

ACL 2023 Tutorial:

Retrieval-based Language Models and Applications

Akari Asai, Sewon Min, Zexuan Zhong, Danqi Chen

CNeRG Reading Group Presentation
(14th November, 2024)

Sayantan Adak
Kiran Purohit

About presenters



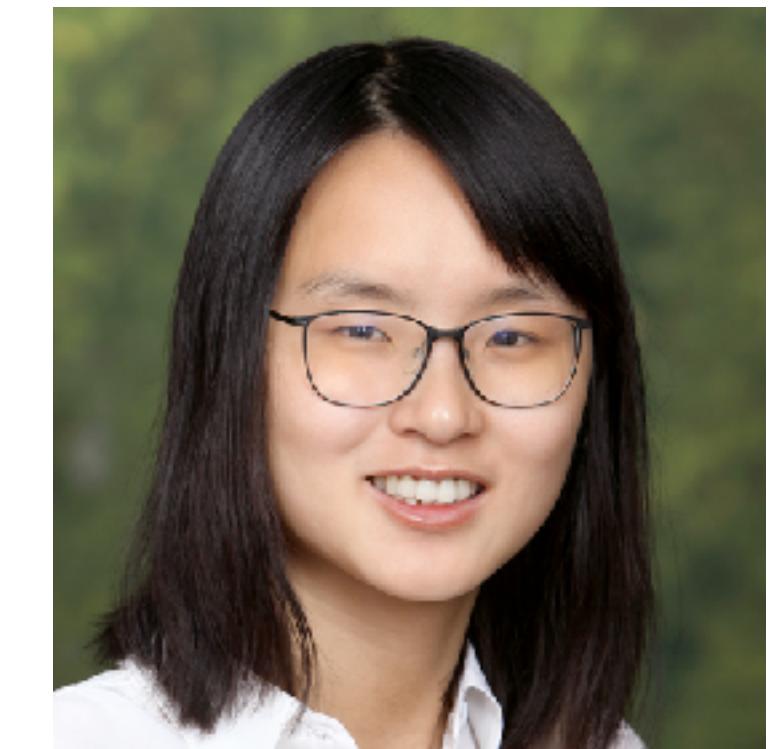
Akari Asai
PhD student
@UW



Sewon Min
PhD student
@UW



Zexuan Zhong
PhD student
@Princeton



Danqi Chen
Faculty
@Princeton

Participation and Q & A

- All tutorial slides and reading list are available at:

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

- Throughout the tutorial, we will provide Q & A on sli.do (also on the website):

<https://tinyurl.com/retrieval-lm-tutorial>



The screenshot shows the sli.do interface for the "ACL 2023 Tutorial: Retrieval-based Methods". The top navigation bar includes a menu icon, the title, and a "Q&A" button. Below the title, event details are listed: "ACL 2023 Tutorial: Retrieval-based Methods", "Jul 6–10, 2023", and "#1430 562". On the left, there are three buttons: "Live interaction", "Switch event", and "Dark mode" with a toggle switch. On the right, there is a large input field with a user icon and the placeholder text "Type your question".

Ask & upvote questions anytime!
(+10 minutes Q & A in 1st and 2nd half)

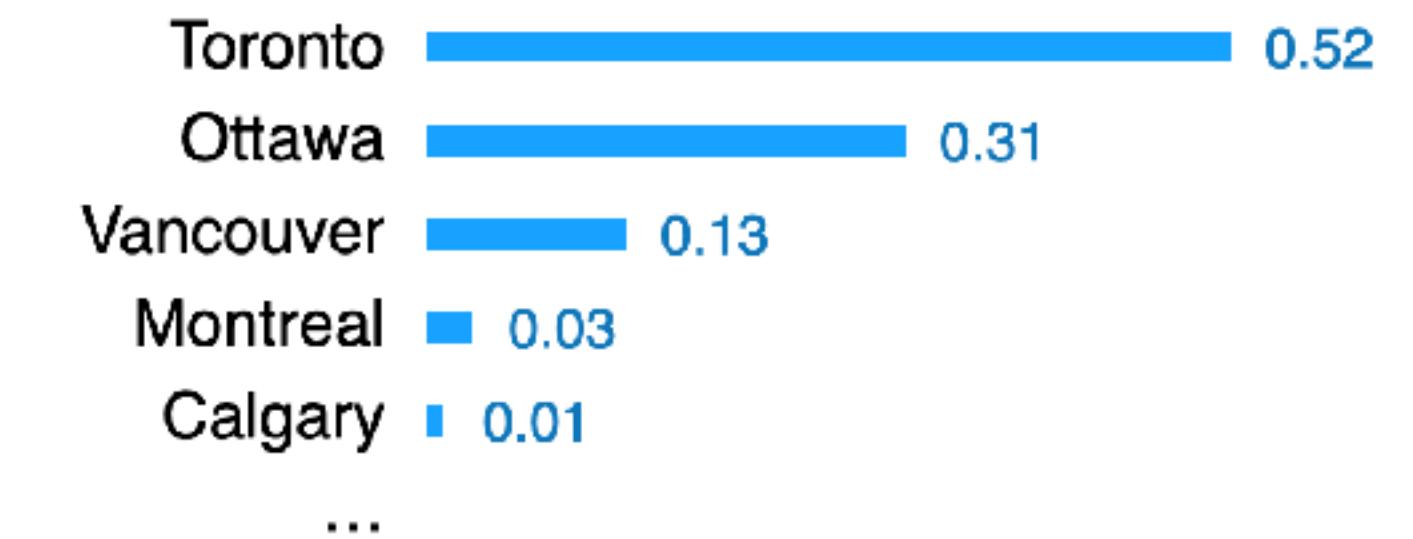
Retrieval-based language models (LMs)

Retrieval-based LMs = Retrieval + LMs

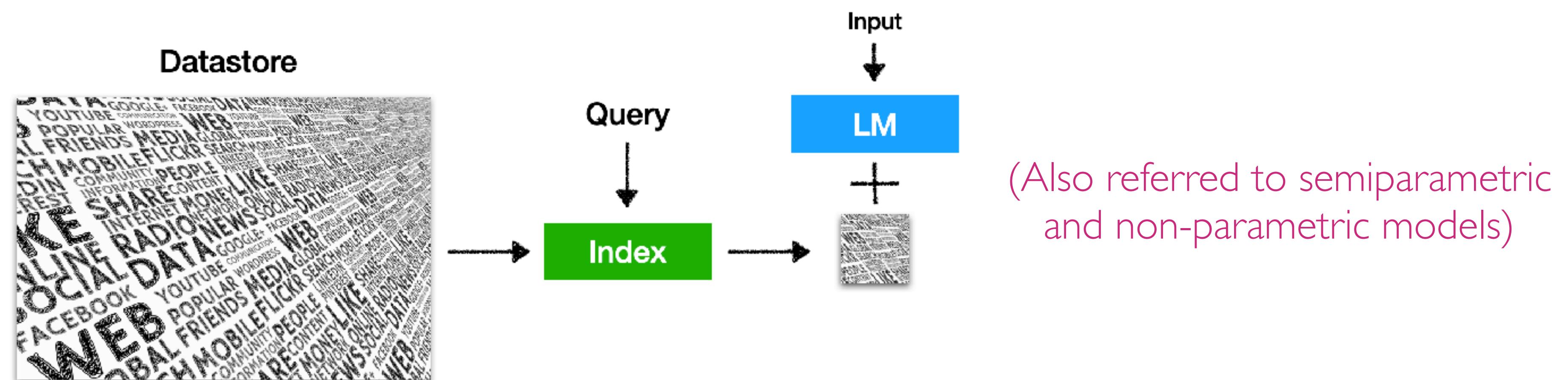
- It is a **language model** $P(x_n | x_1, x_2, \dots, x_{n-1})$

The capital city of Ontario is _____

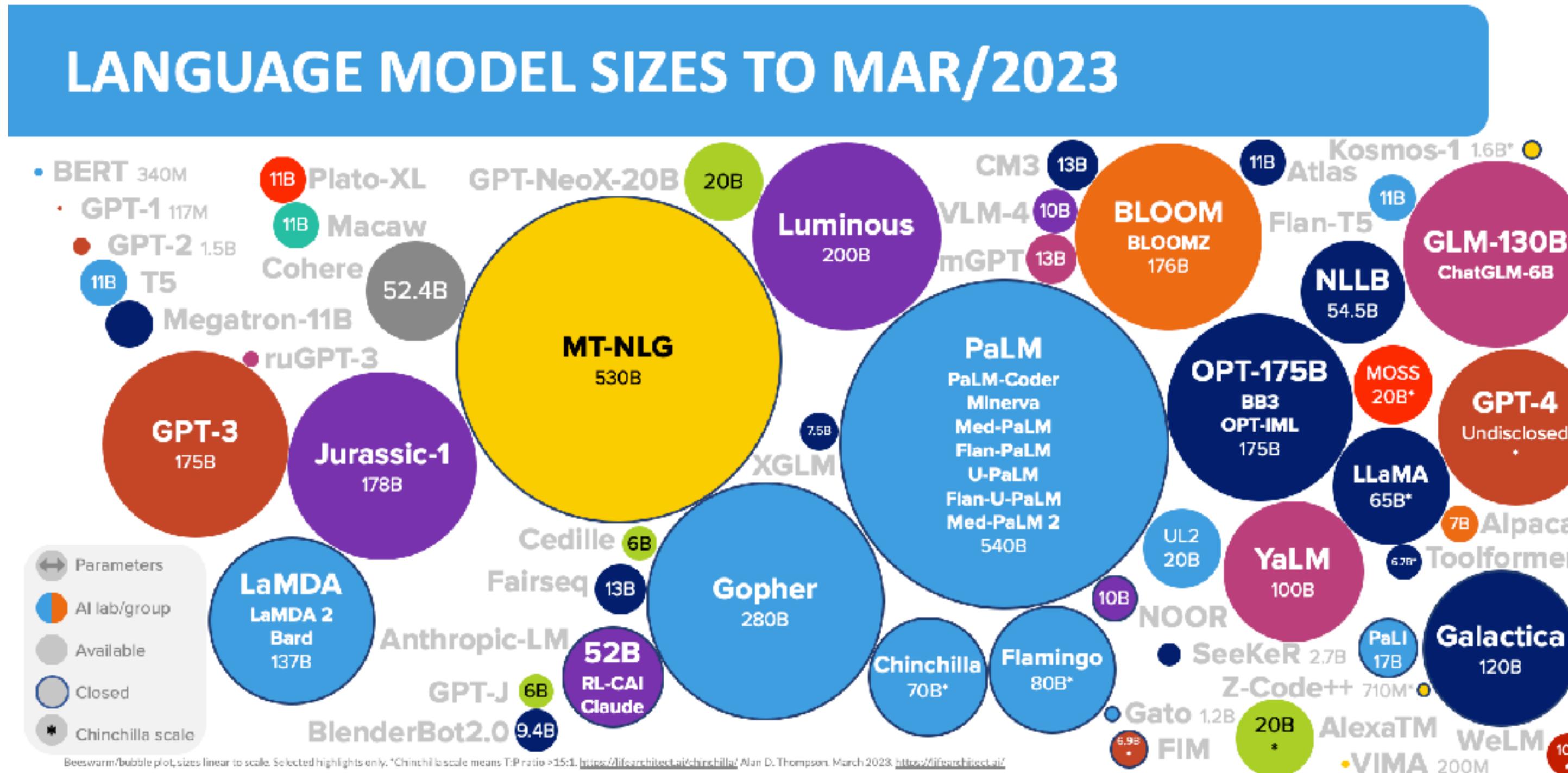
(can be broadly extended to masked language models or encoder-decoder models)



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



The age of large language models (LLMs)



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

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

Retrieval for knowledge-intensive NLP tasks

Representative tasks: open-domain QA, fact checking, entity linking, ..

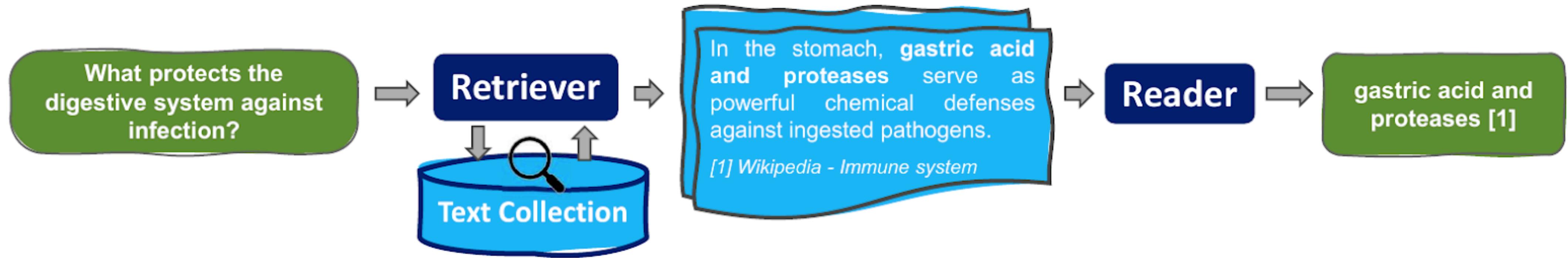


Image: <http://ai.stanford.edu/blog/retrieval-based-NLP/>

Drives a lot of research on better algorithms for **dense retrieval**, e.g., **DPR** (Karpukhin et al., 2020), **ColBERT** (Khattab and Zaharia, 2020), **ANCE** (Xiong et al., 2021), **Contriever** (Izacard et al., 2022), ...

Why retrieval → LMs?

Why retrieval-based LMs?

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



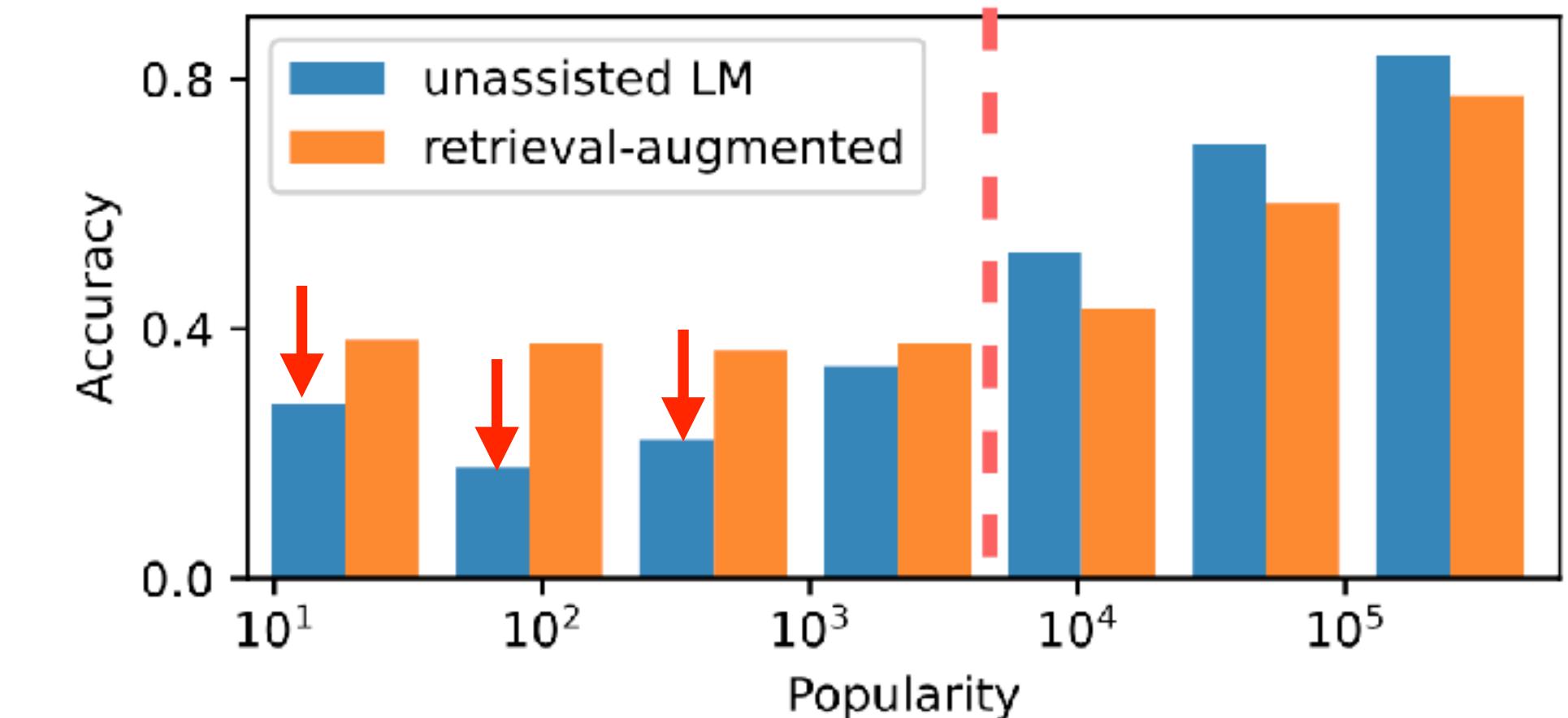
List 5 important papers authored by Geoffrey Hinton



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**, ...
3. "...
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?

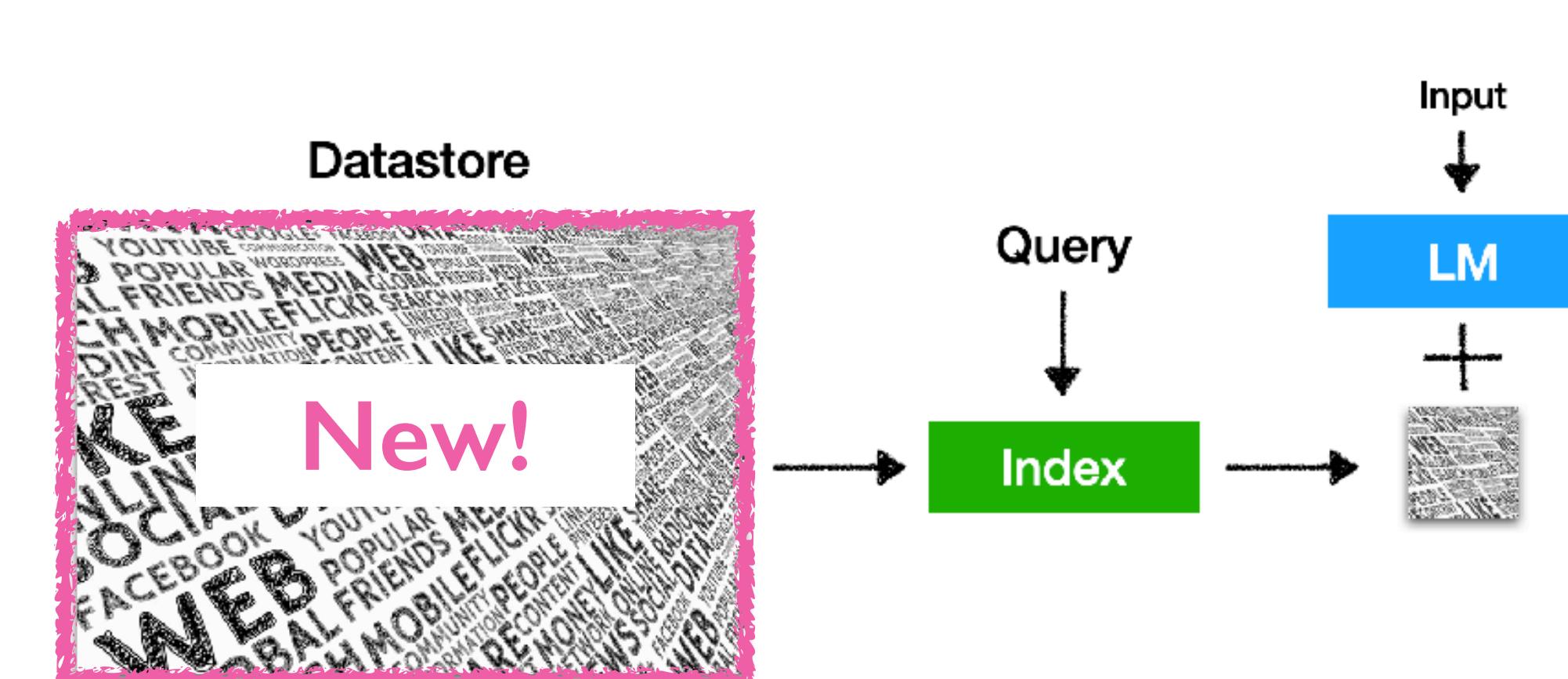


As of my **knowledge cutoff in September 2021**, the CEO of Twitter is **Jack Dorsey**....

The screenshot shows a Google search results page. The search bar contains the query "Who is the CEO of Twitter?". Below the search bar are navigation links for "All", "News", "Images", "Shopping", "Videos", and "More". A status message indicates "About 1,090,000,000 results (0.45 seconds)". The top result is a snippet for "Twitter / CEO" followed by the name "Linda Yaccarino" and a small profile picture of her. The date "Jun 5, 2023" is also visible.

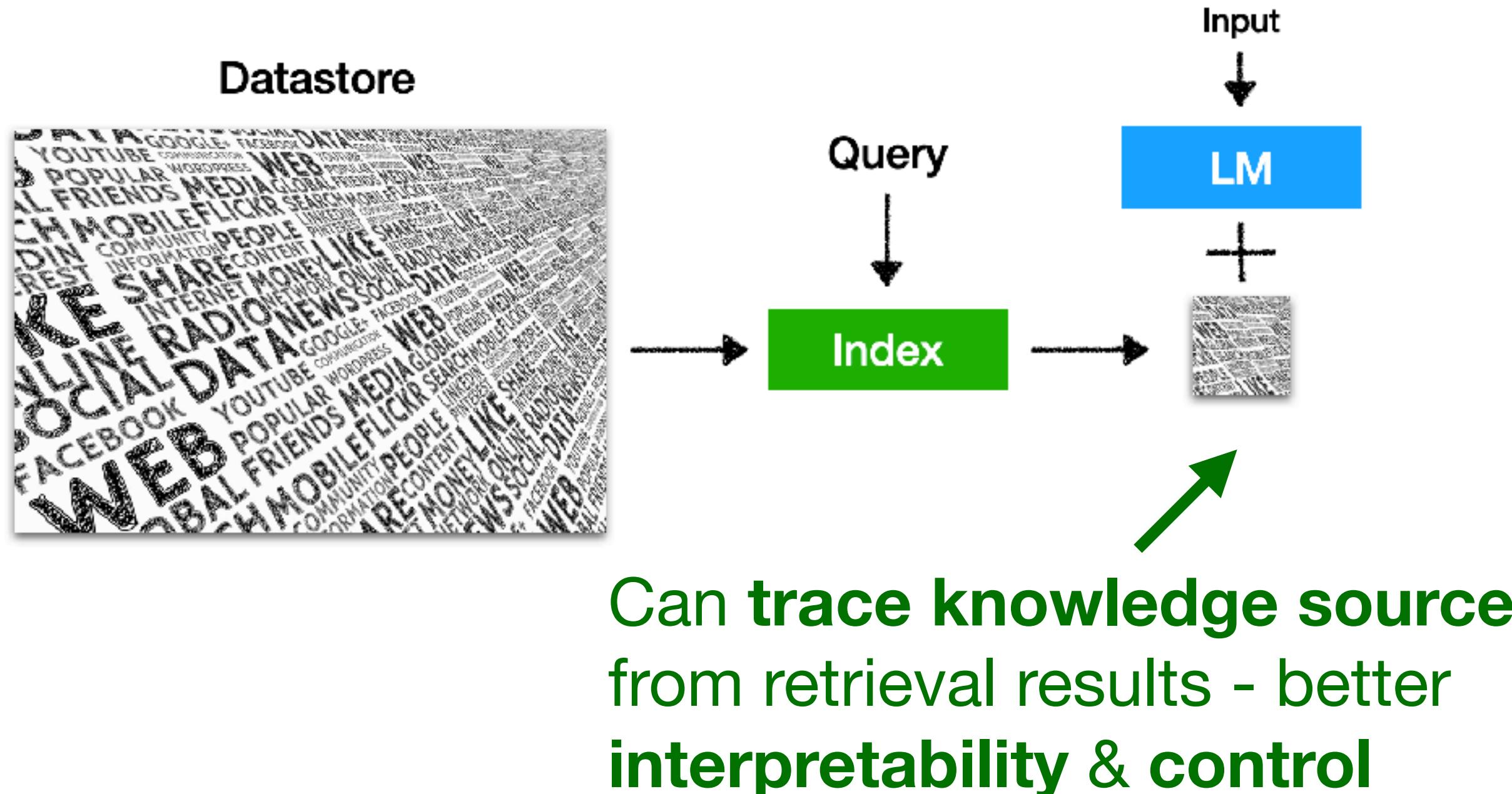


- Existing **knowledge editing** methods are still NOT scalable (**active research!**)
- The datastore can be easily **updated** and **expanded** - even without retraining!

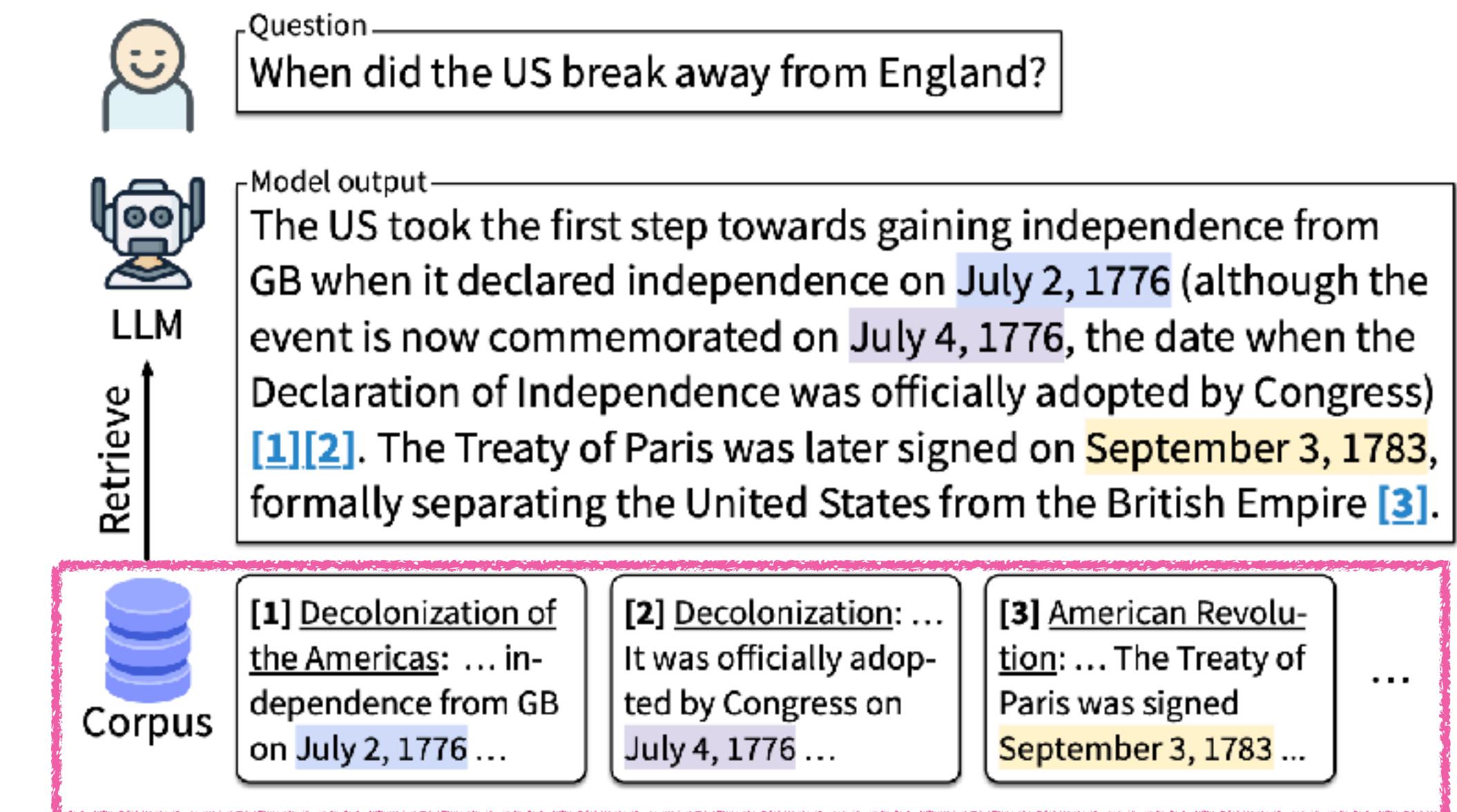


Why retrieval-based LMs?

LLMs' output is challenging to interpret and verify



Generating text with citations



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

Why retrieval-based LMs?

LLMs' output is challenging to interpret and verify

 Create an itinerary for exploring Toronto over a span of two days.  



Like  Dislike  ...

Toronto is a vibrant and diverse city with many attractions and activities to suit different tastes and preferences. Here are some suggestions for creating an itinerary for exploring Toronto over two days:

- **Day 1:** Start your day with a visit to the **CN Tower**, the iconic landmark that offers panoramic views of the city and beyond ¹. You can also experience the thrill of walking on the glass floor or the EdgeWalk, the world's highest hands-free walk ¹. Next, head to the nearby **St. Lawrence Market**, one of the world's best food markets, where you can sample a variety of cuisines and local specialties ². After lunch, take a stroll along **Queen West**, a trendy neighborhood with eclectic shops, galleries, cafes and street art ³. In the evening, enjoy a show at one of the many theaters or comedy clubs in the **Entertainment District**, or catch a game at the **Scotiabank Arena** if you're a sports fan.
- **Day 2:** Explore the history and culture of Toronto at the **Royal Ontario Museum**, one of the largest and most comprehensive museums in North America, featuring exhibits on art, natural history, world cultures and more ⁴. Then, hop on a ferry to the **Toronto Islands**, a group of islands that offer a relaxing escape from the city, with beaches, parks, trails and amusement rides ³ ⁵. You can also rent a bike or kayak to explore the islands at your own pace. For dinner, head to **Chinatown**, one of the largest and most vibrant in North America, where you can find a variety of Asian cuisines and shops ³.

