Retrieval Augmented Generation

Lecture: Information Retrieval 2

29-11-2024

Antonis Krasakis

Disclaimer

Slides are largely-based on the excellent ACL 2023 tutorial:

Retrieval-based Language Models and Applications

Akari Asai, Sewon Min, Zexuan Zhong, Danqi Chen

https://acl2023-retrieval-lm.github.io/

What?

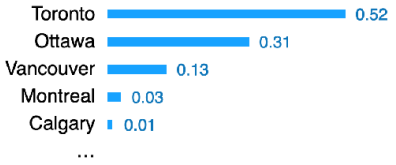
Retrieval-based language models (LMs)

Retrieval-based LMs = Retrieval + LMs

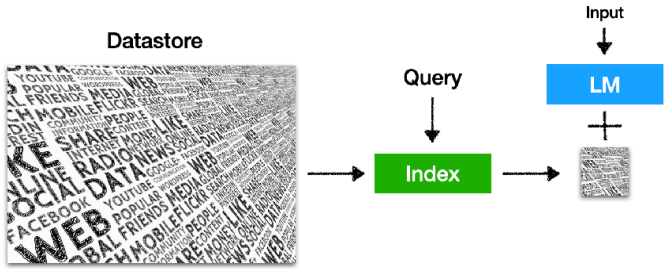
- It is a **language model** $P(x_n \mid x_1, x_2, \cdots, x_{n-1})$

The capital city of Ontario is ___

(can be broadly extended to masked language models or encoder-decoder models)



- It retrieves from an **external datastore** (at least during inference time)



(Also referred to semiparametric and non-parametric models)

The age of large language models (LLMs)

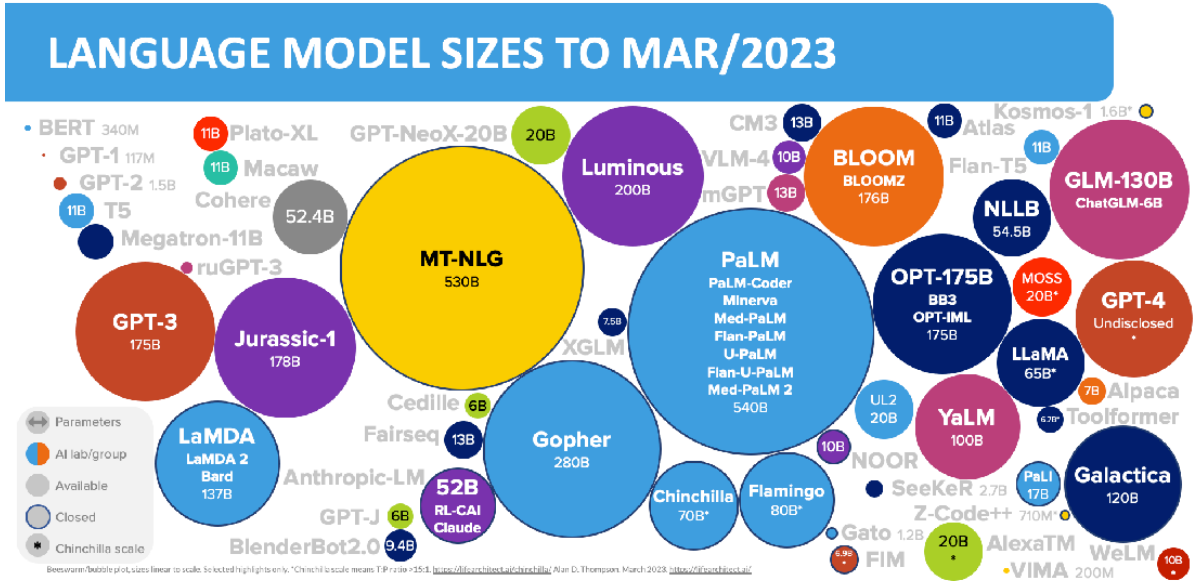


Image: https://lifearchitect.ai/models/

- Transformers-based, **fully parametric**
- Trained on next-token prediction tasks (+ RLHF; not the focus today)
- **Model** size ↑, **data** size↑

When?

A range of target tasks

Question Answering

RETRO (Borgeaud et al., 2021)

REALM (Gu et al, 2020)

ATLAS (Izacard et al, 2023)

Fact verification

RAG (Lewis et al, 2020)

ATLAS (Izacard et al, 2022)

Evi. Generator (Asai et al, 2022)

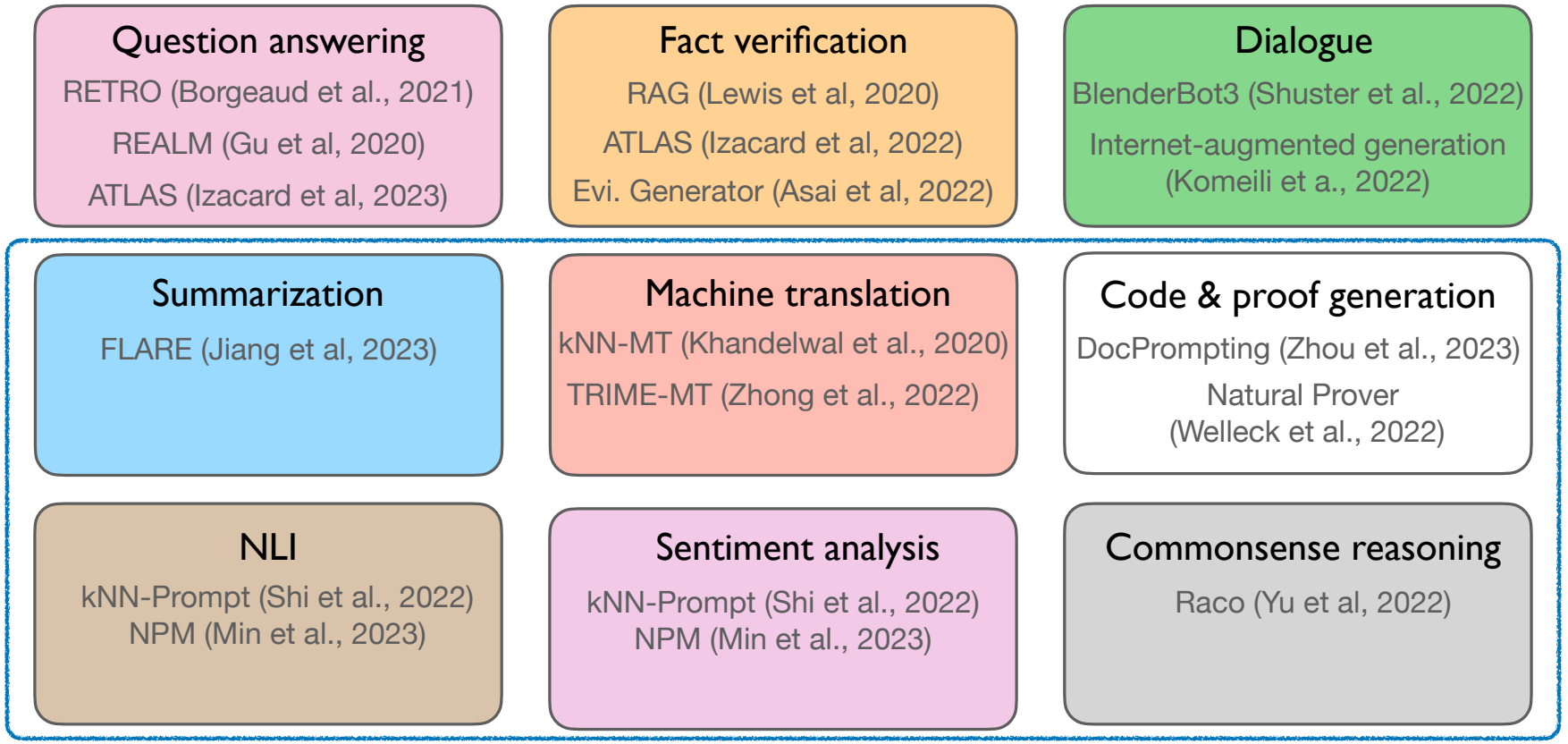
Dialogue

BlenderBot3 (Shuster et al., 2022)

Internet-augmented generation (Komeili et a., 2022)

Retrieval-based LMs have been extensively evaluated on knowledge-intensive tasks

A range of target tasks



More general NLP tasks

Why?

Why retrieval-based LMs?

LLMs can't memorize all (long-tail) knowledge in their parameters

List 5 important papers authored by Geoffrey Hinton

ChatGPT

Geoffrey Hinton is a renowned computer scientist … Here are five important papers authored by him:

1. "**Learning Internal Representations by Error Propagation**" (with **D. E. Rumelhart** and **R. J. Williams**) - This paper, published in **1986**, .. ✓

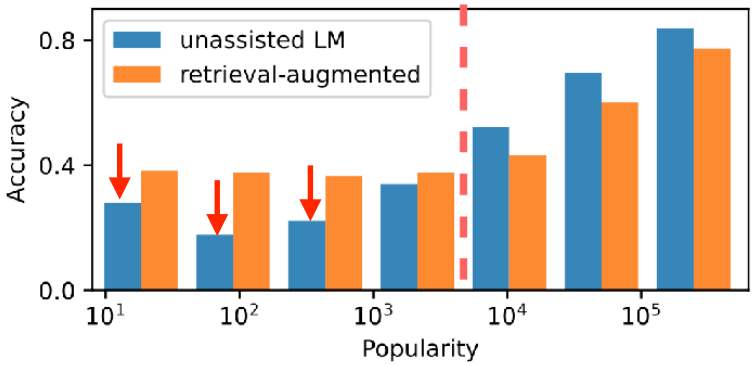
2. "**Deep Boltzmann Machines**" (with **R. Salakhutdinov**) - Published in **2009**, .. ✓

…

4. "**Deep Learning**" (with Y. Bengio and A. Courville) - Published as a book in **2016**,… ✗

5. "**Attention Is All You Need**" (with V. Vaswani, N. Shazeer, et al.) - Published in **2017**, this paper introduced the Transformer model,… ✗

What is Kathy Saltzman's occupation?



(Mallen et al., 2023)

GPT-3 davinci-003: 20%-30% accuracy

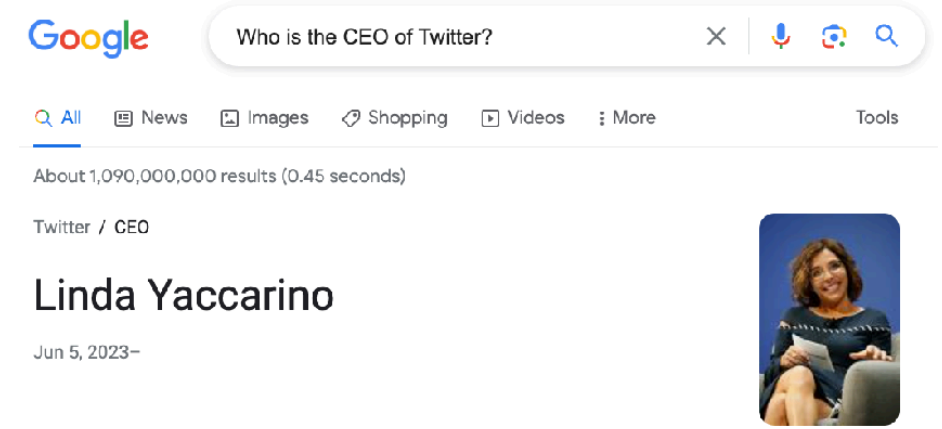
Why retrieval-based LMs?

LLMs' knowledge is easily outdated and hard to update

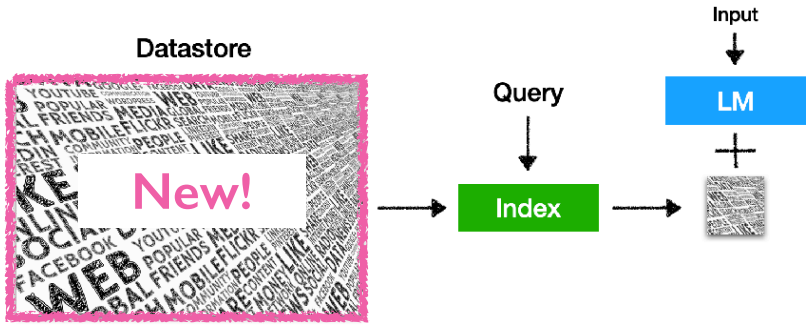
Who is the CEO of Twitter?

ChatGPT

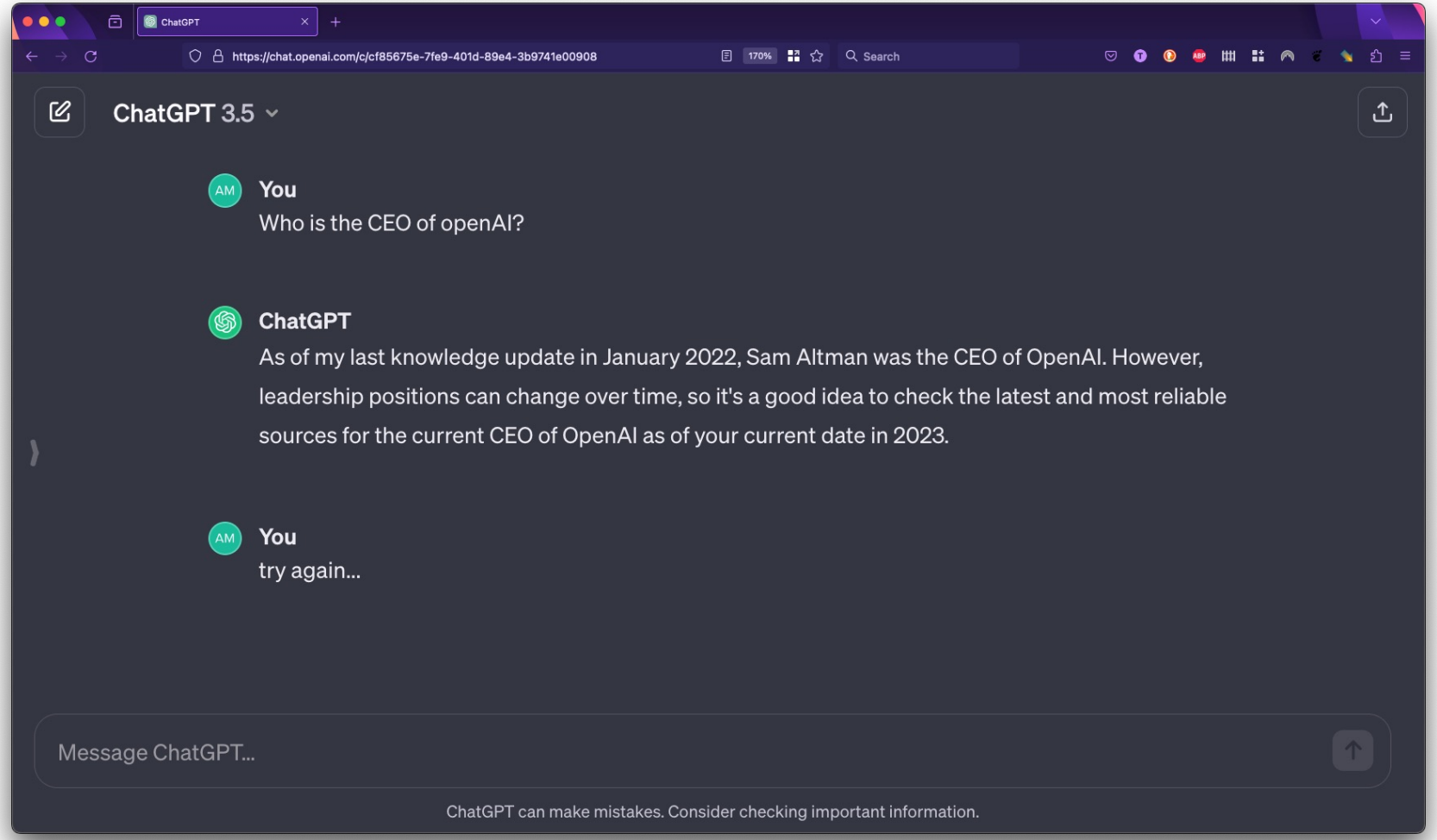
As of my **knowledge cutoff in September 2021**, the CEO of Twitter is **Jack Dorsey**….



- Existing **knowledge editing** methods are still NOT scalable (active research!)
- The datastore can be easily **updated** and **expanded** - even without retraining!

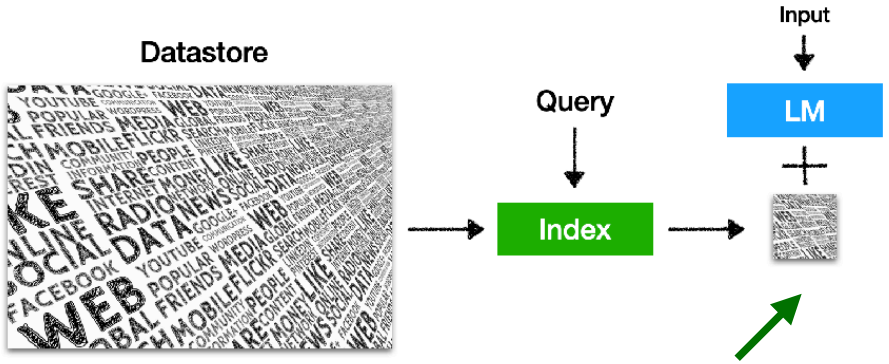


Or in fact…


ChatGPT
https://chat.openai.com/c/cf85675e-7fe9-401d-89e4-3b9741e00908
170%
Search
ChatGPT 3.5
AM
You
Who is the CEO of openAI?
ChatGPT
As of my last knowledge update in January 2022, Sam Altman was the CEO of OpenAI. However, leadership positions can change over time, so it's a good idea to check the latest and most reliable sources for the current CEO of OpenAI as of your current date in 2023.
AM
You
try again...
Message ChatGPT...
ChatGPT can make mistakes. Consider checking important information.

