Sparse RAG

Accelerating Inference of Retrieval-Augmented Generation via Sparse Context Selection

Authors: Yun Zhu[1], Jia-Chen Gu[3], Caitlin Sikora[2], Ho Ko[2], Yinxiao Liu[1], Chu-Cheng Lin[2], Lei Shu[1], Liangchen Luo[1], Lei Meng[2], Bang Liu[4], Jindong Chen[1]

[1]Google DeepMind, [2]Google, [3]University of California, Los Angeles, [4]Université de Montréal & Mila

Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) : A technique for generating output structures while using retrieved nearest-neighbor structures as a reference.

- Two steps: retrieval + generation
- Retrieval:
  - Key:
    - BM25, dense embedding (GTR, XTR, Contriever, etc.)
  - Value
    - Token embedding: knnLM, Spalm
    - Dense embedding: TOME
    - Raw text: Realm, RAG, FID, RETRO, MARGE, DENSPI, DENSEPHRASES
- Generation
  - Concat: Realm, RAG, FID(fusion in decoder)
  - Cross attention: Retro, MeMTransformer, TOME, MARGE

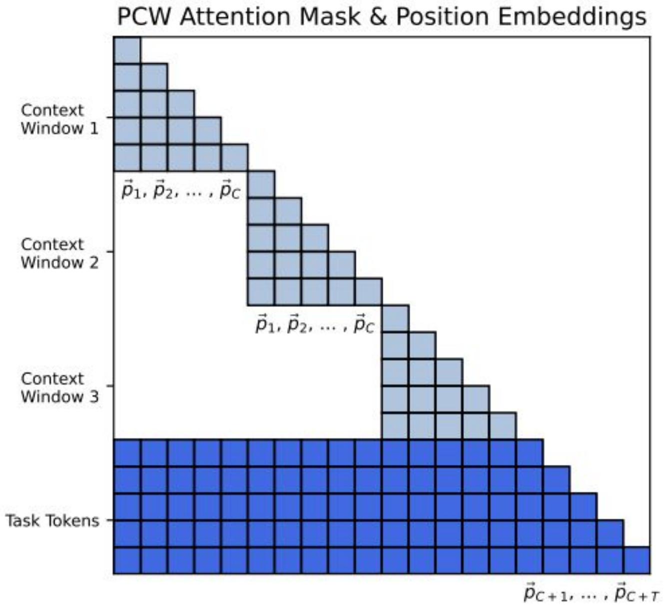
As the LLM becomes larger and larger, it is less practical to introduce extra encoder like FID or RETRO.

- RAG (concat input text) is becoming popular for being simple and practical.

Challenge #1 Latency

- Low Efficiency
  - Native “concat” of documents results in long context
  - Computational costs quadratic in context length

- Existing Solutions
  - Parallel context window (PCW) [2]
    - Block-wise attention. No need to re-train.
    - Downside:
      - Helps on prefill, but not decoding
      - Quality Issues



[2] Nir Ratner, etc. Parallel Context Windows for Large Language Models (PCW)

Challenge #2: Quality

- Error prone to irrelevant documents
  - LLM is not pre-trained to resist the wrong information
  - Quality bottleneck of “retriever”

- Existing Solutions
  - External Ranker/classifier e.g. [1]
    - Train a separate T5 classifier
    - Downside:
      - Maintain extra model & complex flow
      - Should be large enough (~1B) to get good quality (not good for resource constrained scenario like on-device)

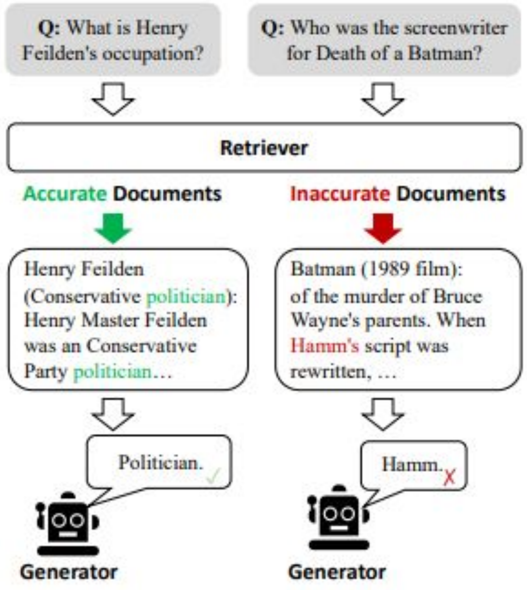


Figure 1: The examples show that a low-quality retriever is prone to introducing a substantial amount of irrelevant information, impeding the generators from acquiring accurate knowledge and potentially misleading them.

