Retrieval-Enhanced Machine Learning
Synthesis and Opportunities

To Eun Kim
Carnegie Mellon University

Alireza Salemi
University of Massachusetts Amherst

Andrew Drozdov
Databricks

Fernando Diaz
Carnegie Mellon University

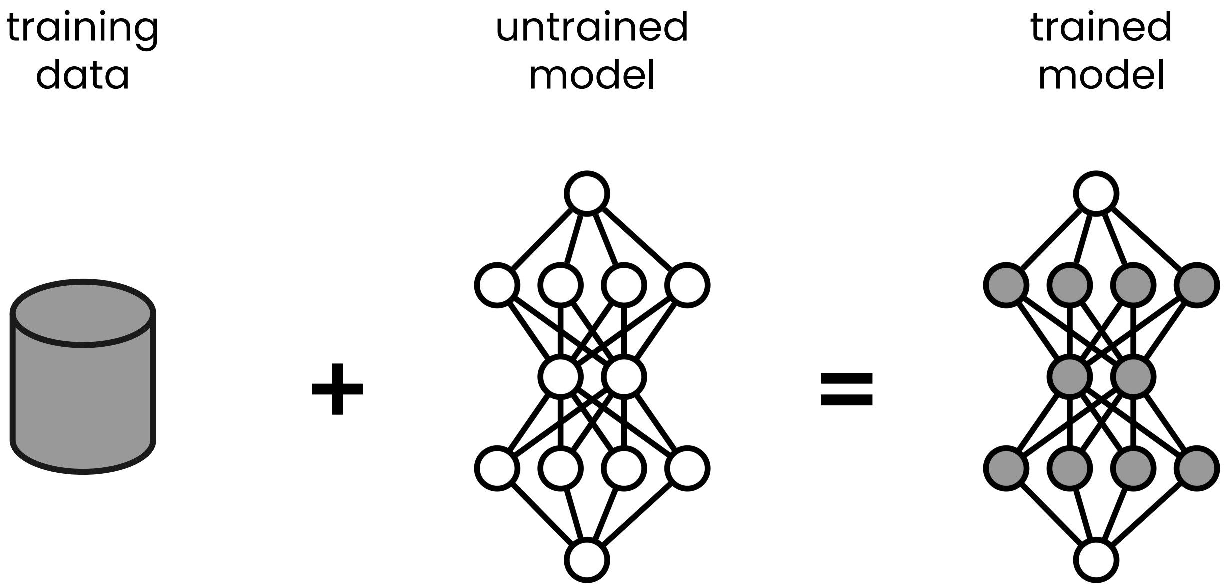
Hamed Zamani
University of Massachusetts Amherst

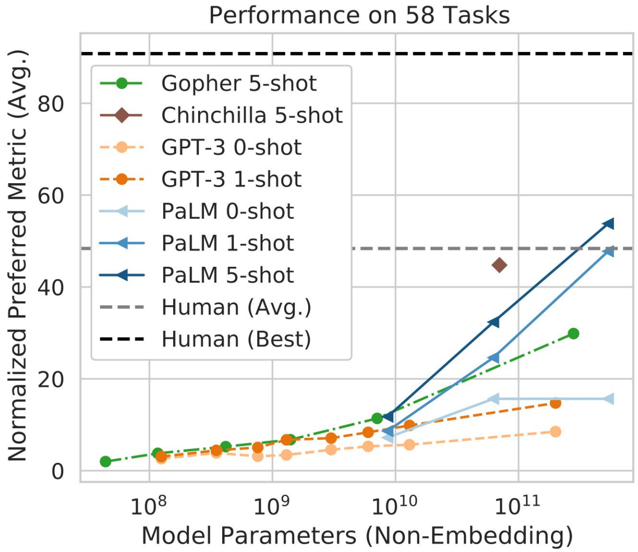
SIGIR-AP 2024

https://retrieval-enhanced-ml.github.io/sigir-ap2024-tutorial/

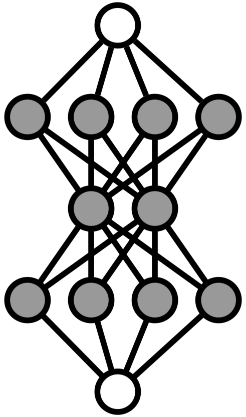
December 9, 2024

Introduction to REML


training data
+
untrained model
=
trained model

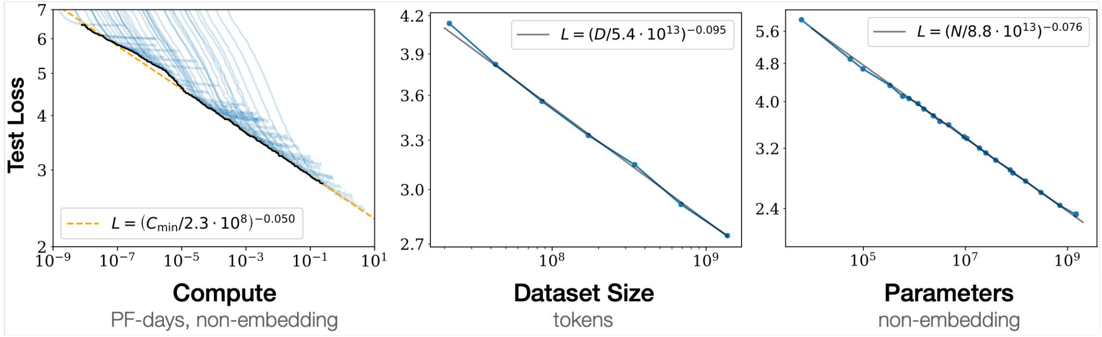


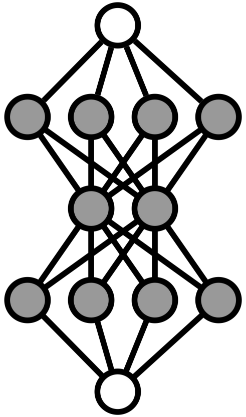
trained model



A Chowdhery, S Narang, J Devlin *et al*. PALM: Scaling Language Modeling with Pathways. 2022.

trained
model

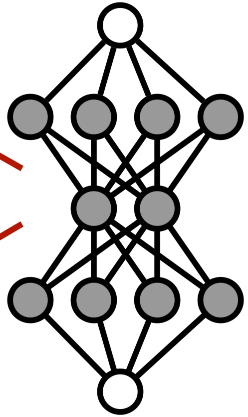




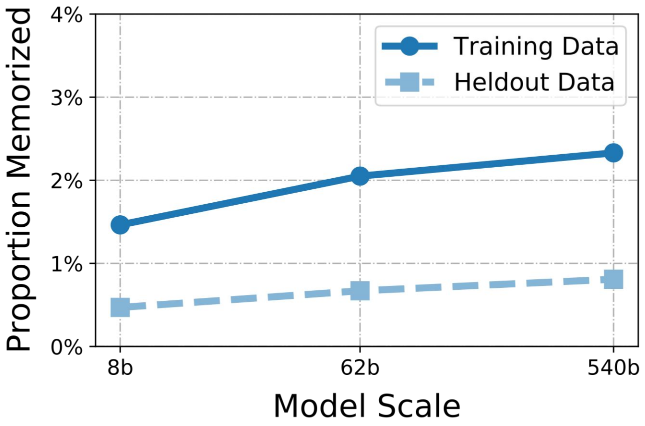
J Kaplan, S McCandlish, T Henighan, T Brown, B Chess, R Child, S Gray, A Radford, J Wu, D Amodei. Scaling laws for neural language models. 2020.

trained
model

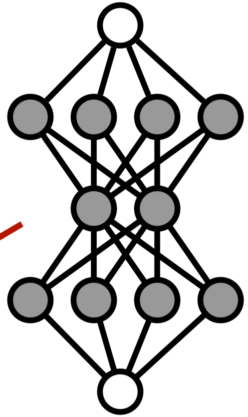
reasoning
(e.g. similarity,
transformation)



knowledge
(e.g. training instances, derived
information)



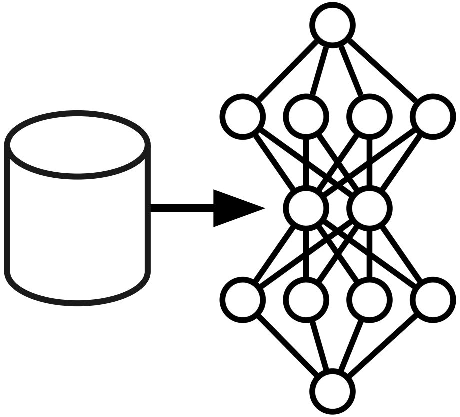
trained
model



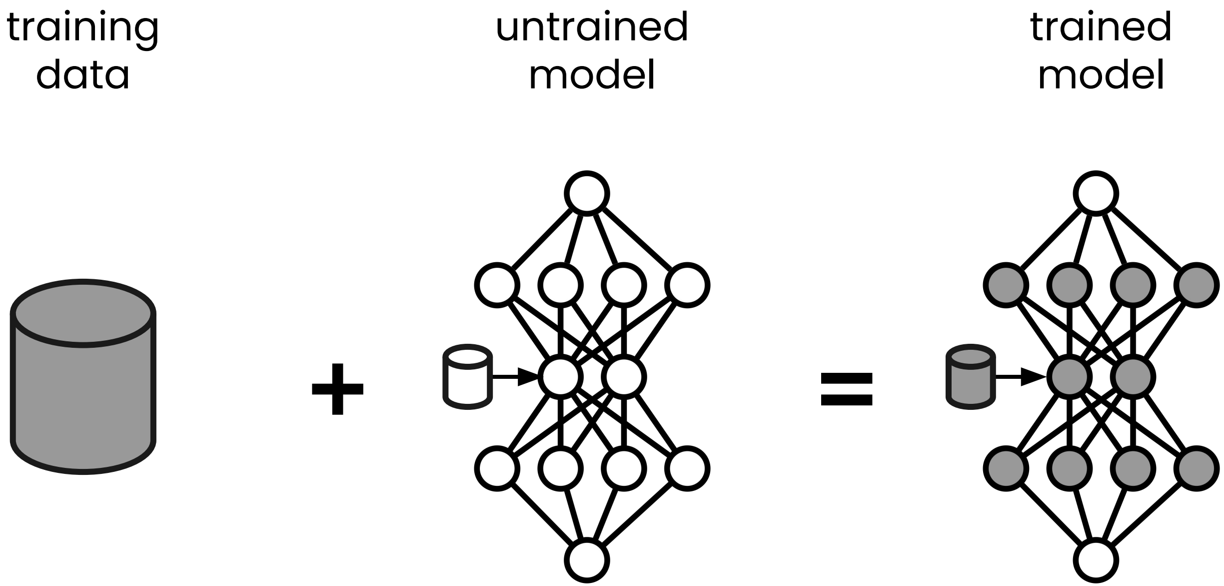
knowledge
(e.g. training instances, derived information)

A Chowdhery, S Narang, J Devlin *et al*. PALM: Scaling Language Modeling with Pathways. 2022.

Retrieval-Enhanced Machine
Learning
(REML)



explicitly support knowledge with access to infinite capacity external storage



learn what to store and how to access

benefits of REML

Model	Retrieval Set	#Database tokens	#Database keys	Valid	Test
Baseline transformer (ours)	-	-	-	21.53	22.96
*k*NN-LM (ours)	Wikipedia	4B	4B	18.52	19.54
Retro	Wikipedia	4B	0.06B	18.46	18.97
Retro	C4	174B	2.9B	12.87	10.23
Retro	MassiveText (1%)	18B	0.8B	18.92	20.33
Retro	MassiveText (10%)	179B	4B	13.54	14.95
Retro	MassiveText (100%)	1792B	28B	**3.21**	**3.92**

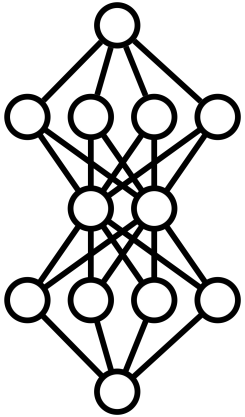
S Borgeaud, A Mensch, J Hoffmann, *et al*. Improving language models by retrieving from trillions of tokens. 2021.

benefits of REML

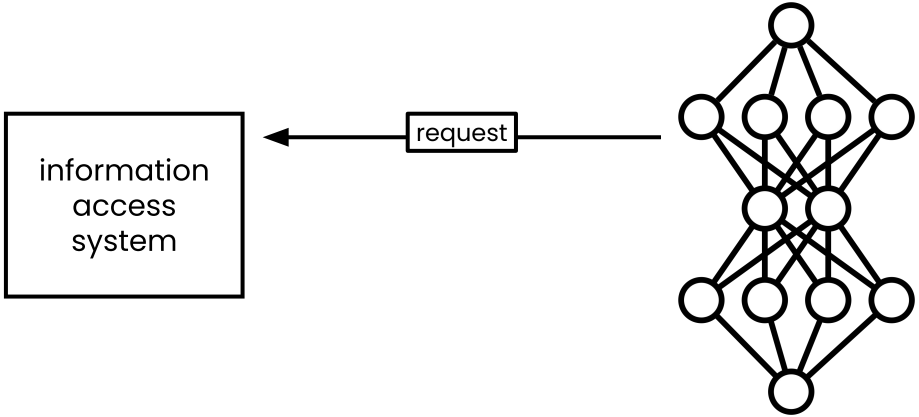
- generalization: concepts not limited by capacity of parameters.
- scalability: parameters offloaded to efficient indexing and retrieval data structures.
- updating: new data can be incorporated into indexing, not retraining.
- transparency: inference can be attributed to specific retrieval requests and results.
- on-device ML: limited capacity machines can perform inference with access to a search API.

Retrieval-Enhanced Machine Learning
(REML)

information
access
system

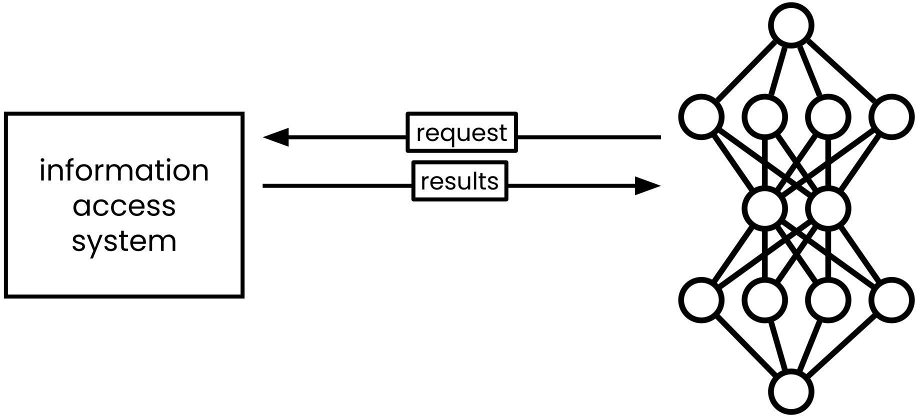


request: expression of information needed for the ML task



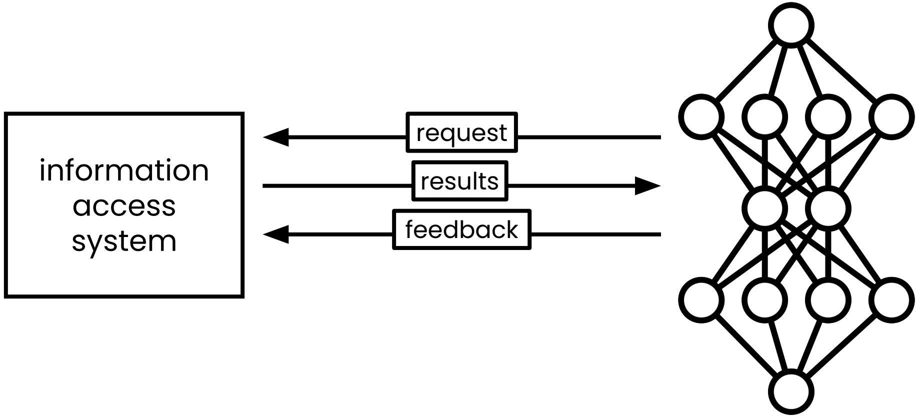
- request interface
  - keyword or NL
  - structured
  - multimedia
  - abstract representation
- request source
  - model input
  - hidden or intermediate representation
  - model output

results: information to help with the ML task



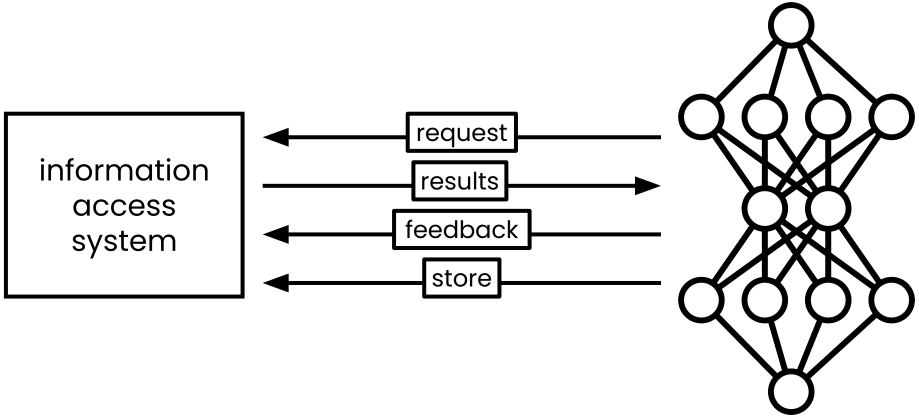
- result interface
  - item, ranking
  - text
  - structured
  - multimedia
  - abstract representation
- result destination
  - model input
  - hidden or intermediate representation
  - model output

<u>feedback</u>: information about the usefulness of the results



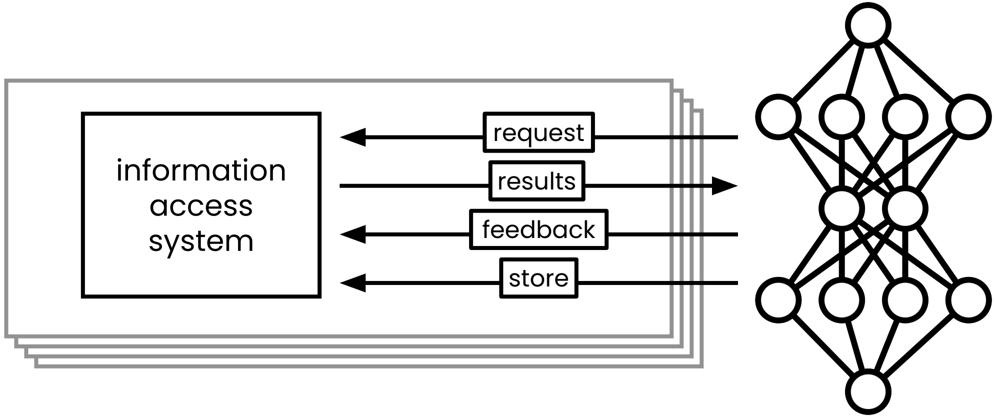
- feedback interface
  - scalar value
  - structured
- feedback source
  - intrinsic performance (e.g. auxiliary task)
  - extrinsic performance (e.g. core task)

store: derived information for future retrieval



- storage interface
    - text
    - structured
    - multimedia
    - abstract representation
- storage incentive
    - cache computation
    - contribute to corpus-level modeling
    - share with other models

<u>multiple requests</u>: retrieve results many times during inference

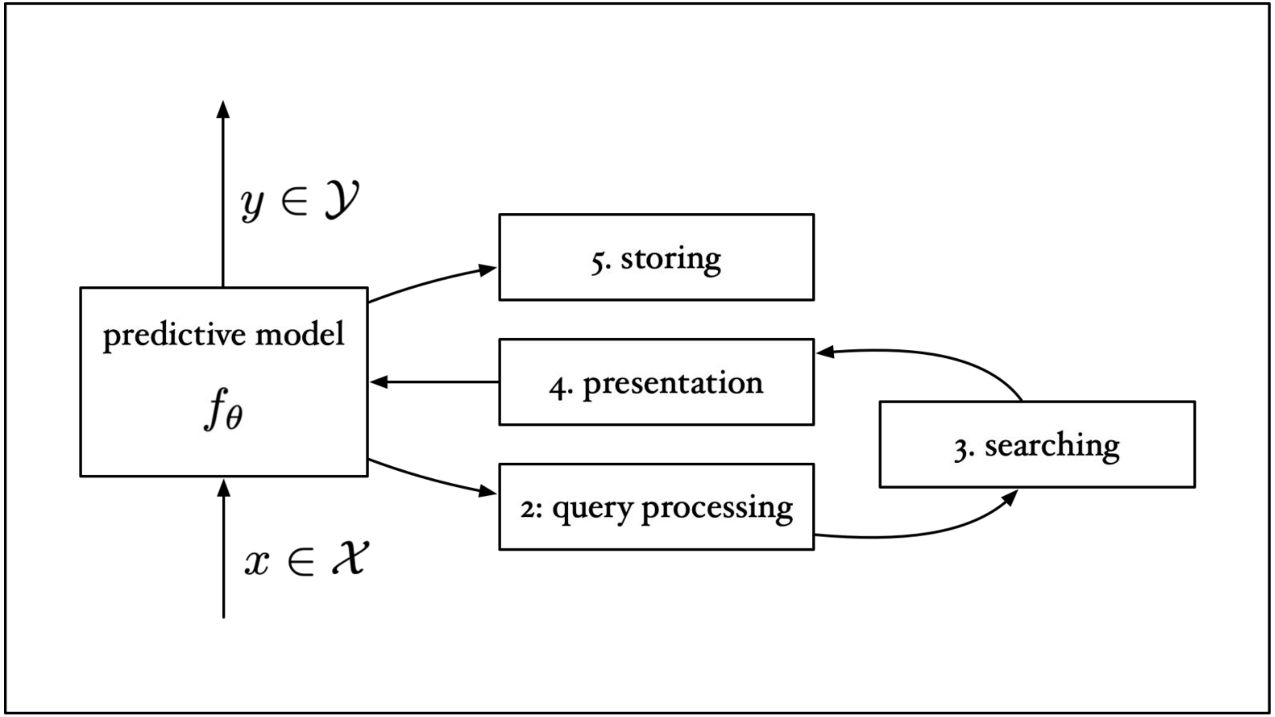


- multiple times during inference for a single instance
- allows multi-hop reasoning
- allows accessing *multiple* IA systems

Objectives of today's tutorial

1. survey and synthesize the variety of REML approaches based on common strategies
2. connect abstract themes to existing information retrieval research
3. outline a set of new open research problems for the information retrieval and ML community.

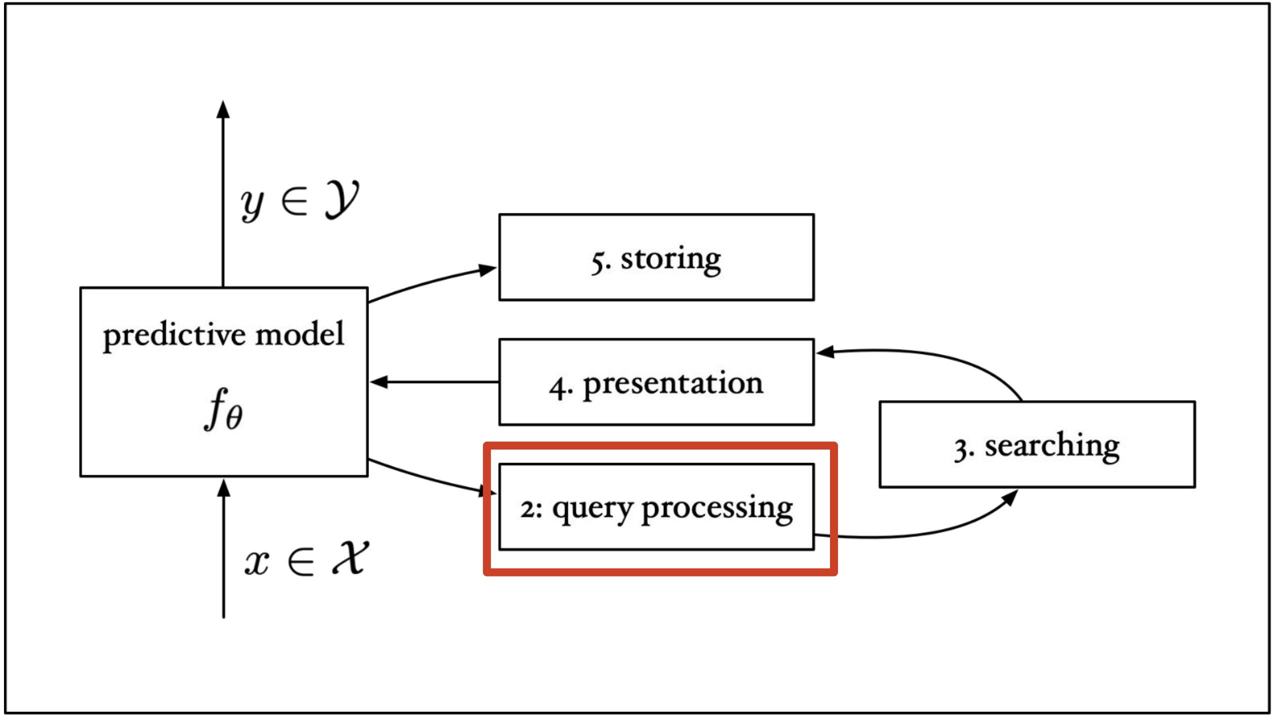
Overview


$y \in \mathcal{Y}$
5. storing
predictive model
f_θ
4. presentation
3. searching
2: query processing
$x \in \mathcal{X}$

questions?

Querying

Overview



1. introduction

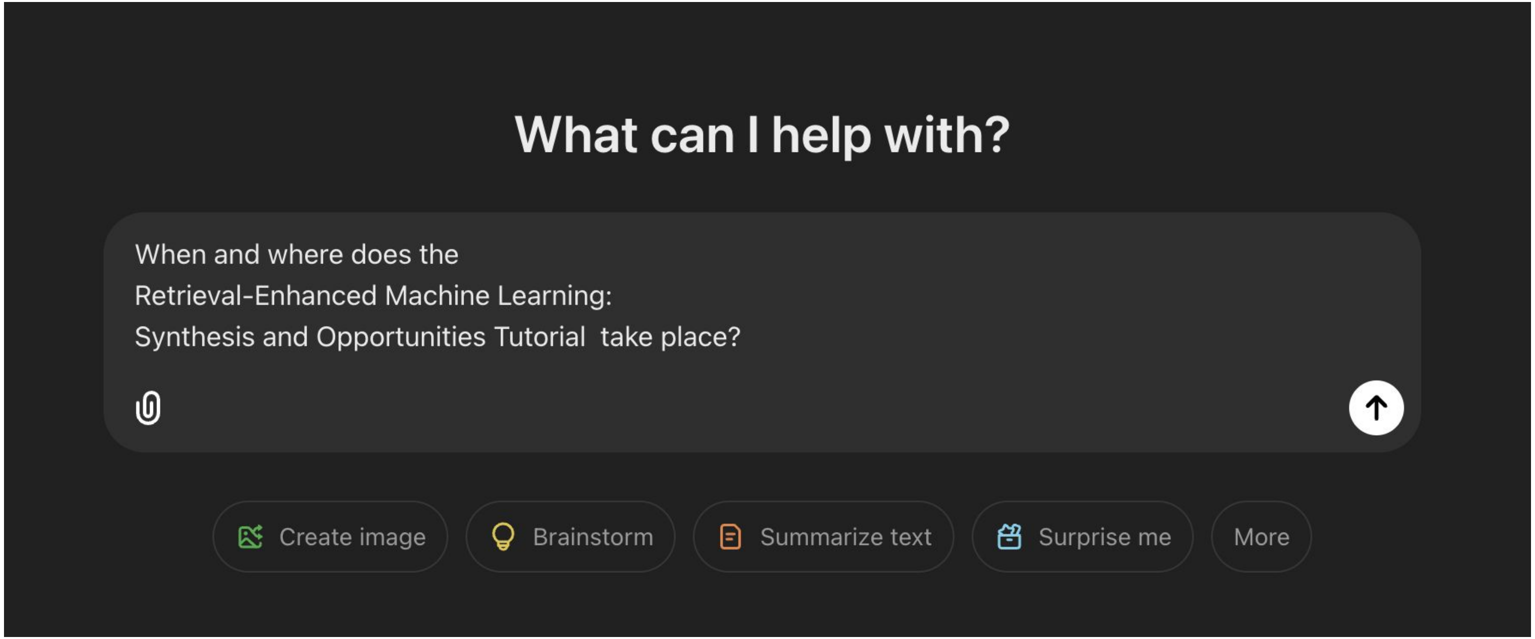
6. optimization

7. evaluation

8. future work

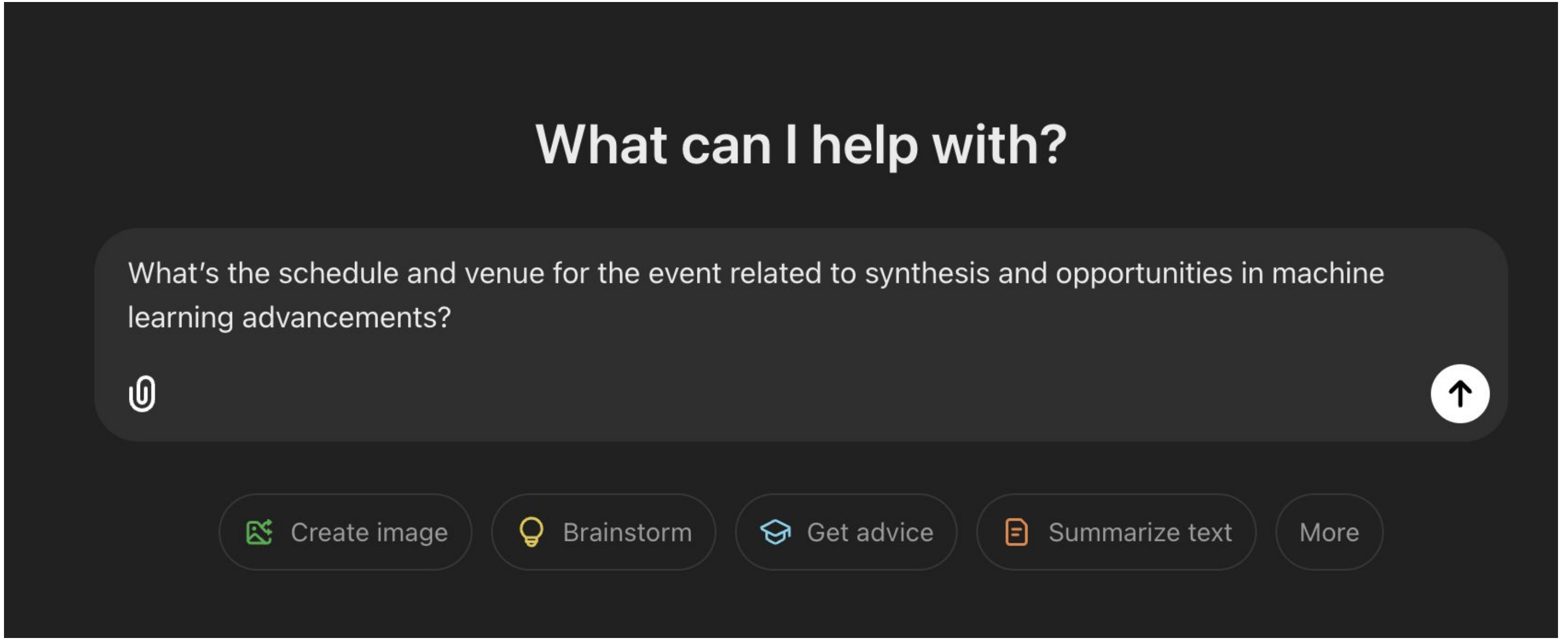
Motivation

Interaction with an REML system starts with the user querying the system for some kind of requests.