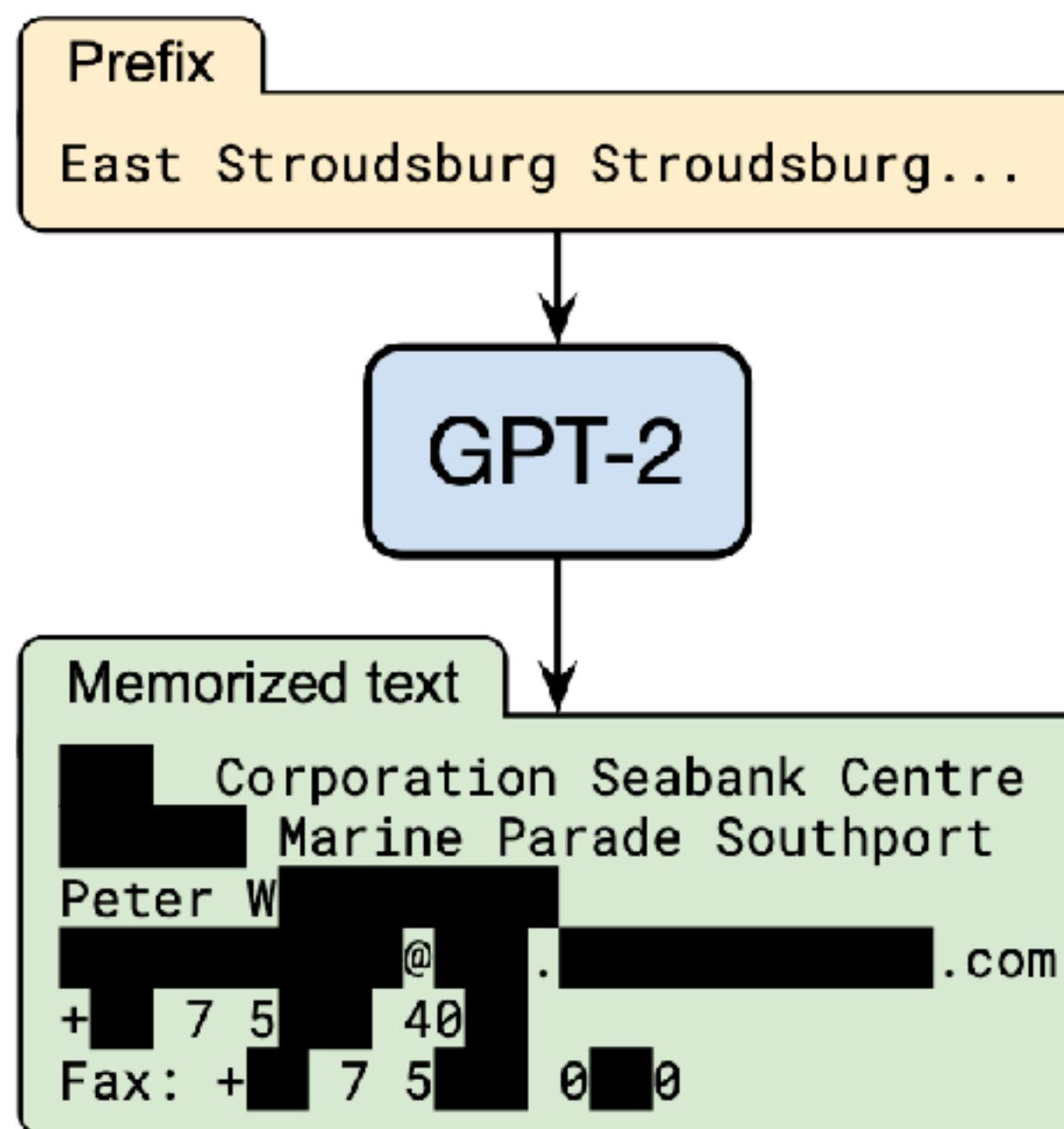
I hope this helps you plan your trip to Toronto. Have fun! 😊

Learn more:

1. cntower.ca
2. travel.usnews.com
3. bing.com
4. rom.on.ca
5. tripadvisor.com

Why retrieval-based LMs?

LLMs are shown to easily leak private training data

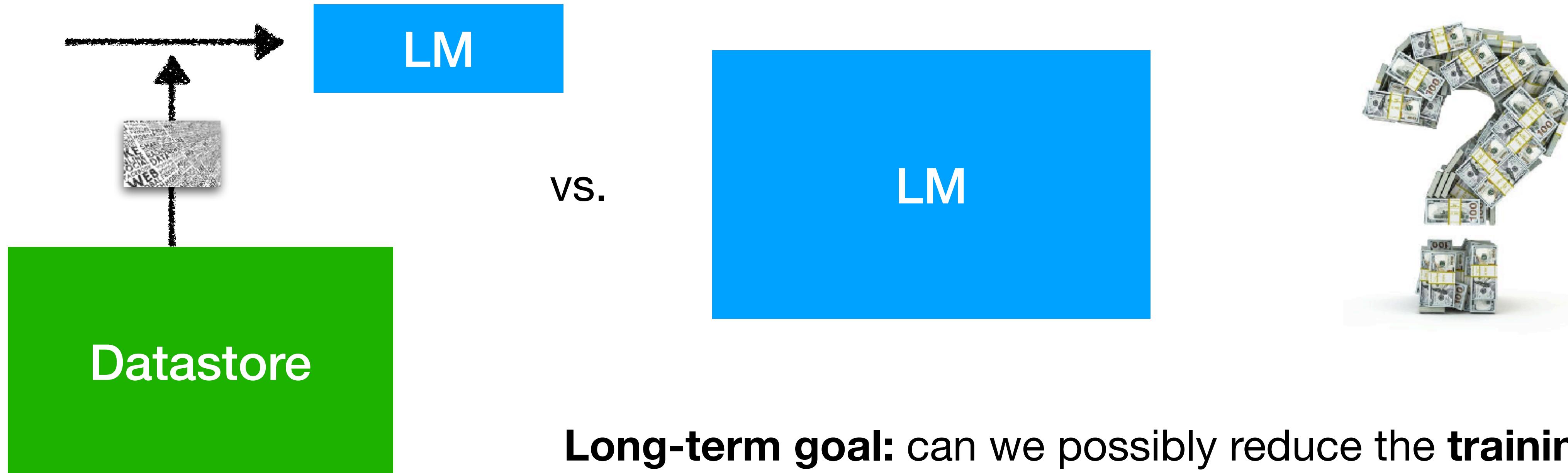


Category	Count
US and international news	109
Log files and error reports	79
License, terms of use, copyright notices	54
Lists of named items (games, countries, etc.)	54
Forum or Wiki entry	53
Valid URLs	50
Named individuals (non-news samples only)	46
Promotional content (products, subscriptions, etc.)	45
High entropy (UUIDs, base64 data)	35
Contact info (address, email, phone, twitter, etc.)	32
Code	31
Configuration files	30
Religious texts	25
Pseudonyms	15
Donald Trump tweets and quotes	12
Web forms (menu items, instructions, etc.)	11
Tech news	11
Lists of numbers (dates, sequences, etc.)	10

Individualization on private data by storing it in the datastore

Why retrieval-based LMs?

LLMs are ***large*** and expensive to train and run



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**”

Goals of the tutorial

- We will cover a number of key developments on retrieval-based LMs (mostly 2020–2023)
 - **Problem definition**
 - **Architecture design**
 - **Training methods**
 - **Applications and extensions**
- This tutorial is **cutting-edge**, and we are still far from understanding how to best develop retrieval-based LMs compared to parametric LLMs:
 - Taxonomies of existing research and key insights
 - Our perspectives on the current challenges & open problems



Schedule

Time	Section	Presenter
14:00—14:15	Section 1: Introduction [Slides]	Danqi
14:15—14:25	Section 2: Definition & Preliminaries [Slides]	Sewon
14:25—15:00	Section 3: Retrieval-based LMs: Architecture [Slides]	Sewon
15:00—15:25	Section 4: Retrieval-based LMs: Training [Slides]	Zexuan
15:25—15:30	Q & A Session I	
 30min coffee break		
16:00—16:25	Section 4 (Cont'd): Retrieval-based LMs: Training [Slides]	Zexuan
16:25—17:00	Section 5: Retrieval-based LMs: Applications [Slides]	Akari
17:00—17:10	Section 6: Extension: Multilingual & Multimodal [Slides]	Akari
17:10—17:20	Section 7: Challenges & Opportunities [Slides]	Danqi
17:20—17:30	Q & A Session II	

Section 2: Definition & Preliminaries

A Retrieval-based LM: Definition

A language model (LM) that uses
an external datastore at test time

A Retrieval-based LM: Definition

A language model (LM) that uses
an external datastore at test time

A Retrieval-based LM: Definition

A language model (LM) that uses
an external datastore at test time

A Retrieval-based LM: Definition

**A language model (LM) that uses
an external datastore at test time**

A language model (LM)

$$P(x_n | x_1, x_2, \dots, x_{n-1})$$

A language model (LM)

$$P(x_n | x_1, x_2, \dots, x_{n-1})$$

Language model (Transformers)

The capital city of Ontario is

x_1

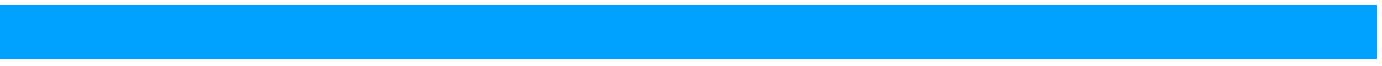
x_2

...

x_{n-1}

A language model (LM)

$$P(x_n | x_1, x_2, \dots, x_{n-1})$$

Toronto  0.52

Ottawa  0.31

Vancouver  0.13

Montreal  0.03

Calgary  0.01

...

Language model (Transformers)

The capital city of Ontario is

x_1

x_2

...

x_{n-1}

A language model (LM): Categories

Toronto

Autoregressive LM

The capital city of Ontario is _____

A language model (LM): Categories

Toronto

Autoregressive LM

The capital city of Ontario is _____

vs

capital

Ontario

Masked LM

The _____ city of _____ is Toronto

A language model (LM): Categories

Toronto

Autoregressive LM

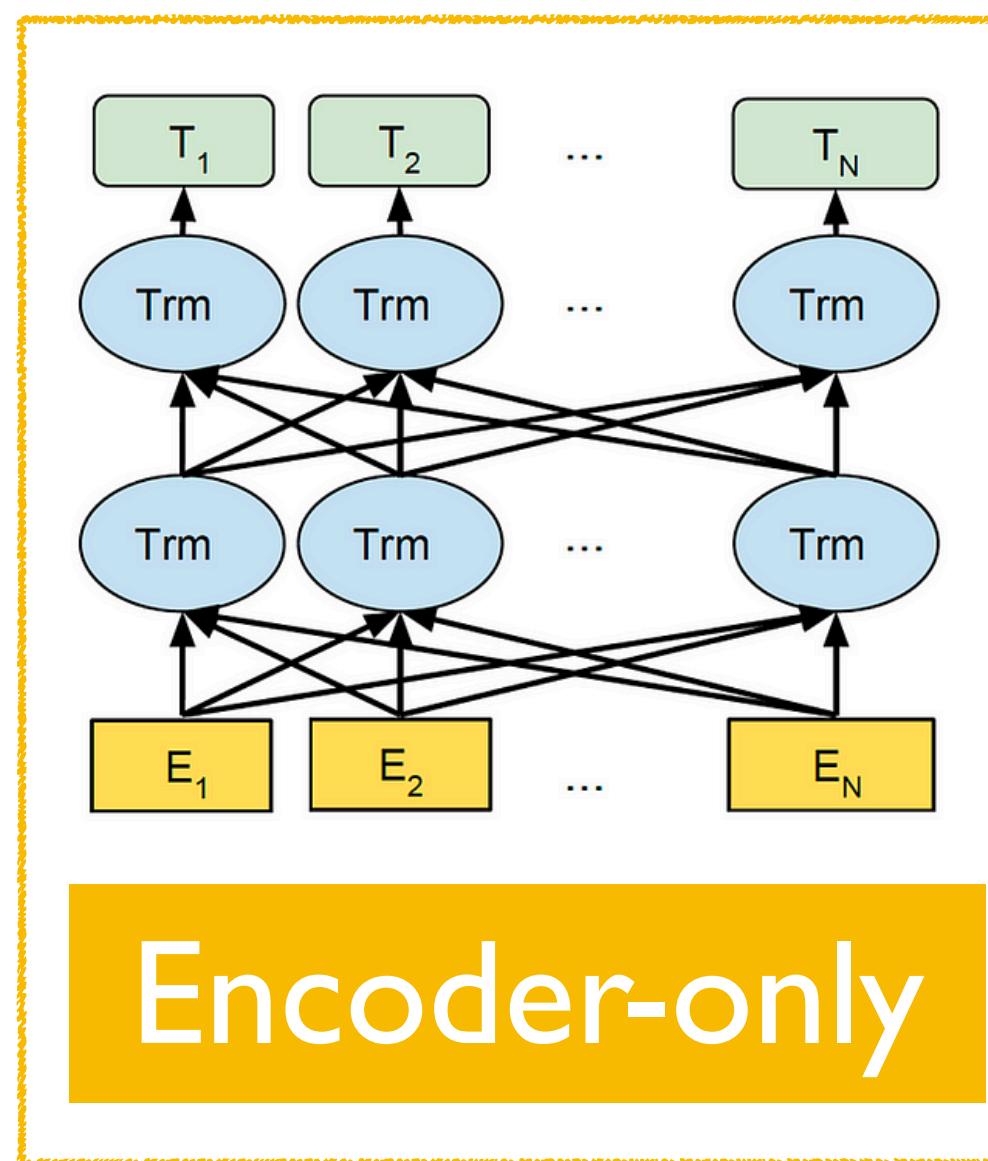
The capital city of Ontario is _____

capital

Ontario

Masked LM

The _____ city of _____ is Toronto



A language model (LM): Categories

Toronto

Autoregressive LM

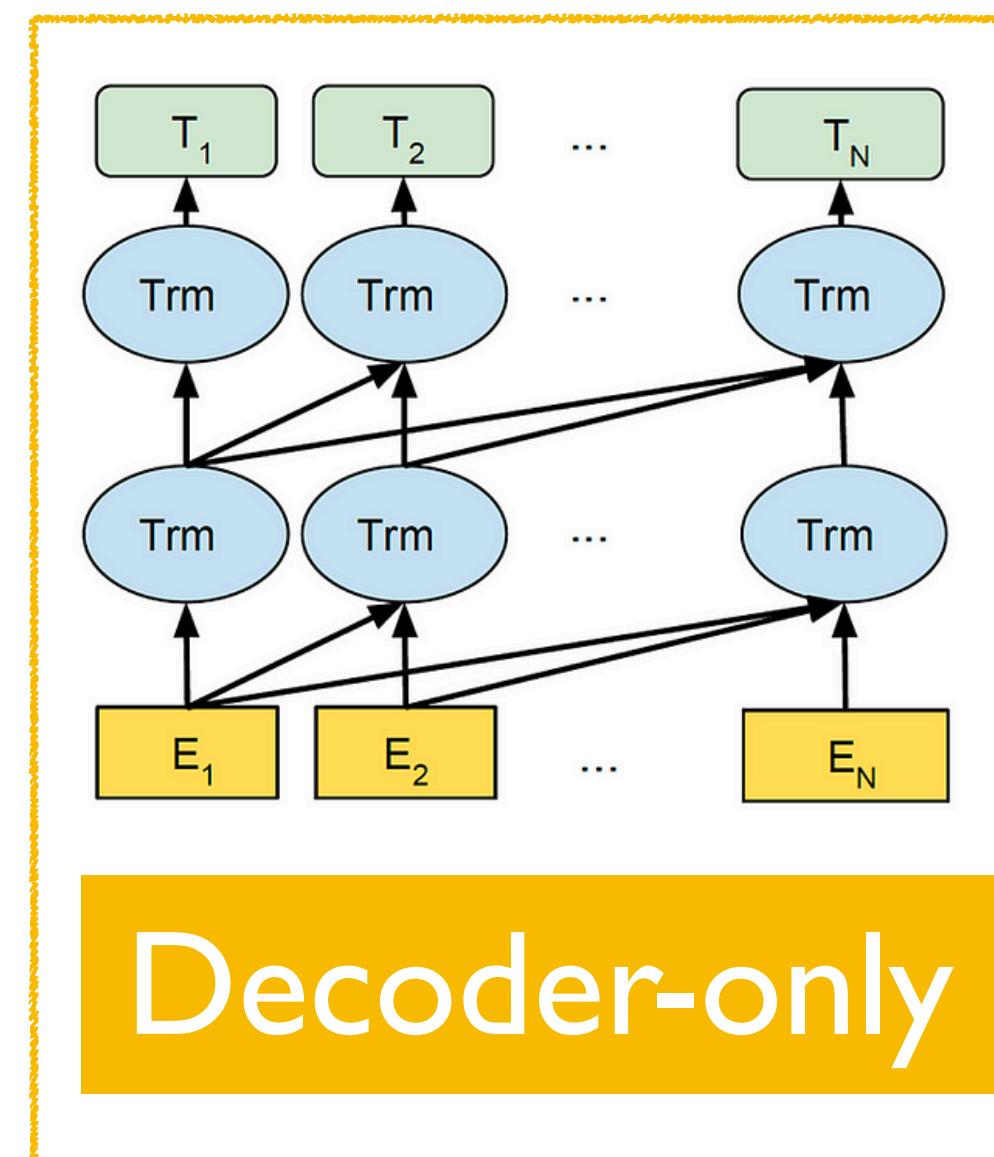
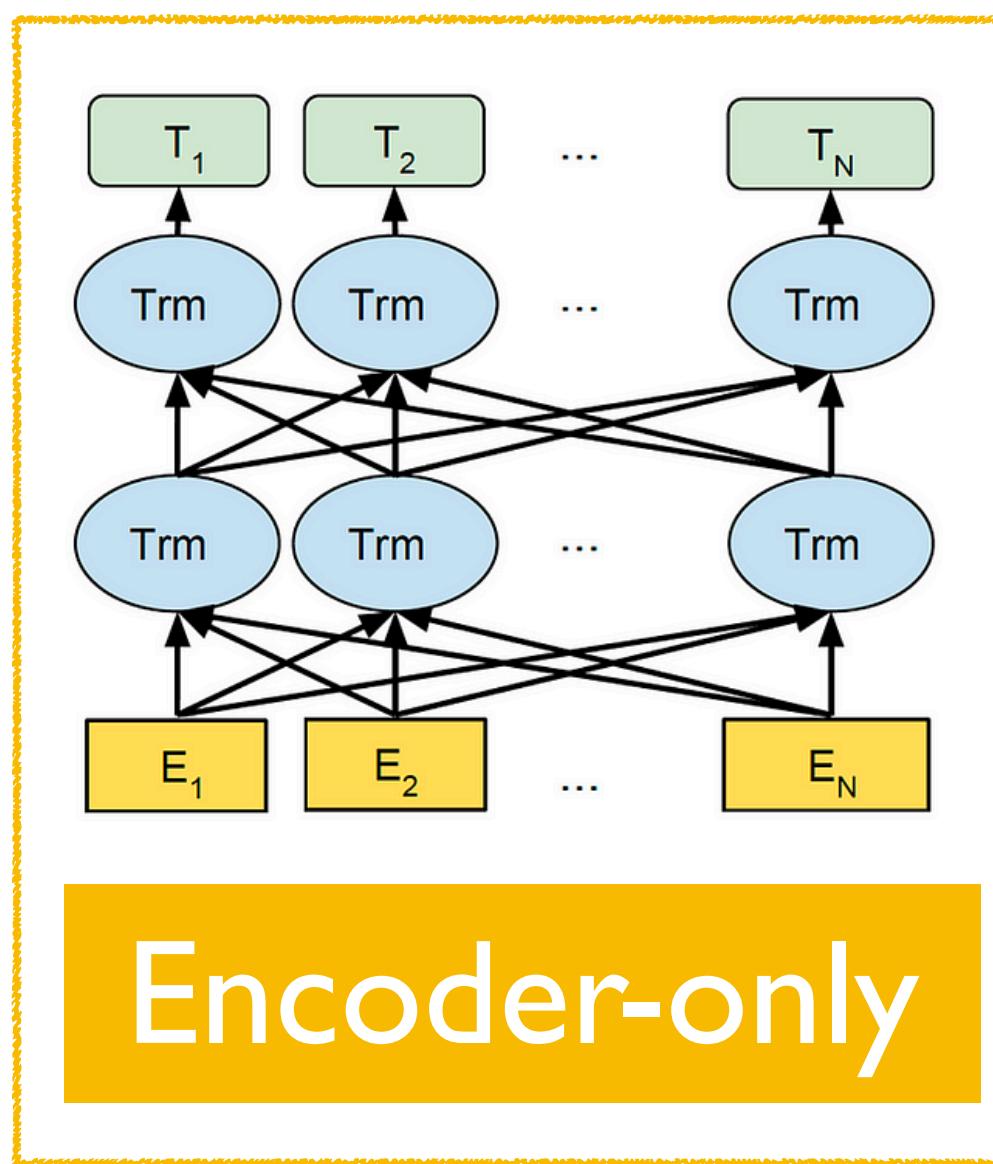
The capital city of Ontario is _____

