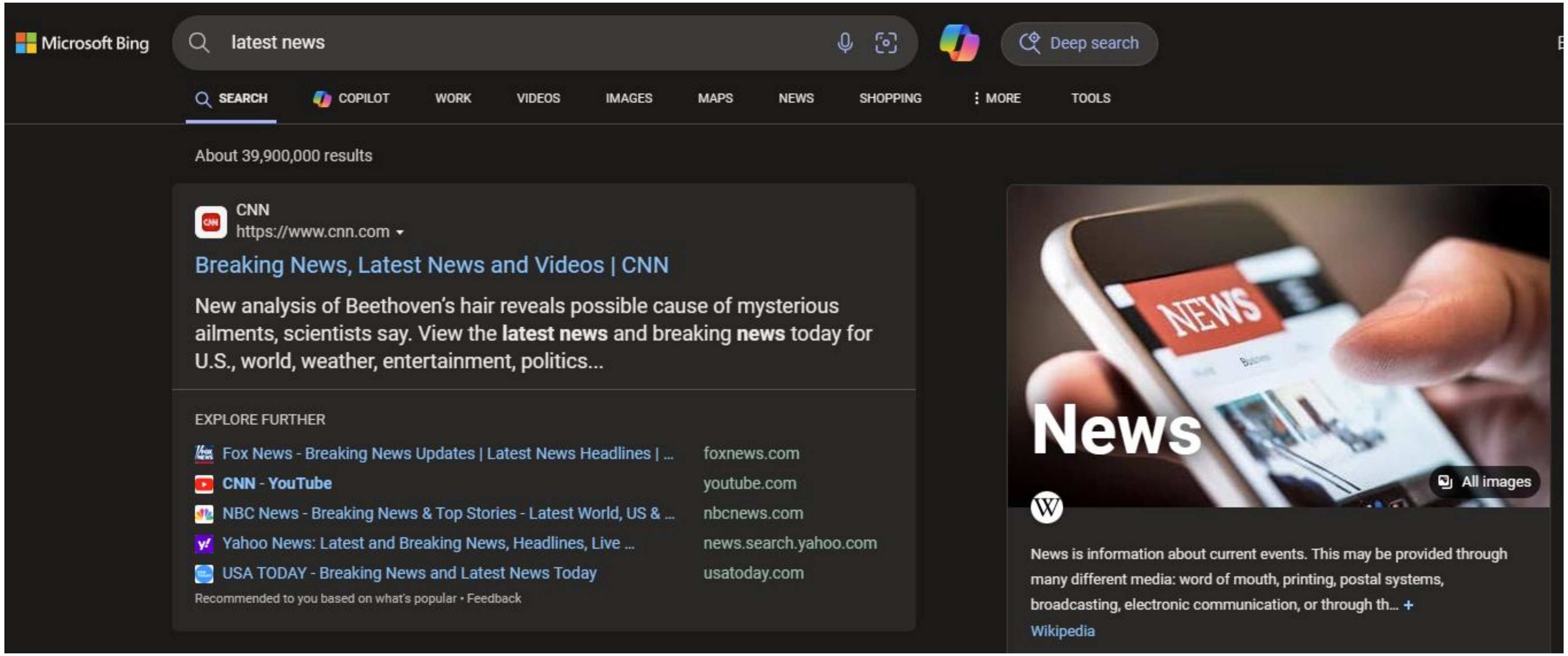
Microsoft

When Search Engine Meets LLMs: Opportunities and Challenges

Liang Wang
Microsoft Research
2024/5

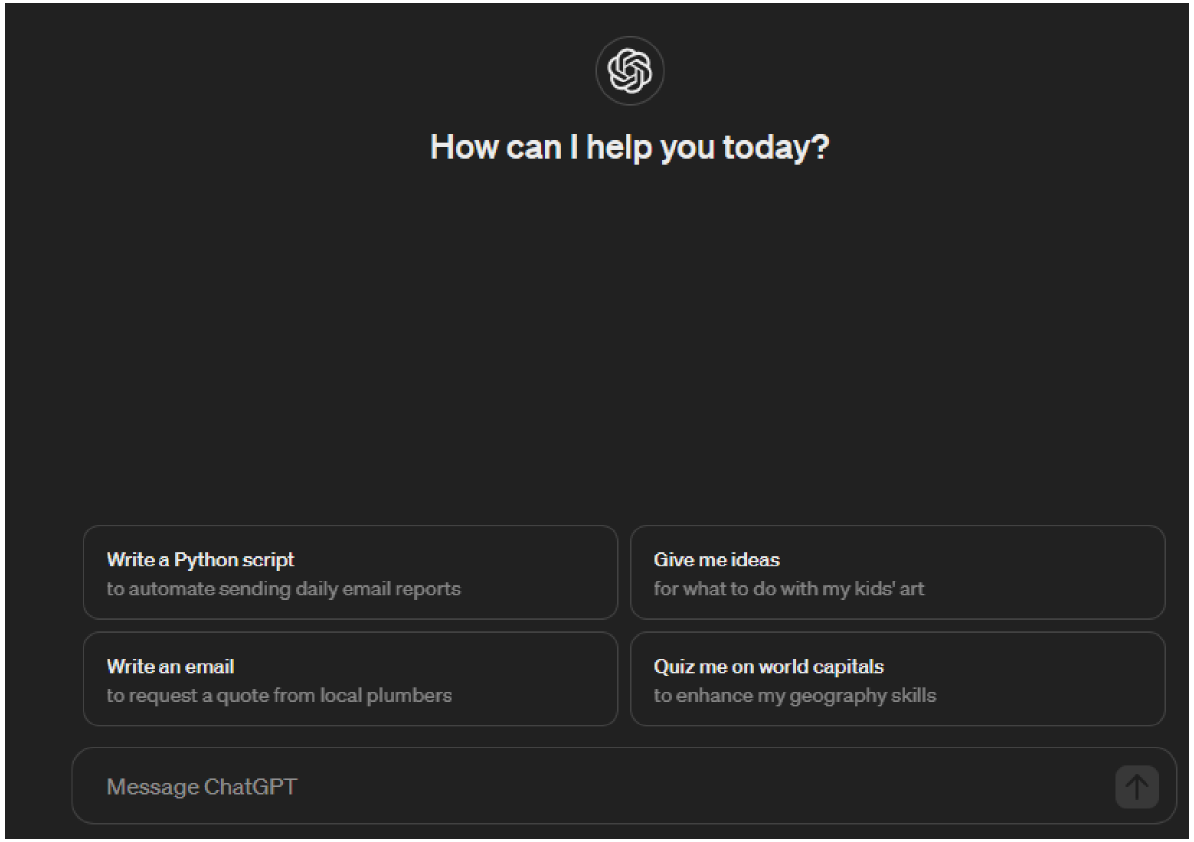
Search Engines

- Given a user query, provide a list of relevant web pages.



Large Language Models (LLMs)

- Especially decoder-only LLMs



https://openai.com/index/chatgpt/

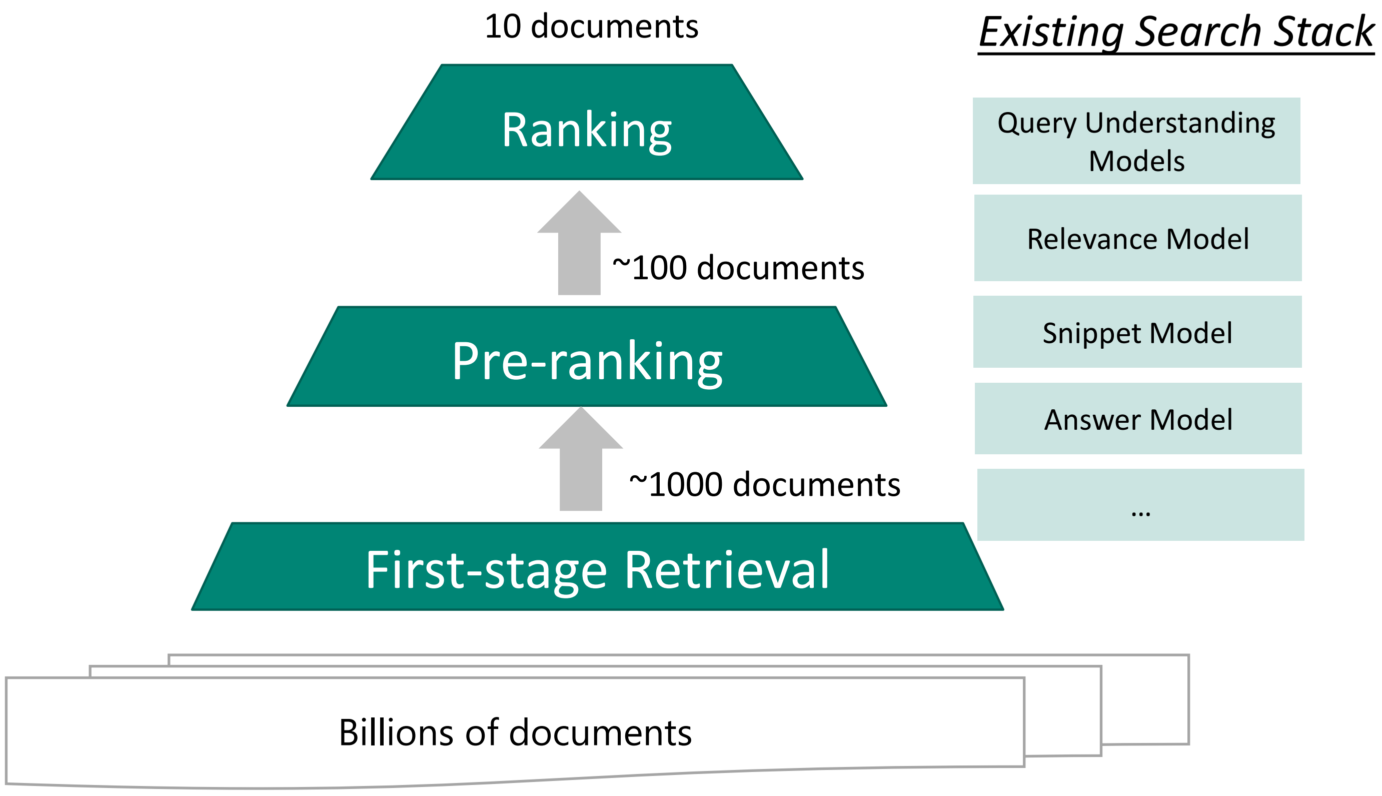
When Search Engines Meet LLMs

- Part 1: how can LLMs help in existing search stacks?
- Part 2: how can search engine augment LLMs?
- Part 3: will LLMs make search engines obsolete?

How can LLMs help in existing search stacks?

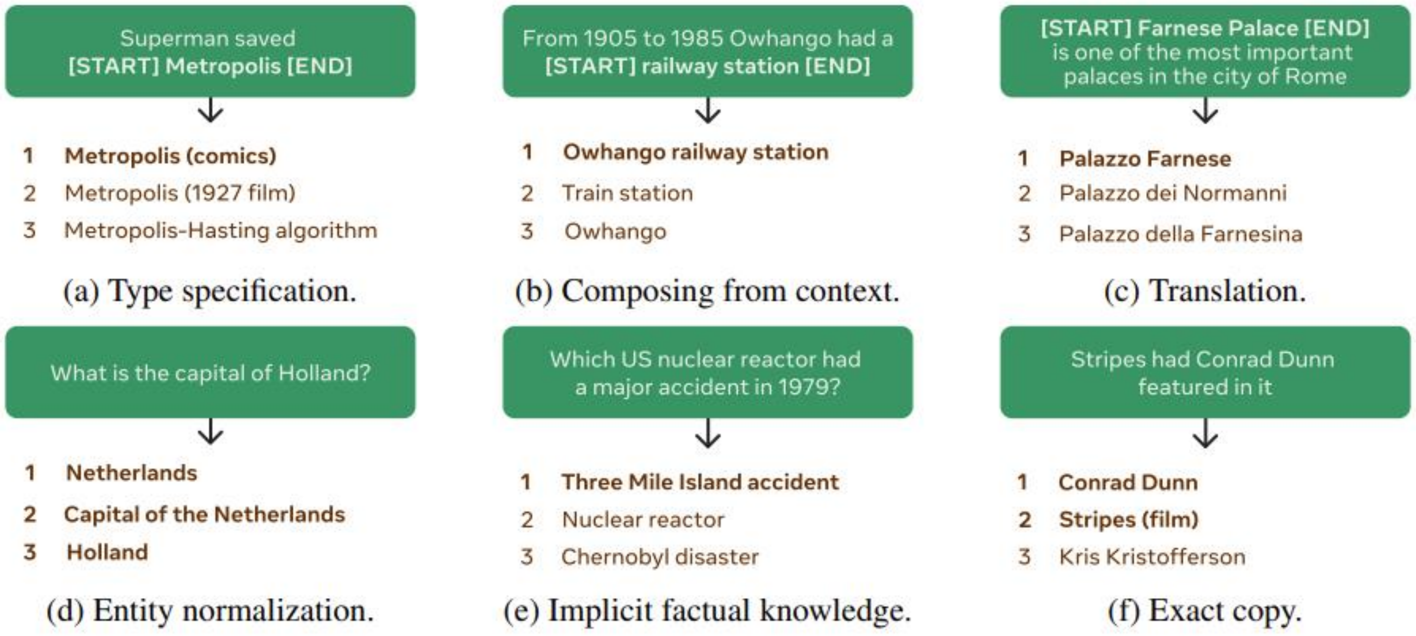
Search Stack

- Retrieval and multi-stage ranking
- Multiple independent and customized components



Generative Retrieval

- Modeling first-stage retrieval as text generation

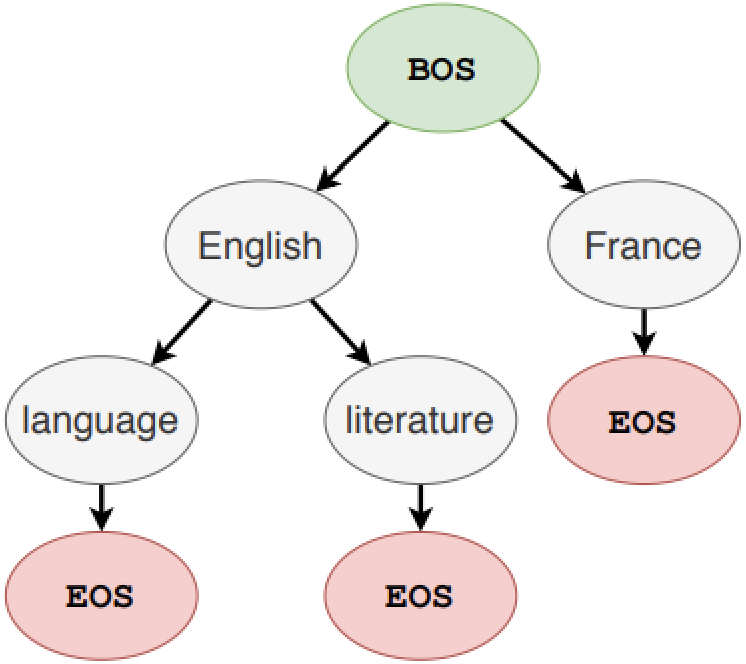


(a) Type specification. (b) Composing from context. (c) Translation. (d) Entity normalization. (e) Implicit factual knowledge. (f) Exact copy.

Autoregressive Entity Retrieval, 2020

Generative Retrieval

- Constrained decoding with Trie tree



Autoregressive Entity Retrieval, 2020

Why Generative Retrieval?

- Consistent with LM pre-training objectives
- No need for maintaining vector index
  - But need to maintain an additional prefix trie
- No need for designing hard negative sampling strategy

Autoregressive Entity Retrieval, 2020

Differentiable Search Index (DSI)

- Indexing task: document token sequences to identifiers
- Retrieval task: query to document identifiers

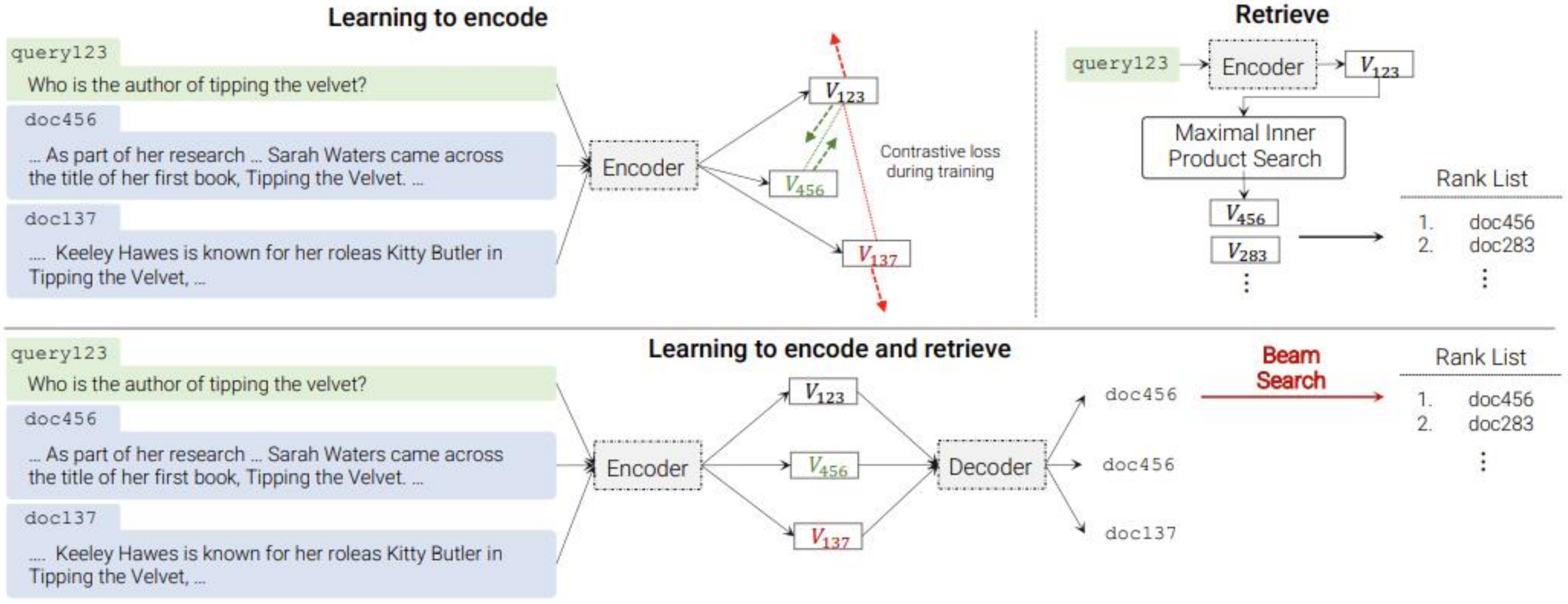
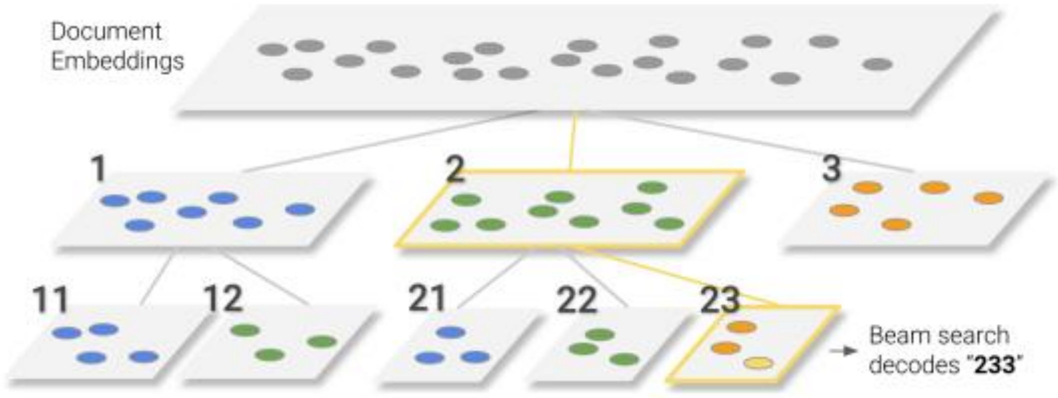


Figure 1: Comparison of dual encoders (top) to differentiable search index (bottom).