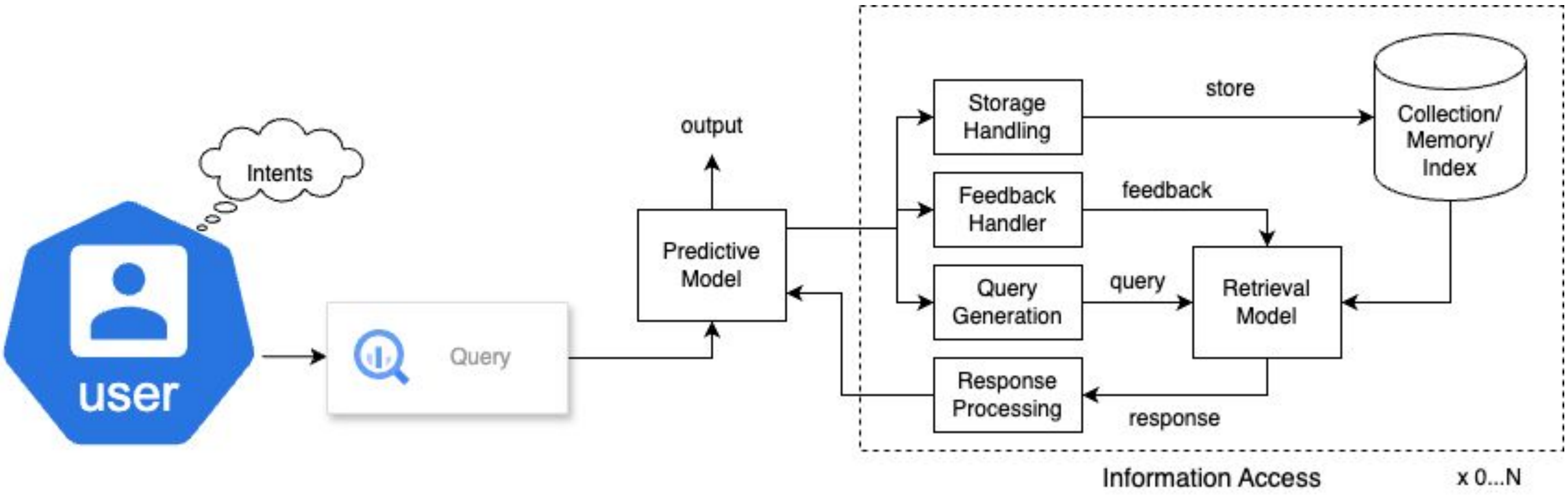
Motivation

- Why query processing is needed in REML?
  - Because of **ambiguity**, **complexity**, and **lack of context** in query!
  - Because the REML system might be able to perform its task with more **efficiency**, **scalability**, and **personalization**!



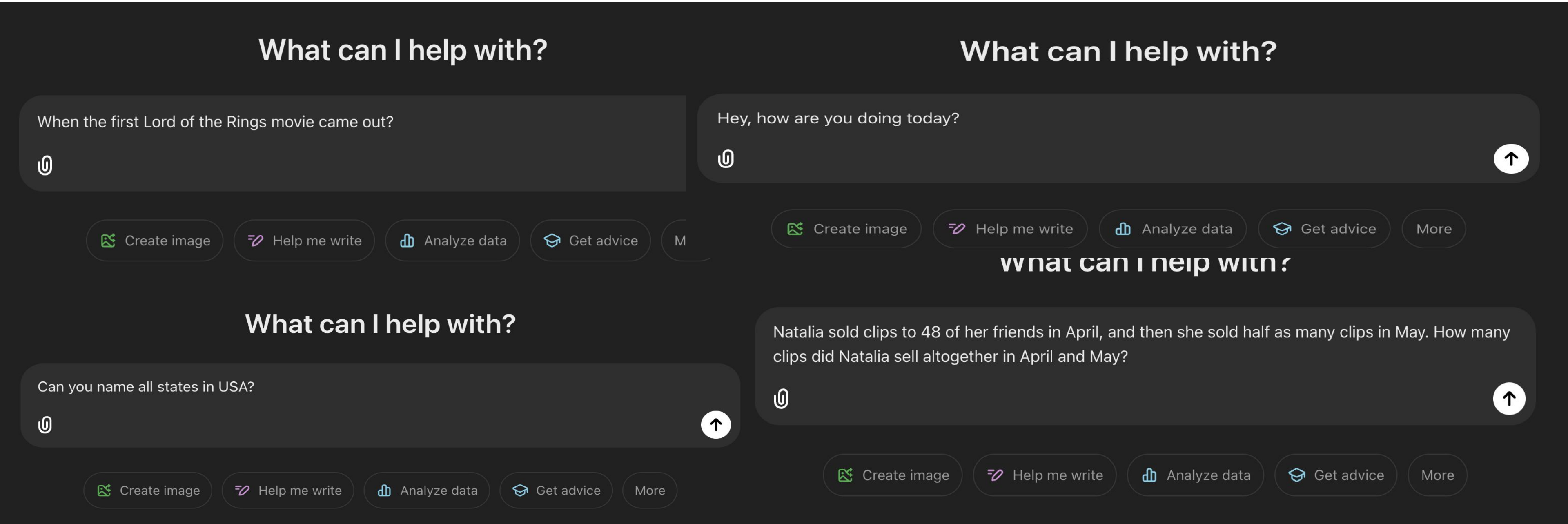
Motivation

- Query processing acts as a bridge between **user intent** and **REML system capabilities.**
  - **Intent** is hidden inside the query.
  - **REML system** may have different **capabilities** in responding to different **intents**.



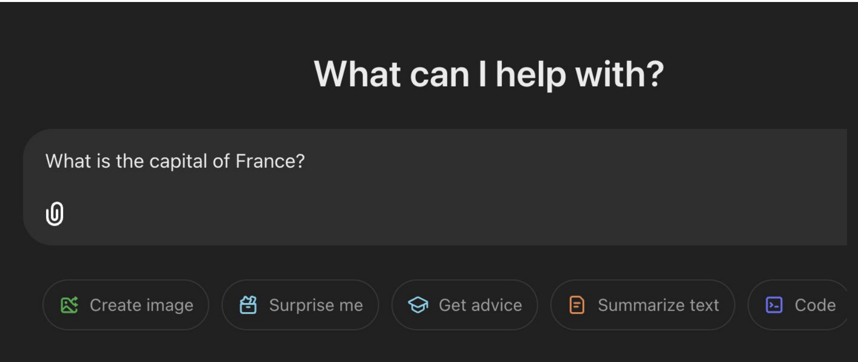
The Main Components of Query Processing

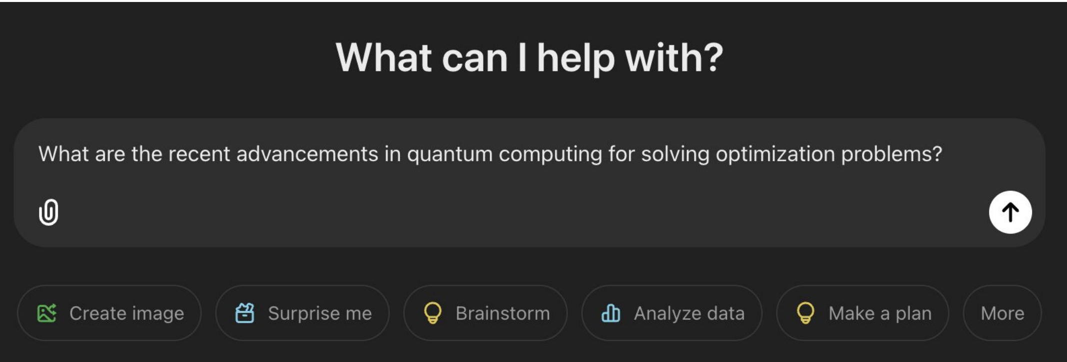
- The query processing in REML needs to answer three questions (first question):
  - **When** to query?
    - Does the question need external information to be answered?
    - Does the predictive model already have the knowledge to answer the query?



The Main Components of Query Processing

- The query processing in REML needs to answer three questions (second question):
    - **Where** to query?
        - We know external information is needed.
        - What kind of knowledge source can help answering the query?
            - General Knowledge Platforms: Wikipedia, Infoplease, etc.
            - Specialized Knowledge Platforms: PubMed, arXiv, etc.
            - News and Current Affairs: BBC news, New York Times, etc.
            - etc.
        - What retrieval approach should be used to answer the query?
            - Term matching: BM25, TF-IDF
            - Semantic search: DPR, ColBERT
            - etc.

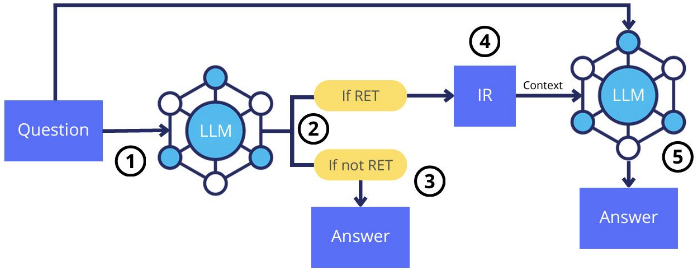


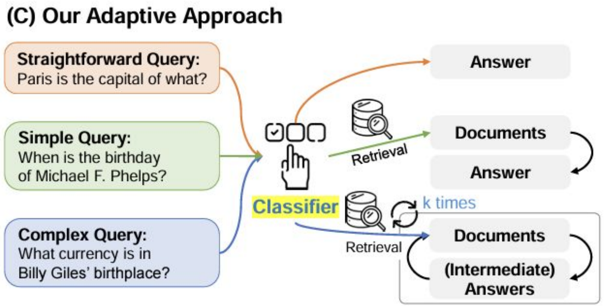


When to Query?

Selecting “when to query” can be modeled in different ways:

- Retrieve when the question is about unpopular entity [1, 2]
  - Wikipedia monthly views [1]
  - Wikipedia entity occurrence [2]
- Retrieve when the predictive model think it needs more context [3, 4]



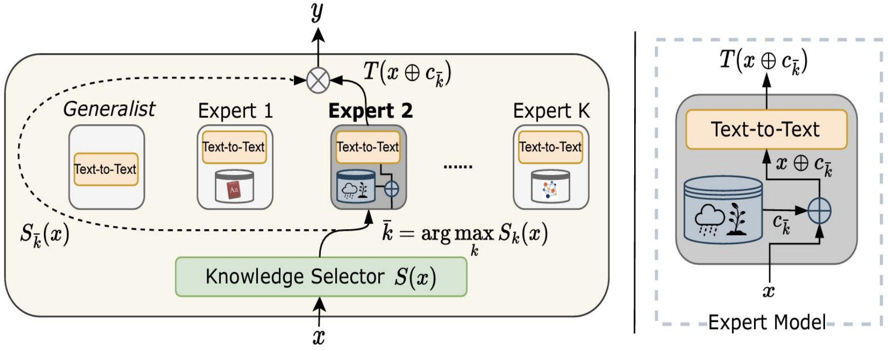


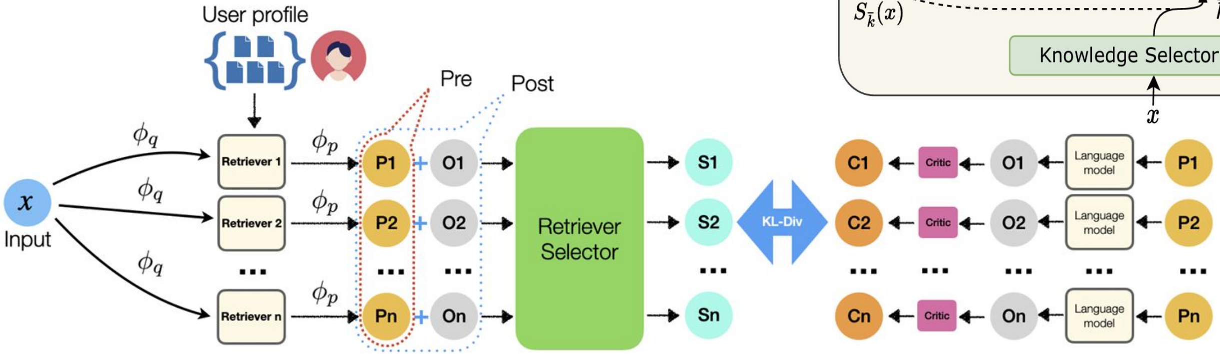
[1] Mallen, A., Asai, A., Zhong, V., Das, R., Khashabi, D., & Hajishirzi, H. (2023). When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 9802–9822). Association for Computational Linguistics.
[2] Maekawa, S., Iso, H., Gurajada, S., & Bhutani, N. (2024). Retrieval Helps or Hurts? A Deeper Dive into the Efficacy of Retrieval Augmentation to Language Models. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers) (pp. 5506–5521). Association for Computational Linguistics.
[3] Tiziano Labruna, Jon Ander Campos, & Gorka Azkune. (2024). When to Retrieve: Teaching LLMs to Utilize Information Retrieval Effectively.
[4] Jeong, S., Baek, J., Cho, S., Hwang, S., & Park, J. (2024). 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 (Volume 1: Long Papers) (pp. 7036–7050). Association for Computational Linguistics.

When & Where to Query?

Selecting "when" and "where" to query can be modeled at the same time:

- KIC: A Mixture of Semi-Parametric Experts [1]
- RSPG: Retriever Selection for Personalized Generation [2]





[1] Xiaoman Pan, Wenlin Yao, Hongming Zhang, Dian Yu, Dong Yu, & Jianshu Chen (2023). Knowledge-in-Context: Towards Knowledgeable Semi-Parametric Language Models. In The Eleventh International Conference on Learning Representations .
[2] Salemi, A., Kallumadi, S., & Zamani, H. (2024). Optimization Methods for Personalizing Large Language Models through Retrieval Augmentation. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 752–762). Association for Computing Machinery.

Where to Query?

Selecting "where to query" can be formulated as what retrieval model should be chosen:

- Zero-shot retriever selection [1]
  - In-domain Performance
    - Using retrieval model with highest in domain score
  - Query Similarity
    - Computing the similarity of the query with the training queries of the retrieval model
  - Query Alteration
    - First step: Retrieve documents using the query with each retrieval model
    - Second step: Alter the query by masking it randomly
    - Third step: Compute the similarity of retrieved documents to the altered query
    - Final step: select the retrieval model with the least standard deviation
- Large Language Model Assisted Retrieval Model Ranking (LARMOR) [2]
  - Query independent and offline
  - Step 1: Generating a set of pseudo queries for the domain
  - Step 2: Generating pseudo relevance labels for retrieved documents
  - Step 3: Score retrieval models based on pseudo queries and pseudo relevance labels
  - Choose the retrieval model based on the score

[1] Khramtsova, E., Zhuang, S., Baktashmotlagh, M., Wang, X., & Zuccon, G. (2023). Selecting which Dense Retriever to use for Zero-Shot Search. In Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region (pp. 223–233). Association for Computing Machinery.
[2] Khramtsova, E., Zhuang, S., Baktashmotlagh, M., & Zuccon, G. (2024). Leveraging LLMs for Unsupervised Dense Retriever Ranking. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 1307–1317). Association for Computing Machinery.

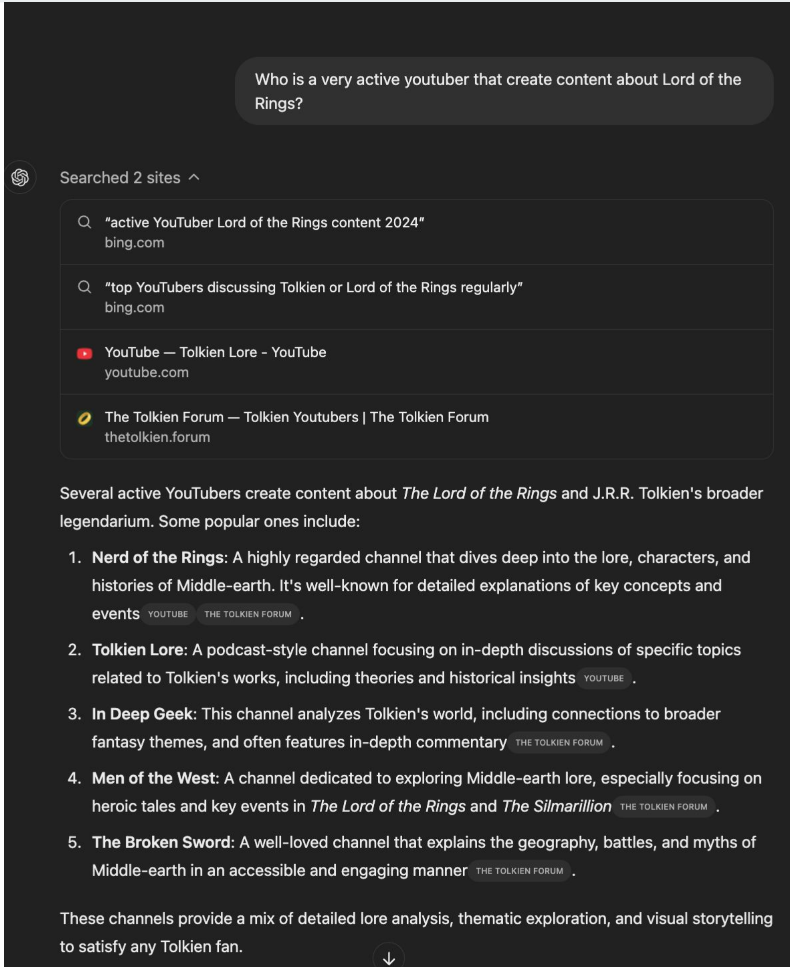
The Main Components of Query Processing

- The query processing in REML needs to answer three questions (third question):
  - **What** to query?
    - What information are we looking in the knowledge source?
    - What are the aspects that can help in answering the query?
    - How many knowledge pieces (documents) should be retrieved?
    - Should we consider all the retrieved information?
  - One simple approach is to use the user input (x) as the query:

$$q = I(x) = x$$

  - Sometimes the REML system needs to reformulate the input from the user to query the information access mechanism:

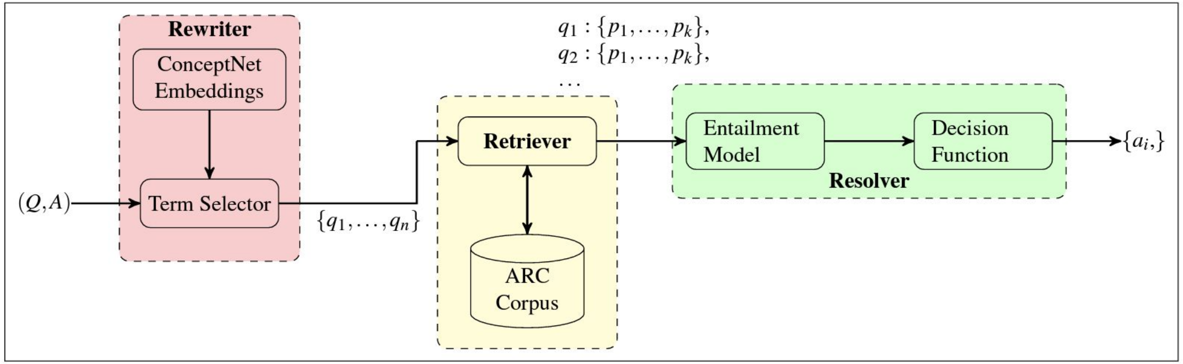
$$q = transform_q(x, context)$$



Different Input Transformation functions: Compression

Compression: not all words or components of the input are relevant for the search objective of the system, we can drop some of them.

- Sequence-to-sequence models for term selection [1, 2, 3, 4]

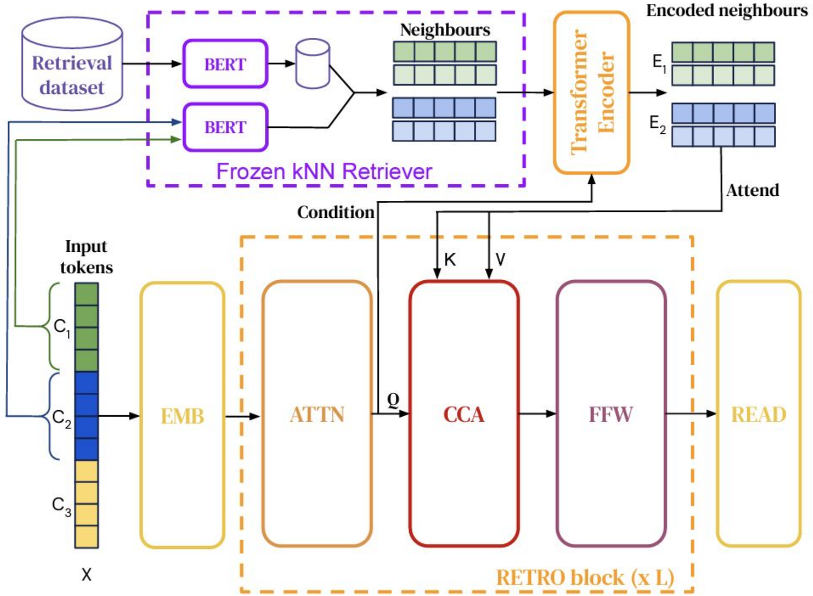


[1] Khashabi, D., Khot, T., Sabharwal, A., & Roth, D. (2017). Learning What is Essential in Questions. In Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017) (pp. 80–89). Association for Computational Linguistics.
[2] Ryan Musa, Xiaoyan Wang, Achille Fokoue, Nicholas Mattei, Maria Chang, Pavan Kapanipathi, Bassem Makni, Kartik Talamadupula, & Michael Witbrock (2019). Answering Science Exam Questions Using Query Reformulation with Background Knowledge. In Automated Knowledge Base Construction (AKBC).
[3] Ni, J., Zhu, C., Chen, W., & McAuley, J. (2019). Learning to Attend On Essential Terms: An Enhanced Retriever-Reader Model for Open-domain Question Answering. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (pp. 335–344). Association for Computational Linguistics.
[4] Yadegari, M., Kamalloo, E., & Rafiei, D. (2022). Detecting Frozen Phrases in Open-Domain Question Answering. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 1990–1996). Association for Computing Machinery.

Different Input Transformation functions: Compression

Compression: not all words or components of the input are relevant for the search objective of the system, we can drop some of them.

- Chunking the input as the query [1]
- Omitting modality in multi-modal tasks [2]

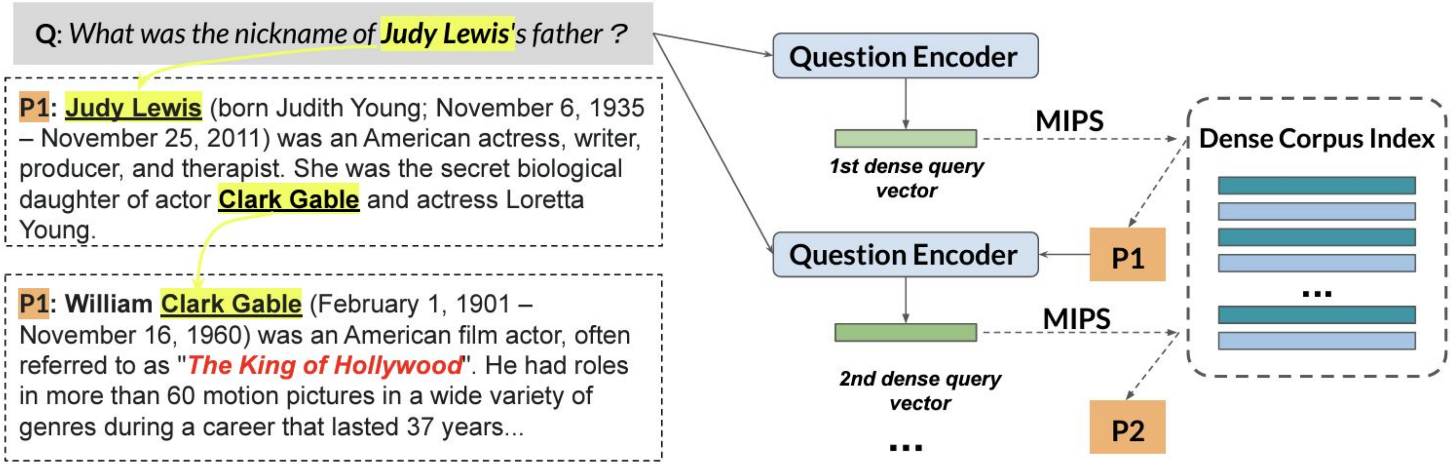


[1] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, & Laurent Sifre. (2022). Improving language models by retrieving from trillions of tokens.
[2] Gui, L., Wang, B., Huang, Q., Hauptmann, A., Bisk, Y., & Gao, J. (2022). KAT: A Knowledge Augmented Transformer for Vision-and-Language. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 956–968). Association for Computational Linguistics.

Different Input Transformation functions: Expansion

Expansion: the input alone may lack essential information required by the search system to yield desired results, we can expand them.

- Multi-hop expansion of query with retrieved results [1, 2]

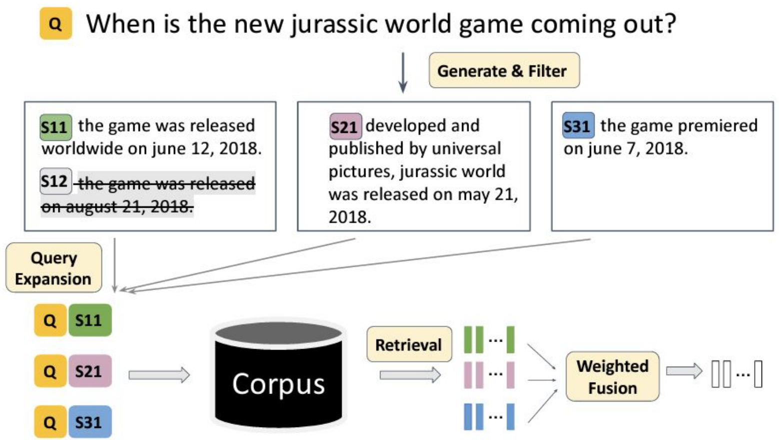


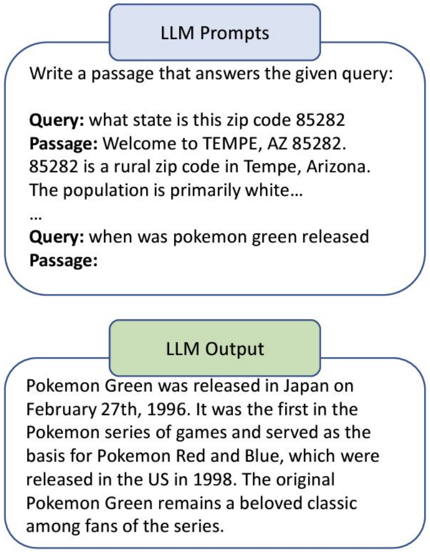
[1] Wenhan Xiong, Xiang Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, & Barlas Oguz (2021). Answering Complex Open-Domain Questions with Multi-Hop Dense Retrieval. In International Conference on Learning Representations.
[2] Zhu, Y., Pang, L., Lan, Y., Shen, H., & Cheng, X. (2021). Adaptive Information Seeking for Open-Domain Question Answering. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (pp. 3615–3626). Association for Computational Linguistics.

Different Input Transformation functions: Expansion

Expansion: the input alone may lack essential information, we can expand them.

- Generative expansion of the input [1, 2, 3, 4]



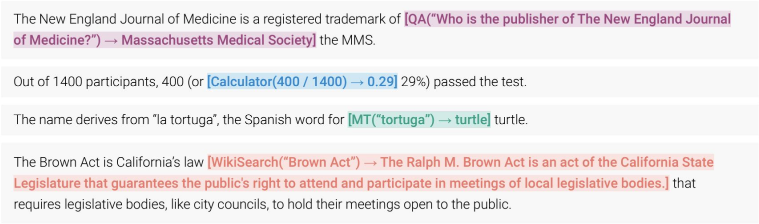


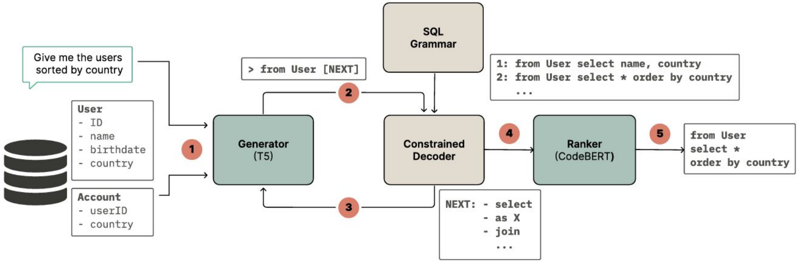
[1] Linqing Liu, Minghan Li, Jimmy Lin, Sebastian Riedel, & Pontus Stenetorp. (2022). Query Expansion Using Contextual Clue Sampling with Language Models.
[2] Chuang, Y.S., Fang, W., Li, S.W., Yih, W.t., & Glass, J. (2023). Expand, Rerank, and Retrieve: Query Reranking for Open-Domain Question Answering. In Findings of the Association for Computational Linguistics: ACL 2023 (pp. 12131–12147). Association for Computational Linguistics.
[3] Mao, Y., He, P., Liu, X., Shen, Y., Gao, J., Han, J., & Chen, W. (2021). Generation-Augmented Retrieval for Open-Domain Question Answering. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (pp. 4089–4100). Association for Computational Linguistics.
[4] Wang, L., Yang, N., & Wei, F. (2023). Query2doc: Query Expansion with Large Language Models. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (pp. 9414–9423). Association for Computational Linguistics.

Different Input Transformation functions: Conversion

Conversion: reshaping the input into a new query based on its inherent structure, instead of mere expansion.

- Raw user input to structured query e.g., API or Database access
  - Structured query generation with supervised training [1, 2, 4, 5]
  - Structured query generation with in-context learning [3]
- During inference query generation [6]



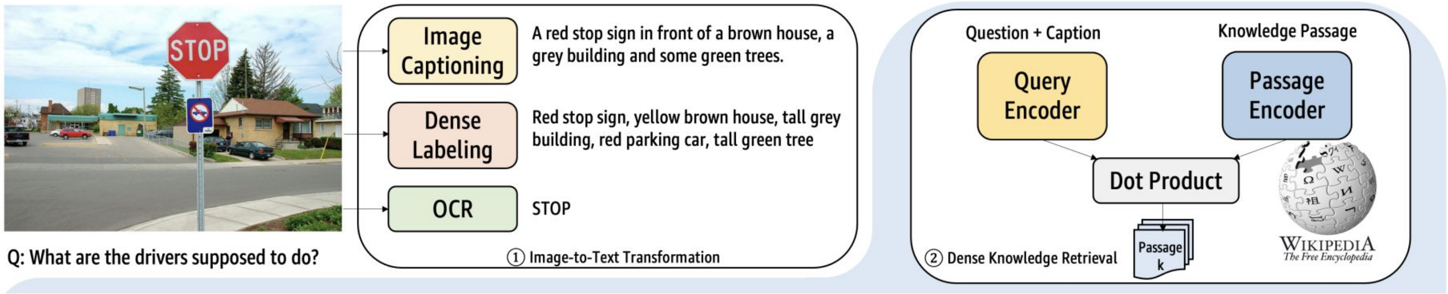


[1] Arcadinho, S., Aparicio, D., Veiga, H., & Alegria, A. (2022). T5QL: Taming language models for SQL generation. In Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM) (pp. 276–286). Association for Computational Linguistics.
[2] Dou, L., Gao, Y., Pan, M. et al. UniSAr: a unified structure-aware autoregressive language model for text-to-SQL semantic parsing. Int. J. Mach. Learn. & Cyber. 14, 4361–4376 (2023). https://doi.org/10.1007/s13042-023-01898-3
[3] Qiao Jin, Yifan Yang, Qingyu Chen, Zhiyong Lu, GeneGPT: augmenting large language models with domain tools for improved access to biomedical information, Bioinformatics, Volume 40, Issue 2, February 2024, btae075, https://doi.org/10.1093/bioinformatics/btae075
[4] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, & Maosong Sun (2024). ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. In The Twelfth International Conference on Learning Representations.
[5] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, & Thomas Scialom (2023). Toolformer: Language Models Can Teach Themselves to Use Tools. In Thirty-seventh Conference on Neural Information Processing Systems.
[6] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, & Hannaneh Hajishirzi (2024). Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In The Twelfth International Conference on Learning Representations.