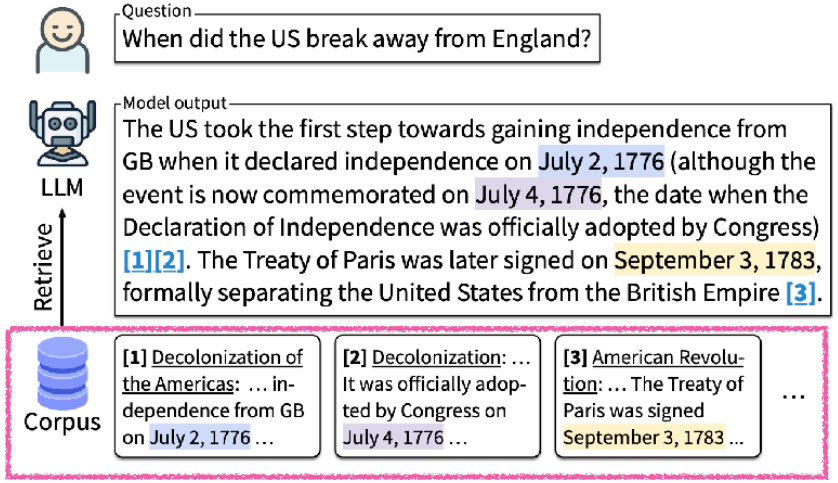
Why retrieval-based LMs?

LLMs' output is challenging to interpret and verify



Can **trace knowledge source** from retrieval results - better **interpretability** & **control**

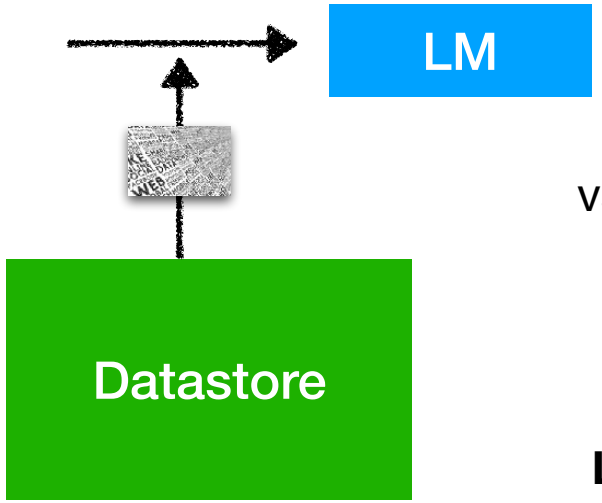
Generating text with citations



(Nakano et al. 2021; Menick et al., 2022; Gao et al., 2023)

Why retrieval-based LMs?

LLMs are *large* and expensive to train and run



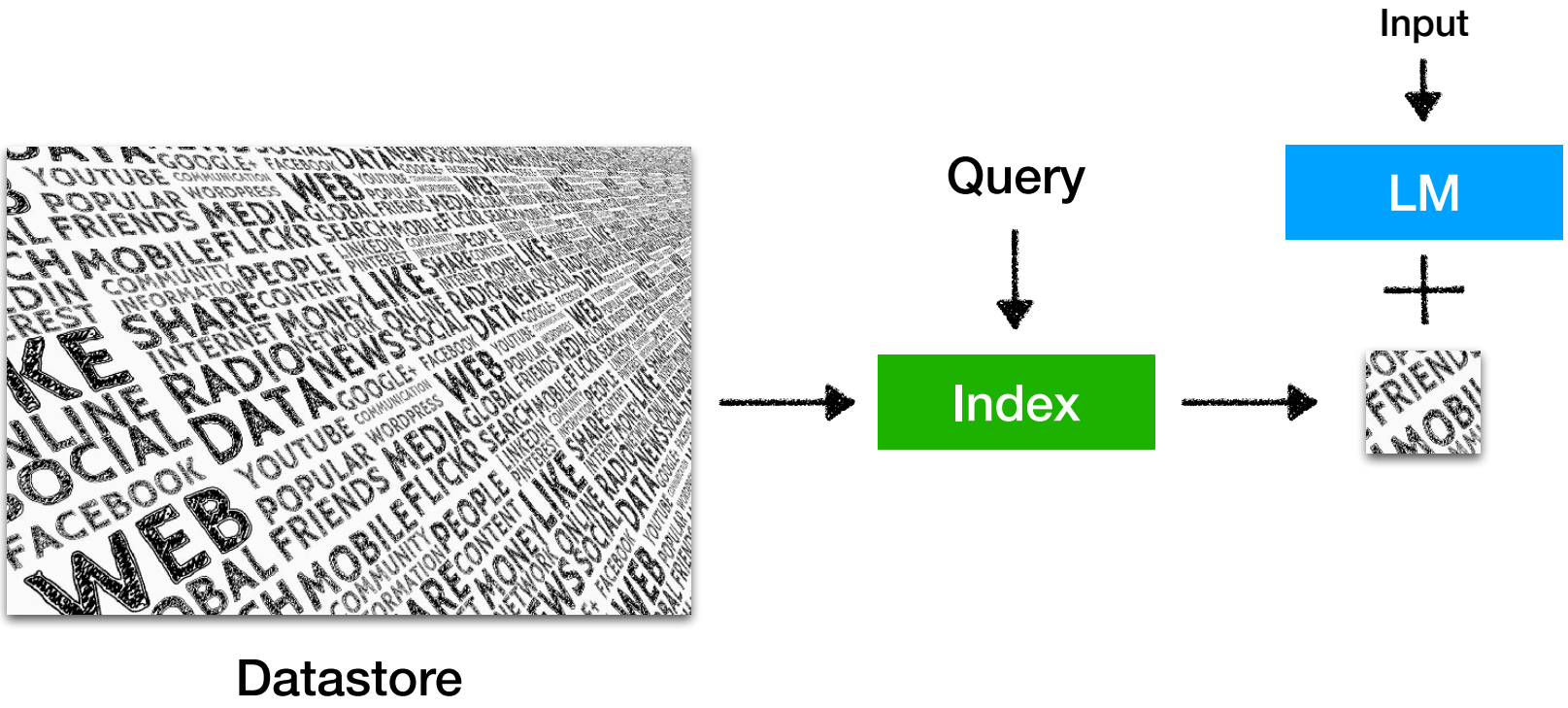
vs.

Long-term goal: can we possibly reduce the **training** and **inference costs,** and scale down the size of LLMs?

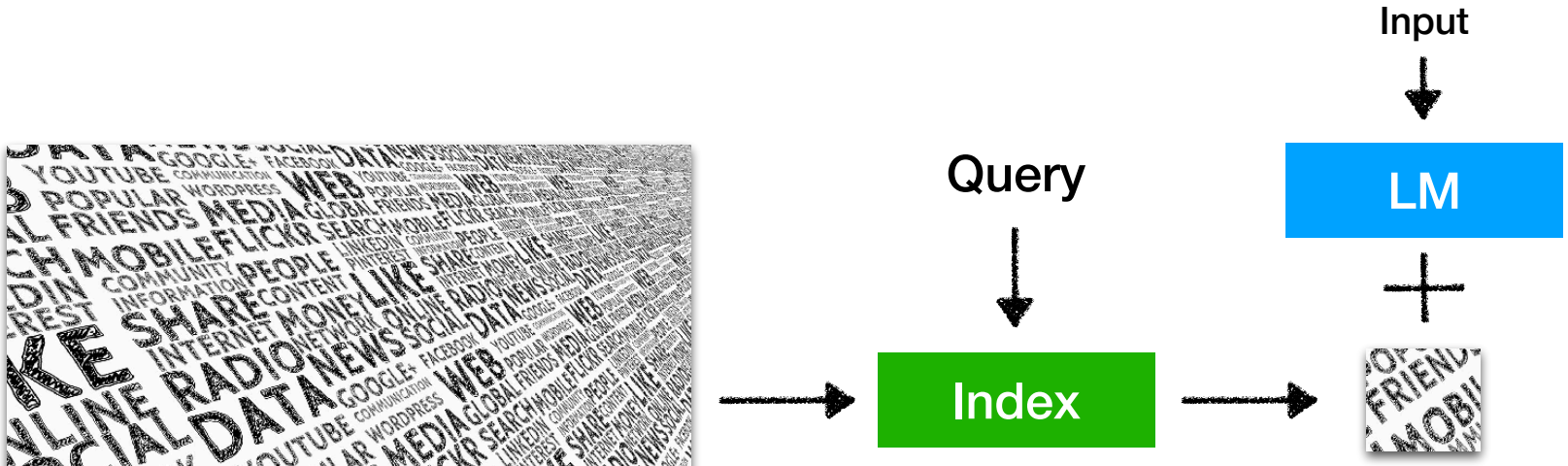
e.g., RETRO (Borgeaud et al., 2021): "obtains comparable performance to GPT-3 on the Pile, despite using **25x fewer parameters**"

How to do it?

Inference



Inference: Datastore



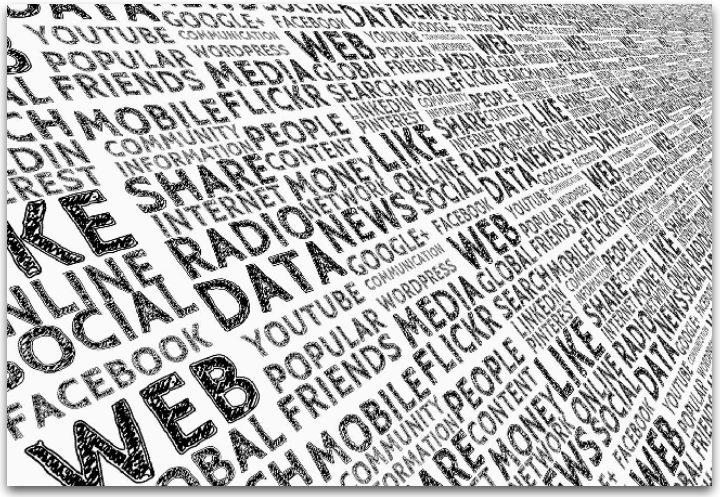
Datastore

Raw text corpus

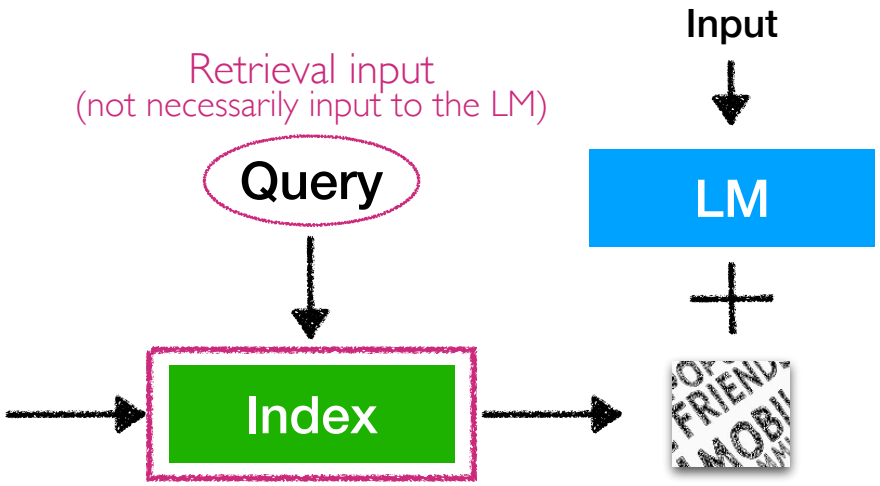
At least billions~trillions of tokens
Not labeled datasets
Not structured data (knowledge bases)

(Extending this definition is possible - some of them in a later part of Section 3)

Inference: Index

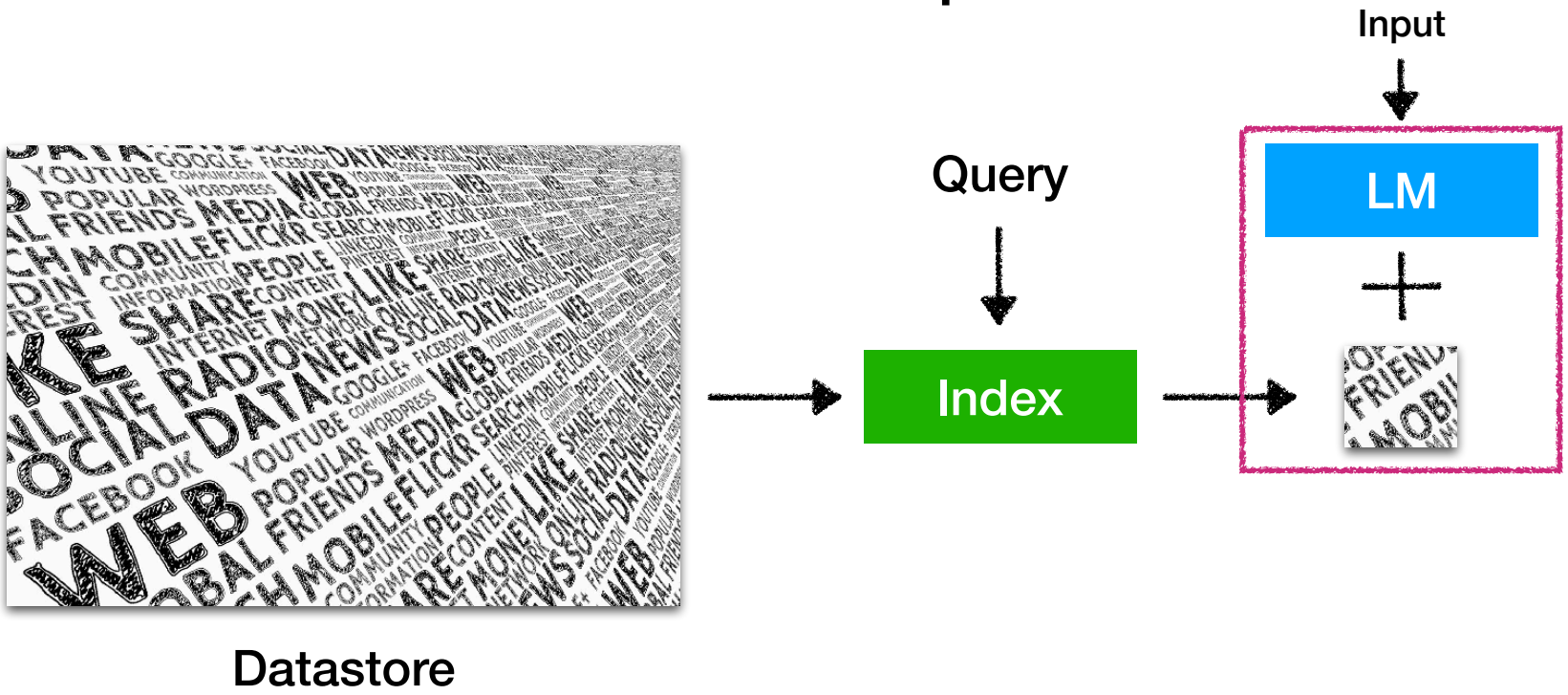


Datastore

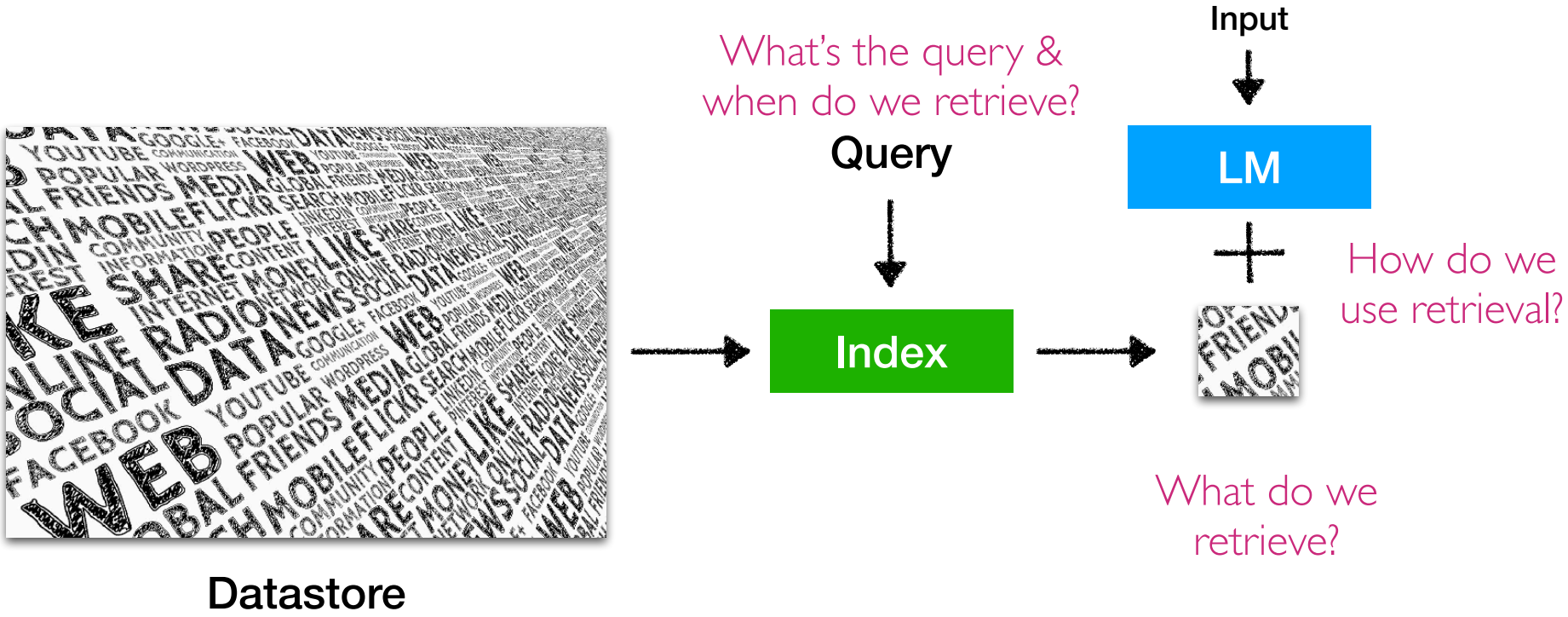


Find a small subset of elements in a datastore that are the most similar to the query