Transformer Memory as a Differentiable Search Index, 2022

Differentiable Search Index (DSI)

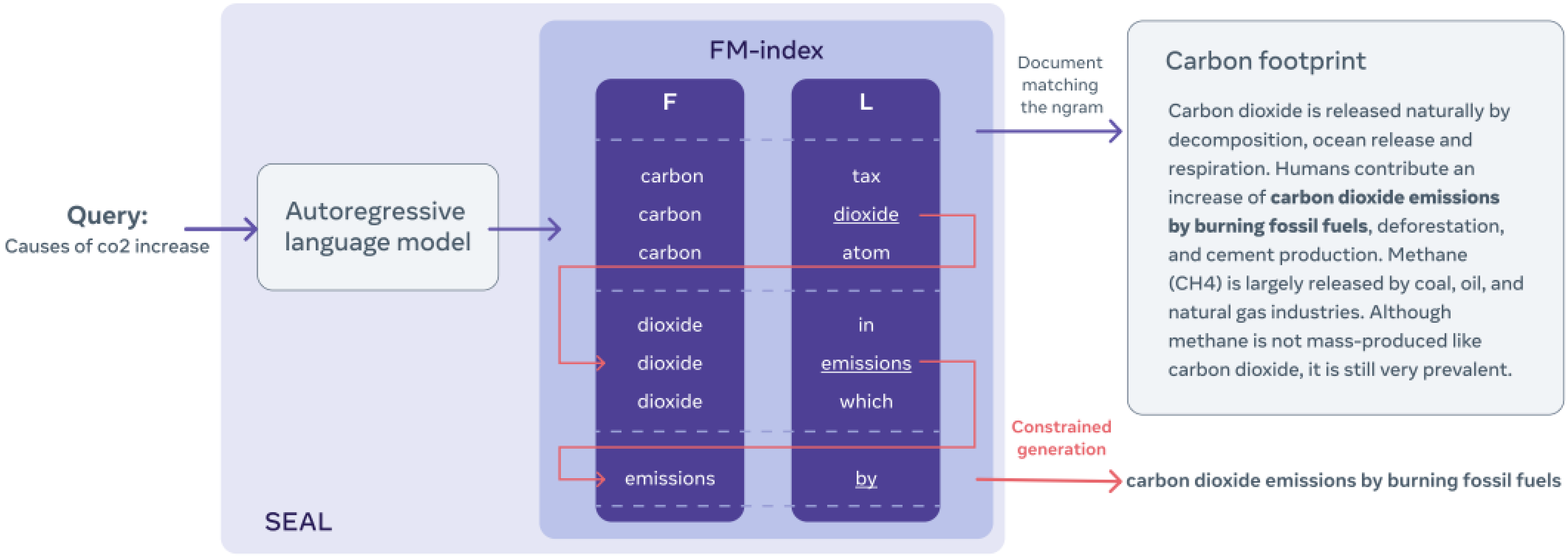
- Generating semantically structured identifiers



Transformer Memory as a Differentiable Search Index, 2022

Generative Retrieval - SEAL

- Use n-grams as identifiers instead of IDs



Autoregressive Search Engines: Generating Substrings as Document Identifiers, 2022

Generative Retrieval - SEAL

- Training tasks
  - Unsupervised samples: random span -> random span
  - Query -> sampled 10-gram
- Decoding with FM-Index
  - A suffix array that efficiently finds possible successors in O(|V| log|V|)

Autoregressive Search Engines: Generating Substrings as Document Identifiers, 2022

Limitations of Generative Retrieval

- Low learning efficiency
- Fail to scale to medium-size corpus

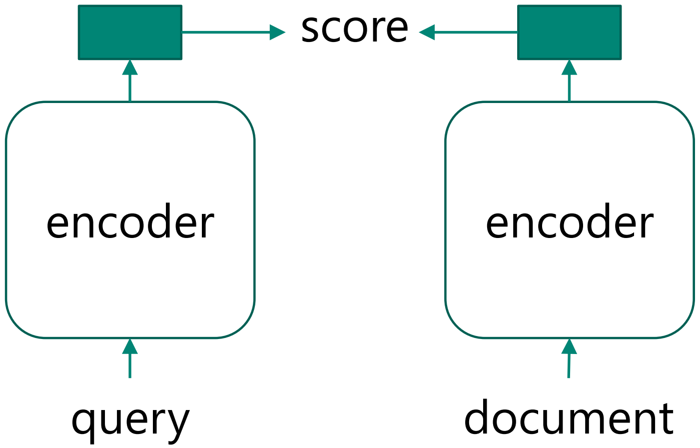
	Model	MSMarco100k			MSMarco1M			MSMarcoFULL		
		At.	Nv.	Sm.	At.	Nv.	Sm.	At.	Nv.	Sm.
Baselines										
BM25		-	65.3	-	-	41.3	-	-	18.4	-
BM25 (w/ doc2query–T5)		-	80.4	-	-	56.6	-	-	27.2	-
GTR-Base		-	83.2	-	-	60.7	-	-	34.8	-
Ours										
(1a)	Labeled Queries (No Indexing)	0.0	1.1	0.0	0.0	0.5	0.0	0.0	0.0	0.0
(2a)	FirstP/DaQ + Labeled Queries (DSI)	0.0	23.9	19.2	2.1	12.4	7.4	0.0	7.5	3.1
(3b)	FirstP/DaQ + D2Q + Labeled Queries	79.2	77.7	76.8	53.3	48.2	47.1	14.2	**13.2**	6.4
(4a)	3b + PAWA (w/ 2D Semantic IDs)	-	-	77.1	-	-	50.2	-	-	9.0
(5)	4a + Consistency Loss (NCI)	-	-	77.1	-	-	50.2	-	-	9.1
(6b)	D2Q only	**80.3**	**78.7**	**78.5**	**55.8**	**55.4**	54.0	**24.2**	**13.3**	11.8
($4a'$)	6b + PAWA (w/ 2D Semantic IDs)	-	-	78.2	-	-	**54.1**	-	-	**17.3**
($4b'$)	6b + Constrained Decoding	-	-	**78.6**	-	-	54.0	-	-	12.0
($5'$)	6b + PAWA (w/ 2D Semantic IDs) + Constrained Decoding	-	-	78.3	-	-	**54.2**	-	-	**17.4**

How Does Generative Retrieval Scale to Millions of Passages?, 2023

Caveats on the Evaluation Protocol

- Where does the retrieval corpus come from?
  - Most successful examples are based on Wikipedia
- What is the size of the retrieval corpus?
  - Most good numbers are based on sub-sampled corpus (e.g., so-called “MS-MARCO 100k”)

LLMs for Embedding-based Dense Retrieval



- Biencoder retriever
  - Matching in a latent vector space
  - Efficient, scalable, overcomes the lexical mismatch problem of BM25

How to improve dense retrievers?

- Late interaction with multiple vectors (ColBERT[1])
  - Cons: increased storage cost and more complicated ANN search algorithm
- Knowledge distillation from re-ranker to retriever (RocketQA[2])
- Iterative hard negative mining (ANCE[3] / AR2[4])
- Continual pre-training specialized for retrieval (E5[5] / SimLM[6] / RetroMAE[7])

1. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT, 2020
2. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering, 2020
3. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval, 2020
4. Adversarial Retriever-Ranker for dense text retrieval, 2021
5. Text Embeddings by Weakly-Supervised Contrastive Pre-training, 2022
6. SimLM: Pre-training with Representation Bottleneck for Dense Passage Retrieval, 2022
7. RetroMAE: Pre-Training Retrieval-oriented Language Models Via Masked Auto-Encoder, 2022

Hard negative mining

- Contrastive learning is sensitive to the quality of hard negatives
  - Hard negatives can be mined based on BM25 or trained dense retrievers

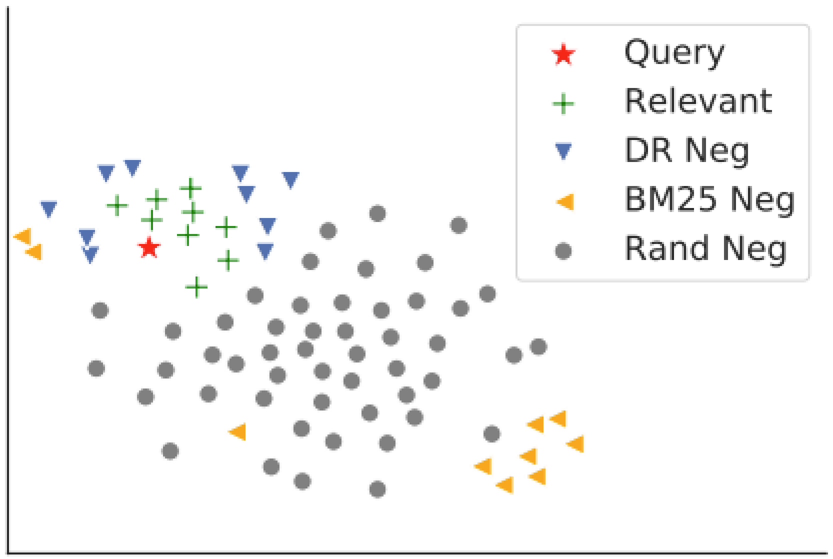


Figure from [Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval](), 2020

Why does hard negatives matter

- Separate between real cat and other objects

Anything with two ears

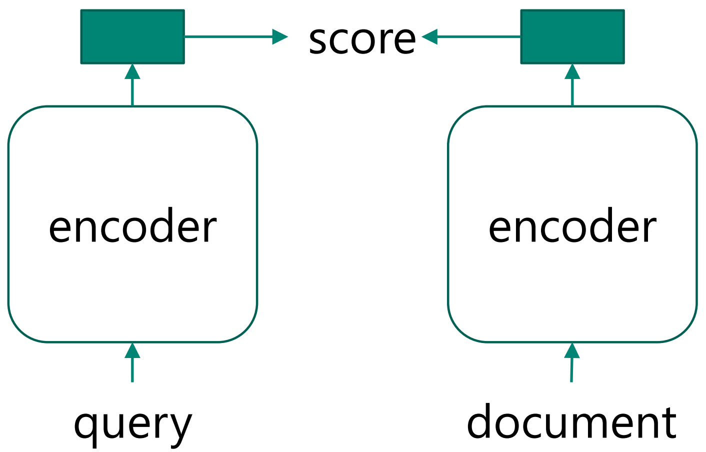
easy negative

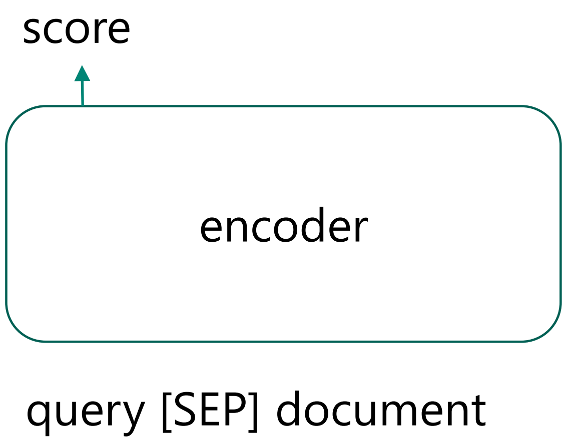
hard negative

Hmm, cats can not walk with two legs

Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval, 2020

Knowledge distillation from re-ranker





- Biencoder retriever
  - Matching in a latent vector space
  - Efficient, scalable, overcomes the lexical mismatch problem of BM25

- Cross-encoder re-ranker
  - Pros: Full interaction between query and document
  - Cons: Not scalable

Knowledge distillation from re-ranker

- Re-ranker as a teacher model
  - KL divergence between the re-ranker and the student retriever

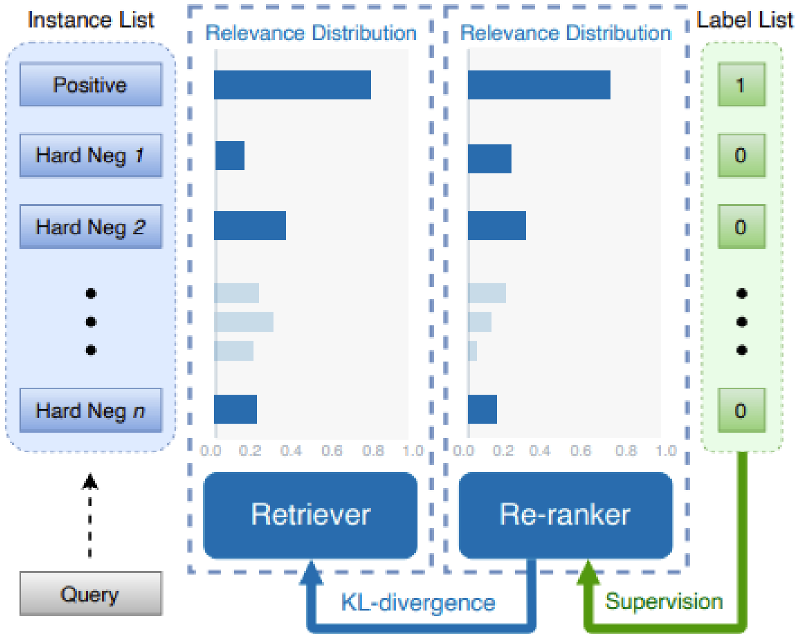
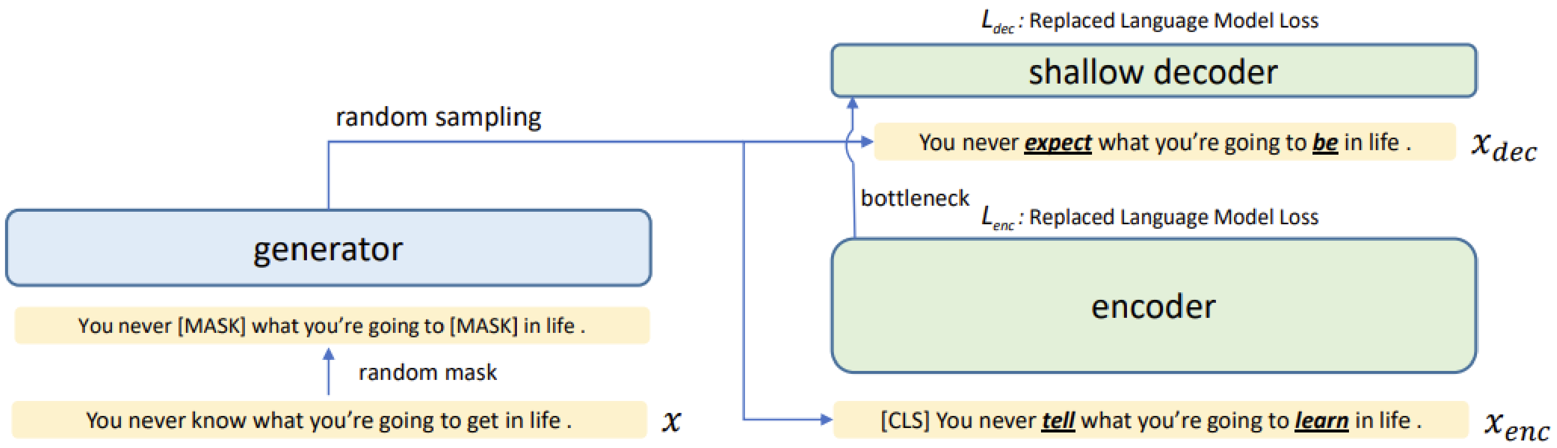


Figure from RocketQAv2: A Joint Training Method for Dense Passage Retrieval and Passage Re-ranking, 2021