Different Input Transformation functions: Conversion

Conversion: reshaping the input into a new query based on its inherent structure, instead of mere expansion.

- Query space conversion
  - Converting modality [1, 2, 3]
    - OCR [1], dense labeling [1], caption generation [1, 2, 3], entity extraction [4]

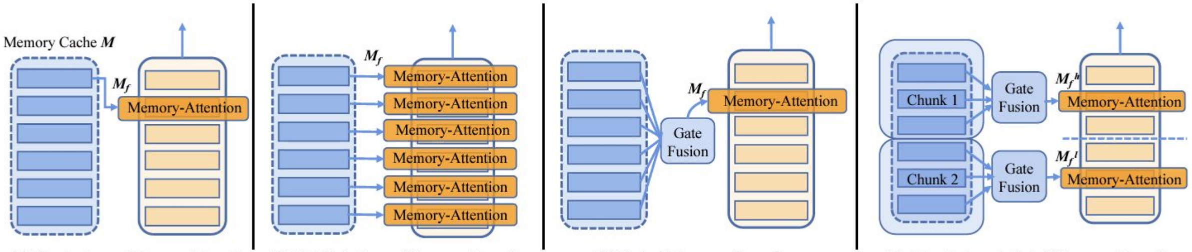


[1] Gao, F., Ping, Q., Thattai, G., Reganti, A., Wu, Y., & Natarajan, P. (2022). Transform-Retrieve-Generate: Natural Language-Centric Outside-Knowledge Visual Question Answering. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 5057-5067).
[2] Salemi, A., Altmayer Pizzorno, J., & Zamani, H. (2023). A Symmetric Dual Encoding Dense Retrieval Framework for Knowledge-Intensive Visual Question Answering. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 110–120). Association for Computing Machinery.
[3] Lin, W., & Byrne, B. (2022). Retrieval Augmented Visual Question Answering with Outside Knowledge. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (pp. 11238–11254). Association for Computational Linguistics.
[4] Wu, J., & Mooney, R. (2022). Entity-Focused Dense Passage Retrieval for Outside-Knowledge Visual Question Answering. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (pp. 8061–8072). Association for Computational Linguistics.

Different Input Transformation functions: Conversion

Conversion: reshaping the input into a new query based on its inherent structure, instead of mere expansion.

- Query space conversion
  - Text to latent space query
    - KNN-LM [1]
    - Neural Turing Machines [2, 3]
    - Memory Transformer [4, 5]



(a) Single-Layer Memory Transfer (b) Multiple-Layer Memory Transfer (c) Gated Memory Transfer (d) Chunk-based Gated Memory Transfer

[1] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, & Mike Lewis (2020). Generalization through Memorization: Nearest Neighbor Language Models. In International Conference on Learning Representations.
[2] Alex Graves, Greg Wayne, & Ivo Danihelka. (2014). Neural Turing Machines.
[3] Caglar Gulcehre, Sarath Chandar, & Yoshua Bengio. (2017). Memory Augmented Neural Networks with Wormhole Connections.
[4] Wan, Z., Yin, Y., Zhang, W., Shi, J., Shang, L., Chen, G., Jiang, X., & Liu, Q. (2022). G-MAP: General Memory-Augmented Pre-trained Language Model for Domain Tasks. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (pp. 6585–6597). Association for Computational Linguistics.
[5] Wu, Q., Lan, Z., Qian, K., Gu, J., Geramifard, A., & Yu, Z. (2022). Memformer: A Memory-Augmented Transformer for Sequence Modeling. In Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022 (pp. 308–318). Association for Computational Linguistics.

Different Input Transformation functions: Decomposition

Decomposition: breaking down a complex input into simpler parts, often to better understand the content and retrieve more accurate results

- Learning to decompose [2, 3]
  - unsupervised data generation and training decomposition model
- Decomposition as a span prediction problem [1]

Type	**Bridging (47%)** requires finding the first-hop evidence in order to find another, second-hop evidence.
Q	Which team does the **player** named 2015 Diamond Head Classics MVP play for?
Q1	Which player named 2015 Diamond Head Classics MVP?
Q2	Which team does ANS play for?
Type	**Intersection (23%)** requires finding an entity that satisfies two independent conditions.
Q	Stories USA starred ✓ which actor and comedian ✓ from 'The Office'?
Q1	Stories USA starred which actor and comedian?
Q2	Which actor and comedian from 'The Office'?
Type	**Comparison (22%)** requires comparing the property of two different entities.
Q	Who was born earlier, Emma Bull or Virginia Woolf?
Q1	Emma Bull was born when?
Q2	Virginia Woolf was born when?
Q3	Which is smaller (Emma Bull, ANS) (Virgina Woolf, ANS)

[1] Min, S., Zhong, V., Zettlemoyer, L., & Hajishirzi, H. (2019). Multi-hop Reading Comprehension through Question Decomposition and Rescoring. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 6097–6109). Association for Computational Linguistics.
[2] Perez, E., Lewis, P., Yih, W.t., Cho, K., & Kiela, D. (2020). Unsupervised Question Decomposition for Question Answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 8864–8880). Association for Computational Linguistics.
[3] Zhou, B., Richardson, K., Yu, X., & Roth, D. (2022). Learning to Decompose: Hypothetical Question Decomposition Based on Comparable Texts. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (pp. 2223–2235). Association for Computational Linguistics.

Conclusion: Unified Equation for Query Generation

Considering all transformations, we the following general query generation equation:

$$Q = decompose(transform_q(x, context), context)$$

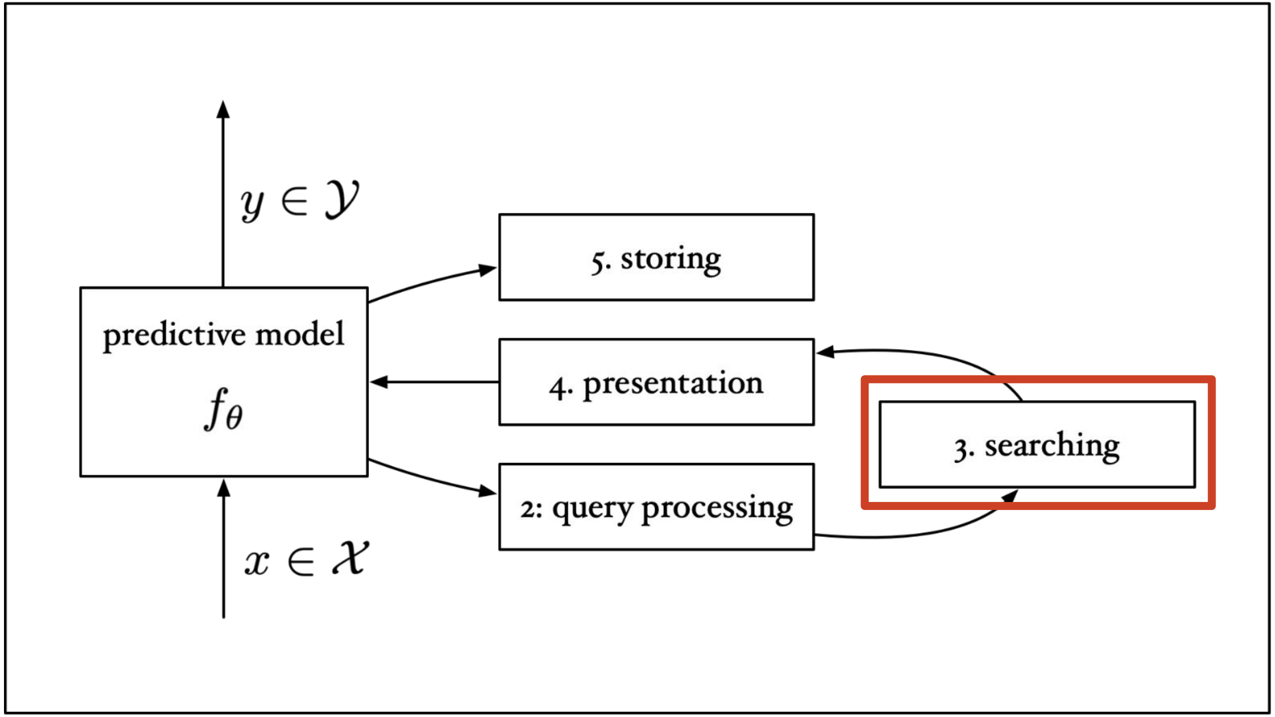
This can be used multiple times in different orders and different combinations to cover all possible query generation cases, such as adaptive retrieval, multi-hop retrieval, etc.

Future Directions:

- Query with instruction and context
  - Requires retrieval models that are capable of instruction following
- Retriever aware query generation
  - Adapting query with retrieval model capabilities

Searching

Overview



1. introduction

6. optimization

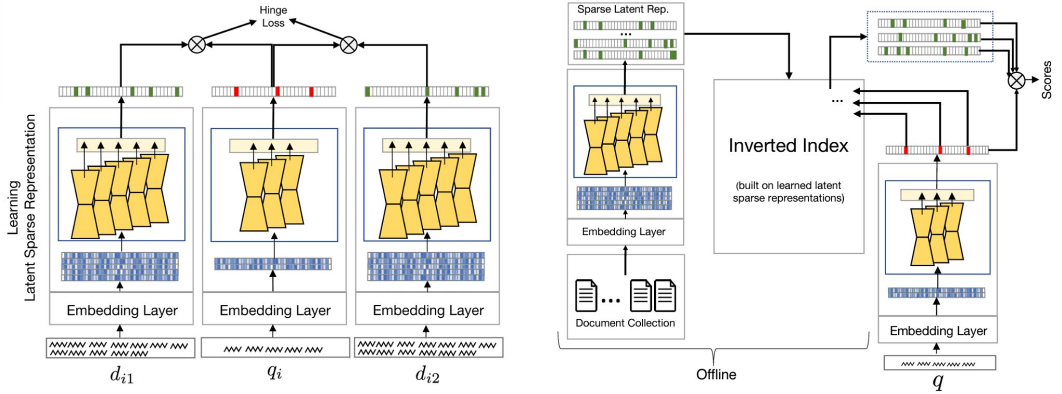
7. evaluation

8. future work

Retrieval with Sparse Representations

In sparse retrieval, the query and documents are converted to a v-dimensional sparse vectors that contain a lot of zero elements.

- Term matching sparse retrieval:
  - TF-IDF [1]
  - BM25 [2]
  - Query Likelihood [3]
- Neural-based sparse retrieval:
  - SPLADE [4]
  - SNRM [5]
- Benefits:
  - Efficient retrieval with inverted index
  - Strong term filtering ability



(a) Training time

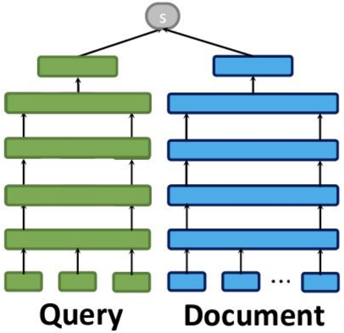
(b) Inference time

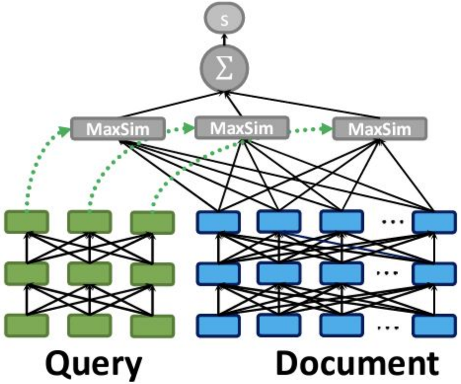
[1] Gerard Salton, & Christopher Buckley (1988). Term-weighting approaches in automatic text retrieval. Information Processing & Management, 24(5), 513-523.
[2] Robertson, S., Walker, S., Jones, S., Hancock-Beaulieu, M., & Gatford, M. (1995). Okapi at TREC-3. In Overview of the Third Text REtrieval Conference (TREC-3) (pp. 109-126). Gaithersburg, MD: NIST.
[3] Ponte, J., & Croft, W. (1998). A language modeling approach to information retrieval. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 275–281). Association for Computing Machinery.
[4] Formal, T., Piwowarski, B., & Clinchant, S. (2021). SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 2288–2292). Association for Computing Machinery.
[5] Zamani, H., Dehghani, M., Croft, W., Learned-Miller, E., & Kamps, J. (2018). From Neural Re-Ranking to Neural Ranking: Learning a Sparse Representation for Inverted Indexing. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (pp. 497–506). Association for Computing Machinery.

Retrieval with Dense Representations

In dense retrieval, the query and documents are converted to a d-dimensional dense vectors and a scoring function is applied over the vectors.

- Single vector retrieval
  - DPR [1] for text retrieval
  - CLIP [2] and DEDR [3] for multi-modal retrieval
- Multi-vector retrieval
  - ColBERT [4]
- Efficient retrieval can be challenging on a large corpus
  - HNSW [5]



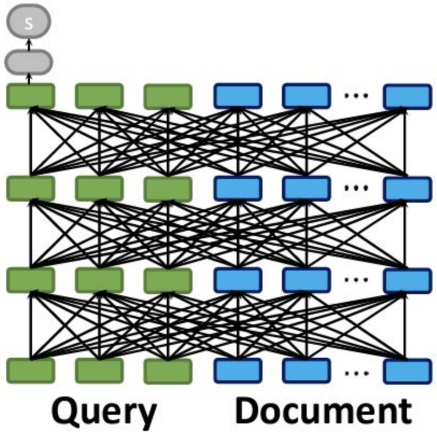


[1] Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.t. (2020). Dense Passage Retrieval for Open-Domain Question Answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 6769–6781). Association for Computational Linguistics.
[2] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, & Ilya Sutskever. (2021). Learning Transferable Visual Models From Natural Language Supervision.
[3] Salemi, A., Altmayer Pizzorno, J., & Zamani, H. (2023). A Symmetric Dual Encoding Dense Retrieval Framework for Knowledge-Intensive Visual Question Answering. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 110–120). Association for Computing Machinery.
[4] Khattab, O., & Zaharia, M. (2020). ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 39–48). Association for Computing Machinery.
[5] Malkov, Y., & Yashunin, D. (2020). Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. IEEE Trans. Pattern Anal. Mach. Intell., 42(4), 824–836.

Reranking

Modern search engines are mainly designed based on a multi-stage cascaded architecture–a stack of ranking models where the first model efficiently retrieves a list of documents and the following models rerank the results from the previous stage.

- First stage retrieves a large set of documents
  - Cheaper and faster than second stage, e.g., BM25
  - Doesn't need to be a strong retrieval model
- Second stage
  - A strong reranking model, such as BERT trained for reranking [1, 2, 3[
  - An LLM designed for reranking [4, 5]
- Challenges
  - trade off between efficiency and effectiveness
  - Lower performance as as size of the first stage grows [6]



[1] Rodrigo Nogueira, & Kyunghyun Cho. (2020). Passage Re-ranking with BERT.
[2] Alireza Salemi, & Hamed Zamani. (2024). Learning to Rank for Multiple Retrieval-Augmented Models through Iterative Utility Maximization.
[3] Salemi, A., & Zamani, H. (2024). Towards a Search Engine for Machines: Unified Ranking for Multiple Retrieval-Augmented Large Language Models. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 741–751). Association for Computing Machinery.
[4] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, & Zhaochun Ren. (2023). Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agents.
[5] Xinyu Zhang, Sebastian Hofstätter, Patrick Lewis, Raphael Tang, & Jimmy Lin. (2023). Rank-without-GPT: Building GPT-Independent Listwise Rerankers on Open-Source Large Language Models.
[6] Mathew Jacob, Erik Lindgren, Matei Zaharia, Michael Carbin, Omar Khattab, & Andrew Drozdov. (2024). Drowning in Documents: Consequences of Scaling Reranker Inference.

Generative Retrieval

A new paradigm where a model generates relevant documents or passages ids directly in response to a query, rather than selecting them from a pre-indexed corpus.

- Generative models
  - DSI [1]
  - RIPOR [2]
  - SEAL [3]
- Challenges
  - Scalability
  - Out-of-domain performance
  - Cost of search

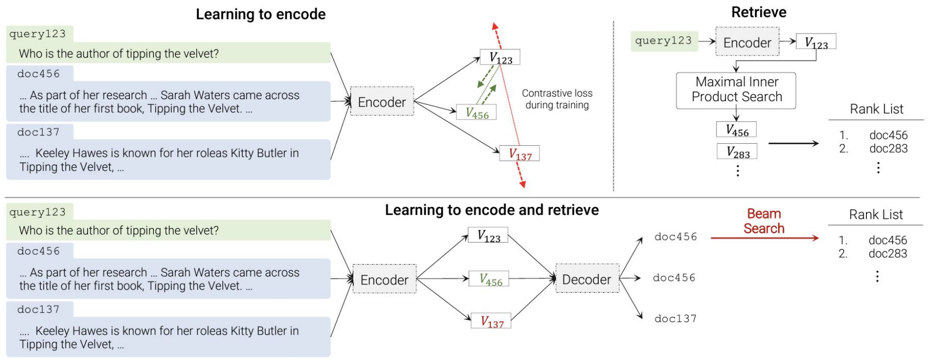


Figure 1: Comparison of dual encoders (top) to differentiable search index (bottom).

[1] Yi Tay, Vinh Q. Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, Tal Schuster, William W. Cohen, & Donald Metzler (2022). Transformer Memory as a Differentiable Search Index. In Advances in Neural Information Processing Systems.
[2] Zeng, H., Luo, C., Jin, B., Sarwar, S., Wei, T., & Zamani, H. (2024). Scalable and Effective Generative Information Retrieval. In Proceedings of the ACM Web Conference 2024 (pp. 1441–1452). Association for Computing Machinery.
[3] Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, & Fabio Petroni (2022). Autoregressive Search Engines: Generating Substrings as Document Identifiers. In Advances in Neural Information Processing Systems.

Conclusion: Unified Equation for Searching

We can define two type of addressing:

- Content-based addressing
- Location-based addressing

$$w_t^{content} = address_{content}(q_t, C_t) = topK(sort(score(q_t, transform_s(C_t))), k)$$

$$w_t^{location} = address_{location}(q_t, context)$$

$$w_t = combine(w_t^{location}, w_t^{content})$$

When we get the address, then it is time for reading:

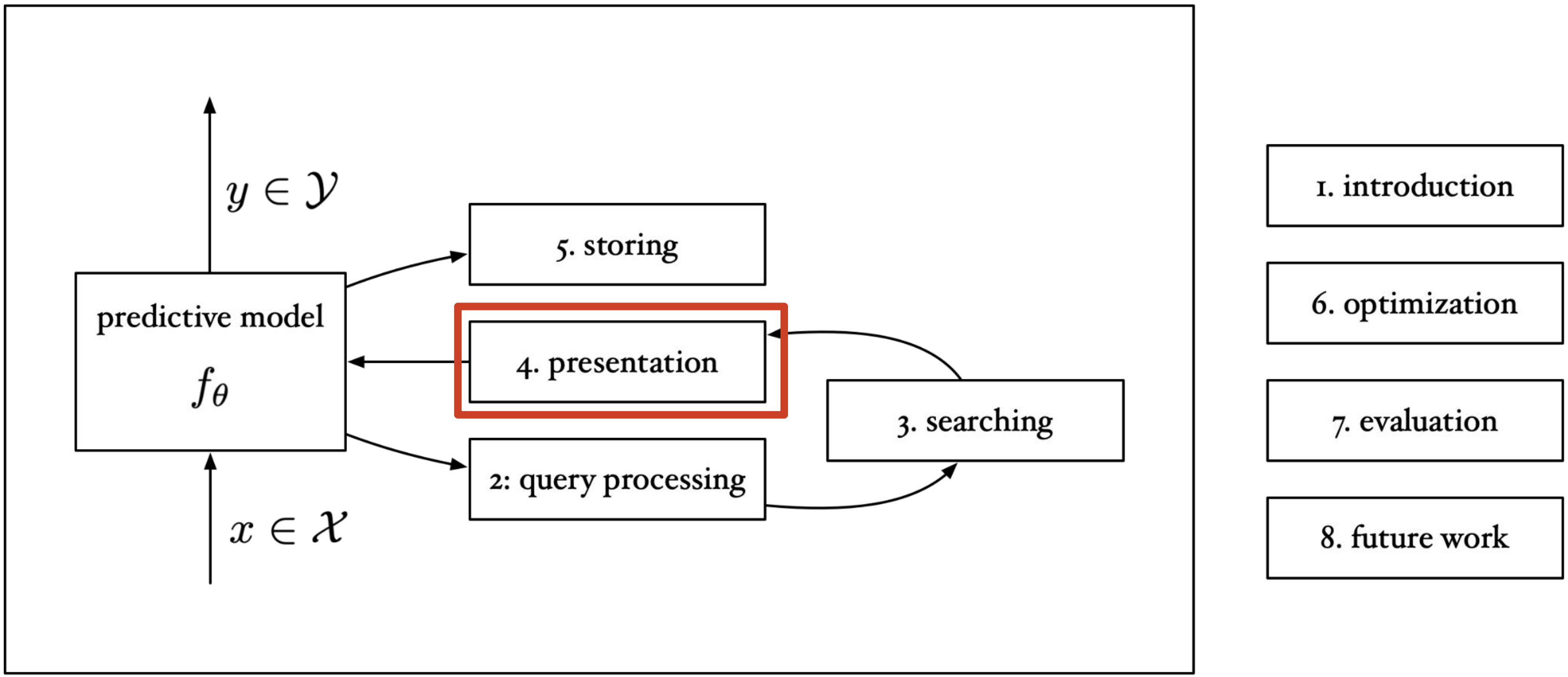
$$r_t = read(w_t, transform_s(C_t)),$$

Future Directions:

- Predictive Model-Aware Retrieval Systems
- Redefining Relevance

Presentation & Consumption

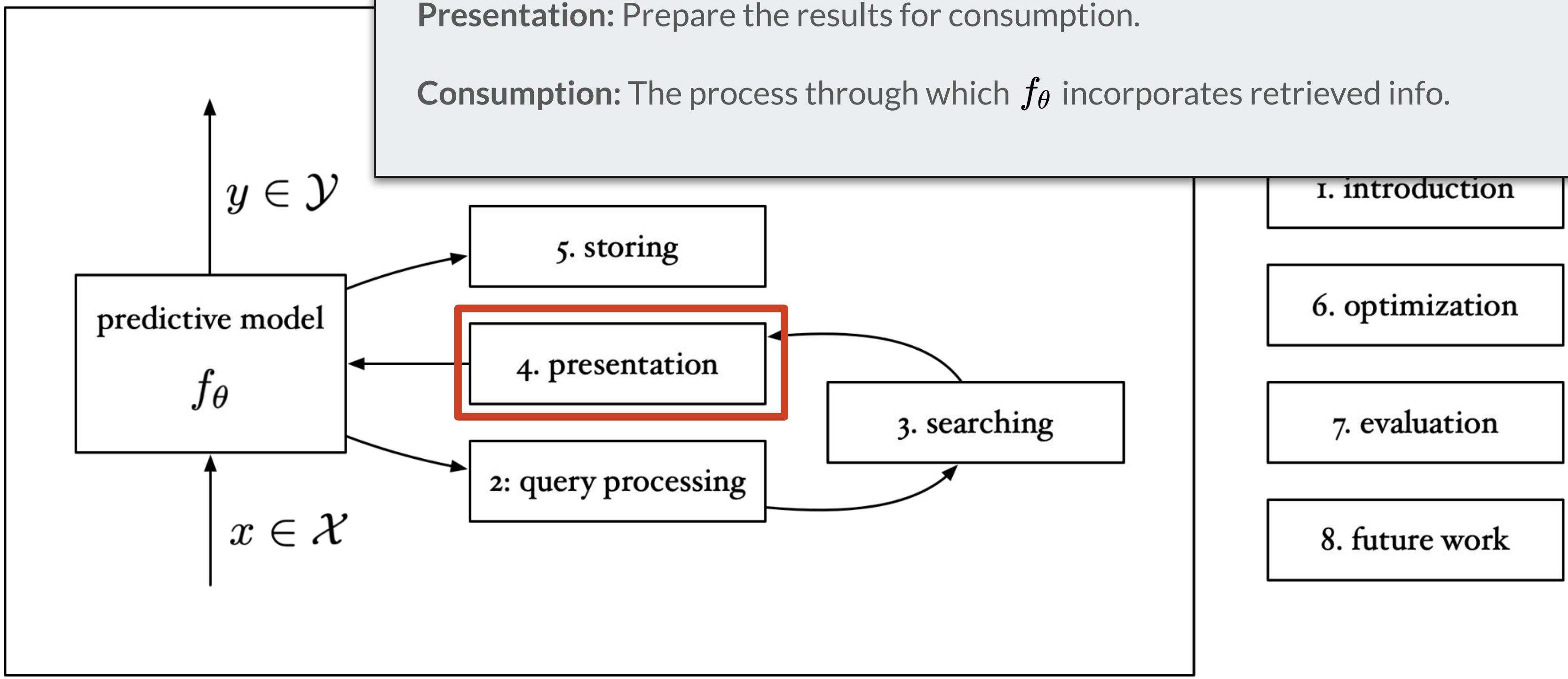
Overview


$y \in \mathcal{Y}$
5. storing
predictive model
f_θ
4. presentation
3. searching
2: query processing
$x \in \mathcal{X}$
1. introduction
6. optimization
7. evaluation
8. future work

Overview

Presentation: Prepare the results for consumption.

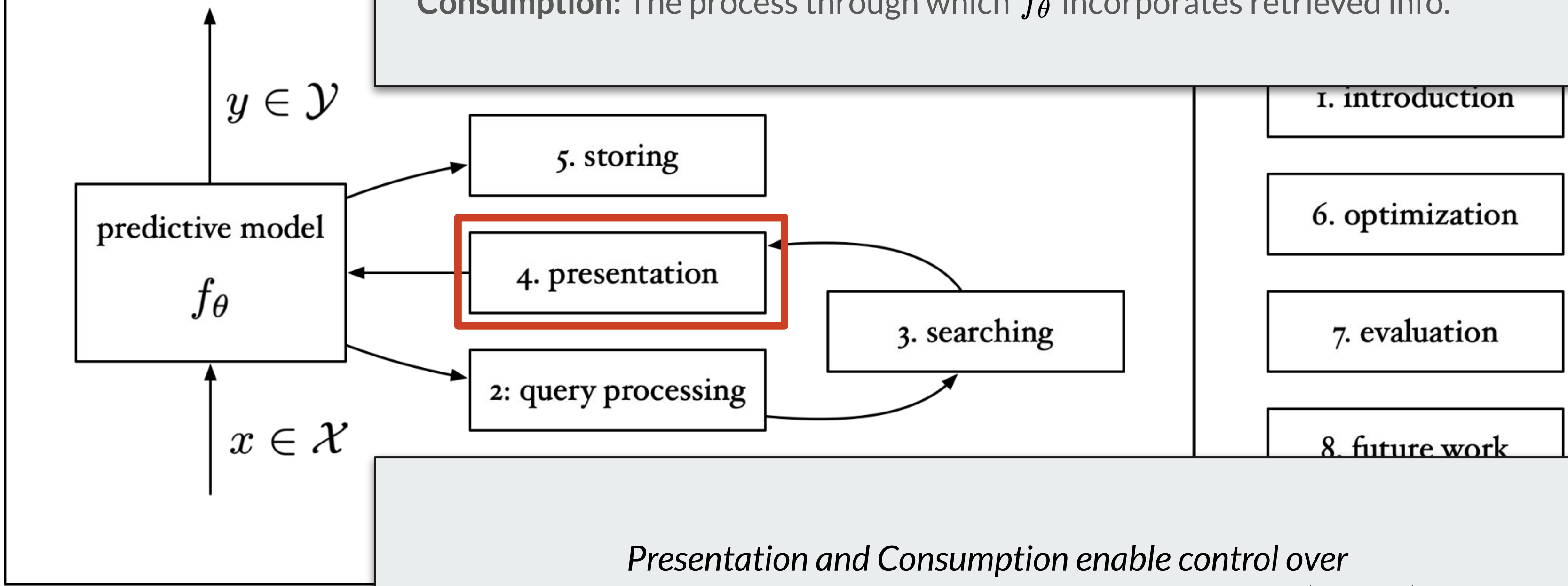
Consumption: The process through which f_θ incorporates retrieved info.



Overview

Presentation: Prepare the results for consumption.

Consumption: The process through which f_θ incorporates retrieved info.



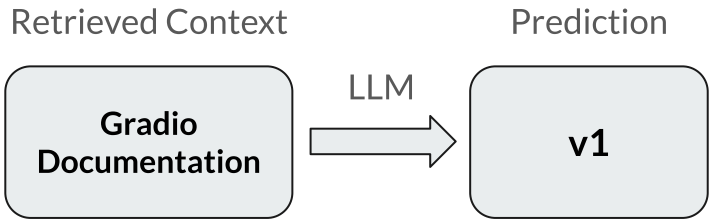
Presentation and Consumption enable control over cost-quality tradeoffs in REML, keeping other components (mostly) fixed.

