power	204.19 g CO ₂ eq/kWh (CodeCarbon 3.2.3 data set); effective 209.9–214.0 g CO ₂ eq/kWh in the 3.3.1 rows
Grid carbon intensity (Hungary)	51.5 W / 46.5 %; 55.0 W / 60.6 %; 62.5 W / 90.1 %
Mean GPU power / utilization: baseline; static $k = 3$; routed	46.1 W; 58.0 W; 10.4 W
Mean CPU power: baseline; $k = 3$ (RAPL); routed (estimate)	15.9 / 19.4 / 17.3 GB
Mean host memory in use: baseline / $k = 3$ / routed	not tracked (WUE 0)
Water footprint	

energy, and emissions, and Sect. 3.3 what the router decided, how its policy was arrived at and how much a single run can resolve. Sect. 3.4 then examines where the retrieved evidence lands in the ranking, Sect. 3.5 repeats the comparison on the European Commission policy corpus, and Sect. 3.6 closes with the memory measurements that test the model of Sect. 2.6.

3.1 Static radius sweep

Table 6 and Fig. 5 report the fixed-radius ablation that motivated the router. Generation time grows linearly with the radius, about five minutes per unit of k over the 382 chunks, while recall saturates: $k = 2$ and $k = 3$ are the best fixed radii (Recall@20 of 0.94667 and 0.94533) and $k = 4$ falls back to the level of $k = 1$ (0.94333) although it sends 62 % more prompt text than $k = 2$. The full-document baseline remains the most accurate (0.95400) at 54 min 54 s and 106.7 Wh, and even the no-context control reaches 0.93933, so contextualization is worth about 1.5 points on this benchmark, most of it from the immediate neighborhood of a chunk. All recall differences among $k \in \{1, \dots, 4\}$ lie inside the single-run noise band (Sect. 3.3): the sweep identifies a

plateau, not an optimum, which is where a per-chunk decision can save cost without losing accuracy.

Table 6 Ablation over static window radii versus full-document context and the routed pipeline (250 questions).

Method	Time (min:s)	Prompt (M chars)	Recall@20	MRR@20	MAR@20
No context (hybrid only)	—	0	0.93933	0.77484	4.03629
Slice $k = 0$ (chunk-only summary)	7:28 ¹	2.11	0.93733	0.78612	3.80827
Slice $k = 1$	12:08 ¹	4.25	0.94333	0.79900	3.72605
Slice $k = 2$	17:28 ¹	6.36	0.94667	0.76799	3.80948
Slice $k = 3$	22:41 ¹	8.43	0.94533	0.78774	4.01999
Slice $k = 4$	28:11 ¹	10.32	0.94333	0.78335 ²	3.95884
Full document (baseline)	54:54	18.73	0.95400	0.78635	3.93548
Routed (this work)	8:45	4.39	0.95200	0.79798	3.91062

¹Wall-clock of the generation loop as recorded in the notebook; these runs were not energy-tracked. Energies of the baseline, the static $k = 3$ slice and the routed pipeline are in Table 7.

²The conference paper printed 0.8933; 0.78335 is the value in the evaluation notebook.

	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$
generation time (min)	7.5	12.1	17.5	22.7	28.2
prompt volume (M chars)	2.11	4.25	6.36	8.43	10.32

Fig. 5 Cost surface of the static radius: generation time grows linearly in k and prompt volume with $2k + 2$ (notebook wall-clock of the generation loop over 382 chunks; $k = 0$ still calls the summarizer with the chunk alone), while the reserved context window, and with it the resident device memory of the runner, stays at 4.54 GiB for every k (Sect. 3.6).

3.2 Primary benchmark: full-document baseline, static slice and routed pipeline

Table 7 is the central result. The routed pipeline reaches Recall@20 of 0.952 against 0.954 for the full-document baseline and 0.94533 for the static $k = 3$ slice, both differences inside the single-run band, with the best MRR@20, MAR@20 and Recall@5

of the three; the recall trajectories (Fig. 8) show it ahead at $K = 5$ and 10, 0.7 points behind at $K = 15$ and 0.2 at $K = 20$. The cost side (Figs. 6 and 7) is not inside any noise band: 261 summarizer calls instead of 382, 23.4 % of the baseline’s prompt text, 8 min 45 s instead of 54 min 54 s (6.3 times faster), 13.22 Wh instead of 106.68 Wh (−87.6 %; GPU energy −81.2 %) and 2.83 instead of 21.78 g CO₂eq (−87.0 %). The tracked scope excludes PDF parsing; the saving survives when it is included. End to end from the PDFs, the routed pipeline needs 10 min 30 s on the same machine (parsing 1 min 41 s, routing 1 min 08 s, summarizing 7 min 39 s; 15.24 Wh for the whole run), against the conference paper’s 71 min 34 s and 29 min 09 s; its routing and summarizing part, 8 min 48 s, reproduces the tracked 8 min 45 s of the September run. The static $k = 3$ slice of our conference paper halves the baseline’s energy (−50.8 %) and time (−56.6 % tracked, −59.3 % end to end) for a 0.9-point loss of Recall@20; the router removes a further 63 % of its generation time and 75 % of its energy (60 % of GPU energy) while adding 0.7 points of recall. A second, independent generation of the same configuration also reached Recall@20 of 0.952 (MRR@20 0.79791, MAR@20 3.92171) in 543.7 s and 13.51 Wh. Per-question tests on that replicate give a paired difference of mean Recall@20 (baseline minus routed) of +0.0020 with a 99 % BCa interval of [−0.0153, +0.0200] and McNemar discordant counts of 5 and 6 ($p = 1.00$; 222 questions solved by both, 17 by neither); for the static $k = 3$ slice the difference is +0.0087, the interval [−0.0067, +0.0260] and the counts 3 and 6 ($p = 0.51$).

3.3 Routing statistics, ablation of the routing policy and the noise floor

Of the 382 chunks, 42 were assigned $k = 2$ by the table gate and 340 reached the SLM, which labeled 121 as self-describing ($k = 0$, no summary) and 219 as needing their neighbors ($k = 2$): $\bar{k} = 1.37$, 31.7 % of the chunks never reached the summarizer, and the 261 summaries average 201 characters; the replicate run decided 120 and 262. The

Table 7 Primary benchmark on 250 multi-hop questions: full-document contextualization versus static $k = 3$ slice and routed pipeline.