[1] Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, Zhen-Hua Ling, Corrective Retrieval Augmented Generation

Sparse RAG

- Sparse RAG
  - Mask inter-document attention scores
  - Number decode memories < Number prefill memories

- Self-selecting: Infuse “corrective” capability into LLM without extra model

Approach	Corrective	No extra Model	Prefill Efficiency	Decode efficiency
RAG	NO	YES	NO	NO
Corrective RAG	YES	NO	NO	NO
PCW RAG	NO	YES	YES	NO
Sparse RAG	YES	YES	YES	YES

Sparse RAG Process

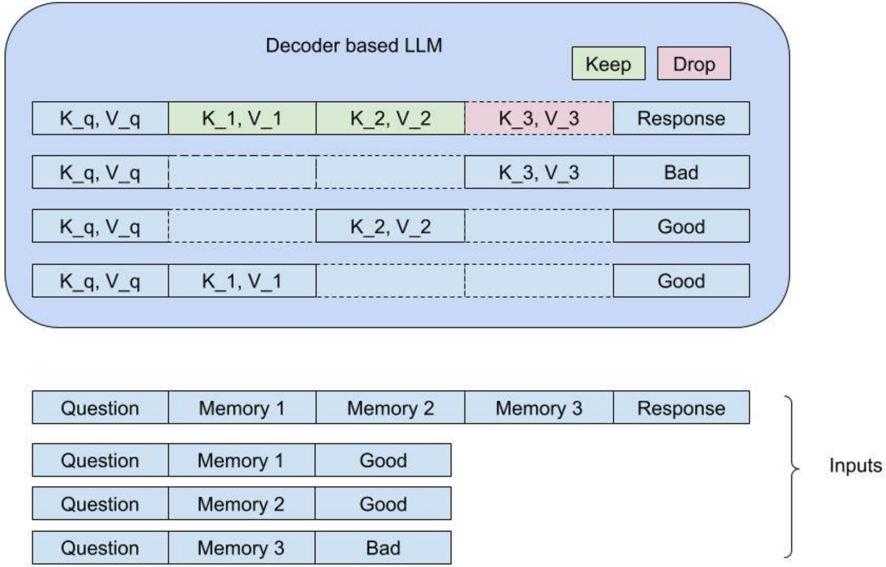
1. Create both “classification” and “generation” tasks.

2. Treat each context independently with PCW (block-wise attention mask).

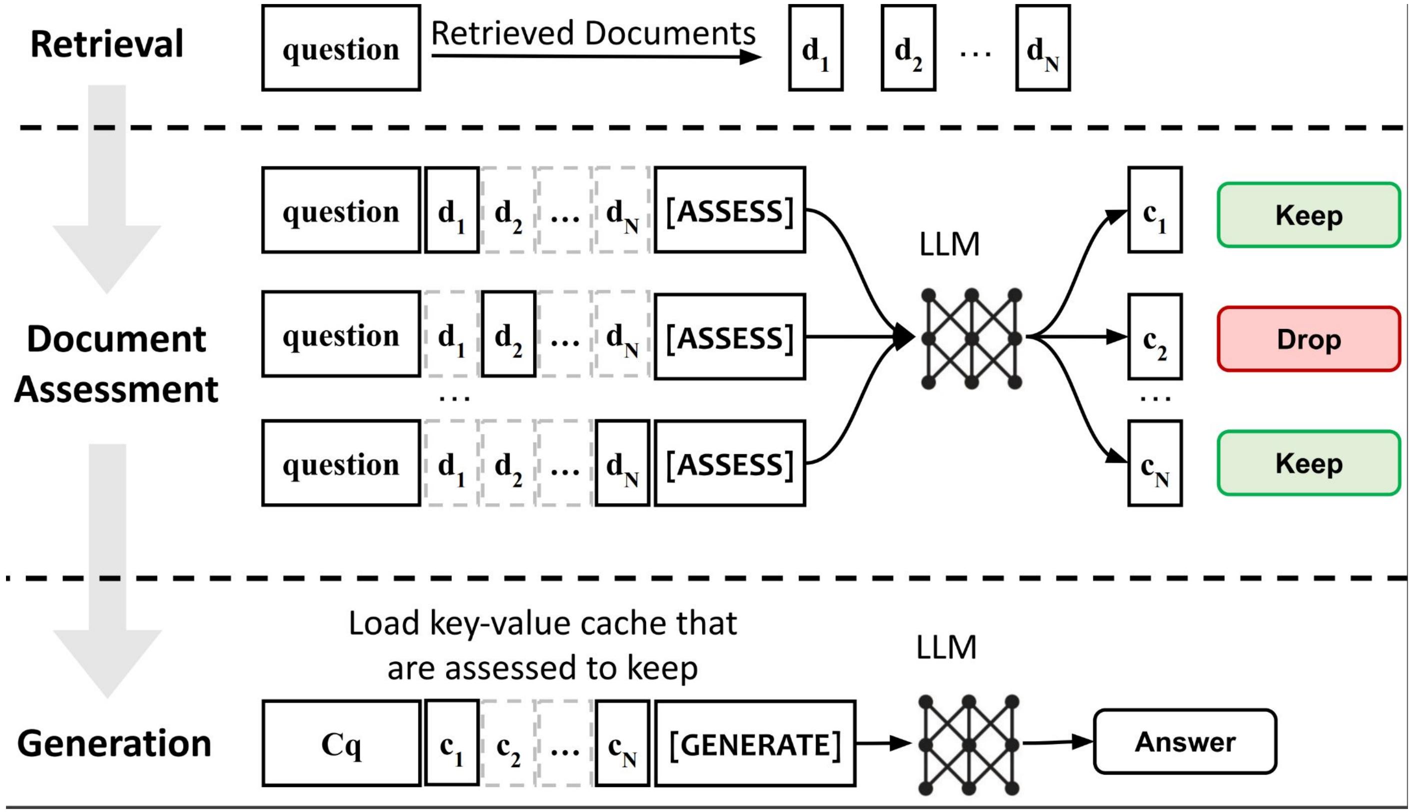
3. Rate each context and select top contexts in parallel (either topK or quality threshold).