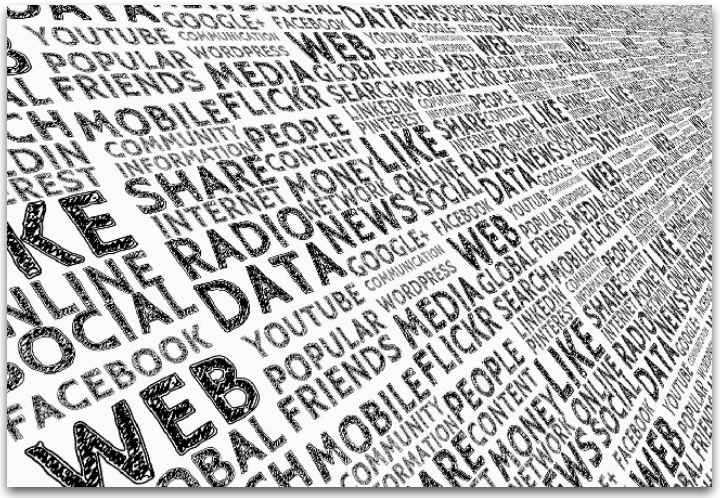
Inference: Incorporation



Questions to answer

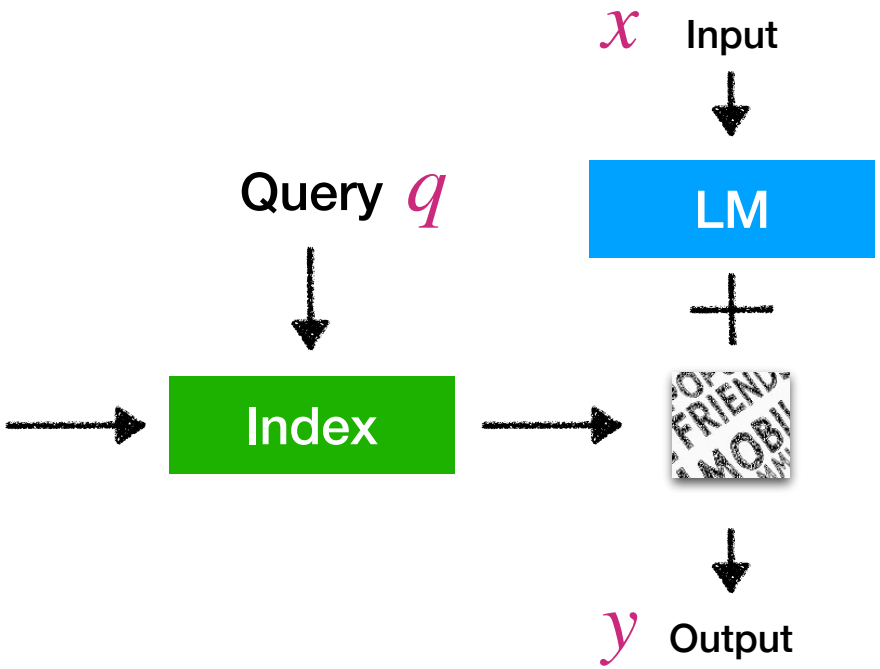


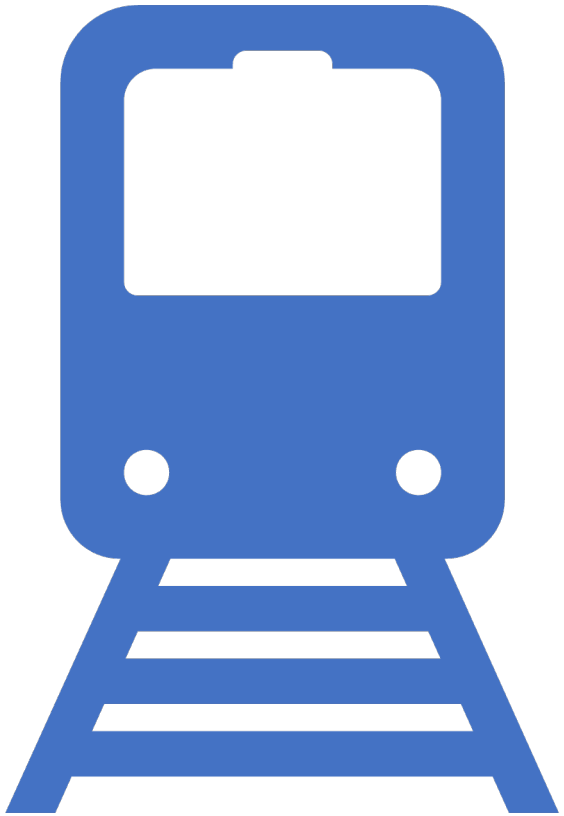
Notations



Datastore

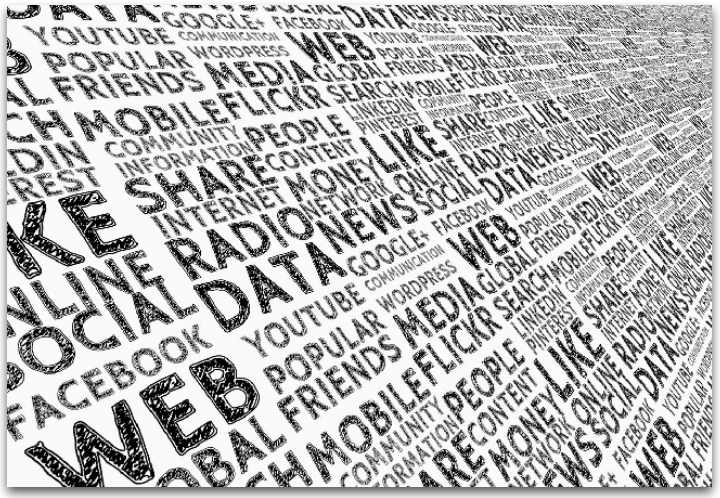
$\mathcal{D}$



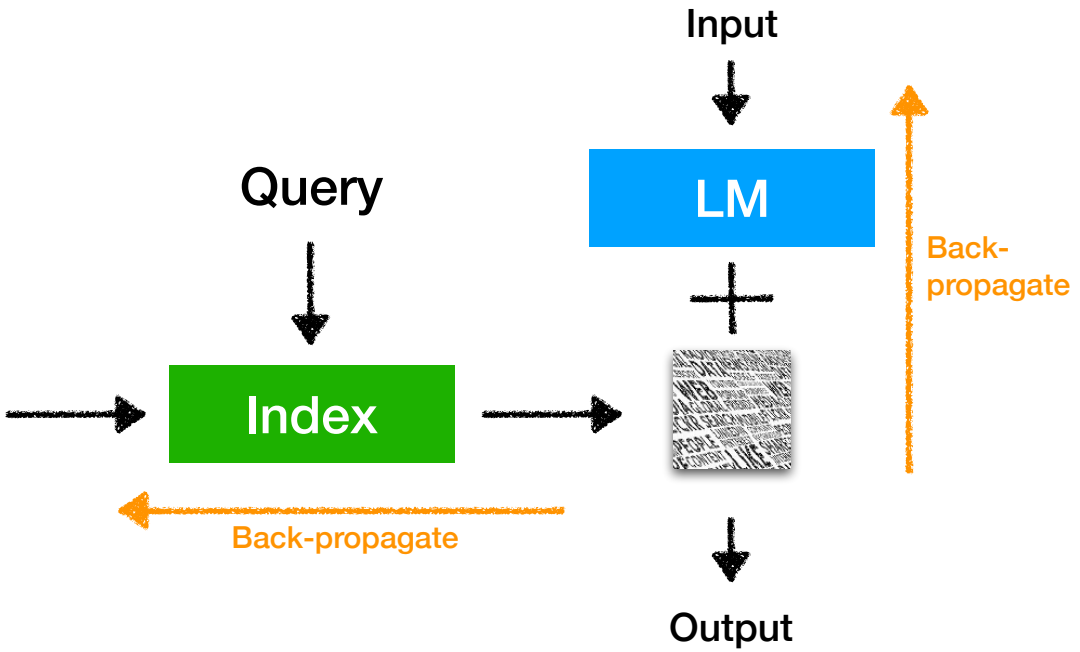


How to train?

Training retrieval-based LMs



Datastore

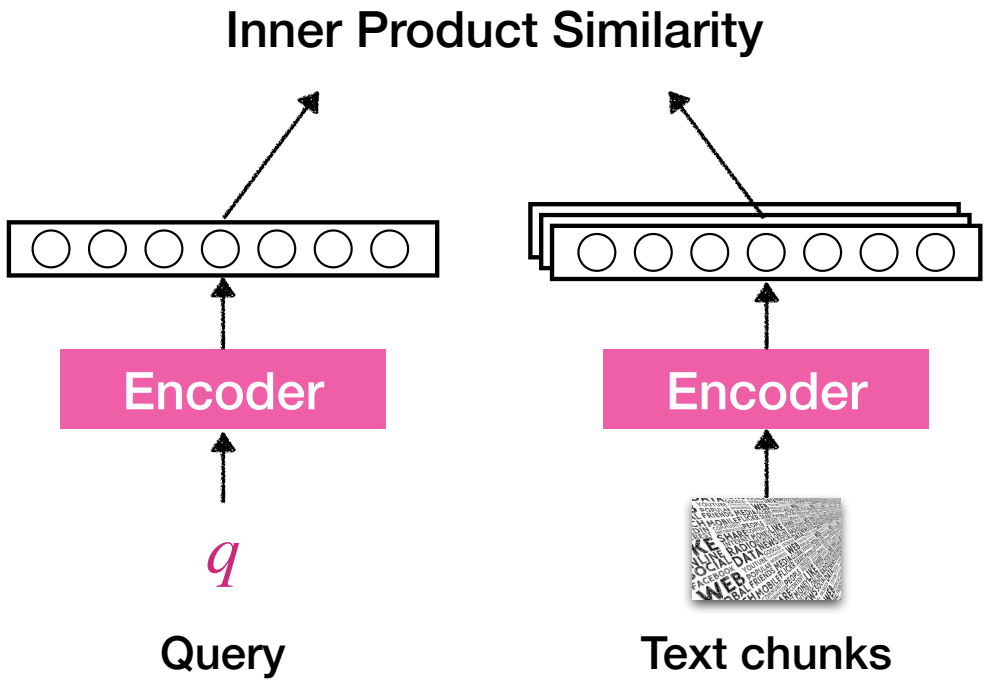


Training language models

Minimize $-\log P_{\mathrm{LM}}(y \mid x)$

GPT PaLM LLaMA GPT-J .....

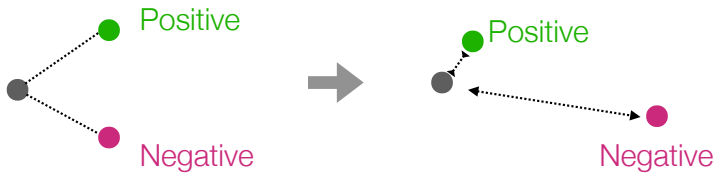
Training dense retrieval models: DPR



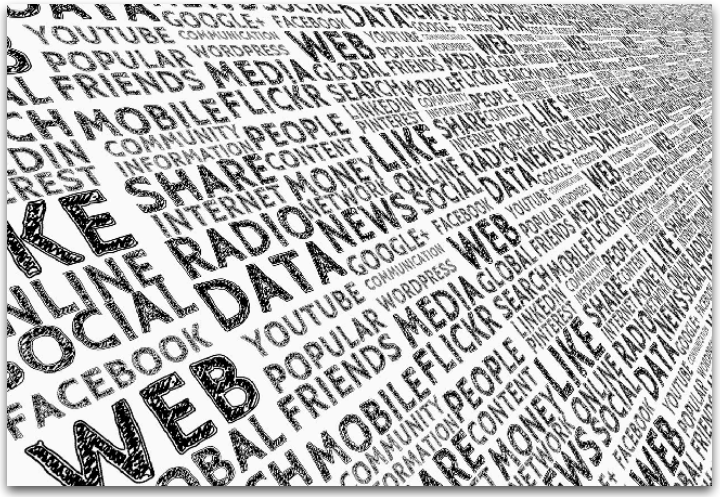
$$L(q, p^+, p_1^-, p_2^-, \ldots, p_n^-)$$

$$= -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^{n} \exp(\text{sim}(q, p_j^-))}$$

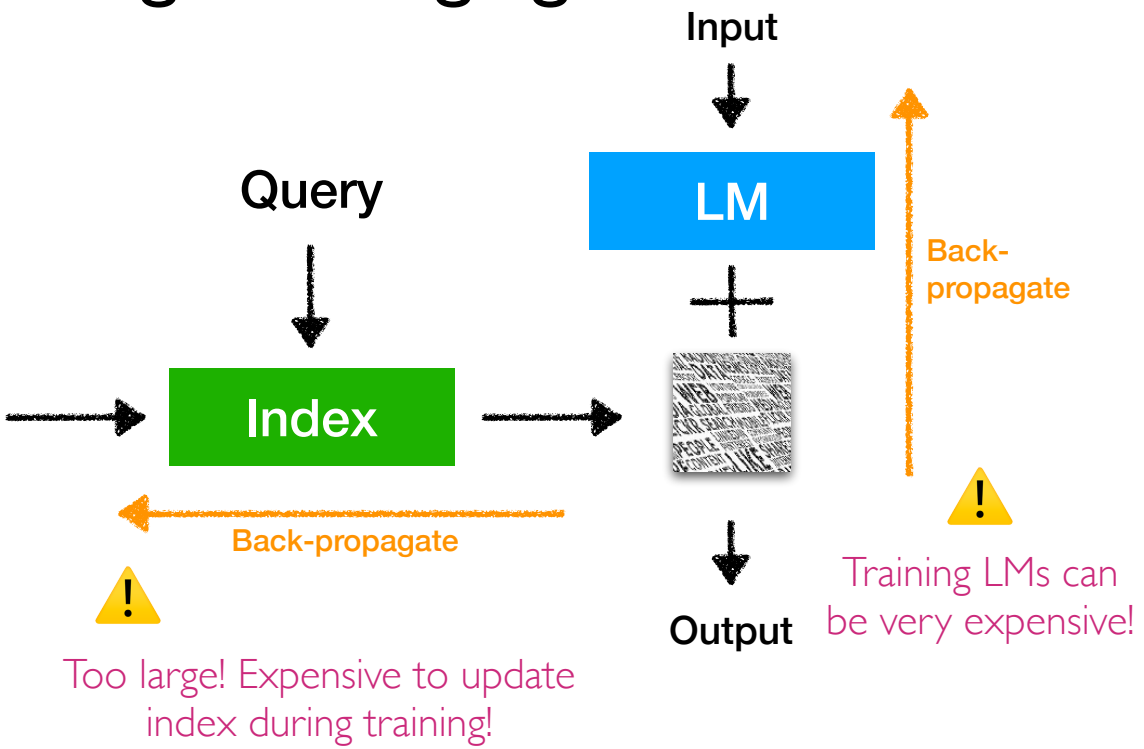
Contrastive learning



Why is training challenging?



Datastore



Training methods for retrieval-based LMs

- Independent training
- Sequential training
- Joint training w/ asynchronous index update
- Joint training w/ in-batch approximation

Training method	👍	👎
Independent training (Ram et al 2023; Khandelwal et al 2020) Sequential training (Borgeaud et al 2021; Shi et al 2023)	✳ Easy to implement: off-the-shelf models ✳ Easy to improve: sub-module can be separately improved	✳ Models are not end-to-end trained — suboptimal performance
Joint training: async update (Guu et al 2020; Izacard et al 2022) Joint training: in-batch approx (Zhong et al 2022; Min et al 2023; Rubin and Berant 2023)	✳ End-to-end trained — very good performance!	✳ Training may be complicated (overhead, batching methods, etc) ✳ Train-test discrepancy still remains

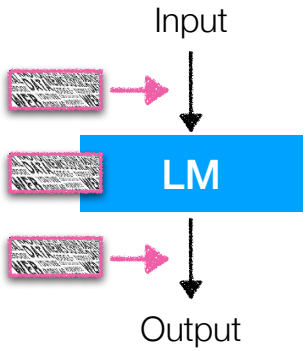
Categorization of retrieval-based LMs

What to retrieve?