Metric	Full doc.	Static $k = 3$	Routed	Δ vs full	Δ vs $k = 3$
Recall@5	0.73733	0.74100	0.75167	+1.4 pt	+1.1 pt
Recall@10	0.86500	0.85900	0.86700	+0.2 pt	+0.8 pt
Recall@15	0.93333	0.91067	0.92600	-0.7 pt	+1.5 pt
Recall@20	0.95400	0.94533	0.95200	-0.2 pt	+0.7 pt
MRR@20	0.78635	0.78774	0.79798	+1.5 %	+1.3 %
MAR@20	3.93548	4.01999	3.91062	-0.6 %	-2.7 %
Summarizer calls	382	382	261	-31.7 %	-31.7 %
Mean radius $\bar{k}$	whole doc.	3	1.37		
Prompt volume (M chars)	18.73	8.43	4.39	-76.6 %	-47.9 %
Generation time (min:s)	54:54	23:48 ¹	8:45	-84.1 % (6.3 $\times$)	-63.2 %
End to end from the PDFs (min:s)	71:34 ²	29:09 ²	10:30 ²	-85.3 % (6.8 $\times$)	-64.0 %
Energy (Wh)	106.68	52.52 ¹	13.22	-87.6 %	-74.8 %
of which GPU (Wh)	46.78	21.73 ¹	8.80	-81.2 %	-59.5 %
Emissions (g CO ₂ eq)	21.78	10.72 ¹	2.83	-87.0 %	-73.6 %

¹Tracked generation of 5 March 2026; the retrieval columns come from the ablation generation of the same configuration. CPU energy: RAPL for the March rows, load estimate for the routed run; GPU energy is comparable across rows. $|Q_{20}^+|$: 248, 246, 248.

²Notebook wall-clock including Docling conversion: the conference-paper runs of March 2026 for the baseline and the static slice (comments in the notebooks); for the routed pipeline, one run of 5 October 2026 from the PDFs (RnD/end.to.end.dynamic.doc.slice.generation.ipynb: parsing 1:41, routing 1:08, summarizing 7:39, 15.24 Wh; 377 chunks with the current Docling release against 382 in March).

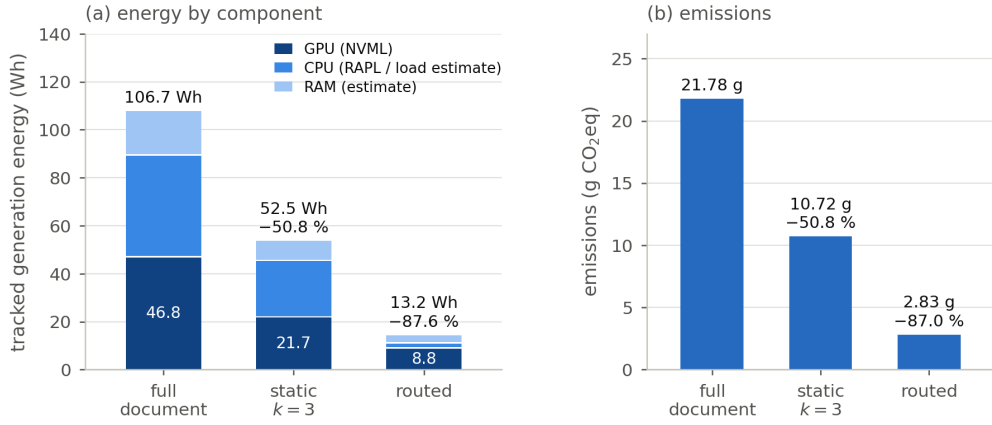


Fig. 6 Tracked generation energy and emissions on the 25-paper corpus: the static $k = 3$ slice halves the baseline’s energy and emissions, the routed pipeline removes 87.6% and 87.0% of them. The GPU component (NVML; -53.6% and -81.2%) is the one measured identically in every run; the CPU component came from RAPL counters in March and from a load-based estimate in September (Table 5).

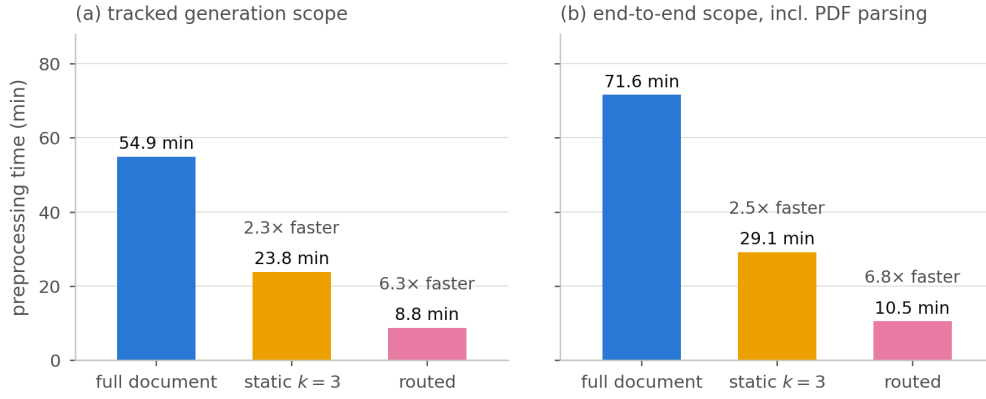


Fig. 7 Preprocessing runtime in the two scopes of Sect. 2.8: tracked generation is 2.3 times (static slice) and 6.3 times (routed) faster than the baseline; end to end, from the PDFs, the static slice is 2.5 times and the routed pipeline 6.8 times faster than the baseline’s conference-paper time (10 min 30 s, of which parsing 1 min 41 s, routing 1 min 08 s and summarizing 7 min 39 s).

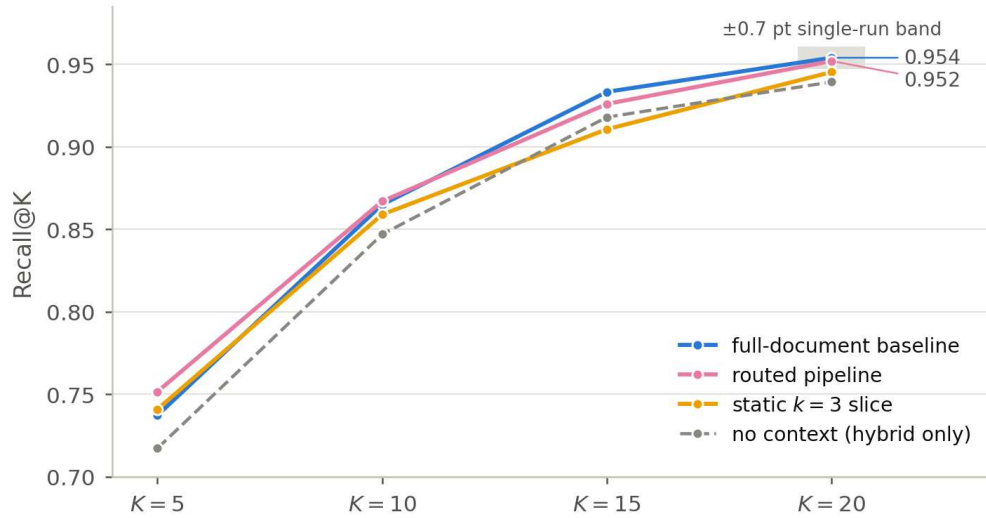


Fig. 8 Recall trajectories on the 250 scientific questions: the routed pipeline leads the baseline at $K = 5$ and 10 and trails it by 0.7 points at $K = 15$ and 0.2 at $K = 20$, inside the single-run band; the static $k = 3$ slice trails beyond $K = 5$; the dashed line is hybrid retrieval without any generated context.

skipped chunks are what the prompt asks for: author blocks, abstracts, conclusions, reference lists, and acknowledgments.