Cost-Quality Tradeoffs in REML

RAG Query: Write some code that implements a ChatBot

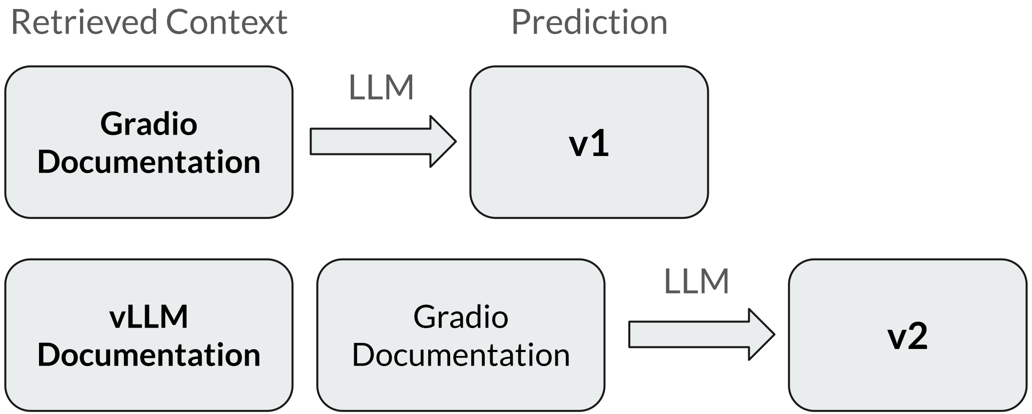
Cost-Quality Tradeoffs in REML

RAG Query: Write some code that implements a ChatBot



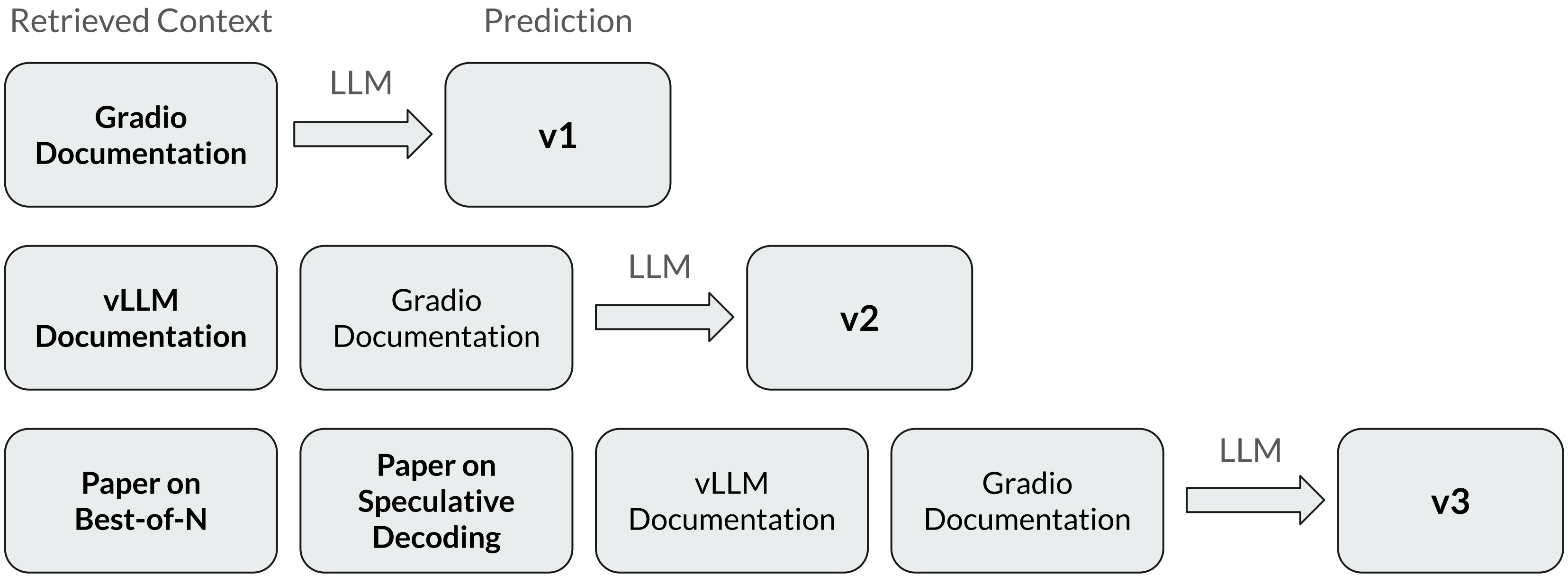
Cost-Quality Tradeoffs in REML

RAG Query: Write some code that implements a ChatBot



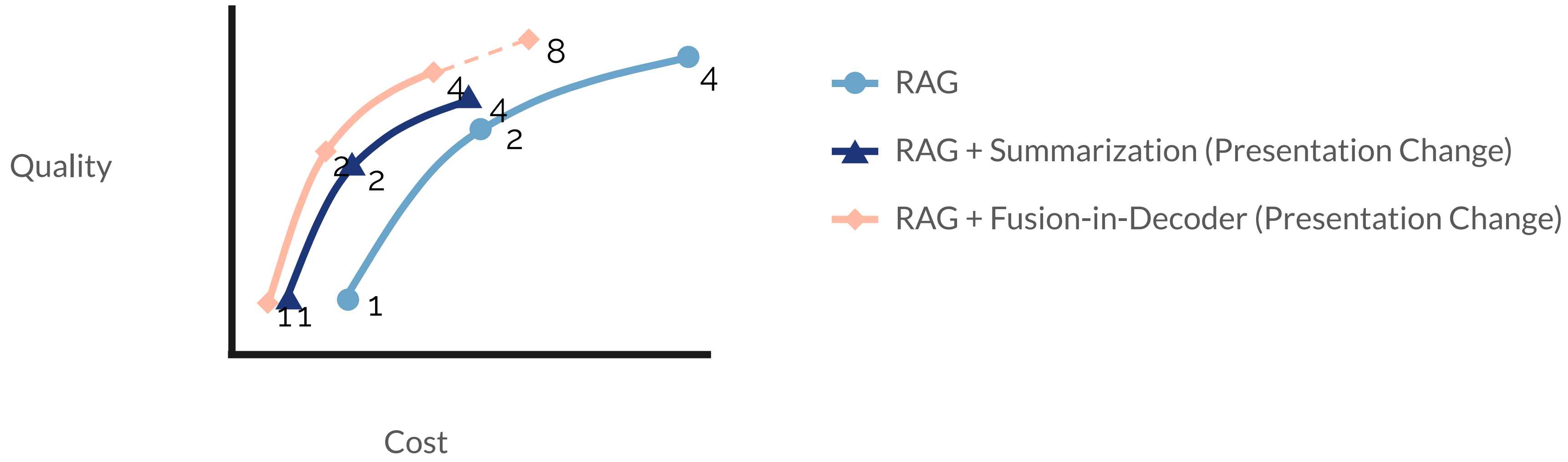
Cost-Quality Tradeoffs in REML

RAG Query: Write some code that implements a ChatBot

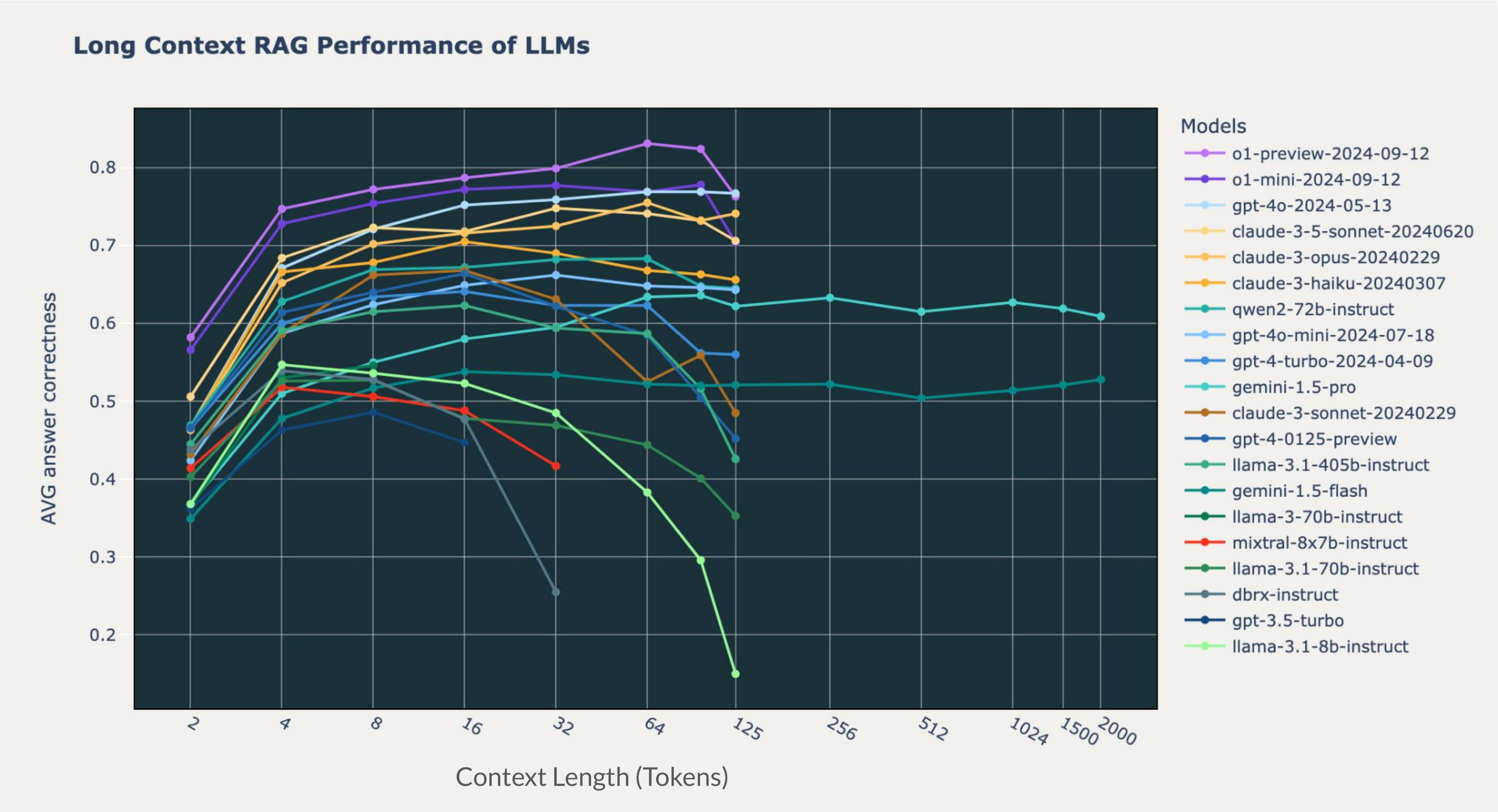


Cost-Quality Tradeoffs in REML

ChatBot Quality using 1, 2, 4 Retrieved Documents (Hypothetical)



Cost-Quality Tradeoffs in REML



https://arxiv.org/abs/2411.03538v1

Cost-Quality Tradeoffs in REML

Inference Scaling for Long-Context Retrieval Augmented Generation

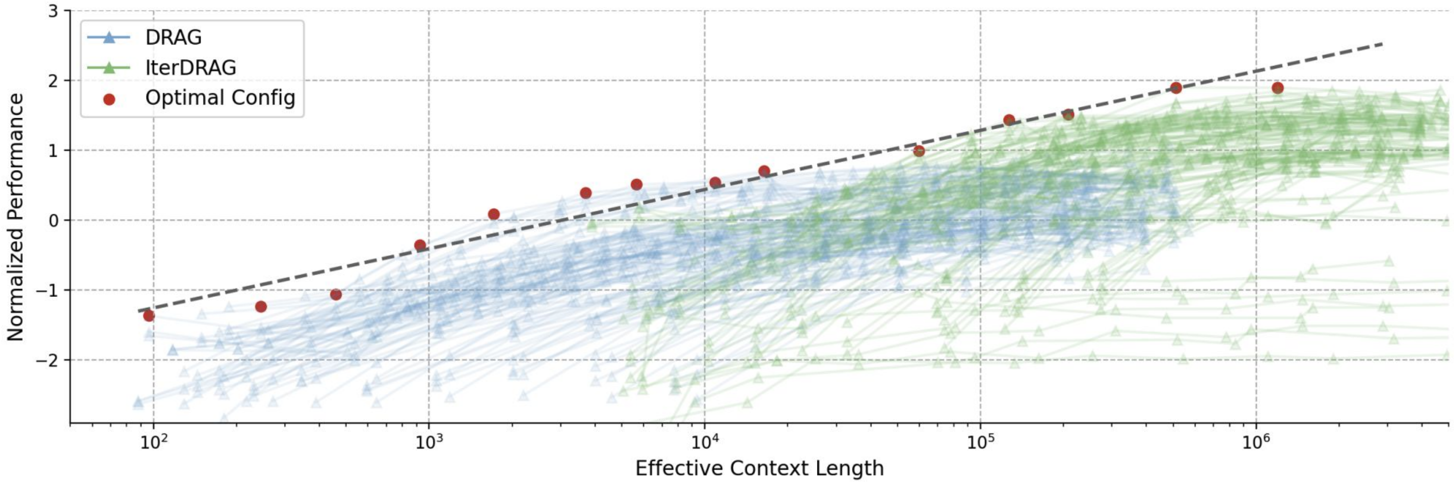
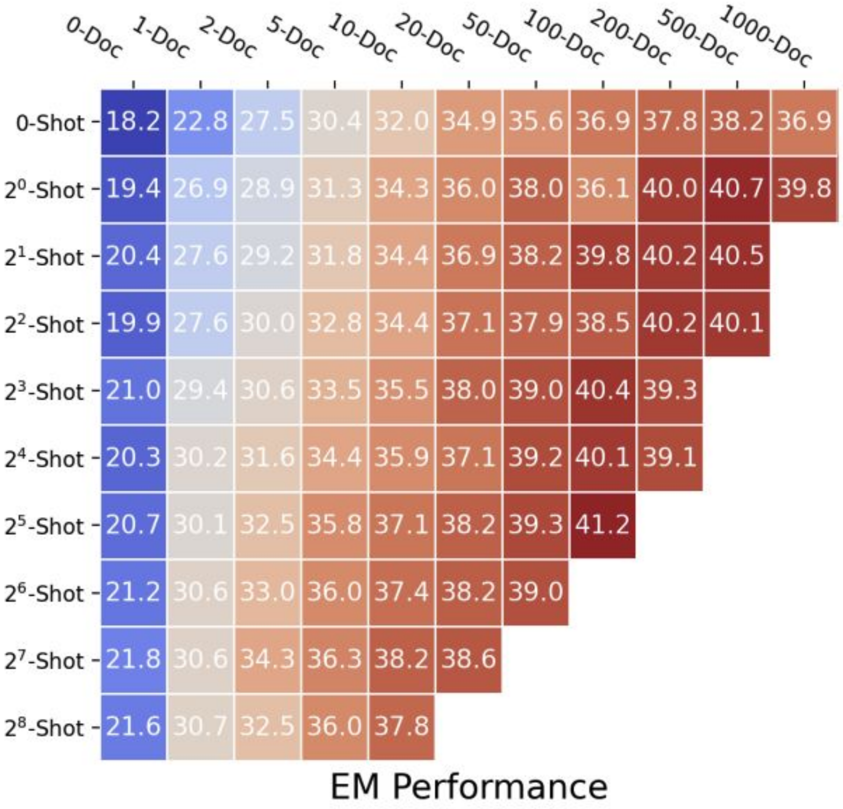


Figure 4 | Normalized performance vs. effective context lengths across datasets. Each line represents a fixed configuration, scaled by varying the number of documents. Red dots indicate the optimal configurations, with the dashed line showing the fitting results. The observed optimal performance can be approximated by a linear relationship with the effective context lengths.

https://arxiv.org/abs/2410.04343v1

Cost-Quality Tradeoffs in REML

Cost is influenced by more than retrieval



Averaged DRAG performance

https://arxiv.org/abs/2410.04343v1

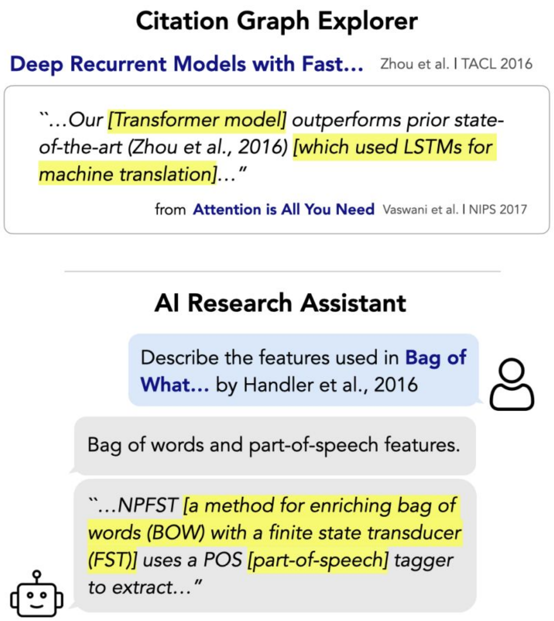
Presentation

When presenting search results to a human reader
the interface is designed to make the findings easily consumed
such as through sorting items by relevance or highlighting salient snippets.

In REML, we follow a similar principle
except the target consumer of the retrieved data is a machine,
which has a different set of limitations and capabilities.

Decontextualization

Question: What is the angle of the Tower of Pisa?	
Passage Retrieval	Prior to restoration work performed between 1990 and 2001, the tower leaned at an angle of 5.5 degrees, but the tower now leans at about 3.99 degrees. This means the top of the Leaning Tower of Pisa is displaced horizontally 3.9 meters (12 ft 10 in) from the center.
Sentence Retrieval	Prior to restoration work performed between 1990 and 2001, the tower leaned at an angle of 5.5 degrees, but the tower now leans at about 3.99 degrees.
Proposition Retrieval	The Leaning Tower of Pisa now leans at about 3.99 degrees.

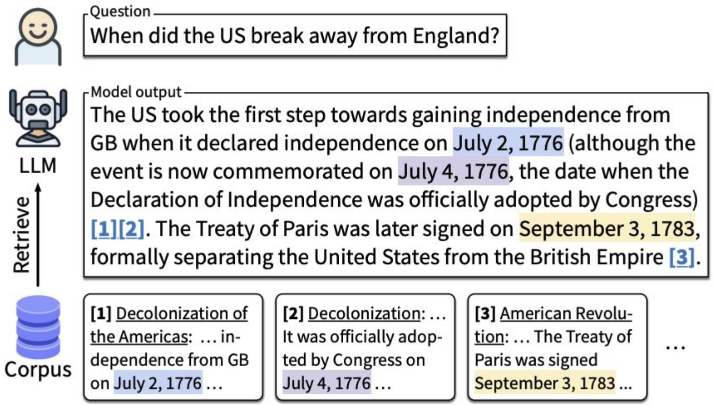


[2312.06648] Dense X Retrieval: What Retrieval Granularity Should We Use?

[2305.14772] A Question Answering Framework for Decontextualizing User-facing Snippets from Scientific Documents

Summarization: Include More Items

[2305.14627] Enabling Large Language Models to Generate Text with Citations

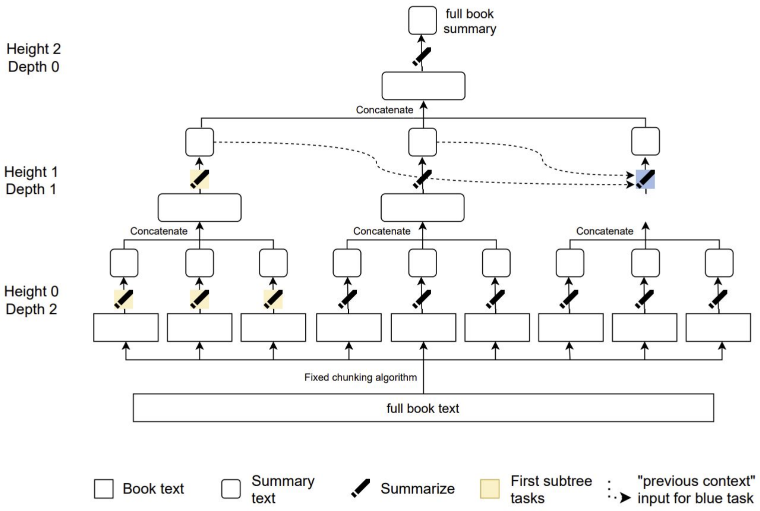


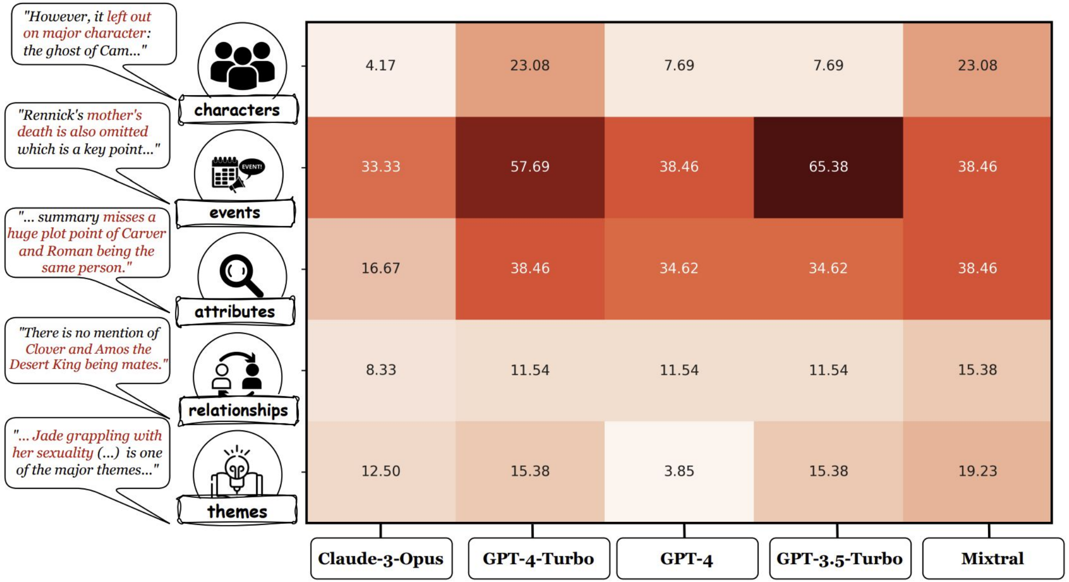
	Fluency (MAUVE)	Correct. (EM Rec.)	Citation Rec.	Citation Prec.
ChatGPT				
VANILLA (5-psg)	66.6	40.4	73.6	72.5
w/ RERANK	77.0	40.2	**84.8**	**81.6**
SUMM (10-psg)	70.0	**43.3**	68.9	61.8
w/ INTERACT	69.0	39.1	73.4	66.5
SNIPPET (10-psg)	69.8	41.4	65.3	57.4
INLINESEARCH	58.7	32.4	58.3	58.2
CLOSEDBOOK	52.7	38.3	26.7	26.7
GPT-4 (VANILLA prompting)				
GPT-4 (5-psg)	67.1	41.3	68.5	75.6
GPT-4 (20-psg)	64.9	44.4	73.0	76.5
LLaMA (VANILLA prompting)				
LLaMA-13B (3-psg)	68.4	26.9	10.6	15.4
Vicuna-13B (3-psg)	82.6	31.9	51.1	50.1
Chat-13B (5-psg)	72.4	35.2	38.4	39.4
Chat-70B (5-psg)	88.3	41.5	62.9	61.3

Tree-Structured Summarization

[2109.10862] Recursively Summarizing Books with Human Feedback

[2404.01261] FABLES: Evaluating faithfulness and content selection in book-length summarization

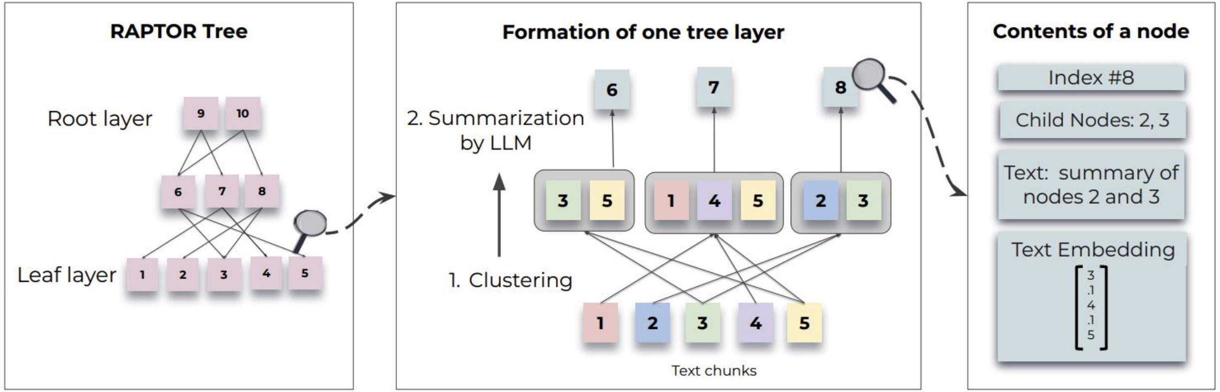


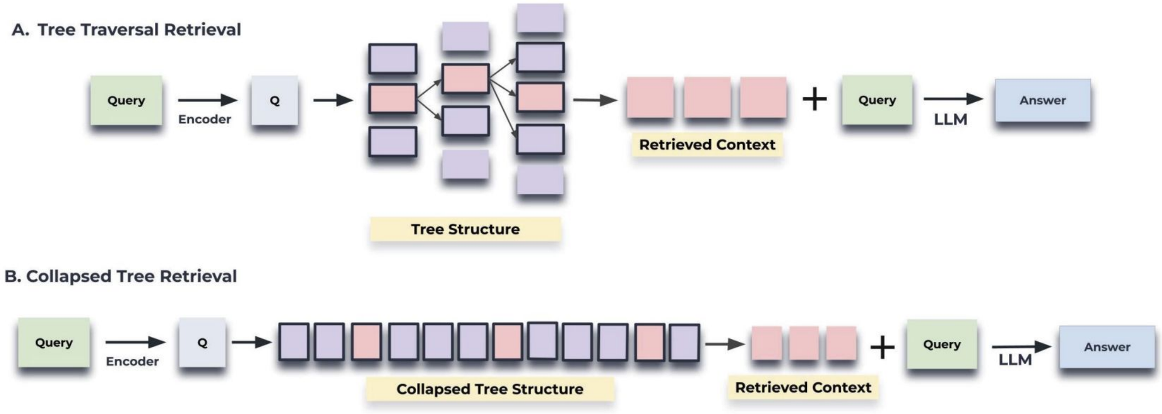


Graph-Structured Summarization

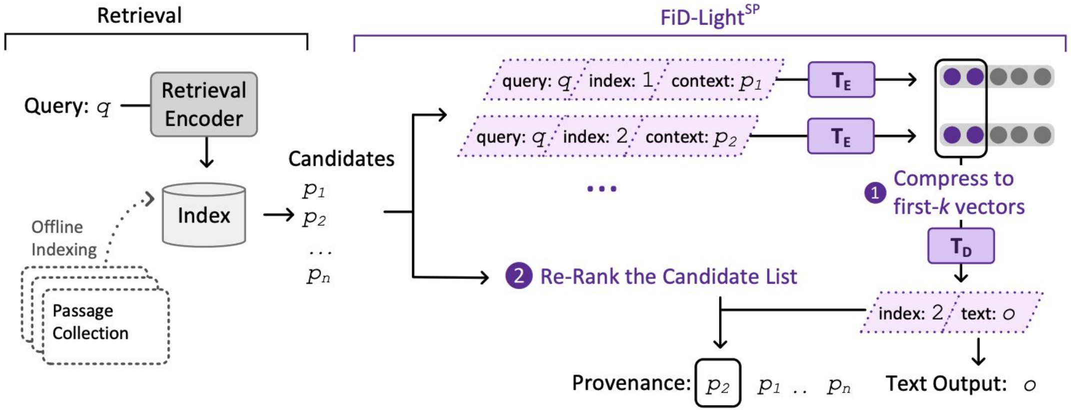
[2401.18059] RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval

Connected to:
[2404.16130] From Local to Global: A Graph RAG Approach to Query-Focused Summarization





Compressed Representation



Less Token Vectors

More Retrieved Items

[[2209.14290] FiD-Light: Efficient and Effective Retrieval-Augmented Text Generation]()

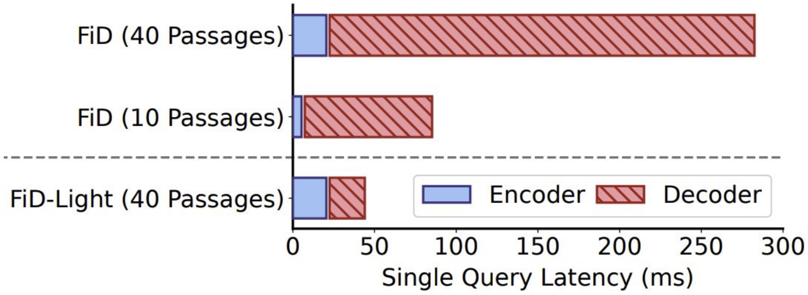
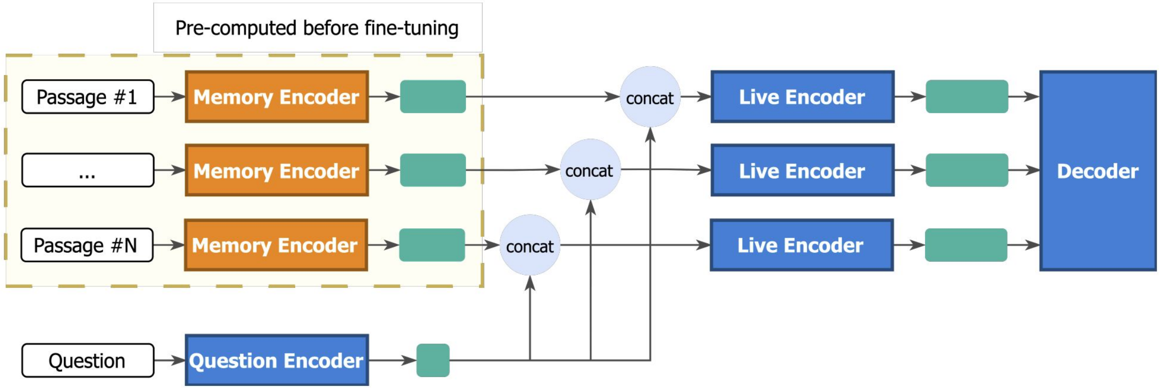
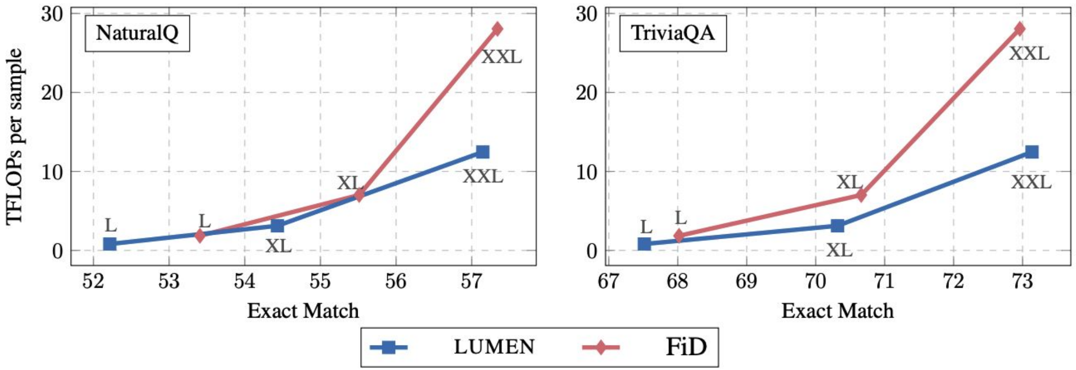


Figure 1: Average inference latency for a query of FiD & FiD-Light (T5-Base on a single TPUv4).

Incremental Representation



[2301.10448] Pre-computed memory or on-the-fly encoding? A hybrid approach to retrieval augmentation makes the most of your compute



Improving Quality via Truncation

[2004.13012] Choppy: Cut Transformer For Ranked List Truncation

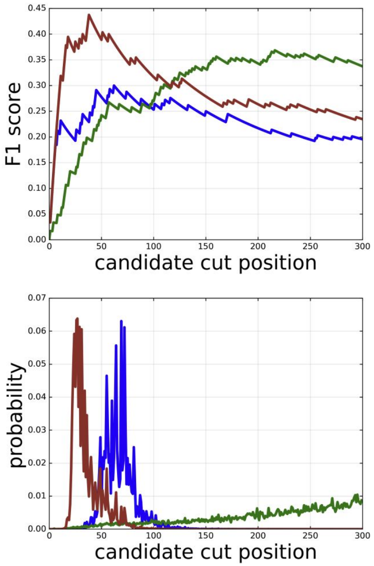


Figure 1: Top: F1 at various cut positions for 3 training queries from Robust04 BM25. Bottom: Choppy's softmax predictions for the same queries.

	BM25		DRMM	
	F1	DCG	F1	DCG
Oracle	0.367	1.176	0.375	1.292
Fixed-k (5)	0.158	-0.261	0.151	0.010
Fixed-k (10)	0.209	-0.708	0.197	-0.407
Fixed-k (50)	0.239	-5.807	0.261	-5.153
Greedy-k	0.248	-0.116	0.263	0.266
BiCut	0.244	-	0.262	-
Choppy	**0.272**	**-0.041**	**0.268**	**0.295**
Rel. % Gain	+11.5%	-	+2.29%	-

Table 1: Average F1 and DCG performance on Robust04. Choppy achieves state-of-the-art performance. "Gain" reports relative performance gain over BiCut model.

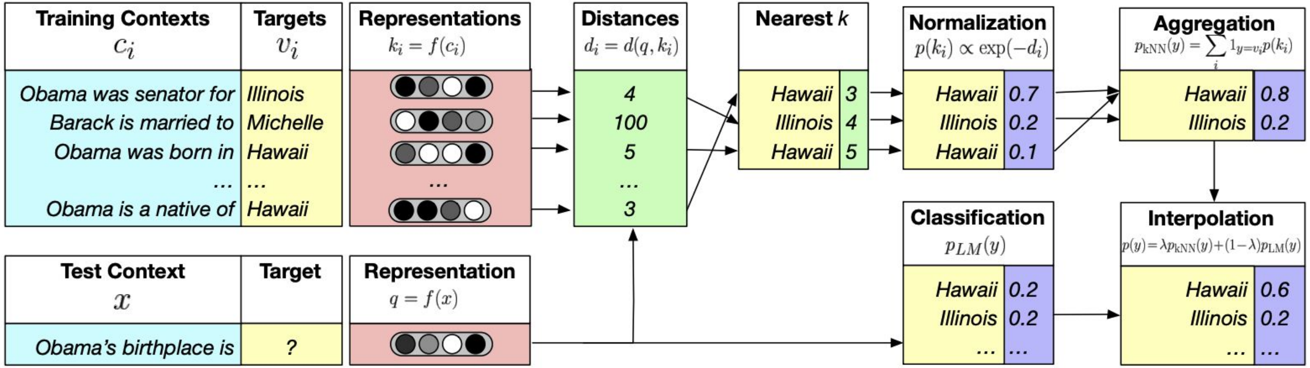
Consumption

In REML, ideally, the prediction model (f_θ)
would consume all the retrieved information simultaneously,
yet our systems are computationally limited.