capital

Ontario

Masked LM

The _____ city of _____ is Toronto



A language model (LM): Categories

Toronto

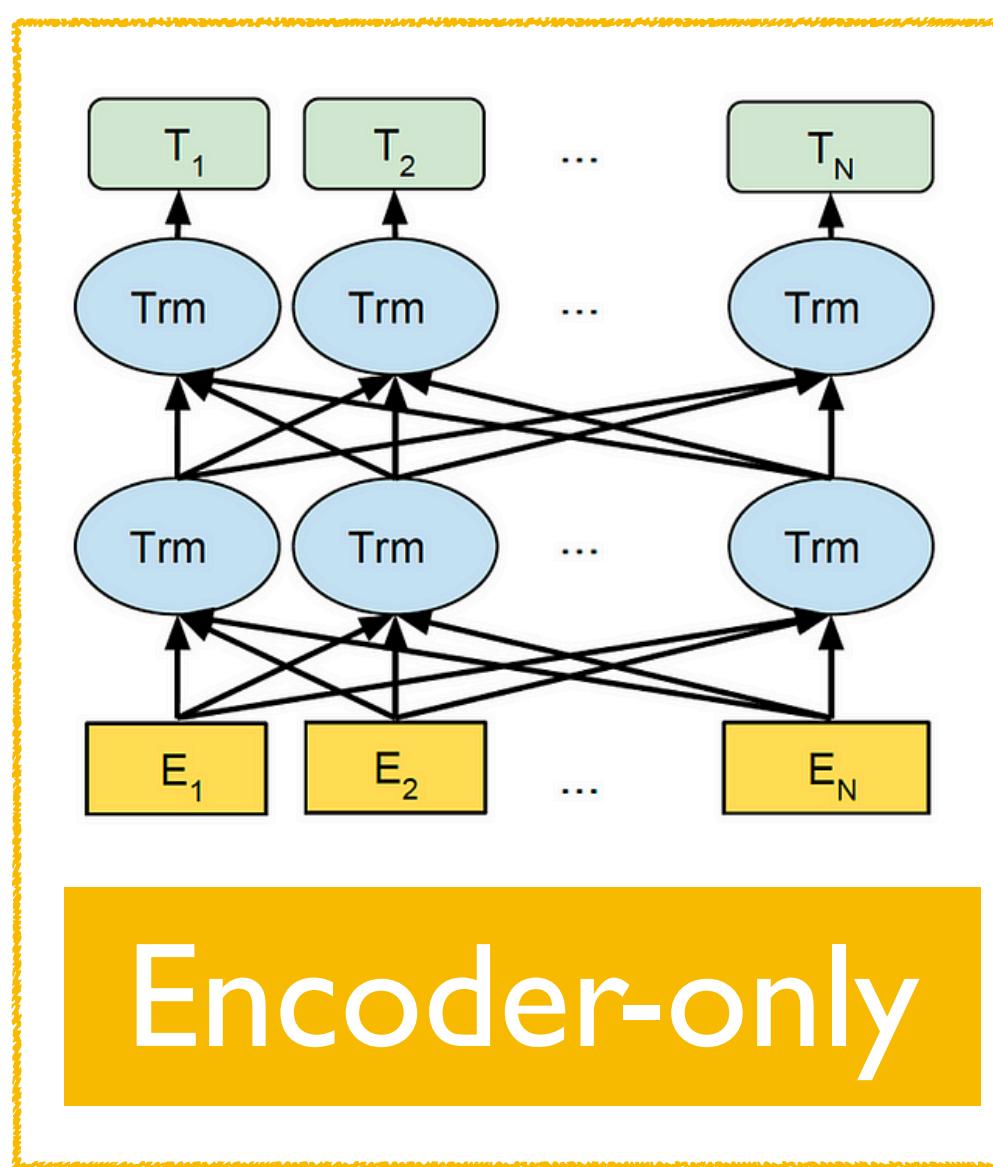
Autoregressive LM

The capital city of Ontario is _____

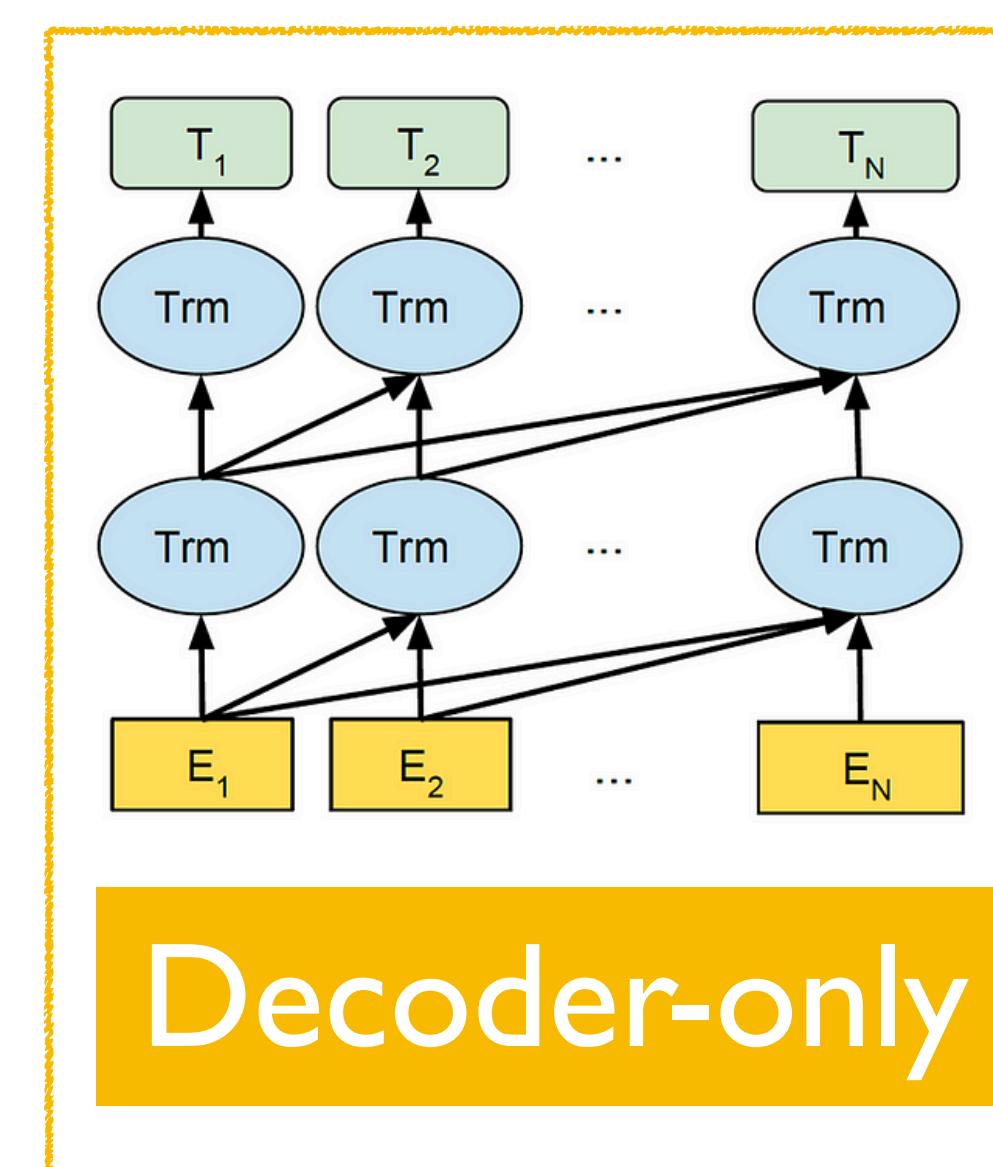
capital Ontario

Masked LM

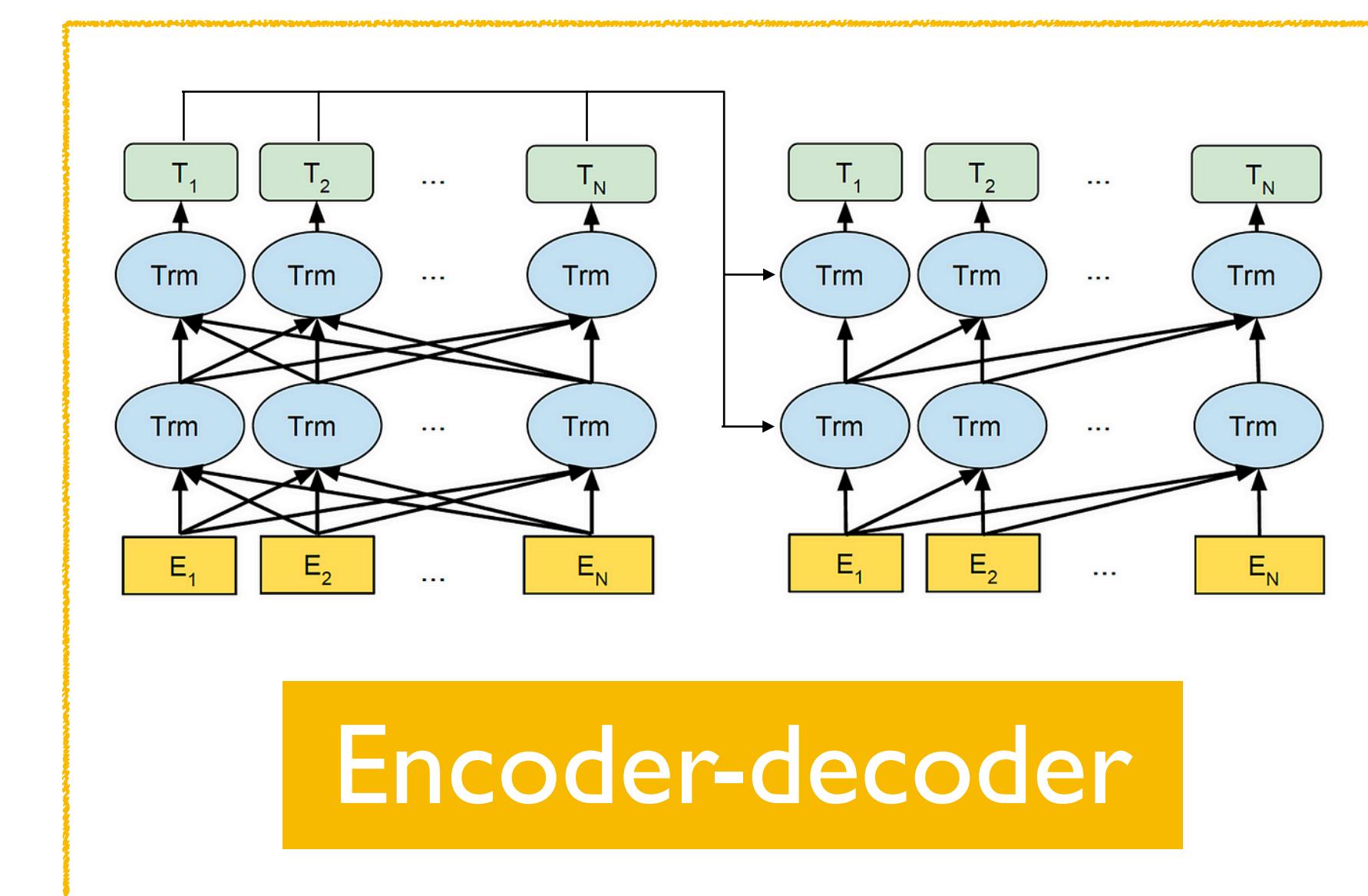
The _____ city of _____ is Toronto



Encoder-only



Decoder-only

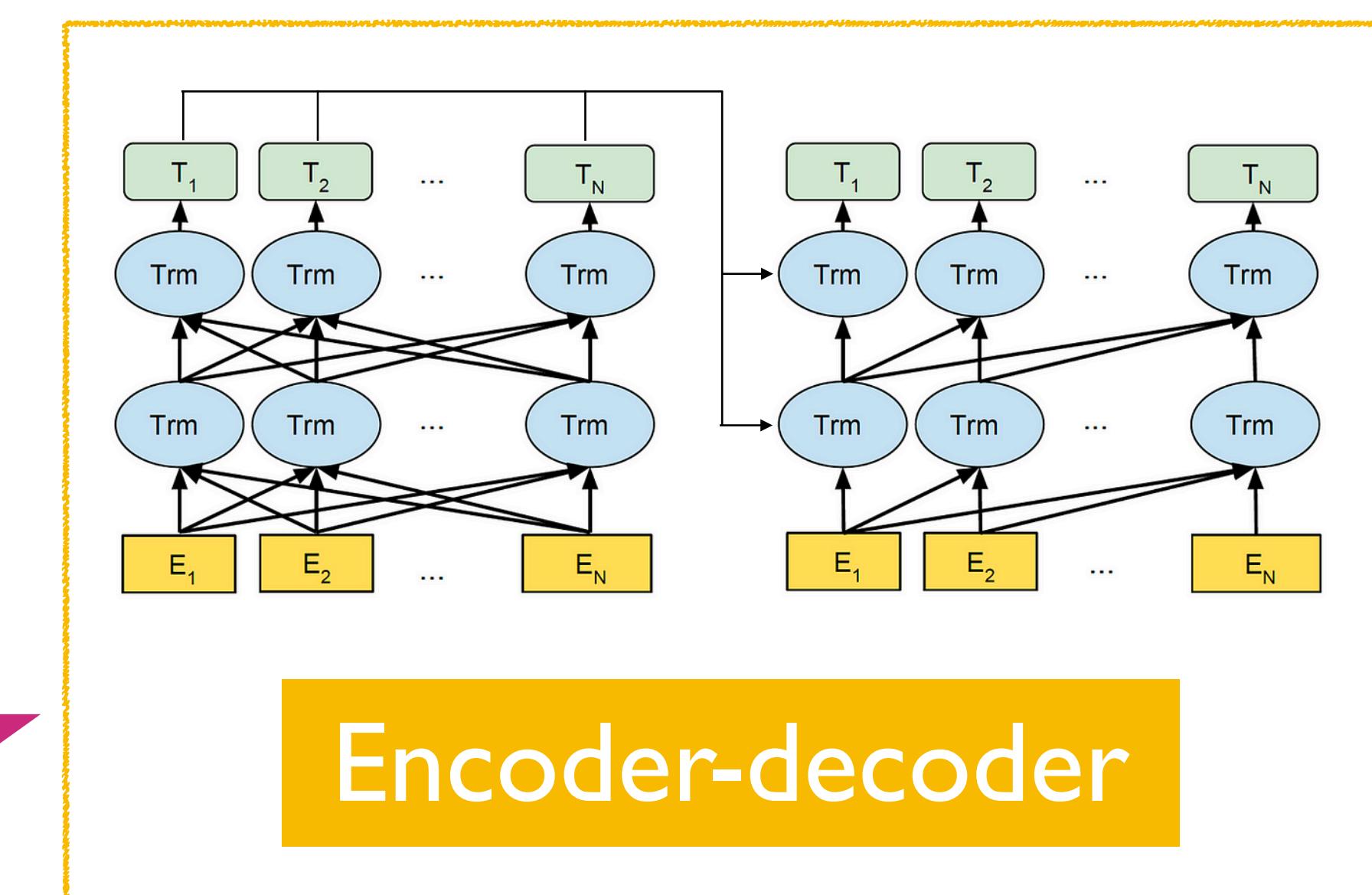
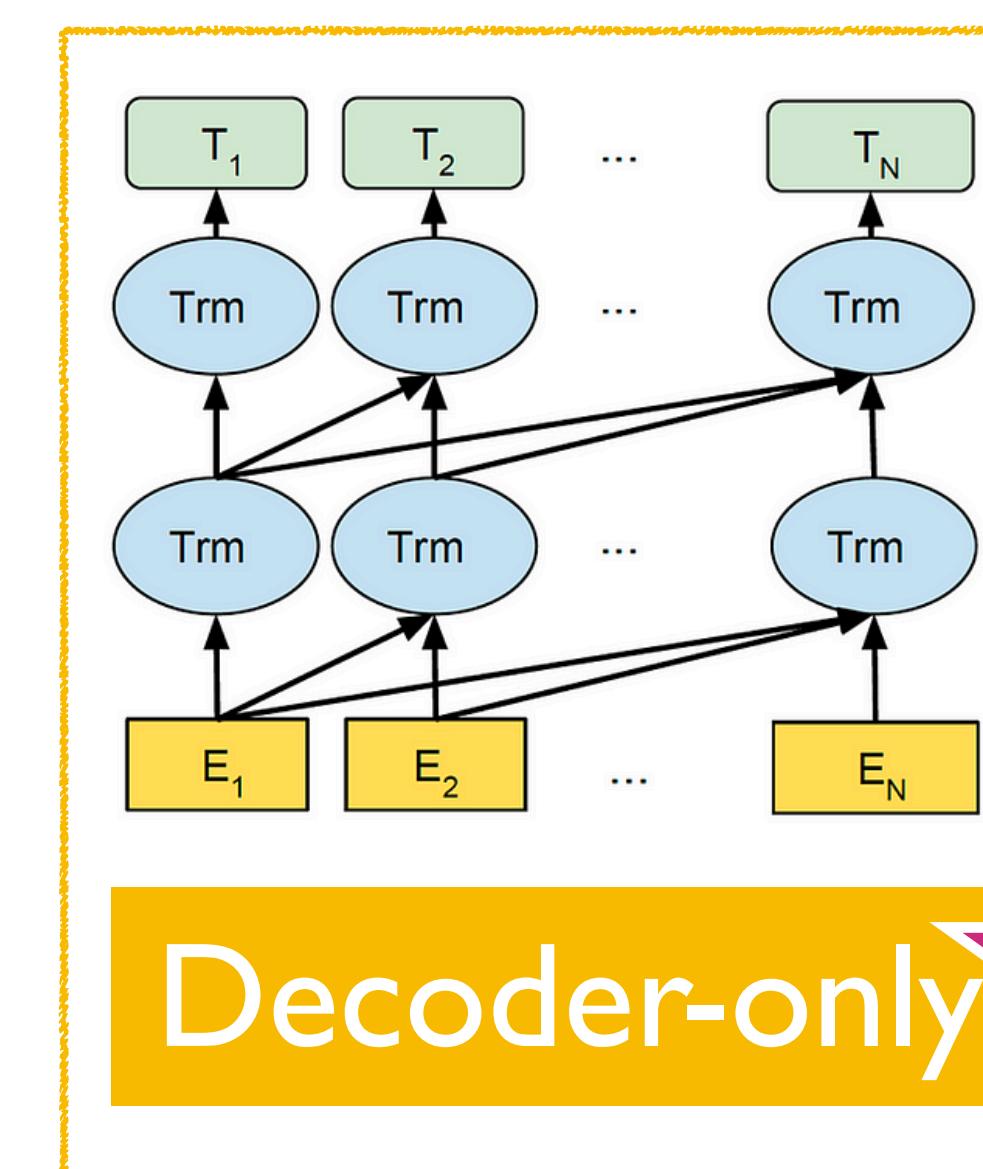
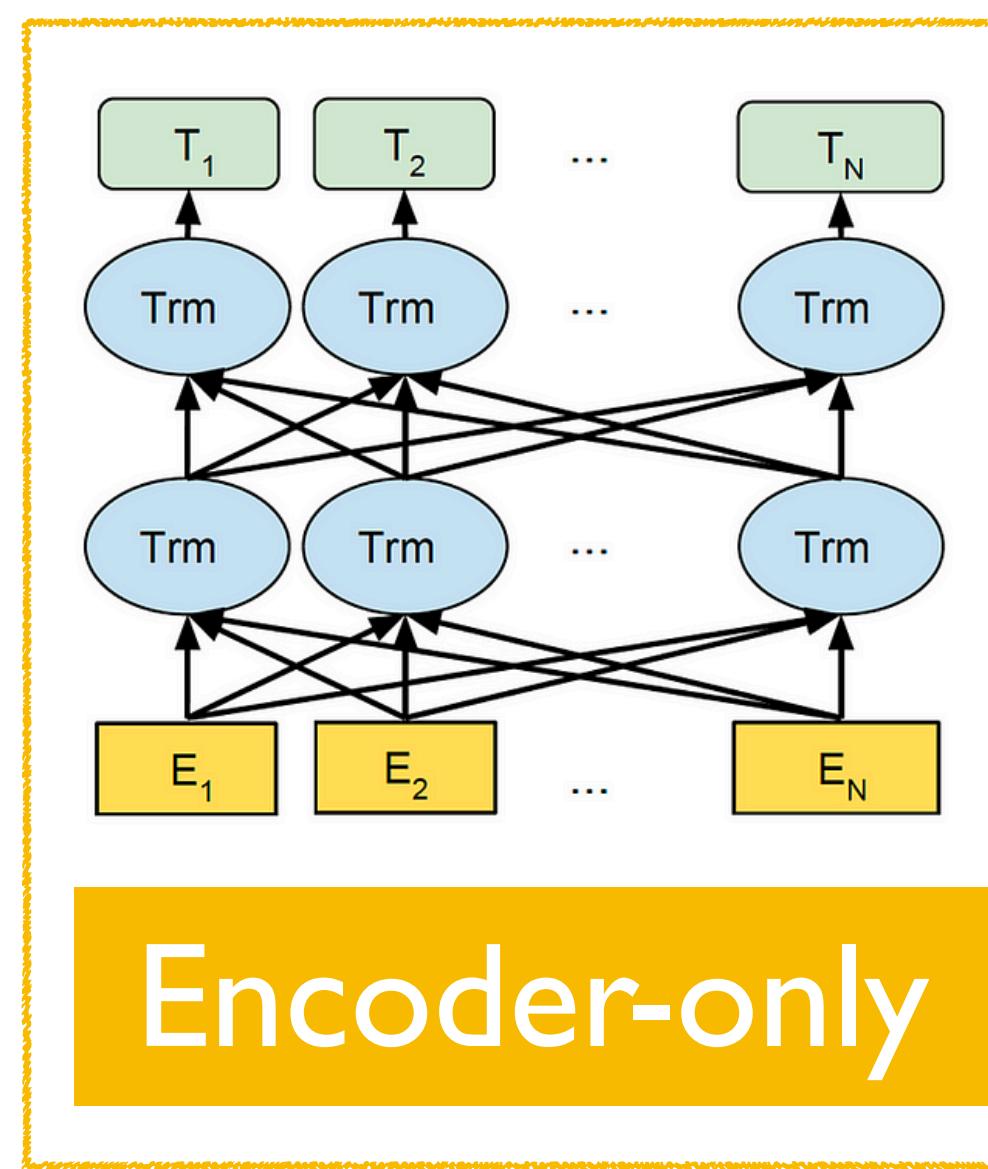


Encoder-decoder

A language model (LM): Categories

Toronto
Autoregressive LM 
The capital city of Ontario is _____

capital Ontario
Masked LM
The _____ city of _____ is Toronto



A language model (LM): Prompting

A language model (LM): Prompting

The capital city of Ontario is

LM

Toronto

Fact probing

A language model (LM): Prompting

The capital city of Ontario is

LM

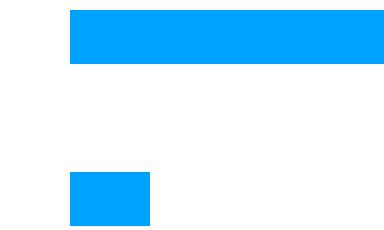
Toronto

Fact probing

Cheaper than an iPod. It was

LM

great
terrible



Sentiment
analysis

A language model (LM): Prompting

The capital city of Ontario is

LM

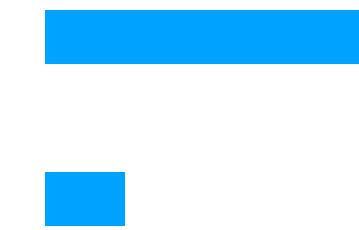
Toronto

Fact probing

Cheaper than an iPod. It was

LM

great
terrible



Sentiment
analysis

“Hello” in French is

LM

Bonjourno

Translation

A language model (LM): Prompting

The capital city of Ontario is

LM

Toronto

Fact probing

Cheaper than an iPod. It was

LM

great
terrible

Sentiment
analysis

“Hello” in French is

LM

Bonjourno

Translation

I'm good at math. $5 + 8 \times 12 =$

LM

101

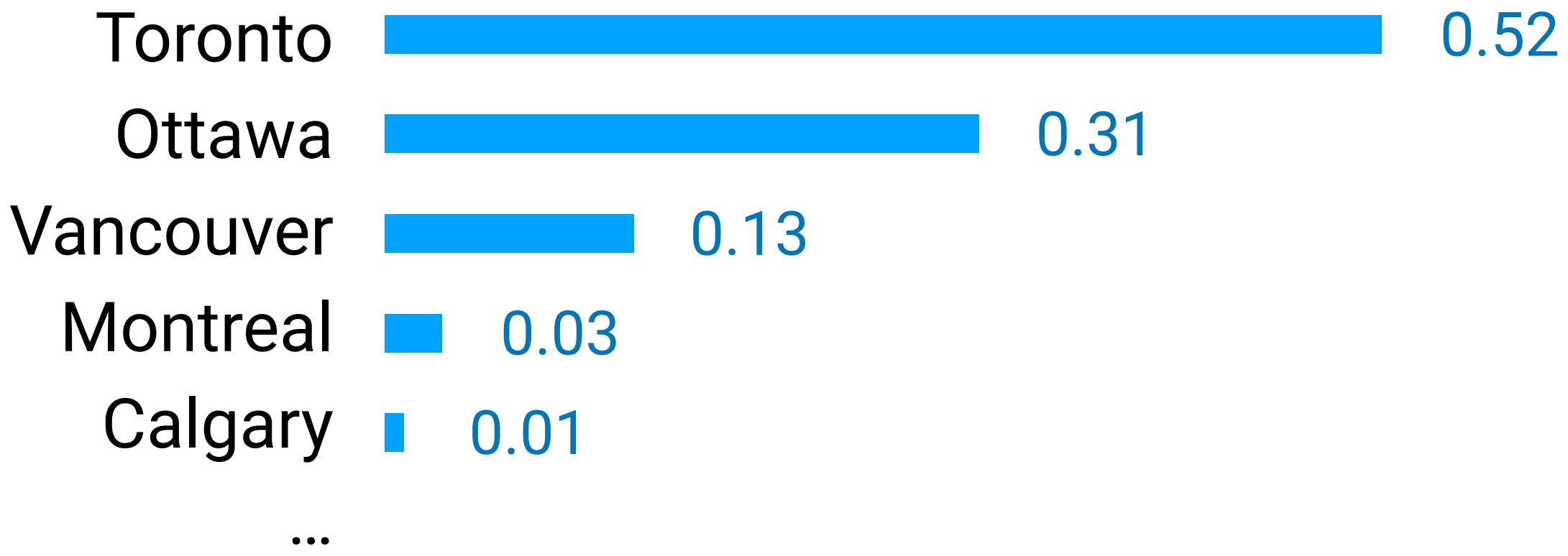
Arithmetic

A language model (LM)

Often evaluated with

A language model (LM)

Often evaluated with

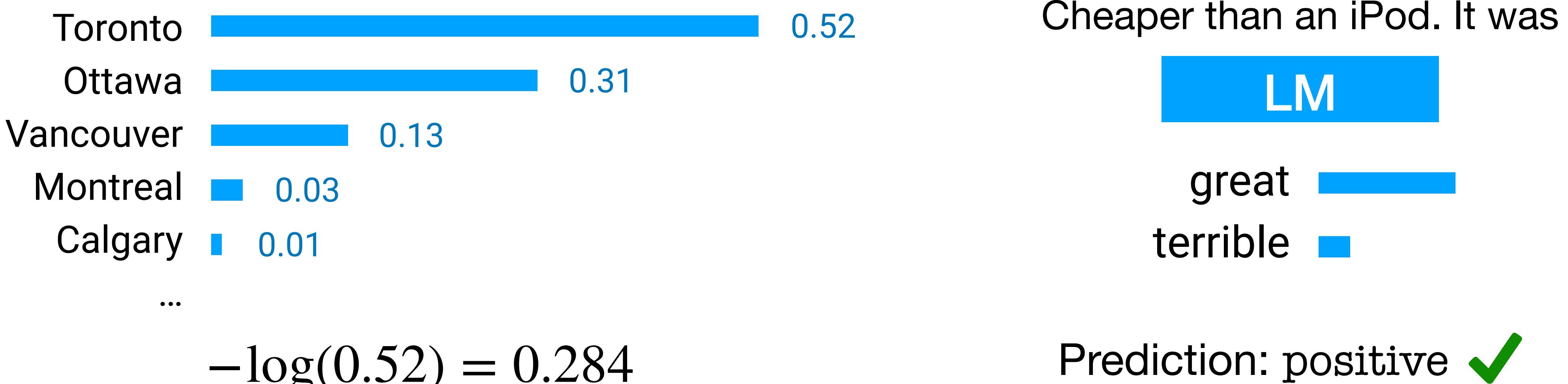


$$-\log(0.52) = 0.284$$

Perplexity

A language model (LM)

Often evaluated with



Perplexity

Downstream accuracy

(Zero-shot or few-shot in-context learning,
or fine-tuning)

(More in Section 5)