This policy is the endpoint of an ablation of more than 40 simulated policies recorded in the repository. A first four-class genre router with tables at $k = 3$ scored Recall@20 of 0.9380, below fixed $k = 2$ (0.94667) and even below the no-context control, because

wide slices produced degenerate table summaries (for one large table, a verbatim copy of a prompt example); gating tables to $k = 1$ recovered part of the loss (0.9420 in two identical live runs). An offline simulator that re-assembles a policy from stored summaries and replays the exact retrieval procedure then ranked alternatives without summarizer noise: $k = 3$ is never needed once tables are gated below it, skipping self-describing chunks costs nothing and gives the best MRR, MAR and Recall@5, length- or position-only heuristics (0.9373–0.9447) cannot replace the classifier, and a per-question oracle over $k \in \{1, 2, 3\}$ reaches 0.964, so the headroom above fixed $k = 2$ is 1.7 points and only 17 of the 226 gold chunks are sensitive to k at all. The two-class prompt with tables at $k = 2$ was the best offline (0.9507) and, with held-out exemplars, achieved 0.9487, then 0.952 live.

The noise floor was measured with a constant $k = 2$: regeneration reproduced Recall@20 exactly as 0.94667. Yet, only 26 of 380 summaries were identical, 238 of 250 questions received different top-20 lists, and MRR@20 increased from 0.768 to 0.792. In contrast, the same request sequence, issued twice from a freshly loaded model, yielded 380 identical summaries. The summarizer thus depends deterministically on the request history rather than on a seed, and a single run resolves about 0.7 points of Recall@20 and 2 points of MRR@20; hence, policies are ranked offline and confirmed live, and the two independent generations of the final configuration (0.952 and 0.952) show that its result is not a favorable draw. Moving the exemplars to held-out papers changed the router’s letter for 15 % of the chunks but did not affect the question’s outcome in the offline replay.

3.4 Retrieval quality and ranking dynamics

The ranking metrics explain why skipping a third of the summaries does not hurt: a summary prepended to a chunk that already states what it is about only dilutes its embedding, and embedding such chunks raw lifts Recall@5 above the baseline (0.75167

against 0.73733) and MRR@20 to 0.79798, the highest in this study. With MAR@20 of 3.91, the gold chunks sit, on average, at rank four of twenty in the primacy region of the U-shaped accessibility curve (Fig. 9); the baseline (3.94) and the static slice (4.02) are marginally deeper. Questions with no relevant chunk retrieved are few and similar across systems (17, 4, 3, and 2 of 250 at $K = 5, 10, 15, 20$ for the routed pipeline against 17, 6, 2, and 2).

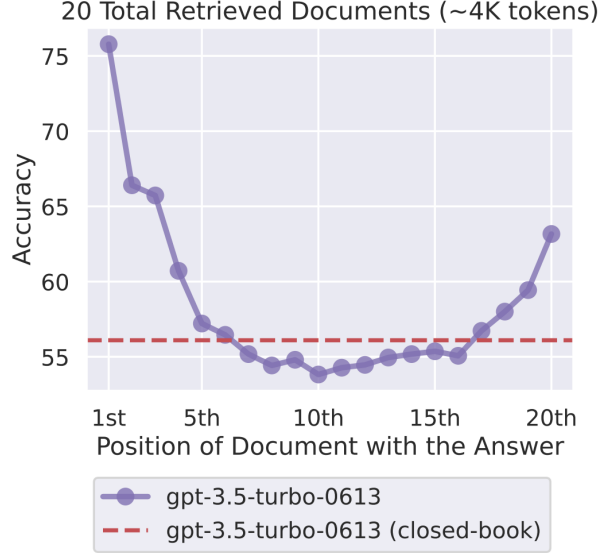


Fig. 9 The “lost in the middle” effect, reproduced from Liu et al. [56]: answer accuracy of a language model as a function of the position of the relevant passage among 20 retrieved documents is highest at the first positions, lowest in the middle, and only partly recovers at the end. The curve is shown as a general, model-agnostic tendency in long-context reading and is not a measure of our system; it motivates interpreting the mean average ranks in Table 7 (about four of twenty) as evidence placed in the primacy region. Reproduced from Figure 1 of Liu et al. [56], © the authors, licensed under CC BY 4.0.

3.5 Cross-domain robustness on European Commission policy documents

Table 8 repeats the static comparison on the policy corpus: the fixed $k = 3$ slice reaches Recall@20 of 0.9625 against 0.96875 for the full-document baseline (−0.6 points), MRR@20 0.813 against 0.822 and MAR@20 3.66 against 3.52, in 252 s instead of

858 s (−70.6 %) and 5.08 instead of 18.00 Wh (−71.8 %; GPU energy −72.1 %). The recall gap equals that on the scientific corpus. The efficiency gain is larger because the documents are longer (up to 60 chunks, 103 000 characters), the regime in which Proposition 1 predicts the widest gap; regulatory text is thus served by the bounded slice at least as well as scientific prose. The router has not yet been run here: its prompt names abstracts and conclusions and needs policy-specific exemplars (executive summaries, legal bases, annex tables).

Table 8 Out-of-distribution evaluation on the European Commission policy corpus (full document versus static slice). 80 multi-hop questions, 4 documents, 141 chunks; the routing agent has not yet been run on this corpus.

Metric	Full document	Static $k = 3$	$\Delta k = 3$ vs full
Recall@5	0.78958	0.75833	−3.1 pt
Recall@10	0.90625	0.90000	−0.6 pt
Recall@15	0.95000	0.94375	−0.6 pt
Recall@20	0.96875	0.96250	−0.6 pt
MRR@20	0.82218	0.81296	−1.1 %
MAR@20	3.51667	3.65625	+4.0 %
Generation time, tracked scope	14 min 18 s	4 min 12 s	−70.6 %
Energy, tracked scope (Wh)	18.00	5.08	−71.8 %
of which GPU (Wh)	14.53	4.05	−72.1 %
Emissions (g CO ₂ eq)	3.68	1.04	−71.8 %

3.6 Hardware profiling

The memory model of Sect. 2.6 was checked after the fact on the benchmark machine, first with the summarizer runner alone and then with the other GPU users of the pipeline resident. Alone, the `gemma3:4b-it-qat` runner occupies 4.54 GiB of the 8 GiB device at the 16 384-token window and 4.82 GiB at 30 000, independent of the radius and of the document, with all 35 layers on the GPU under both server releases, and evaluates the full-document prompt of the longest paper (19 900 tokens) in 5.5 s. The device is shared, however: the Docling converter keeps 1.8 GiB of device memory after one PDF, and the fp32 embedder, as the LanceDB registry loads it, a further

4.1 to 4.4 GiB after embedding the corpus, so a notebook kernel that has parsed and embedded holds 5.9 to 6.1 GiB. Under that load the runner can no longer be placed on the device (Fig. 10): the March release keeps 12 of 35 layers on the GPU, fills the card to 7.4 GiB and evaluates the same prompt in 10.3s; the current release keeps none and needs 13.4s; the server’s host memory grows from 2.6 to 6.2 GiB. The two releases degrade differently because they place layers with different code, Ollama’s own engine against llama.cpp’s automatic fit, but both keep every layer on the device with up to 1.8 GiB held by others, which is why the architecture of Fig. 1 stops the runner before the embedder loads. Peak device use over the whole pipeline was 7 695 MiB. Prefix caching did not help the baseline either: under the March release every chunk re-evaluated the whole document even with a constant header (18 200 to 19 900 tokens per call), and the current release reuses the document prefix only when the preceding chunk is shorter than the 1 024-token sliding window, so the baseline’s time and energy are those of a full re-evaluation per chunk.