Continual pre-training

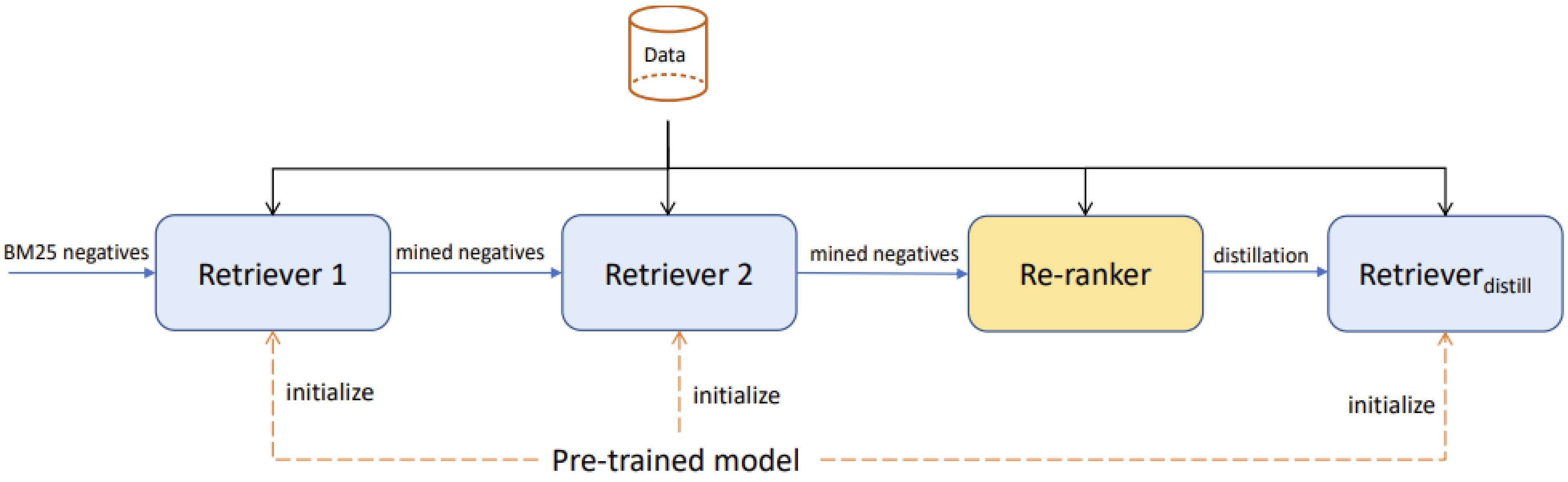
- Representation bottleneck
  - Learn to compress input into a vector with self-supervised learning
  - Pre-training on target corpus



[SimLM: Pre-training with Representation Bottleneck for Dense Passage Retrieval](), 2022

Continual pre-training

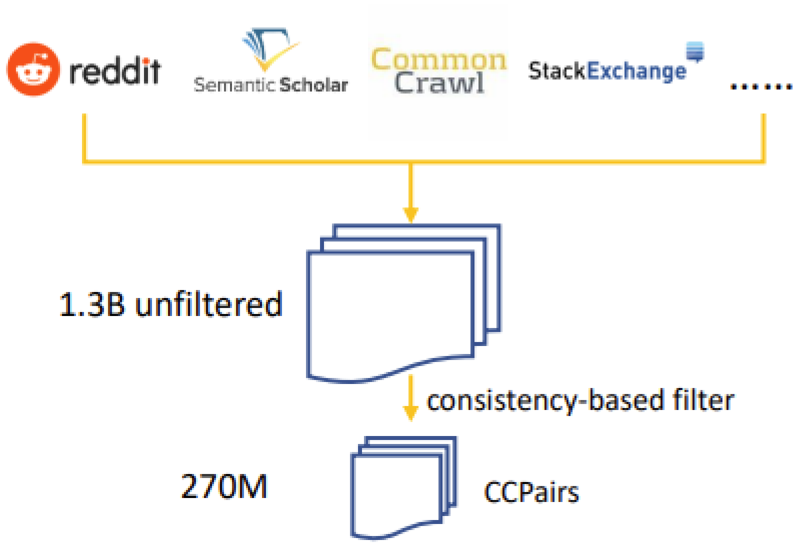
- Combining them all

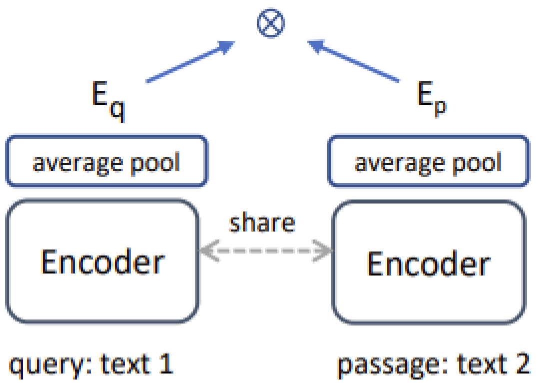


SimLM: Pre-training with Representation Bottleneck for Dense Passage Retrieval, 2022

Continual pre-training

- Weakly-supervised contrastive pre-training (E5 Text Embeddings)
  - Pre-train with billions of text pairs from various domains
  - Better out-of-domain performance





Text Embeddings by Weakly-Supervised Contrastive Pre-training, 2022

The Importance of Large Batch Size

- Larger batch size will introduce more in-batch negatives
  - E5 uses batch size 32k for pre-training
- Implementation
  - Naïve gradient accumulation will not work
  - All gather with multi-gpu training

Text Embeddings by Weakly-Supervised Contrastive Pre-training, 2022

GradCache

- How to apply large batch size when GPU memory is limited?
  - Key observation: gradients w.r.t embedding vectors does not depend on model parameters

$$\mathcal{L} = -\frac{1}{|S|} \sum_{s_i \in S} \log \frac{exp(f(s_i)^\top g(t_{r_i})/\tau)}{\sum_{t_j \in T} exp(f(s_i)^\intercal g(t_j)/\tau)}$$

$$\frac{\partial \mathcal{L}}{\partial f(s_i)} = -\frac{1}{|S|} \left(g(t_{r_i}) - \sum_{t_j \in T} p_{ij} g(t_j) \right),$$

$$\frac{\partial \mathcal{L}}{\partial g(t_j)} = -\frac{1}{|S|} \left(\epsilon_j - \sum_{s_i \in S} p_{ij} f(s_i) \right),$$

where

$$\epsilon_j = \begin{cases} f(s_k) & \text{if } \exists\, k \text{ s.t. } r_k = j \\ 0 & \text{otherwise} \end{cases}$$

Scaling deep contrastive learning batch size under memory limited setup, 2021

GradCache

- Step 1: Graph-less forward
  - Save embedding vectors but not other intermediate activations
- Step 2: Representation gradient computation and caching
- Step 3: Sub-batch gradient accumulation
- Step 4: Run optimization step

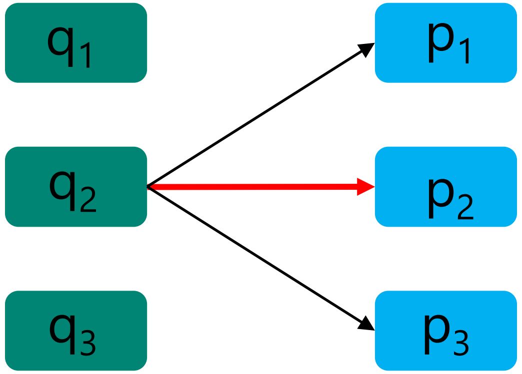
$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \Theta} &= \sum_{\hat{S}_j \in \mathbb{S}} \sum_{s_i \in \hat{S}_j} \frac{\partial \mathcal{L}}{\partial f(s_i)} \frac{\partial f(s_i)}{\partial \Theta} \\
&= \sum_{\hat{S}_j \in \mathbb{S}} \sum_{s_i \in \hat{S}_j} \mathbf{u}_i \frac{\partial f(s_i)}{\partial \Theta}
\end{aligned}
$$

Scaling deep contrastive learning batch size under memory limited setup, 2021

Same-tower Negatives

- Four groups of contrastive pairs

$$\mathcal{L}_c = \frac{\exp(\mathtt{sim}(q_i, p_i)/\tau)}{\sum_{j \in \mathcal{B}} \exp(\mathtt{sim}(q_i, p_j)/\tau)},$$

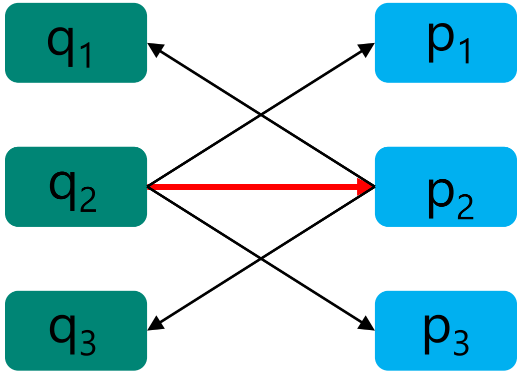


SamToNe: Improving Contrastive Loss for Dual Encoder Retrieval Models with Same Tower Negatives, 2023

Same-tower Negatives

- Four groups of contrastive pairs

$$\mathcal{L}_c = \frac{\exp(\mathtt{sim}(q_i, p_i)/\tau)}{\sum_{j \in \mathcal{B}} \exp(\mathtt{sim}(q_i, p_j)/\tau)},$$

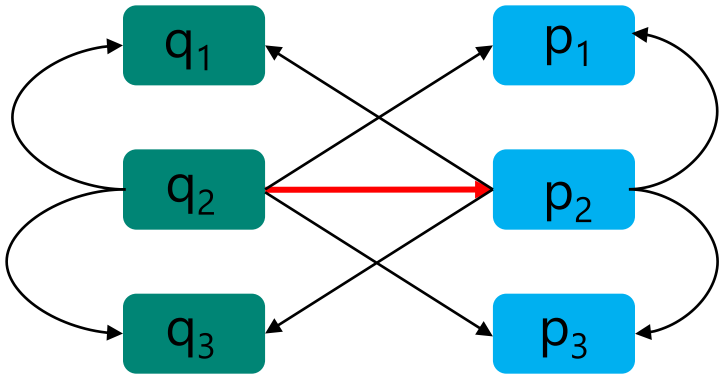


SamToNe: Improving Contrastive Loss for Dual Encoder Retrieval Models with Same Tower Negatives, 2023

Same-tower Negatives

- Four groups of contrastive pairs

$$\mathcal{L}_S = \frac{e^{\mathtt{sim}(q_i,p_i)/\tau}}{\sum_{j\in\mathcal{B}} e^{\mathtt{sim}(q_i,p_j)/\tau} + \sum_{j\in\mathcal{B},j\neq i} e^{\mathtt{sim}(q_i,q_j)/\tau}},$$



SamToNe: Improving Contrastive Loss for Dual Encoder Retrieval Models with Same Tower Negatives, 2023

Decoder-only vs Encoder-only Embeddings

- A common conception: *bi-directional encoders make more sense for IR.*

https://github.com/UKPLab/sentence-transformers/issues/1363

The IR Problem

- What is the most fundamental issue for IR?

The IR Problem

- What is the most fundamental issue for IR?

It is Representation Learning

- What is the most important lesson for representation learning?

It is Scaling Law

⬇

Large Language Models

Large Language Models (LLMs)

- Decoder-only language models by scaling up model and data sizes
  - Capabilities: in-context learning / instruction following

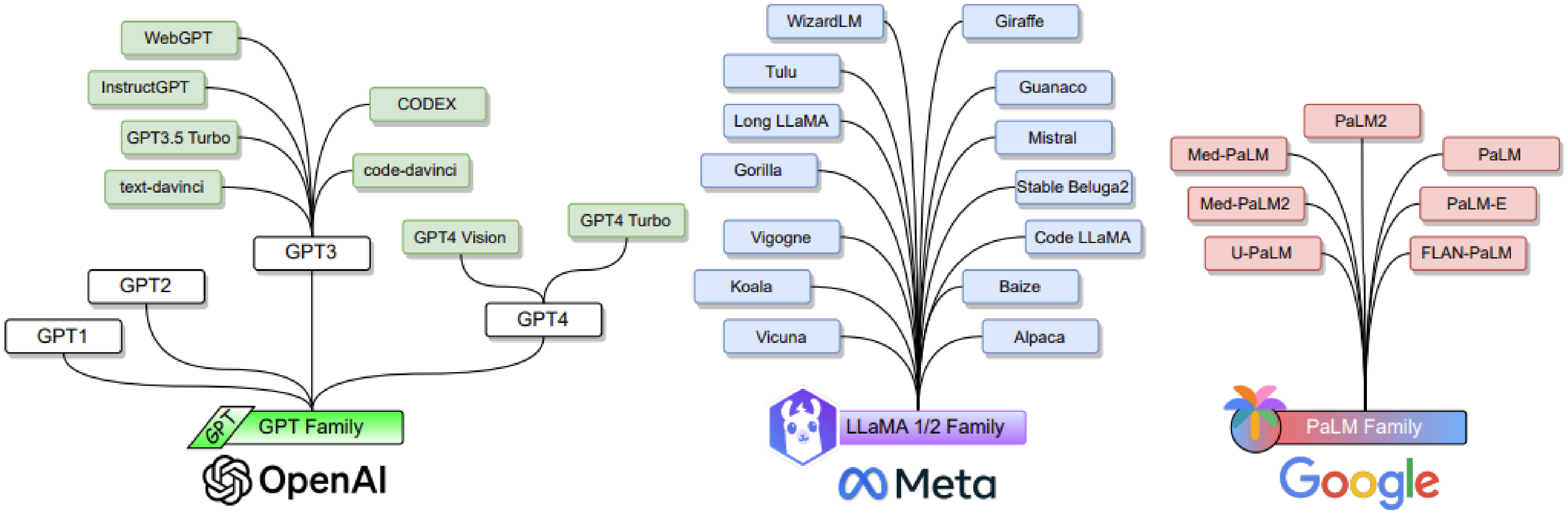


Figure from [Large Language Models: A Survey](), 2024

LLMs + IR

- RankLLaMA
  - train retriever and re-ranker by initializing from LLaMA-2

	Model size	Source prev.	top-k	DEV MRR@10	DEV R@1k	DL19 nDCG@10	DL20 nDCG@10
Retrieval							
BM25 (Lin et al., 2021)	-	-	$\|C\|$	18.4	85.3	50.6	48.0
ANCE (Xiong et al., 2021)	125M	-	$\|C\|$	33.0	95.9	64.5	64.6
CoCondenser (Gao and Callan, 2022b)	110M	-	$\|C\|$	38.2	98.4	71.7	68.4
GTR-base (Ni et al., 2022)	110M	-	$\|C\|$	36.6	98.3	-	-
GTR-XXL (Ni et al., 2022)	4.8B	-	$\|C\|$	38.8	99.0	-	-
OpenAI Ada2 (Neelakantan et al., 2022)	?	-	$\|C\|$	34.4	98.6	70.4	67.6
bi-SimLM (Wang et al., 2023)	110M	-	$\|C\|$	39.1	98.6	69.8	69.2
RepLLaMA	7B	-	$\|C\|$	**41.2**	**99.4**	**74.3**	**72.1**
Reranking							
monoBERT (Nogueira et al., 2019)	110M	BM25	1000	37.2	85.3	72.3	72.2
cross-SimLM (Wang et al., 2023)	110M	bi-SimLM	200	43.7	98.7	74.6	72.7
RankT5 (Zhuang et al., 2023)	220M	GTR	1000	43.4	98.3	-	-
RankLLaMA	7B	RepLLaMA	200	44.9	99.4	75.6	77.4
RankLLaMA-13B	13B	RepLLaMA	200	**45.2**	**99.4**	**76.0**	**77.9**

This number is very hard to move

[Fine-Tuning LLaMA for Multi-Stage Text Retrieval](), 2023

LLMs + IR

- A common conception: *bi-directional encoders make more sense for IR.*

Unlike generation, retrieval models do not need to be decoder-only ✓

Decoder-only models can not be good retrieval models ✗

Decoder-only models underperform bi-directional encoders at comparable model size ✓

SGPT

- Based on GPT-Neo and GPT-J from 125M to 5.8B
- Weighted mean pooling

$$v = \sum_{i=1}^{S} w_i h_i \quad \text{where} \quad w_i = \frac{i}{\sum_{i=1}^{S} i}$$

SGPT: GPT sentence embeddings for semantic search, 2022

SGPT

- Strong results on OOD settings (BEIR benchmark)

Training (→)	Unsupervised		U. + U.	Unsupervised + Supervised			Unsupervised + Unsupervised + Supervised			
Model (→) **Dataset (↓)**	[41] **BM25**	SGPT-CE **SGPT-6.1B**	[27] **cpt-text-L♥**	[44] **BM25+CE♣**	[17] **TAS-B♣**	SGPT-BE **SGPT-5.8B**	[20] **Contriever♠**	[29] **GTR-XXL♦**	OpenAI Embeddings [27] **cpt-text-L♥**	 **cpt-text-XL♥**
MS MARCO	0.228	0.290		0.413‡	0.408‡	0.399‡		**0.442**‡		
TREC-COVID	0.688	0.791	0.427	0.757	0.481	**0.873**	0.596	0.501	0.562	0.649
BioASQ	0.488	**0.547**		0.523	0.383	0.413		0.324		
NFCorpus	0.306	0.347	0.369	0.350	0.319	0.362	0.328	0.342	0.380	**0.407**
NQ	0.326	0.401		0.533	0.463	0.524	0.498	**0.568**		
HotpotQA	0.602	0.699	0.543	**0.707**	0.584	0.593	0.638	0.599	0.648	0.688
FiQA-2018	0.254	0.401	0.397	0.347	0.300	0.372	0.329	0.467	0.452	**0.512**
Signal-1M (RT)	0.330	0.323		**0.338**	0.289	0.267		0.273		
TREC-NEWS	0.405	0.466		0.431	0.377	**0.481**		0.346		
Robust04	0.425	0.480		0.475	0.427	**0.514**		0.506		
ArguAna	0.472	0.286	0.392	0.311	0.429	0.514	0.446	**0.540**	0.469	0.435
Touché-2020	**0.347**	0.234	0.228	0.271	0.162	0.254	0.230	0.256	0.309	0.291
CQADupStack	0.326	**0.420**		0.370	0.314	0.381	0.345	0.399		
Quora	0.808	0.794	0.687	0.825	0.835	0.846	0.865	**0.892**	0.677	0.638
DBPedia	0.320	0.370	0.312	0.409	0.384	0.399	0.413	0.408	0.412	**0.432**
SCIDOCS	0.165	0.196		0.166	0.149	**0.197**	0.165	0.161	0.177†	
FEVER	0.649	0.725	0.638	**0.819**	0.700	0.783	0.758	0.740	0.756	0.775
Climate-FEVER	0.186	0.161	0.161	0.253	0.228	**0.305**	0.237	0.267	0.194	0.223
SciFact	0.611	0.682	0.712	0.688	0.643	0.747	0.677	0.662	0.744	**0.754**
Sub-Average	0.477	0.499	0.442	0.520	0.460	**0.550**	0.502	0.516	0.509	0.528
Average	0.428	0.462		0.476	0.395	**0.490**		0.458		
Best on	1	2	0	3	0	**5**	0	3	0	4

SGPT: GPT sentence embeddings for semantic search, 2022

E5 Mistral

- Diverse synthetic data
- Better foundation model
- Instruction-informed embeddings

Brainstorm a list of potentially useful text retrieval tasks.
Here are a few examples for your reference:
- Provided a scientific claim as query, retrieve documents that help verify or refute the claim.
- Search for documents that answers a FAQ-style query on children's nutrition.