A Retrieval-based LM: Definition

**A language model (LM) that uses
an external datastore at test time**

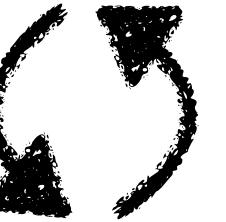
A Retrieval-based LM: Definition

A language model (LM) that uses
an external datastore at test time

Typical LMs



The capital city of Ontario is **Toronto**



LM

Training time

The capital city of Ontario is _____



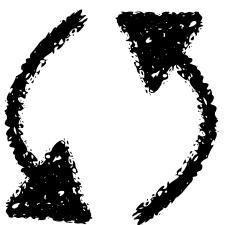
LM

Test time

Retrieval-based LMs



The capital city of Ontario is **Toronto**



LM

Training time



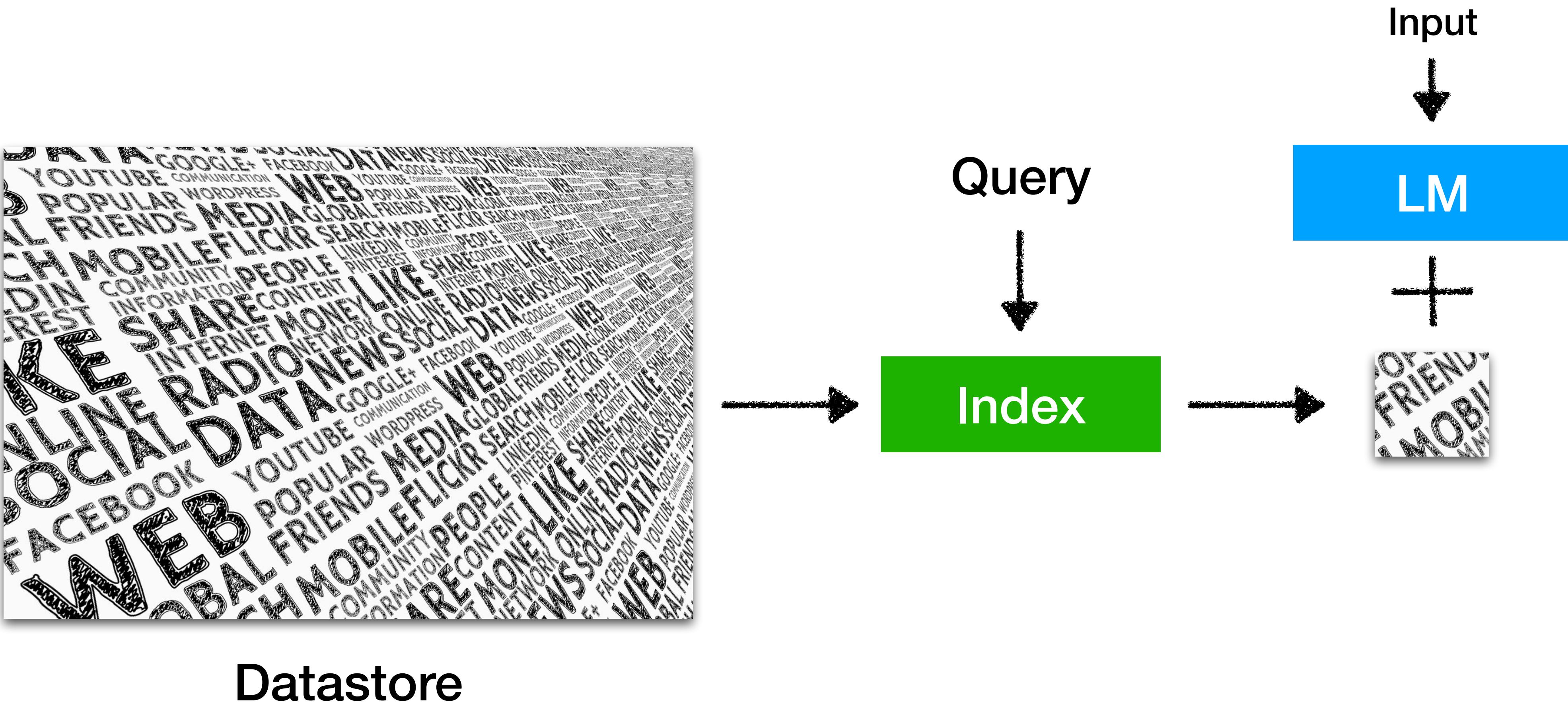
The capital city of Ontario is _____



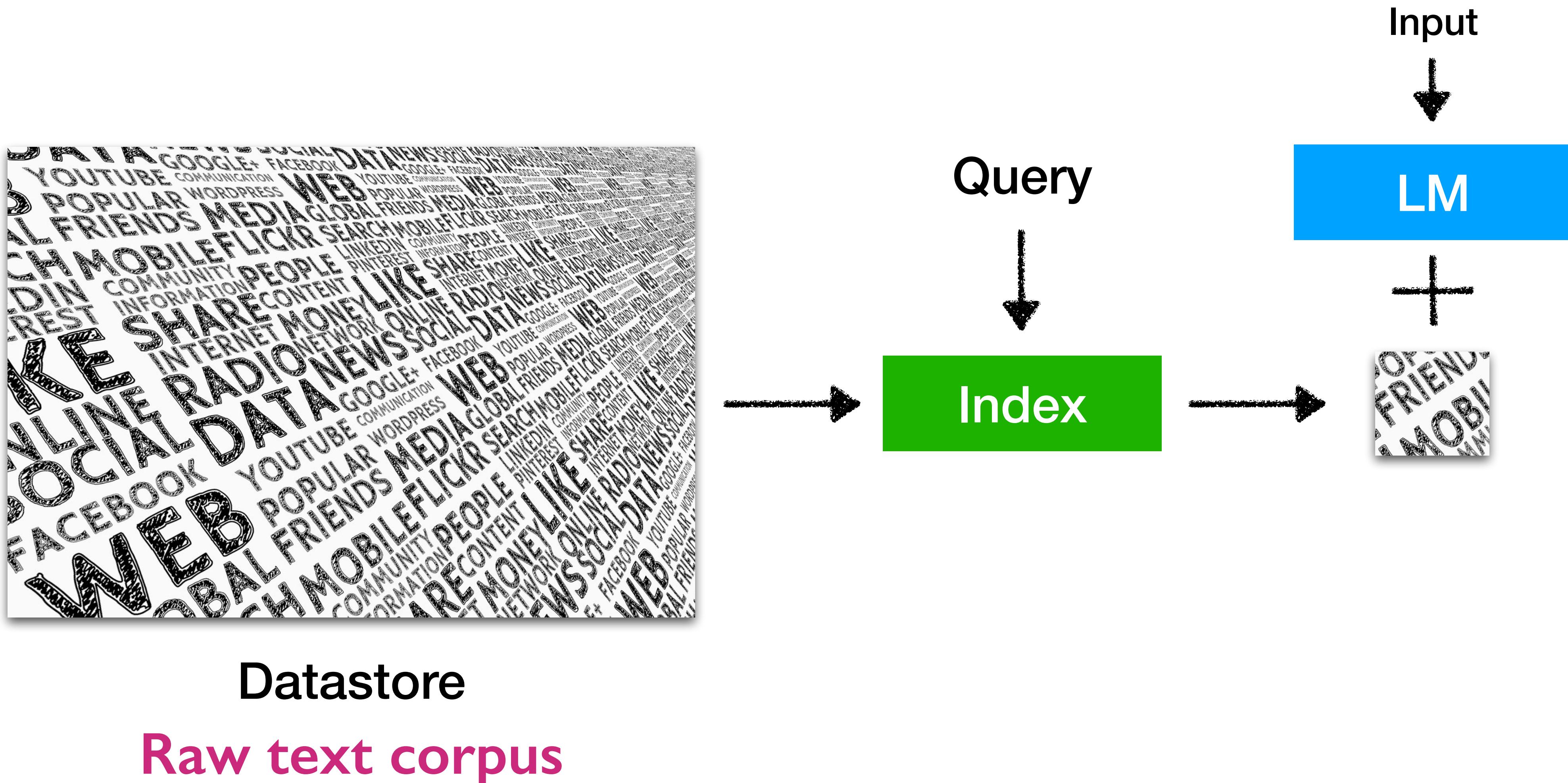
LM

Test time

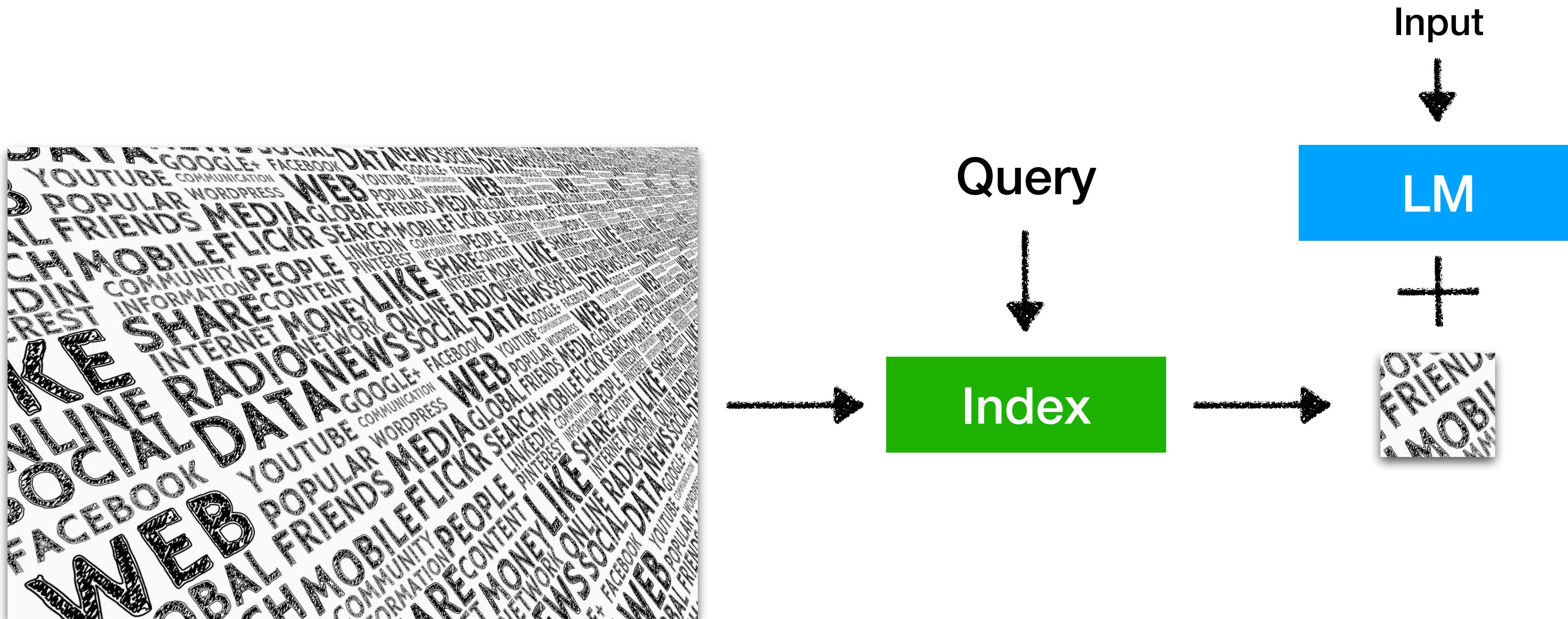
Inference



Inference: Datastore



Inference: Datastore

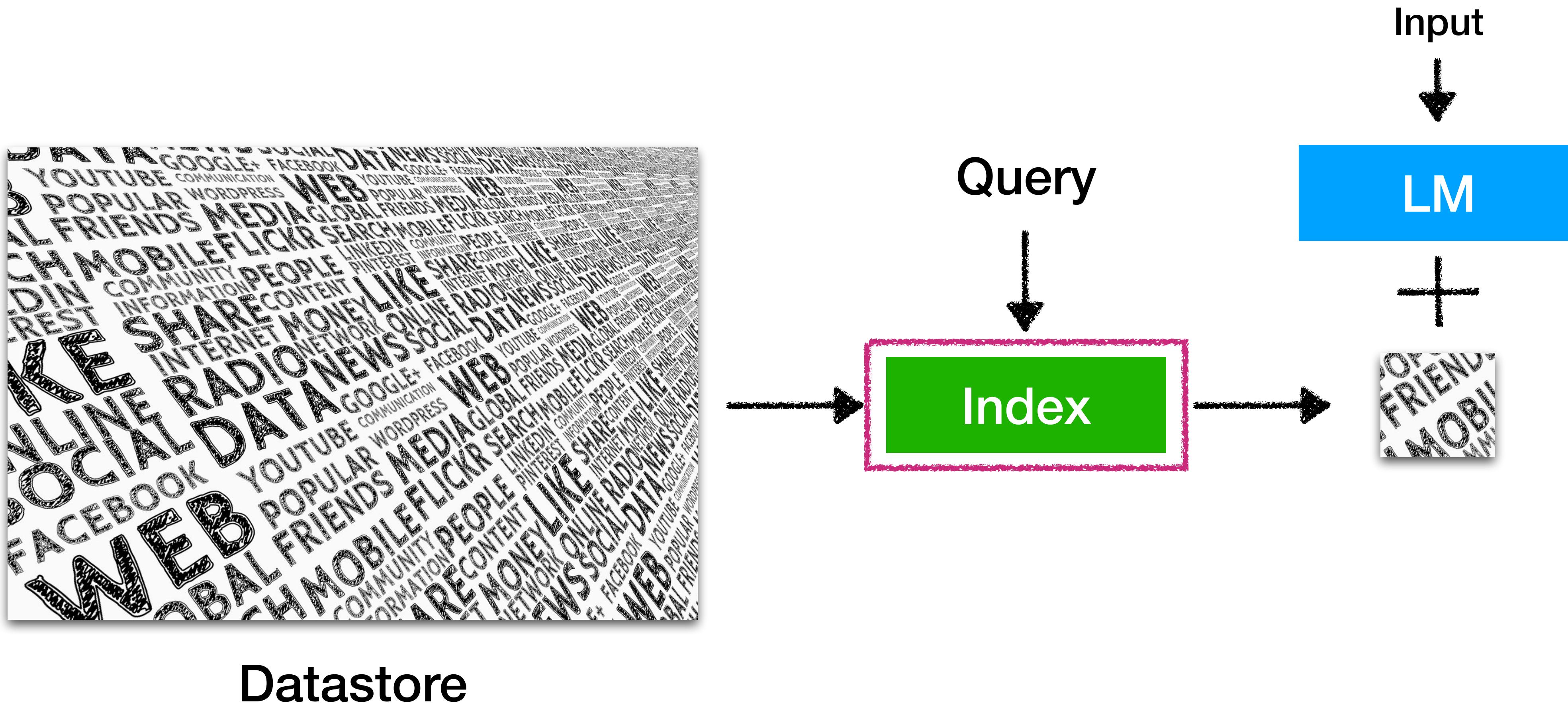


Datastore

Raw text corpus

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

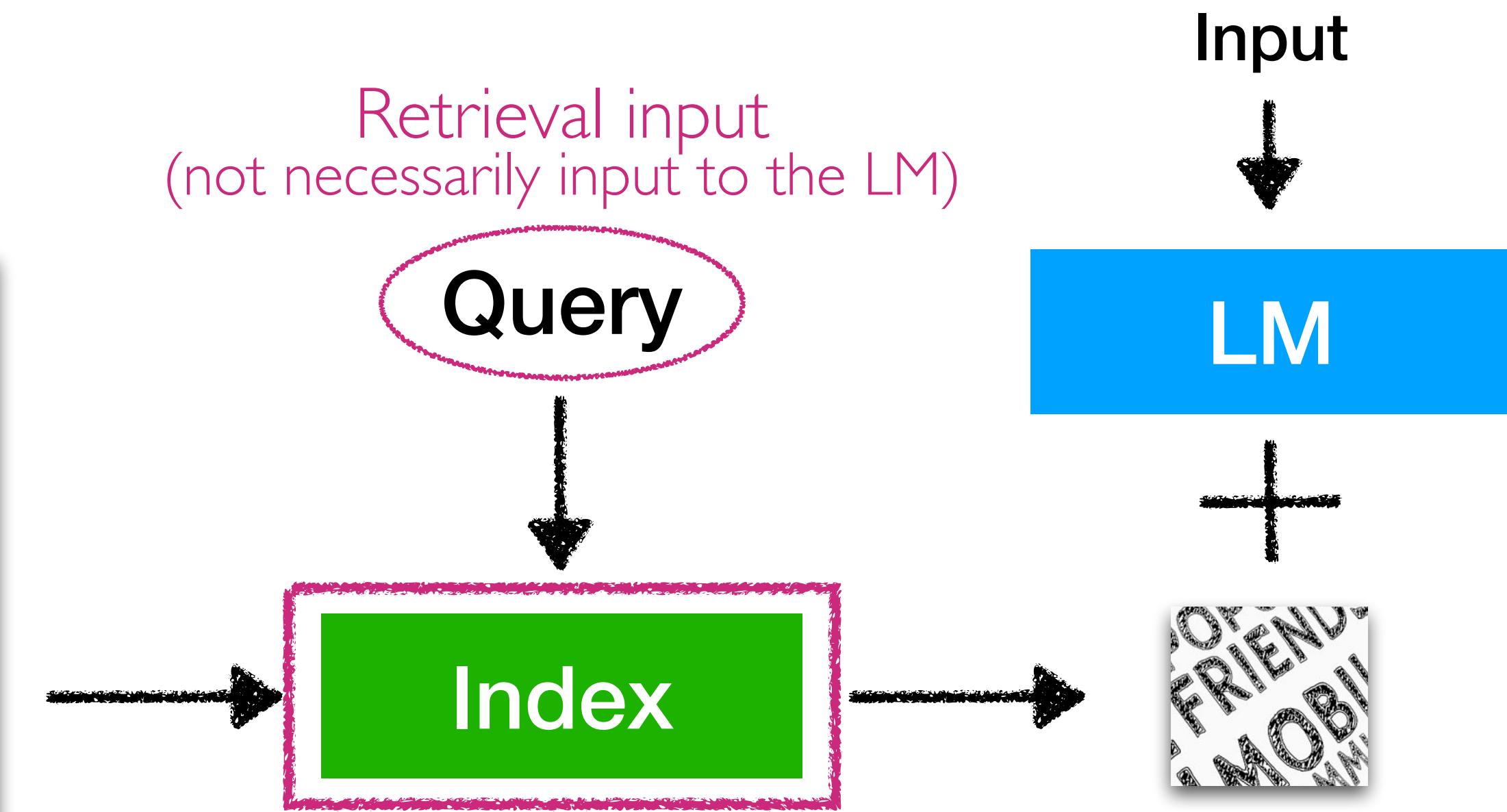
Inference: Index



Inference: Index



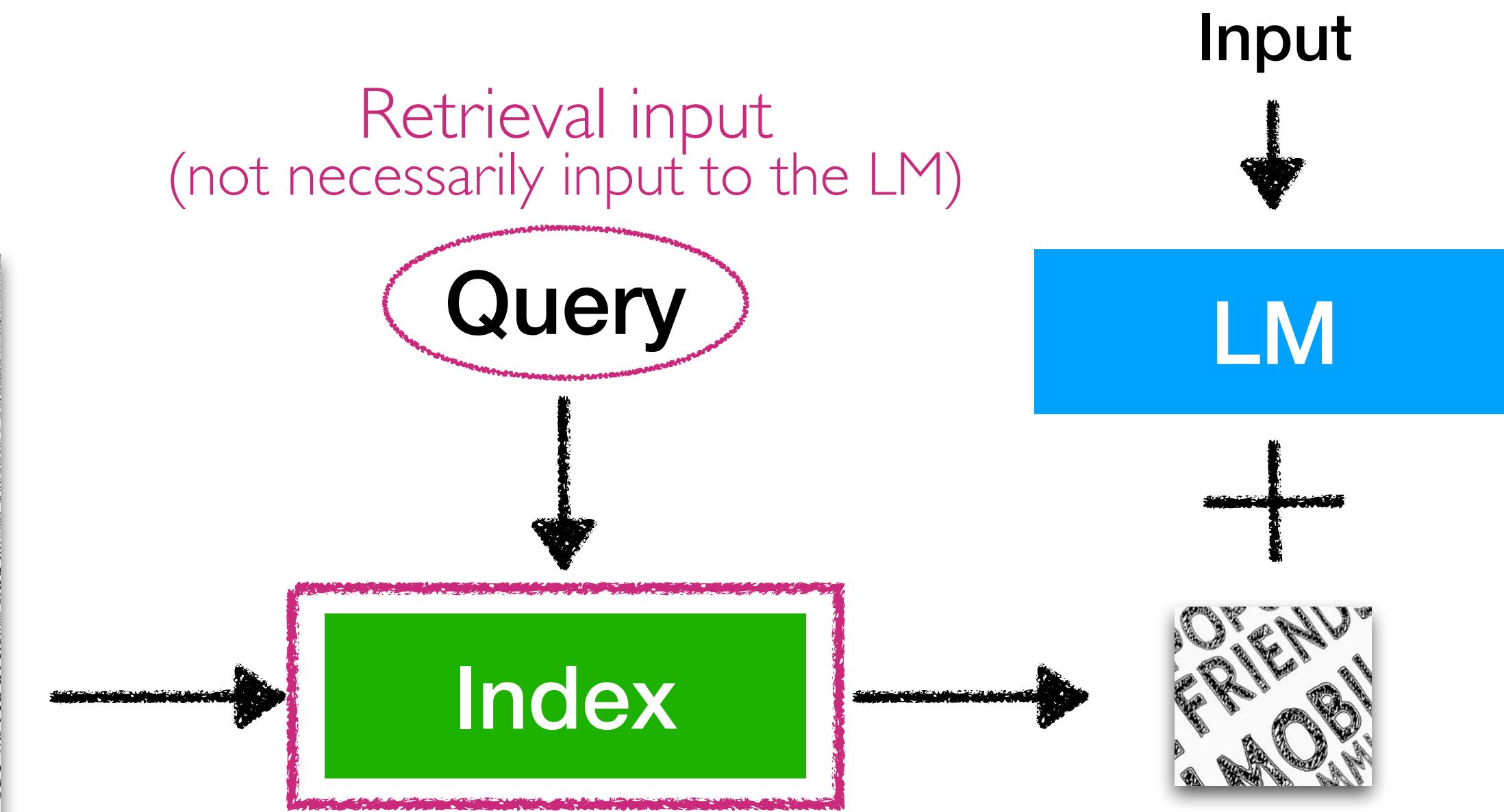
Datastore



Inference: Index



Datastore



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

Inference: Index

Goal: find a small subset of elements in a datastore
that are the most similar to the query

Inference: Index

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Inference: Index

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Example

$$\text{sim}(i, j) = \frac{\text{tf}_{i,j} \times \log \frac{N}{\text{df}_i}}{\# \text{ of occurrences of } i \text{ in } j}$$

of total docs
of docs containing i

Inference: Index

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Example $\text{sim}(i, j) = \frac{\text{tf}_{i,j} \times \log \frac{N}{\text{df}_i}}{\# \text{ of occurrences of } i \text{ in } j}$

of total docs
of docs containing i

Example $\text{sim}(i, j) = \underline{\text{Encoder}(i)} \cdot \underline{\text{Encoder}(j)}$

Maps the text into an h -dimensional vector

Inference: Index

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Example

$$\text{sim}(i, j) = \frac{\text{tf}_{i,j} \times \log \frac{N}{\text{df}_i}}{\# \text{ of occurrences of } i \text{ in } j}$$

of total docs
of docs containing i

Example

$$\text{sim}(i, j) = \underline{\text{Encoder}(i)} \cdot \underline{\text{Encoder}(j)}$$

Maps the text into an h -dimensional vector

An entire field of
study on how to get
(or learn) the
similarity function
better
(We'll see some in
Section 4)

Inference: Index

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Index: given q , return $\arg \text{Top-}k_{d \in \mathcal{D}} \text{sim}(q, d)$ through fast nearest neighbor search

Inference: Index

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Index: given q , return $\arg \text{Top-}k_{d \in \mathcal{D}} \text{sim}(q, d)$ through fast nearest neighbor search

k elements from a datastore

Inference: Index

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Can be a totally separate research area on
how to do this fast & accurate

Index: given q , return $\arg \text{Top-}k_{d \in \mathcal{D}} \text{sim}(q, d)$ through fast nearest neighbor search

k elements from a datastore

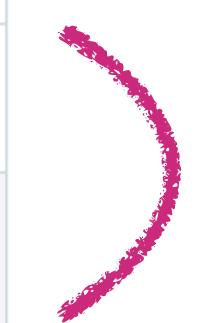
Software: FAISS, Distributed FAISS, SCaNN, etc...

Software: FAISS, Distributed FAISS, SCaNN, etc...

Method	Class name	<code>index_factory</code>	Main parameters	Bytes/vector	Exhaustive	Comments
Exact Search for L2	<code>IndexFlatL2</code>	"Flat"	<code>d</code>	<code>4*d</code>	yes	brute-force
Exact Search for Inner Product	<code>IndexFlatIP</code>	"Flat"	<code>d</code>	<code>4*d</code>	yes	also for cosine (normalize vectors beforehand)
Hierarchical Navigable Small World graph exploration	<code>IndexHNSWFlat</code>	"HNSW,Flat"	<code>d, M</code>	$4*d + x * M * 2 * 4$	no	
Inverted file with exact post-verification	<code>IndexIVFFlat</code>	"IVFx,Flat"	<code>quantizer, d, nlists, metric</code>	$4*d + 8$	no	Takes another index to assign vectors to inverted lists. The 8 additional bytes are the vector id that needs to be stored.
Locality-Sensitive Hashing (binary flat index)	<code>IndexLSH</code>	-	<code>d, nbits</code>	<code>ceil(nbites/8)</code>	yes	optimized by using random rotation instead of random projections
Scalar quantizer (SQ) in flat mode	<code>IndexScalarQuantizer</code>	"SQ8"	<code>d</code>	<code>d</code>	yes	4 and 6 bits per component are also implemented.
Product quantizer (PQ) in flat mode	<code>IndexPQ</code>	"PQx", "PQ" "M" "x" "nbites"	<code>d, M, nbits</code>	$\text{ceil}(M * nbites / 8)$	yes	
IVF and scalar quantizer	<code>IndexIVFScalarQuantizer</code>	"IVFx,SQ4" "IVFx,SQ8"	<code>quantizer, d, nlists, qtype</code>	$SQfp16: 2 * d + 8, SQ8: d + 8 \text{ or } SQ4: d/2 + 8$	no	Same as the <code>IndexScalarQuantizer</code>
IVFADC (coarse quantizer+PQ on residuals)	<code>IndexIVFPQ</code>	"IVFx,PQ" "y" "x" "nbites"	<code>quantizer, d, nlists, M, nbits</code>	$\text{ceil}(M * nbites/8) + 8$	no	
IVFADC+R (same as IVFADC with re-ranking based on codes)	<code>IndexIVFPQR</code>	"IVFx,PQy+z"	<code>quantizer, d, nlists, M, nbits, M_refine, nbits_refine</code>	<code>M+M_refine+8</code>	no	

Software: FAISS, Distributed FAISS, SCaNN, etc...

Method	Class name	<code>index_factory</code>	Main parameters	Bytes/vector	Exhaustive	Comments
Exact Search for L2	<code>IndexFlatL2</code>	<code>"Flat"</code>	<code>d</code>	<code>4*d</code>	yes	brute-force
Exact Search for Inner Product	<code>IndexFlatIP</code>	<code>"Flat"</code>	<code>d</code>	<code>4*d</code>	yes	also for cosine (normalize vectors beforehand)
Hierarchical Navigable Small World graph exploration	<code>IndexHNSWFlat</code>	<code>"HNSW,Flat"</code>	<code>d, M</code>	<code>4*d + x * M * 2 * 4</code>	no	
Inverted file with exact post-verification	<code>IndexIVFFlat</code>	<code>"IVFx,Flat"</code>	<code>quantizer, d, nlists, metric</code>	<code>4*d + 8</code>	no	Takes another index to assign vectors to inverted lists. The 8 additional bytes are the vector id that needs to be stored.
Locality-Sensitive Hashing (binary flat index)	<code>IndexLSH</code>	-	<code>d, nbits</code>	<code>ceil(nbites/8)</code>	yes	optimized by using random rotation instead of random projections
Scalar quantizer (SQ) in flat mode	<code>IndexScalarQuantizer</code>	<code>"SQ8"</code>	<code>d</code>	<code>d</code>	yes	4 and 6 bits per component are also implemented.
Product quantizer (PQ) in flat mode	<code>IndexPQ</code>	<code>"PQx", "PQ" "M" "x" "nbites"</code>	<code>d, M, nbits</code>	<code>ceil(M * nbites / 8)</code>	yes	
IVF and scalar quantizer	<code>IndexIVFScalarQuantizer</code>	<code>"IVFx,SQ4", "IVFx,SQ8"</code>	<code>quantizer, d, nlists, qtype</code>	<code>SQfp16: 2 * d + 8, SQ8: d + 8 or SQ4: d/2 + 8</code>	no	Same as the <code>IndexScalarQuantizer</code>
IVFADC (coarse quantizer+PQ on residuals)	<code>IndexIVFPQ</code>	<code>"IVFx,PQ" "y" "x" "nbites"</code>	<code>quantizer, d, nlists, M, nbits</code>	<code>ceil(M * nbites/8)+8</code>	no	
IVFADC+R (same as IVFADC with re-ranking based on codes)	<code>IndexIVFPQR</code>	<code>"IVFx,PQy+z"</code>	<code>quantizer, d, nlists, M, nbits, M_refine, nbits_refine</code>	<code>M+M_refine+8</code>	no	



Exact Search

Software: FAISS, Distributed FAISS, SCaNN, etc...

Method	Class name	<code>index_factory</code>	Main parameters	Bytes/vector	Exhaustive	Comments
Exact Search for L2	<code>IndexFlatL2</code>	"Flat"	<code>d</code>	<code>4*d</code>	yes	brute-force
Exact Search for Inner Product	<code>IndexFlatIP</code>	"Flat"	<code>d</code>	<code>4*d</code>	yes	also for cosine (normalize vectors beforehand)
Hierarchical Navigable Small World graph exploration	<code>IndexHNSWFlat</code>	"HNSW,Flat"	<code>d, M</code>	$4*d + x * M * 2 * 4$	no	
Inverted file with exact post-verification	<code>IndexIVFFlat</code>	"IVFx,Flat"	<code>quantizer, d, nlists, metric</code>	$4*d + 8$	no	Takes another index to assign vectors to inverted lists. The 8 additional bytes are the vector id that needs to be stored.
Locality-Sensitive Hashing (binary flat index)	<code>IndexLSH</code>	-	<code>d, nbits</code>	<code>ceil(nbites/8)</code>	yes	optimized by using random rotation instead of random projections
Scalar quantizer (SQ) in flat mode	<code>IndexScalarQuantizer</code>	"SQ8"	<code>d</code>	<code>d</code>	yes	4 and 6 bits per component are also implemented.
Product quantizer (PQ) in flat mode	<code>IndexPQ</code>	"PQx", "PQM"x"nbits"	<code>d, M, nbits</code>	$\text{ceil}(M * nbites / 8)$	yes	
IVF and scalar quantizer	<code>IndexIVFScalarQuantizer</code>	"IVFx,SQ4" "IVFx,SQ8"	<code>quantizer, d, nlists, qtype</code>	$SQfp16: 2 * d + 8, SQ8: d + 8 \text{ or } SQ4: d/2 + 8$	no	Same as the <code>IndexScalarQuantizer</code>
IVFADC (coarse quantizer+PQ on residuals)	<code>IndexIVFPQ</code>	"IVFx,PQ"y"x"nbits"	<code>quantizer, d, nlists, M, nbits</code>	$\text{ceil}(M * nbites/8)+8$	no	
IVFADC+R (same as IVFADC with re-ranking based on codes)	<code>IndexIVFPQR</code>	"IVFx,PQy+z"	<code>quantizer, d, nlists, M, nbits, M_refine, nbits_refine</code>	<code>M+M_refine+8</code>	no	

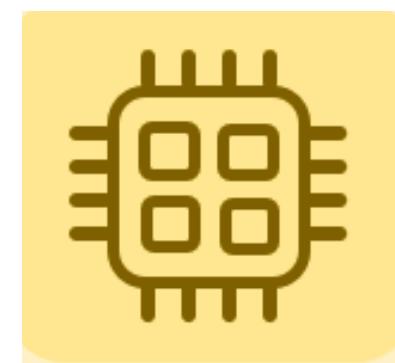
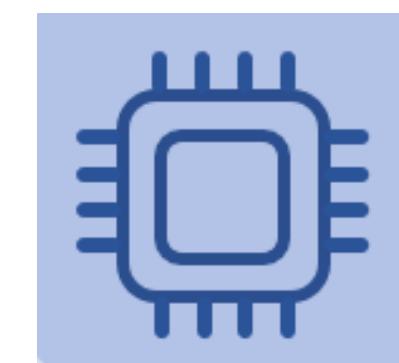
Exact Search

Approximate Search
(Relatively easy to scale to ~1B elements)

Software: FAISS, Distributed FAISS, SCaNN, etc...

Method	Class name	<code>index_factory</code>	Main parameters	Bytes/vector	Exhaustive	Comments
Exact Search for L2	<code>IndexFlatL2</code>	"Flat"	<code>d</code>	<code>4*d</code>	yes	brute-force
Exact Search for Inner Product	<code>IndexFlatIP</code>	"Flat"	<code>d</code>	<code>4*d</code>	yes	also for cosine (normalize vectors beforehand)
Hierarchical Navigable Small World graph exploration	<code>IndexHNSWFlat</code>	"HNSW,Flat"	<code>d, M</code>	$4*d + x * M * 2 * 4$	no	
Inverted file with exact post-verification	<code>IndexIVFFlat</code>	"IVFx,Flat"	<code>quantizer, d, nlists, metric</code>	$4*d + 8$	no	Takes another index to assign vectors to inverted lists. The 8 additional bytes are the vector id that needs to be stored.
Locality-Sensitive Hashing (binary flat index)	<code>IndexLSH</code>	-	<code>d, nbits</code>	<code>ceil(nbites/8)</code>	yes	optimized by using random rotation instead of random projections
Scalar quantizer (SQ) in flat mode	<code>IndexScalarQuantizer</code>	"SQ8"	<code>d</code>	<code>d</code>	yes	4 and 6 bits per component are also implemented.
Product quantizer (PQ) in flat mode	<code>IndexPQ</code>	"PQx", "PQM"x"nbits"	<code>d, M, nbits</code>	$\text{ceil}(M * nbites / 8)$	yes	
IVF and scalar quantizer	<code>IndexIVFScalarQuantizer</code>	"IVFx,SQ4" "IVFx,SQ8"	<code>quantizer, d, nlists, qtype</code>	$SQfp16: 2 * d + 8, SQ8: d + 8 \text{ or } SQ4: d/2 + 8$	no	Same as the <code>IndexScalarQuantizer</code>
IVFADC (coarse quantizer+PQ on residuals)	<code>IndexIVFPQ</code>	"IVFx,PQ"y"x"nbits"	<code>quantizer, d, nlists, M, nbits</code>	$\text{ceil}(M * nbites/8)+8$	no	
IVFADC+R (same as IVFADC with re-ranking based on codes)	<code>IndexIVFPQR</code>	"IVFx,PQy+z"	<code>quantizer, d, nlists, M, nbits, M_refine, nbits_refine</code>	<code>M+M_refine+8</code>	no	

Exact Search



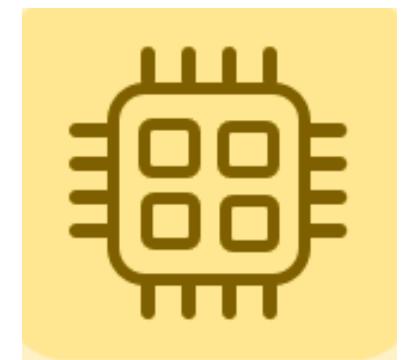
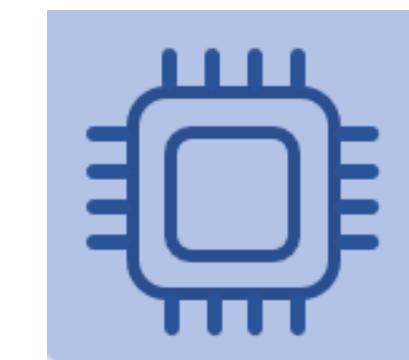
CPU vs. GPU

Approximate Search
(Relatively easy to scale to ~1B elements)

Software: FAISS, Distributed FAISS, SCaNN, etc...

Method	Class name	<code>index_factory</code>	Main parameters	Bytes/vector	Exhaustive	Comments
Exact Search for L2	<code>IndexFlatL2</code>	"Flat"	<code>d</code>	<code>4*d</code>	yes	brute-force
Exact Search for Inner Product	<code>IndexFlatIP</code>	"Flat"	<code>d</code>	<code>4*d</code>	yes	also for cosine (normalize vectors beforehand)
Hierarchical Navigable Small World graph exploration	<code>IndexHNSWFlat</code>	"HNSW,Flat"	<code>d, M</code>	<code>4*d + x * M * 2 * 4</code>	no	
Inverted file with exact post-verification	<code>IndexIVFFlat</code>	"IVFx,Flat"	<code>quantizer, d, nlists, metric</code>	<code>4*d + 8</code>	no	Takes another index to assign vectors to inverted lists. The 8 additional bytes are the vector id that needs to be stored.
Locality-Sensitive Hashing (binary flat index)	<code>IndexLSH</code>	-	<code>d, nbits</code>	<code>ceil(nbites/8)</code>	yes	optimized by using random rotation instead of random projections
Scalar quantizer (SQ) in flat mode	<code>IndexScalarQuantizer</code>	"SQ8"	<code>d</code>	<code>d</code>	yes	4 and 6 bits per component are also implemented.
Product quantizer (PQ) in flat mode	<code>IndexPQ</code>	"PQx", "PQM"x"nbits"	<code>d, M, nbits</code>	<code>ceil(M * nbites / 8)</code>	yes	
IVF and scalar quantizer	<code>IndexIVFScalarQuantizer</code>	"IVFx,SQ4" "IVFx,SQ8"	<code>quantizer, d, nlists, qtype</code>	<code>SQfp16: 2 * d + 8, SQ8: d + 8 or SQ4: d/2 + 8</code>	no	Same as the <code>IndexScalarQuantizer</code>
IVFADC (coarse quantizer+PQ on residuals)	<code>IndexIVFPQ</code>	"IVFx,PQ"y"x"nbits"	<code>quantizer, d, nlists, M, nbits</code>	<code>ceil(M * nbites/8)+8</code>	no	
IVFADC+R (same as IVFADC with re-ranking based on codes)	<code>IndexIVFPQR</code>	"IVFx,PQy+z"	<code>quantizer, d, nlists, M, nbits, M_refine, nbits_refine</code>	<code>M+M_refine+8</code>	no	

Exact Search



CPU vs. GPU

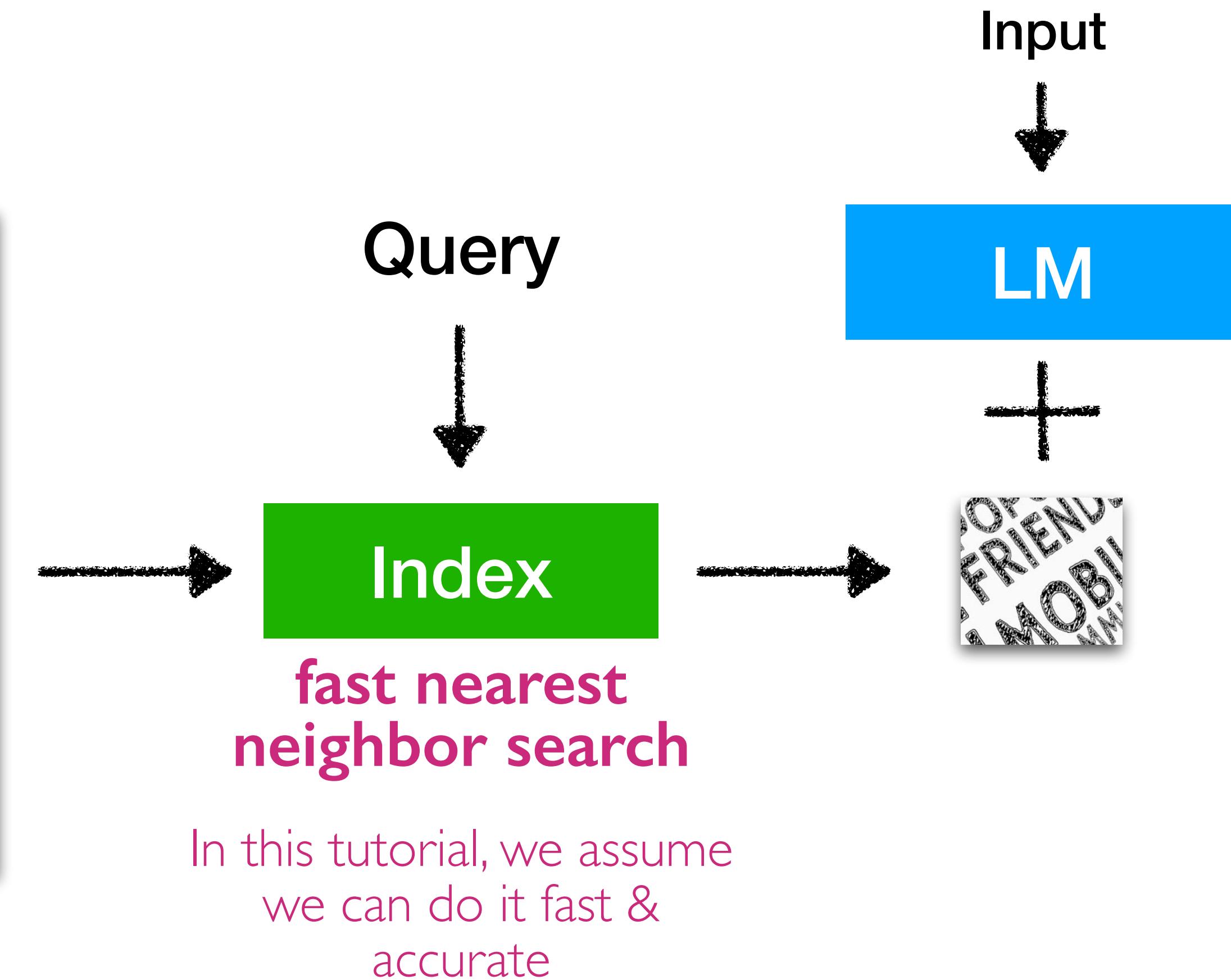
Approximate Search
(Relatively easy to scale to ~1B elements)