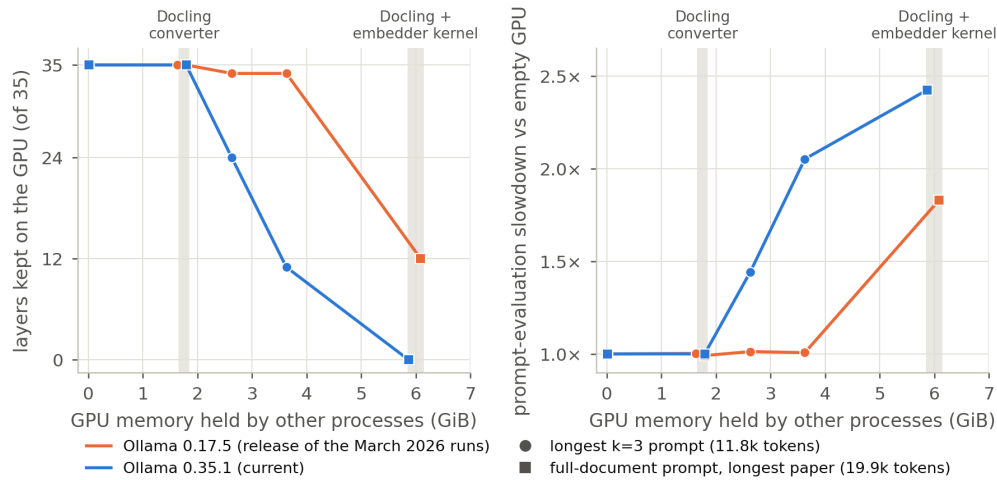


Fig. 10 Layer placement and prompt-evaluation slowdown of the runner under shared GPU memory. Measured for the `gemma3:4b-it-qat` runner at the 30 000-token window as a function of the device memory other processes already hold, for the server release of the March runs (0.17.5) and the current one (0.35.1); circles: longest $k = 3$ prompt, squares: full-document prompt of the longest paper. Shaded bands: measured footprint of the Docling converter (1.8 GiB) and of a kernel that has parsed and embedded the corpus (5.9–6.1 GiB).

1243 4 Discussion

1244

1245 4.1 Practical applicability

1246

1247 For an authority indexing thousands of pages on-premises, the relevant trade-off is
1248 among accuracy, time, and energy. On it, the full-document baseline buys its last 0.2
1249 points of Recall@20 with 8.1 times the generation energy and 6.3 times the wall-clock
1250 of the routed pipeline, and the static $k = 3$ slice sits in between, at half the baseline’s
1251 energy for a 0.9-point loss. Because every recall gap lies inside the ± 0.7 -point band of a
1252 single run while the energy gaps exceed run-to-run variation by an order of magnitude,
1253 the Pareto choice is unambiguous: bounded, routed contextualization dominates, and
1254 the savings scale with corpus size, recur at every re-index and grow with document
1255 length through the baseline’s paging penalty. The routing agent is the part of this
1256 result that was tuned, and its history shows where its gain came from. The router’s
1257 development path (0.938, 0.942, 0.9487, 0.952) shows that its gain came from the class-
1258 to-radius mapping rather than from prompt wording; it costs one token per chunk,
1259 about a minute per corpus, and no second model. Its limits are equally clear: the oracle
1260 headroom above fixed $k = 2$ is 1.7 points, and the summarizer’s run-to-run variation is
1261 of the same order, so further gains must come from the summarizer or from multi-run
1262 evaluation.

1273

1274

1275 4.2 Limitations and future directions

1276

1277 The slice is intra-document: a chunk whose meaning depends on another document,
1278 an annex cited in a report, or a standard referenced by an audit, receives no help.
1279 Thousand-page audit or financial filings have not been tested; Proposition 1 predicts
1280 the largest gains there, but also the largest exposure to cross-section dependencies.
1281 The routing agent also suggests how such cases could be handled without giving up
1282 locality. The natural extension is edge-to-cloud collaborative routing: the agent already
1283 exposes a confidence margin, so near-tie chunks, or questions a local generator cannot

answer from the retrieved context, can be escalated selectively to a larger model while low-entropy work stays local. Next steps are a router run on the policy corpus with policy-specific exemplars, memory instrumentation during the benchmark runs themselves rather than after the fact (Sect. 3.6), and sufficient repeated generations to replace single-run bands with interval estimates.

5 Conclusions

Index-time contextual retrieval can be made to fit the memory, time, and energy budget of a single consumer GPU without giving up its accuracy: bounding each chunk’s context to a window of neighbors makes prompt length and reserved memory independent of document length, and a one-token routing decision of the same small model removes a third of the summaries and most of the remaining cost. On 250 multi-hop questions over 25 scientific papers the routed pipeline matched full-document contextualization within the noise of a single run (Recall@20 0.952 against 0.954) with better ranking metrics, at 12 % of its generation energy and 16 % of its generation time, and the fixed-radius variant transferred to European Commission policy documents with the same accuracy gap and a 72 % energy saving. The pipeline is precise (its metrics are reproducible per question), explainable (every routing decision is logged with its evidence), and provable (every run carries an energy and emission ledger), the PEP-AI footing we consider necessary for AI-assisted compliance auditing at the edge of the energy system.

Declarations

Ethics approval and consent to participate. Not applicable.

Consent for publication. Not applicable.

Availability of data and materials. The data and code supporting this article
 are in the public project repository at [71]; the revision used for this article (v1.0.0)
 is archived at Zenodo under the DOI <https://doi.org/10.5281/zenodo.23337978>, so
 that it remains available independently of GitHub. The code is released under the
 MIT license and the data and figures produced by the authors under CC BY 4.0; the
 source PDFs keep the licenses of their publishers. Under `RnD/` the repository holds:
 the source PDFs of the scientific corpus in `input/` and of the exemplar papers in
`input_router/`; the frozen chunk files in `split_documents/` (scientific corpus), `split_`
`documents/smoking/` (policy corpus), `split_documents_router/` (exemplar papers)
 and `split_documents_end_to_end/` (the end-to-end run of Sect. 3.2); the question
 sets in `q_and_a/Gemini/scientific_multi_chunk_control.json` and `q_and_a/`
`Gemini/smoking.json`; the generated contexts of every run in `preprocessed_chunks/`;
 the indexed LanceDB tables in `db/`; the per-run energy ledger `emissions_data/`
`emissions.csv`; the end-to-end timings in `benchmarking_results/preprocessing/`;
 the hand-labeled radius set and routing records in `dynamic_slice_length/`, with the
 live-run records in `dynamic_slice_length/router_experiments/real_runs/` and
 the per-question retrieval results and policy simulations in `dynamic_slice_length/`
`router_experiments/results/`; the complete pipeline (parsing, routing agent, slice
 generation, indexing, evaluation and the offline policy simulator) and the notebooks of
 every reported run; the verification scripts that recompute each reported number from
 these records in `RnD/verification/` (its README maps each script to the table it
 checks; some scripts need a GPU and a running Ollama server); and the scripts that
 draw the figures in `RnD/nature_article/images/`. The policy-corpus PDFs are the
 public European Commission documents named in Table 1. Five of the 25 papers
 of the scientific corpus are distributed by their publishers under licenses that do not
 permit redistribution; their source PDFs, frozen chunk files and gold radius labels
 are therefore not included, the public copy covers the remaining 20 papers, and the