Please adhere to the following guidelines:
- Specify what the query is, and what the desired documents are.
- Each retrieval task should cover a wide range of queries, and should not be too specific.

Your output should always be a python list of strings only, with about 20 elements, and each element corresponds to a distinct retrieval task in one sentence. Do not explain yourself or output anything else. Be creative!

["Retrieve company's financial reports for a given stock ticker symbol.",
"Given a book name as a query, retrieve reviews, ratings and summaries of that book.",
"Search for scientific research papers supporting a medical diagnosis for a specified disease.“
… *(omitted for space)*]

new session

You have been assigned a retrieval task: *{task}*
Your mission is to write one text retrieval example for this task in JSON format. The JSON object must contain the following keys:
- **"user_query"**: a string, a random user search query specified by the retrieval task.
- **"positive_document"**: a string, a relevant document for the user query.
- **"hard_negative_document"**: a string, a hard negative document that only appears relevant to the query.

Please adhere to the following guidelines:
- The "user_query" should be *{query_type}*, *{query_length}*, *{clarity}*, and diverse in topic.
- All documents should be at least *{num_words}* words long.
- Both the query and documents should be in *{language}*.

… *(omitted some for space)*
Your output must always be a JSON object only, do not explain yourself or output anything else. Be creative!

{**"user_query":** "How to use Microsoft Power BI for data analysis",
"positive_document": "Microsoft Power BI is a sophisticated tool that requires time and practice to master. In this tutorial, we'll show you how to navigate Power BI … *(omitted)* ",
“hard_negative_document”: “Excel is an incredibly powerful tool for managing and analyzing large amounts of data. Our tutorial series focuses on how you…*(omitted)*” }

Improving text embeddings with large language models, 2024

E5 Mistral

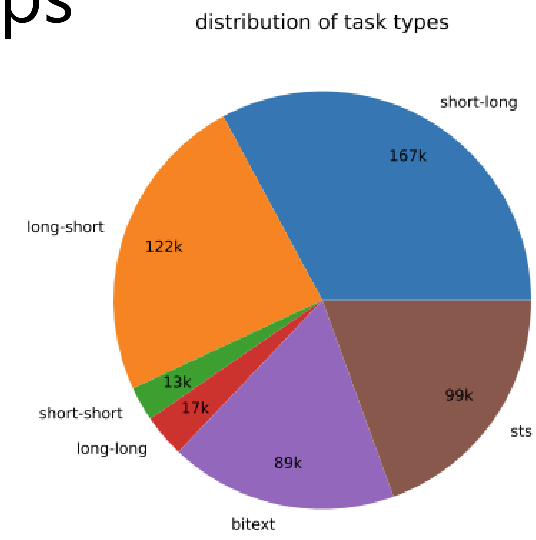
- Diverse synthetic data by prompting GPT-4
  - Asymmetric matching: short-long, long-short, short-short, long-long
  - Symmetric matching: semantic similarity, bitext retrieval
- Instruction-informed embeddings

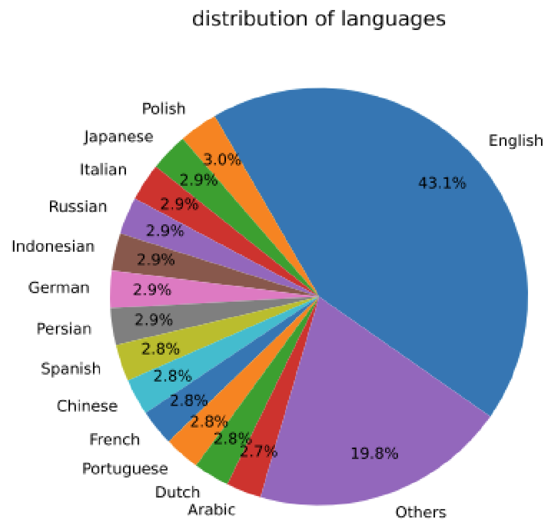
$$q_{\text{inst}}^{+} = \text{Instruct: \{task\_definition\}} \setminus n \text{ Query: } \{q^{+}\}$$

Improving text embeddings with large language models, 2024

E5 Mistral

- Fine-tuning takes less than 1k steps
  - No contrastive pre-training



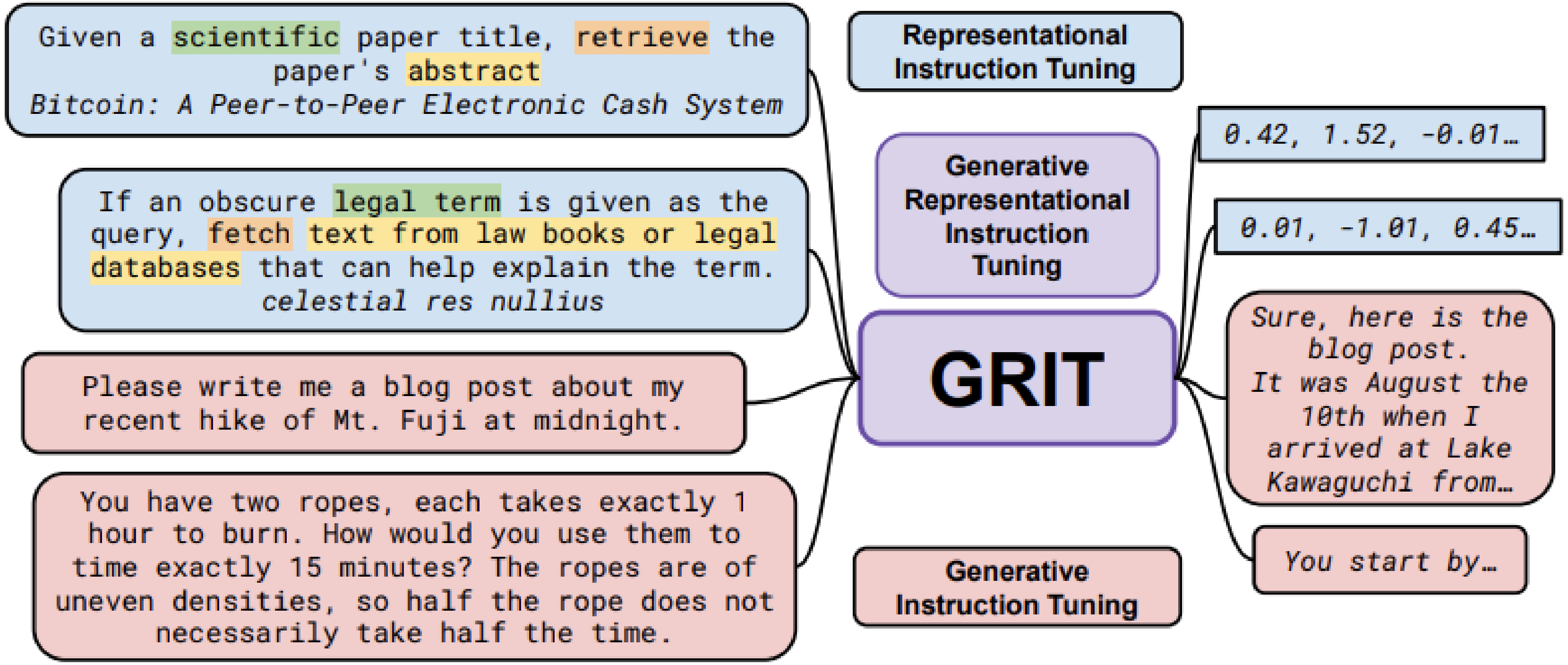


Model	BEIR Retrieval (15 datasets)	MTEB Average (56 datasets)
OpenAI Ada-002	49.3	61.0
Cohere-embed-english-v3.0	55.0	64.5
voyage-lite-01-instruct	55.6	64.5
UAE-Large-V1 [22]	54.7	64.6
$E5_{mistral\text{-}7b}$ + full data	**56.9**	**66.6**

Improving text embeddings with large language models, 2024

GritLM: Unifying Text Generation and Embeddings

- Two sides of the same coin



Generative representational instruction tuning, 2024

GritLM: Unifying Text Generation and Embeddings

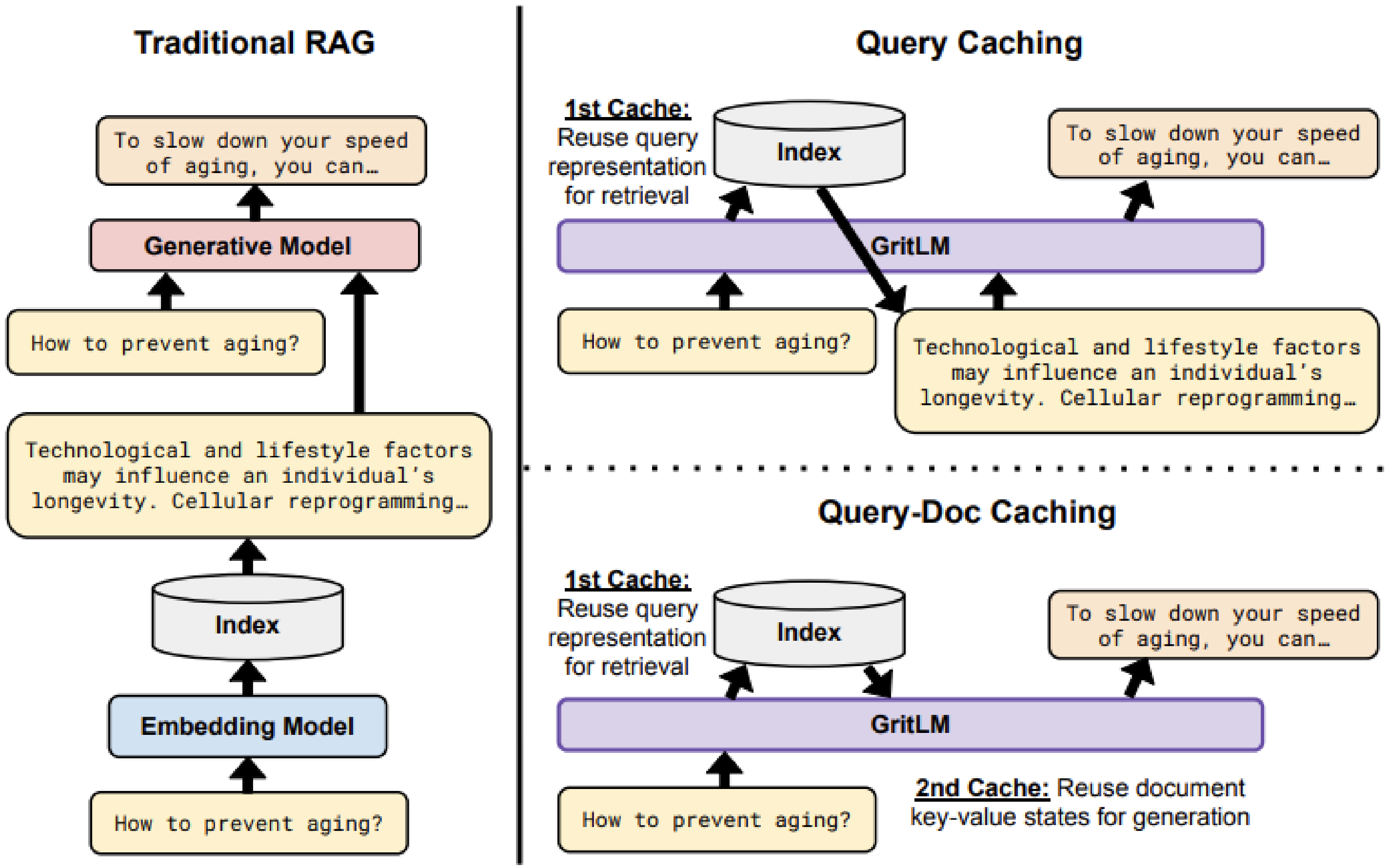
- Mutual enhancement

Dataset (→) Setup (→) Metric (→)	MMLU 0 FS EM	GSM8K 8 FS, CoT EM	BBH 3 FS, CoT EM	TyDi QA 1 FS, GP F1	HumanEval 0 FS pass@1	Alpaca 0 FS, 1.0 % Win	Avg.
Proprietary models♥							
GPT-4-0613	81.4	95.0	89.1	65.2	86.6†	91.2	84.8
Other Open Models♥							
GPT-J 6B	27.7	2.5	30.2	9.4	9.8	0.0	13.3
SGPT BE 5.8B	24.4	1.0	0.0	22.8	0.0	0.0	8.0
Zephyr 7B β	58.6	28.0	44.9	23.7	28.5	85.8	44.9
Llama 2 7B	41.8	12.0	39.3	51.2	12.8♦	0.0	26.2
Llama 2 13B	52.0	25.0	48.9	56.5	18.3♦	0.0	33.5
Llama 2 70B	64.5	55.5	66.0	**62.6**	29.9♦	0.0	46.4
Llama 2 Chat 13B	53.2	9.0	40.3	32.1	19.6†	91.4	40.9
Llama 2 Chat 70B	60.9	59.0	49.0	44.4	34.3†	<u>94.5</u>	57.0
Tülu 2 7B	50.4	34.0	48.5	46.4	24.5†	73.9	46.3
Tülu 2 13B	55.4	46.0	49.5	53.2	31.4	78.9	52.4
Tülu 2 70B	<u>67.3</u>	**73.0**	<u>68.4</u>	53.6	41.6	86.6	<u>65.1</u>
Mistral 7B	60.1	44.5	55.6	55.8	30.5	0.0	41.1
Mistral 7B Instruct	53.0	36.0	38.5	27.8	34.0	75.3	44.1
Mixtral 8x7B Instruct	**68.4**	<u>65.0</u>	55.9	24.3	**53.5**	**94.8**	60.3
GRITLM							
Emb.-only 7B	23.5	1.0	0.0	21.0	0.0	0.0	7.6
Gen.-only 7B	57.5	52.0	55.4	56.6	34.5	75.4	55.2
GRITLM 7B	57.6	57.5	54.8	55.4	32.8	74.8	55.5
GRITLM 8X7B	66.7	61.5	**70.2**	<u>58.2</u>	<u>53.4</u>	84.0	**65.7**

Generative representational instruction tuning, 2024

GritLM: Unifying Text Generation and Embeddings

- Potential to re-use KV cache for RAG



Generative representational instruction tuning, 2024

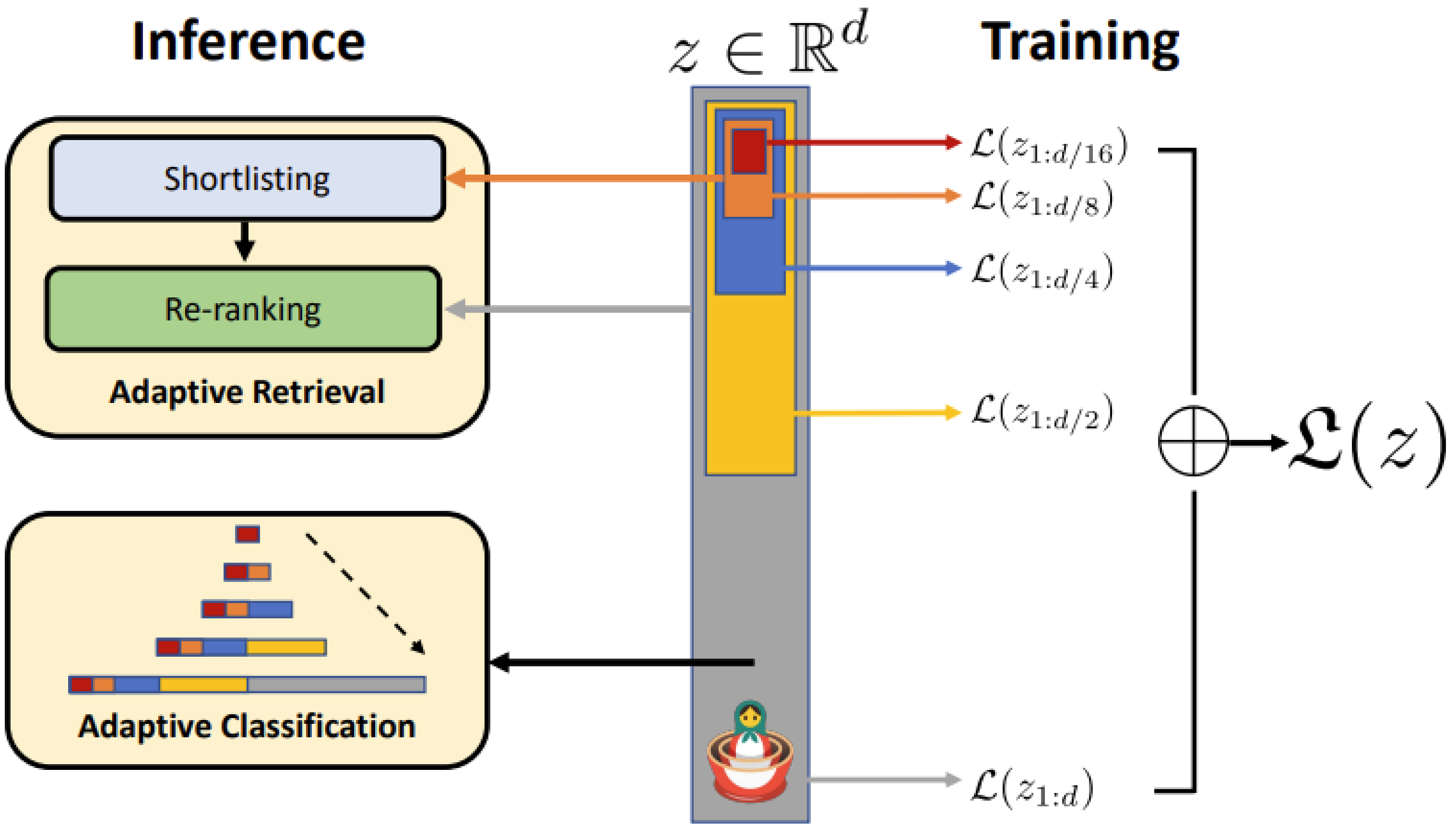
Challenges to Deploy LLM-based Embeddings

- Inference cost
  - Lower precision inference
  - Better kernel implementation: FlashAttention-2 etc.
  - Distillation to smaller models
- Storage cost due to high embedding dimensions
  - Vector Quantization

FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning, 2023
The Faiss library, 2024, 2024

Matryoshka Embeddings

- Flexible embedding dimension within one model



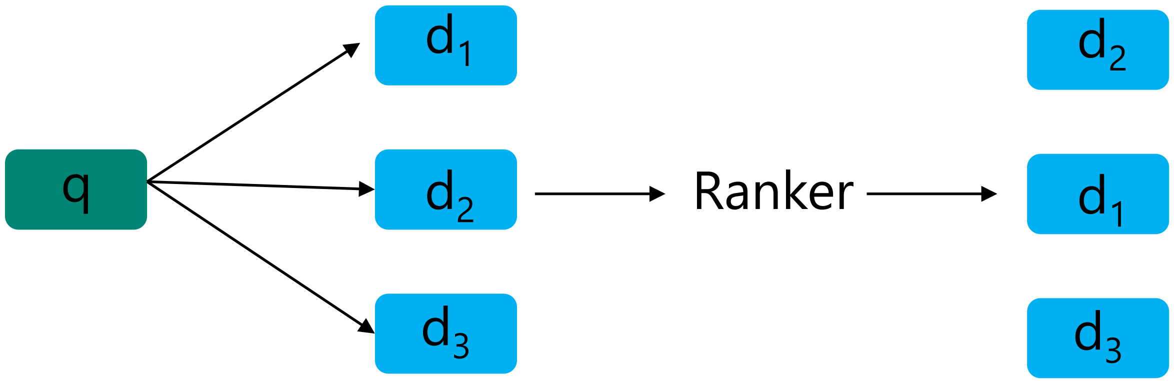
Matryoshka Representation Learning, 2022

Caveats on Embedding Distance Metric

- Cosine similarity
  - Bounded within the interval [-1, 1]
- Dot product
  - Unbounded, can be any real-valued number (theoretically)
- Both do not satisfy triangle inequality
  - Under dot product, a text may not have the highest score with itself.