Query

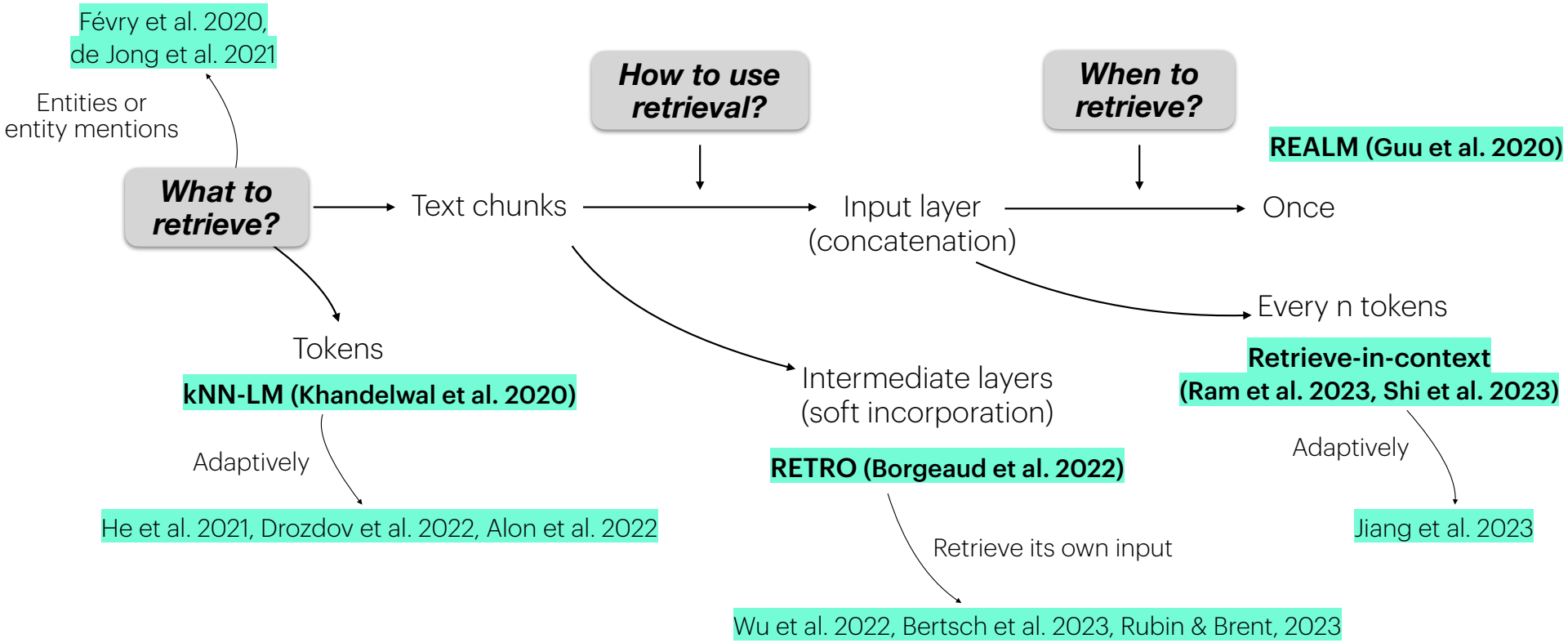
Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?

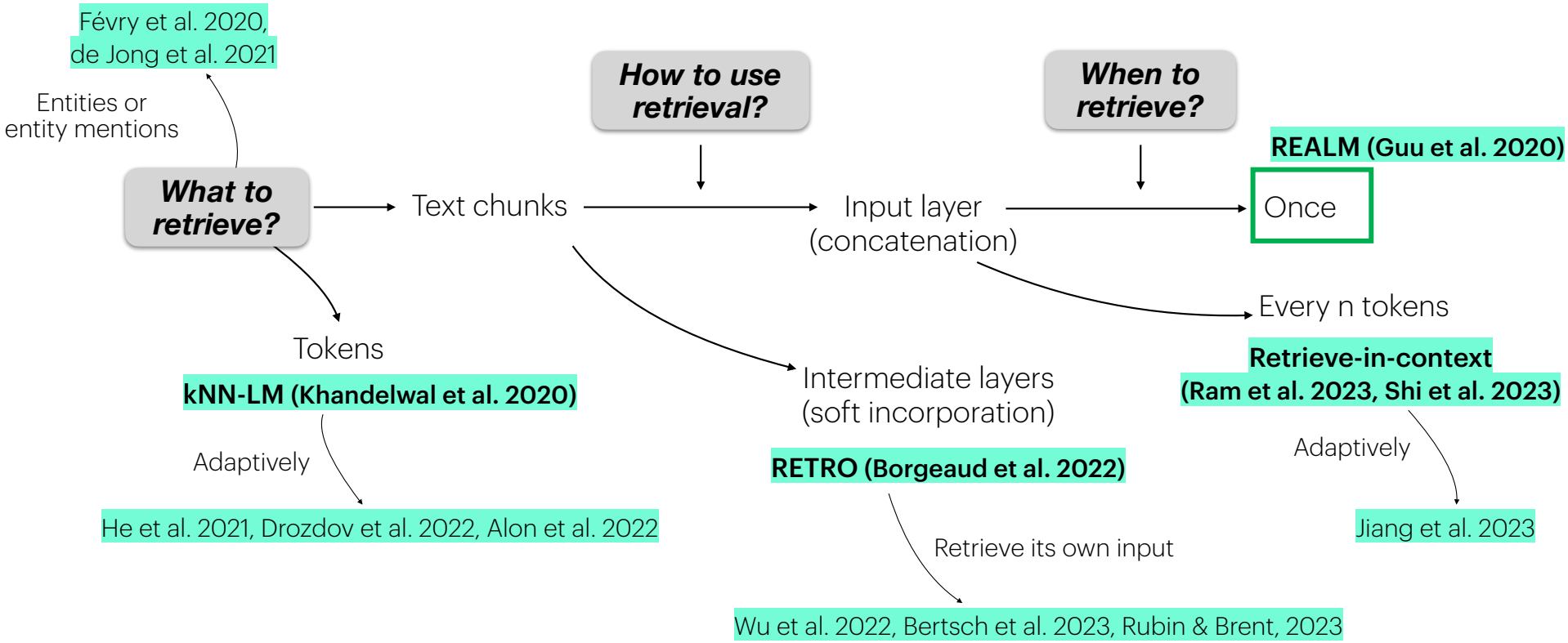


When to retrieve?

Architectures (roadmap)



Architectures (roadmap)



RePLUG: Retrieval-Augmented Black-Box Language Models

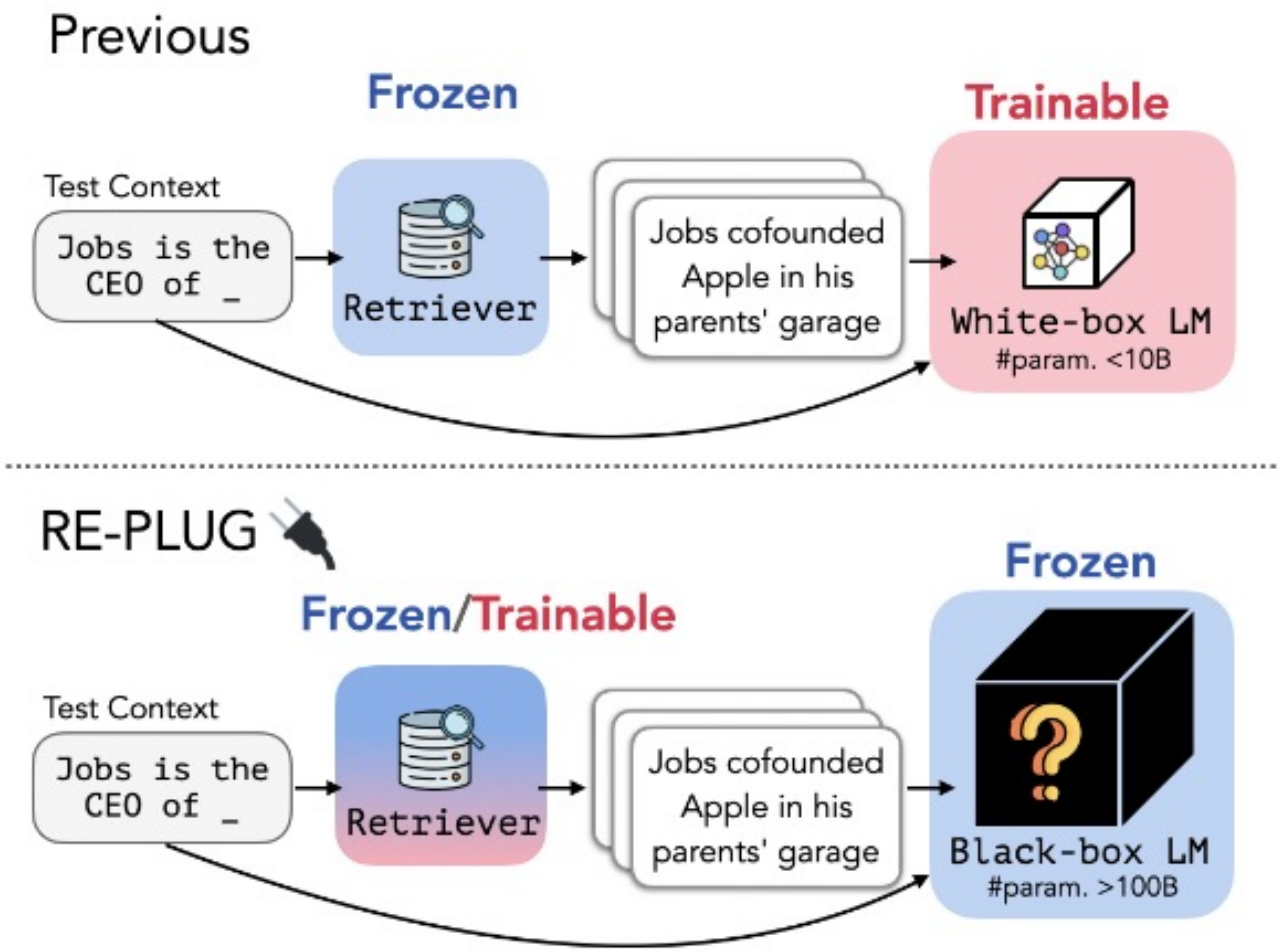
[Shi et.al. 2023]

RePLUG: Retrieval-Augmented Black-Box Language Models [Shi et.al. 2023]

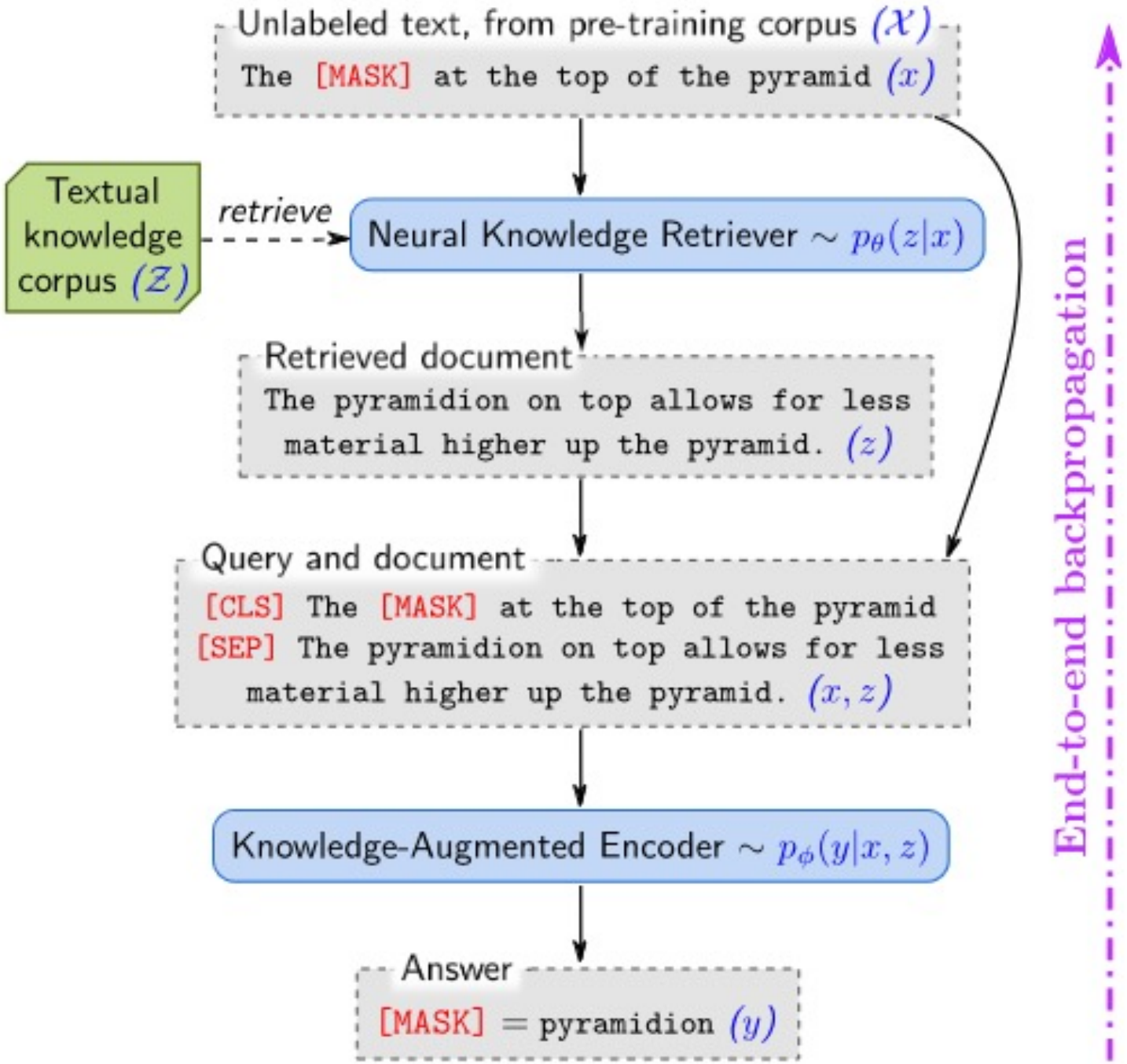
- decoder-only LM
- adding documents in the prompt as retrieval augmentation
  - no LM finetuning needed
  - [depends on the LM]

open issues:

- handling long/many documents
  - positional interpolation?
    - lost-in-middle effects
- limited controllability of LM (only via instructions/prompting)



REALM
[Guu et.al. 2020]

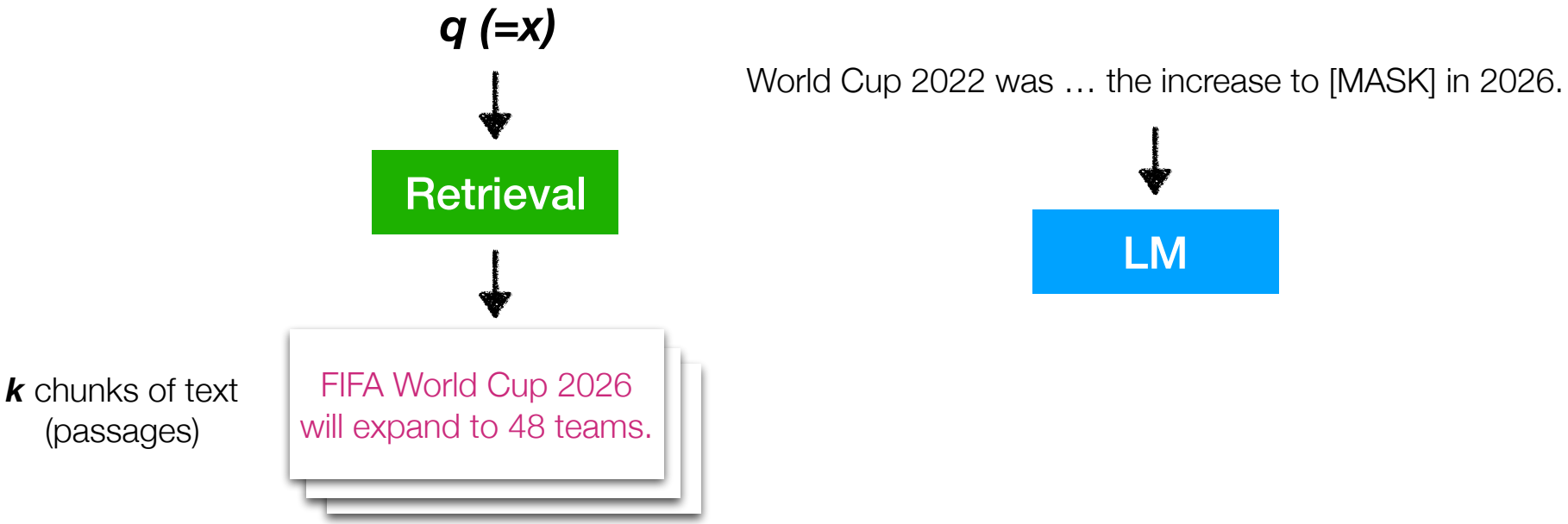


REALM
[Guu et.al. 2020]

- encoder only LM
  - BERT based
- task: QA
  - assumes answer is found as is in documents
  - predicts [answer_start], [answer_end]
- marginalizes over likelihood of document relevance and answer start/end

REALM (Guu et al 2020)