corpus-level statistics that depend on all 25 papers can be regenerated in full only once
those five papers are added back from the publishers' sites. The question sets, the
energy ledger, the routing and retrieval records and the per-question results of every
reported run are complete. Because the summarizer's output is not bit-reproducible
(Sect. 3.3), regenerated summaries differ from the archived ones within the single-run
band reported there.

Competing interests. The authors declare that they have no competing interests.

Funding. This research received no specific funding.

Authors' contributions. M.T.S.: conceptualization, methodology, software, mea-
surements, formal analysis, writing of the original draft. R.L.: conceptualization,
supervision, review and editing of the manuscript. All authors read and approved the
final manuscript.

Acknowledgements. An earlier version of this work was presented at the Student
Research Conference of the University of Debrecen (UD) and published as a conference
paper [70]. It was supported by the local UD Talent Program, the Doctoral School of
Informatics, and the Faculty of Informatics of the University of Debrecen.

Use of generative AI tools. The authors acknowledge the use of generative
artificial intelligence tools, namely Gemini (Google), Claude (Anthropic), ChatGPT
(OpenAI), and Grammarly, in preparing this manuscript and the associated research: for
language editing; Gemini for the initial literature survey and for generating the silver-
standard question sets under the protocol of Sect. 2.1; Gemini and ChatGPT as coding
assistants; and Claude to draft and condense this journal version from the conference
manuscript and the repository's experimental records, and to run the dynamic-radius
routing experiments under the authors' direction. The core concepts, methodology,
and scientific contributions are the authors' original work; all AI-generated outputs
were produced under human supervision, validated manually, and critically reviewed

1427 and edited by the authors, who remain fully responsible for the accuracy, integrity,
1428
1429 and originality of the content.

1430

1431

1432 **References**

1433

1434 [1] Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language
1435 model. *Journal of Machine Learning Research* **3**(Feb), 1137–1155 (2003)

1436

1437
1438 [2] Zhu, Y., Yuan, H., Wang, S., Liu, J., Liu, W., Deng, C., Chen, H., Liu, Z., Dou,
1439 Z., Wen, J.-R.: Large language models for information retrieval: A survey. *ACM*
1440 *Transactions on Information Systems* **44**(1), 1–54 (2025) [https://doi.org/10.1145/](https://doi.org/10.1145/3748304)
1441 [3748304](https://doi.org/10.1145/3748304)

1442

1443
1444 [3] Besiroglu, T., Bergerson, S.A., Michael, A., Heim, L., Luo, X., Thompson, N.:
1445 The Compute Divide in Machine Learning: A Threat to Academic Contribution
1446 and Scrutiny? <https://arxiv.org/abs/2401.02452> (Retrieved: 2026-01-24) (2024)

1447

1448
1449 [4] Sevilla, J., Heim, L., Ho, A., Besiroglu, T., Hobbhahn, M., Villalobos, P.: Compute
1450 trends across three eras of machine learning. In: 2022 International Joint Conference
1451 on Neural Networks (IJCNN), pp. 1–8. IEEE, Piscataway, NJ, USA (2022).
1452 <https://doi.org/10.1109/ijcnn55064.2022.9891914>

1453

1454
1455 [5] Thompson, N.C., Greenewald, K., Lee, K., Manso, G.F.: The Computational
1456 Limits of Deep Learning (2020)

1457

1458
1459 [6] Hooker, S.: The hardware lottery. *Communications of the ACM* **64**(12), 58–65
1460 (2021) <https://doi.org/10.1145/3467017>

1461

1462
1463 [7] Schwartz, R., Dodge, J., Smith, N.A., Etzioni, O.: Green ai. *Communications of*
1464 *the ACM* **63**(12), 54–63 (2020) <https://doi.org/10.1145/3381831>

1465

1466
1467 [8] Strubell, E., Ganesh, A., McCallum, A.: Energy and policy considerations for deep
1468

- learning in NLP. In: Korhonen, A., Traum, D., Màrquez, L. (eds.) Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3645–3650. Association for Computational Linguistics, Florence, Italy (2019). <https://doi.org/10.18653/v1/P19-1355> . <https://aclanthology.org/P19-1355/>
- [9] Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.-M., Rothchild, D., So, D., Texier, M., Dean, J.: Carbon Emissions and Large Neural Network Training (2021)
- [10] Luccioni, S., Jernite, Y., Strubell, E.: Power hungry processing: Watts driving the cost of ai deployment? In: The 2024 ACM Conference on Fairness, Accountability and Transparency. FAccT '24, pp. 85–99. ACM, New York, NY, USA (2024). <https://doi.org/10.1145/3630106.3658542>
- [11] Li, P., Yang, J., Islam, M.A., Ren, S.: Making AI Less "Thirsty": Uncovering and Addressing the Secret Water Footprint of AI Models (2025). <https://arxiv.org/abs/2304.03271>
- [12] Vries, A.: The growing energy footprint of artificial intelligence. *Joule* **7**(10), 2191–2194 (2023) <https://doi.org/10.1016/j.joule.2023.09.004>
- [13] Wu, C.-J., Raghavendra, R., Gupta, U., Acun, B., Ardalani, N., Maeng, K., Chang, G., Aga, F., Huang, J., Bai, C., *et al.*: Sustainable AI: Environmental implications, challenges and opportunities. In: Proceedings of Machine Learning and Systems 4 (MLSys 2022) (2022). arXiv:2111.00364
- [14] Henderson, P., Hu, J., Romoff, J., Brunskill, E., Jurafsky, D., Pineau, J.: Towards the systematic reporting of the energy and carbon footprints of machine learning. *Journal of Machine Learning Research* **21**(248), 1–43 (2020). arXiv:2002.05651
- [15] Lacoste, A., Luccioni, A., Schmidt, V., Dandres, T.: Quantifying the Carbon