LLMs for Ranking

- Task definition (also called "re-ranking")
  - Given a query and a list of document, return a ranked list based on relevancy



Zero-shot Pointwise Ranking

- Prompt LLMs whether the document contains answer for the query
  - Take the log probability of "Yes" as the relevance score

- Shortcomings
  - Scores are uncalibrated
  - Couples with the tokenizer/vocabulary

Holistic evaluation of language models, 2022

RankLLaMA

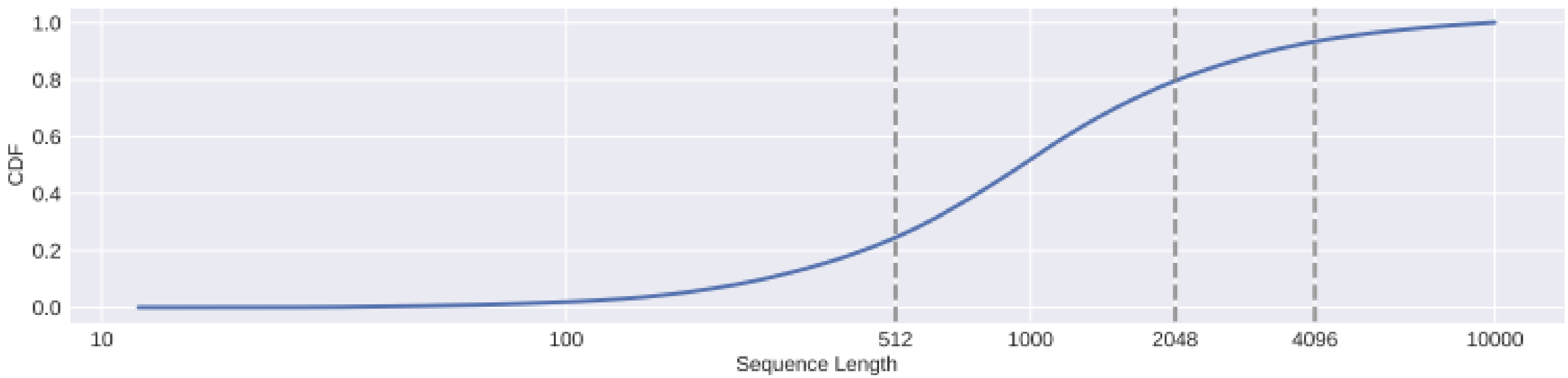
- Fine-tune LLMs for pointwise ranking

	Model size	Source prev.	top-k	DEV MRR@10	DEV R@1k	DL19 nDCG@10	DL20 nDCG@10
Retrieval							
BM25 (Lin et al., 2021)	-	-	$\|C\|$	18.4	85.3	50.6	48.0
ANCE (Xiong et al., 2021)	125M	-	$\|C\|$	33.0	95.9	64.5	64.6
CoCondenser (Gao and Callan, 2022b)	110M	-	$\|C\|$	38.2	98.4	71.7	68.4
GTR-base (Ni et al., 2022)	110M	-	$\|C\|$	36.6	98.3	-	-
GTR-XXL (Ni et al., 2022)	4.8B	-	$\|C\|$	38.8	99.0	-	-
OpenAI Ada2 (Neelakantan et al., 2022)	?	-	$\|C\|$	34.4	98.6	70.4	67.6
bi-SimLM (Wang et al., 2023)	110M	-	$\|C\|$	39.1	98.6	69.8	69.2
RepLLaMA	7B	-	$\|C\|$	**41.2**	**99.4**	**74.3**	**72.1**
Reranking							
monoBERT (Nogueira et al., 2019)	110M	BM25	1000	37.2	85.3	72.3	72.2
cross-SimLM (Wang et al., 2023)	110M	bi-SimLM	200	43.7	98.7	74.6	72.7
RankT5 (Zhuang et al., 2023)	220M	GTR	1000	43.4	98.3	-	-
RankLLaMA	7B	RepLLaMA	200	44.9	99.4	75.6	77.4
RankLLaMA-13B	13B	RepLLaMA	200	**45.2**	**99.4**	**76.0**	**77.9**

Fine-tuning llama for multi-stage text retrieval, 2023

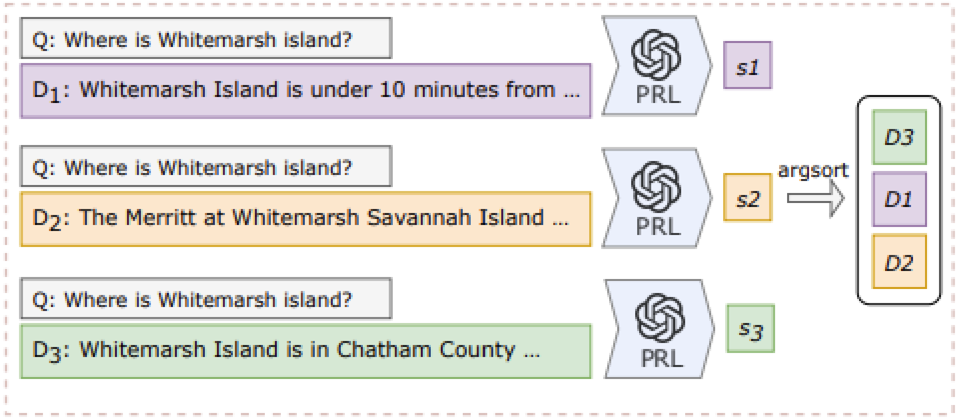
RankLLaMA

- Fine-tune LLMs for pointwise ranking
  - Naturally supports long document ranking

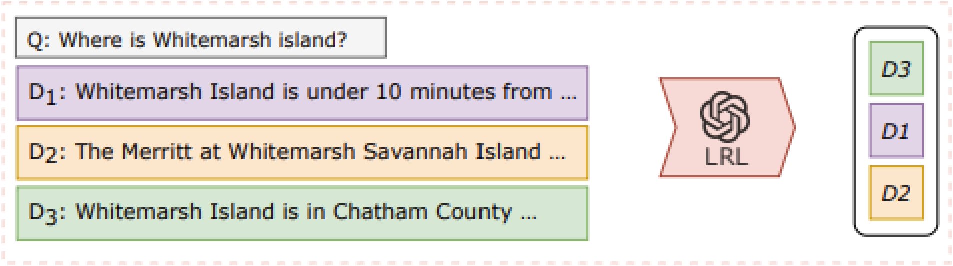


Fine-tuning llama for multi-stage text retrieval, 2023

Zero-shot Listwise Ranking



(a) Pointwise reranking pipeline.

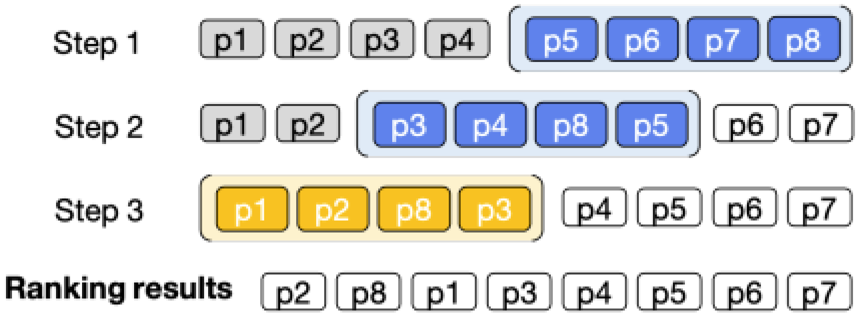


(b) Listwise reranking pipeline.

Is chatgpt good at search? investigating large language models as re-ranking agent, 2023
Zero-shot listwise document reranking with a large language model, 2023

Zero-shot Listwise Ranking

- Use sliding window if the documents are too much
  - Sliding window of 4 documents with stride 2



Is chatgpt good at search? investigating large language models as re-ranking agent, 2023
Zero-shot listwise document reranking with a large language model, 2023

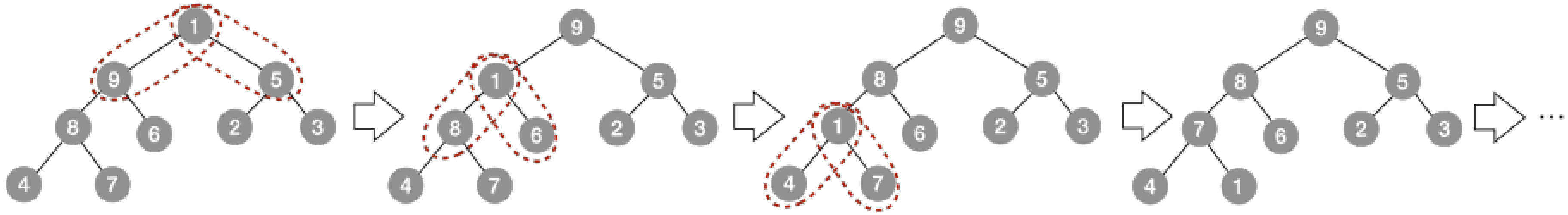
Zero-shot Listwise Ranking

	Source		DL19		DL20		DL21	
	prev.	top-k	nDCG@10	MRR@10	nDCG@10	MRR@10	nDCG@10	MRR@10
				Zero-shot				
(1) BM25	None	$\|C\|$	0.5058	0.7024	0.4796	0.6533	0.4458	0.4981
(2) Contriever	None	$\|C\|$	0.4454	0.5928	0.4213	0.5408	–	–
(3) UPR	BM25	100	0.5910	0.6494	0.5958	0.7247	0.5621	0.6956
(4) PRL	BM25	100	0.5975	0.7347	0.6088	0.7699	0.5678	0.7148
(5) LRL	BM25	100	0.6580	0.8517	0.6224	0.8230	0.5996	**0.8113**
(6) LRL	UPR	10	0.6382	0.8320	0.6357	**0.8256**	0.5867	0.7543
(7) LRL	UPR	20	0.6561	**0.8659**	**0.6364**	0.8129	0.6035	0.7464
(8) LRL	PRL	10	0.6369	0.8085	0.6116	0.7841	0.5844	0.7315
(9) LRL	PRL	20	**0.6650**	0.8405	0.6349	0.8237	**0.6260**	0.7689
				Supervised				
(a) DPR	None	$\|C\|$	0.6297	0.7388	0.6480	0.8184	–	–
(b) TCT_ColBERT	None	$\|C\|$	0.7210	**0.8864**	0.6854	0.8392	0.5001	0.6527
(c) MonoBERT	BM25	1000	0.7233	0.8566	0.7218	0.8530	0.6098	0.7278
(d) MonoELECTRA	DPR	1000	**0.7557**	0.8748	**0.7450**	**0.8650**	–	–

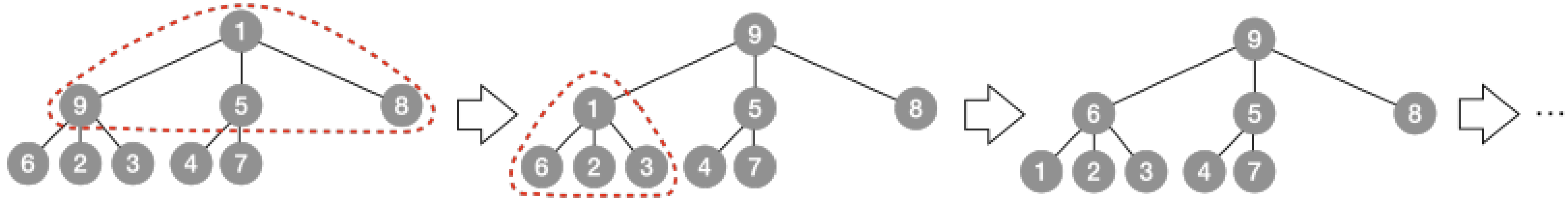
Zero-shot listwise document reranking with a large language model, 2023

Zero-shot Setwise Ranking

- Borrow the wisdom from the classic sorting algorithms



(a) Heapify with Pairwise prompting (comparing 2 documents at a time).



(b) Heapify with our Setwise prompting (comparing 4 documents at a time).

A Setwise Approach for Effective and Highly Efficient Zero-shot Ranking with Large Language Models, 2023

LLMs are Strong Data Generators

LLMs for Query Generation

- Generate pseudo-queries from documents

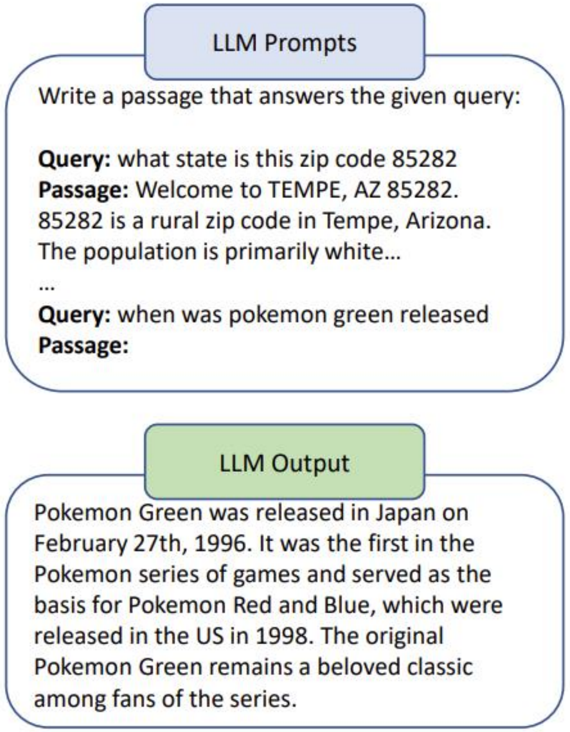
- Doc2query[1]
  - Trained on labeled <document, query> pairs
  - Document expansion

- Gecko[2]
  - Zero-shot prompting LLMs

1. Document expansion by query prediction, 2019
2. Gecko: Versatile text embeddings distilled from large language models, 2024

LLMs for Document Generation

- Query2doc: generate documents from query
  - Query expansion



Query expansion by generating pseudo-documents

Query2doc: Query Expansion with Large Language Models, 2023

LLMs for Document Generation

- Query2doc augmented BM25 is a strong zero-shot retriever

Method	Fine-tuning	MS MARCO dev			TREC DL 19	TREC DL 20
		MRR@10	R@50	R@1k	nDCG@10	nDCG@10
Sparse retrieval						
BM25	✗	18.4	58.5	85.7	51.2*	47.7*
+ query2doc	✗	$21.4^{+3.0}$	$65.3^{+6.8}$	$91.8^{+6.1}$	$\mathbf{66.2}^{+15.0}$	$\mathbf{62.9}^{+15.2}$
BM25 + RM3	✗	15.8	56.7	86.4	52.2	47.4
docT5query (Nogueira and Lin)	✓	**27.7**	**75.6**	**94.7**	64.2	-
Dense retrieval w/o distillation						
ANCE (Xiong et al., 2021)	✓	33.0	-	95.9	64.5	64.6
HyDE (Gao et al., 2022)	✗	-	-	-	61.3	57.9
$\text{DPR}_{\text{bert-base}}$ (our impl.)	✓	33.7	80.5	95.9	64.7	64.1
+ query2doc	✓	$\mathbf{35.1}^{+1.4}$	$\mathbf{82.6}^{+2.1}$	$\mathbf{97.2}^{+1.3}$	$\mathbf{68.7}^{+4.0}$	$\mathbf{67.1}^{+3.0}$

Query2doc: Query Expansion with Large Language Models, 2023

LLMs for Relevance Judgments

- Training data generation / LLM-based evaluation metric
  - Often better than human annotators

Gecko: Versatile text embeddings distilled from large language models, 2024

Synthetic Datasets Generation

- Datasets generation by prompting GPT-4

You have been assigned a retrieval task: *{task}*
Your mission is to write one text retrieval example for this task in JSON format. The JSON object must contain the following keys:
- **"user_query"**: a string, a random user search query specified by the retrieval task.
- **"positive_document"**: a string, a relevant document for the user query.
- **"hard_negative_document"**: a string, a hard negative document that only appears relevant to the query.

Please adhere to the following guidelines:
- The "user_query" should be *{query_type}*, *{query_length}*, *{clarity}*, and diverse in topic.
- All documents should be at least *{num_words}* words long.
- Both the query and documents should be in *{language}*.
- ... *(omitted some for space)*

Your output must always be a JSON object only, do not explain yourself or output anything else. Be creative!