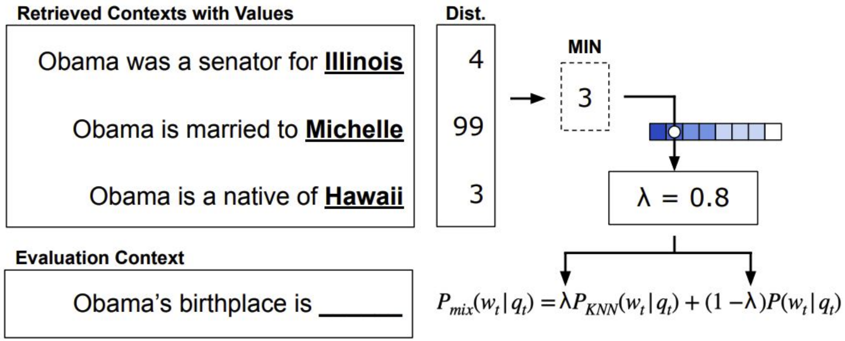
The effectiveness of f_θ is influenced by consumption-related choices
including the connection between inputs (independent vs. joint),
the connection input-output (extractive vs. abstractive),
and the granularity of output (token vs. phrase-level).

Independent, Extractive, Token-level

[1911.00172] Generalization through Memorization: Nearest Neighbor Language Models (kNN-LM)



[2210.15859] You can't pick your neighbors, or can you? When and how to rely on retrieval in the kNN-LM



Independent, Extractive, Phrase-level

[2307.06962] Copy Is All You Need

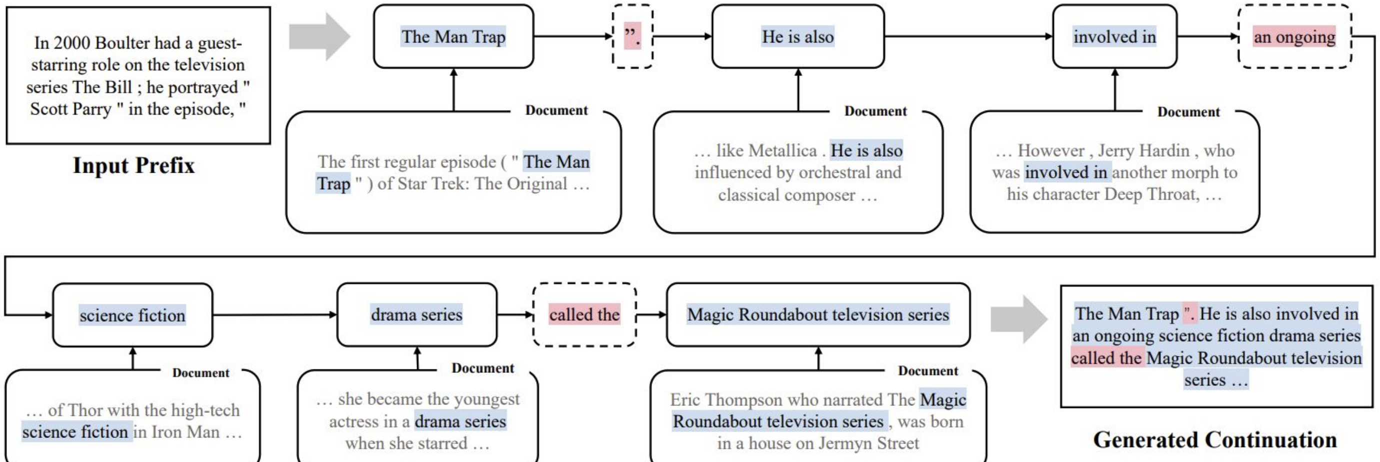


Figure 2: An example generated by CoG on the test set of WikiText-103. The dotted squares denote that the content (highlighted in red)is copied from the token vocabulary, and the solid squares denote that the content (highlighted in blue) is copied from other documents.

Independent, Extractive, Phrase-level (Cont.)

[2405.19325] Nearest Neighbor Speculative Decoding for LLM Generation and Attribution

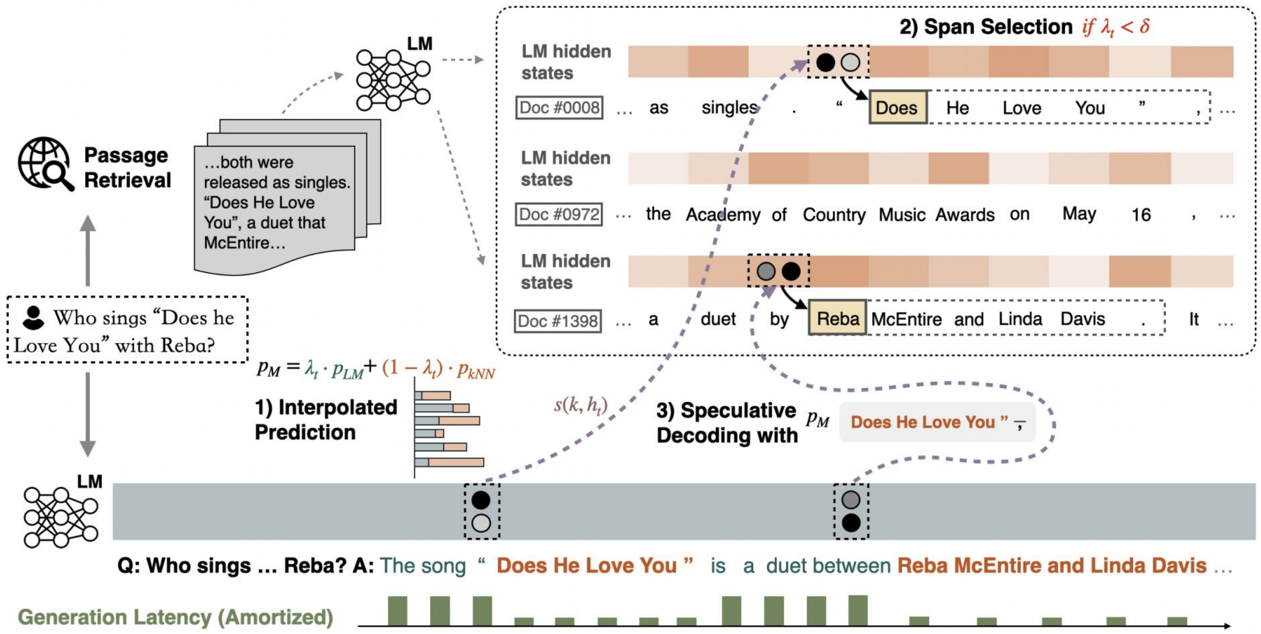
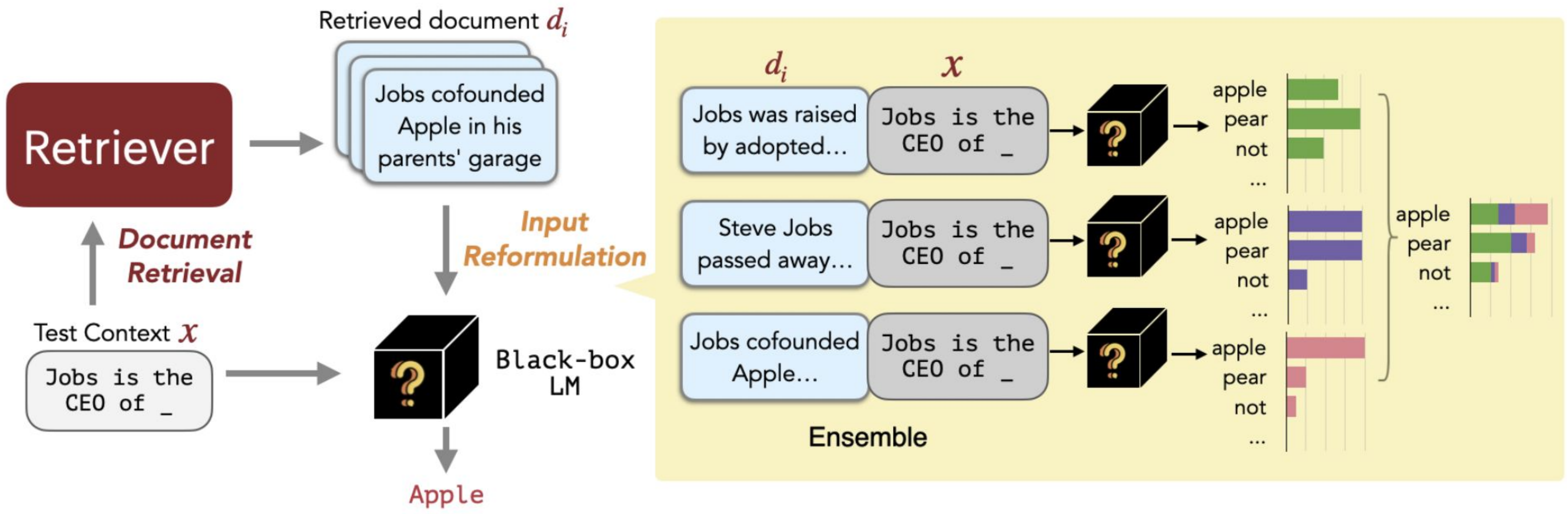


Figure 1 The NEST approach first locates the tokens in the corpus using the LM hidden states. The retrieval distribution $p_{k\text{-NN}}$ is dynamically interpolated with p_{LM} based on the retriever's uncertainty λ_t. The token and its n-gram continuation are then selected from the mixture distribution $p_{\mathcal{M}}$, while the final span length is determined by speculative decoding to remove undesired tokens. The spans incorporated in the final generation provide direct attribution and amortize the generation latency.

Independent, Abstractive

[2301.12652] REPLUG: Retrieval-Augmented Black-Box Language Models



Consuming Information in Latent Space

[2102.02557] Adaptive Semiparametric Language Models

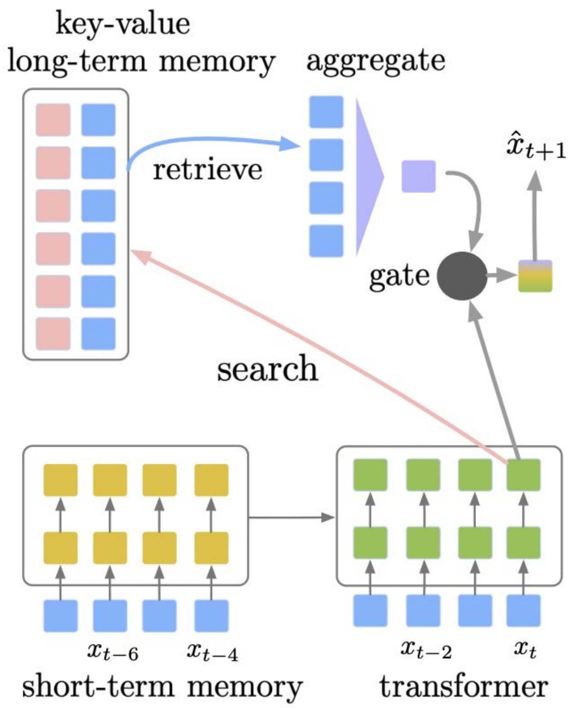


Figure 1: Our language model architecture has three main components: (i) a transformer that processes the current local context, (ii) a short-term memory module which stores hidden states from an extended context, (iii) and a key-value (hidden state-output token) database that stores compressed long-term context. At each timestep, our model combines the current context and short-term memory with a mechanism similar to transformer-XL. It then retrieves a set of past output tokens that are used in a similar context from the long-term memory module. These past output tokens are then encoded and aggregated to a single vector that represents long-term information. We use a context-dependent gate to combine information from multiple sources for making a final prediction.

Modes of Information Injection (Input-Output)

Extractive (Output-only)

- kNN-LM
- Copy is all you need
- NEST

Abstractive (Contextual)

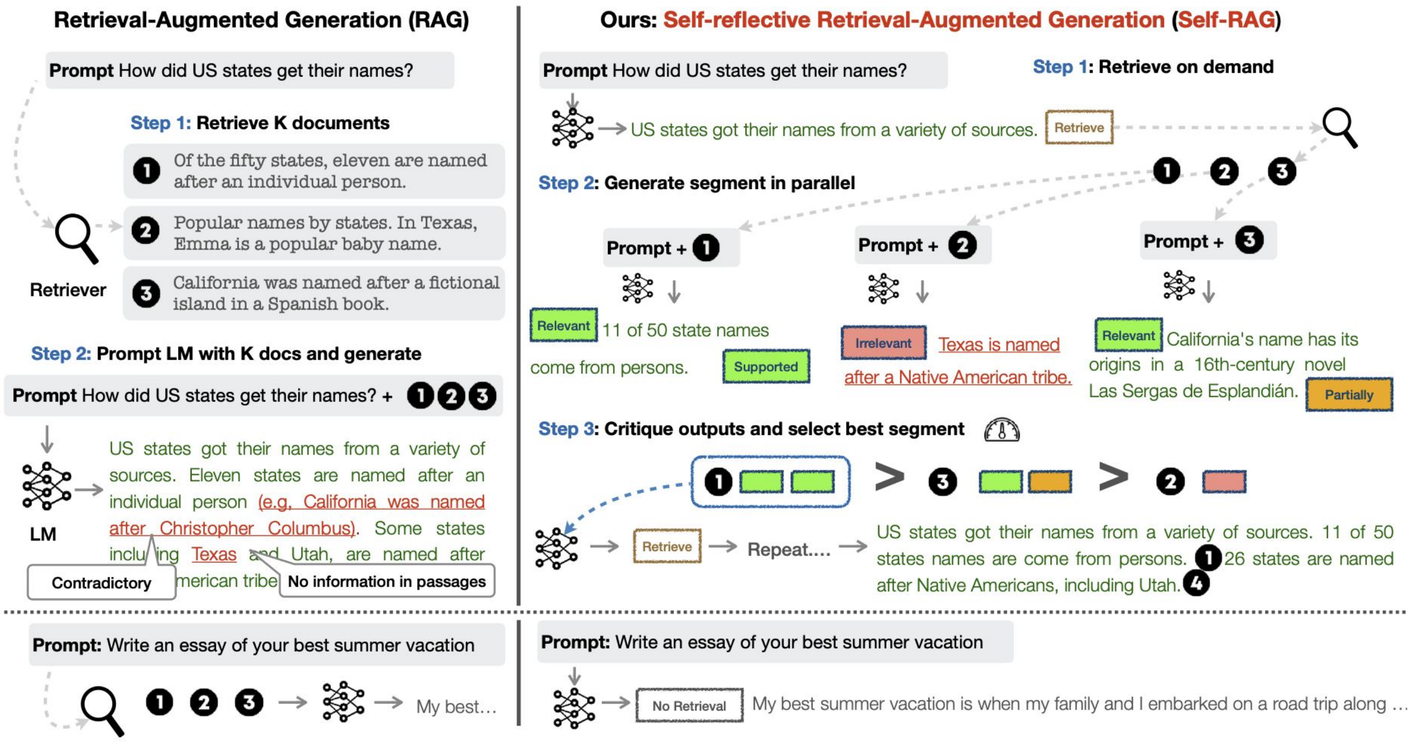
- FiD
- REPLUG

Abstractive (Latent)

- SPaLM

Reasoning in Consumption

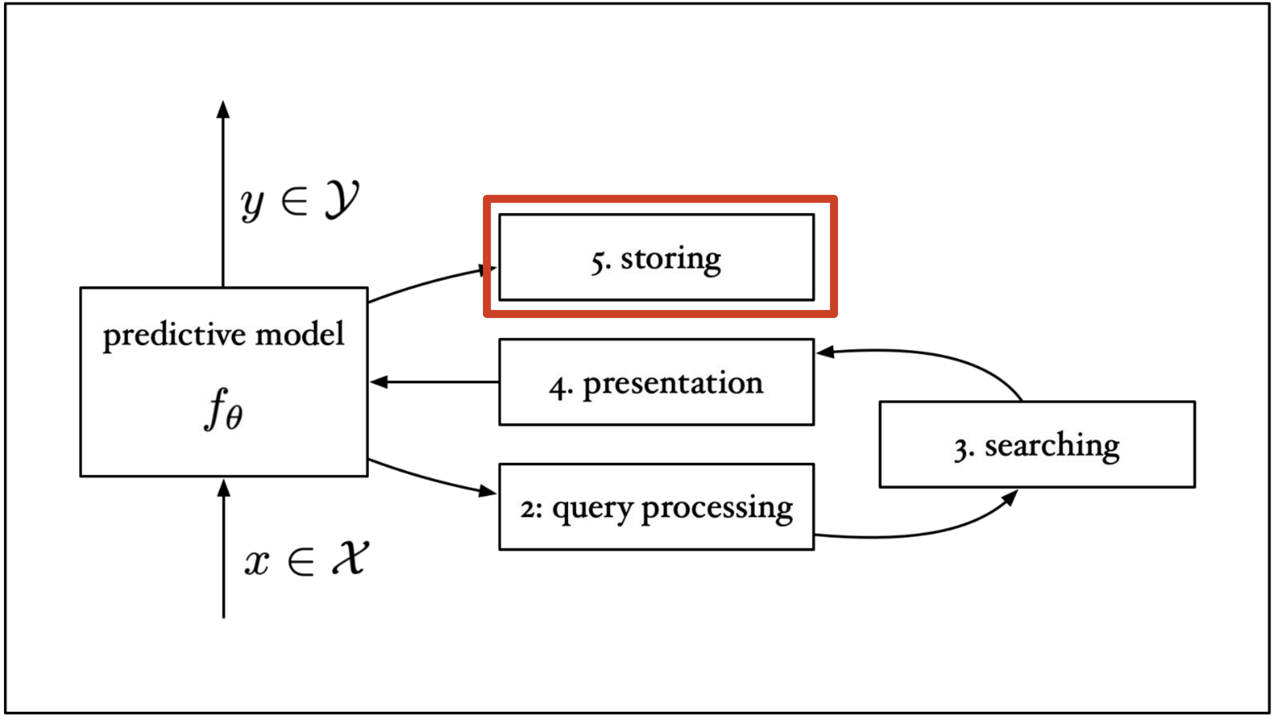
[2310.11511] Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection



questions?

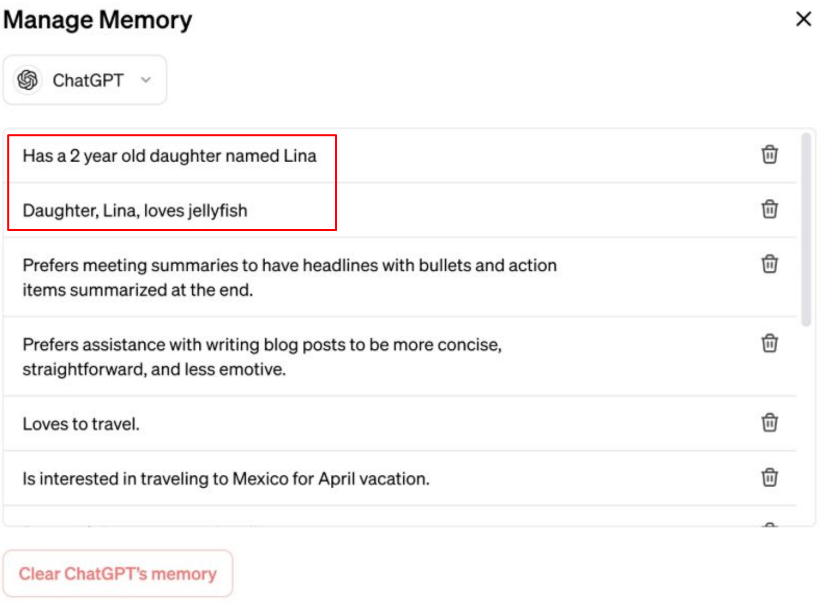
Storing

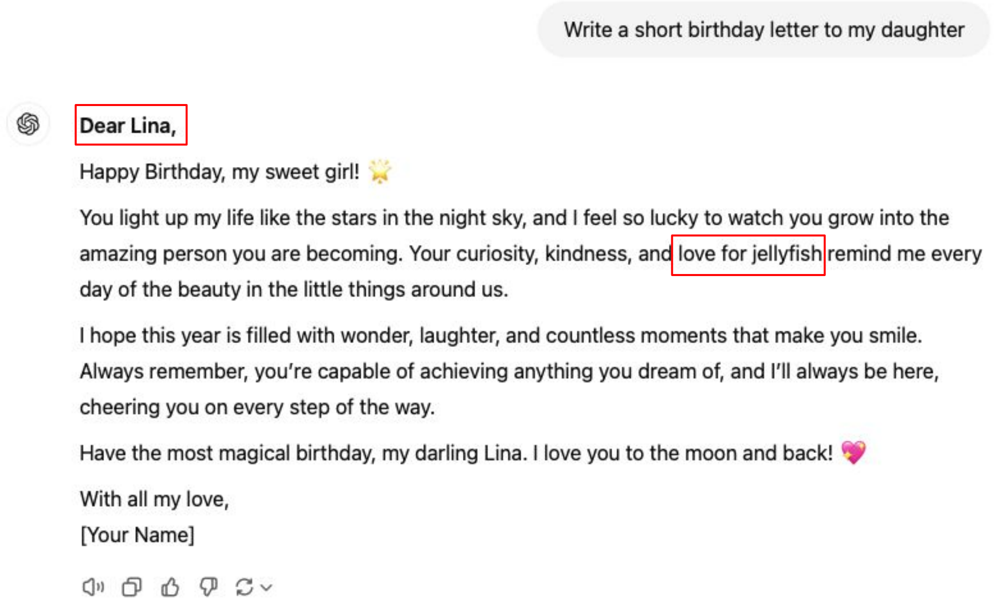
Overview


$y \in \mathcal{Y}$
5. storing
predictive model
f_θ
4. presentation
3. searching
2: query processing
$x \in \mathcal{X}$

Motivation

As an optional but critical component of REML,
a predictive model can **archive** some information that will later be useful.





Motivation

Cache Computation

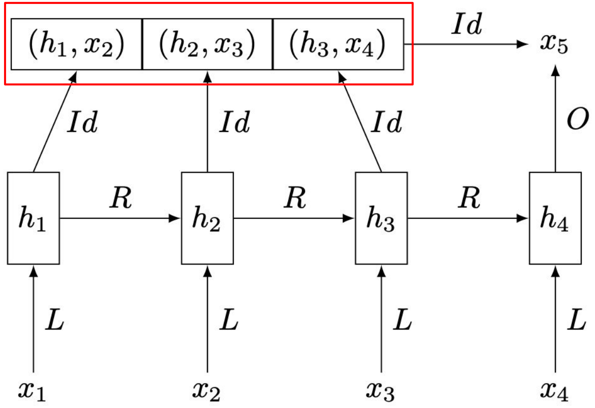
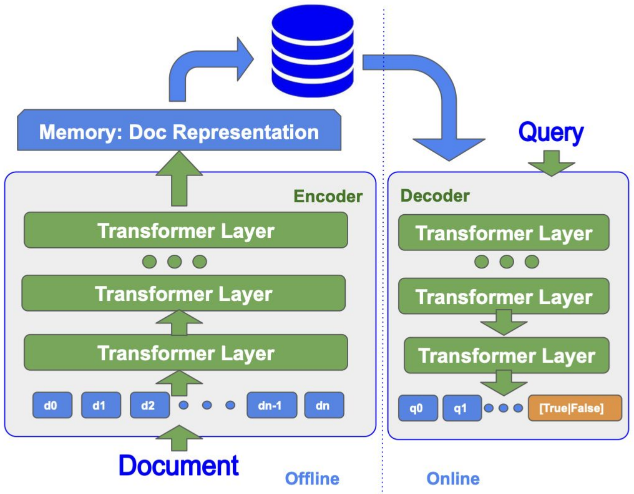


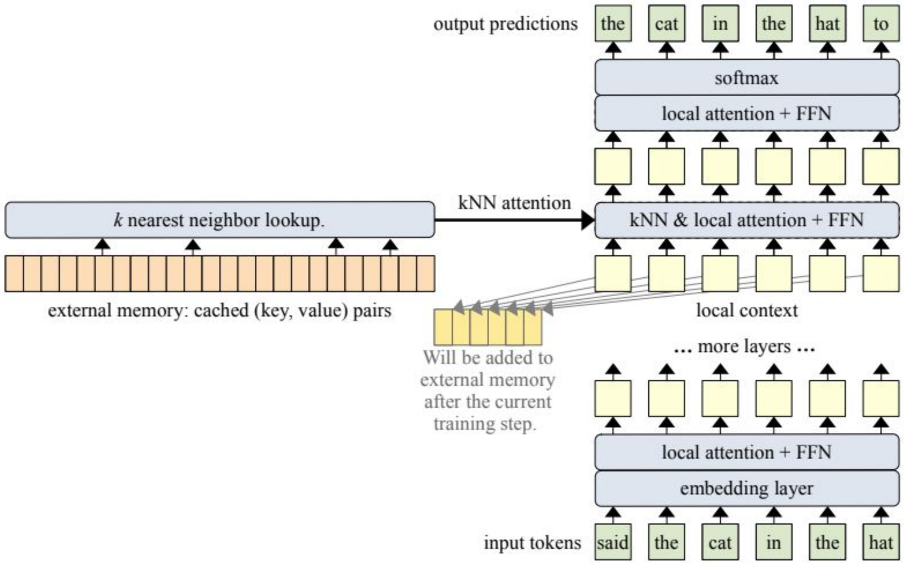
Figure 1: The neural cache stores the previous hidden states in memory cells. They are then used as keys to retrieve their corresponding word, that is the next word. There is no transformation applied to the storage during writing and reading.

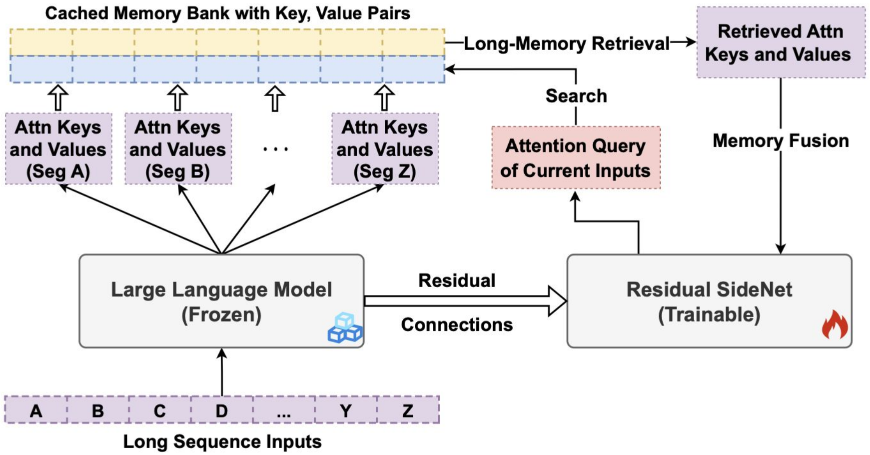


[1] Grave, E., et al. (2017). Improving Neural Language Models with a Continuous Cache (ICLR).
[2] Hui, K., et al. (2022). ED2LM: Encoder-Decoder to Language Model for Faster Document Re-ranking Inference (ACL).

Motivation

Long Context Modeling





[1] Wu, Y., et al. (2022). Memorizing Transformers (ICLR).
[2] Wang W., et al. (2023). Augmenting Language Models with Long-Term Memory (NeurIPS).

Storage Operations

- Address Generation
  - Determines where to store and read

$$w_t^{content} = address_{content}(q_t, C_t) = topK(sort(score(q_t, transform_s(C_t))), k)$$
$$w_t^{location} = address_{location}(q_t, context)$$
$$w_t = combine(w_t^{location}, w_t^{content})$$

- Read
  - Retrieves stored information (searching)

$$r_t = read(w_t, transform_s(C_t)),$$

- Write
  - Updates storage with new data

$$C_{t+1} = write(w_t, C_t, payload_t)$$

Phases of Storage Operations

Storage Construction

Offline or Online construction

Storage Management

Where to store

When to store

What to store

How to store

Storage Construction

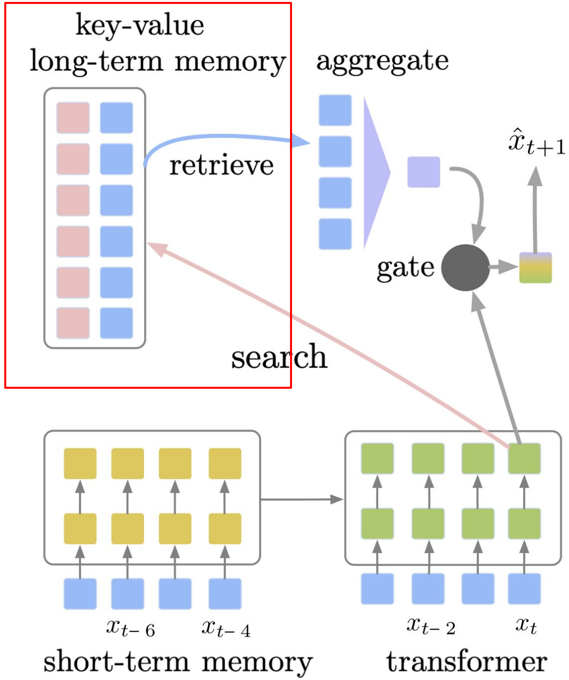
Storage Construction

Offline or Online construction

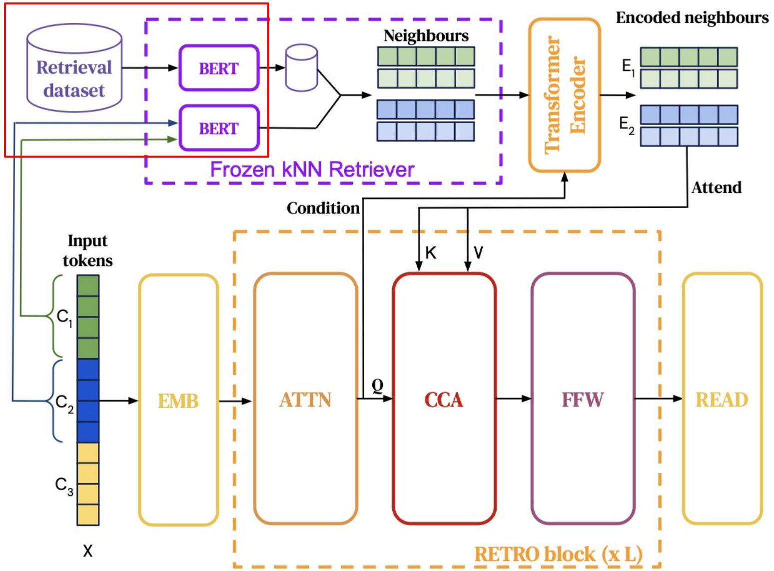
Storage Construction (offline)

$$\mathcal{D} = \{(k_i, v_i) \mid d \in C,\ k_i = transform_k(d), v_i = transform_v(d)\}$$

Offline Storage Construction



SPALM [1]

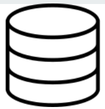


RETRO [2]

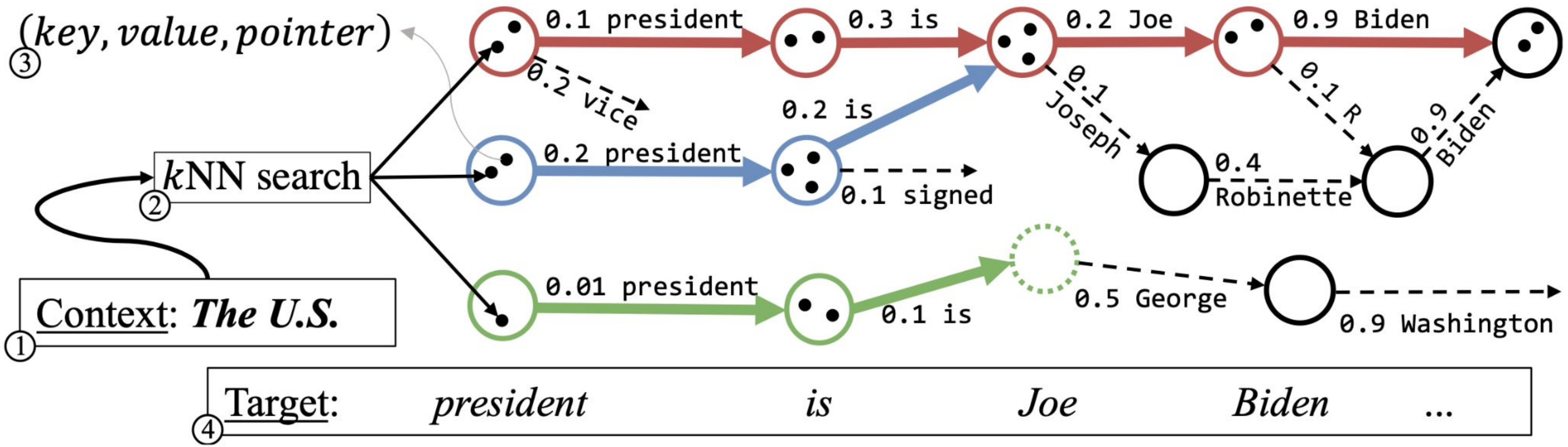
[1] Yogatama, D., et al. (2021). Adaptive Semiparametric Language Models (TACL).
[2] Borgeaud, S., et al. (2022). Improving language models by retrieving from trillions of tokens (Arxiv).