1519 Emissions of Machine Learning (2019). <https://arxiv.org/abs/1910.09700>
1520
1521 [16] Lannelongue, L., Grealey, J., Inouye, M.: Green algorithms: Quantifying the
1522 carbon footprint of computation. *Advanced Science* **8**(12), 2100707 (2021) [https:](https://doi.org/10.1002/advs.202100707)
1523 [//doi.org/10.1002/advs.202100707](https://doi.org/10.1002/advs.202100707)
1524
1525
1526
1527 [17] Tabbakh, A., Al Amin, L., Islam, M., Mahmud, G.M.I., Chowdhury, I., Mukta,
1528 M.: Towards sustainable ai: a comprehensive framework for green ai. *Discover*
1529 *Sustainability* **5** (2024) <https://doi.org/10.1007/s43621-024-00641-4>
1530
1531
1532
1533 [18] European Parliament and Council of the European Union: Directive (EU)
1534 2023/1791 of 13 September 2023 on energy efficiency and amending Regula-
1535 tion (EU) 2023/955 (recast). *Official Journal of the European Union*, L 231, 20
1536 September 2023, pp. 1–111 (2023)
1537
1538
1539
1540 [19] European Parliament and Council of the European Union: Regulation (EU)
1541 2018/1999 of 11 December 2018 on the Governance of the Energy Union and
1542 Climate Action. *Official Journal of the European Union*, L 328, 21 December 2018,
1543 pp. 1–77 (2018)
1544
1545
1546
1547
1548 [20] European Commission: Digitalising the energy system – EU action plan.
1549 Communication COM(2022) 552 final, Brussels, 18 October 2022 (2022)
1550
1551
1552 [21] European Parliament and Council of the European Union: Regulation (EU)
1553 2016/679 of 27 April 2016 on the protection of natural persons with regard to the
1554 processing of personal data and on the free movement of such data (General Data
1555 Protection Regulation). *Official Journal of the European Union*, L 119, 4 May
1556 2016, pp. 1–88 (2016)
1557
1558
1559
1560
1561 [22] European Parliament and Council of the European Union: Regulation (EU)
1562 2024/1689 of 13 June 2024 laying down harmonised rules on artificial intelligence
1563
1564

- (Artificial Intelligence Act). Official Journal of the European Union, L, 12 July 2024 (2024)
- [23] Jørgensen, B., Ma, Z.: Regulating ai in the energy sector: A scoping review of eu laws, challenges, and global perspectives. *Energies* **18**, 2359 (2025) <https://doi.org/10.3390/en18092359>
- [24] Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., Zhang, J.: Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE* **107**(8), 1738–1762 (2019) <https://doi.org/10.1109/JPROC.2019.2918951>
- [25] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. <https://arxiv.org/abs/2005.11401> (Retrieved: Dec. 27, 2025) (2021)
- [26] Vinayan Kozhipuram, A., Shailendra, S., Kadel, R.: Retrieval-augmented generation vs. baseline llms: A multi-metric evaluation for knowledge-intensive content. *Information* **16**(9) (2025) <https://doi.org/10.3390/info16090766>
- [27] Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., Shoham, Y.: In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics* **11**, 1316–1331 (2023) <https://doi.org/10.1162/tac1.a.00605>
- [28] Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., Yih, W.-t.: Dense Passage Retrieval for Open-Domain Question Answering (2020). <https://arxiv.org/abs/2004.04906>
- [29] Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M.:

1611 Okapi at TREC-3. In: Proceedings of the Third Text REtrieval Conference (TREC-
1612 3). NIST Special Publication 500-225, pp. 109–126. National Institute of Standards
1613 and Technology, Gaithersburg, MD, USA (1995)
1614
1615
1616
1617 [30] Robertson, S., Zaragoza, H.: The probabilistic relevance framework: BM25 and
1618 beyond. *Foundations and Trends in Information Retrieval* **3**(4), 333–389 (2009)
1619 <https://doi.org/10.1561/15000000019>
1620
1621
1622 [31] Khattab, O., Zaharia, M.: ColBERT: Efficient and effective passage search
1623 via contextualized late interaction over BERT. In: Proceedings of the 43rd
1624 International ACM SIGIR Conference on Research and Development in Infor-
1625 mation Retrieval (SIGIR 2020), pp. 39–48. ACM, New York, NY, USA (2020).
1626 <https://doi.org/10.1145/3397271.3401075>
1627
1628
1629 [32] Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., Zaharia, M.: ColBERTv2:
1630 Effective and Efficient Retrieval via Lightweight Late Interaction. [https://arxiv.](https://arxiv.org/abs/2112.01488)
1631 [org/abs/2112.01488](https://arxiv.org/abs/2112.01488) (Retrieved 2026-01-02) (2022)
1632
1633
1634 [33] Kiran, S.: Hybrid retrieval-augmented generation (rag) systems with embedding
1635 vector databases. *International Journal of Scientific Research in Computer Science,*
1636 *Engineering and Information Technology* **11**, 2694–2702 (2025) [https://doi.org/](https://doi.org/10.32628/CSEIT25112702)
1637 [10.32628/CSEIT25112702](https://doi.org/10.32628/CSEIT25112702)
1638
1639 [34] Wang, X., Wang, Z., Gao, X., Zhang, F., Wu, Y., Xu, Z., Shi, T., Wang, Z.,
1640 Li, S., Qian, Q., Yin, R., Lv, C., Zheng, X., Huang, X.: Searching for best
1641 practices in retrieval-augmented generation. In: Proceedings of the 2024 Conference
1642 on Empirical Methods in Natural Language Processing (EMNLP 2024) (2024).
1643 arXiv:2407.01219
1644
1645 [35] Merola, C., Singh, J.: Reconstructing Context: Evaluating Advanced Chunking
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656

- Strategies for Retrieval-Augmented Generation. <https://arxiv.org/abs/2504.19754> (Retrieved 2026-01-02) (2025)
- [36] Smith, B., Troynikov, A.: Evaluating chunking strategies for retrieval. Technical report, Chroma (July 2024)
- [37] Qu, R., Tu, R., Bao, F.: Is Semantic Chunking Worth the Computational Cost? <https://arxiv.org/abs/2410.13070> (Retrieved: 2026-01-24) (2024)
- [38] Chen, T., Wang, H., Chen, S., Yu, W., Ma, K., Zhao, X., Zhang, H., Yu, D.: Dense X Retrieval: What Retrieval Granularity Should We Use? (2023)
- [39] Auer, C., Lysak, M., Nassar, A., Dolfi, M., Livathinos, N., Vagenas, P., Ramis, C.B., Omenetti, M., Lindlbauer, F., Dinkla, K., Mishra, L., Kim, Y., Gupta, S., Lima, R.T., Weber, V., Morin, L., Meijer, I., Kuropiatnyk, V., Staar, P.W.J.: Docling Technical Report. <https://arxiv.org/abs/2408.09869> (Retrieved 2026-01-02) (2024)
- [40] Verma, P.: S2 Chunking: A Hybrid Framework for Document Segmentation Through Integrated Spatial and Semantic Analysis. <https://arxiv.org/abs/2501.05485> (Retrieved: 2026-01-24) (2025)
- [41] Günther, M., Mohr, I., Williams, D.J., Wang, B., Xiao, H.: Late Chunking: Contextual Chunk Embeddings Using Long-Context Embedding Models (2024)
- [42] Anthropic: Introducing Contextual Retrieval. <https://www.anthropic.com/news/contextual-retrieval> (Retrieved: 2025-12-31) (2024)
- [43] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems 30 (NeurIPS 2017), pp. 5998–6008 (2017). arXiv:1706.03762
- [44] Tay, Y., Dehghani, M., Bahri, D., Metzler, D.: Efficient transformers: A survey.