More info: <https://github.com/facebookresearch/faiss/wiki>

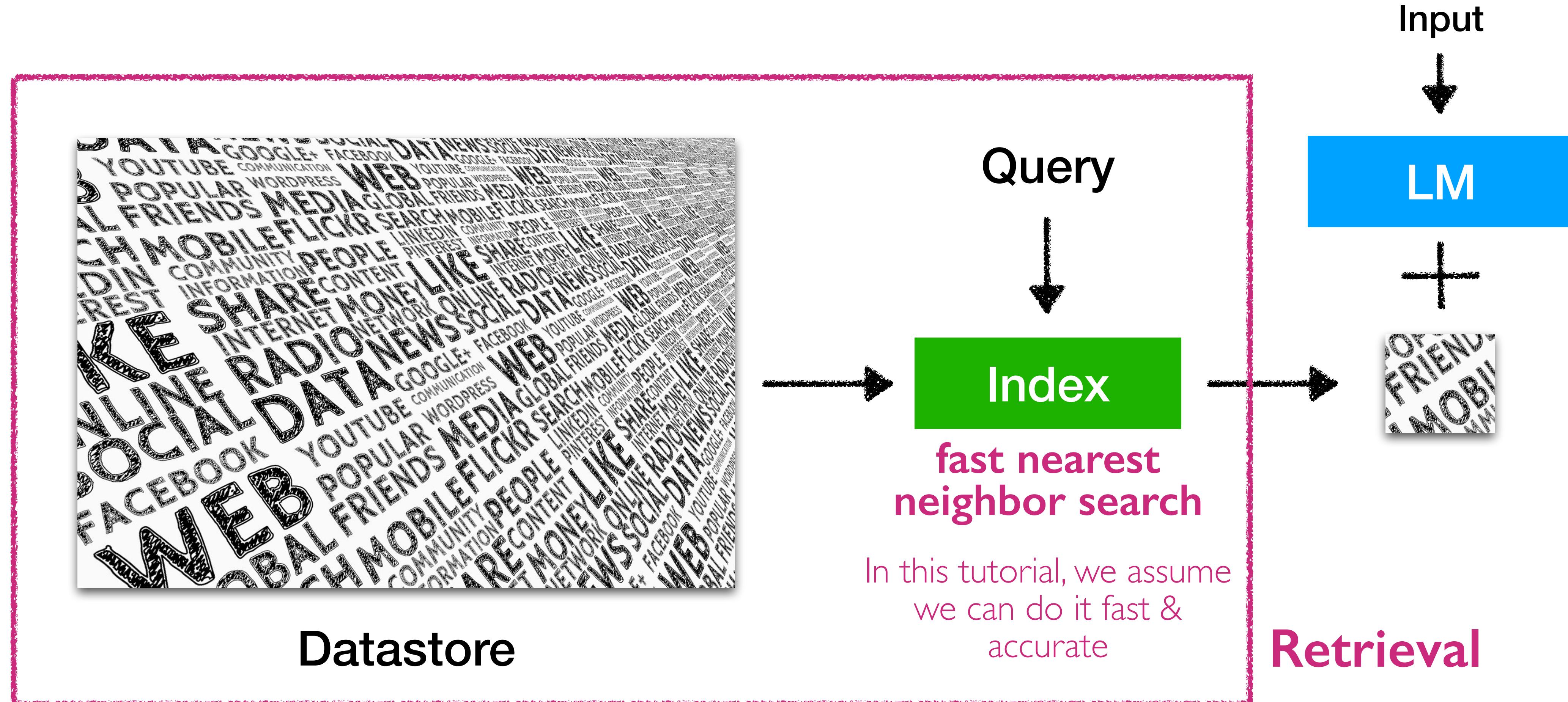
Inference: Search



Datastore



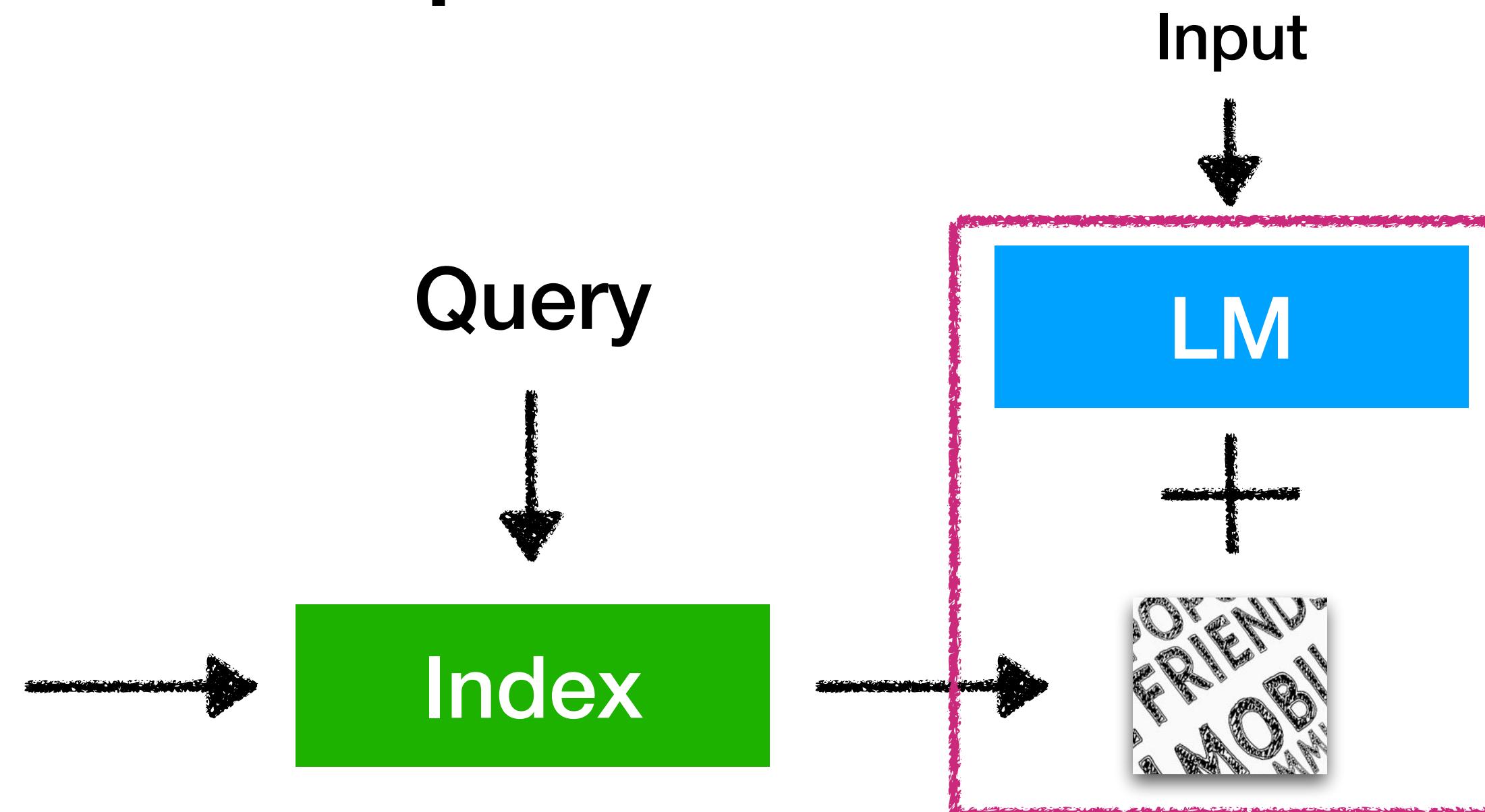
Inference: Search



Inference: Incorporation



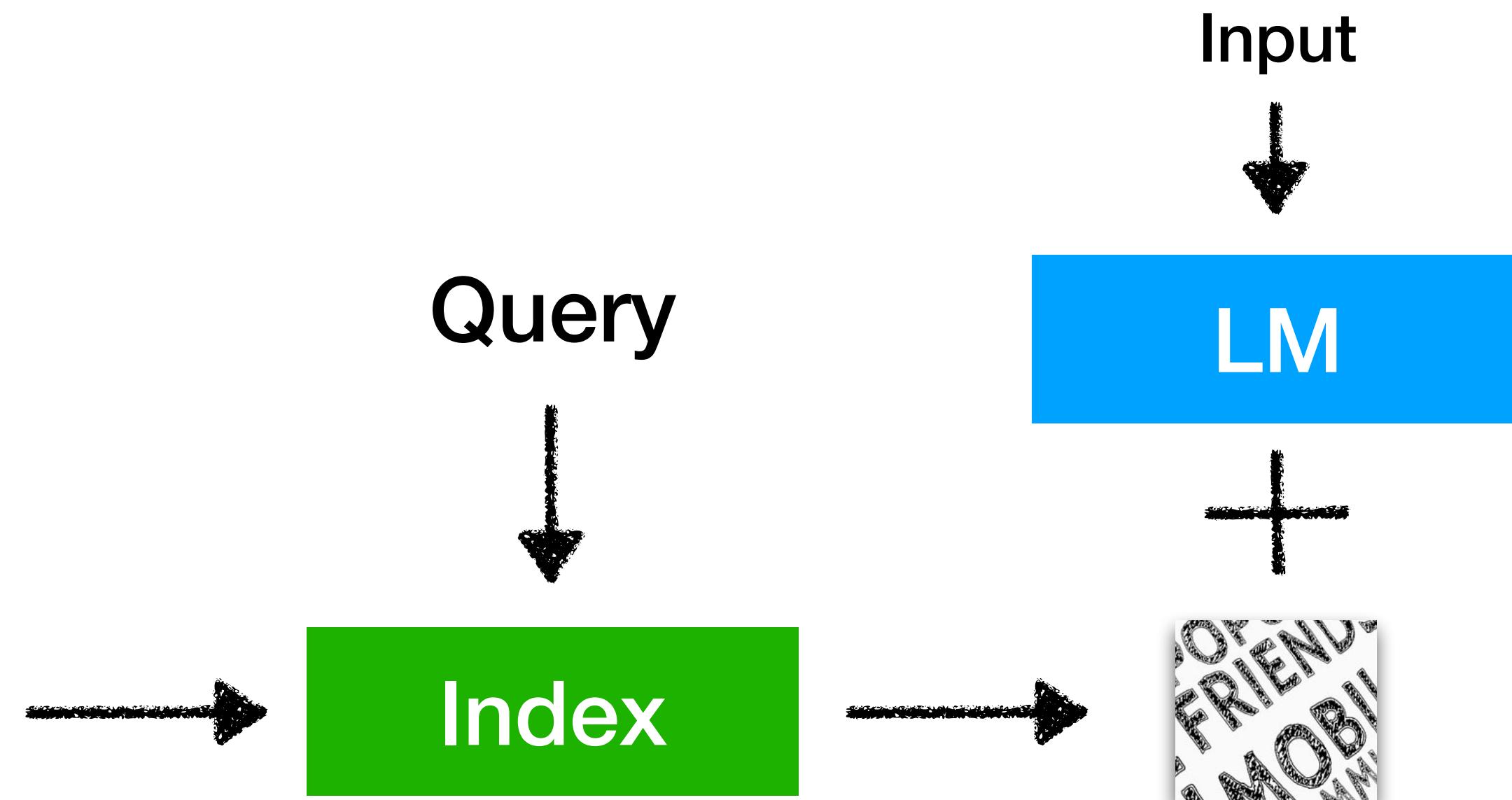
Datastore



Questions to answer



Datastore



Questions to answer

What's the query &
when do we retrieve?

Input

LM

Query

+

Index

FOR
FRIENDS
MOBILE



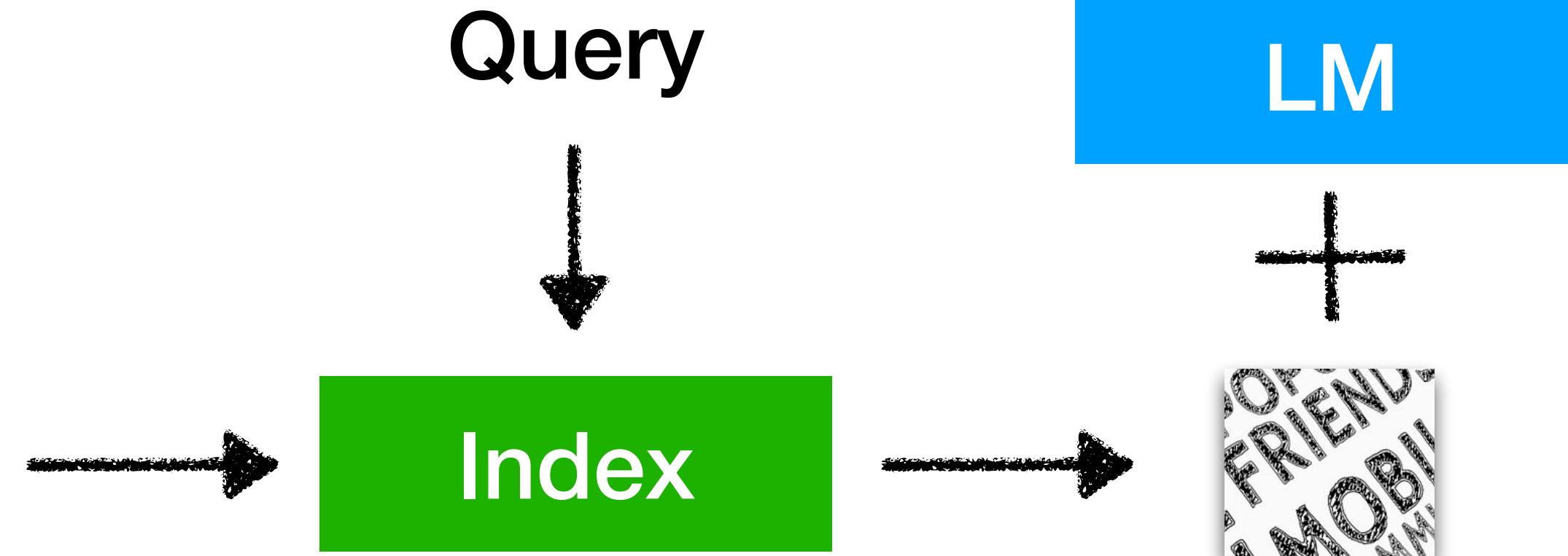
Datastore

Questions to answer



Datastore

What's the query &
when do we retrieve?



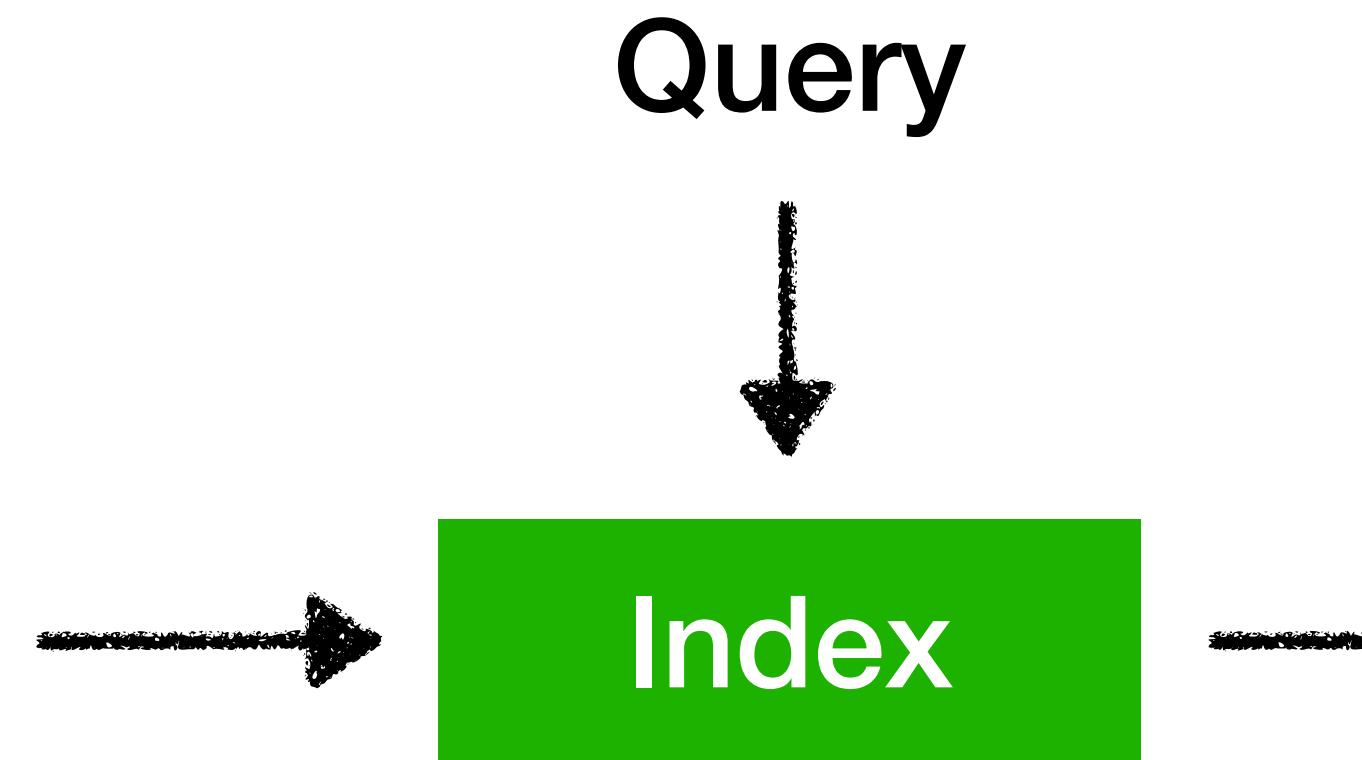
What do we
retrieve?

Questions to answer



Datastore

What's the query &
when do we retrieve?



How do we
use retrieval?

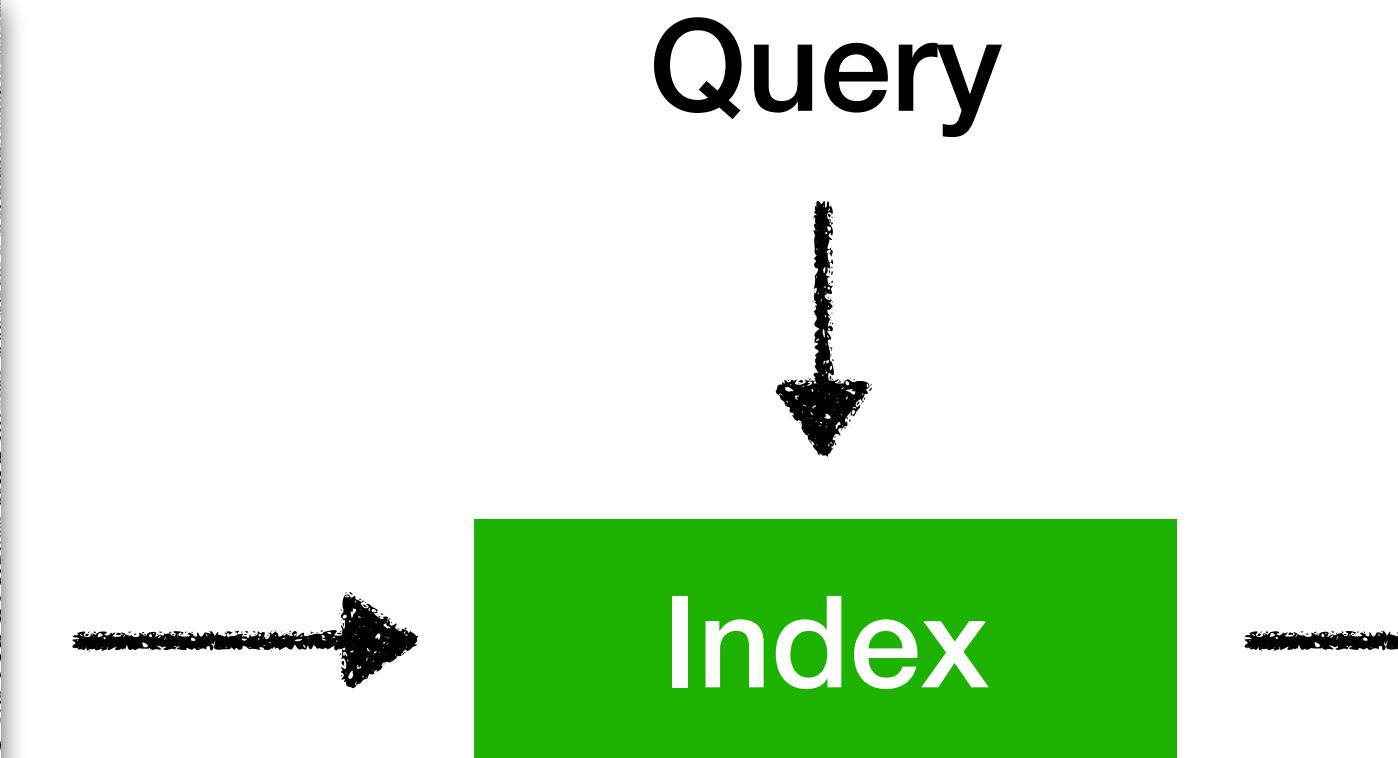
What do we
retrieve?

Questions to answer



Datastore

What's the query &
when do we retrieve?



Input
↓



How do we
use retrieval?

What do we
retrieve?

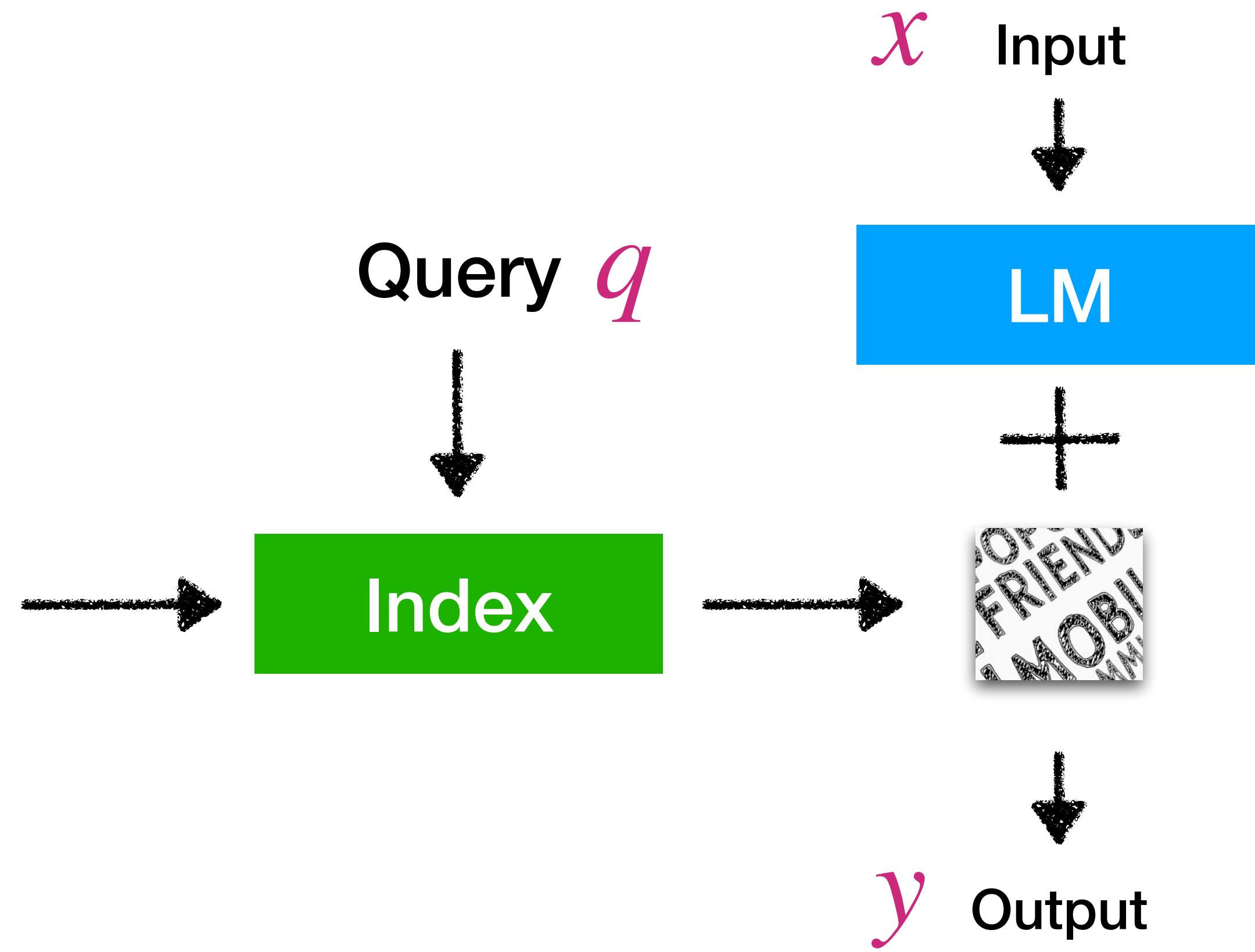
We'll answer these questions in Section 3!

Notations



Datastore

D



Section 3: Retrieval-based LM:Architecture

Categorization of retrieval-based LMs

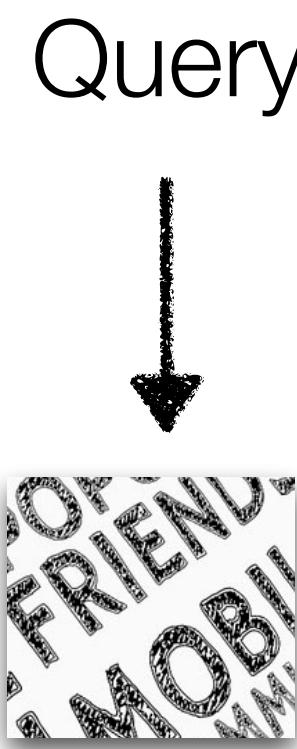
Categorization of retrieval-based LMs

What to retrieve?



Categorization of retrieval-based LMs

What to retrieve?



Text chunks (passages)?

Categorization of retrieval-based LMs

What to retrieve?



Text chunks (passages)?

Tokens?

Categorization of retrieval-based LMs

What to retrieve?



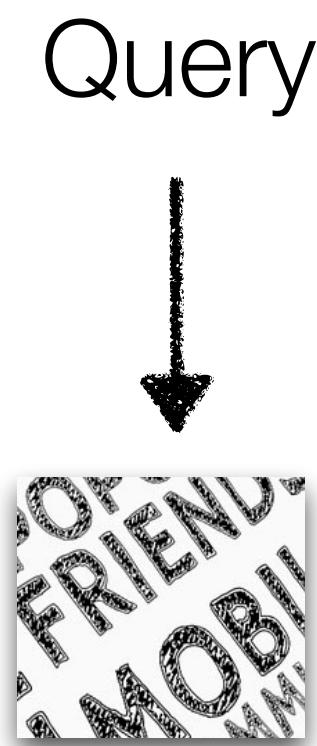
Text chunks (passages)?

Tokens?

Something else?

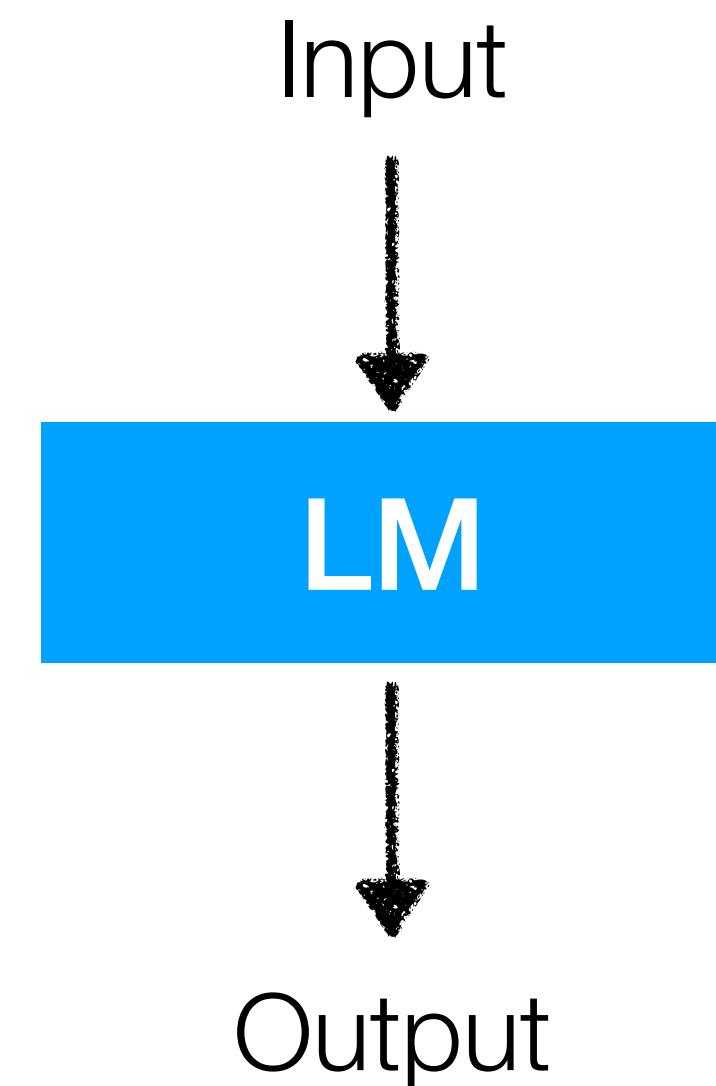
Categorization of retrieval-based LMs

What to retrieve?



Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



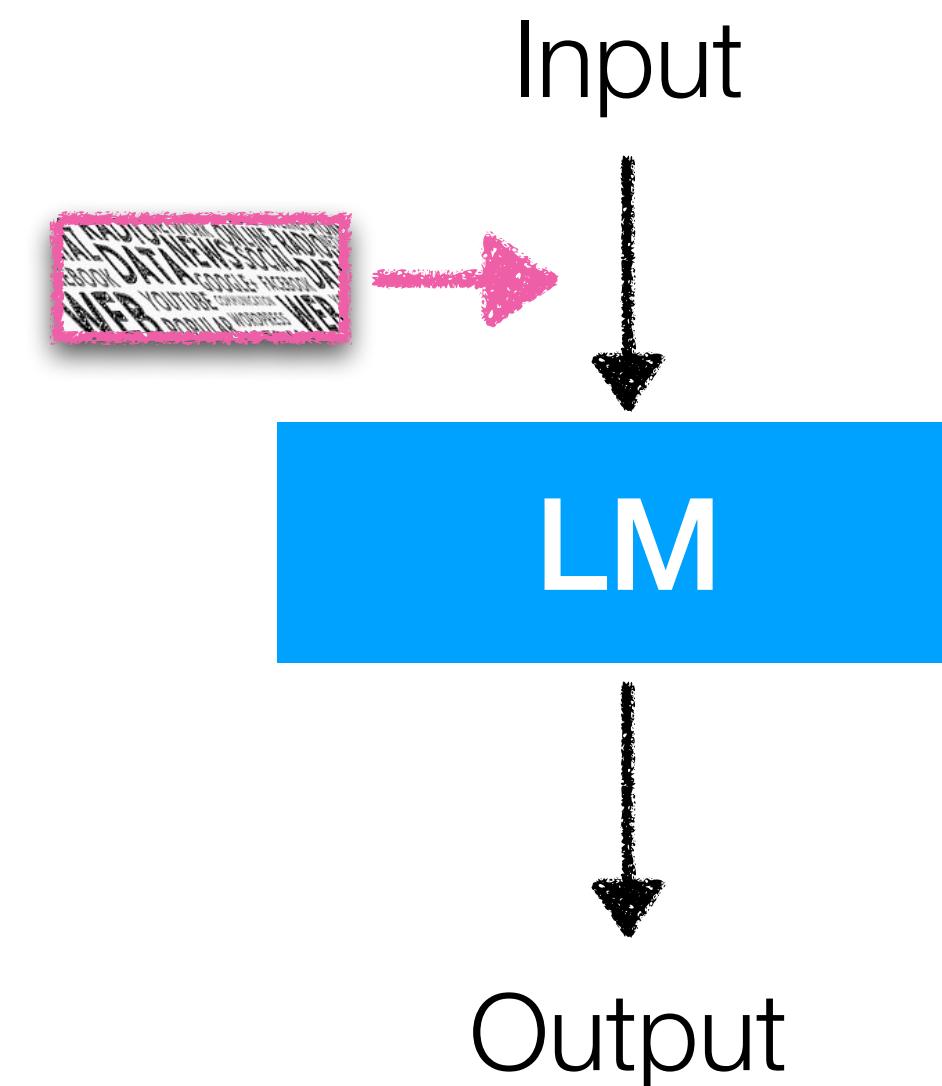
Categorization of retrieval-based LMs

What to retrieve?



- Text chunks (passages)?
- Tokens?
- Something else?

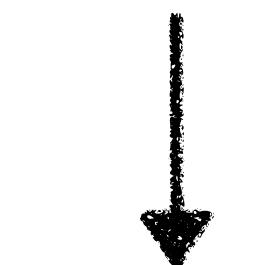
How to use retrieval?



Categorization of retrieval-based LMs

What to retrieve?

Query



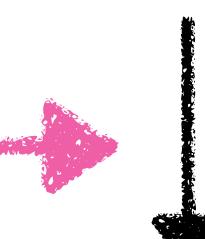
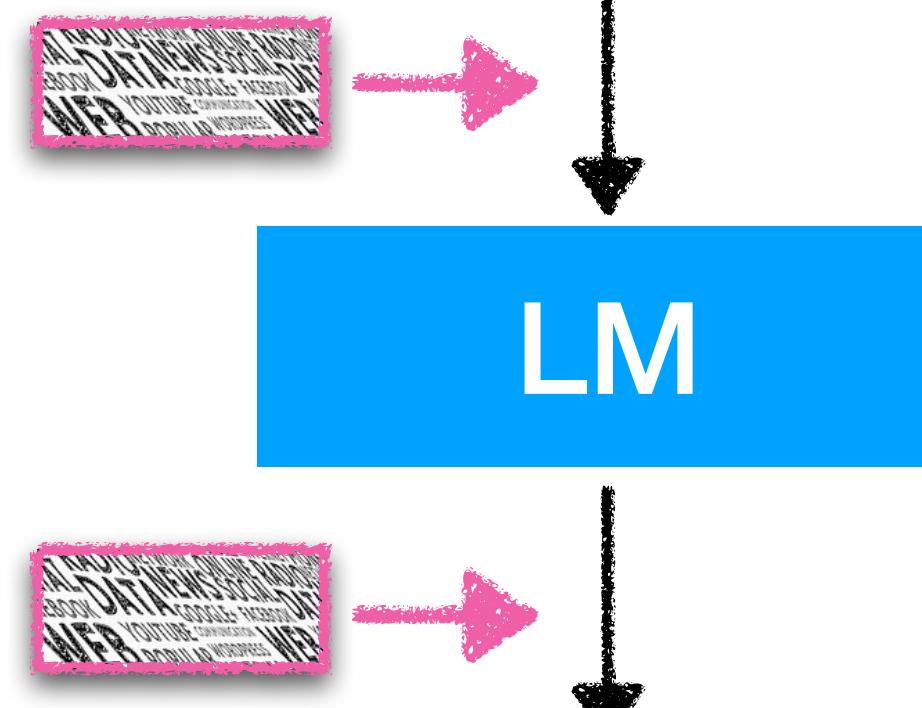
Text chunks (passages)?