4. Perform generation using top contexts.

5. Use block-wise attention mask for generation so you can reuse the KV cache for top documents.



Overview



Retrieval
question
Retrieved Documents
d_1
d_2
...
d_N
Document Assessment
question
d_1
d_2
...
d_N
[ASSESS]
LLM
c_1
Keep
c_2
Drop
c_N
Keep
Generation
Load key-value cache that are assessed to keep
Cq
c_1
c_2
...
c_N
[GENERATE]
LLM
Answer

Data Augmentation with LLMs

- For datasets that do not include golden per-context relevance labels, we use a multi-LLM labeling system to create them.

- We created a human-labeled ground truth subsample of the Natural Questions dataset and compared different model combinations to find the best performing labeling system, settling on Gemini Ultra with Palm2 XL as a critic model.

Auto-labeling method		Average F1	F1 Label 0	F1 Label 1
Rater model	Critic model			
PALM2 XL	n/a	0.729	0.765	0.694
PALM2 XL	PALM2 XL	0.781	0.820	0.741
Gemini Ultra	n/a	0.761	0.807	0.716
Gemini Ultra	Gemini Ultra	0.704	0.747	0.660
PALM2 XL	Gemini Ultra	0.728	0.776	0.680
Gemini Ultra	PALM2 XL	**0.821**	**0.861**	**0.781**

Experimental Setup

- Datasets
  - PopQA, QMSum, TriviaQA, HotpotQA
- Baselines
  - RAG - Concatenation RAG w/ full attention
  - LLM Lingua - Compressed prompt w/ full attention
  - PCW RAG - Parallel Context Window, no relevance assessment
  - Corrective RAG - Separate T5 relevance classifier
- Experiment config
  - XXS Gemini + LoRA (relevance classification and generation task w/ block-wise attention)
- Inference Setup and Metrics
  - Latency
    - Prefill stage: Encoding Speed (ES) (tokens per second t/s)
    - Decoding stage: Decoding Speed (DS) (tokens per second t/s)
  - Quality
    - Exact Match (EM) (binary exact match, averaged over responses)
    - RougeLSum for summarization quality
    - F1: Token-wise F1 score for response
      precision = # common tokens / prediction length
      recall = # common tokens / target length

Main Results

Approach		PopQA				QMSum (Summarization)				TriviaQA				HotpotQA (Multi-hop)			
	Encoding Speed (t/s)	F1	Exact Match	K	Decoding Speed (t/s)	F1	Rouge Sum	K	Decoding Speed (t/s)	Exact Match	F1	K	Decoding Speed (t/s)	Exact Match	F1	K	Decoding Speed (t/s)
RAG (closed book)	139.75	5.79	0	0	23.81	15.43	12.93	0	23.81	0	8.01	0	23.81	0	5.03	0	23.81
RAG (off-the-shelf)	56.28	12.76	0.33	20	6.65	20.37	12.67	20	6.65	0	12.21	20	6.65	0	11.75	10	10.31
LLMLingua	-	12.15	1.96	7.84	-	22.28	18.37	4.45	-	2.6	16.3	9.9	-	1.33	14.67	6.5	-
RAG	56.28	69.99	65.43	20	6.65	21.43	18.2	20	6.65	46.2	53.03	20	6.65	43	**55.85**	10	10.31
PCW-RAG	147.58	69.54	65.04	20	6.65	20.18	16.95	20	6.65	46	53.2	20	6.65	38.83	50.03	10	10.31
CRAG	-	70.99	66.52	8.9	-	-	-	-	-	-	-	-	-	-	-	-	-
Sparse RAG	**147.58**	**71.16**	**67.71**	**7.84**	**12.28**	**23.96**	**20.1**	**4.45**	**16.05**	**47.05**	**55.1**	**9.9**	**10.18**	**43.5**	55.3	**6.5**	**13**

Speed
higher = better

Quality
higher = better

Input length
lower = better

Conclusion

- Sparse RAG effectively addresses the computational burdens of RAG.
  - Masking inter-document attention
  - Parallel context assessment and selection
  - Efficient KV cache management

- Improves quality by attending only to highly relevant tokens.

- Outperforms baselines in latency and quality on 4 datasets.
  (long/short form generation, QA/Summarization tasks, multi-hop reasoning)

- Works with a single model that is small enough to run on-device.

Acknowledgements

Thanks to all of our collaborators!

Thanks to the conference organizers for creating this opportunity to share our findings with the broader research community.