{**"user_query":** "How to use Microsoft Power BI for data analysis",
"positive_document": "Microsoft Power BI is a sophisticated tool that requires time and practice to master. In this tutorial, we'll show you how to navigate Power BI ... *(omitted)* ",
“hard_negative_document”: “Excel is an incredibly powerful tool for managing and analyzing large amounts of data. Our tutorial series focuses on how you...*(omitted)*” }

Model	BEIR Retrieval (15 datasets)	MTEB Average (56 datasets)
OpenAI Ada-002	49.3	61.0
Cohere-embed-english-v3.0	55.0	64.5
voyage-lite-01-instruct	55.6	64.5
UAE-Large-V1 [22]	54.7	64.6
$E5_{mistral\text{-}7b}$ + full data	**56.9**	**66.6**

[Improving Text Embeddings with Large Language Models](), 2024

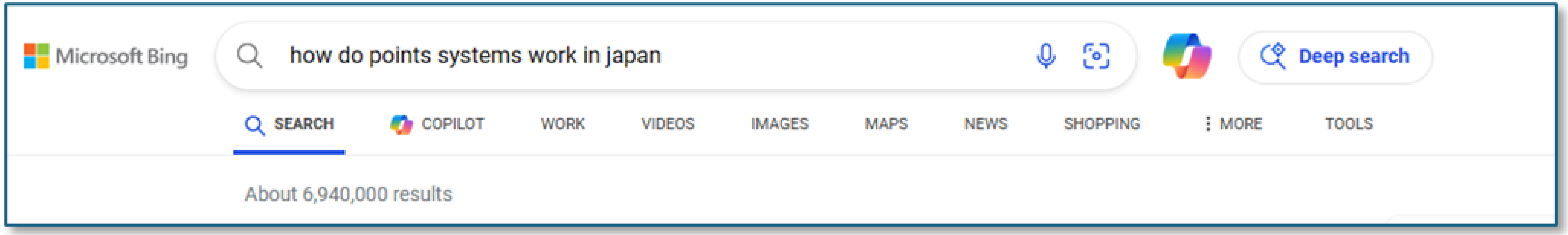
LLMs for Ranking in Production

- Bing saw the largest relevancy jump after integrating GPT-4

https://searchengineland.com/microsoft-bing-saw-the-largest-relevancy-jump-in-search-in-two-decades-392969

LLMs for Ranking in Production

- Bing Deep Search: better search results with a bit patience

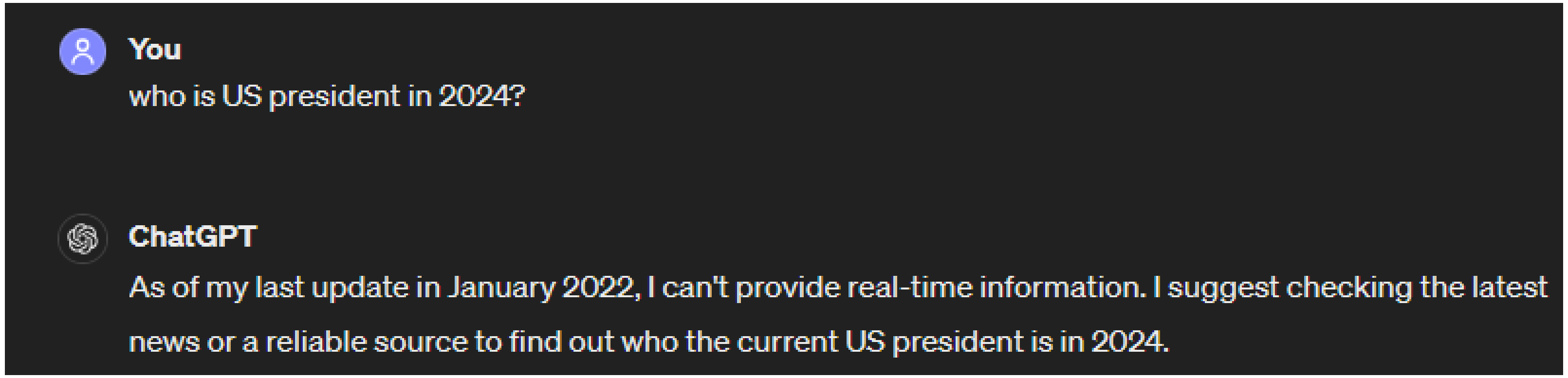


Bing Deep Search feature: [Introducing Deep Search | Search Quality Insights (bing.com)]()
Demo: [https://twitter.com/JordiRib1/status/1771214752485691797]()

How can search engines augment LLMs?

Limitations of LLMs

- Static parametric knowledge
  - Unaware of latest events
  - Unaware of private information
  - Non-trivial to inject new knowledge through fine-tuning



RAG Pipeline

- Retrieve, prompt construction, generate

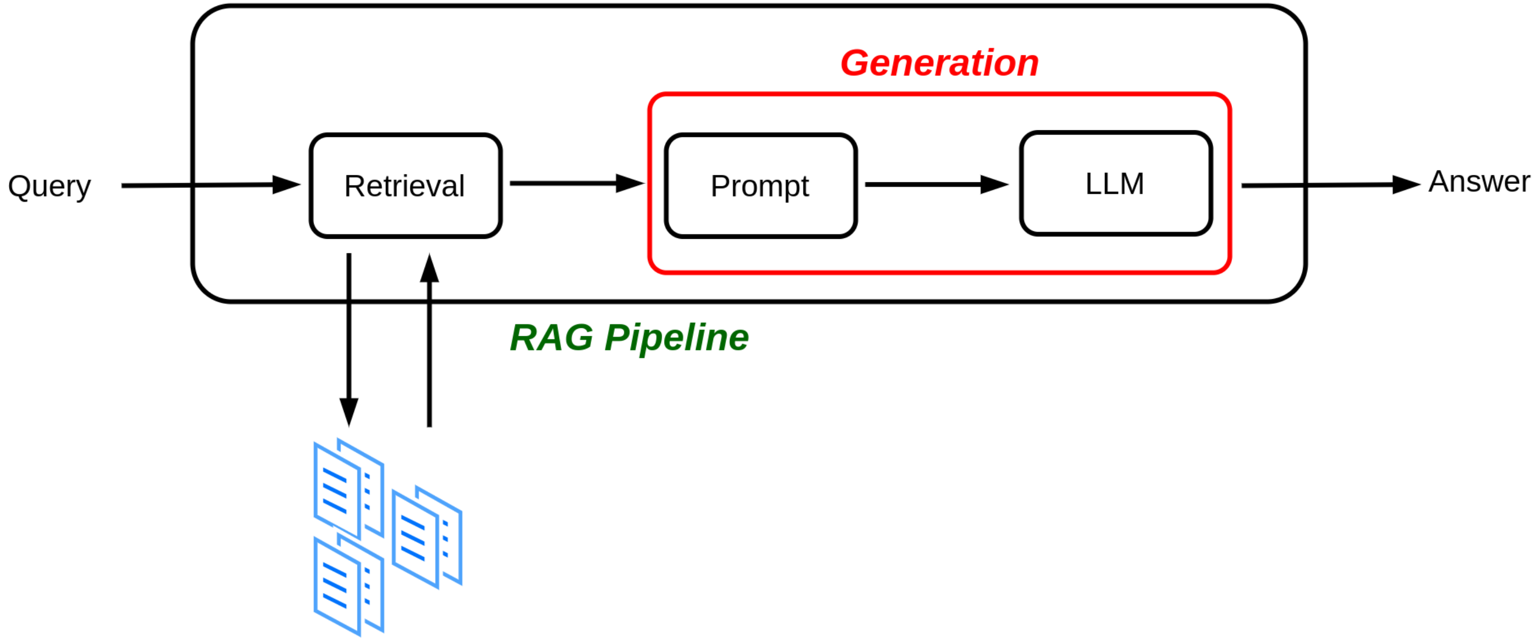
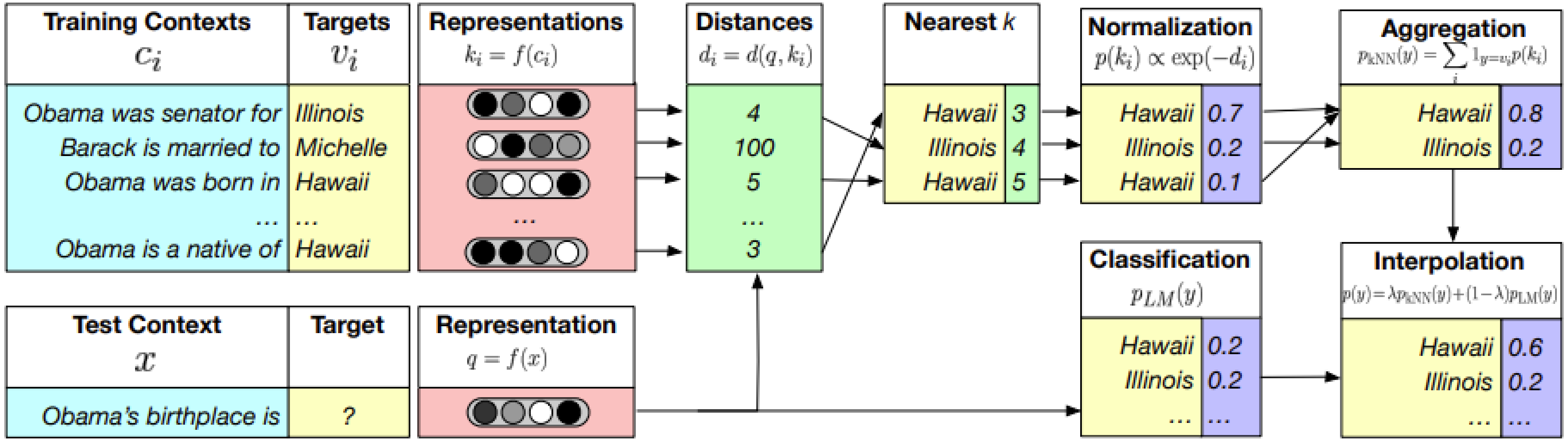


Figure credit to https://mallahyari.github.io/rag-ebook/03_prepare_data.html

kNN-LM

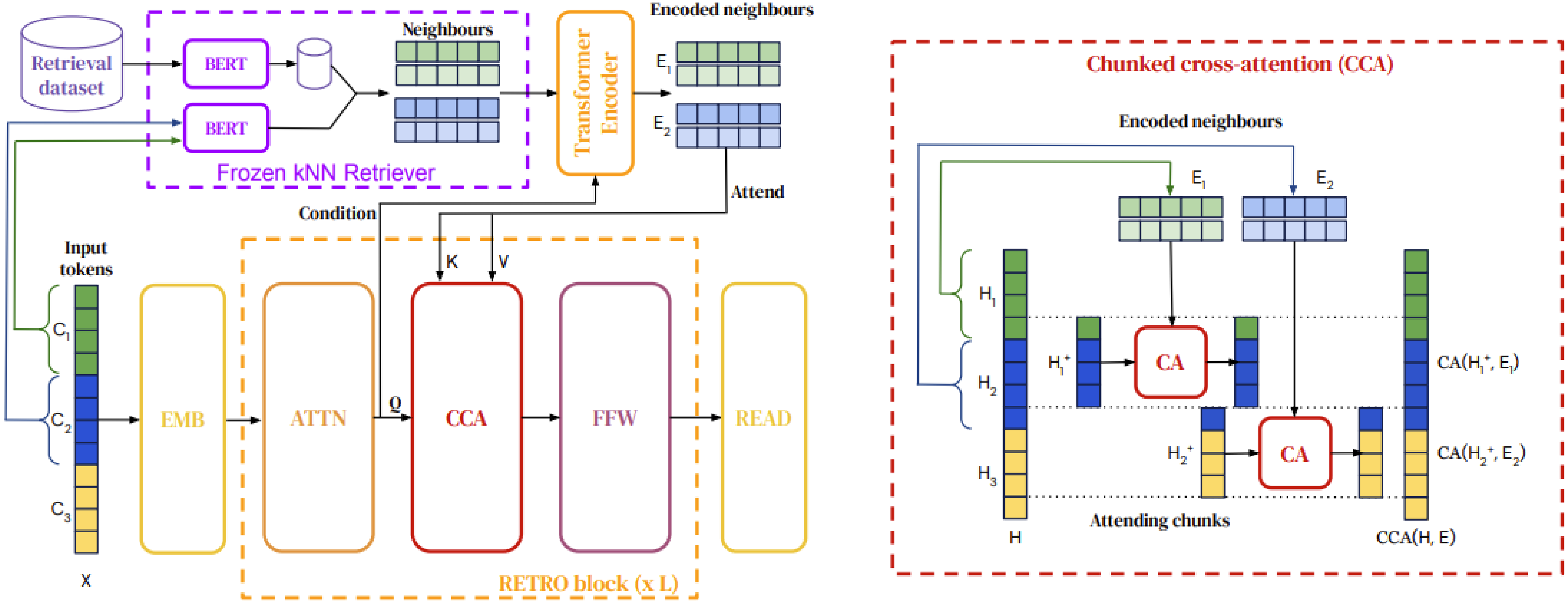
- Output fusion
  - No training or architecture modification is required
  - Interpretable and scalable



Generalization through Memorization: Nearest Neighbor Language Models, 2019

RETRO

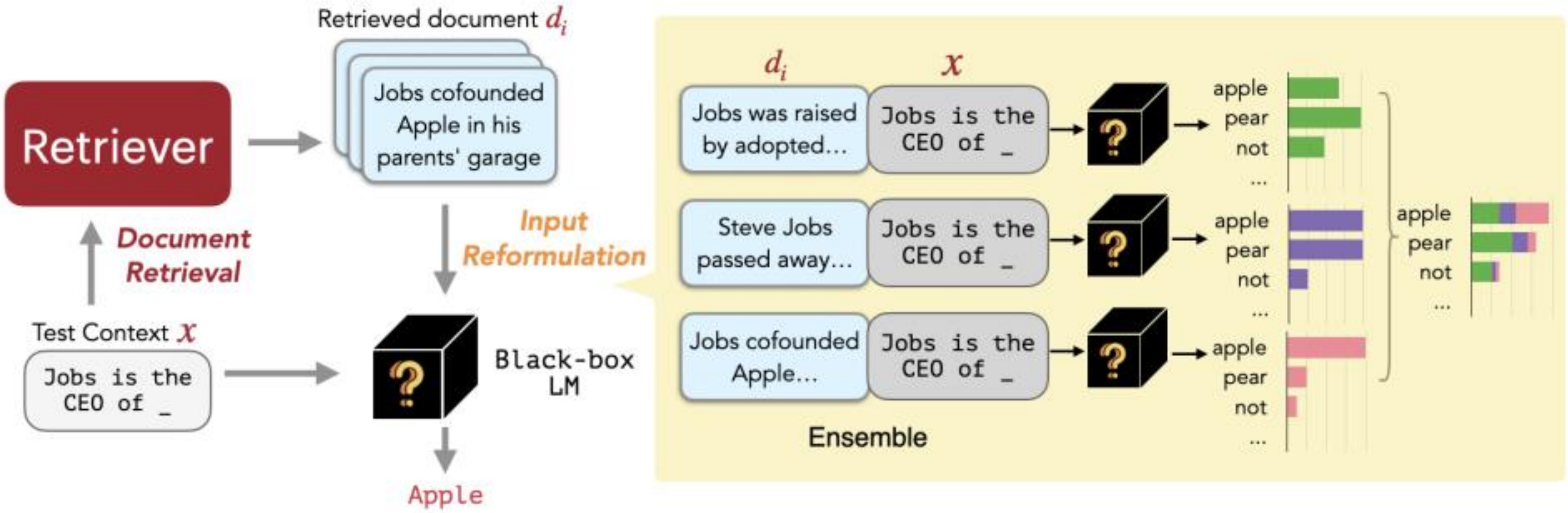
- Intermediate fusion through chunked cross-attention
  - More fine-grained fusion but requires additional training



Improving language models by retrieving from trillions of tokens, 2021

REPLUG

- Input fusion
  - Applicable to API-only proprietary LLMs



Replug: Retrieval-augmented black-box language models, 2023

RAG Agents

- Most NLP tasks only require one-step action
- Agents
  - Decision making capability
  - Tool use
- Search engine is a powerful tool

WebGPT

- Agent's action space
  - Step 1: supervised learning with human labeled data
  - Step 2: RLHF

Command	Effect
`Search <query>`	Send `<query>` to the Bing API and display a search results page
`Clicked on link <link ID>`	Follow the link with the given ID to a new page
`Find in page: <text>`	Find the next occurrence of `<text>` and scroll to it
`Quote: <text>`	If `<text>` is found in the current page, add it as a reference
`Scrolled down <1, 2, 3>`	Scroll down a number of times
`Scrolled up <1, 2, 3>`	Scroll up a number of times
`Top`	Scroll to the top of the page
`Back`	Go to the previous page
`End: Answer`	End browsing and move to answering phase
`End: <Nonsense, Controversial>`	End browsing and skip answering phase

WebGPT: Browser-assisted question-answering with human feedback, 2021

WebGPT

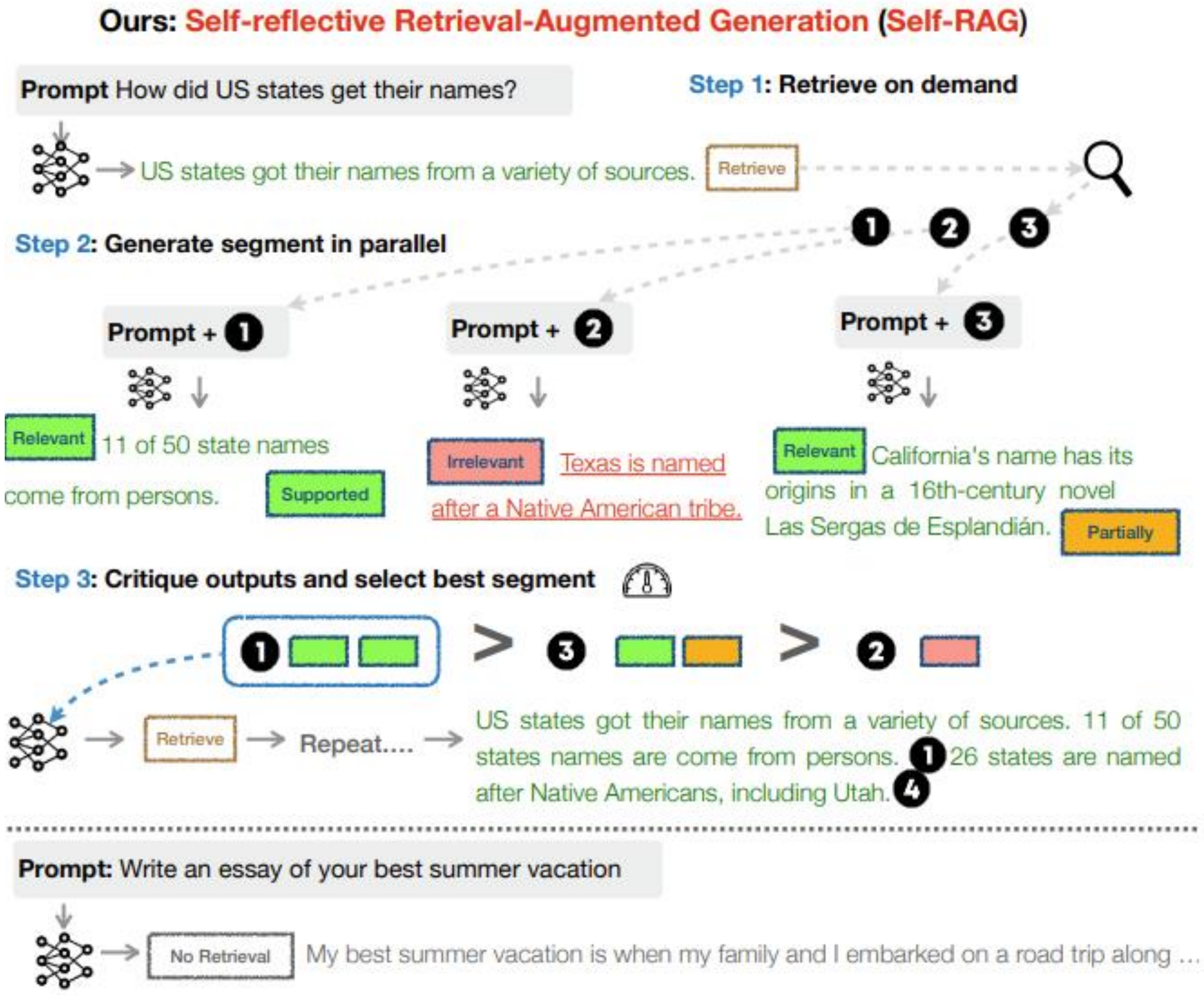
- Answer with references
  - WebGPT has likely inspired products such as New Bing and Perplexity.ai etc.