Tokens?

Something else?

How to use retrieval?

Input



Output

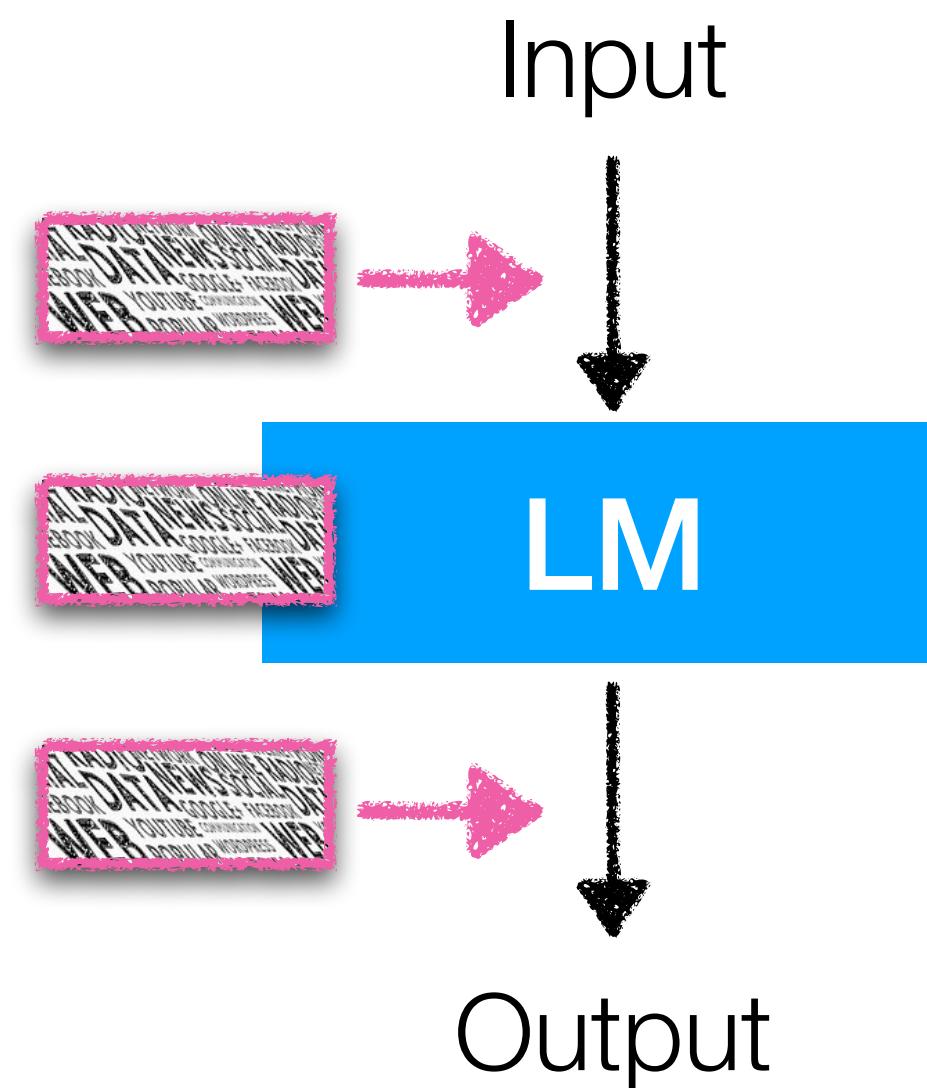
Categorization of retrieval-based LMs

What to retrieve?



Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



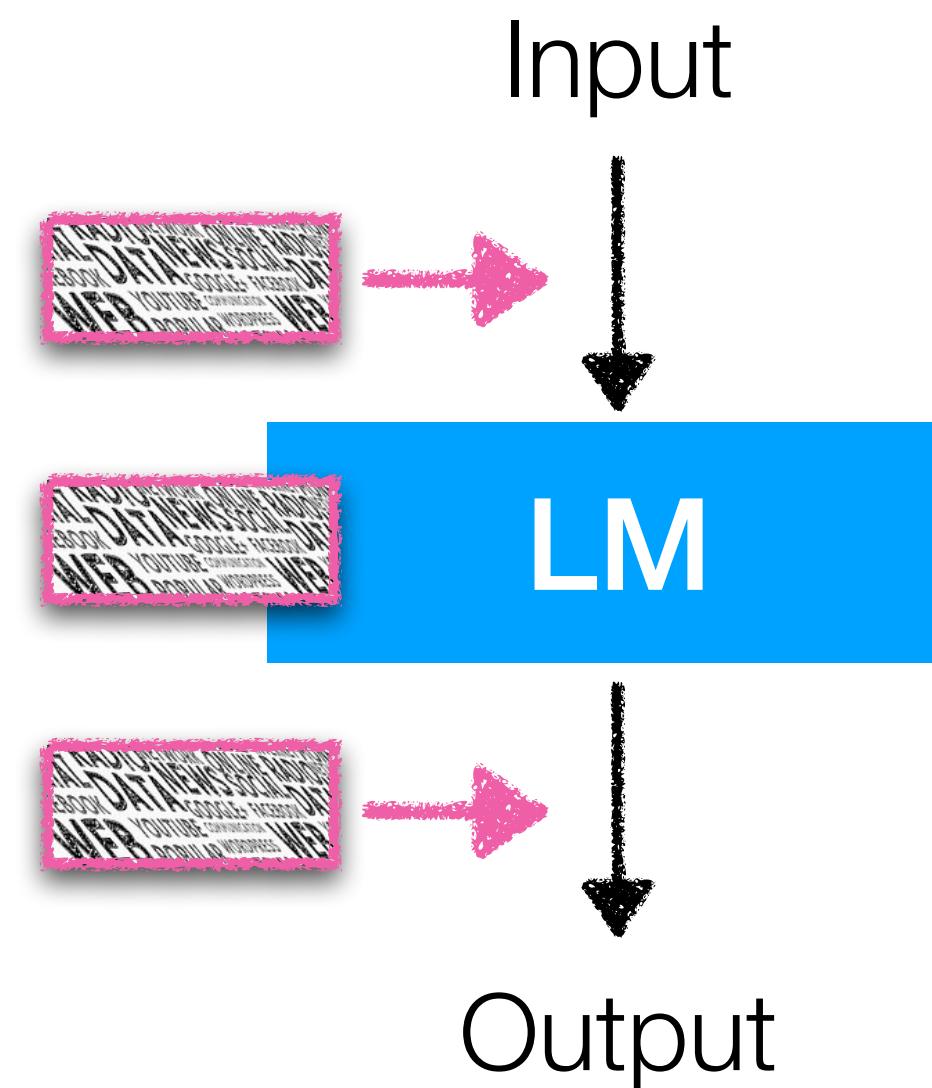
Categorization of retrieval-based LMs

What to retrieve?



Text chunks (passages)?
Tokens?
Something else?

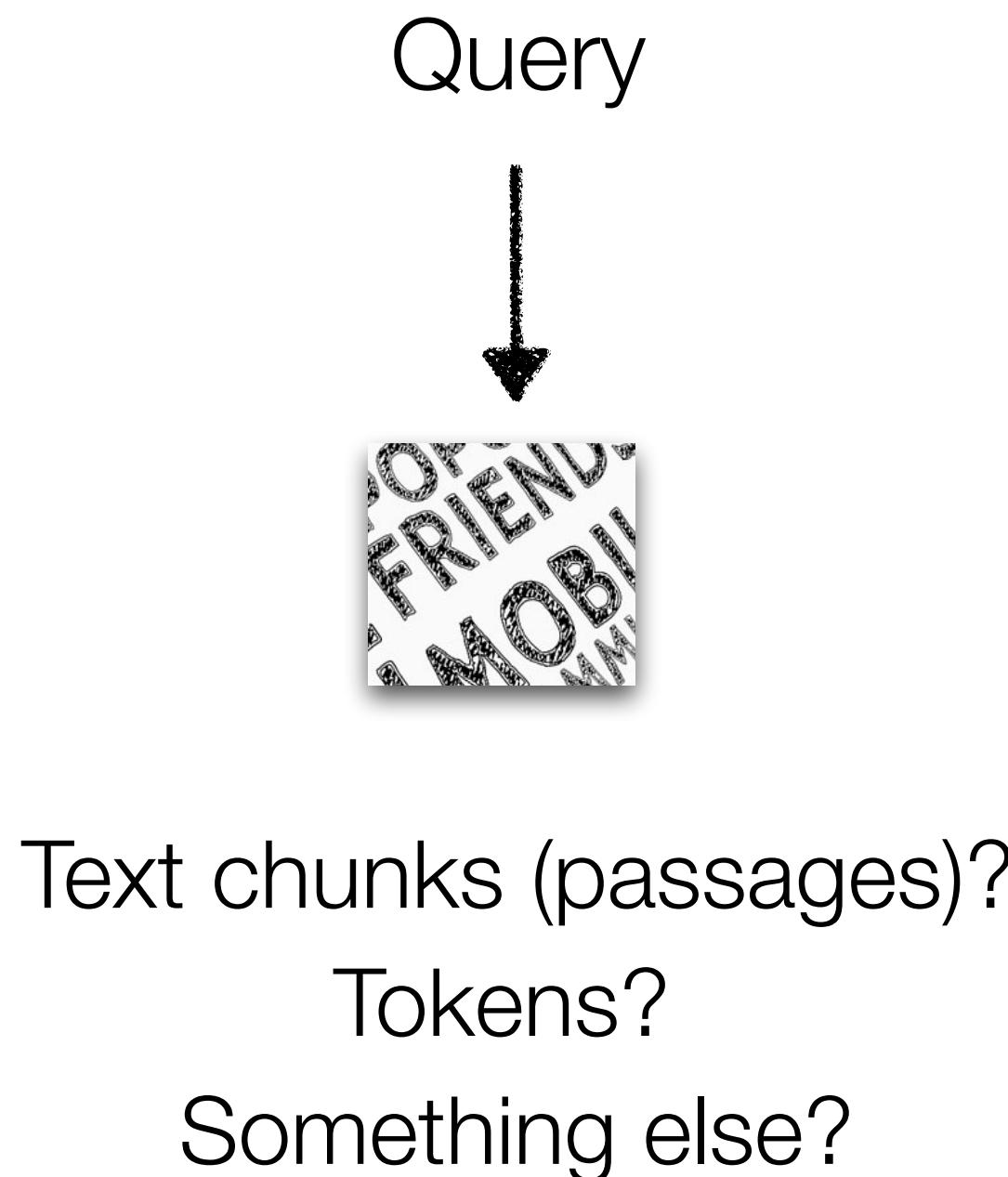
How to use retrieval?



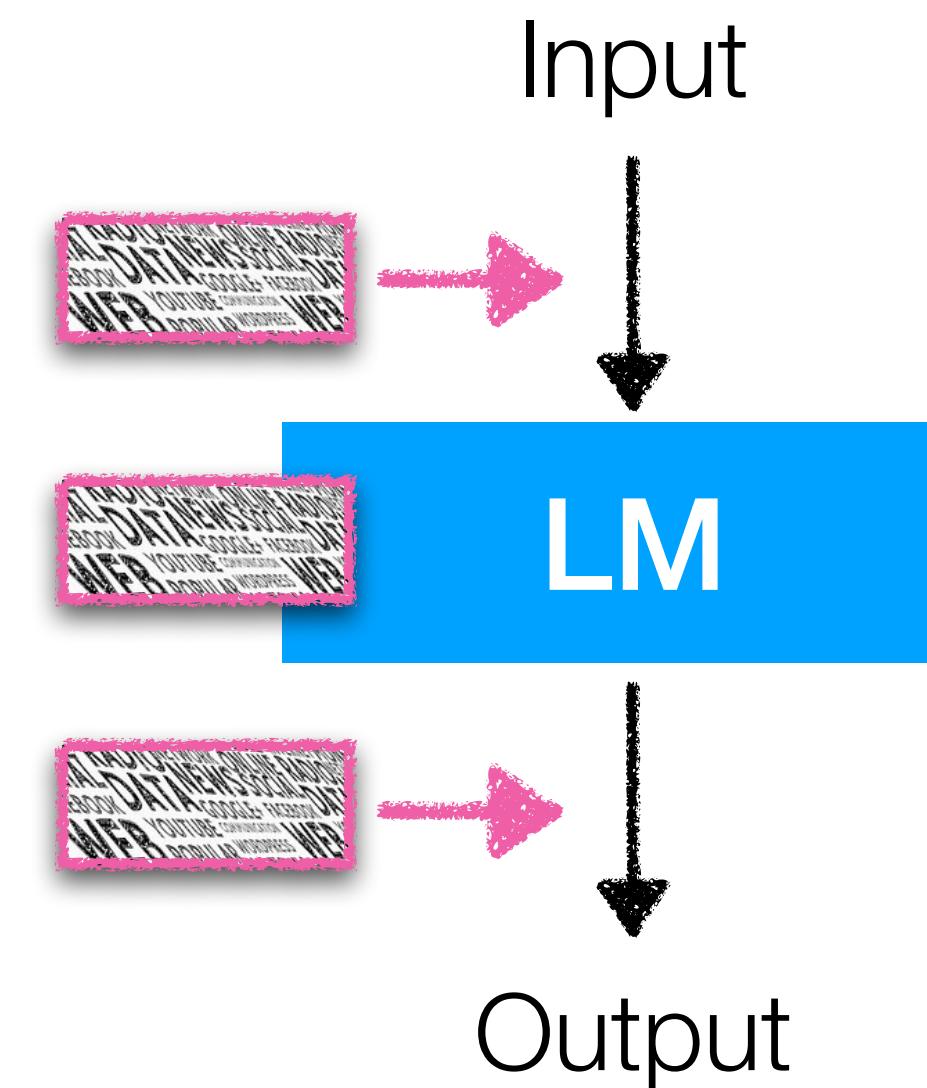
When to retrieve?

Categorization of retrieval-based LMs

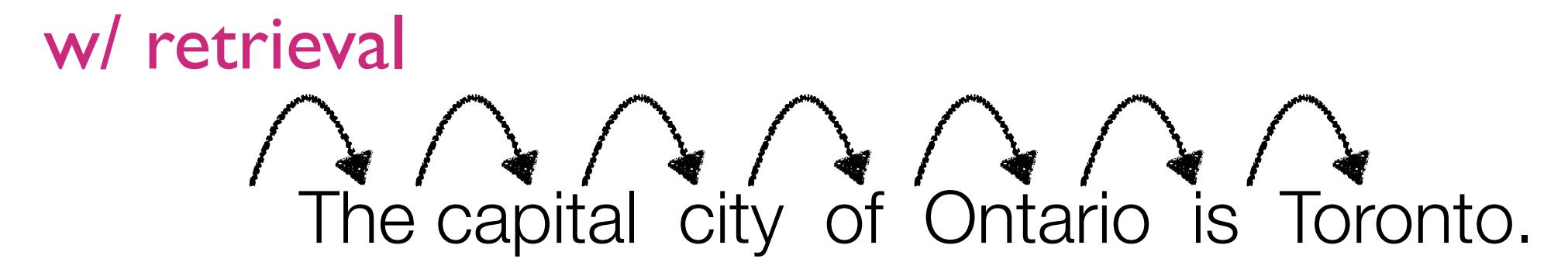
What to retrieve?



How to use retrieval?

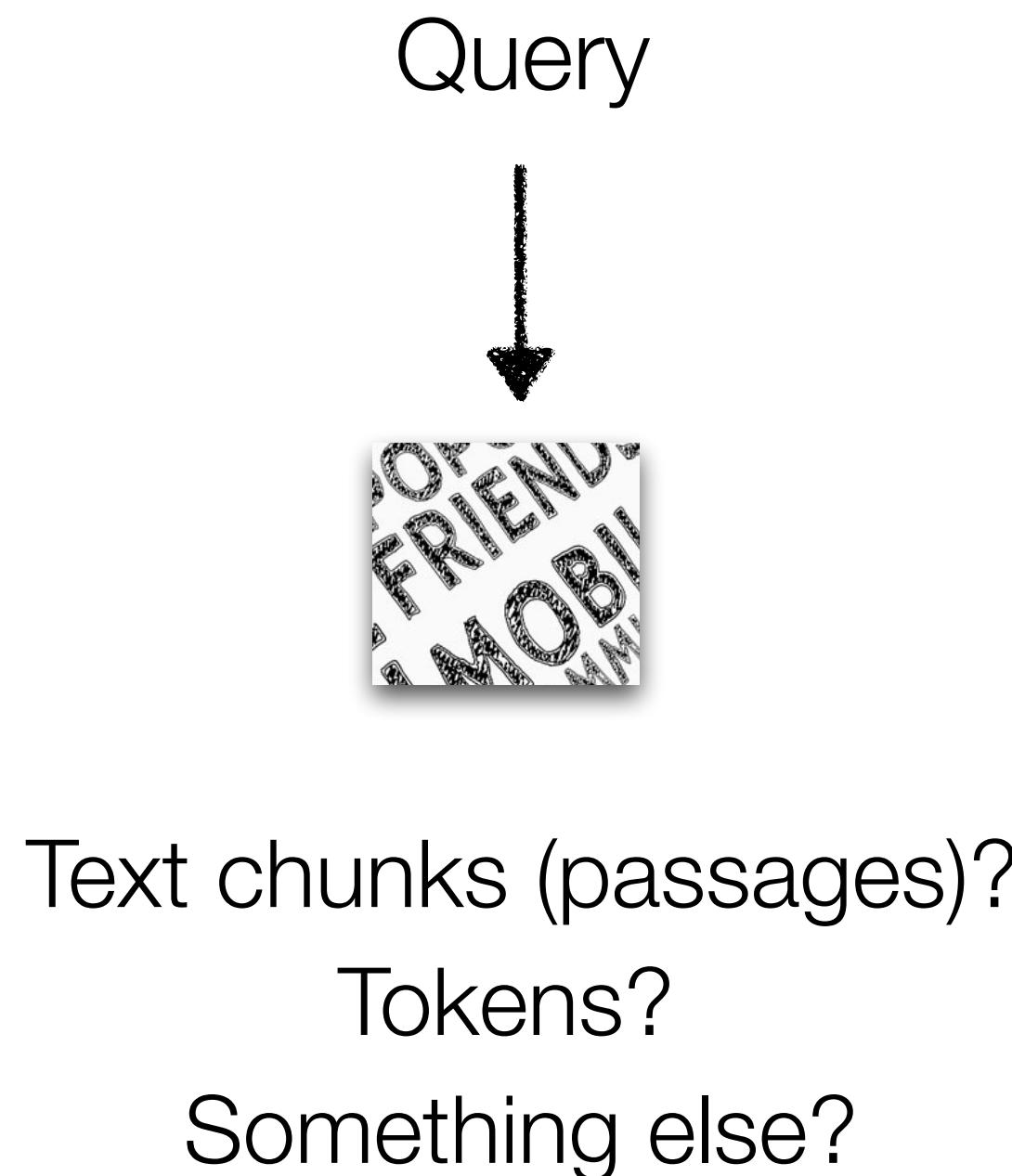


When to retrieve?

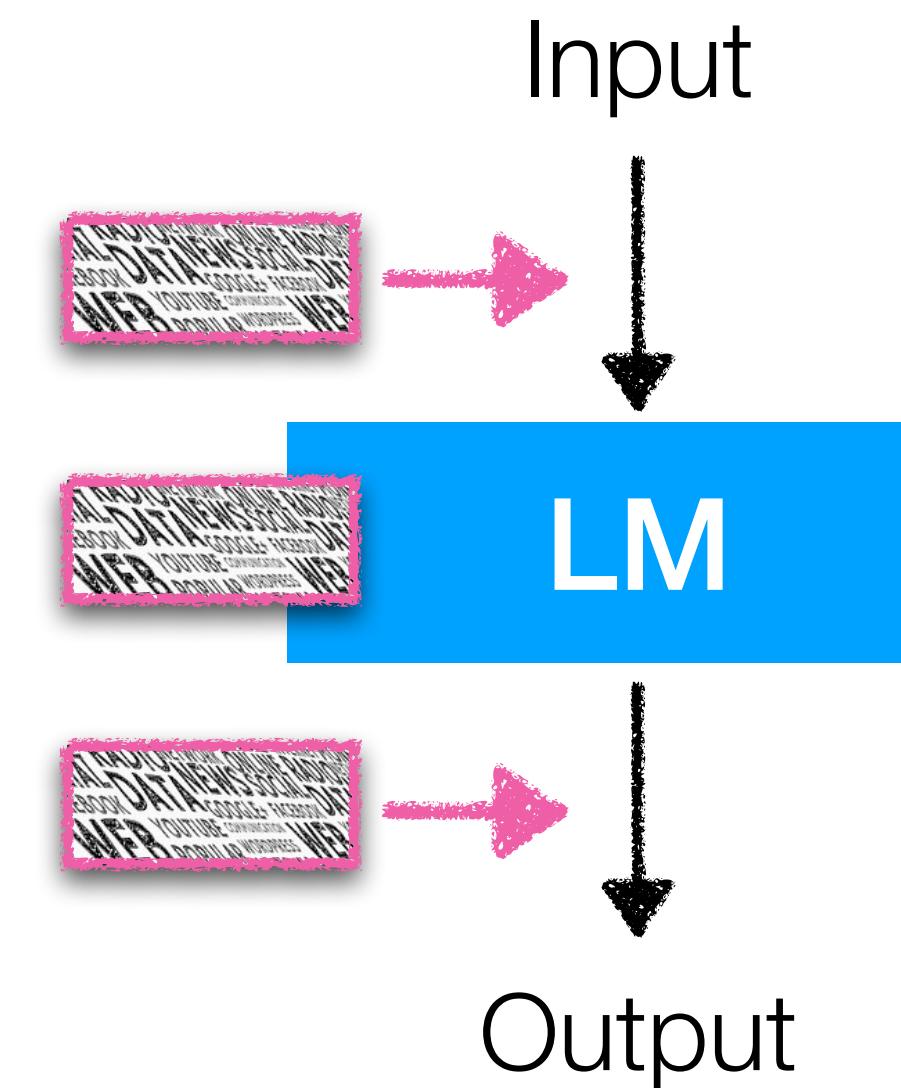


Categorization of retrieval-based LMs

What to retrieve?



How to use retrieval?

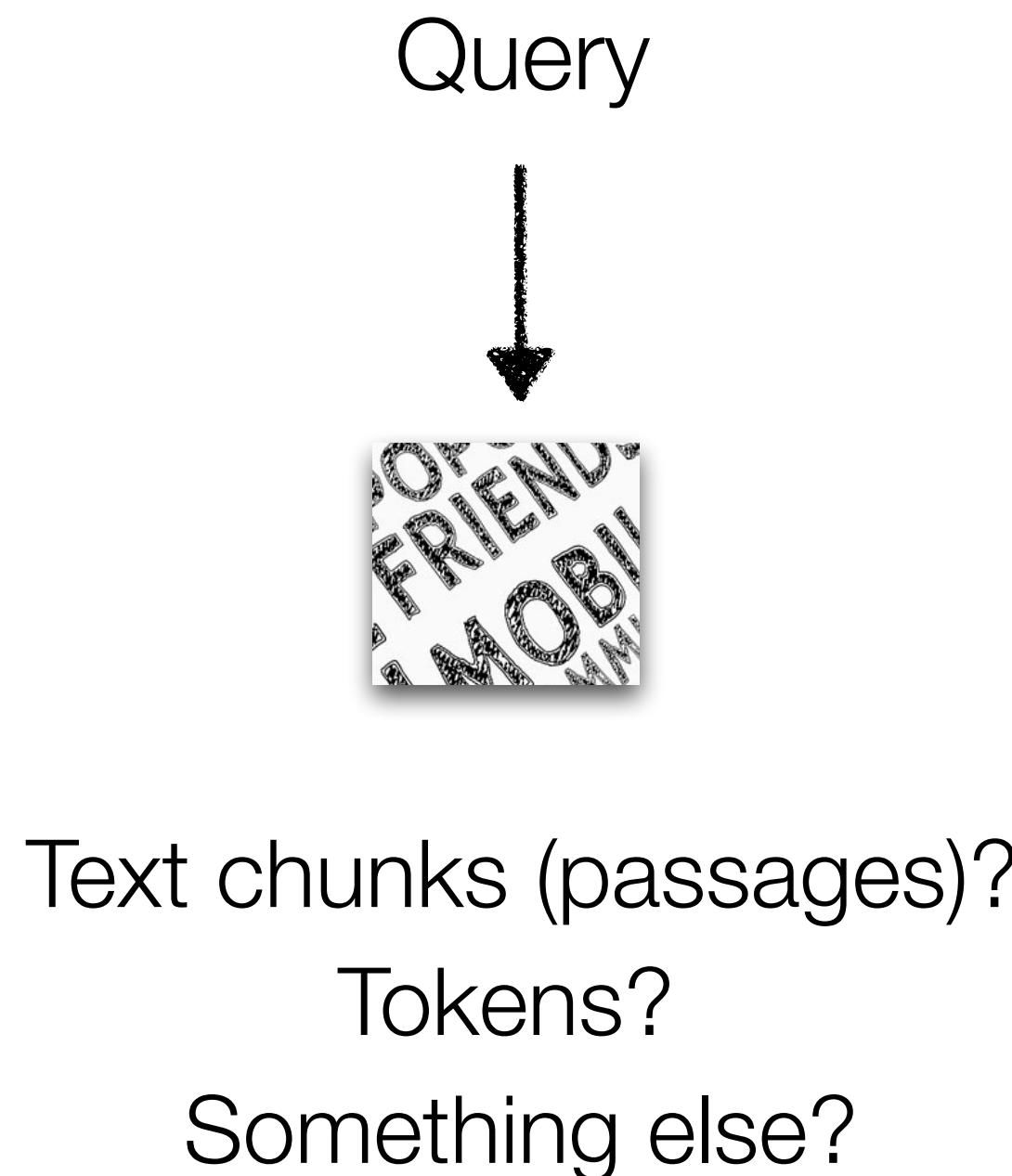


When to retrieve?

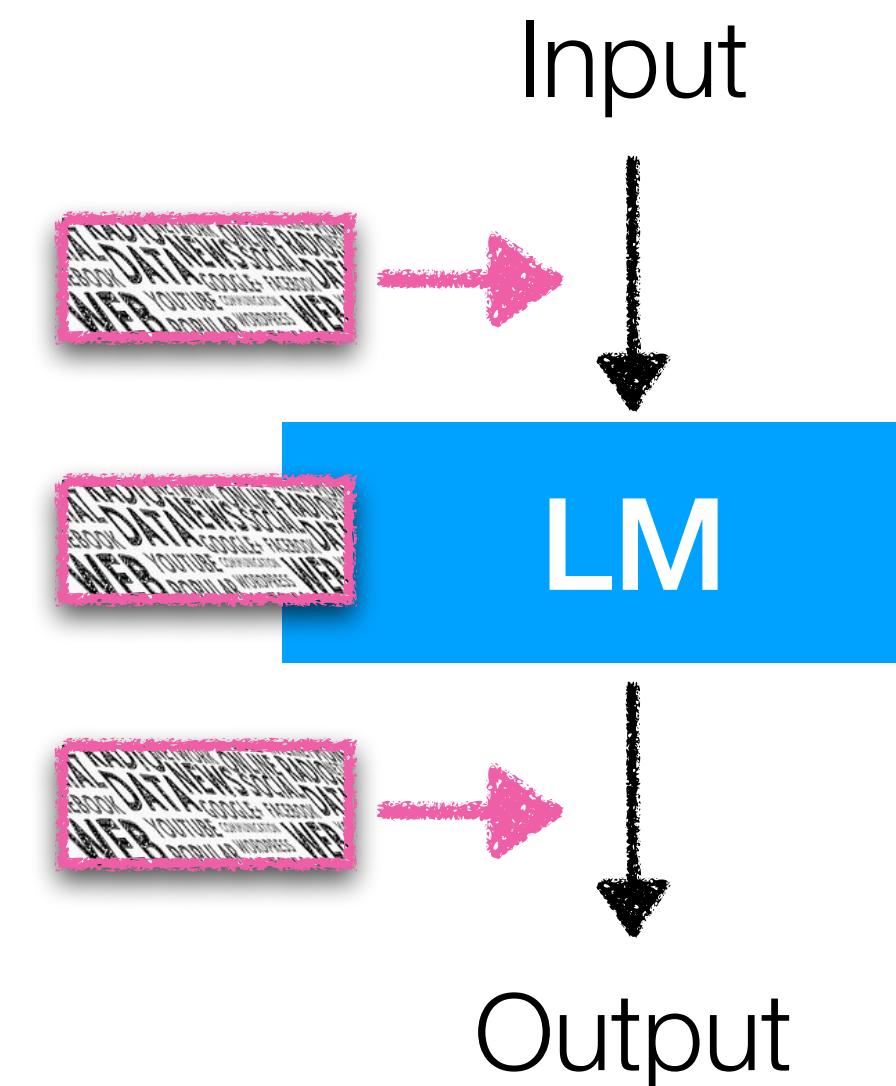


Categorization of retrieval-based LMs

What to retrieve?



How to use retrieval?



When to retrieve?