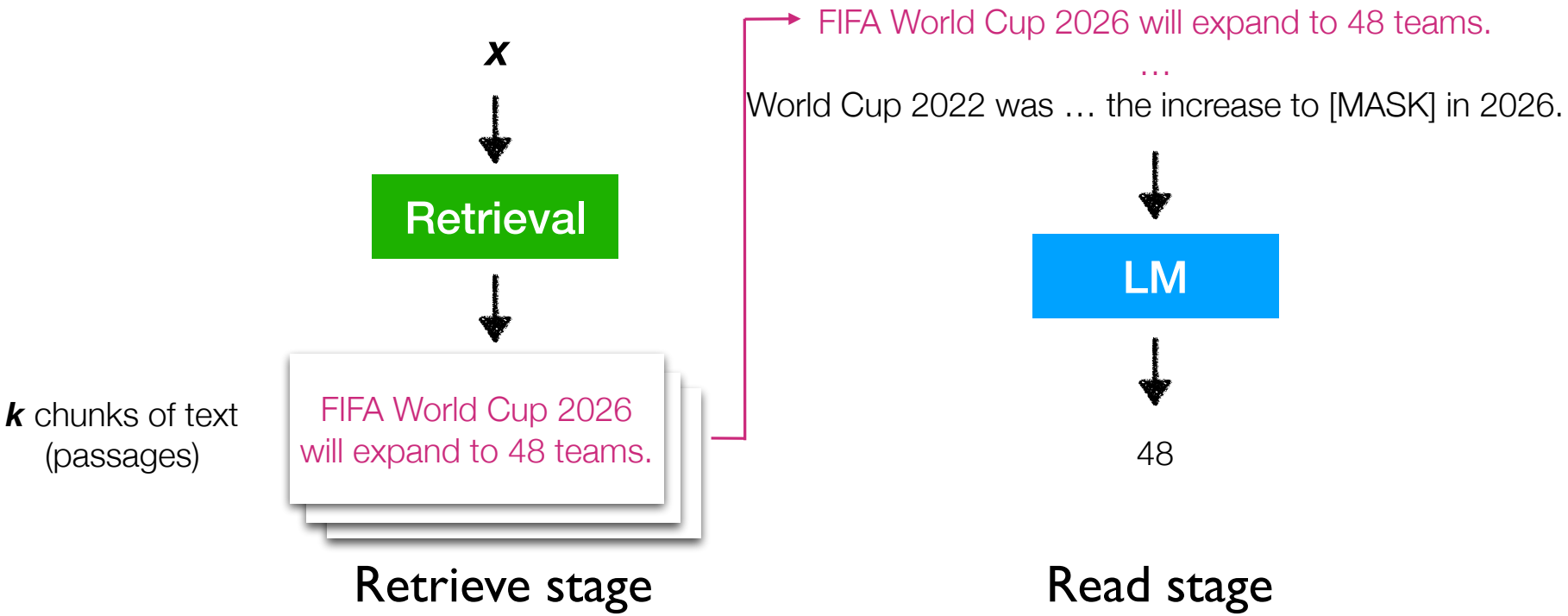
x = World Cup 2022 was the last with 32 teams before the increase to [MASK] in 2026.



Guu et al. 2020. “REALM: Retrieval-Augmented Language Model Pre-Training”

REALM (Guu et al 2020)

x = World Cup 2022 was the last before the increase to [MASK] in the 2026 tournament.



Guu et al. 2020. "REALM: Retrieval-Augmented Language Model Pre-Training"

REALM: (1) Retrieve stage

x = World Cup 2022 was … the increase to [MASK] in 2026.

FIFA World Cup 2026 will expand to 48 teams.

In 2022, the 32 national teams involved in the tournament.

Team USA celebrated after winning its match against Iran …

Wikipedia
13M chunks (passages)
(called *documents* in the paper)

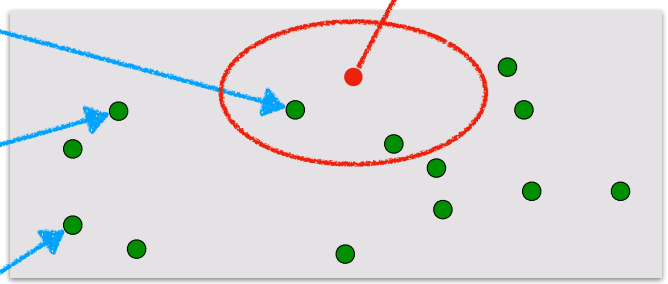
Encoder

Encoder

Encoder

Encoder

Fast nearest neighbor search



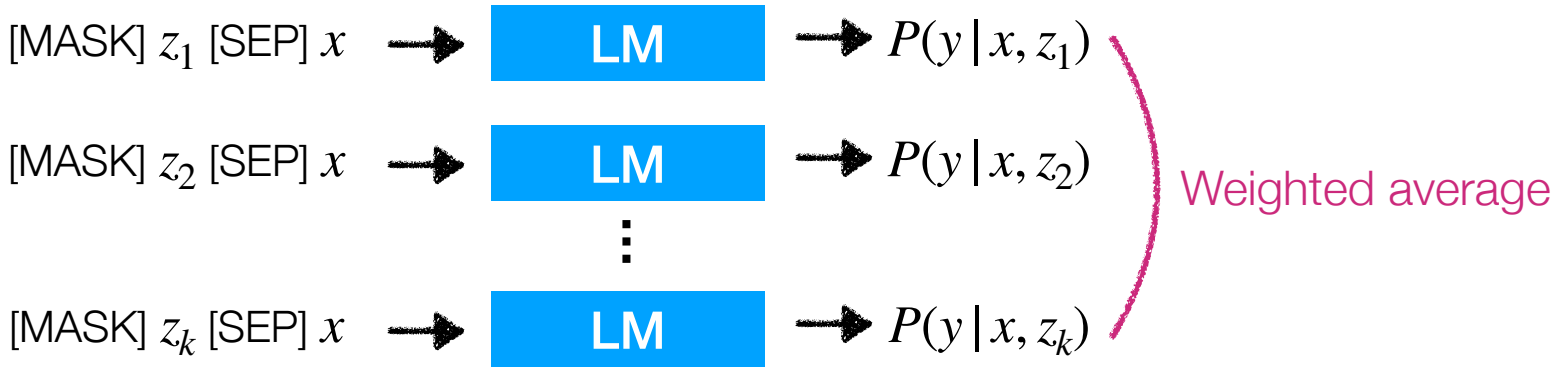
$$\mathbf{z} = \text{Encoder}(z)$$

$$\mathbf{x} = \text{Encoder}(x)$$

$$z_1, \ldots, z_k = \text{argTop-}k\,(\mathbf{x} \cdot \mathbf{z})$$

k retrieved blocks

REALM: (2) Read stage



0 if not one of top k

$$\sum_{z \in \mathcal{D}} P(z \mid x) P(y \mid x, z)$$

Need to approximate
→ Consider top k chunks only

$z \in \mathcal{D}$

$P(z \mid x)$: from the retrieve stage

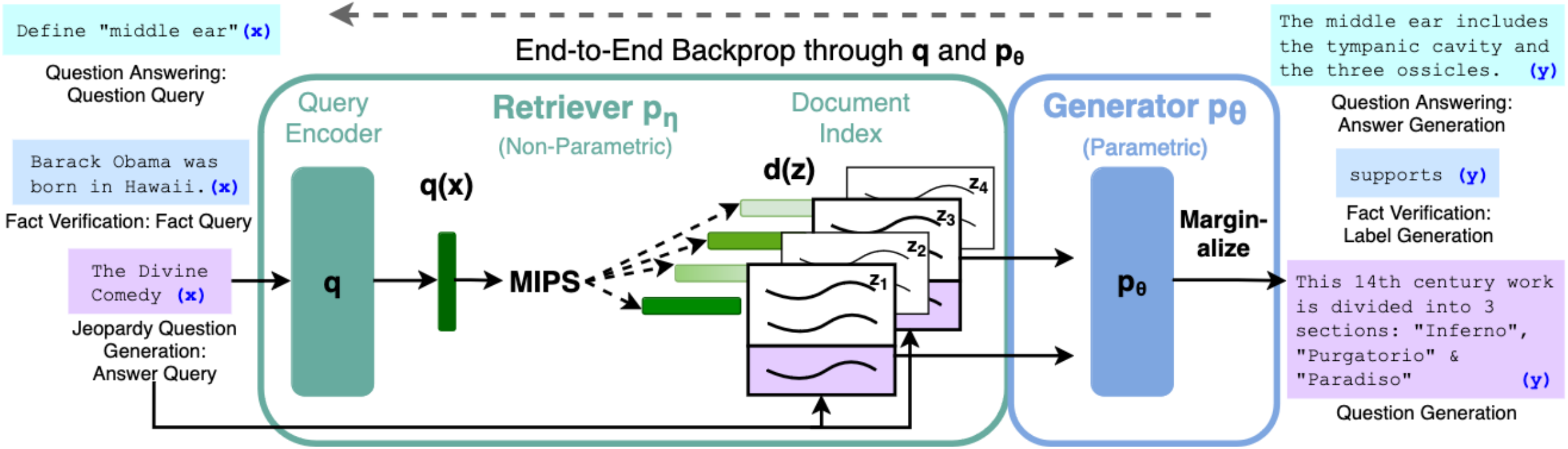
$P(y \mid x, z)$: from the read stage

Retrieval-Augmented-Generation

[Lewis et.al. 2020]

Retrieval-Augmented-Generation [Lewis et.al. 2020]

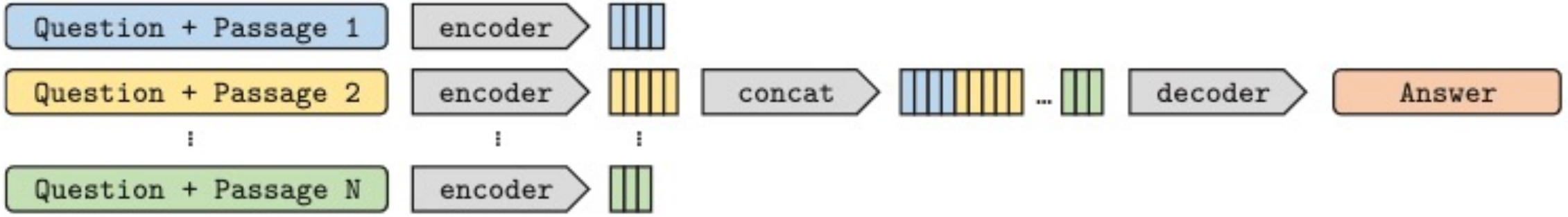
- encoder-decoder model (BART) for generation
- training:
  - retriever (DPR): index frozen; training only the query encoder, based on the probability the GT answer is generated given this document
  - LM: next token prediction



Fusion-in-Decoder (FiD) [Izacard et.al. 2020]

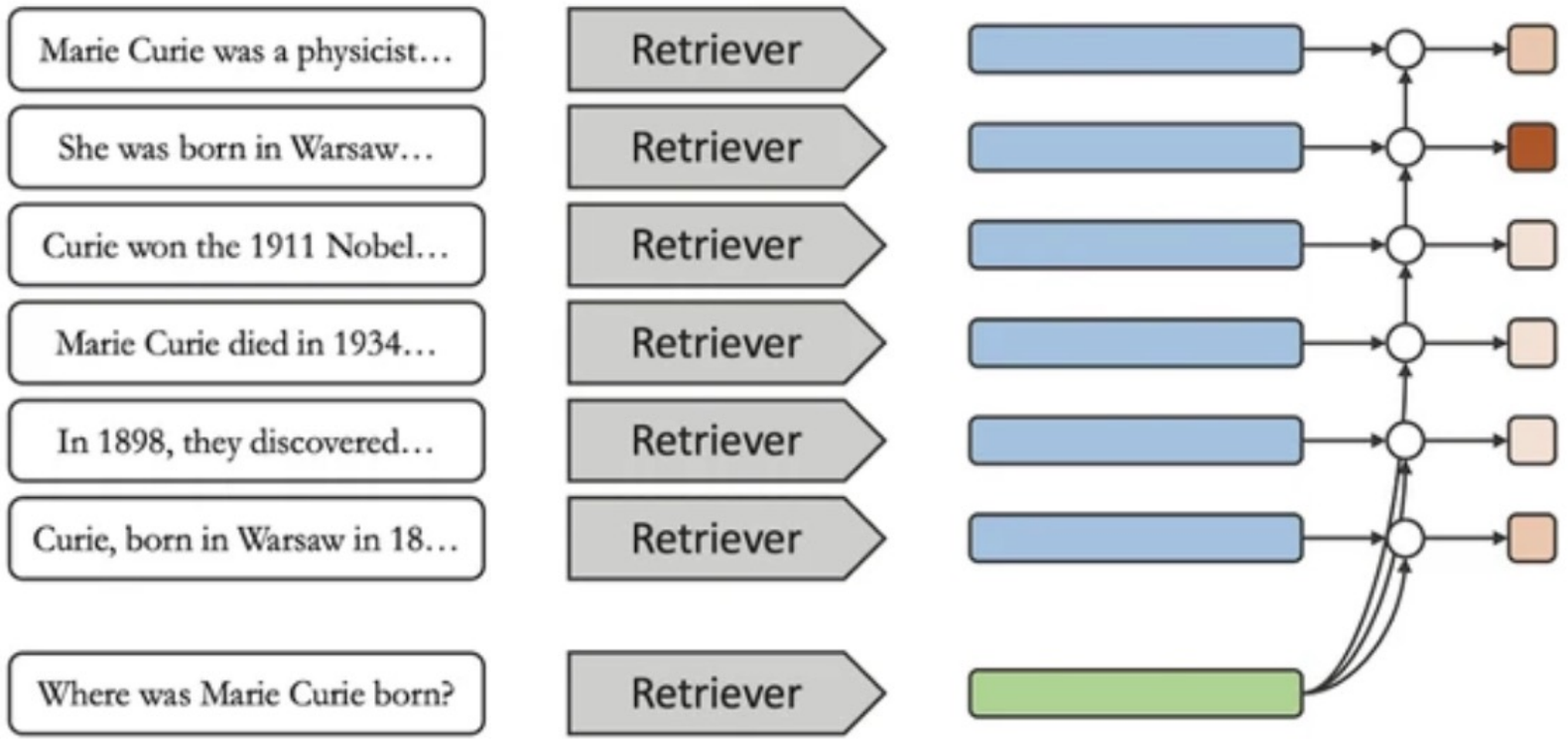
Fusion-in-Decoder (FiD) [Izacard et.al. 2020]

- similar to RAG, but
  - T5 instead of BART; retriever is static
  - encode **all documents at once**, pass document embeddings to decoder
  - outperforms RAG on QA
    - why??



Fusion-in-Decoder Knowledge-Distillation (FiD-KD) [Izacard et.al. 2021]

How to train the retriever?



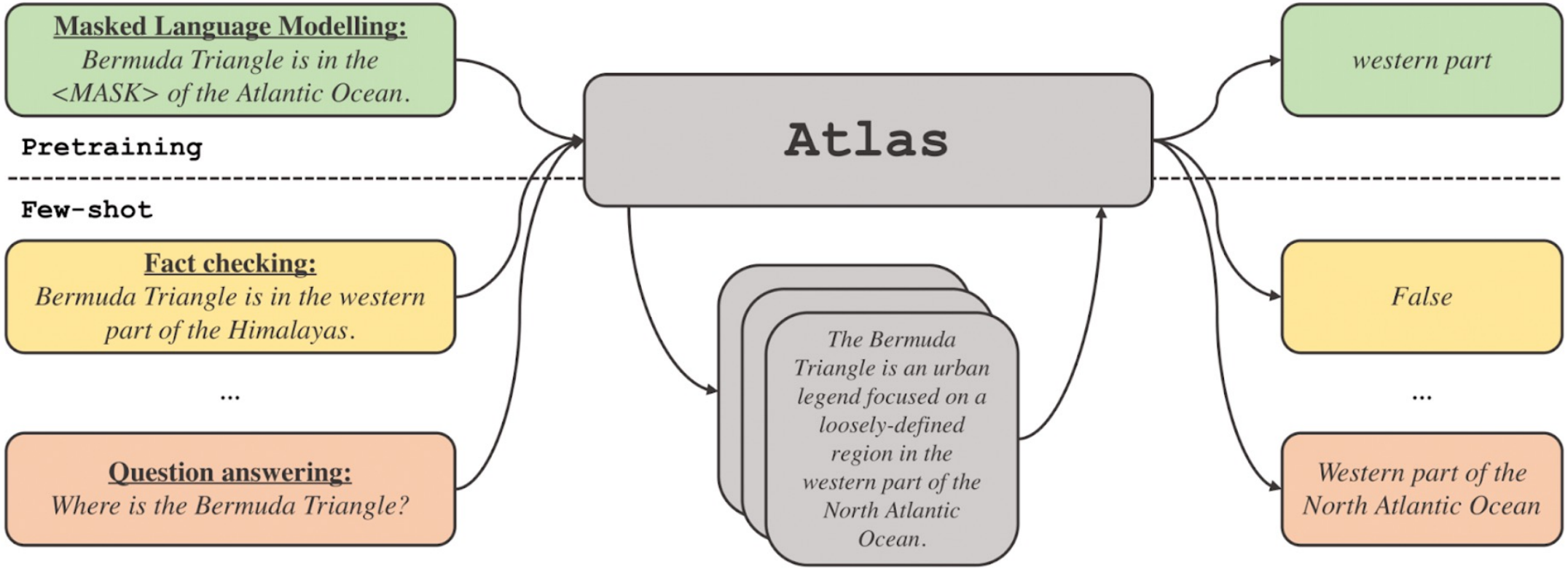
Fusion-in-Decoder Knowledge-Distillation (FiD-KD) [Izacard et.al. 2021]

How to train the retriever?