Question	Why did we decide that certain words were "bad" and shouldn't be used in social settings?
Answer	We have all been taught that there are certain words that should not be used in public, at least not without good reason. But why do some words fall into this category while others do not? It turns out that there is not a single answer to this question, as each word has a different history and carries different meanings in different cultures and families [2]. However, there are some general categories that "bad" words tend to fall into. "Bad" words generally relate to parts of life that we don't like talking about in public, like bathroom functions, or negative ways of talking about people's religion, skin color, ability, or gender [3]. Often, people say bad words when they are experiencing strong emotions, and those words are linked to our emotions [3].
References (titles only)	[1, 2, 3] Why Are Some Words 'Bad'? | Vermont Public Radio (www.vpr.org) [4] On Words: 'Bad' Words and Why We Should Study Them | UVA Today (news.virginia.edu) [5] The Science of Curse Words: Why The &@$! Do We Swear? (www.babbel.com)

WebGPT: Browser-assisted question-answering with human feedback, 2021

Self-RAG

- RAG with self-reflection
  - Open-source data and models
  - Retrieve on-demand
  - Easy to train with next-token-prediction



Self-RAG: Learning to retrieve, generate, and critique through self-reflection, 2023

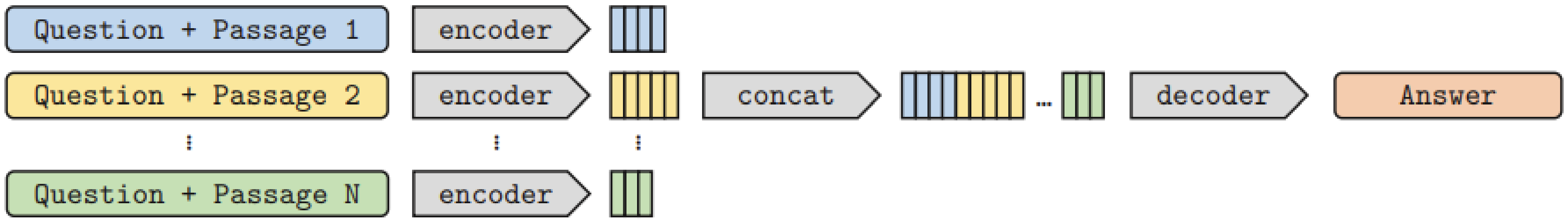
RAG versus Long-context LLMs

- RAG requires long-context modeling (many documents)
- Long-context modeling can sometimes make RAG unnecessary

Retrieval meets long context large language models, 2023

Fusion-in Decoder (FiD)

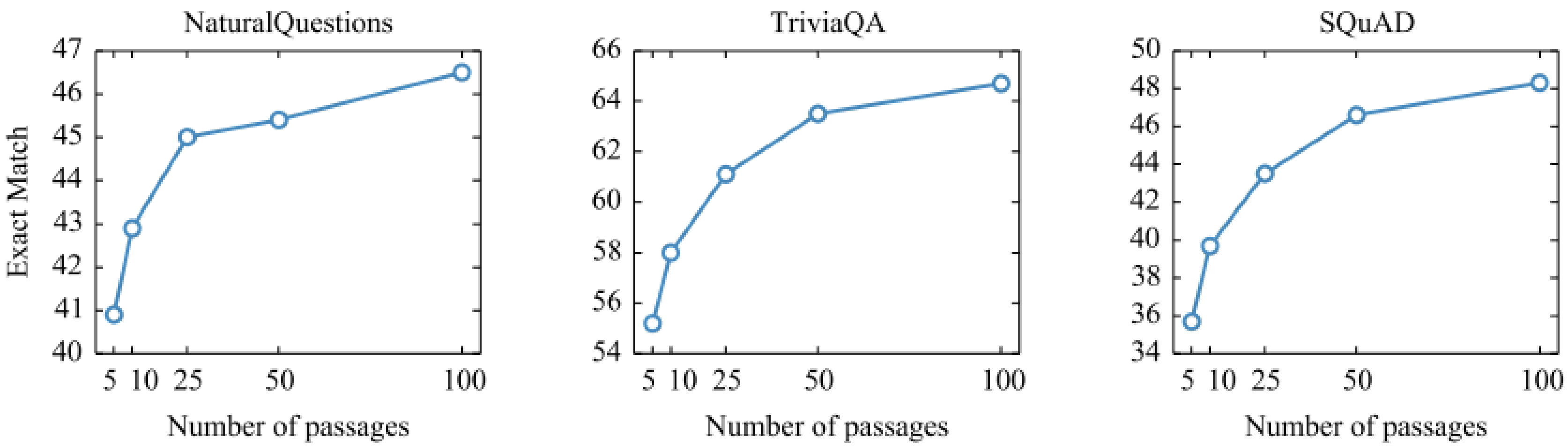
- Incorporating many passages for encoder-decoder architecture
  - Bypasses the long-context modeling issue



Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering, 2020

Fusion-in Decoder (FiD)

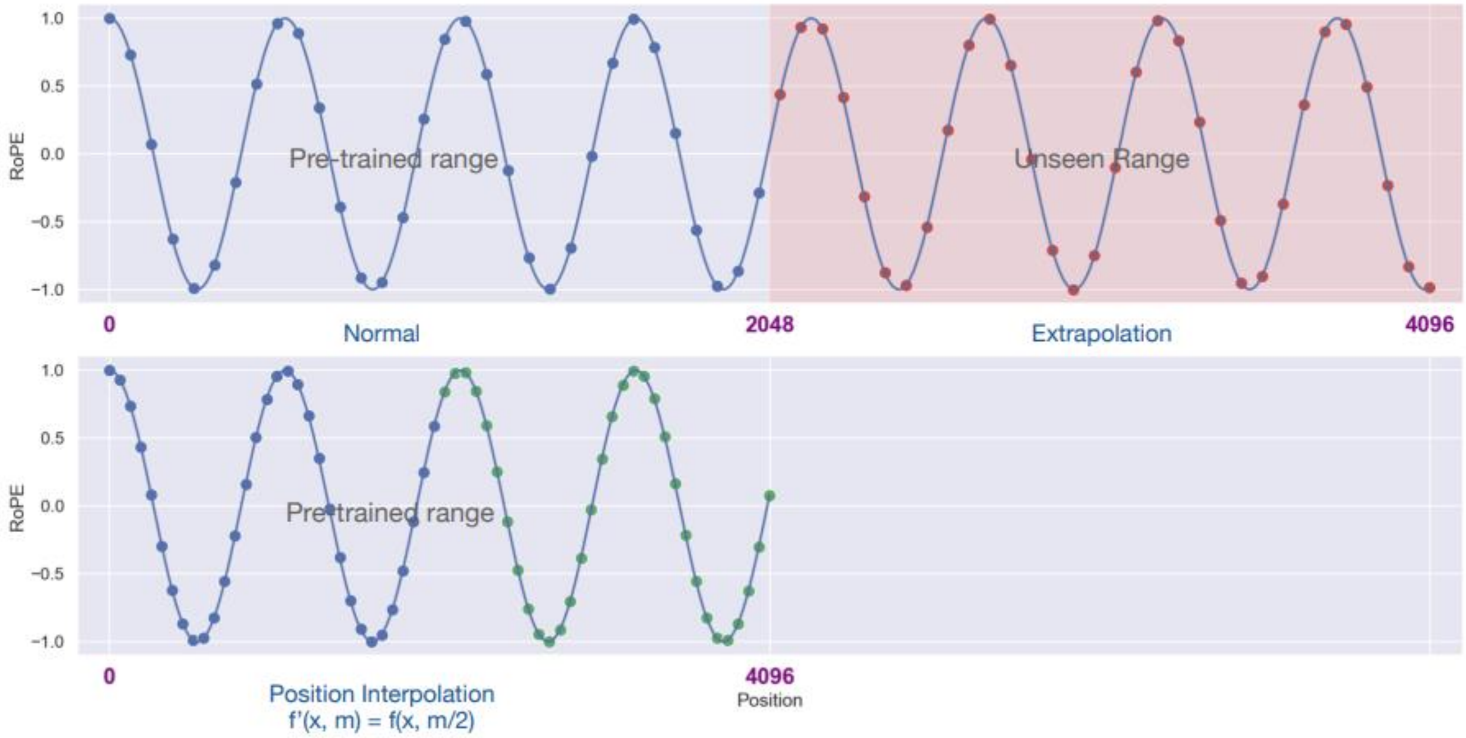
- Performance improves as FiD incorporates more passages
  - This is not the case (empirically) for decoder-only LLMs for now



Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering, 2020

Position Interpolation

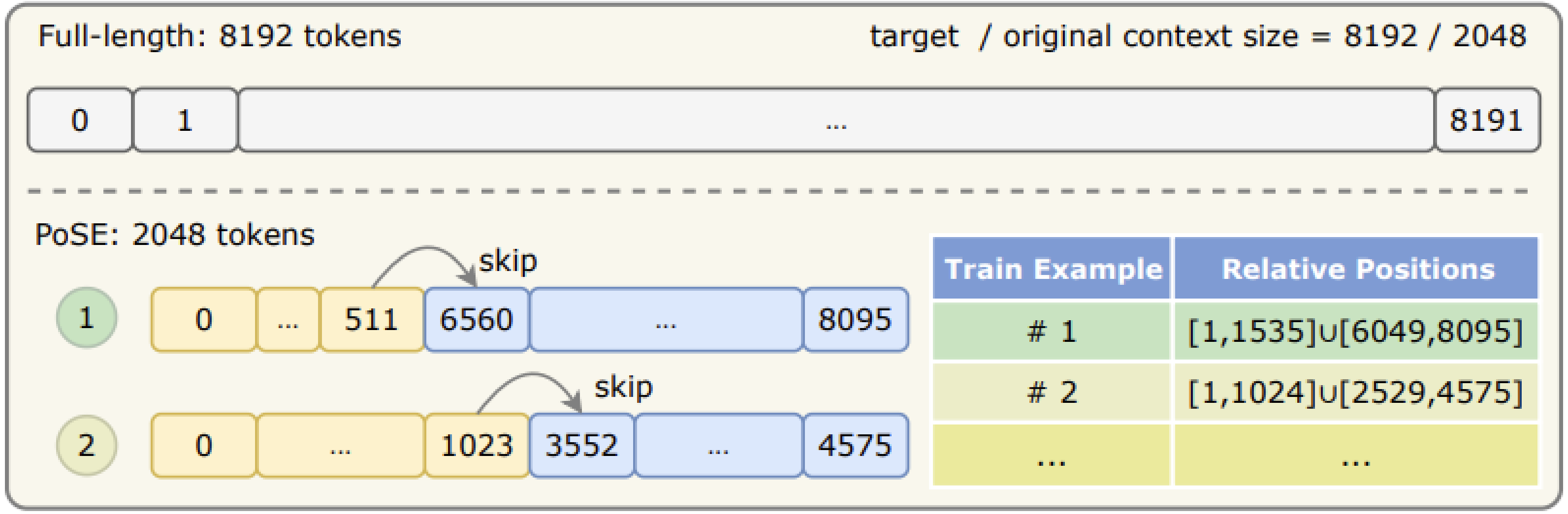
- RoPE positional interpolation -> full fine-tuning



Extending context window of large language models via positional interpolation, 2023

PoSE

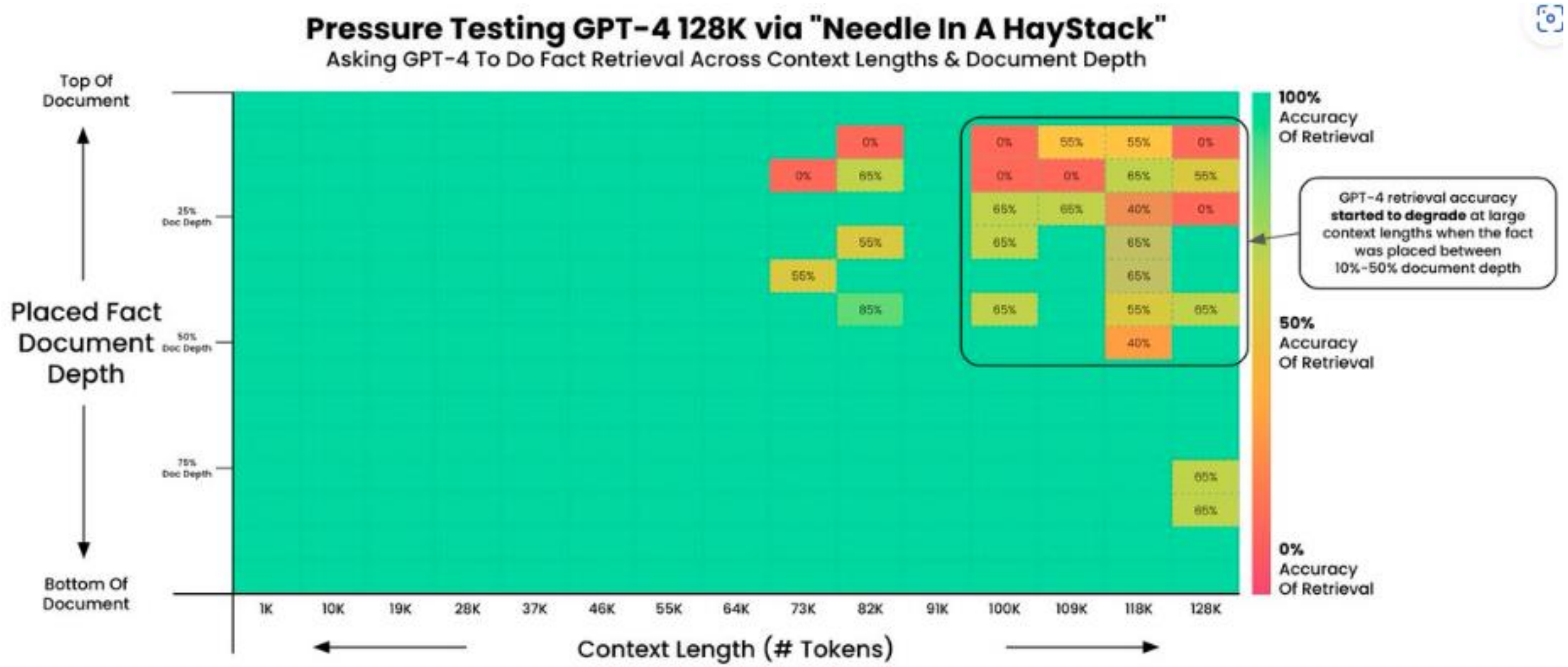
- Positional Skip-wise training
  - Context window extension by training on the short sequences only



Pose: Efficient context window extension of llms via positional skip-wise training, 2023