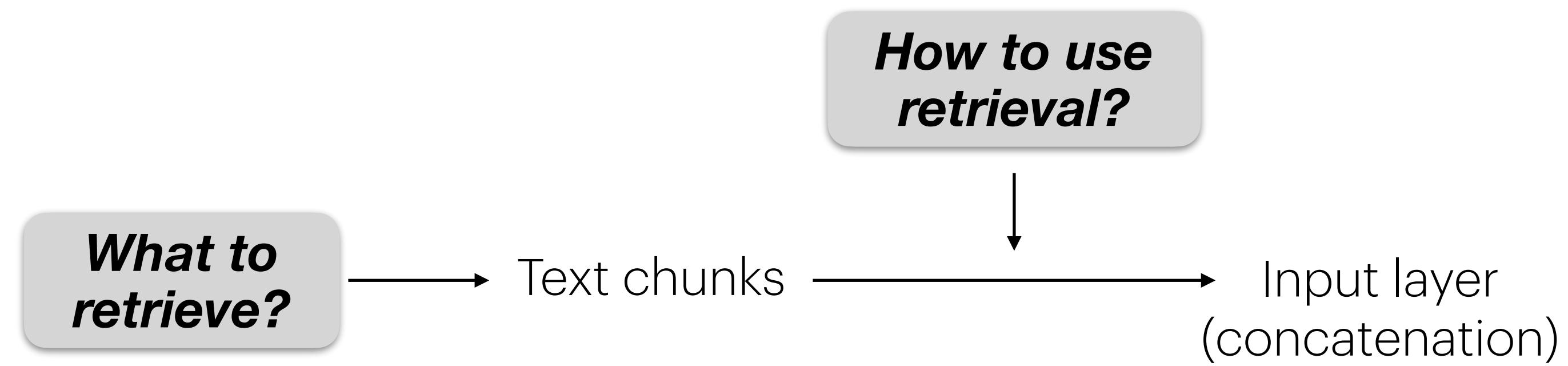
Roadmap

Roadmap

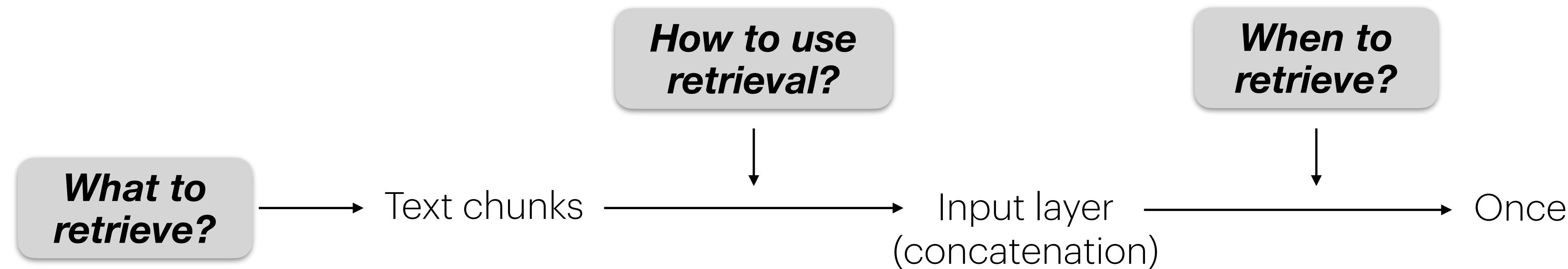
**What to
retrieve?**

→ Text chunks

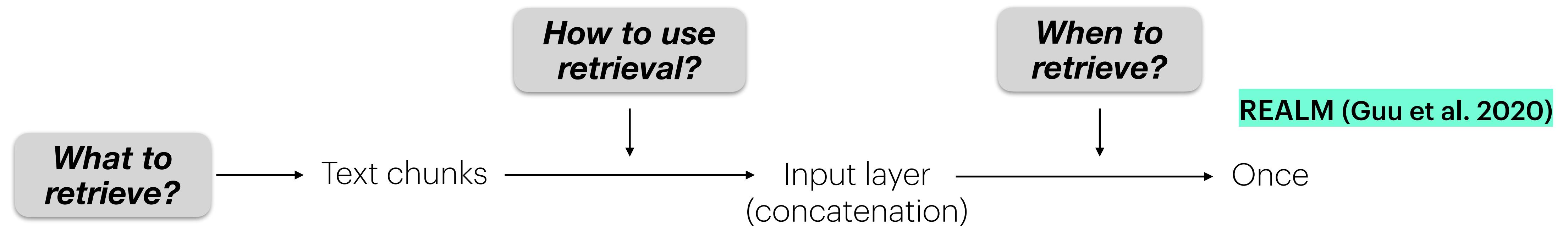
Roadmap



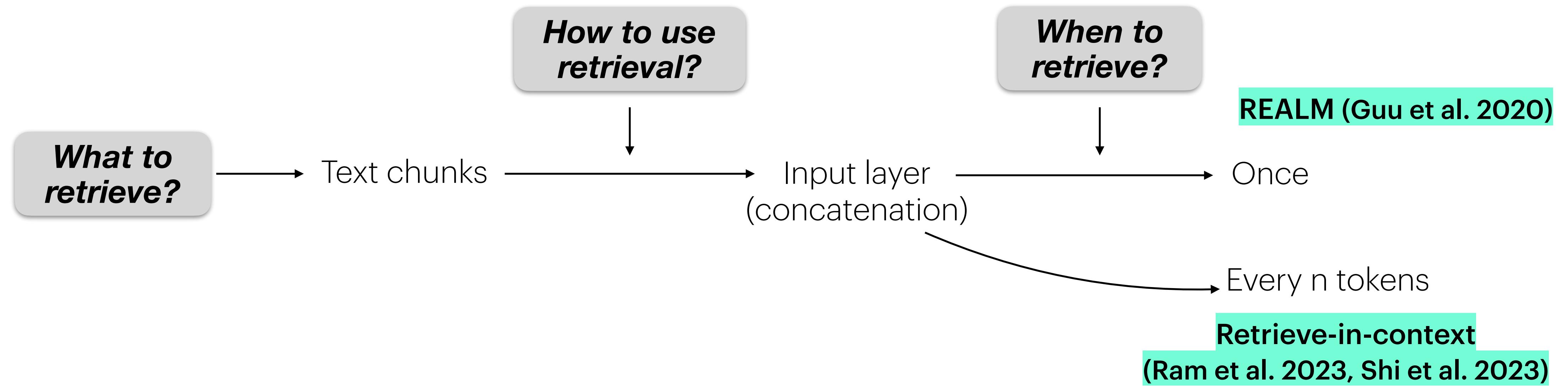
Roadmap



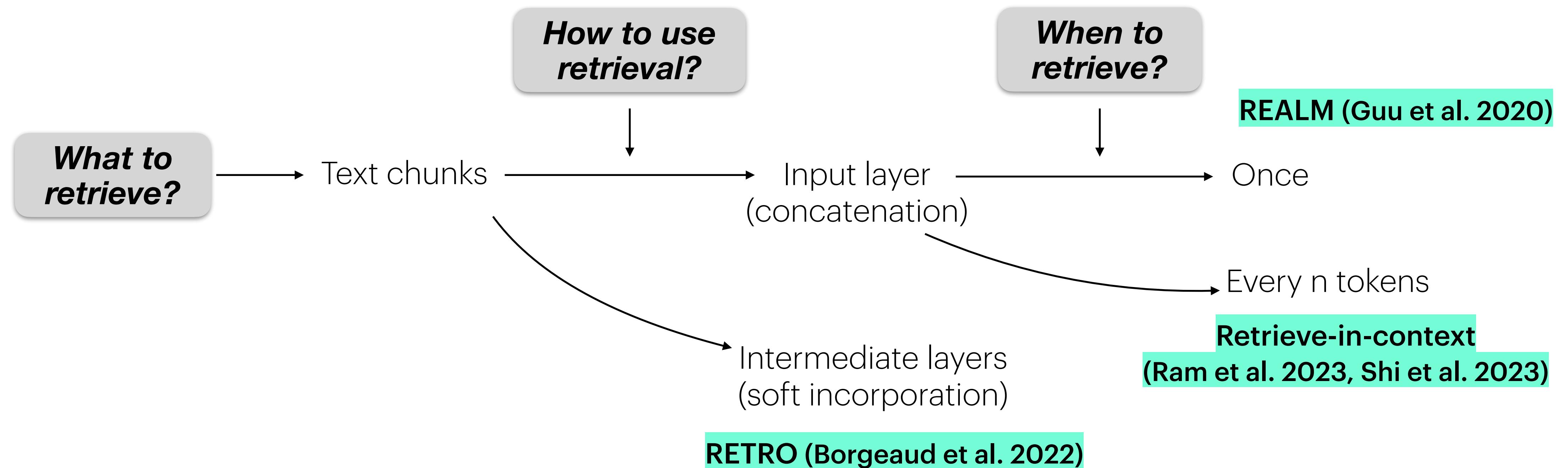
Roadmap



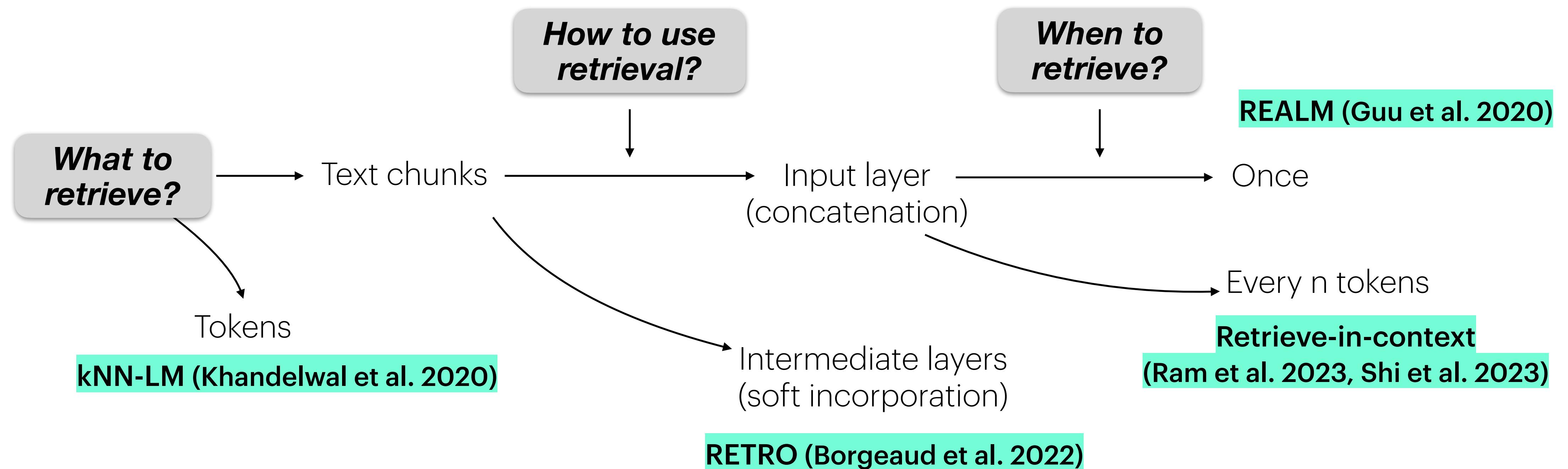
Roadmap



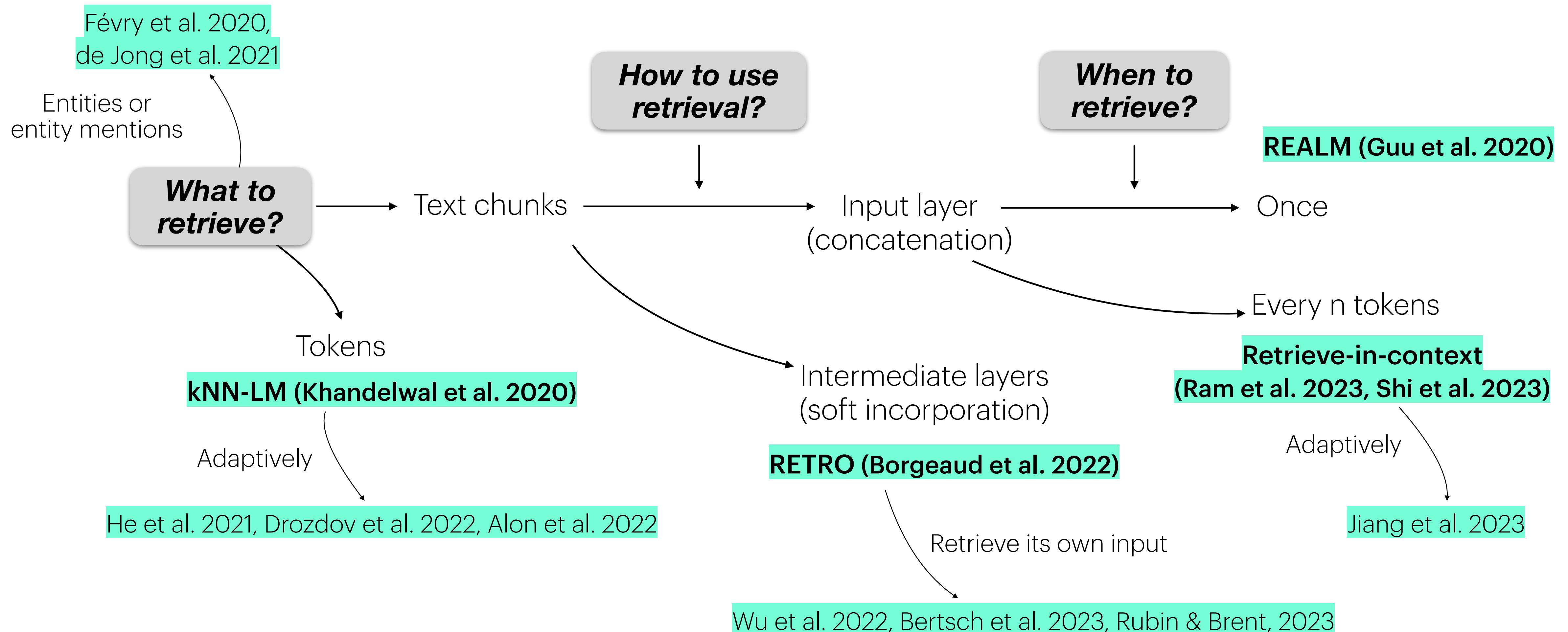
Roadmap



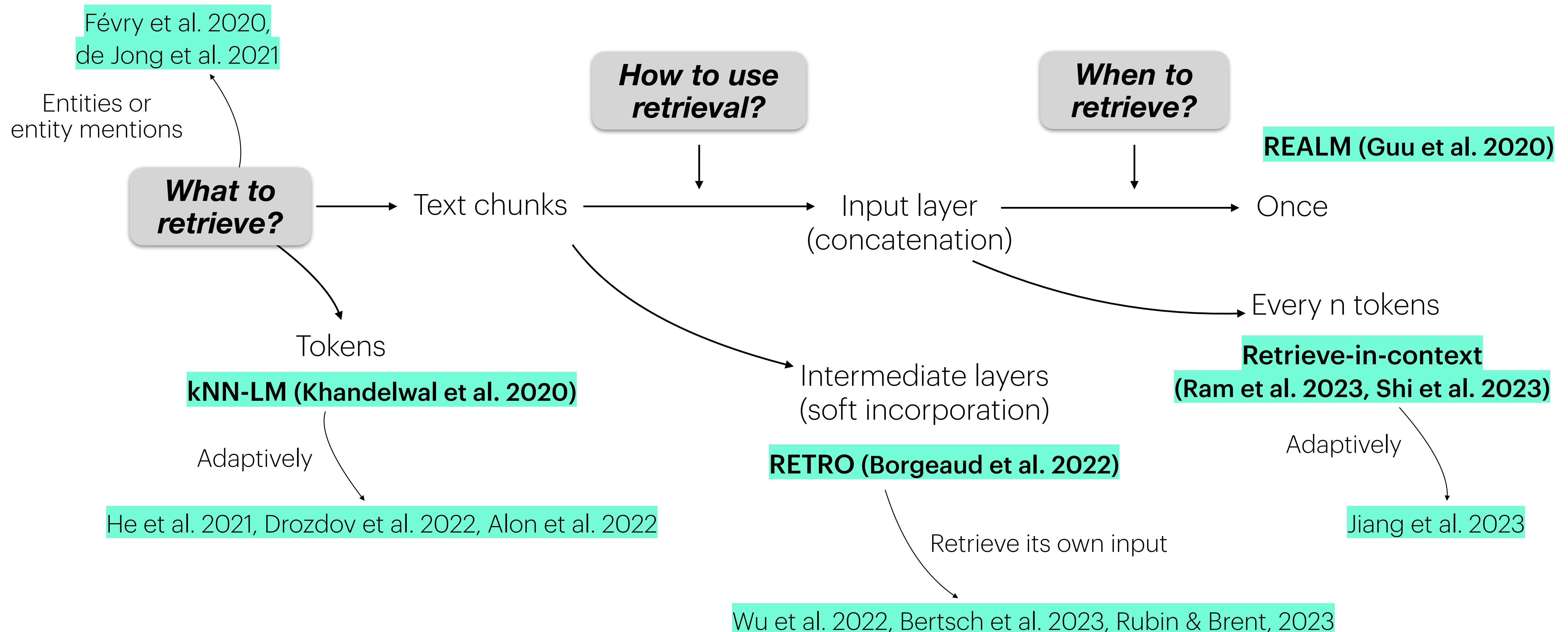
Roadmap



Roadmap



Roadmap



This is only about “architecture”
Section 4 will categorize & discuss “training”

REALM (Guu et al 2020)

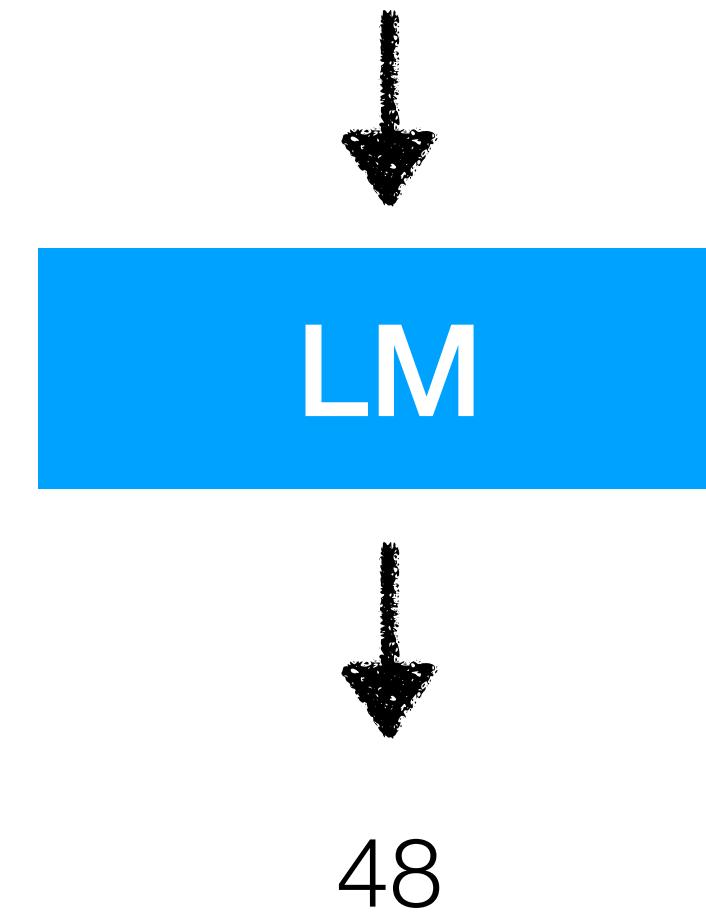
REALM (Guu et al 2020)

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

REALM (Guu et al 2020)

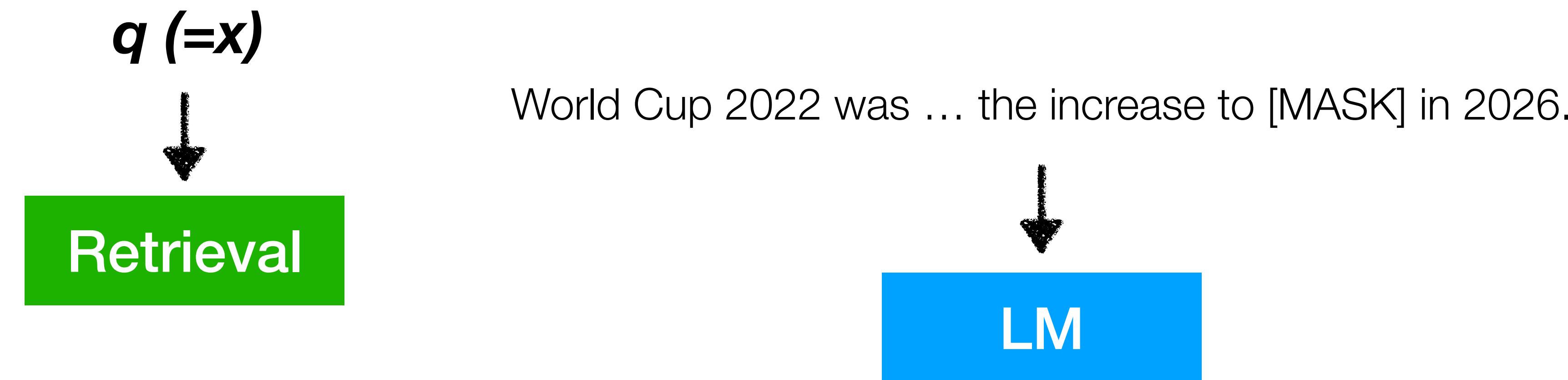
x = World Cup 2022 was the last with 32 teams before the increase to [MASK] in 2026.

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



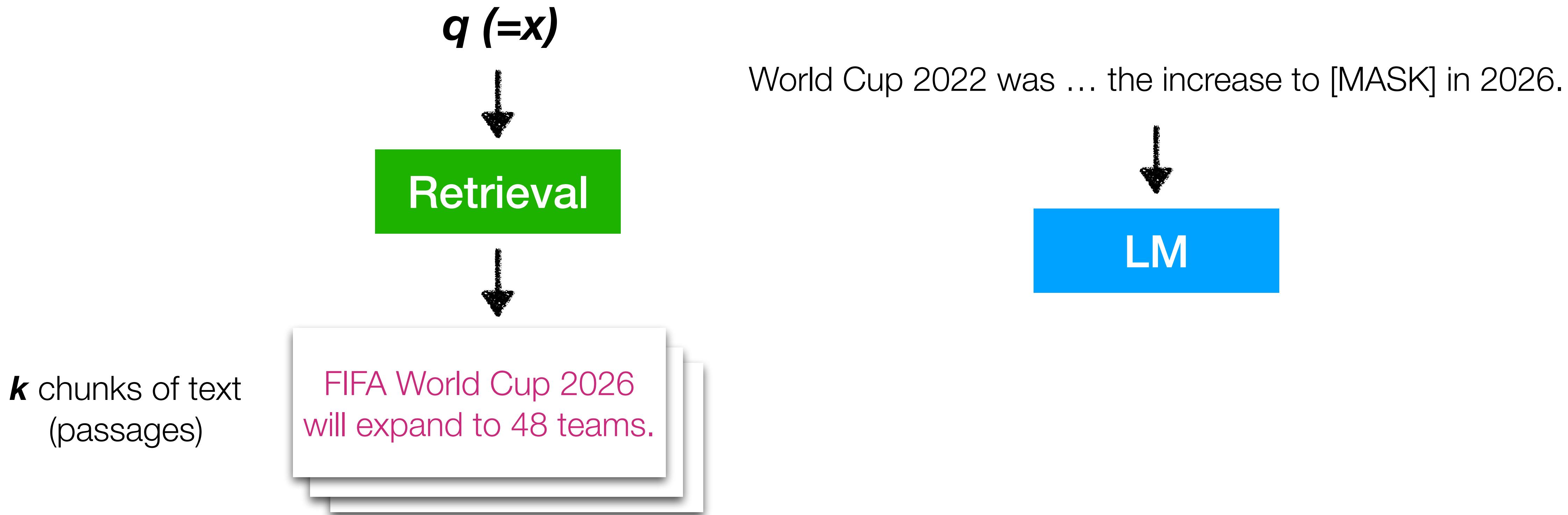
REALM (Guu et al 2020)

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



REALM (Guu et al 2020)

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



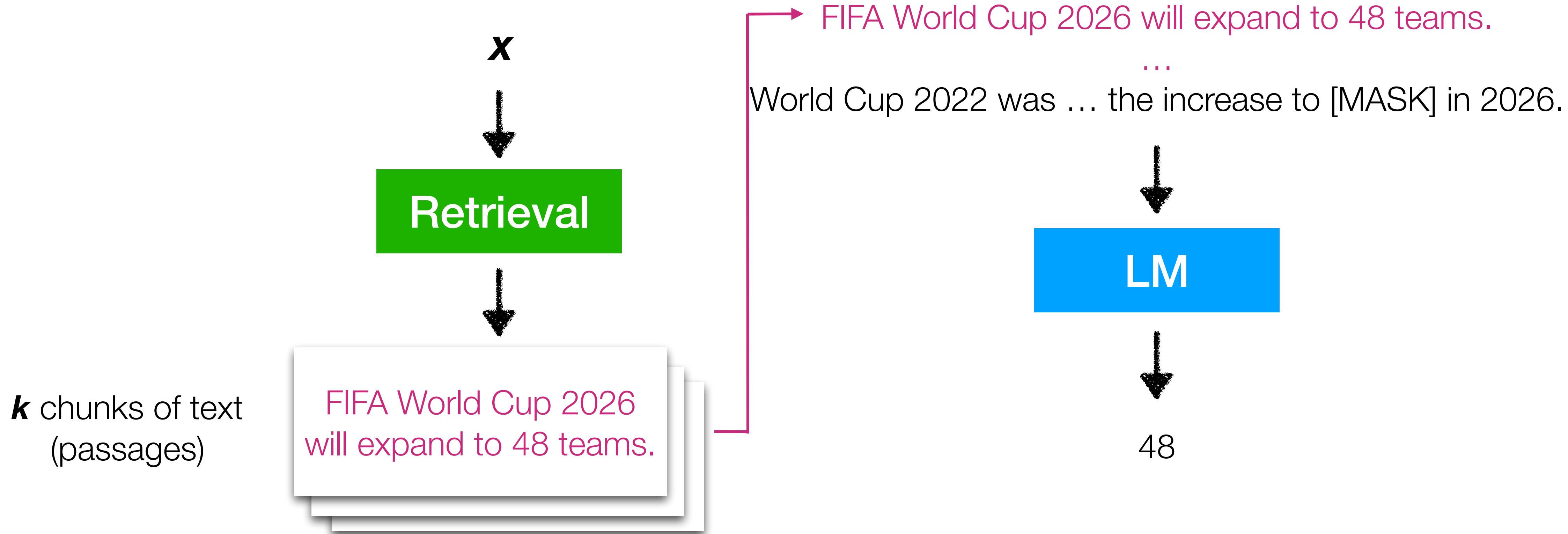
REALM (Guu et al 2020)

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



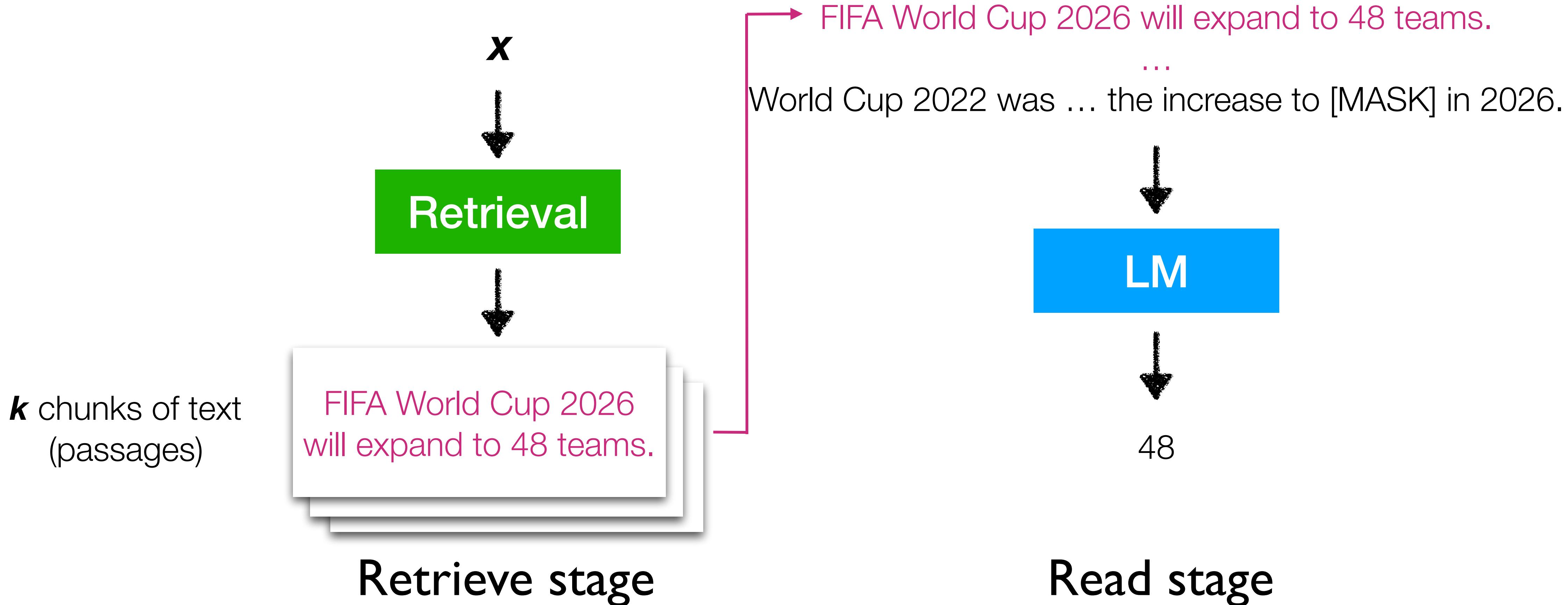
REALM (Guu et al 2020)

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



REALM (Guu et al 2020)

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



REALM: (I) Retrieve stage

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 ...

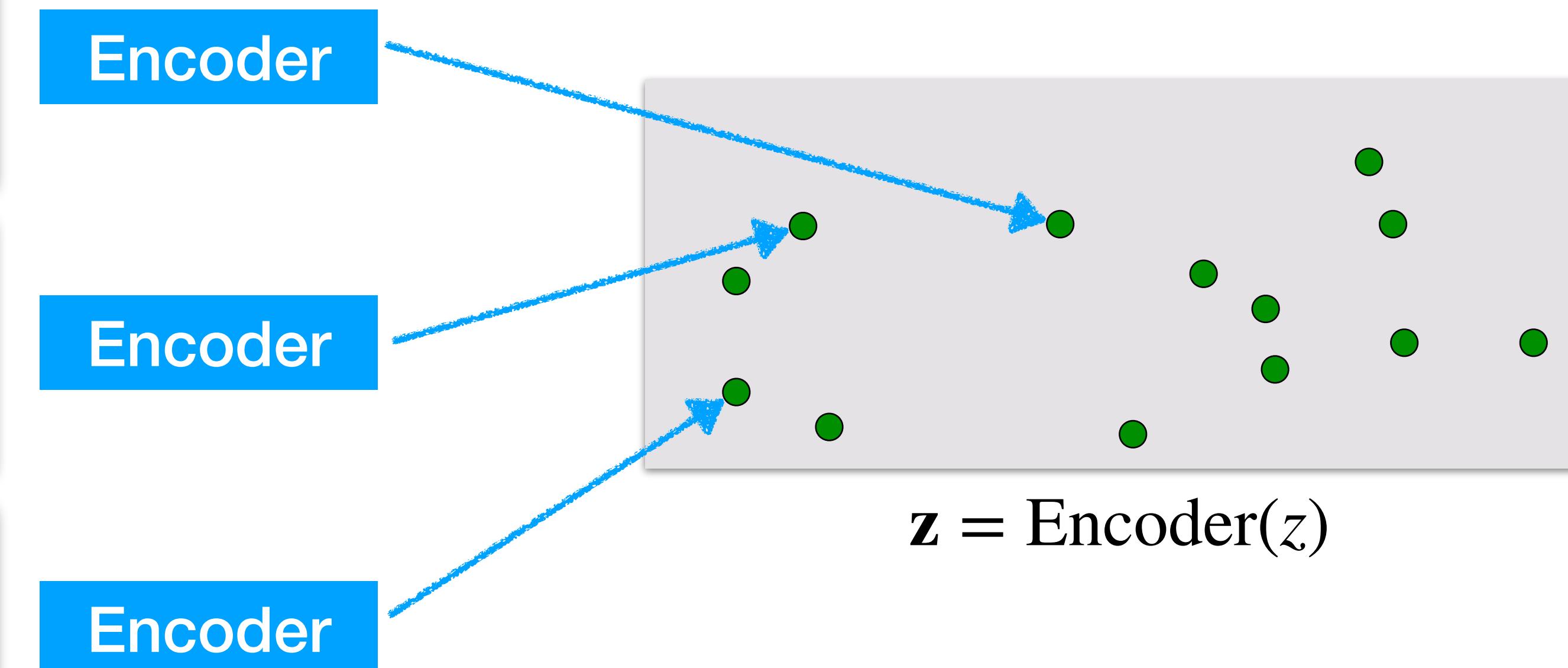
Wikipedia
13M chunks (passages)
(called *documents* in the paper)

REALM: (I) Retrieve stage

FIFA World Cup 2026 will expand to 48 teams.