- no document relevance judgements needed; only QA pairs
- distil training signal (reader → retriever) :
  - Measure “contribution” of each document while generating answer (LM)

Few-shot learning with RAG models (Atlas) [Izacard et.al. 2022]

joint retriever/reader pre-training & few shot capabilities



Few-shot learning with RAG models (Atlas) [Izacard et.al. 2022]

joint retriever/reader pre-training & few shot capabilities

- joint unsupervised pretraining (ICT) of retriever & reader
  - instead of iteratively freezing & training each model
- few-shot (64 examples) finetuning: decent performance

Atlas: Retriever training

Similarity based on retrieval encoder

$$P_{\text{retr}}(z \mid q) = \frac{\exp(s(z, q))}{\sum_{k=1}^{K} \exp(s(z_k, q))}$$

How likely each document is retrieved

KL Divergence

Prob of the gold labels if augmenting this text chunk

$$P_{\text{ppl}}(z \mid q, y) = \frac{\exp(\log P_{\text{LM}}(y \mid q, z))}{\sum_{k=1}^{K} \exp(\log P_{\text{LM}}(y \mid q, z_k))}$$

How much each document improves the ppl

Perplexity Distillation

Retrieve the text that can help LM encoders improve perplexity

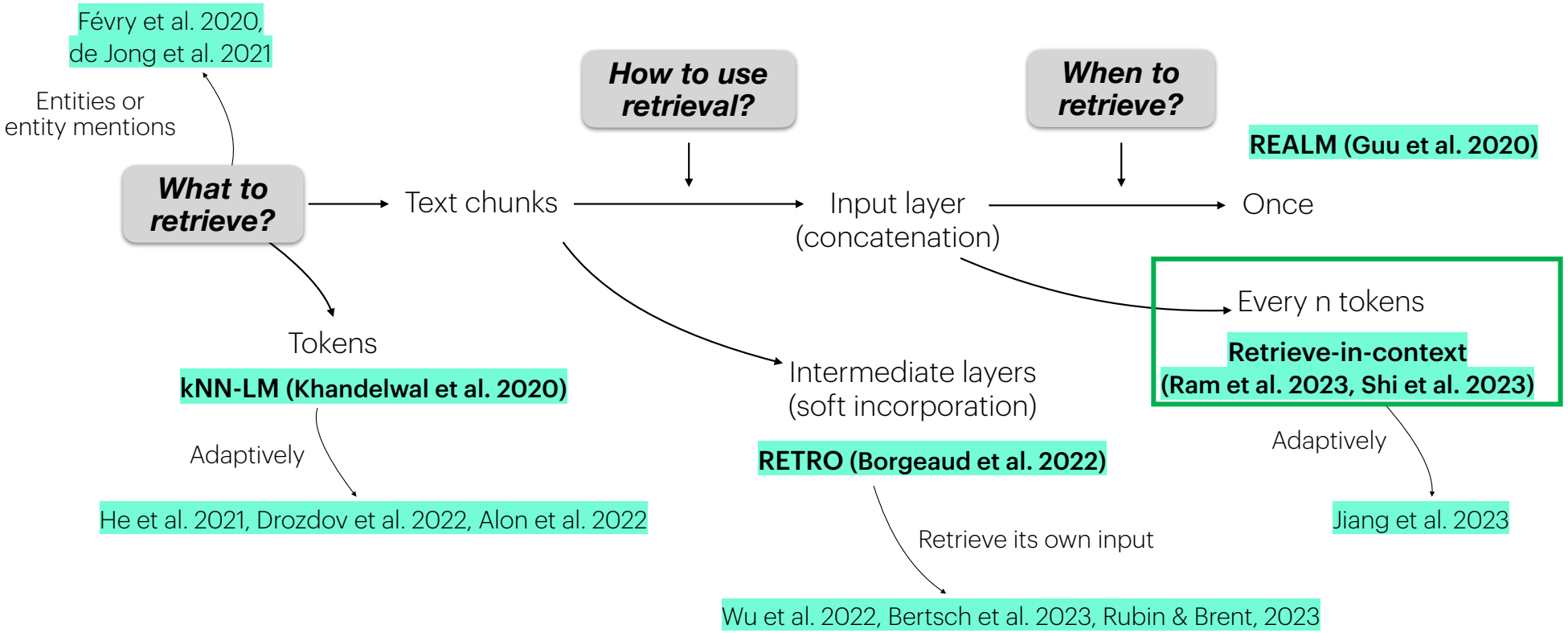
Model	NQ		TriviaQA filtered		TriviaQA unfiltered	
	64-shot	Full	64-shot	Full	64-shot	Full
GPT-3 (Brown et al., 2020)	29.9	-	-	-	71.2	-
Gopher (Rae et al., 2021)	28.2	-	57.2	-	61.3	-
Chinchilla (Hoffmann et al., 2022)	35.5	-	64.6	-	72.3	-
PaLM (Chowdhery et al., 2022)	39.6	-	-	-	81.4	-
RETRO (Borgeaud et al., 2021)	-	45.5	-	-	-	-
FiD (Izacard & Grave, 2020)	-	51.4	-	67.6	-	80.1
FiD-KD (Izacard & Grave, 2021)	-	54.7	-	73.3	-	-
R2-D2 (Fajcik et al., 2021)	-	55.9	-	69.9	-	-
ATLAS	**42.4**	**60.4**	**74.5**	**79.8**	**84.7**	**89.4**

Where do we stand?

Steps that seem to help:

- improving retriever (reader distillation, DPR → Contriever)
- iterative/joint pretraining

Architectures (roadmap)



Retrieval-in-context in LM

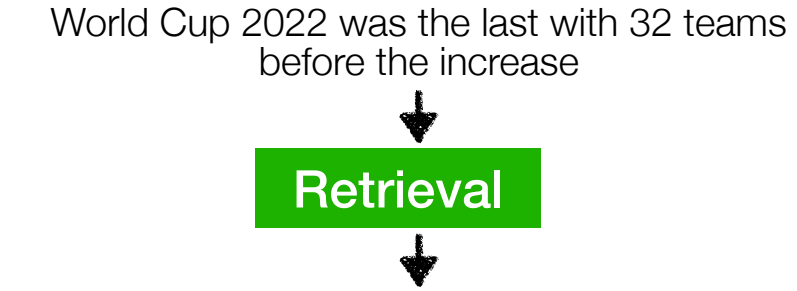
How frequent should retrieval be?

Retrieval-in-context in LM

How frequent should retrieval be?

Retrieval-in-context in LM

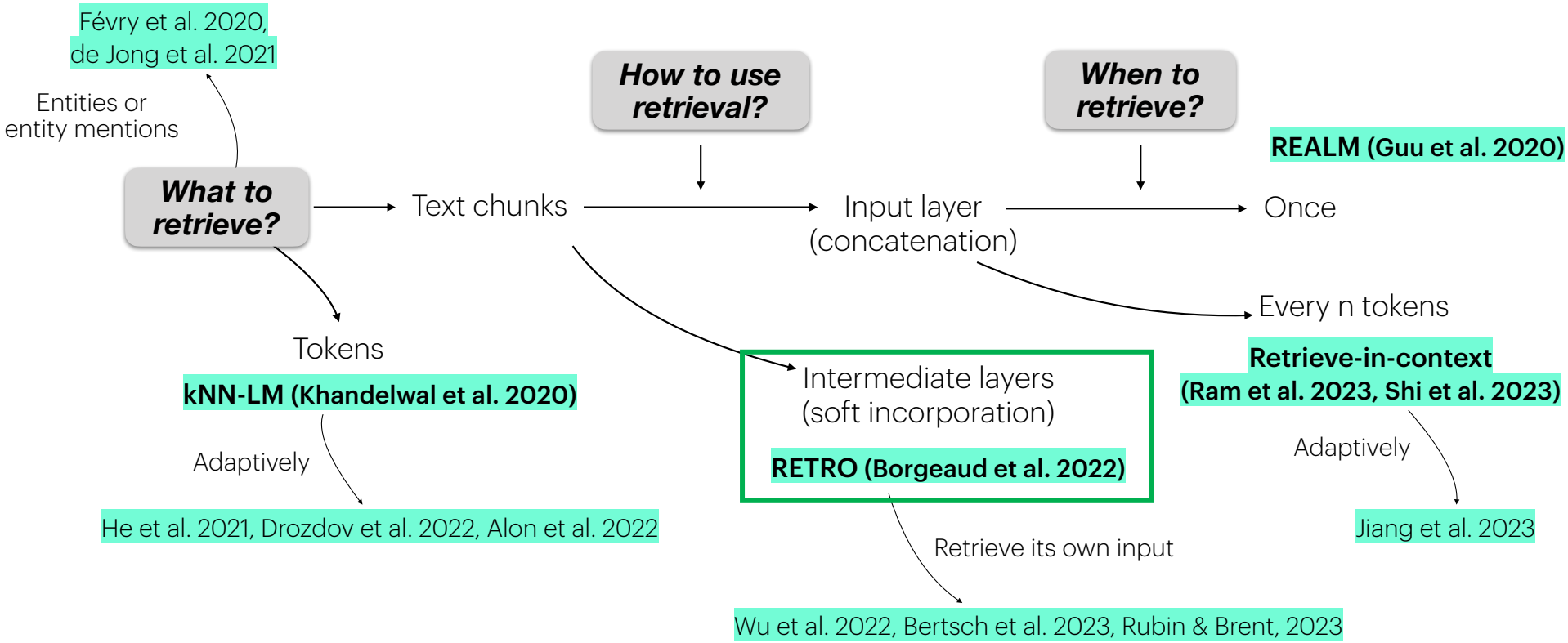
How frequent should retrieval be?



Retrieval-in-context in LM

How frequent should retrieval be?

Architectures (roadmap)



RETRO

Beyond simple concatenation of retrieved passages to the LM

RETRO (Borgeaud et al. 2021)

✓ Incorporation in the “intermediate layer” instead of the “input” layer
→ designed for *many* blocks, *frequently*, more *efficiently*

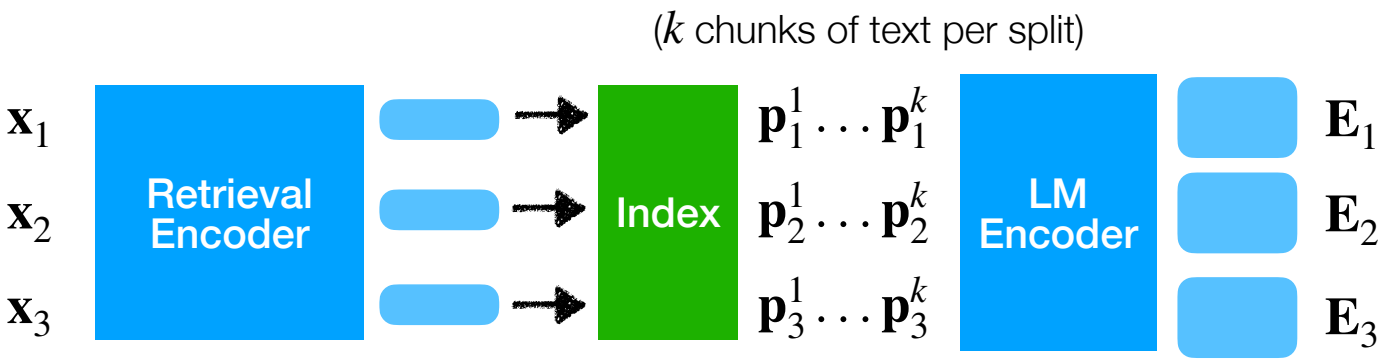
✓ Scale the datastore to retrieve from

Borgeaud et al. 2021. “Improving language models by retrieving from trillions of tokens”

RETRO (Borgeaud et al. 2021)

x = World Cup 2022 was the last with 32 teams, before the increase to

$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$



(A $r \times k \times d$ matrix)

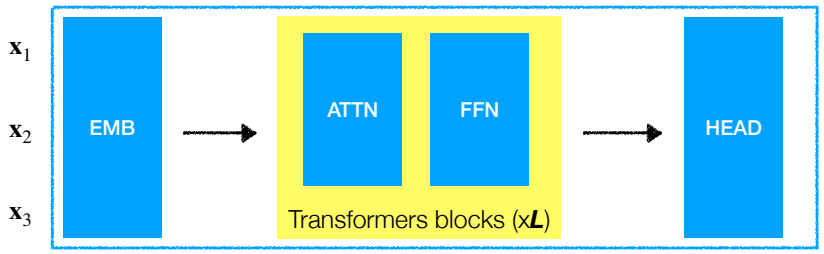
(r = # tokens per text block)

(d = hidden dimension)

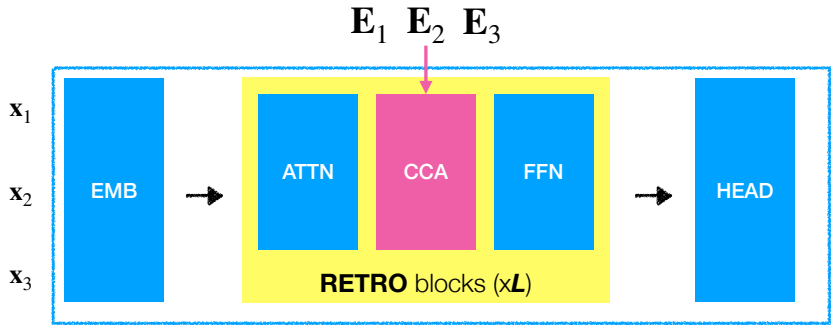
(k = # retrieved blocks per split)

RETRO Chunked Cross Attention architecture

Regular decoder

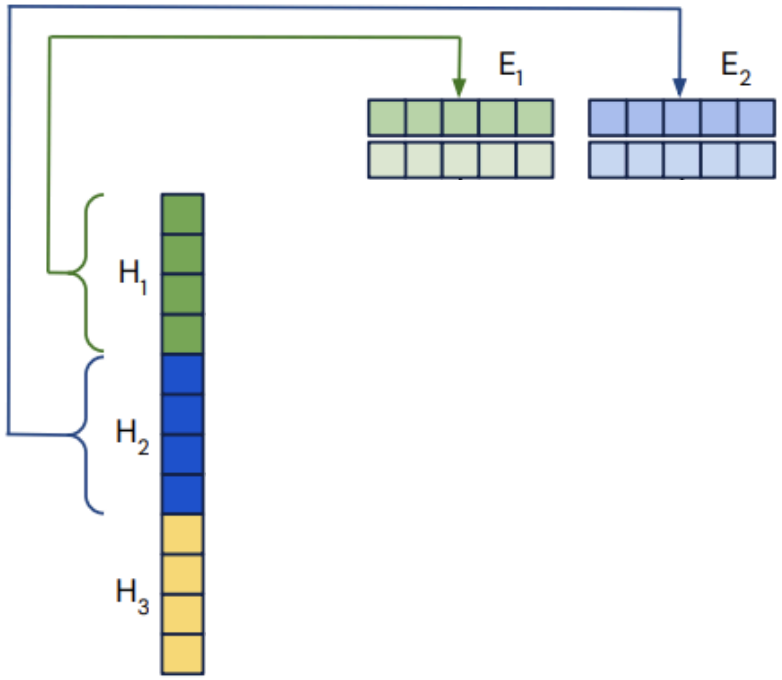


Decoder in RETRO



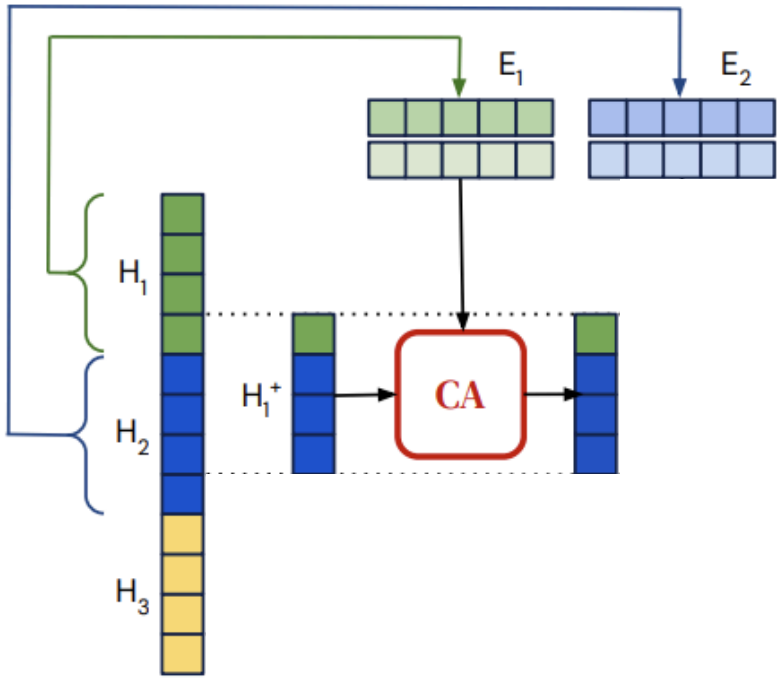
Chunked Cross Attention (CCA)