Storage Construction (offline)

$$\mathcal{D} = \{(k_i, v_i) \mid d \in C,\ k_i = transform_k(d), v_i = transform_v(d)\}$$



Offline Storage Construction

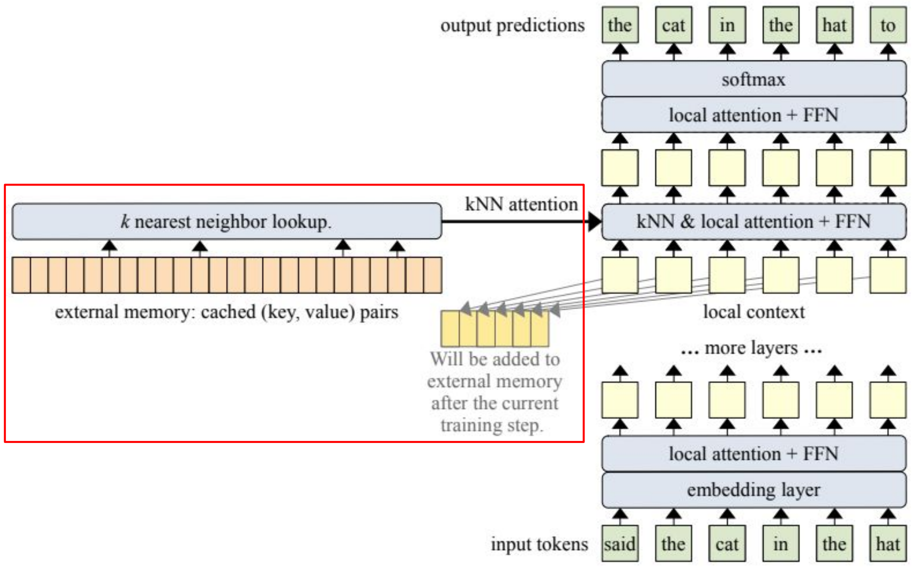


RETOMATON [1]

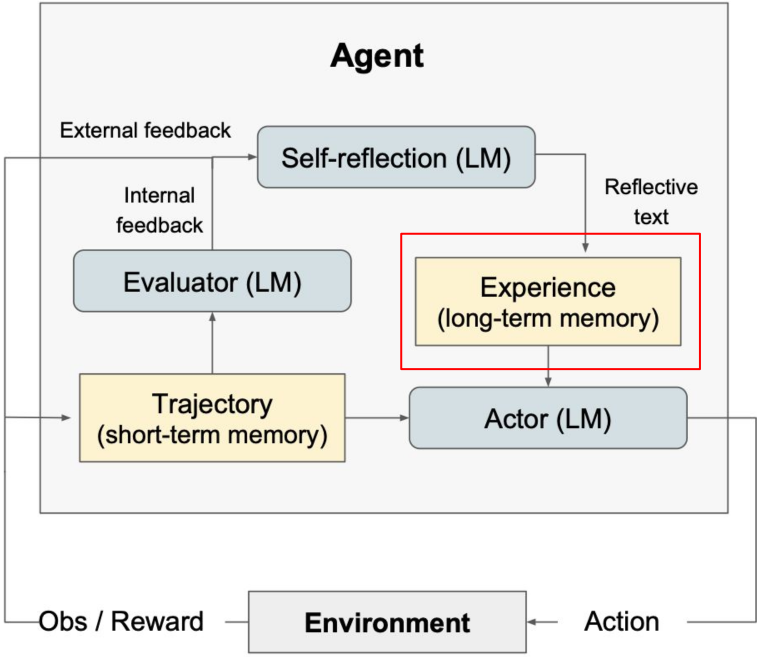
[1] Alon, U., et al. (2022). Neuro-symbolic language modeling with automaton-augmented retrieval (ICML).

Storage Construction (online)

Online Storage Construction



Memorizing Transformer [1]



Reflexion [2]

[1] Wu, Y., et al. (2022). Memorizing Transformers (ICLR).
[2] Shinn, N., et al. (2023). Reflexion: Language Agents with Verbal Reinforcement Learning (NeurIPS).

Storage Management

Storage Management

Where to store

When to store

What to store

How to store

Storage Management (where to store)

Where to store

- Sequential appending to the next available slot (chronological)
- Overwrite old or unnecessary data

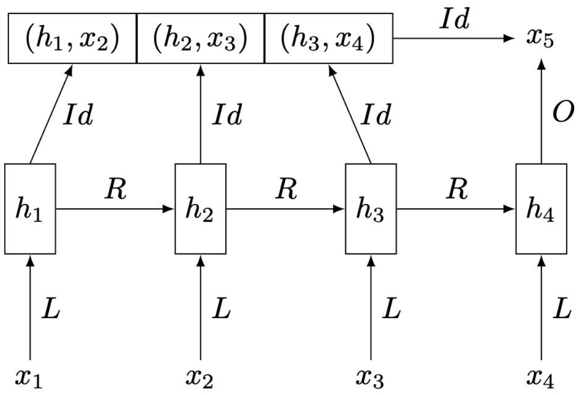
$$w_t^{content} = address_{content}(q_t, C_t) = topK(sort(score(q_t, transform_s(C_t))), k)$$
$$w_t^{location} = address_{location}(q_t, context)$$
$$w_t = combine(w_t^{location}, w_t^{content})$$

$$C_{t+1} = write(w_t, C_t, payload_t)$$

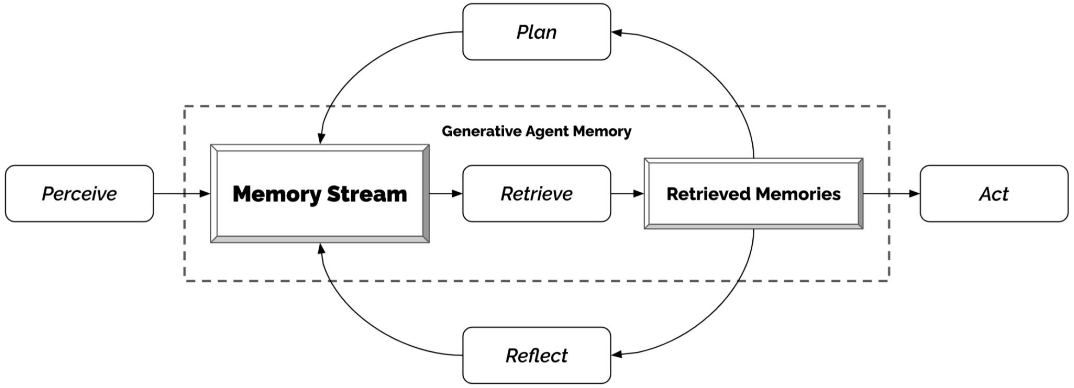
Storage Management (where to store)

Where to store

- Sequential appending to the next available slot (chronological)
  - Neural Cache Model [1]
  - Generative Agents [2]
  - What if the storage becomes full? FIFO queue style management [3, and many other agent works]
- Overwrite on old or unnecessary data



Neural Cache Model [1]



Generative Agents [2]

[1] Grave, E., et al. (2017). Improving Neural Language Models with a Continuous Cache (ICLR).
[2] Park, J.S., et al. (2023). Generative Agents: Interactive Simulacra of Human Behavior (UIST).
[3] Rae, J.W., et al. (2020). Compressive Transformers for Long-Range Sequence Modelling (ICLR).

Storage Management (where to store)

Where to store

- Sequential appending to the next available slot (chronological)
- Overwrite on old or unnecessary data
  - Memory Networks [1]
    - An erasure module that scores the utility of each entry in the slot to discard least useful entries.
  - Neural Cache Model [2]
    - Discarding oldest entries and manage the storage like a queue.

[1] Weston, J., et al. (2015). Memory Networks (ICLR).
[2] Grave, E., et al. (2017). Improving Neural Language Models with a Continuous Cache (ICLR).

Storage Management (when/what to store)

When/What to store

- **Storage Staleness**
  - Retriever's parameter can be updated while there are storage updates.
    - E.g., Retriever and Predictive Models are often trained jointly.
    - The storage/index becomes stale.

- When to update?
  - Synchronous update (every training step)
  - Asynchronous update (every T training steps)
- What to update?
  - Full index update
  - Partial index update

	Synchronous	Asynchronous
Full	Synchronous Full Update	Asynchronous Full Update
Partial	Synchronous Partial Update	Asynchronous Partial Update

Storage Management (when/what to store)

	Synchronous	Asynchronous
Full	Synchronous Full Update	Asynchronous Full Update
Partial	Synchronous Partial Update	Asynchronous Partial Update

When/What to store

- Updating the full index every training step
- Attempted in Unlimiformer [1] and RPT [2]
- However, large computational overhead [3].

$$N \times P_{retr}$$

Number of documents in index

The number of parameters of a retriever

[1] Bertsch, A., et al. (2023). Unlimiformer: Long-Range Transformers with Unlimited Length Input (NeurIPS).
[2] Rubin, O., et al. (2024). Retrieval-Pretrained Transformer: Long-range Language Modeling with Self-retrieval (TACL).
[3] Izacard, G., et al. (2024). Atlas: few-shot learning with retrieval augmented language models (JMLR).

Storage Management (when/what to store)

	Synchronous	Asynchronous
Full	Synchronous Full Update	Asynchronous Full Update
Partial	Synchronous Partial Update	Asynchronous Partial Update

When/What to store

- Updating the full index every *T* training steps.
- Allowing temporary storage staleness
- Attempted in REALM [1], Atlas [2], REPLUG [3] , and EMAT [4]
  - REALM: update the full index every 500 training steps
  - EMAT: Full index update only after each training epoch.
- Less computational overhead [2].

$$\frac{N \times P_{retr}}{B \times K \times P_{lm} \times T}$$

Batch Size

Number of docs retrieved and consumed

Every T training steps

Parameter size of LM

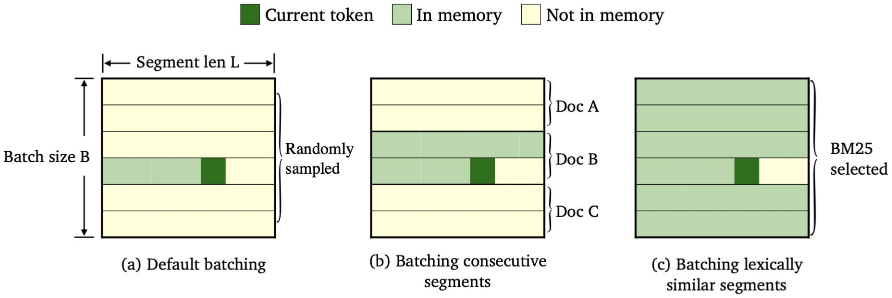
[1] Guu, K., et al. (2020). REALM: retrieval-augmented language model pre-training (ICLM).
[2] Izacard, G., et al. (2024). Atlas: few-shot learning with retrieval augmented language models (JMLR).
[3] Shi, W., et al (2024). REPLUG: Retrieval-Augmented Black-Box Language Models (NAACL).
[4] Wu, Y., et al. (2022). An efficient Memory-Augmented Transformer for Knowledge-Intensive NLP Tasks (EMNLP).

Storage Management (when/what to store)

	Synchronous	Asynchronous
Full	Synchronous Full Update	Asynchronous Full Update
Partial	Synchronous Partial Update	Asynchronous Partial Update

When/What to store

- Updating part of the index every training step.
  - Selecting a batch of entries to update
- Attempted in TRIME [1] and NPM [2]
  - TRIME: selection of batch through lexical similarity (BM25)
  - NPM: selection of batch through in-document sampling
    - Building BM25 index with pre-training corpus is expensive
    - Therefore, select a batch by grouping entities from the same document.



TRIME [1]

[1] Zhong, Z., et al. (2022). Training Language Models with Memory Augmentation (EMNLP).
[2] Min, S., et al. (2023). Nonparametric Masked Language Modeling (ACL).

Storage Management (when/what to store)

	Synchronous	Asynchronous
Full	Synchronous Full Update	Asynchronous Full Update
Partial	Synchronous Partial Update	Asynchronous Partial Update

When/What to store

- Rarely used in the literature
    - May degrade the training performance by a large margin.

Storage Management (when/what to store)

	Synchronous	Asynchronous
Full	Synchronous Full Update	Asynchronous Full Update
Partial	Synchronous Partial Update	Asynchronous Partial Update

Avoid the problem

When/What to store

- Avoid re-indexing
  - Attempted in REALM [1], Atlas [2], RAG [3], LongMem [4]
  - Query-side Training
    - Fix the parameters for document encoder
    - Only train the query encoder
    - → Embeddings of the documents (keys) are fixed → do not need to refresh the index
    - Impact of query-side training varies greatly for different tasks [2]

[1] Guu, K., et al. (2020). REALM: retrieval-augmented language model pre-training (ICLM).
[2] Izacard, G., et al. (2024). Atlas: few-shot learning with retrieval augmented language models (JMLR).
[3] Lewis, P., et al (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks (NeurIPS).
[4] Wang W., et al. (2023). Augmenting Language Models with Long-Term Memory (NeurIPS).

Storage Management (how to store)

- Entry Representation
  - Index compression
- Architectural Choice
  - Key-Value structure
  - List structure

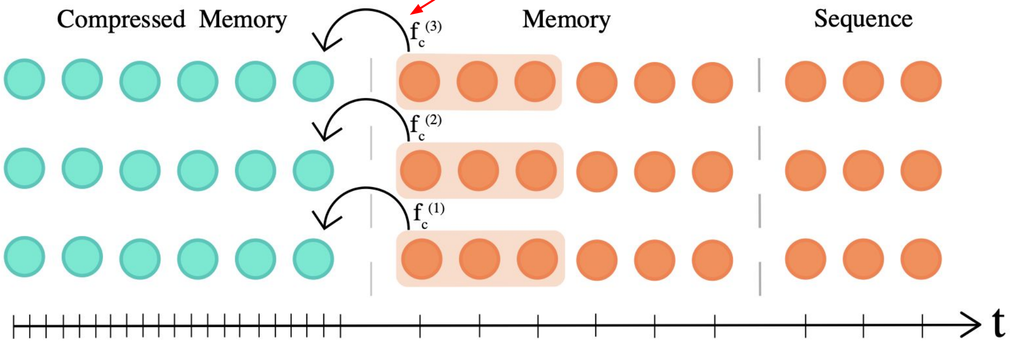
How to store

Storage Management (how to store)

- Entry Representation
  - Index compression [1 ,2, 3]
    - mean/max pooling, 1D convolution,
      erasure of low-usage memories, and quantization [3]
  - At inference time, REML model can attend to the compressed/quantized memory, reducing the memory footprint and cost.
- Architectural Choice
  - Key-Value structure
  - List structure

How to store

Compression strategy



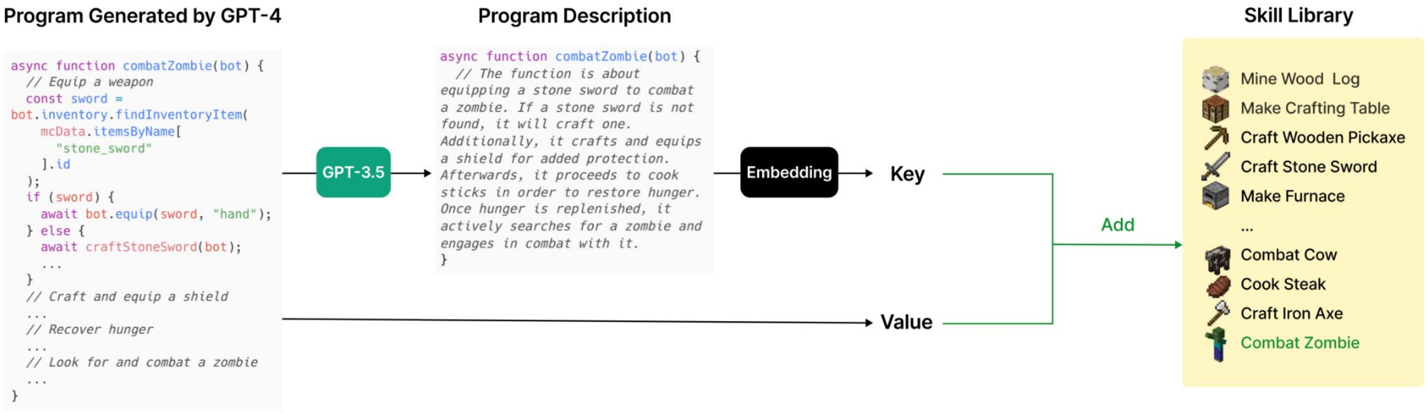
Transformer-XL style FIFO-fashioned memory management [1]

[1] Rae, J.W., et al. (2020). Compressive Transformers for Long-Range Sequence Modelling (ICLR).
[2] Wu, C.Y., et al. (2022). MeMViT: Memory-Augmented Multiscale Vision Transformer for Efficient Long-Term Video Recognition (Arxiv)
[3] Izacard, G., et al. (2024). Atlas: few-shot learning with retrieval augmented language models (JMLR).

Storage Management (how to store)

How to store

- Entry Representation
  - Index compression
  - Quantization
- Architectural Choice
  - List structure: Reflexion [1], Generative Agents [2]
  - Key-Value structure: Voyager [3], Synapse [4]

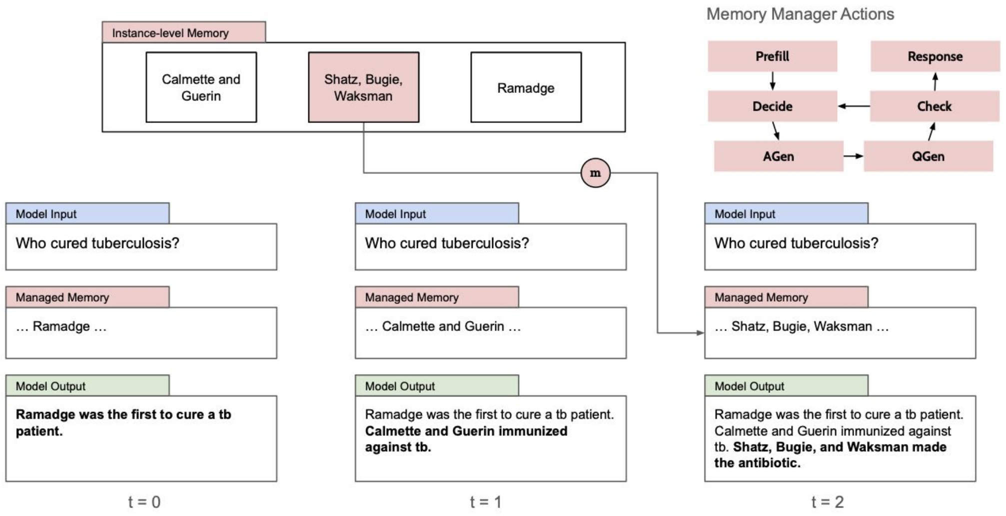


Voyager [3]

[1] Shinn, N., et al. (2023). Reflexion: Language Agents with Verbal Reinforcement Learning (NeurIPS).
[2] Park, J.S., et al. (2023). Generative Agents: Interactive Simulacra of Human Behavior (UIST).
[3] Wang, G., et al. (2024). Voyager: An Open-Ended Embodied Agent with Large Language Models (TMLR).
[4] Zheng, L., et al. (2024). Synapse: Trajectory-as-Exemplar Prompting with Memory for Computer Control (ICLR).

Future Work

- Shared Storage
  - One retriever serving multiple predictive models.
- Storage Staleness
  - No perfect way to solve this problem.
- Storing enables new capabilities.
  - Managing contextual memories with storage.
  - Retrieval-Driven Memory Manager (ReDMM).



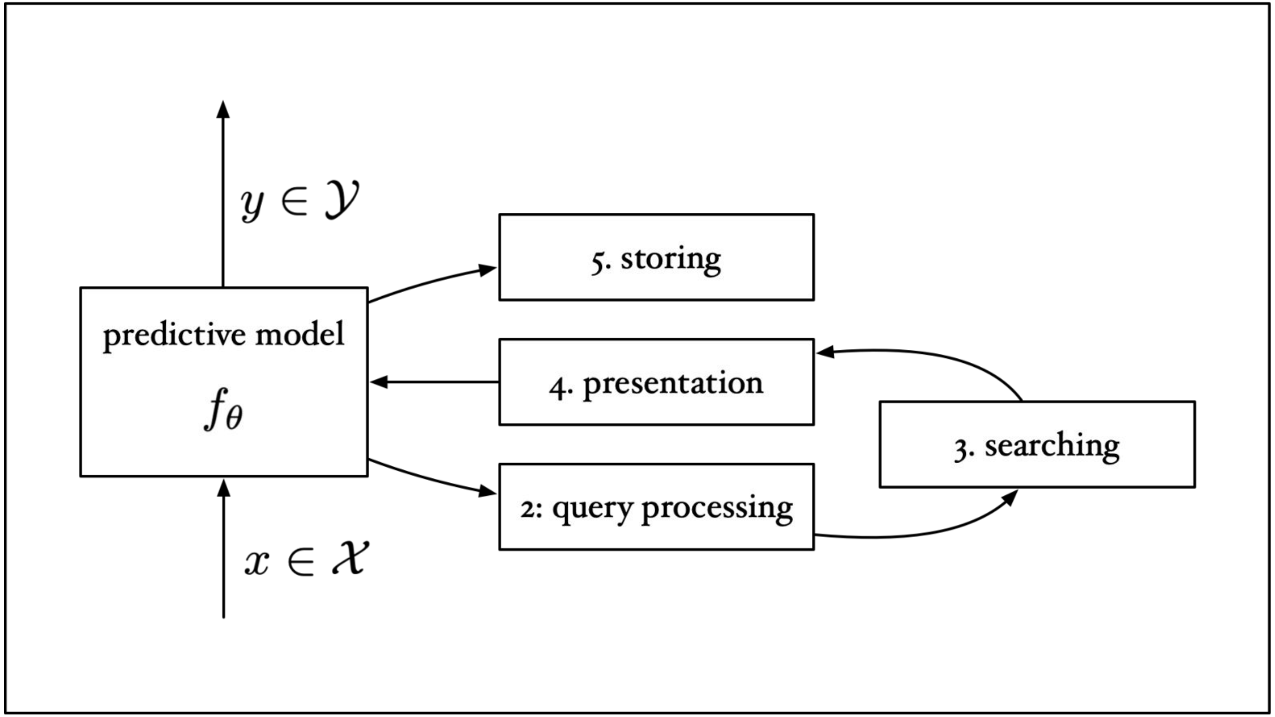
ReDMM [1]

[1] Drozdov, A. (2024). *Unlocking Natural Language Generalization with Adaptive Retrieval-based Methods (Dissertation; UMass Amherst).*

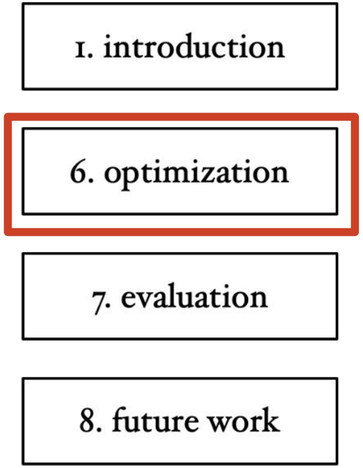
questions?

Optimization

Overview

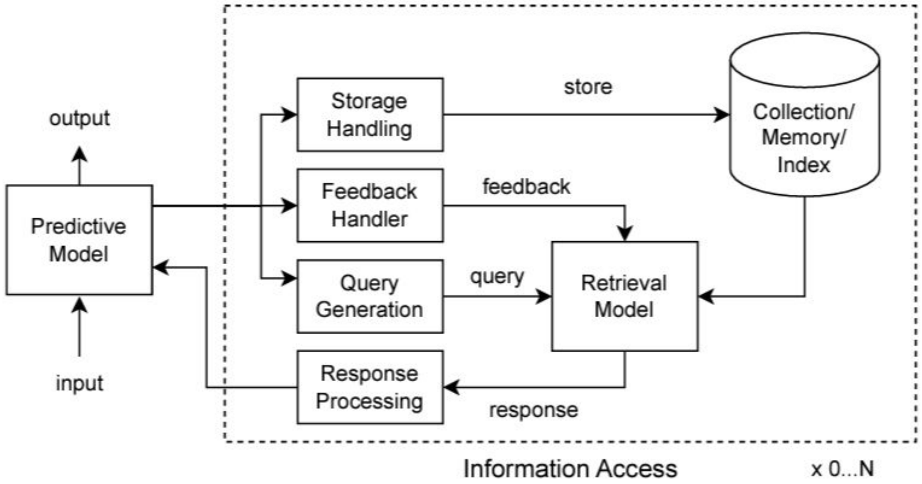


$y \in \mathcal{Y}$
predictive model
f_θ
$x \in \mathcal{X}$
5. storing
4. presentation
2: query processing
3. searching



1. introduction
6. optimization
7. evaluation
8. future work

Optimization in REML



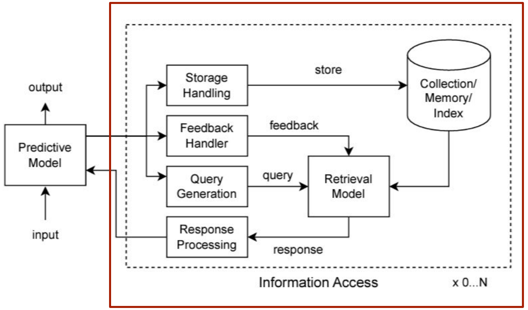
How to optimize the retrieval model(s)?

Retrieval Model Optimization:
No REML-specific Optimization

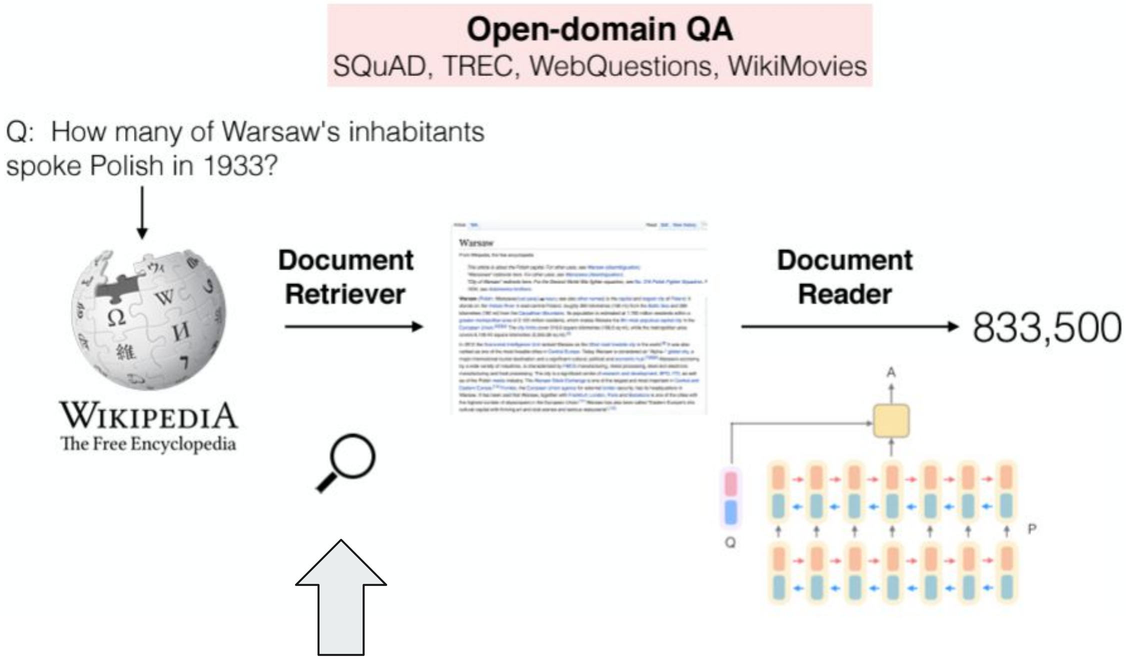
Assumption:
Retrieval optimization is independent of the downstream REML task.

Examples:

- TF-IDF
- BM25
- Language models (e.g., QL)
- Zero-shot and few-shot prompting of instruction-following LLMs for re-ranking
- SQL query submitted to databases
- Learning to rank models learned from REML-independent data
  - E.g., a neural ranking model trained on MS MARCO
  - Data can come from explicit or implicit signals from different applications.
- …



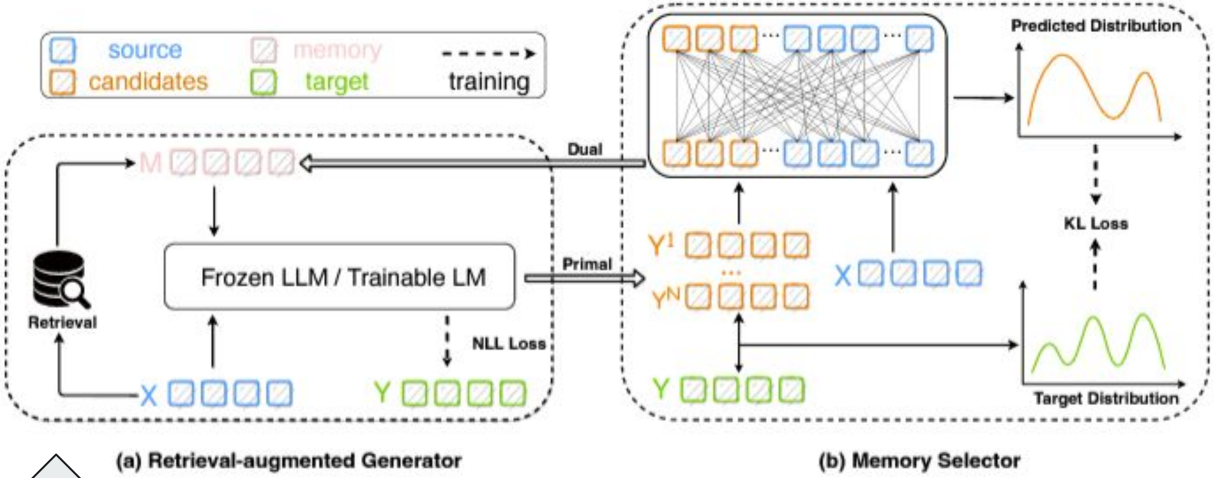
Dr.QA



Elasticsearch implementation of TF-IDF

Danqi Chen, Adam Fisch, Jason Weston, Antoine Bordes. “Reading Wikipedia to Answer Open-Domain Questions” ACL 2017.