1703 ACM Computing Surveys **55**(6), 1–28 (2022) <https://doi.org/10.1145/3530811>
1704
1705 [45] Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R.,
1706 Gray, S., Radford, A., Wu, J., Amodei, D.: Scaling Laws for Neural Language
1707 Models (2020)
1708
1709
1710
1711 [46] Pope, R., Douglas, S., Chowdhery, A., Devlin, J., Bradbury, J., Heek, J., Xiao, K.,
1712 Agrawal, S., Dean, J.: Efficiently scaling transformer inference. In: Proceedings of
1713 Machine Learning and Systems 5 (MLSys 2023) (2023). arXiv:2211.05102
1714
1715
1716
1717 [47] Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C.H., Gonzalez, J.E.,
1718 Zhang, H., Stoica, I.: Efficient memory management for large language model
1719 serving with PagedAttention. In: Proceedings of the 29th Symposium on Operating
1720 Systems Principles (SOSP 2023). ACM, New York, NY, USA (2023). <https://doi.org/10.1145/3600006.3613165> . arXiv:2309.06180
1721
1722
1723
1724
1725
1726 [48] Ainslie, J., Lee-Thorp, J., Jong, M., Zemlyanskiy, Y., Lebrón, F., Sanghai, S.: GQA:
1727 Training generalized multi-query transformer models from multi-head checkpoints.
1728 In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language
1729 Processing (EMNLP 2023) (2023). arXiv:2305.13245
1730
1731
1732
1733
1734 [49] Sheng, Y., Zheng, L., Yuan, B., Li, Z., Ryabinin, M., Chen, B., Liang, P., Ré,
1735 C., Stoica, I., Zhang, C.: FlexGen: High-throughput generative inference of large
1736 language models with a single GPU. In: Proceedings of the 40th International
1737 Conference on Machine Learning (ICML 2023) (2023). arXiv:2303.06865
1738
1739
1740
1741 [50] Dao, T., Fu, D.Y., Ermon, S., Rudra, A., Ré, C.: FlashAttention: Fast and memory-
1742 efficient exact attention with IO-awareness. In: Advances in Neural Information
1743 Processing Systems 35 (NeurIPS 2022) (2022). arXiv:2205.14135
1744
1745
1746
1747 [51] Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The Long-Document Transformer
1748

- (2020) 1749
1750
- [52] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., 1751
Kalenichenko, D.: Quantization and training of neural networks for efficient 1752
integer-arithmetic-only inference. In: Proceedings of the IEEE Conference on 1753
Computer Vision and Pattern Recognition (CVPR 2018), pp. 2704–2713 (2018). 1754
<https://doi.org/10.1109/CVPR.2018.00286> 1755
1756
1757
1758
1759
- [53] Shen, M., Umar, M., Maeng, K., Suh, G.E., Gupta, U.: Towards Understanding 1760
Systems Trade-offs in Retrieval-Augmented Generation Model Inference. [https:](https://arxiv.org/abs/2412.11854) 1761
[/arxiv.org/abs/2412.11854](https://arxiv.org/abs/2412.11854) (Retrieved: 2025-12-31) (2024) 1762
1763
1764
1765
- [54] Zhao, S., Shao, Y., Huang, Y., Song, J., Wang, Z., Wan, C., Ma, L.: Understanding 1766
the Design Decisions of Retrieval-Augmented Generation Systems. [https://arxiv.](https://arxiv.org/abs/2411.19463) 1767
[org/abs/2411.19463](https://arxiv.org/abs/2411.19463) (Retrieved: 2025-12-31) (2025) 1768
1769
1770
1771
- [55] Samsi, S., Zhao, D., McDonald, J., Li, B., Michaleas, A., Jones, M., Bergeron, W., 1772
Kepner, J., Tiwari, D., Gadepally, V.: From words to watts: Benchmarking the 1773
energy costs of large language model inference. In: 2023 IEEE High Performance 1774
Extreme Computing Conference (HPEC), pp. 1–9. IEEE, Piscataway, NJ, USA 1775
(2023). <https://doi.org/10.1109/HPEC58863.2023.10363447> . arXiv:2310.03003 1776
1777
1778
1779
1780
- [56] Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, 1781
P.: Lost in the middle: How language models use long contexts. Transactions of 1782
the Association for Computational Linguistics **12**, 157–173 (2024) [https://doi.org/](https://doi.org/10.1162/tacl_a_00638) 1783
[10.1162/tacl_a_00638](https://doi.org/10.1162/tacl_a_00638) . Open access under CC BY 4.0; preprint arXiv:2307.03172 1784
1785
1786
1787
1788
- [57] Guo, X., Vosoughi, S.: Serial Position Effects of Large Language Models. [https:](https://arxiv.org/abs/2406.15981) 1789
[/arxiv.org/abs/2406.15981](https://arxiv.org/abs/2406.15981) (Retrieved 2026-01-02) (2024) 1790
1791
1792
- [58] Menschikov, M., Kharitonov, A., Kotyga, M., Porvatov, V., Zhukovskaya, A., 1793
1794

1795 Kagramanyan, D., Shvetsov, E., Burnaev, E.: Beyond Early-Token Bias: Model-
1796 Specific and Language-Specific Position Effects in Multilingual LLMs. <https://arxiv.org/abs/2505.16134> (Retrieved 2026-01-02) (2025)
1797
1798
1799
1800
1801 [59] Hsieh, C.-Y., Chuang, Y.-S., Li, C.-L., Wang, Z., Le, L.T., Kumar, A., Glass, J.,
1802 Ratner, A., Lee, C.-Y., Krishna, R., Pfister, T.: Found in the Middle: Calibrating
1803 Positional Attention Bias Improves Long Context Utilization (2024)
1804
1805
1806
1807 [60] Lee, R.J., Goel, S., Ramchandran, K.: Quantifying Positional Biases in Text
1808 Embedding Models. <https://arxiv.org/abs/2412.15241> (Retrieved 2026-01-02)
1809 (2025)
1810
1811
1812 [61] Yao, J., Liu, S., Wang, Y., Mei, L., Bi, B., Ge, Y., Li, Z., Cheng, X.: Who is in
1813 the Spotlight: The Hidden Bias Undermining Multimodal Retrieval-Augmented
1814 Generation. <https://arxiv.org/abs/2506.11063> (Retrieved: 2026-01-24) (2025)
1815
1816
1817
1818 [62] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M.,
1819 Wang, H.: Retrieval-Augmented Generation for Large Language Models: A Survey
1820 (2024). <https://arxiv.org/abs/2312.10997>
1821
1822
1823
1824 [63] Liu, J.: LlamaIndex. GitHub repository. Accessed: Oct. 19, 2026 (2022). https://github.com/run-llama/llama_index
1825
1826
1827
1828 [64] Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., Manning, C.D.: RAPTOR:
1829 Recursive Abstractive Processing for Tree-Organized Retrieval (2024). <https://arxiv.org/abs/2401.18059>
1830
1831
1832
1833
1834 [65] Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S.,
1835 Larson, J.: From Local to Global: A Graph RAG Approach to Query-Focused
1836 Summarization (2024)
1837
1838
1839
1840