Chunked Cross Attention



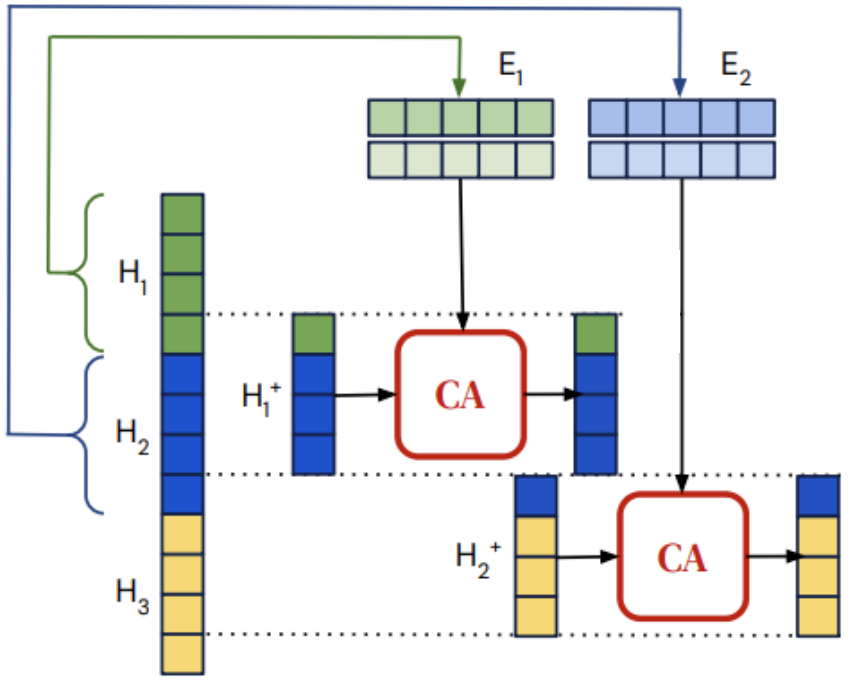
Outputs from the previous layer

Chunked Cross Attention



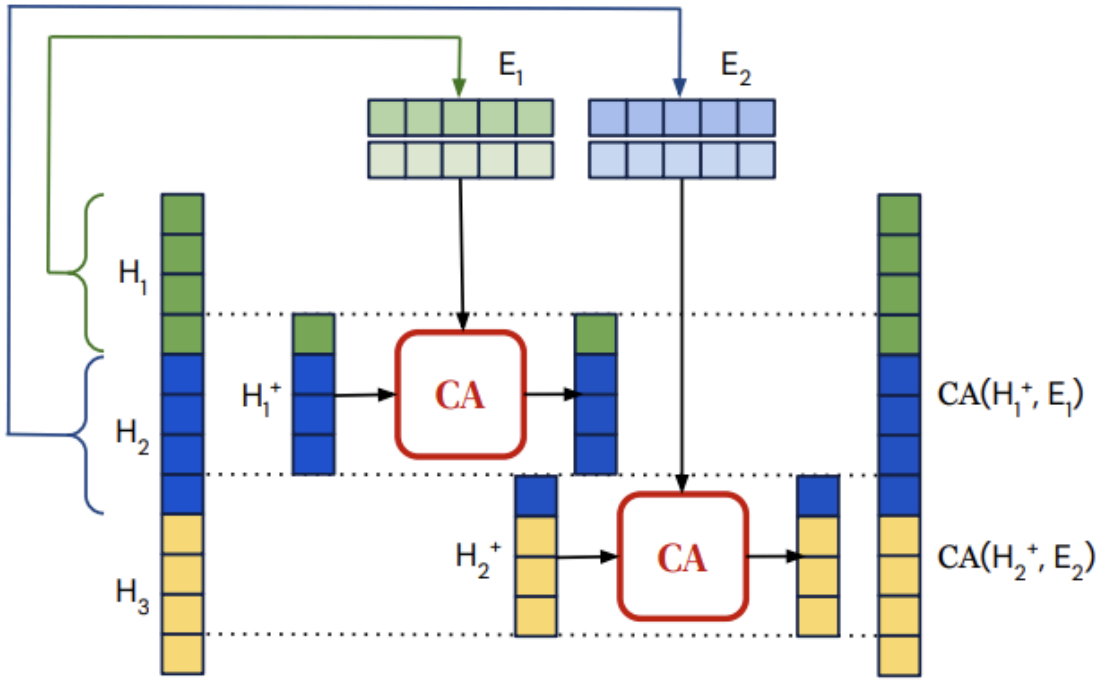
Outputs from the previous layer H

Chunked Cross Attention



Outputs from the previous layer H

Chunked Cross Attention



Outputs from the previous layer H

Inputs to the next layer

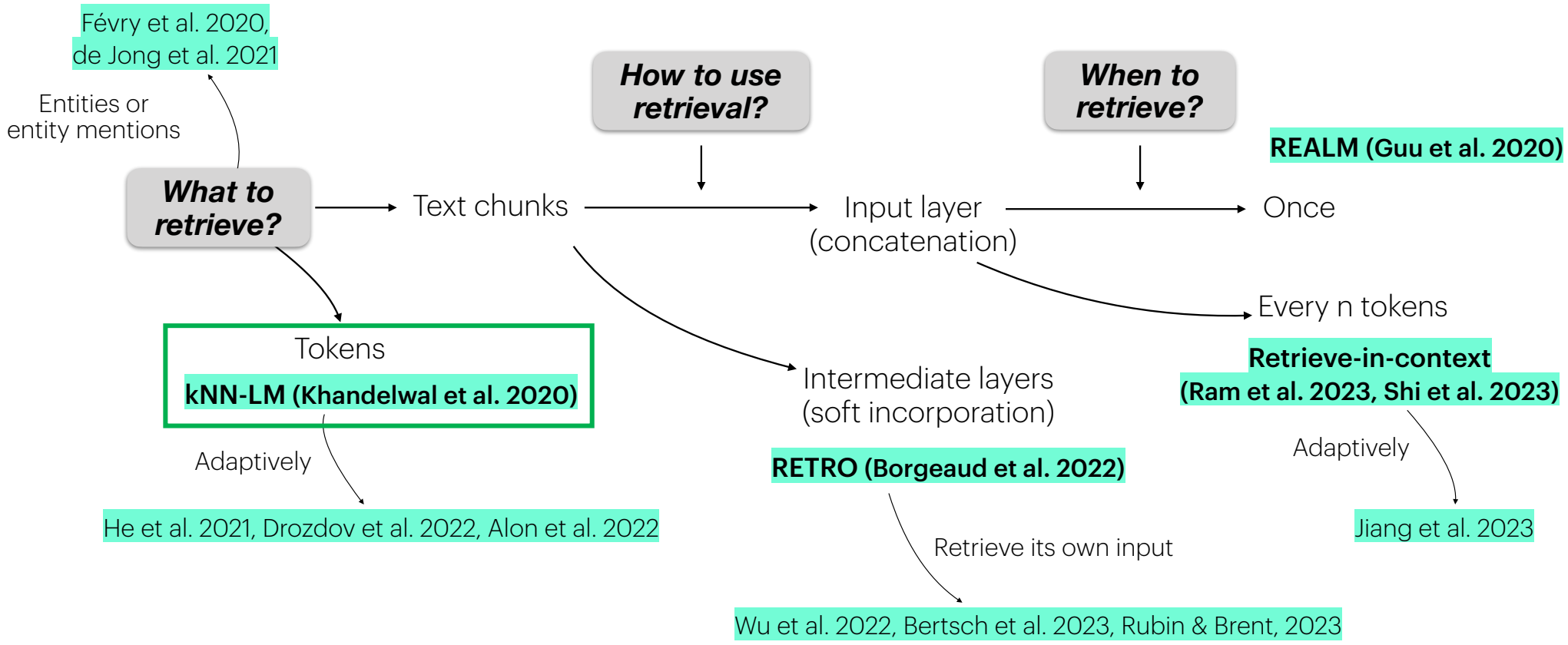
Results

Perplexity: The lower the better

Model	Retrieval Set	#Database tokens	#Database keys	Valid	Test
Adaptive Inputs (Baevski and Auli, 2019)	-	-	-	17.96	18.65
SPALM (Yogatama et al., 2021)	Wikipedia	3B	3B	17.20	17.60
*k*NN-LM (Khandelwal et al., 2020)	Wikipedia	3B	3B	16.06	16.12
Megatron (Shoeybi et al., 2019)	-	-	-	-	10.81
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**

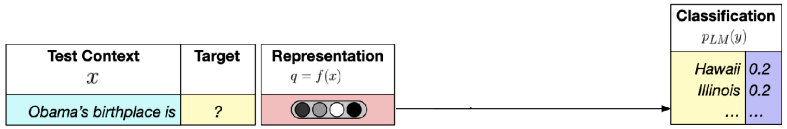
Significant improvements by retrieving from 1.8 trillion tokens

Architectures (roadmap)

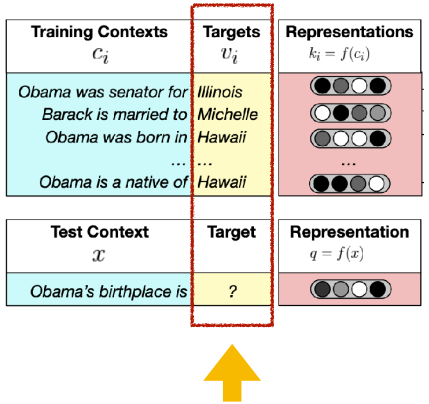


kNN-LM: Generating by retrieving (or "memorizing"?)

Language Model

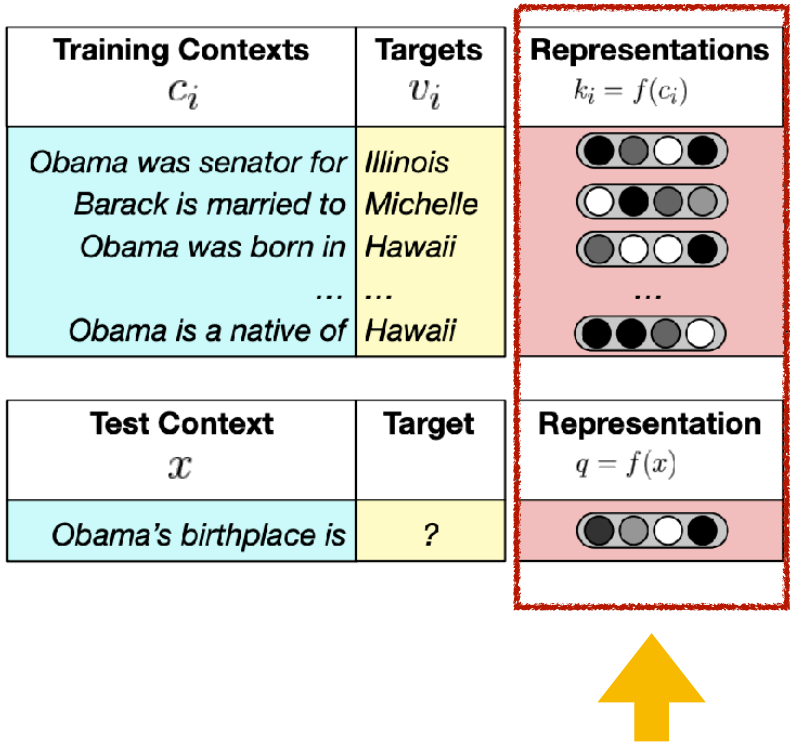


kNN-LM (Khandelwal et al. 2020)



Which tokens in a datastore are close to the next token?

kNN-LM (Khandelwal et al. 2020)



Which tokens in a datastore are close to the next token?

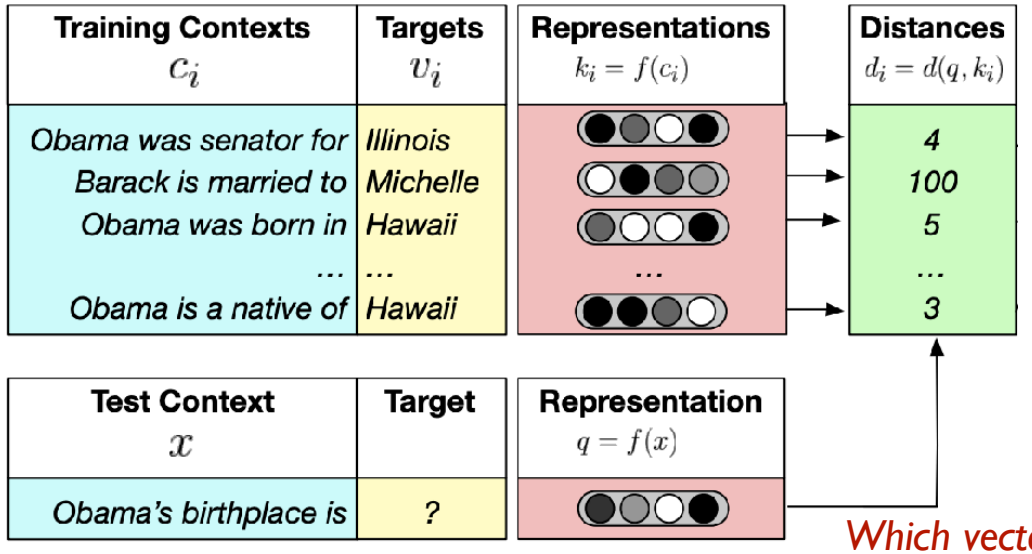
=

Which prefixes in a datastore are close to the prefix we have?

=

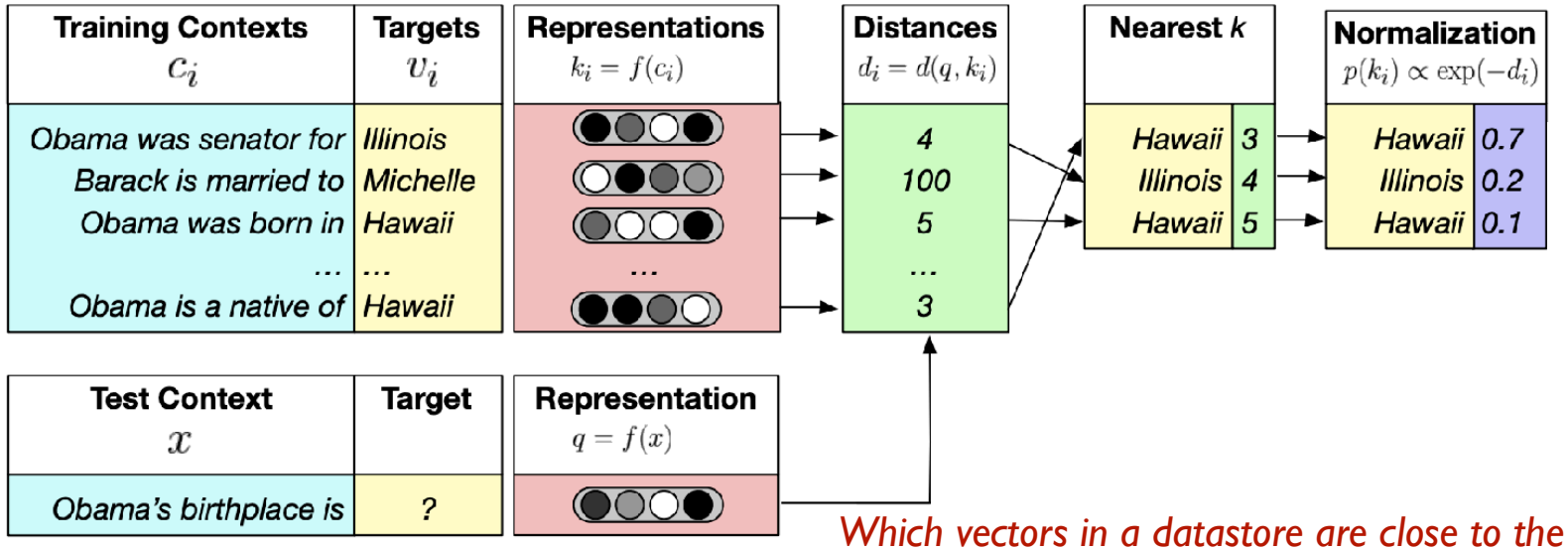
Which vectors in a datastore are close to the vector we have?

kNN-LM (Khandelwal et al. 2020)



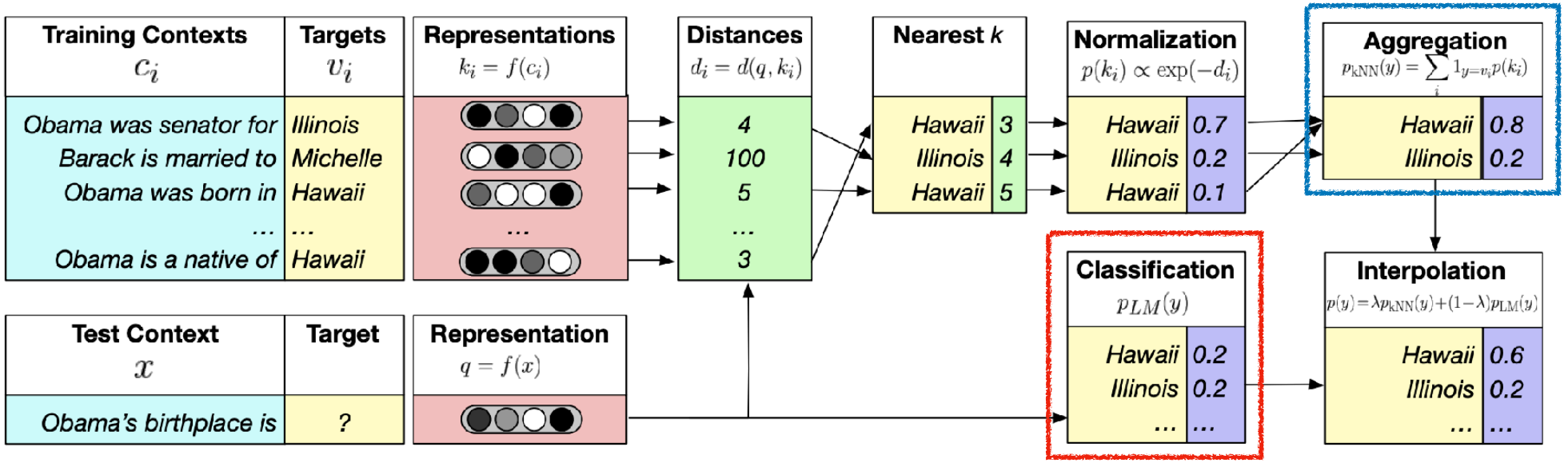
Which vectors in a datastore are close to the vector we have?