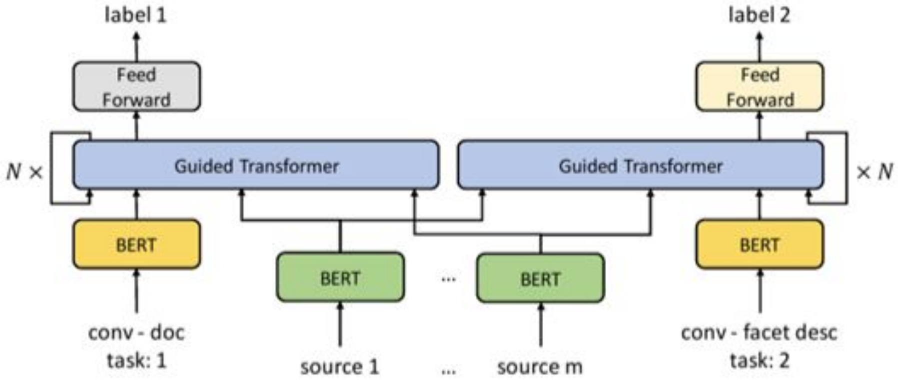
SelfMem

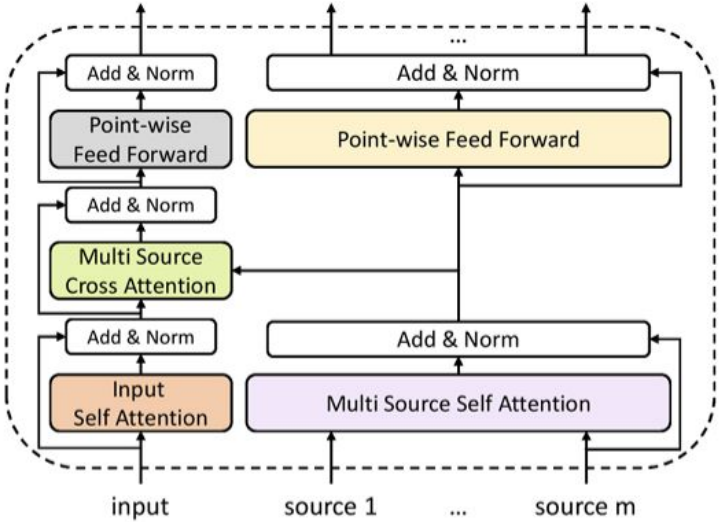


BM25 with default parameters.

Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, Rui Yan. “Lift yourself up: retrieval-augmented text generation with self-memory” NeurIPS 2023.

Guided Transformer

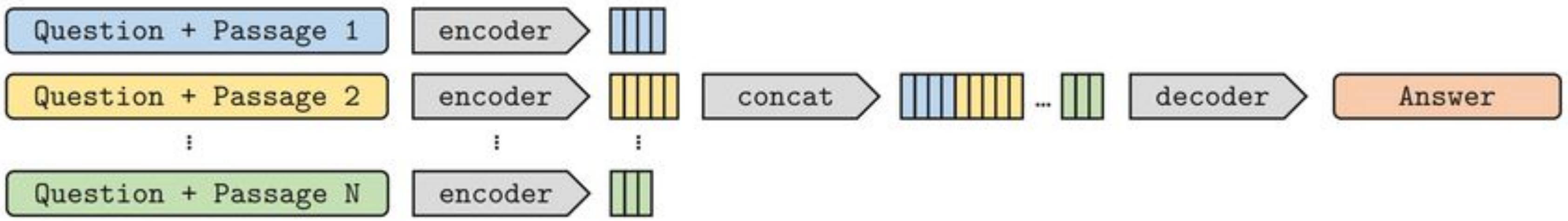


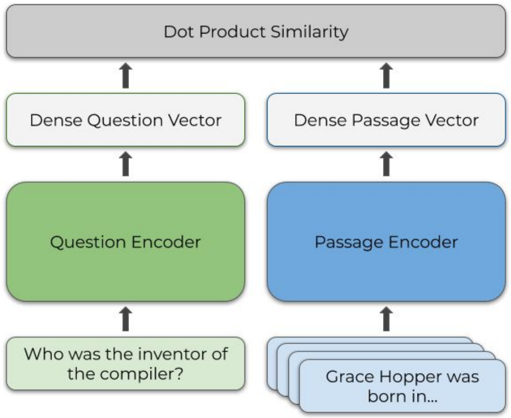


Query likelihood with Dirichlet prior smoothing.

Helia Hashemi, Hamed Zamani, W. Bruce Croft. "Guided Transformer: Leveraging Multiple External Sources for Representation Learning in Conversational Search" SIGIR 2020.

Fusion-in-Decoder

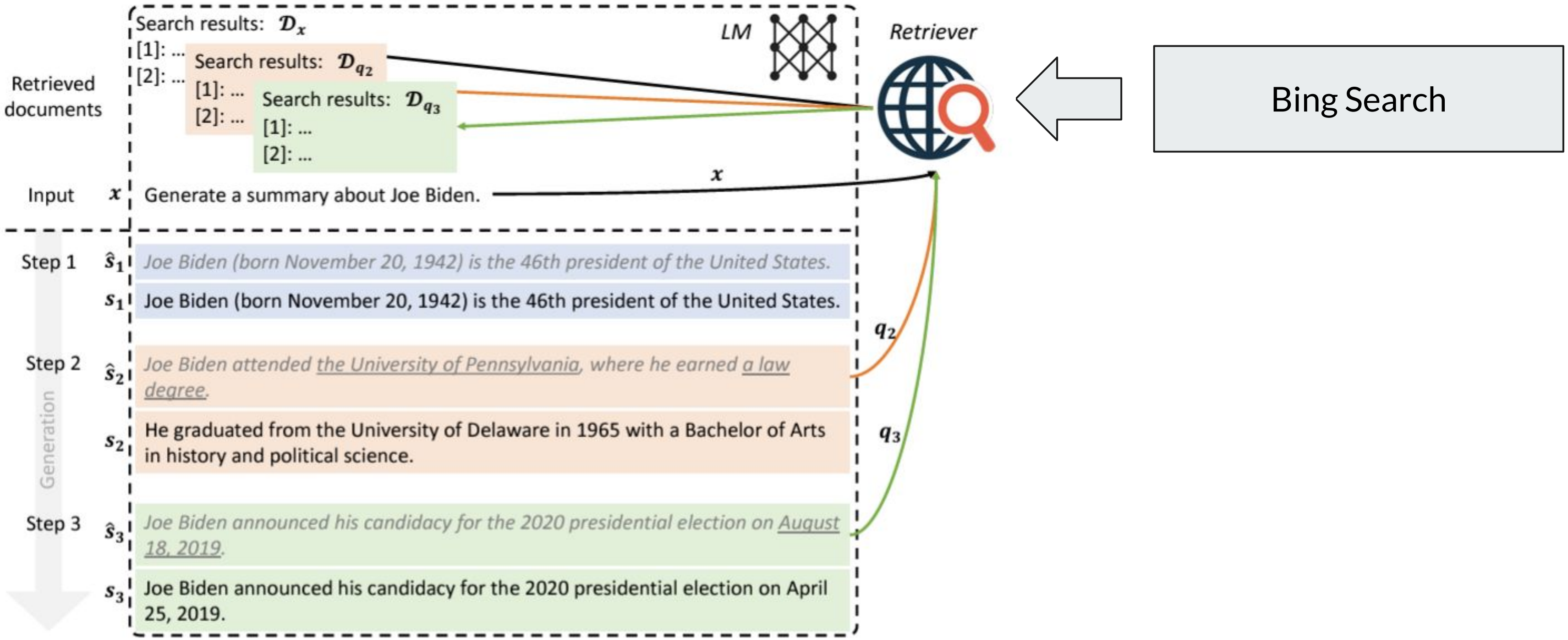




DPR trained on MS MARCO.

Gautier Izacard, Edouard Grave. “Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering” EACL 2021.

Active RAG



Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, Graham Neubig. "Active Retrieval Augmented Generation" EMNLP 2023.

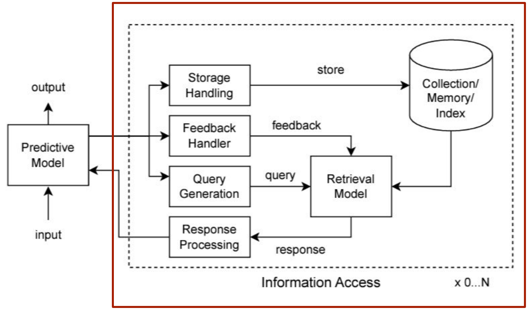
Retrieval Model Optimization: Conditional Optimization

Assumption:
Retrieval model is optimized, conditioned on the predictive model.

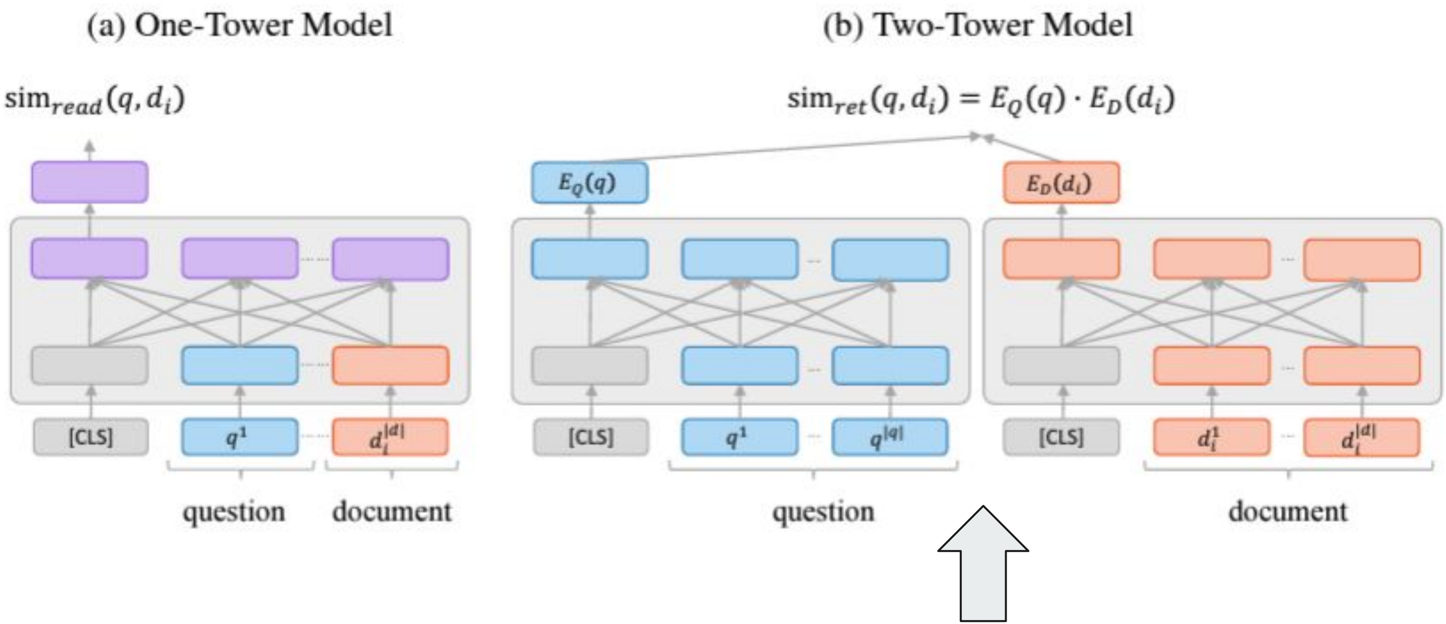
$$\omega^{(t+1)} = \arg\min_{\omega} \frac{1}{|T|} \sum_{(x,y)\in T} L\left(f_{\theta^{(t)}}\left(x; g_{\omega}\right), y\right)$$

Examples:

- Knowledge distillation from the predictive model to the retrieval model.
- Reinforcement learning where the reward model is computed based on the predictive model's output.



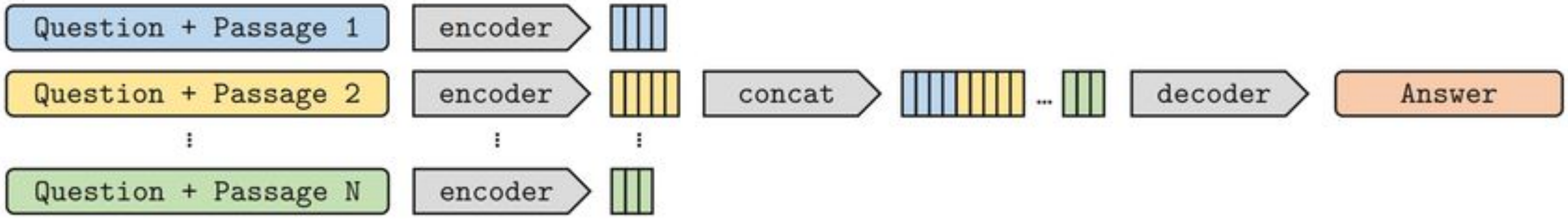
Fusion-in-Decoder with Knowledge Distillation

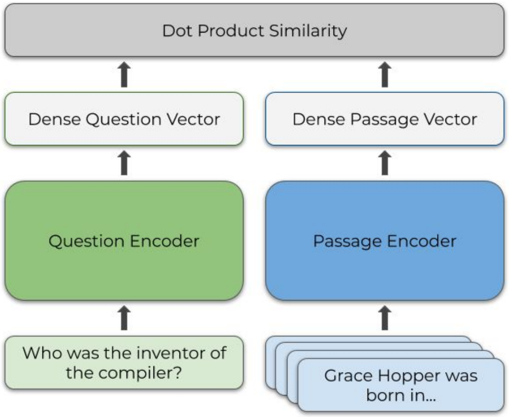


DPR trained on signals from BERT (answer span selector).

Sohee Yang and Minjoon Seo. "Is Retriever Merely an Approximator of Reader?" arxiv 2020.

Fusion-in-Decoder with Knowledge Distillation





DPR trained on signals from FiD.

Gautier Izacard, Edouard Grave. "Distilling Knowledge from Reader to Retriever for Question Answering" ICLR 2021.

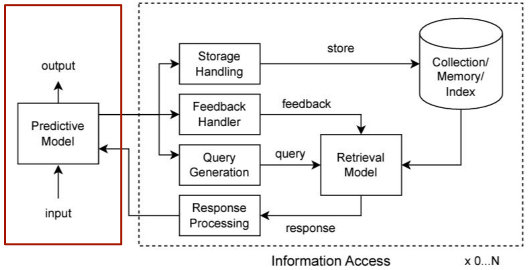
Predictive Model Optimization: No Optimization / Independent Optimization

Assumption:
Predictive model optimization is independent of the retrieval model.

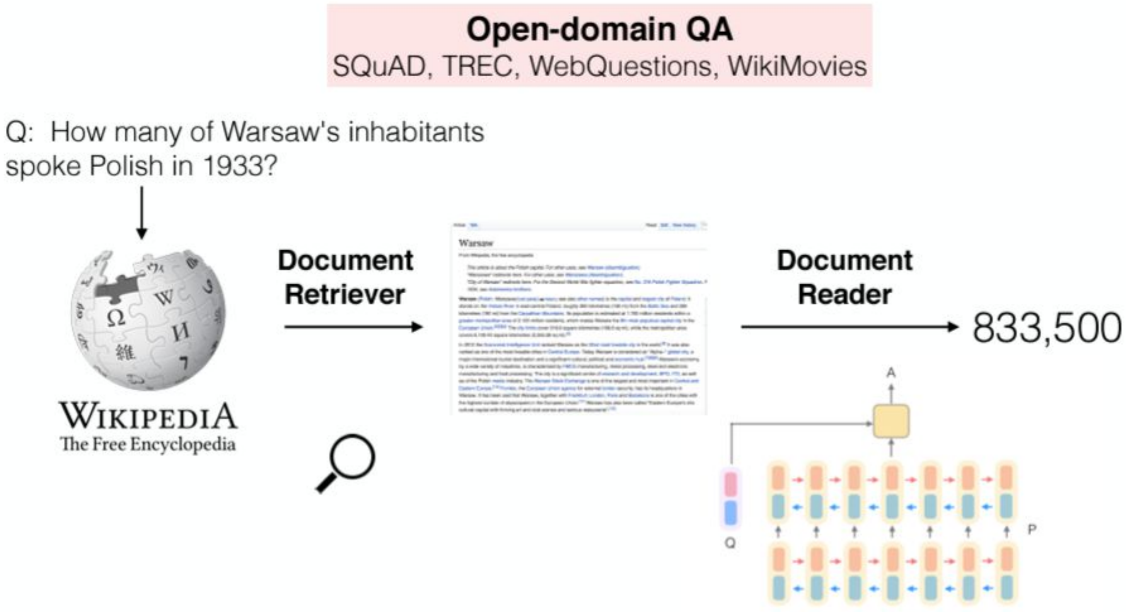
Examples:

- Using black-box large language models as predictive models.
- Optimizing predictive models by assuming that the retrieval model is optimal (using groundtruth relevance labels)

$$\theta^* = \arg\min_\theta \frac{1}{|T|} \sum_{(x,y)\in T} L\left(f_\theta\left(x; g_{\text{opt}}\right), y\right)$$



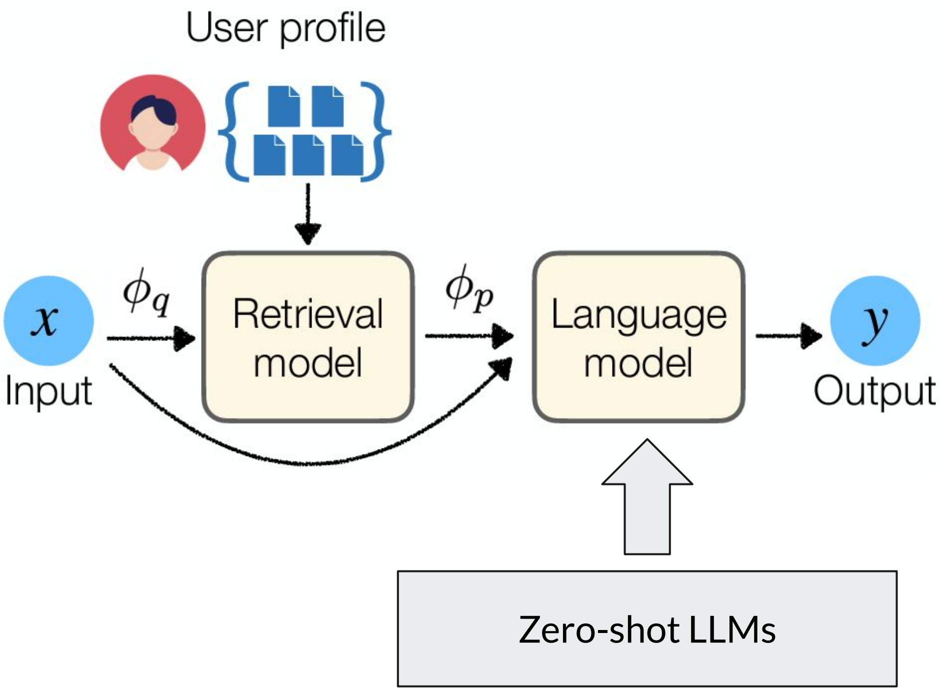
Dr.QA



Reader trained on gold documents.

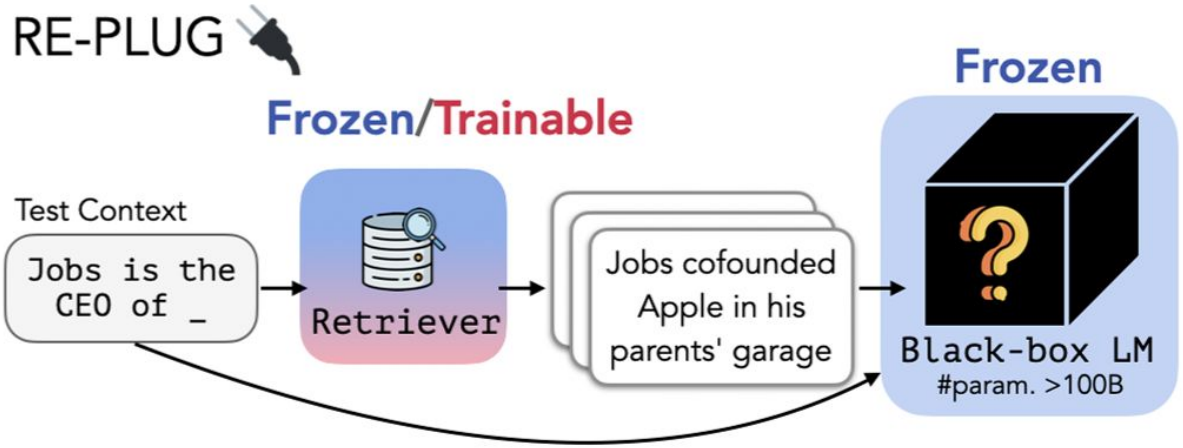
Danqi Chen, Adam Fisch, Jason Weston, Antoine Bordes. "Reading Wikipedia to Answer Open-Domain Questions" ACL 2017.

RAG for Personalized Generation



Alireza Salemi, Sheshera Mysore, Michael Bendersky, Hamed Zamani.
"LaMP: When Large Language Models Meet Personalization" ACL 2024.

RAG for Personalized Generation



Zero-shot LLMs

Weijia Shi et al. “REPLUG: Retrieval-Augmented Black-Box Language Models” NAACL 2024.

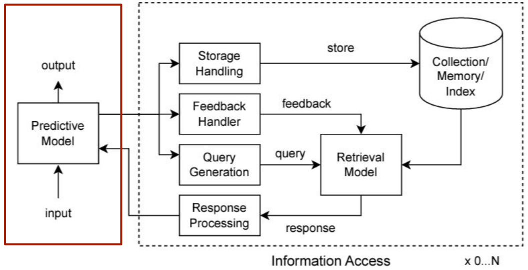
Predictive Model Optimization: Conditional Optimization

Assumption:
Predictive model is optimized, conditioned on retrieval quality.

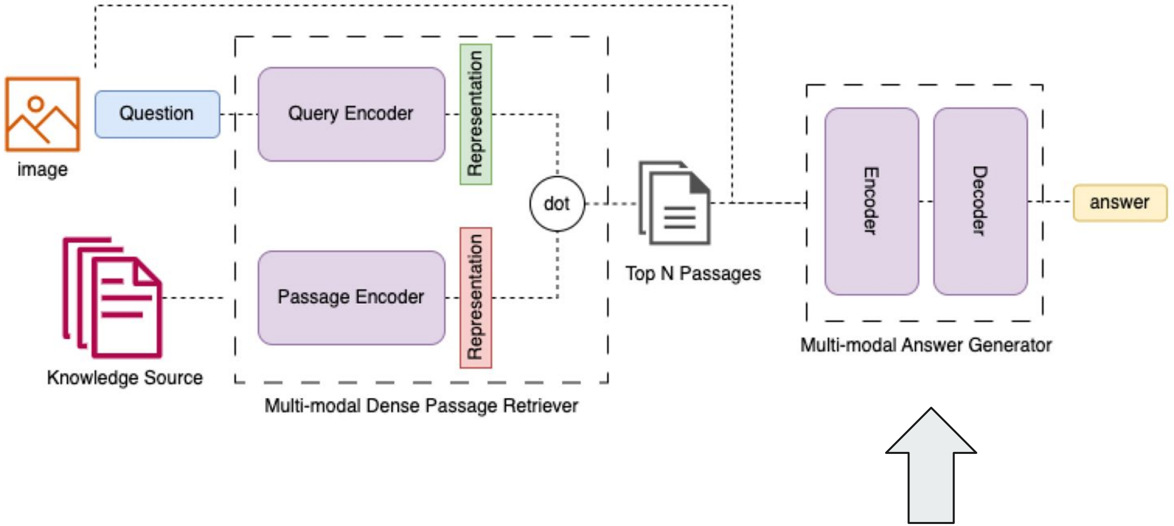
Examples:

- Optimizing predictive models using the results from the retrieval model's output.

$$\theta^{(t)} = \arg\min_\theta \frac{1}{|T|} \sum_{(x,y)\in T} L\left(f_\theta\left(x; g_{\omega^{(t)}}\right), y\right)$$



MM-FiD



Trained by feeding retrieved passages.

Alireza Salemi, Juan Altmayer Pizzorno, Hamed Zamani. “A Symmetric Dual Encoding Dense Retrieval Framework for Knowledge-Intensive Visual Question Answering” SIGIR 2023.

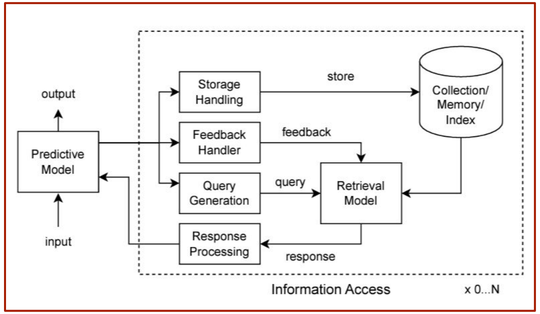
Joint Optimization of Retrieval and Predictive Models

Assumption:
Retrieval and predictive model parameters are optimized jointly.

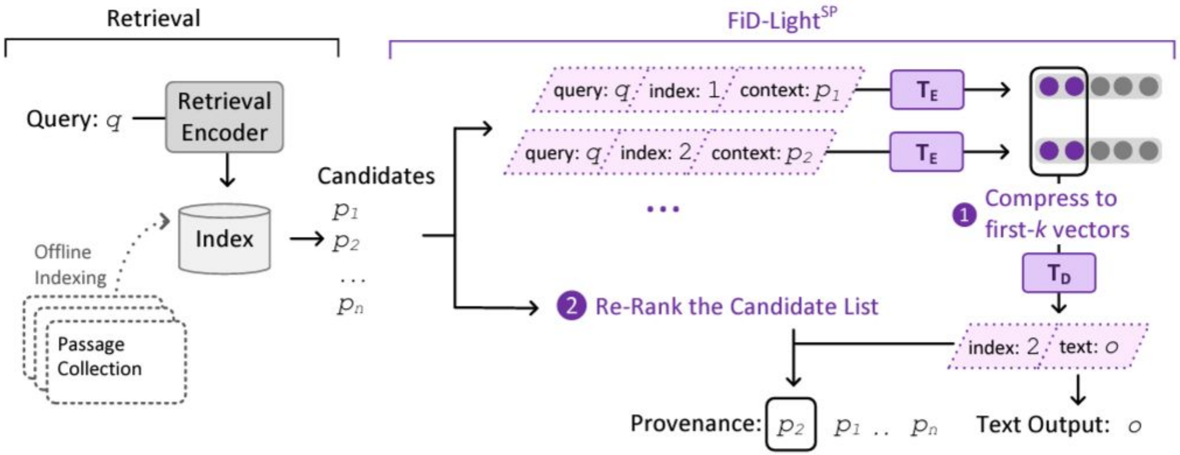
Examples:

- Joint multi-task optimization of retrieval and predictive models.
- End-to-end optimization.

$$\theta^*, \omega^* = \arg\min_{\theta,\omega} \frac{1}{|T|} \sum_{(x,y)\in T} L\left(f_\theta\left(x; g_\omega\right), y\right)$$



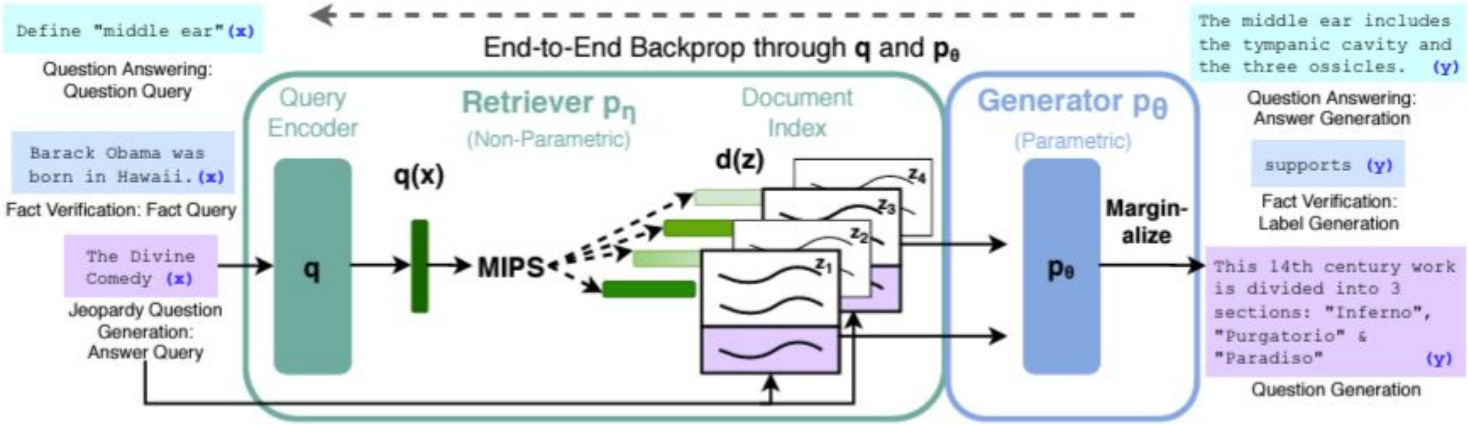
FiD-Light



Joint reranking and generation training

Sebastian Hofstatter, Jiecao Chen, Karthik Raman, Hamed Zamani. "FiD-Light: Efficient and Effective Retrieval-Augmented Text Generation" SIGIR 2023.

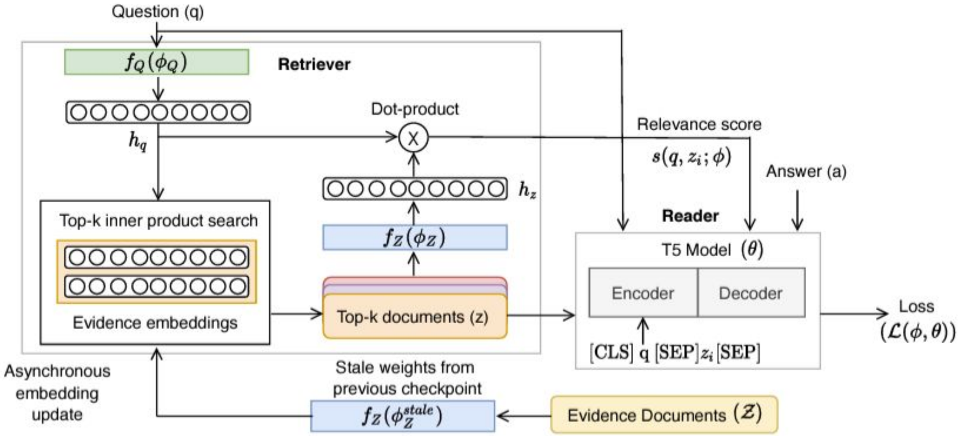
RAG



End-to-end RAG with marginalization assumption.

Patrick Lewis et al. “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks” NeurIPS 2020.

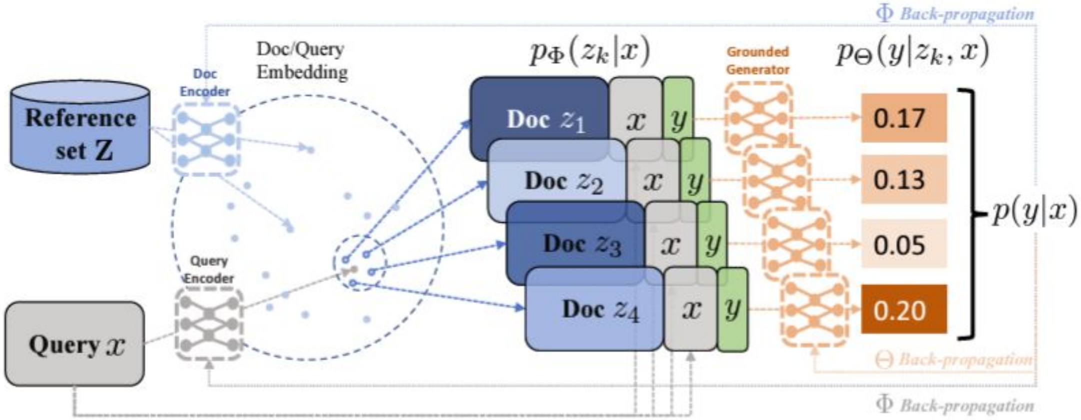
End-to-End Retriever-Reader Training



Asynchronous end-to-end RAG training.