Challenges – Long Context Understanding

- Lost in the middle
  - Needle in haystack: an easy task that many LLMs fail

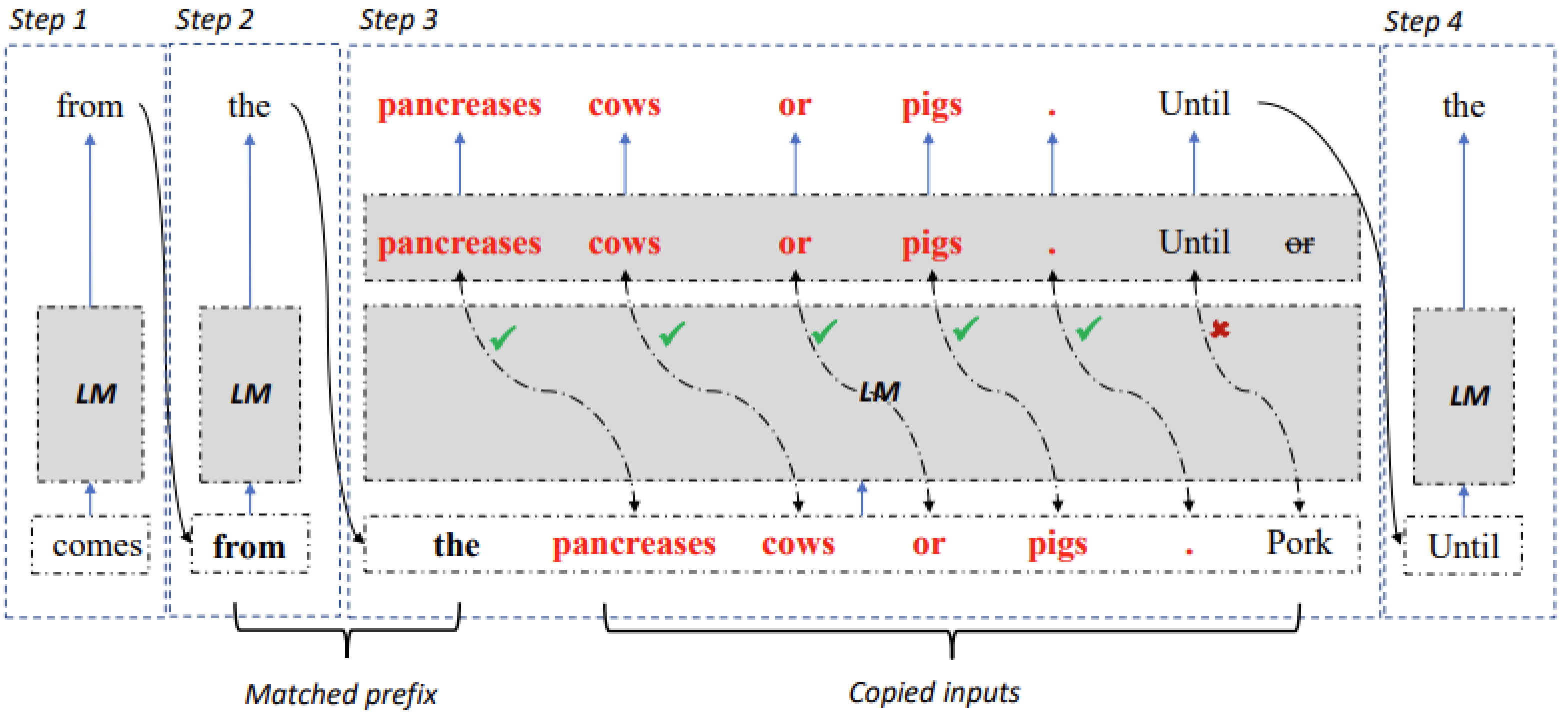


Lost in the middle: How language models use long contexts, 2023

Figure from [Arize]

Challenges – Inference Efficiency

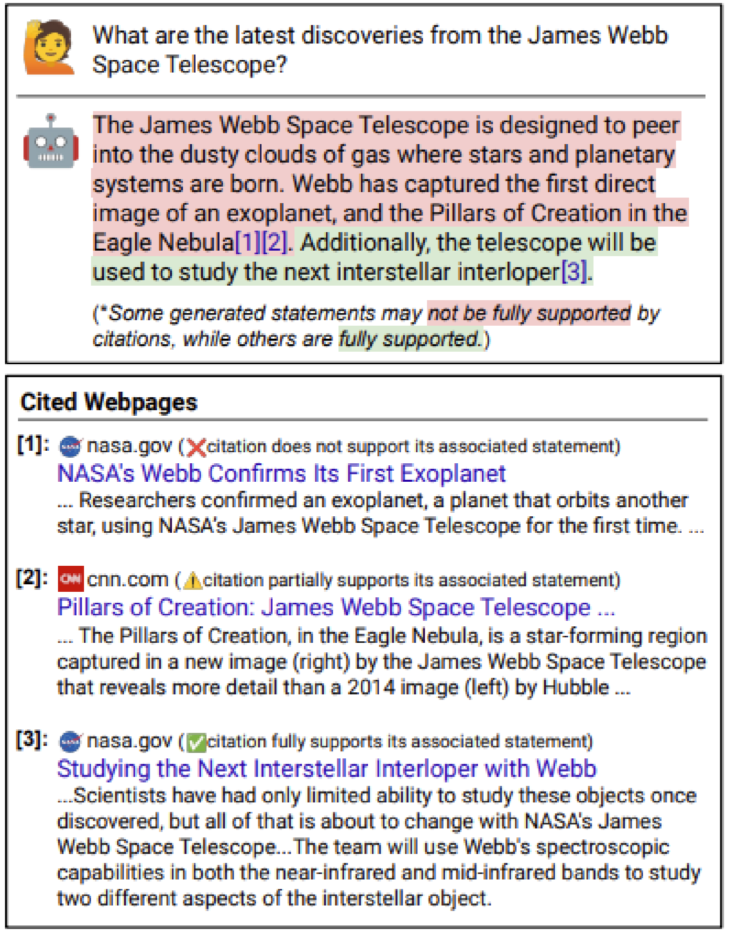
- Inference with Reference
  - Speculative decoding without the need for a small LM



Inference with reference: Lossless acceleration of large language models, 2023

Challenges – Source Attribution

- LLM-generated contents may not be fully supported by its sources

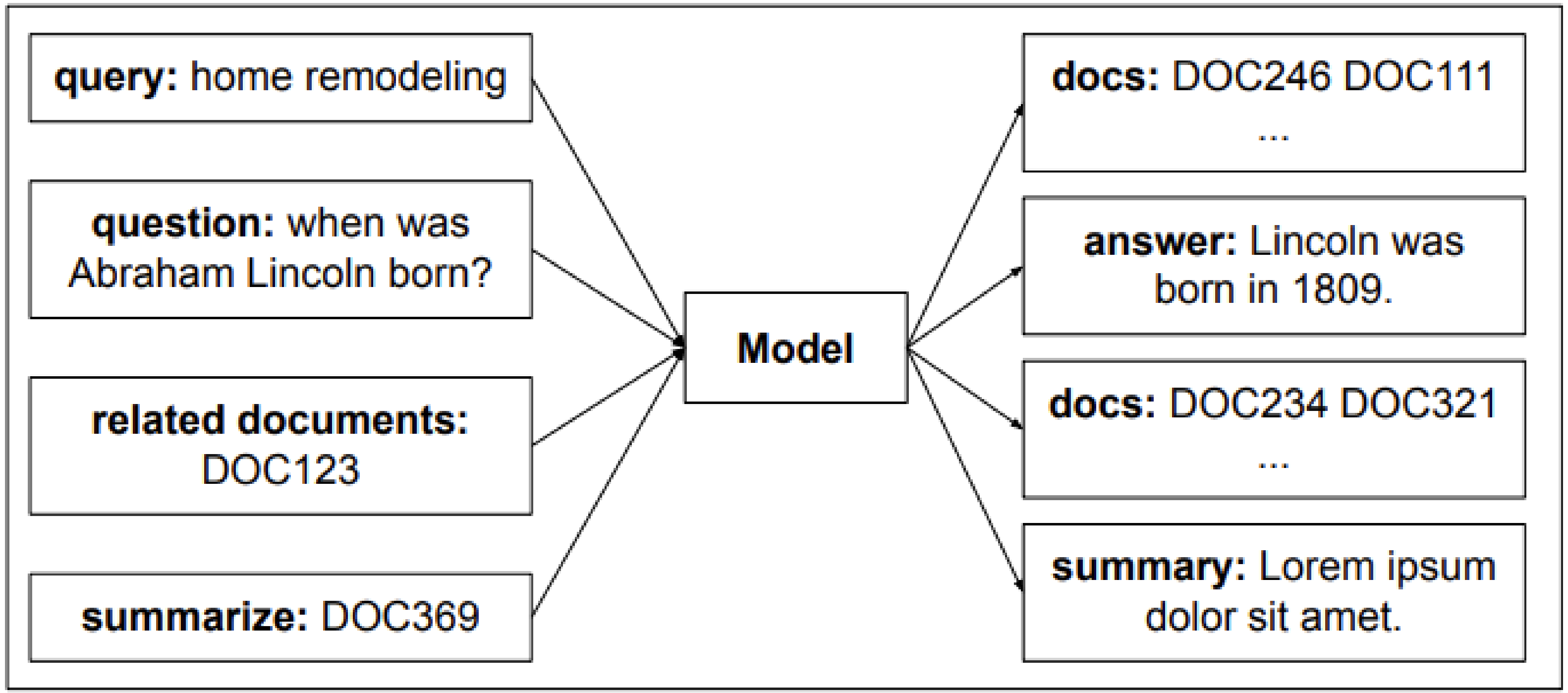


Evaluating verifiability in generative search engines, 2023

Will LLMs make search engines obsolete?

A Proposal from Google

- Ideally, LLMs memorize and reason over the entire corpus
  - The DSI model is a proof-of-concept of this proposal



Rethinking Search: Making Domain Experts out of Dilettantes, 2021

A Proposal from Google

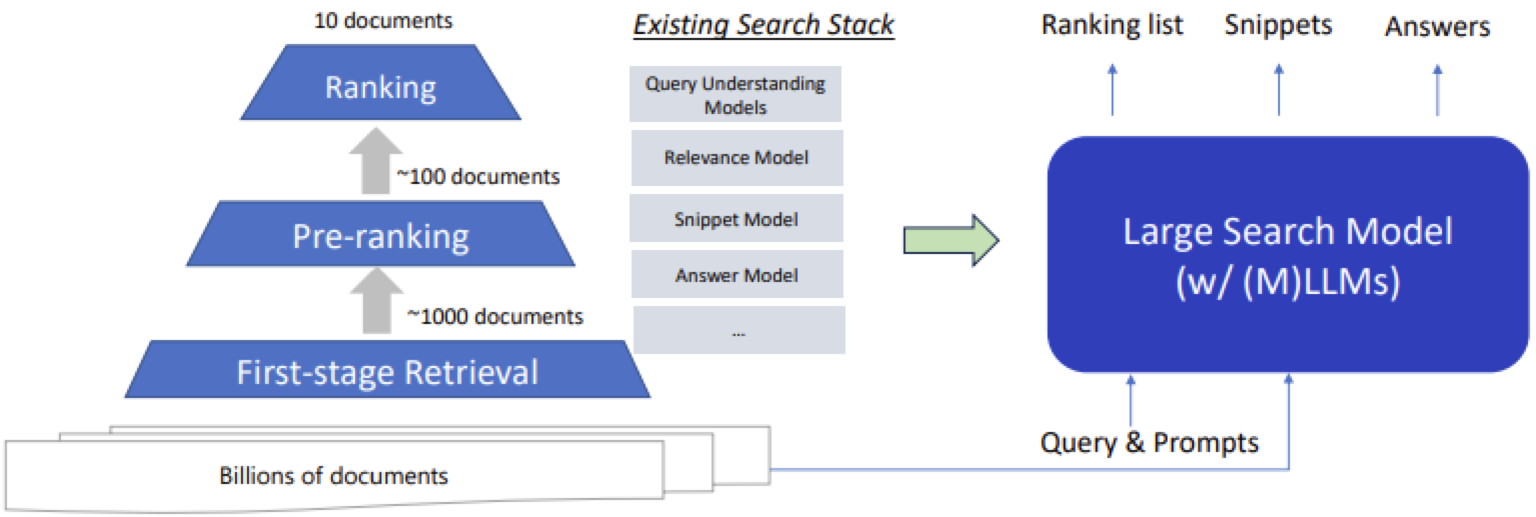
- Reality check: limited success
  - Now only works for small corpus or well-structured corpus (e.g., Wikipedia)
  - Operating in the ID space is hard to scale
  - Hallucination for autoregressive generation

	Model	MSMarco100k			MSMarco1M			MSMarcoFULL		
		At.	Nv.	Sm.	At.	Nv.	Sm.	At.	Nv.	Sm.
Baselines										
BM25		-	65.3	-	-	41.3	-	-	18.4	-
BM25 (w/ doc2query–T5)		-	80.4	-	-	56.6	-	-	27.2	-
GTR-Base		-	83.2	-	-	60.7	-	-	34.8	-
Ours										
(1a)	Labeled Queries (No Indexing)	0.0	1.1	0.0	0.0	0.5	0.0	0.0	0.0	0.0
(2a)	FirstP/DaQ + Labeled Queries (DSI)	0.0	23.9	19.2	2.1	12.4	7.4	0.0	7.5	3.1
(3b)	FirstP/DaQ + D2Q + Labeled Queries	79.2	77.7	76.8	53.3	48.2	47.1	14.2	**13.2**	6.4
(4a)	3b + PAWA (w/ 2D Semantic IDs)	-	-	77.1	-	-	50.2	-	-	9.0
(5)	4a + Consistency Loss (NCI)	-	-	77.1	-	-	50.2	-	-	9.1
(6b)	D2Q only	**80.3**	**78.7**	**78.5**	**55.8**	**55.4**	54.0	**24.2**	**13.3**	11.8
($4a'$)	6b + PAWA (w/ 2D Semantic IDs)	-	-	78.2	-	-	**54.1**	-	-	**17.3**
($4b'$)	6b + Constrained Decoding	-	-	**78.6**	-	-	54.0	-	-	12.0
($5'$)	6b + PAWA (w/ 2D Semantic IDs) + Constrained Decoding	-	-	78.3	-	-	**54.2**	-	-	**17.4**

Rethinking Search: Making Domain Experts out of Dilettantes, 2021
How Does Generative Retrieval Scale to Millions of Passages?, 2023

Large Search Model

- Embedding based first-stage retrieval
- LLMs reason over thousands of retrieved documents
  - Ranking, answer generation, snippets, related searches etc.



Large Search Model: Redefining Search Stack in the Era of LLMs, 2023

Large Search Model

- Proof-of-concept results
  - Joint listwise ranking and RAG

	MS MARCO	TREC DL 19	TREC DL 20
BM25	18.4	51.2	47.7
ANCE [Xiong et al., 2021]	33.0	64.5	64.6
$E5_{large-v2}$ [Wang et al., 2022]	38.4	70.9	72.1
Ours (Listwise rank + $LLaMA_{7b}$)	**41.7**	**72.9**	**74.0**

- Challenges
  - Long-context understanding
  - Efficiency
  - Data curation and evaluation

Large Search Model: Redefining Search Stack in the Era of LLMs, 2023

Obstacles for LLM-native Search

- Efficient continual learning of new knowledge
- (Almost) no hallucination
- Inference cost and latency

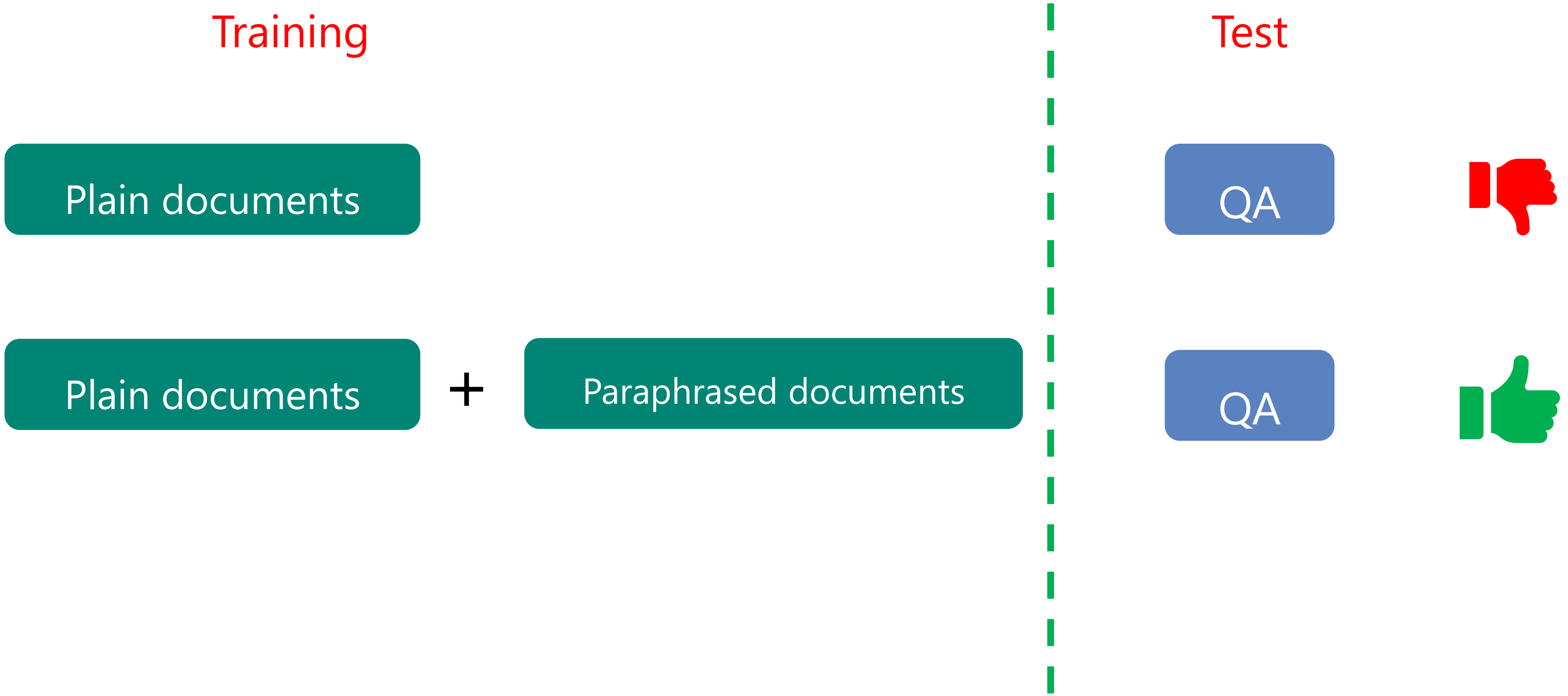
Future Research Focus

General-purpose Embeddings

- General-modality
  - Text, code, image, audio, video etc.
- Long-context support
  - Accurately encode information from long sequences
- Customizable
  - “System prompt” for embeddings?
- Internet scale
  - Efficient ANN search, storage

Continual Learning of LLMs

- Efficiently injecting new knowledge into LLMs



Physics of Language Models: Part 3.1, Knowledge Storage and Extraction, 2023

Efficient and Reliable RAG

- Simple
  - Existing pipelines are complex
- Fast
  - Pre-filling KV cache, auto-regressive generation
- Accurate
  - Hallucination is unacceptable in many scenarios
  - Robust to irrelevant retrieval results and domain shifts

Conclusion

- How can LLMs help in existing search stack?
  - Generative retrieval
  - Text retrieval and ranking by leveraging LLMs
  - Synthetic data generation in all directions

- How can search engines augment LLMs
  - Retrieval-augmented generation
  - Agents with retrieval capability

- Will LLMs make search engines obsolete?
  - LLMs and search engines are likely complementary in foreseeable future

Microsoft

Thank you