kNN-LM (Khandelwal et al. 2020)



Which vectors in a datastore are close to the vector we have?

kNN-LM (Khandelwal et al. 2020)

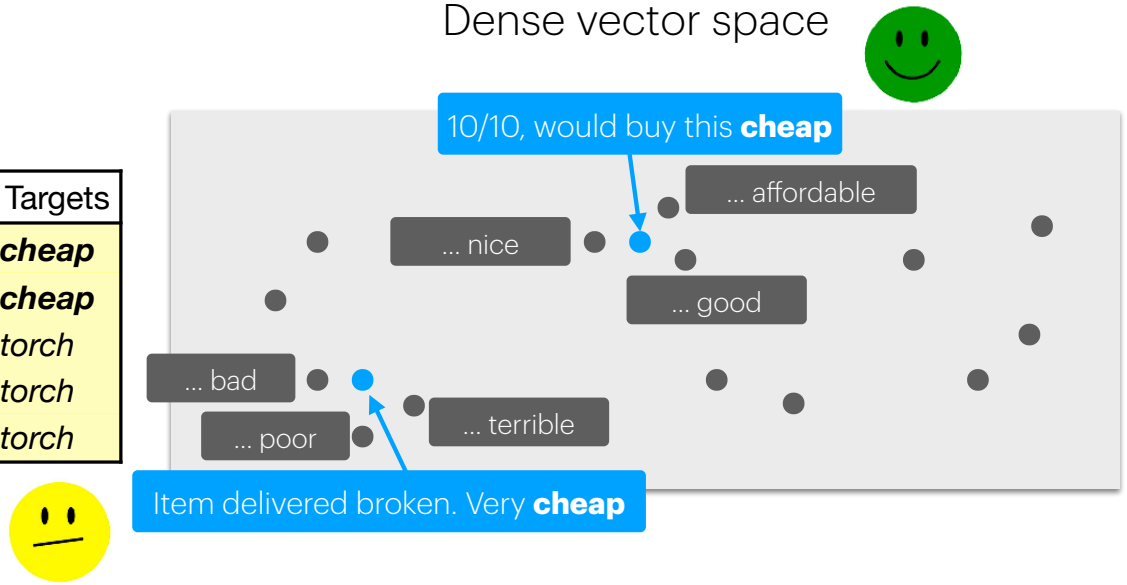


λ: hyperparameter

$$P_{k\text{NN-LM}}(y \mid x) = (1 - \lambda) P_{\text{LM}}(y \mid x) + \lambda P_{k\text{NN}}(y \mid x)$$

kNN-LM - why?

Training contexts	Targets
10/10, would buy this	***cheap***
Item delivered broken. Very	***cheap***
To check the version of PyTorch, you can use	*torch*
You are permitted to bring a	*torch*
A group of infections … one of the	*torch*



Summary

	What do retrieve?	How to use retrieval?	When to retrieve?
REALM (Guu et al 2020)	Text chunks	Input layer	Once
Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)	Text chunks	Input layer	Every n tokens
RETRO (Borgeaud et al. 2021)	Text chunks	Intermediate layers	Every n tokens
kNN-LM (Khandelwal et al. 2020)	Tokens	Output layer	Every token

More fine-grained; Can be better at rare patterns & out-of-domain

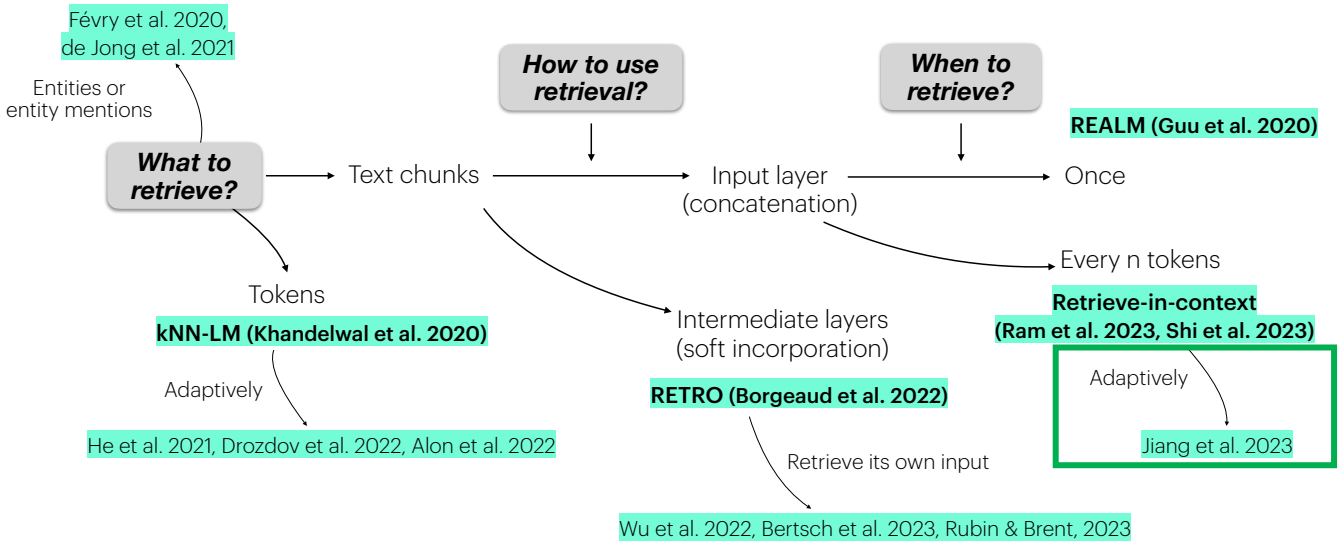
Can be very efficient (as long as kNN search is fast)

(Wikipedia) 13M vs. 4B

Datastore is expensive in space: given the same data, # text chunks vs. # tokens

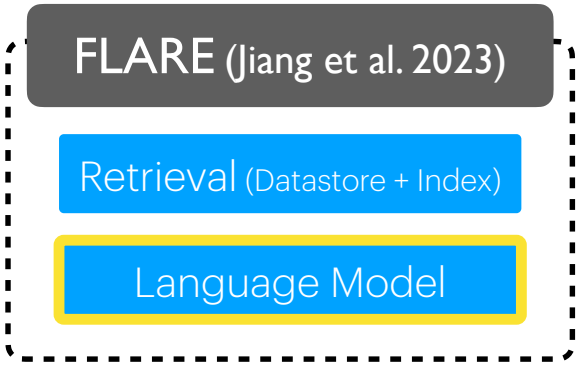
No cross attention between input and retrieval results

Can we retrieve only when needed?



Adaptive retrieval of *chunks*

- Judge necessity



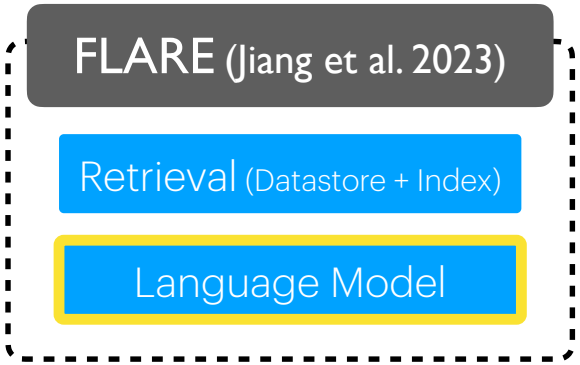
Input: Generate a summary about Joe Biden.

Joe Biden (born November 20, 1942) is the 46th president of the United States.

Jiang et al. “Active Retrieval Augmented Generation”

Adaptive retrieval of *chunks*

- Judge necessity



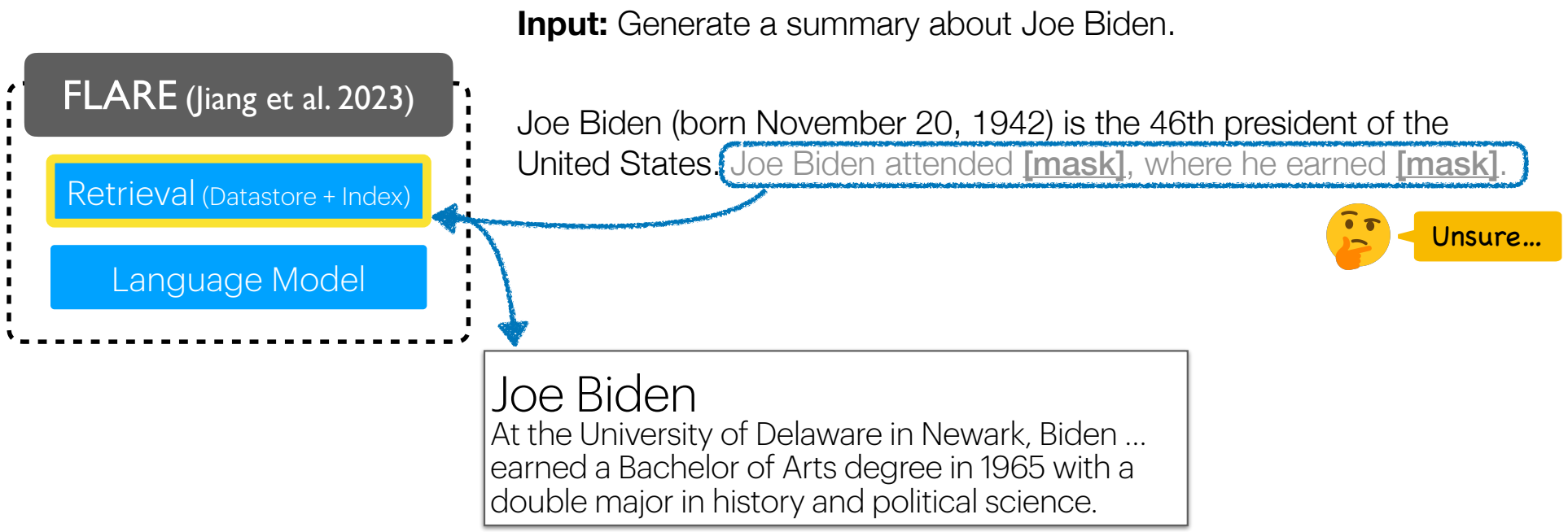
Input: Generate a summary about Joe Biden.

Joe Biden (born November 20, 1942) is the 46th president of the United States. Joe Biden attended **the University of Pennsylvania**, where he earned **a law degree**.

Jiang et al. “Active Retrieval Augmented Generation”

Adaptive retrieval of *chunks*

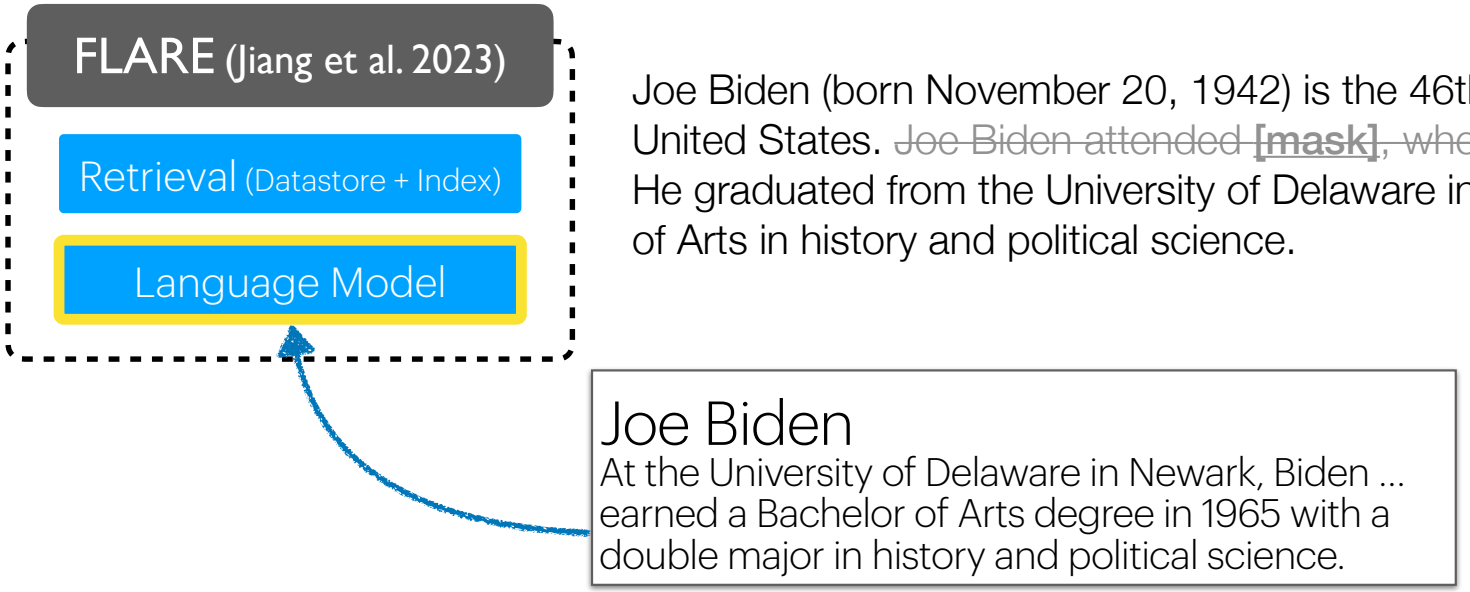
- Judge necessity



Jiang et al. “Active Retrieval Augmented Generation”

Adaptive retrieval of *chunks*

- Judge necessity



Input: Generate a summary about Joe Biden.

Joe Biden (born November 20, 1942) is the 46th president of the United States. ~~Joe Biden attended **[mask]**, where he earned **[mask]**.~~ He graduated from the University of Delaware in 1965 with a Bachelor of Arts in history and political science.

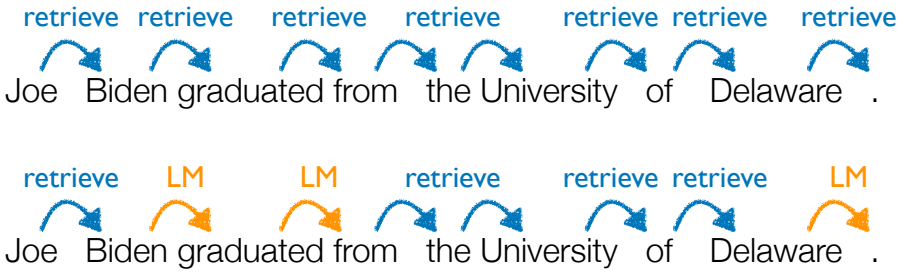
Jiang et al. “Active Retrieval Augmented Generation”

Also with kNN-LM!

[kNN-LM] does it in a per-token basis instead of sentence/chunk level

Adaptive retrieval of *tokens*

- Judge necessity

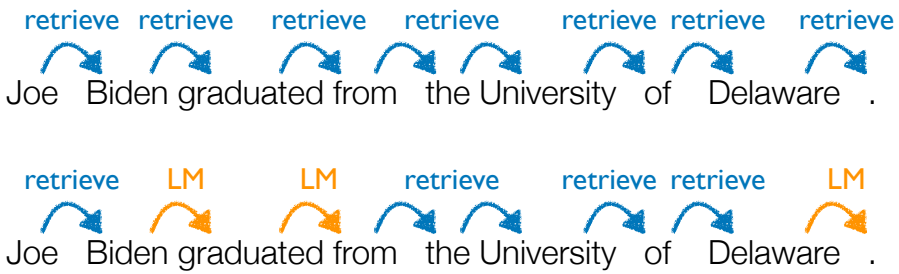


He et al. 2021. “Efficient Nearest Neighbor Language Models”

Also with kNN-LM!

Adaptive retrieval of *tokens*

- Judge necessity



$$P_{k\text{NN-LM}}(y \mid x) = (1 - \lambda(x))P_{\text{LM}}(y \mid x) + \lambda(x)P_{k\text{NN}}(y \mid x)$$

A function of the input ***x***

$\rightarrow \ \lambda = 0 \quad if \ \lambda < \gamma$

He et al. 2021. "Efficient Nearest Neighbor Language Models"

predict the interpolation parameter λ(x):

trade-off between memorization (knn-LM) and LM predictions

Summary

	What do retrieve?	How to use retrieval?	When to retrieve?
REALM (Guu et al 2020)	Text chunks	Input layer	Once
Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)	Text chunks	Input layer	Every n tokens
RETRO (Borgeaud et al. 2021)	Text chunks	Intermediate layers	Every n tokens
kNN-LM (Khandelwal et al. 2020)	Tokens	Output layer	Every token
FLARE (Jiang et al. 2023)	Text chunks	Input layer	Every n tokens ***(adaptive)***
Adaptive kNN-LM (He et al 2021, Alon et al 2022, etc)	Tokens	Output layer	Every n tokens ***(adaptive)***

Questions?