- [66] Jeong, S., Baek, J., Cho, S., Hwang, S.J., Park, J.C.: Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2024) (2024). arXiv:2403.14403
- [67] Asai, A., Wu, Z., Wang, Y., Sil, A., Hajishirzi, H.: Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In: The Twelfth International Conference on Learning Representations (ICLR 2024) (2024). arXiv:2310.11511
- [68] Ong, I., Almahairi, A., Wu, V., Chiang, W.-L., Wu, T., Gonzalez, J.E., Kadous, M.W., Stoica, I.: RouteLLM: Learning to Route LLMs with Preference Data (2024)
- [69] Chen, L., Zaharia, M., Zou, J.: FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance (2023)
- [70] Sándor, M.T., Lakatos, R.: Efficient local retrieval-augmented generation via neighboring document slice contextualization. In: 2026 IEEE 4th Conference on Information Technology and Data Science (CITDS), pp. 257–262 (2026). <https://doi.org/10.1109/CITDS70571.2026.11706860>
- [71] Sándor, M.T., Lakatos, R.: Document Slice Contextual Retrieval, public repository, version v1.0.0: software, data and verification scripts for the dynamic document slice article. Zenodo. Software, data and verification scripts; version v1.0.0, 13 October 2026 (2026). <https://doi.org/10.5281/zenodo.23337978> . <https://github.com/martinsandor707/Document.Slice.Contextual.Retrieval.Public>
- [72] Gemma Team, Google DeepMind: Gemma 3 Technical Report (2025). <https://arxiv.org/abs/2503.19786>

1887 [73] Ollama: Ollama: Get up and running with large language models locally. <https://ollama.com/>. Accessed: Apr. 18, 2026 (2024)

1888

1889

1890

1891 [74] Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z.,

1892 Li, Z., Li, D., Xing, E.P., Zhang, H., Gonzalez, J.E., Stoica, I.: Judging LLM-as-

1893 a-Judge with MT-Bench and Chatbot Arena. <https://arxiv.org/abs/2306.05685>

1894 (Retrieved: 2026-01-24) (2023)

1895

1896

1897

1898 [75] Rebholz-Schuhmann, D., Jimeno Yepes, A.J., Mulligen, E.M., Kang, N., Kors,

1899 J., Milward, D., Corbett, P., Buyko, E., Beisswanger, E., Hahn, U.: Calbc silver

1900 standard corpus. *Journal of Bioinformatics and Computational Biology* **08**(01),

1901 163–179 (2010) <https://doi.org/10.1142/S0219720010004562>

1902

1903

1904

1905

1906 [76] Livathinos, N., Auer, C., Nassar, A., Lima, R., Lysak, M., Ebouky, B.,

1907 Berrospi Ramis, C., Dolfi, M., Vagenas, P., Omenetti, M., Dinkla, K., Kim,

1908 Y., Weber, V., Morin, L., Meijer, I., Kuropiatnyk, V., Strohmeyer, T., Gur-

1909 buz, A., Staar, P.: Advanced Layout Analysis Models for Docling (2025). <https://doi.org/10.48550/arXiv.2509.11720>

1910

1911

1912

1913

1914

1915 [77] IBM Research: Docling Concepts: Chunking. Deep Search for Scientific Discovery

1916 (DS4SD), (2024). Deep Search for Scientific Discovery (DS4SD)

1917

1918

1919 [78] Nussbaum, Z., Morris, J.X., Duderstadt, B., Mulyar, A.: Nomic Embed: Training

1920 a Reproducible Long Context Text Embedder (2024)

1921

1922

1923

1924 [79] Muennighoff, N., Tazi, N., Magne, L., Reimers, N.: MTEB: Massive text embedding

1925 benchmark. In: Proceedings of the 17th Conference of the European Chapter

1926 of the Association for Computational Linguistics (EACL 2023), pp. 2014–2037

1927 (2023). arXiv:2210.07316

1928

1929

1930

1931 [80] LanceDB: LanceDB: Developer-friendly, serverless vector database. <https://www.lancedb.com/>

1932

- lancedb.com/. Accessed: Apr. 18, 2026 (2024) 1933
- [81] Voorhees, E.M.: The TREC-8 question answering track report. In: Proceedings 1934
of the Eighth Text REtrieval Conference (TREC-8). NIST Special Publication 1935
500-246, pp. 77–82. National Institute of Standards and Technology, Gaithersburg, 1936
MD, USA (1999) 1937
1938
1939
1940
1941
1942
- [82] Efron, B.: Better bootstrap confidence intervals. *Journal of the American Statisti-* 1943
cal Association **82**(397), 171–185 (1987) [https://doi.org/10.1080/01621459.1987.](https://doi.org/10.1080/01621459.1987.10478410) 1944
[10478410](https://doi.org/10.1080/01621459.1987.10478410) 1945
1946
1947
1948
- [83] Smucker, M.D., Allan, J., Carterette, B.: A comparison of statistical significance 1949
tests for information retrieval evaluation. In: Proceedings of the Sixteenth ACM 1950
Conference on Conference on Information and Knowledge Management. CIKM 1951
'07, pp. 623–632. Association for Computing Machinery, New York, NY, USA 1952
(2007). <https://doi.org/10.1145/1321440.1321528> 1953
1954
1955
1956
1957
- [84] Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Courn- 1958
peau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., *et al.*: SciPy 1.0: 1959
Fundamental algorithms for scientific computing in Python. *Nature Methods* **17**, 1960
261–272 (2020) <https://doi.org/10.1038/s41592-019-0686-2> 1961
1962
1963
1964
- [85] McNemar, Q.: Note on the sampling error of the difference between correlated 1965
proportions or percentages. *Psychometrika* **12**(2), 153–157 (1947) [https://doi.org/](https://doi.org/10.1007/BF02295996) 1966
[10.1007/BF02295996](https://doi.org/10.1007/BF02295996) 1967
1968
1969
1970
- [86] Dietterich, T.G.: Approximate statistical tests for comparing supervised clas- 1971
sification learning algorithms. *Neural Computation* **10**(7), 1895–1923 (1998) 1972
1973
1974
1975
1976
- [87] Courty, B., Schmidt, V., Goyal-Kamal, MarionCoutarel, Feld, B., Lecourt, J., 1977
1978

1979 LiamConnell, SabAmine, inimaz, supatomic, Léval, M., Blanche, L., Cruveiller, A.,
1980 ouminasara, Zhao, F., Joshi, A., Bogroff, A., Saboni, A., Lavoreille, H., Laskaris,
1981 N., Abati, E., Blank, D., Wang, Z., Catovic, A., alencon, Stechly, M., Bauer, C.,
1982 Lucas-Otavio, JPW, MinervaBooks: mlco2/codecarbon: v2.4.1. Zenodo (2024).
1983 <https://doi.org/10.5281/zenodo.11171501>
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024