Devendra Singh Sachan et al. “End-to-End Training of Neural Retrievers for Open-Domain Question Answering” ACL 2021.

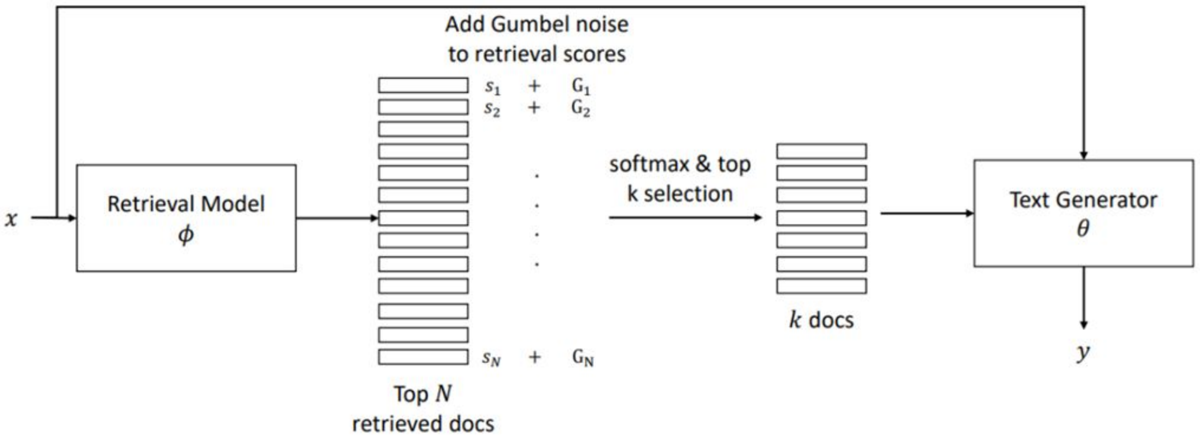
RetGen



End-to-end RAG with marginalization assumption.

Yizhe Zhang et al. “RetGen: A Joint framework for Retrieval and Grounded Text Generation Modeling” AAAI 2022.

Stochastic RAG



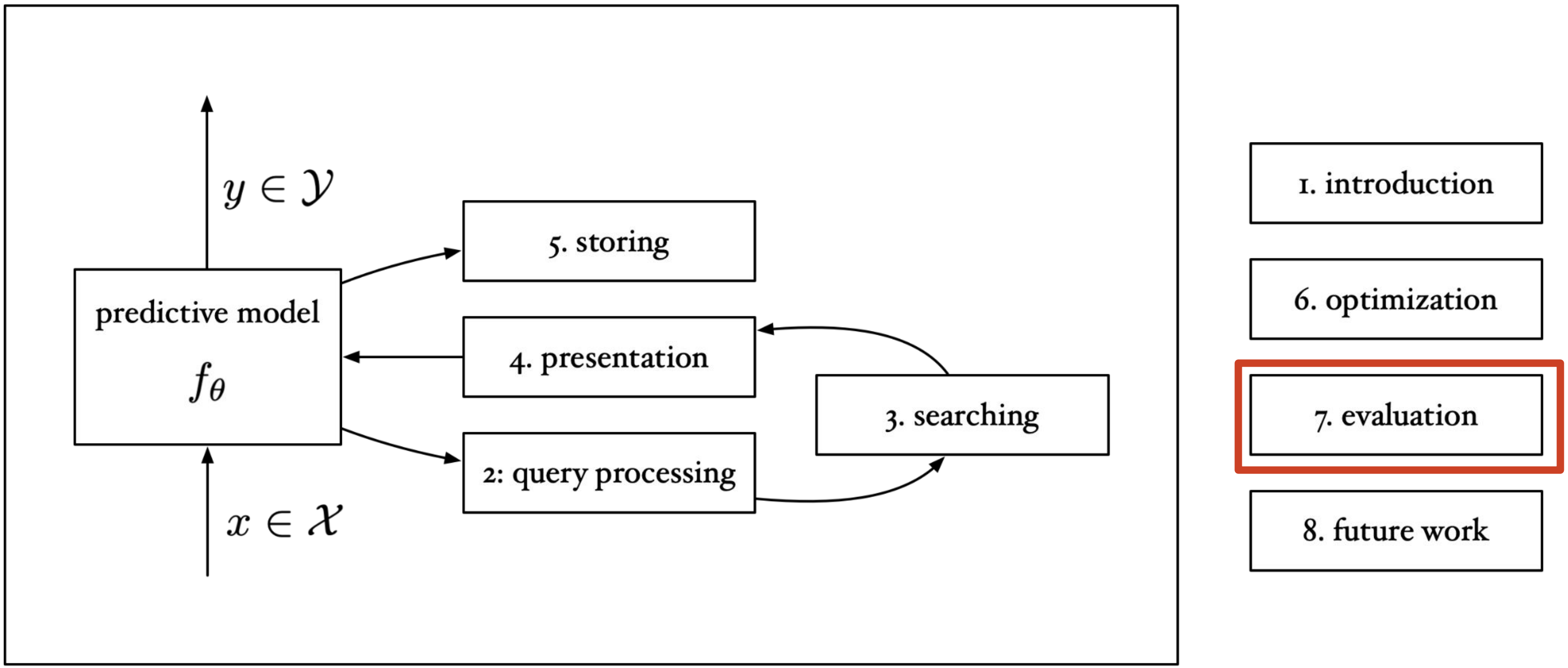
End-to-end RAG with stochastic ranking.

Hamed Zamani and Michael Bendersky "Stochastic RAG: End-to-End Retrieval-Augmented Generation through Expected Utility Maximization" SIGIR 2024.

questions?

Evaluation

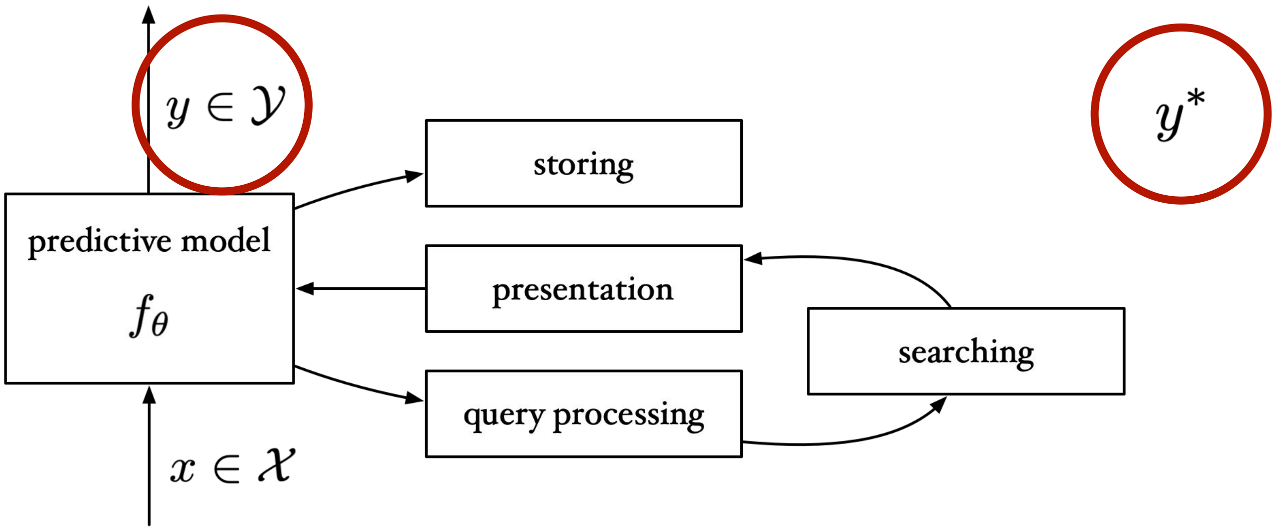
Overview


$y \in \mathcal{Y}$
predictive model
f_θ
$x \in \mathcal{X}$
5. storing
4. presentation
2: query processing
3. searching
1. introduction
6. optimization
7. evaluation
8. future work

evaluation

- need to understand whether a change to the system—including a full replacement—is better than keeping the status quo
- extrinsic evaluation: final performance of the predictive model using a task-specific metric.
- intrinsic evaluation: performance of a component of the system using a local measure of quality
    - can be an efficient approximation for an extrinsic evaluation.
    - can measure some independent value such as resource consumption.

extrinsic evaluation



extrinsic metrics

predictive model output

task labels

$$\mathbb{E}_{\langle x,y\rangle \sim \mathcal{E}}[\mu(f_\theta(x), y)] \approx \sum_{\langle x,y\rangle \in E} \mu(f_\theta(x), y)$$

task metric

- extrinsic evaluation computes the empirical estimate of the expected value of the task metric using labeled data.
- labeled data should be sampled according the target distribution

extrinsic metrics

- precision measures the relevant fraction of the output.
- recall measures the fraction of relevant claims in the output.
- back-translation measures the probability of an input derived from the output that are similar to the input.

$$\mu_P(\mathcal{C}_y, \mathcal{C}_{y^*}) = \frac{|\mathcal{C}_y \cap \mathcal{C}_{y^*}|}{|\mathcal{C}_y|}$$

$$\mu_R(\mathcal{C}_y, \mathcal{C}_{y^*}) = \frac{|\mathcal{C}_y \cap \mathcal{C}_{y^*}|}{|\mathcal{C}_{y^*}|}$$

$$\mu_B(\mathcal{X}_y, x) = \frac{|\mathcal{X}_y \cap \{x\}|}{|\mathcal{X}_y|}$$

$\mathcal{C}_y$ claims in prediction y

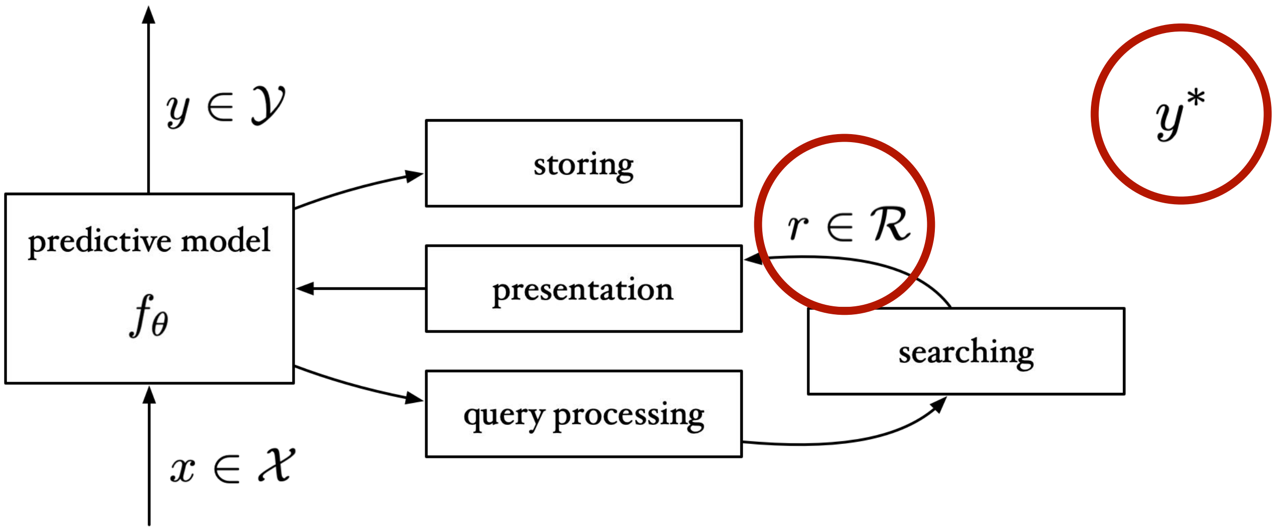
$\mathcal{C}_{y^*}$ claims in target y^*

$\mathcal{X}_y$ input derived from prediction y

D Ru, L Qiu, X Hu, T Zhang, P Shi, S Chang, C Jiayang, C Wang, S Sun, H Li, Z Zhang, B Wang, J Jiang, T He, Z Wang, P Liu, Y Zhang, Z Zhang. RAGChecker: a fine-grained framework for diagnosing retrieval-augmented generation. In The thirty-eight conference on neural information processing systems datasets and benchmarks track, 2024.

S Es, J James, L Espinosa Anke, S Schockaert. RAGAs: automated evaluation of retrieval augmented generation. In Nikolaos Aletras and Orphee De Clercq, editors, Proceedings of the 18th conference of the european chapter of the association for computational linguistics: system demonstrations, 150--158, 2024.

intrinsic evaluation: retrieval



retrieval metrics

retrieval output relevance labels

$$\mathbb{E}_{\langle x,y\rangle\sim\mathcal{E}}[\mu(f_\theta(x),y)] \propto \sum_{\langle x,\tilde{y}\rangle\in\tilde{E}} \tilde{\mu}(g_\omega(x),\tilde{y})$$

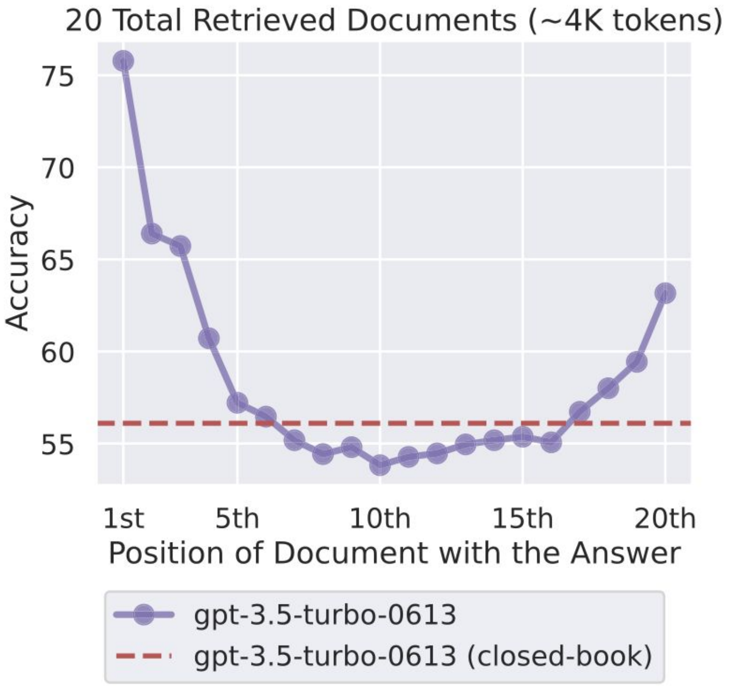
ranking metric

- classic retrieval metrics support human searchers and correlation with human task performance.
- can reuse existing metrics and new relevance judgments to measure component performance
  - relevance judgements should be task-specific

Alireza Salemi and Hamed Zamani. Towards a search engine for machines: unified ranking for multiple retrieval-augmented large language models. In Proceedings of the 47th international acm sigir conference on research and development in information retrieval, 2024.
Alireza Salemi and Hamed Zamani. Learning to rank for multiple retrieval-augmented models through iterative utility maximization. 2024.

retrieval metrics

- traditional retrieval metrics assume that position of relevant item is monotonically related to task performance
- REML models may not obey this!
- top and bottom of the ranking influence task performance!



Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: how language models use long contexts. Transactions of the Association for Computational Linguistics, 12:157–173, 02 2024.

retrieval metrics

optimal consumption of retrieval output

task labels

$$\mathbb{E}_{\langle x,y\rangle \sim \mathcal{E}}[\mu(f_\theta(x), y)] \approx \sum_{\langle x,y\rangle \in E} \mu(h(g_\omega(x)), y)$$

task metric

- alternatively, can transform the retrieval outputs into the same space as the task output and use the task metric
- assumes optimal consumer model

retrieval metrics

- for example, for claim-based evaluation, we can inspect the claims in the retrieval.

$$\mu_P(\mathcal{C}_r, \mathcal{C}_{y^*}) = \frac{|\mathcal{C}_r \cap \mathcal{C}_{y^*}|}{|\mathcal{C}_r|}$$

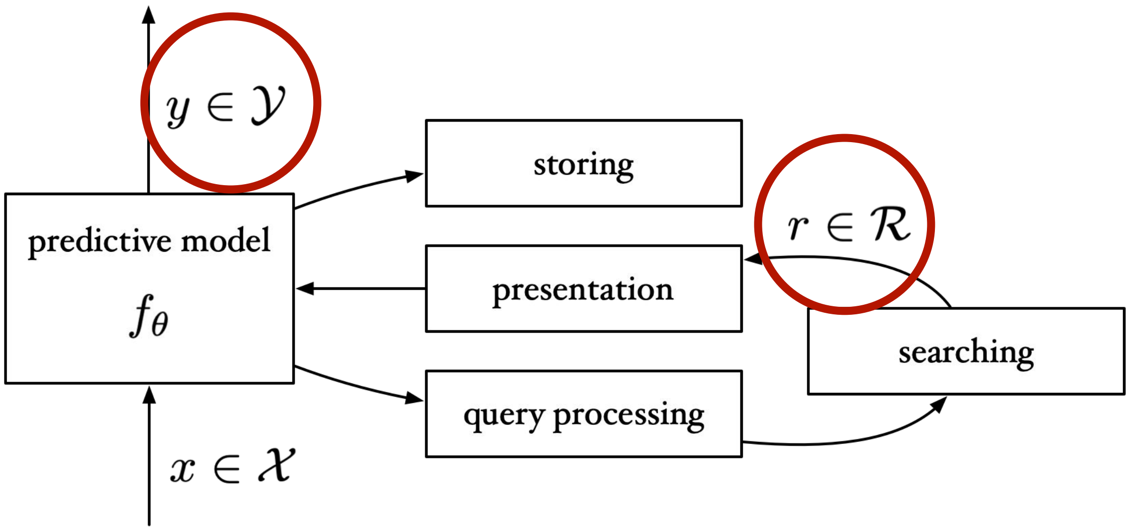
$$\mu_R(\mathcal{C}_r, \mathcal{C}_{y^*}) = \frac{|\mathcal{C}_r \cap \mathcal{C}_{y^*}|}{|\mathcal{C}_{y^*}|}$$

$\mathcal{C}_r$ claims in retrieval r

$\mathcal{C}_{y^*}$ claims in target y^*

D Ru, L Qiu, X Hu, T Zhang, P Shi, S Chang, C Jiayang, C Wang, S Sun, H Li, Z Zhang, B Wang, J Jiang, T He, Z Wang, P Liu, Y Zhang, Z Zhang. RAGChecker: a fine-grained framework for diagnosing retrieval-augmented generation. In The thirty-eight conference on neural information processing systems datasets and benchmarks track, 2024.

intrinsic evaluation: interaction



interaction metrics

retrieval performance

$$\sum_{\langle x,y\rangle \in E} \mu(h(g_\omega(x)), y)$$

predictive performance

$$\sum_{\langle x,y\rangle \in E} \mu(f(g_\omega(x)), y)$$

- in addition to evaluating the retrieval component in isolation, we can also study the relationship between the retrieval performance with in optimal consumption and retrieval performance with predictive model consumption

interaction metrics: faithfulness

- faithfulness measures the degree to which claims in output are supported by the retrieval.
- low faithfulness suggests that claims in the the output are not supported by the retrieval
- high faithfulness suggests that claims in the the output are supported by the retrieval

$$\mu_F(\mathcal{C}_y, \mathcal{C}_r) = \frac{|\mathcal{C}_y \cap \mathcal{C}_r|}{|\mathcal{C}_y|}$$

$\mathcal{C}_y$ claims in prediction y

$\mathcal{C}_r$ claims in retrieval r

D Ru, L Qiu, X Hu, T Zhang, P Shi, S Chang, C Jiayang, C Wang, S Sun, H Li, Z Zhang, B Wang, J Jiang, T He, Z Wang, P Liu, Y Zhang, Z Zhang. RAGChecker: a fine-grained framework for diagnosing retrieval-augmented generation. In The thirty-eight conference on neural information processing systems datasets and benchmarks track, 2024.

S Es, J James, L Espinosa Anke, S Schockaert. RAGAs: automated evaluation of retrieval augmented generation. In Nikolaos Aletras and Orphee De Clercq, editors, Proceedings of the 18th conference of the european chapter of the association for computational linguistics: system demonstrations, 150--158, 2024.

interaction metrics: utilization

- utilization measures the degree to which *relevant* claims in retrieval are present in the output.
- low utilization suggests that claims in the the retrieval are not present in the output
- high utilization suggests that claims in the the retrieval are present in the output

$$\mu_U(\mathcal{C}_y, \mathcal{C}_r, \mathcal{C}_{y^*}) = \frac{|\mathcal{C}_y^* \cap \mathcal{C}_r^*|}{|\mathcal{C}_r^*|}$$

$$\mathcal{C}_y^* = \mathcal{C}_y \cap \mathcal{C}_{y^*}$$
$$\mathcal{C}_r^* = \mathcal{C}_r \cap \mathcal{C}_{y^*}$$

D Ru, L Qiu, X Hu, T Zhang, P Shi, S Chang, C Jiayang, C Wang, S Sun, H Li, Z Zhang, B Wang, J Jiang, T He, Z Wang, P Liu, Y Zhang, Z Zhang. RAGChecker: a fine-grained framework for diagnosing retrieval-augmented generation. In The thirty-eight conference on neural information processing systems datasets and benchmarks track, 2024.

interaction metrics: sensitivity

- sensitivity measures the degree to which *nonrelevant* claims in output are present in the retrieval.
- low sensitivity suggests that nonrelevant claims in the the output might come from the retrieval.
- high sensitivity suggests that nonrelevant claims in the the output might not come from the retrieval.

$$\mu_S(\mathcal{C}_y, \mathcal{C}_r, \mathcal{C}_{y^*}) = \frac{|\mathcal{C}_y^- \cap \mathcal{C}_r^-|}{|\mathcal{C}_y|}$$

$$\mathcal{C}_y^- = \mathcal{C}_y \setminus \mathcal{C}_{y^*}$$
$$\mathcal{C}_r^- = \mathcal{C}_r \setminus \mathcal{C}_{y^*}$$

D Ru, L Qiu, X Hu, T Zhang, P Shi, S Chang, C Jiayang, C Wang, S Sun, H Li, Z Zhang, B Wang, J Jiang, T He, Z Wang, P Liu, Y Zhang, Z Zhang. RAGChecker: a fine-grained framework for diagnosing retrieval-augmented generation. In The thirty-eight conference on neural information processing systems datasets and benchmarks track, 2024.

interaction metrics: hallucination

- hallucination measures the degree to which *nonrelevant* claims in output are not present in the retrieval.
- low hallucination suggests that nonrelevant claims in the the output might come from the retrieval.
- high hallucination suggests that nonrelevant claims in the the output might not come from the retrieval.

$$\mu_H(\mathcal{C}_y, \mathcal{C}_r, \mathcal{C}_{y^*}) = \frac{|\mathcal{C}_y^- \setminus \mathcal{C}_r^-|}{|\mathcal{C}_y|}$$

$$\mathcal{C}_y^- = \mathcal{C}_y \setminus \mathcal{C}_{y^*}$$
$$\mathcal{C}_r^- = \mathcal{C}_r \setminus \mathcal{C}_{y^*}$$

D Ru, L Qiu, X Hu, T Zhang, P Shi, S Chang, C Jiayang, C Wang, S Sun, H Li, Z Zhang, B Wang, J Jiang, T He, Z Wang, P Liu, Y Zhang, Z Zhang. RAGChecker: a fine-grained framework for diagnosing retrieval-augmented generation. In The thirty-eight conference on neural information processing systems datasets and benchmarks track, 2024.

interaction metrics: knowledge

- knowledge measures the degree to which *relevant* claims in output are not present in the retrieval.
- low knowledge suggests that relevant claims in the the output might come from the retrieval.
- high knowledge suggests that relevant claims in the the output might not come from the retrieval.

$$\mu_K(\mathcal{C}_y, \mathcal{C}_r, \mathcal{C}_{y^*}) = \frac{|\mathcal{C}_y^* \setminus \mathcal{C}_r|}{|\mathcal{C}_y|}$$

$$\mathcal{C}_y^* = \mathcal{C}_y \cap \mathcal{C}_{y^*}$$

D Ru, L Qiu, X Hu, T Zhang, P Shi, S Chang, C Jiayang, C Wang, S Sun, H Li, Z Zhang, B Wang, J Jiang, T He, Z Wang, P Liu, Y Zhang, Z Zhang. RAGChecker: a fine-grained framework for diagnosing retrieval-augmented generation. In The thirty-eight conference on neural information processing systems datasets and benchmarks track, 2024.

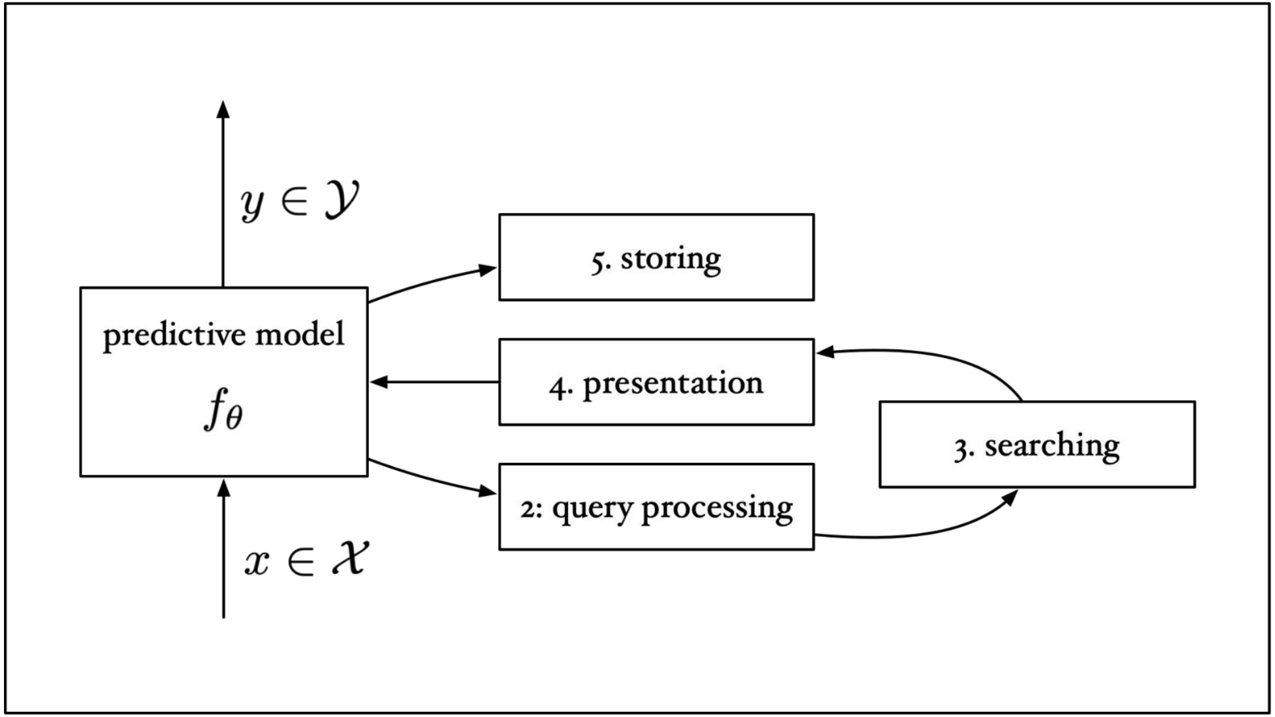
interaction metrics: knowledge

End-to-End Evaluation (§8.3)		
Task	**Datasets**	**Corpus**
Entity Related QA	PopQA[141], EntityQuestions[197]	Wikipedia
Current Events Related QA	RealtimeQA[97]	News Websites
Science Related Multiple-choice QA	ARC [28]	Subset of Web
Science Related QA	Qasper[34]	Scientific Articles
Story Related Long-form QA	NarrativeQA[110]	A Long Story
Query-based Summarization	QMSum[269]	A Meeting Transcript
Personalized Classification and Generation	LaMP[186]	A User Profile
End-to-End & Retrieval Evaluation (§8.3)		
Open-domain Multi-Hop QA	2WikiMultiHopQA[71], HotpotQA[165, 248]	Wikipedia
Open-domain Short-form QA	Natural Questions[113, 165], TriviaQA[93, 165], StrategyQA[55]	Wikipedia
Open-domain Long-form QA	ELI5[48, 165], ASQA[54]	Wikipedia
Dialogue Generation	Wizard of Wikipedia[38, 165]	Wikipedia
Slot Filling	ZeroShot RE[122, 165], T-REx[44, 165]	Wikipedia
Entity Linking	AIDA CoNLL-YAGO[72, 165], WNED-WIKI/CWEB [1, 165]	Wikipedia
Fact Verification	FEVER[165, 212]	Wikipedia
Open-domain Visual QA	OK-VQA[143, 172]	Wikipedia
Open-domain Visual QA	FVQA[221]	A Supporting Facts Set

questions?

Future Directions & Conclusion

Overview



1. introduction

6. optimization

7. evaluation

8. future work

querying

- Query with Instruction. developing transformation functions for query generation that produce task and query-specific instructions alongside the query can significantly enhance the retrieval model's capacity to fulfill the requirements of the predictive model.
- Retrieval System Aware Query Generation. tailoring query generation to the retrieval model to ensure that queries meet the model's unique requirements, improving retrieval effectiveness.
- Dissociated Interface between Retrieval and Predictive Model. training both retrieval and predictive models jointly to learn a shared hidden space, enabling more effective communication.

presentation and consumption

- Task-Specialized Presentation and Consumption. improve document representation specific to the task.
- Proactive REML. providing retrieval results relevant to the predictive model context without an explicit query (i.e., recommendation-enhanced ML).

storing

- Shared Storage. supporting multiple predictive models sharing a single collection and pushing relevant content to shared storage.
- Storage Staleness. adaptive storage mechanisms that can dynamically align with retriever updates, ensuring data integrity and model efficiency.

optimization

- Effective and Efficient End-to-End Optimization. understanding of exploration and exploitation of information items provided by the information access system is required.
- Learning from Online and Session-based Feedback. Using the feedback provided by the predictive model during an inference session and its users to adjust the REML output is critical to develop effective interactive REML systems.
- Efficient Approximation of Feedback for Optimization. developing efficient and accurate feedback approximations could substantially reduce the cost of REML training.
- One Information Access and Multiple Predictive Models. optimizing information access components that provide service to multiple predictive models, aggregating and calibrating feedback across predictive models, and “personalizing” the retrieval result lists for each predictive model are important future directions.

optimization

- Effective and Efficient End-to-End Optimization. understanding of exploration and exploitation of information items provided by the information access system is required.
- Learning from Online and Session-based Feedback. Using the feedback provided by the predictive model during an inference session and its users to adjust the REML output is critical to develop effective interactive REML systems.
- Efficient Approximation of Feedback for Optimization. developing efficient and accurate feedback approximations could substantially reduce the cost of REML training.
- One Information Access and Multiple Predictive Models. optimizing information access components that provide service to multiple predictive models, aggregating and calibrating feedback across predictive models, and “personalizing” the retrieval result lists for each predictive model are important future directions.

evaluation

- Formalizing Component Evaluation. need to develop more formal methods for sampling contexts, labels, and metrics for extrinsic and intrinsic evaluation metrics

conclusion

- REML provides a formal framework for studying retrieval as a component in modern ML systems
- suggests multiple avenues for existing IR methods to advance ML
  - much existing ML research is reproducing classic IR results
- suggests multiple avenues for new ML architecture to advance IR
  - much existing IR research is focusing on existing IR paradigm

questions?