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

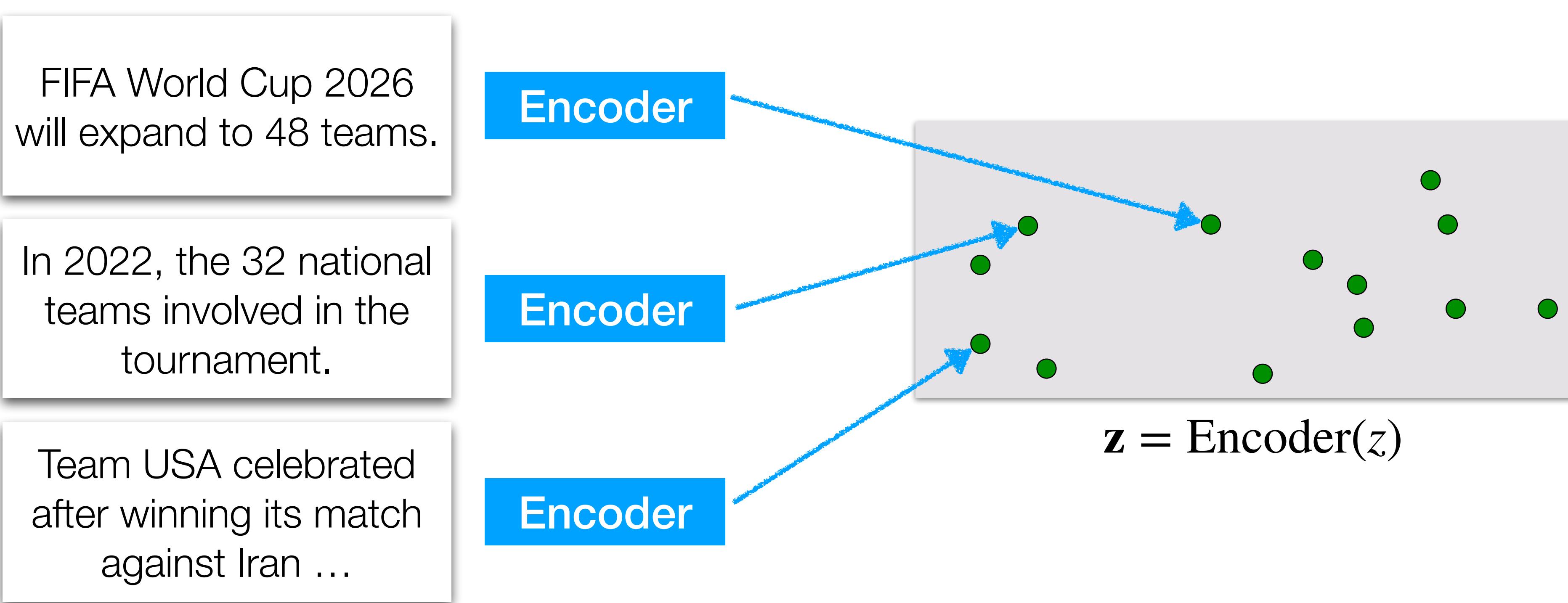
Team USA celebrated after winning its match against Iran ...



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

REALM: (I) Retrieve stage

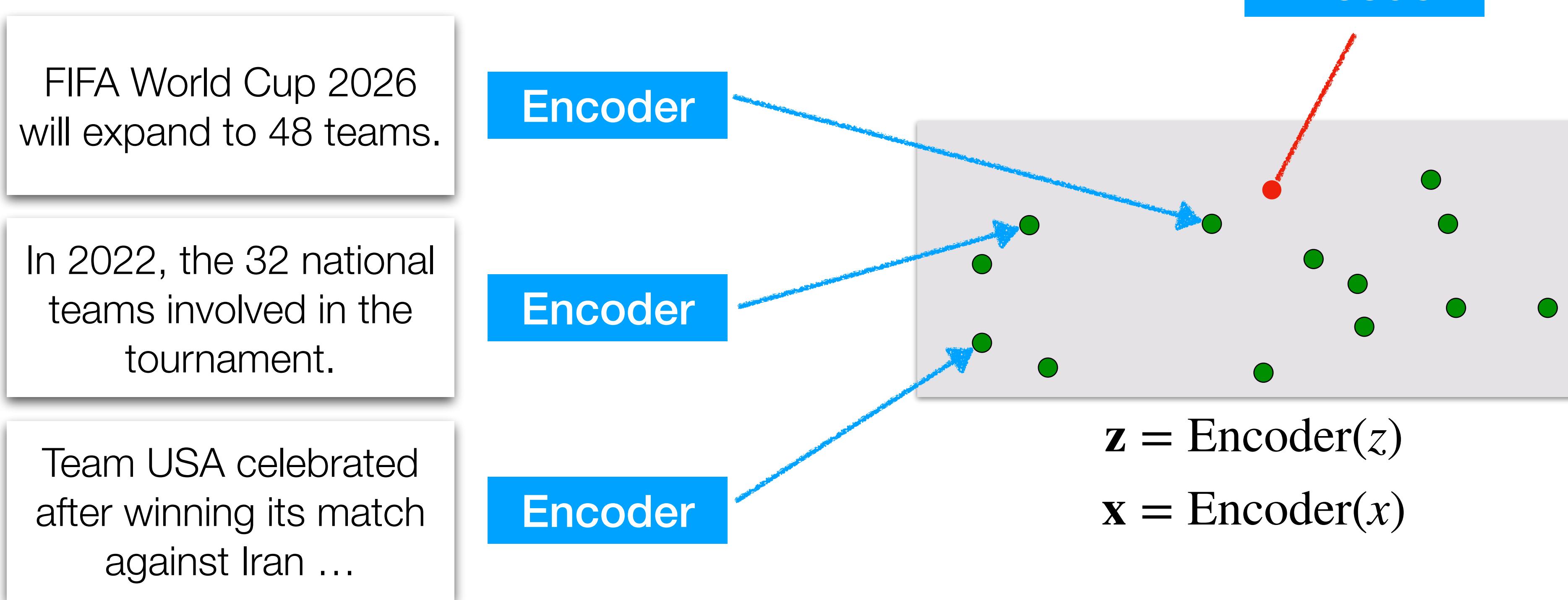
x = World Cup 2022 was ... the increase to [MASK] in 2026.



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

REALM: (I) Retrieve stage

\mathbf{x} = World Cup 2022 was ... the increase to [MASK] in 2026.

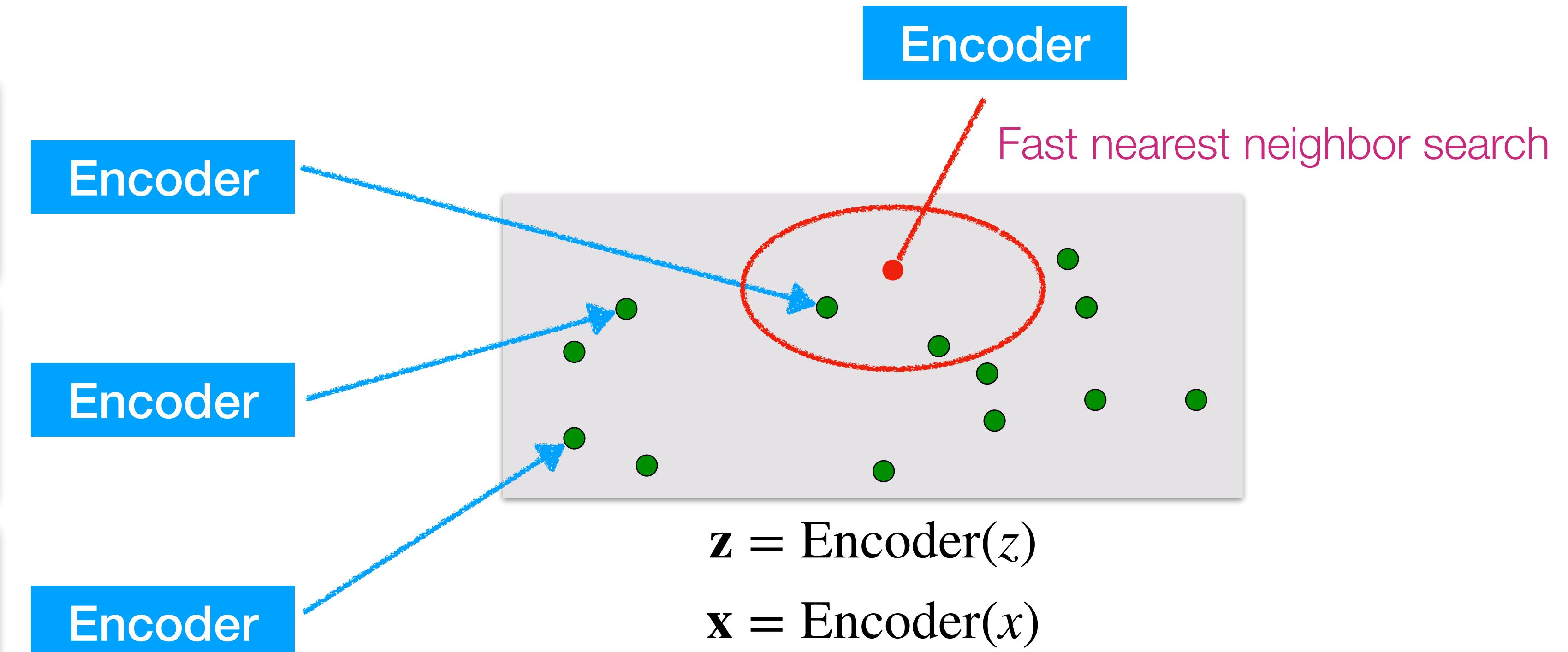


Wikipedia
13M chunks (passages)
(called *documents* in the paper)

REALM: (I) Retrieve stage

\mathbf{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)

REALM: (I) Retrieve stage

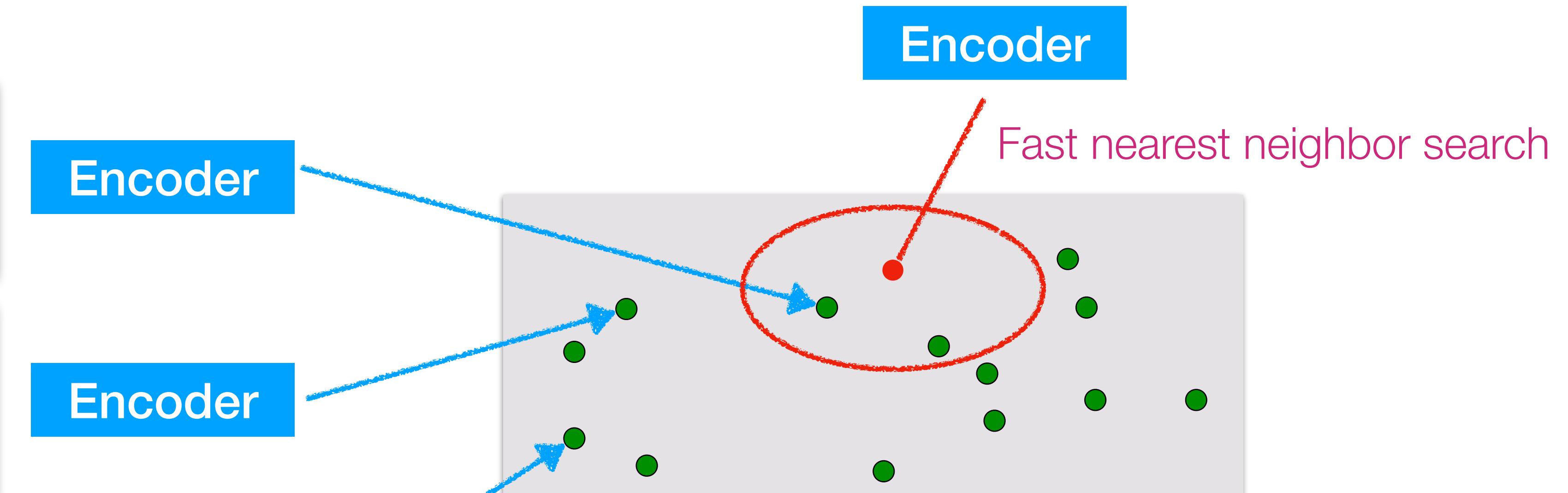
\mathbf{x} = World Cup 2022 was ... the increase to [MASK] in 2026.

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)



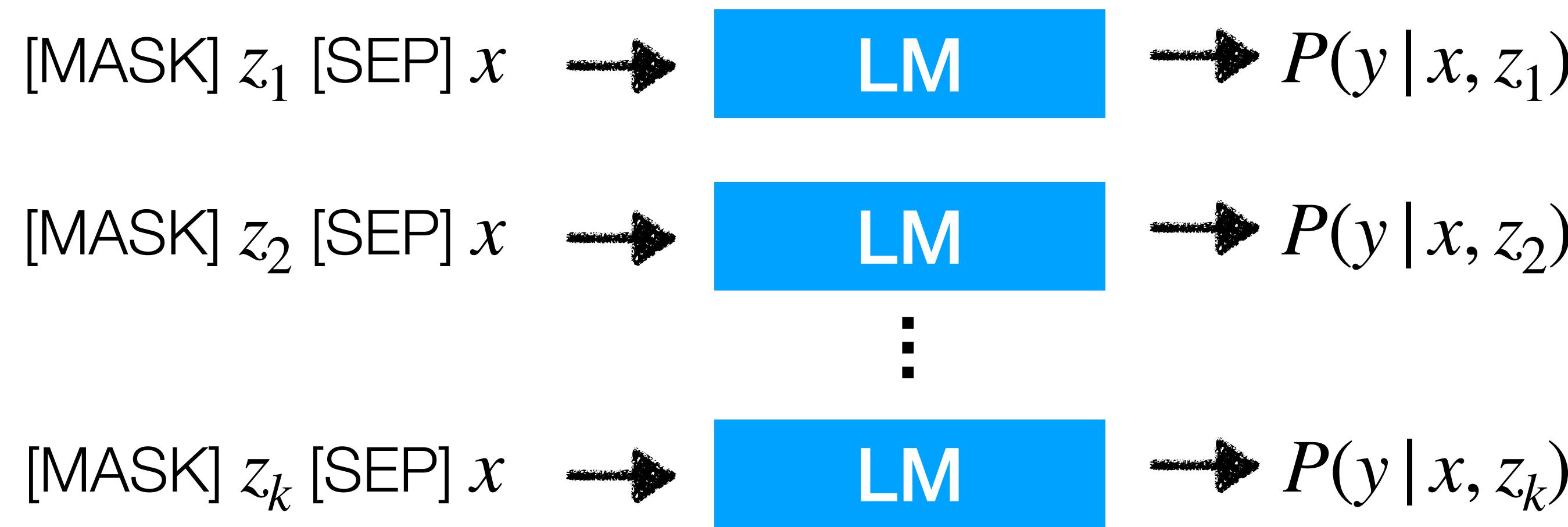
$$\mathbf{z} = \text{Encoder}(\mathbf{z})$$

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

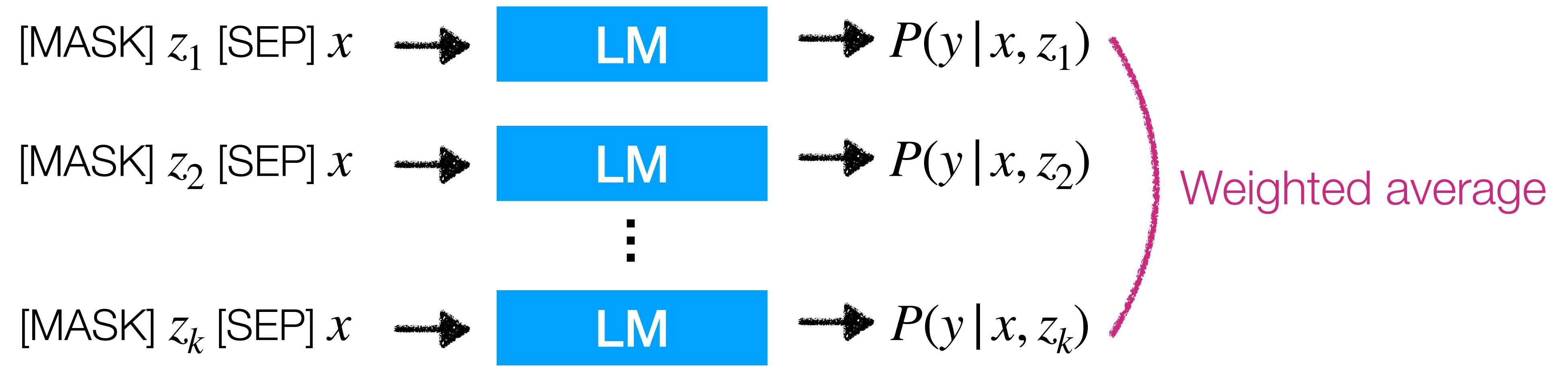
$$z_1, \dots, z_k = \text{argTop-}k(\mathbf{x} \cdot \mathbf{z})$$

k retrieved chunks

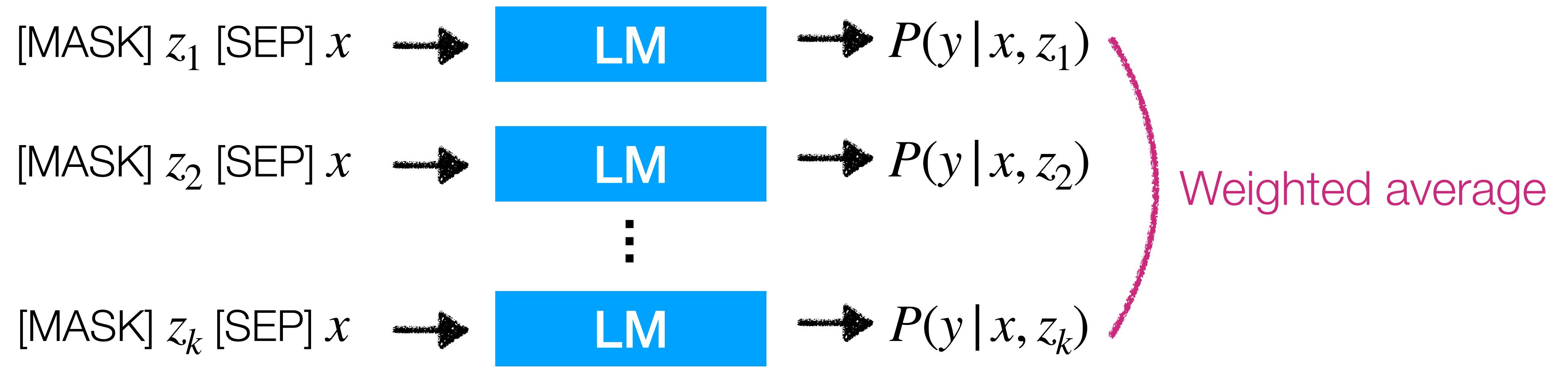
REALM: (2) Read stage



REALM: (2) Read stage

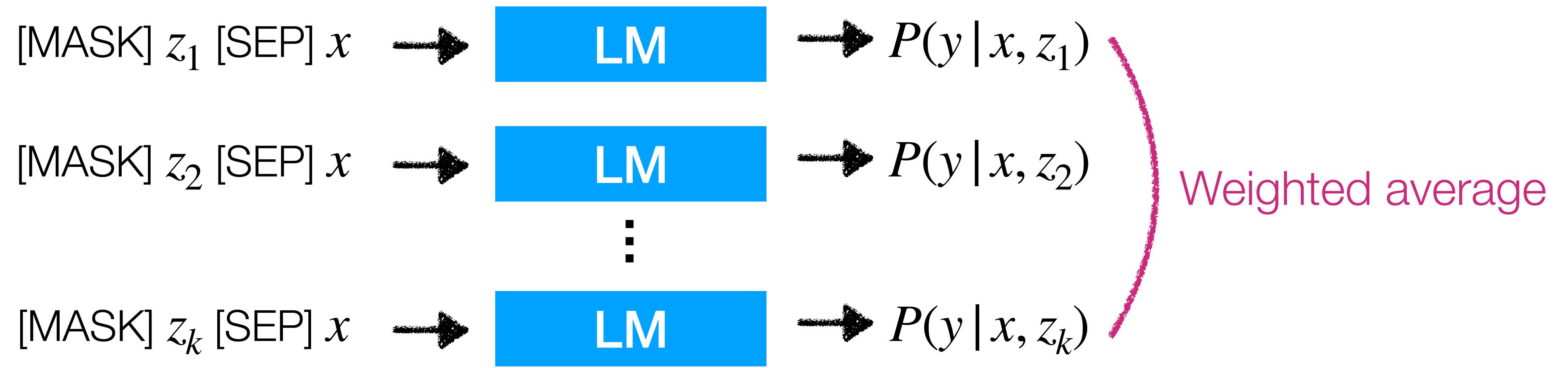


REALM: (2) Read stage



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

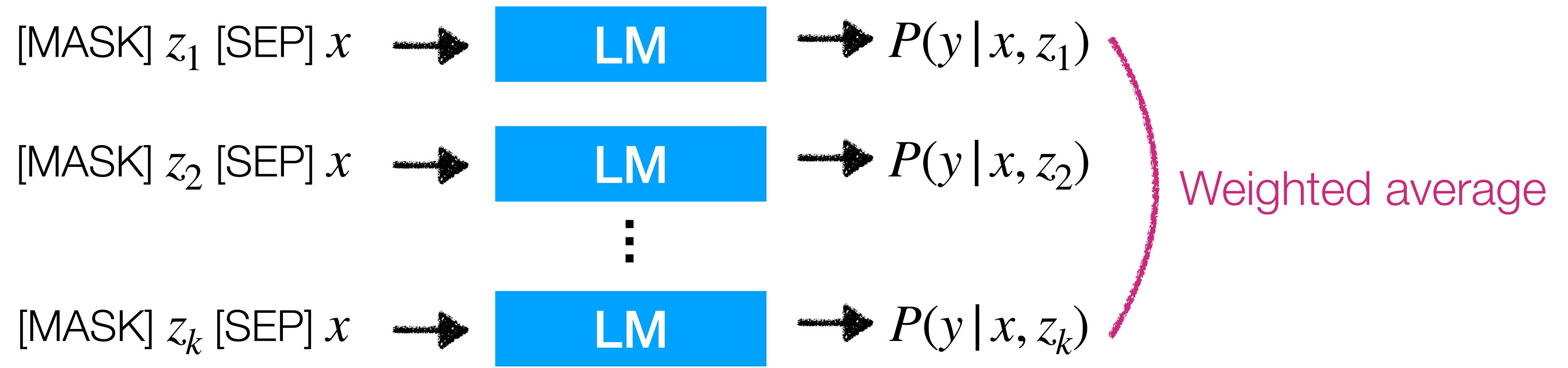
REALM: (2) Read stage



$$\sum_{z \in \mathcal{D}} \underline{P(z | x)} \underbrace{P(y | x, z)}$$

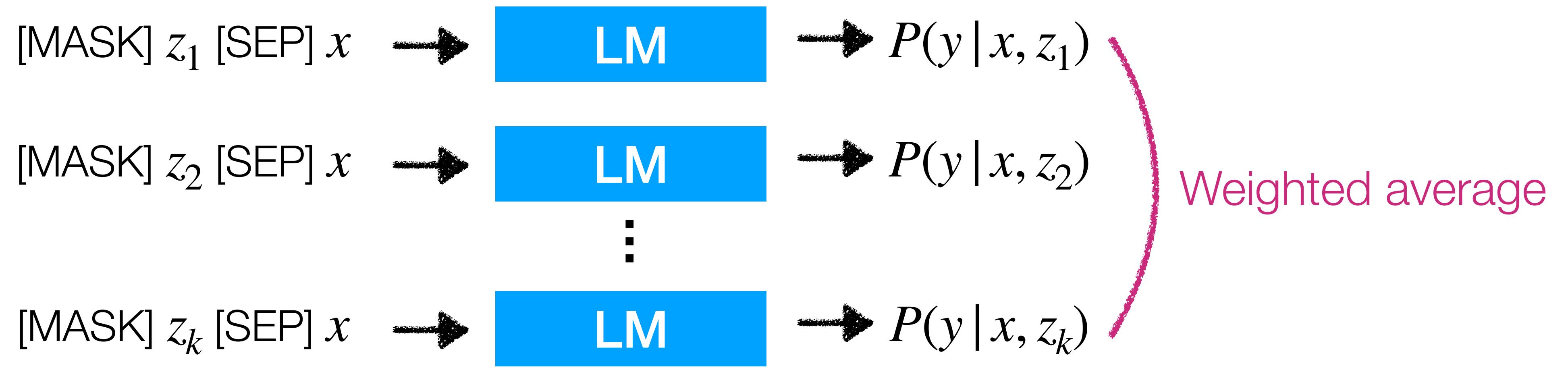
from the
retrieve stage

REALM: (2) Read stage



$$\sum_{z \in \mathcal{D}} \underbrace{P(z | x)}_{\text{from the retrieve stage}} \underbrace{P(y | x, z)}_{\text{from the read stage}}$$

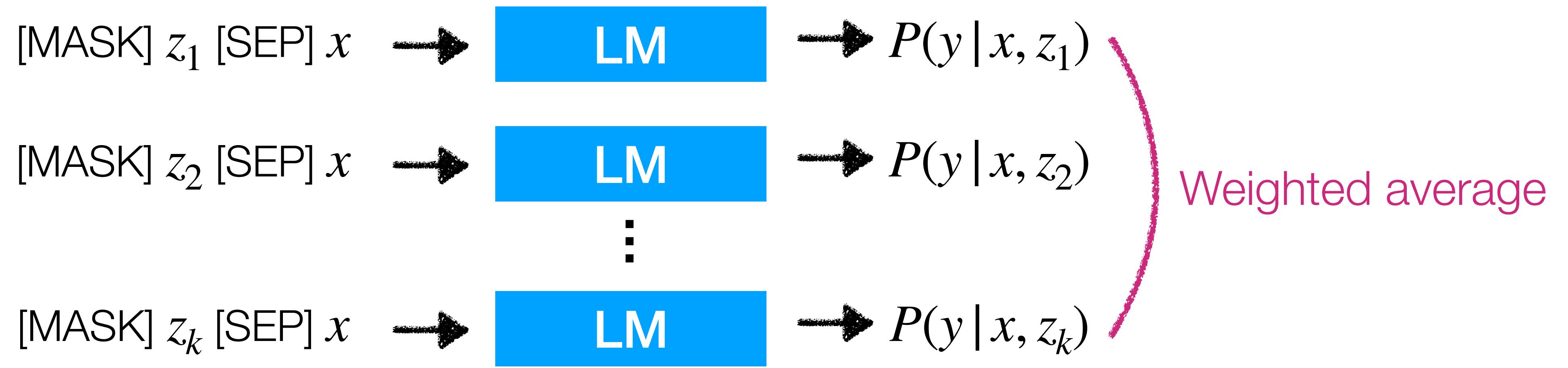
REALM: (2) Read stage



Need to approximate
→ Consider top k chunks only

$$\sum_{z \in \mathcal{D}} \underbrace{P(z | x)}_{\text{from the retrieve stage}} \underbrace{P(y | x, z)}_{\text{from the read stage}}$$

REALM: (2) Read stage



Need to approximate
→ Consider top k chunks only

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

from the retrieve stage from the read stage

0 if not one of top k

REALM (Guu et al 2020)

What to retrieve?

- Chunks
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

REALM (Guu et al 2020)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

REALM (Guu et al 2020)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer ✓
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

REALM (Guu et al 2020)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer ✓
- Intermediate layers
- Output layer

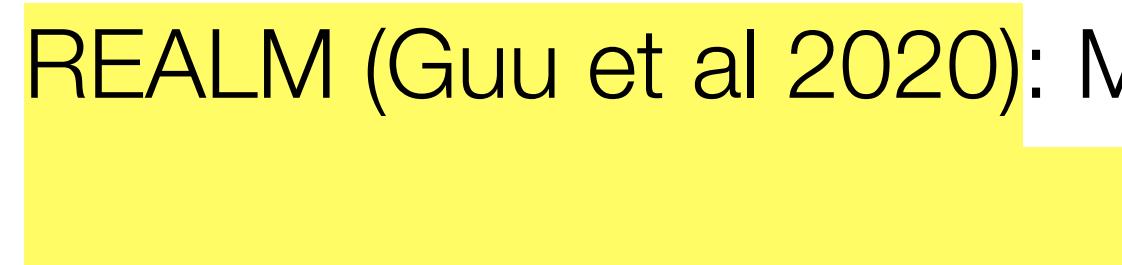
When to retrieve?

- Once ✓
- Every n tokens ($n > 1$)
- Every token

REALM and subsequent work

REALM and subsequent work

- * REALM (Guu et al 2020): MLM followed by fine-tuning on open-domain QA



REALM and subsequent work

- * REALM (Guu et al 2020): MLM followed by fine-tuning on open-domain QA
- * DPR (Karpukhin et al 2020): Pipeline training instead of joint training, fine-tuned on open-domain QA (no explicit language modeling)

REALM and subsequent work

- * REALM (Guu et al 2020): MLM followed by fine-tuning on open-domain QA
- * DPR (Karpukhin et al 2020): Pipeline training instead of joint training, fine-tuned on open-domain QA (no explicit language modeling)
- * RAG (Lewis et al 2020): “Generative” instead of “masked language modeling”, fine-tuned on open-domain QA & knowledge intensive tasks (no explicit language modeling)

REALM and subsequent work

- * REALM (Guu et al 2020): MLM followed by fine-tuning on open-domain QA
- * DPR (Karpukhin et al 2020): Pipeline training instead of joint training, fine-tuned on open-domain QA (no explicit language modeling)
- * RAG (Lewis et al 2020): “Generative” instead of “masked language modeling”, fine-tuned on open-domain QA & knowledge intensive tasks (no explicit language modeling)
- * Atlas (Izcard et al 2022): Combine RAG with retrieval-based language model pre-training based on the encoder-decoder architecture (more to come in Section 4), fine-tuned on open-domain QA & other QA tasks

REALM and subsequent work

- * REALM (Guu et al 2020): MLM followed by fine-tuning on open-domain QA
- * DPR (Karpukhin et al 2020): Pipeline training instead of joint training, fine-tuned on open-domain QA (no explicit language modeling)
- * RAG (Lewis et al 2020): “Generative” instead of “masked language modeling”, fine-tuned on open-domain QA & knowledge intensive tasks (no explicit language modeling)
- * Atlas (Izcard et al 2022): Combine RAG with retrieval-based language model pre-training based on the encoder-decoder architecture (more to come in Section 4), fine-tuned on open-domain QA & other QA tasks

For a while, mainly evaluated on
knowledge-intensive tasks (e.g. open-domain QA) with fine-tuning
(more context in Section 5)

REALM and subsequent work

- * REALM (Guu et al 2020): MLM followed by fine-tuning, focusing on open-domain QA
- * DPR (Karpukhin et al 2020): Pipeline training instead of joint training, focusing on open-domain QA (no explicit language modeling)
- * RAG (Lewis et al 2020): “Generative” instead of “masked language modeling”, focusing on open-domain QA & knowledge intensive tasks (no explicit language modeling)
- * Atlas (Izcard et al 2022): Combine RAG with retrieval-based language model pre-training based on the encoder-decoder architecture (more to come in Section 4), focusing on open-domain QA & knowledge intensive tasks
- * Papers that follow this approach focusing on **LM perplexity** have come out quite recently (Shi et al. 2023, Ram et al. 2023)

Retrieval-in-context LM

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

Ram et al. 2023. “In-Context Retrieval-Augmented Language Models”
Shi et al. 2023. “REPLUG: Retrieval-Augmented Black-Box Language Models”

Retrieval-in-context LM

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

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



Retrieval



* Can use multiple text blocks too (see the papers!)

FIFA World Cup 2026 will expand to 48 teams.

Retrieval-in-context LM

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

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



Retrieval

* Can use multiple text blocks too (see the papers!)



FIFA World Cup 2026 will expand to 48 teams. World Cup 2022 was the last with 32 teams, before the increase to



LM



48 in the 2026 tournament.

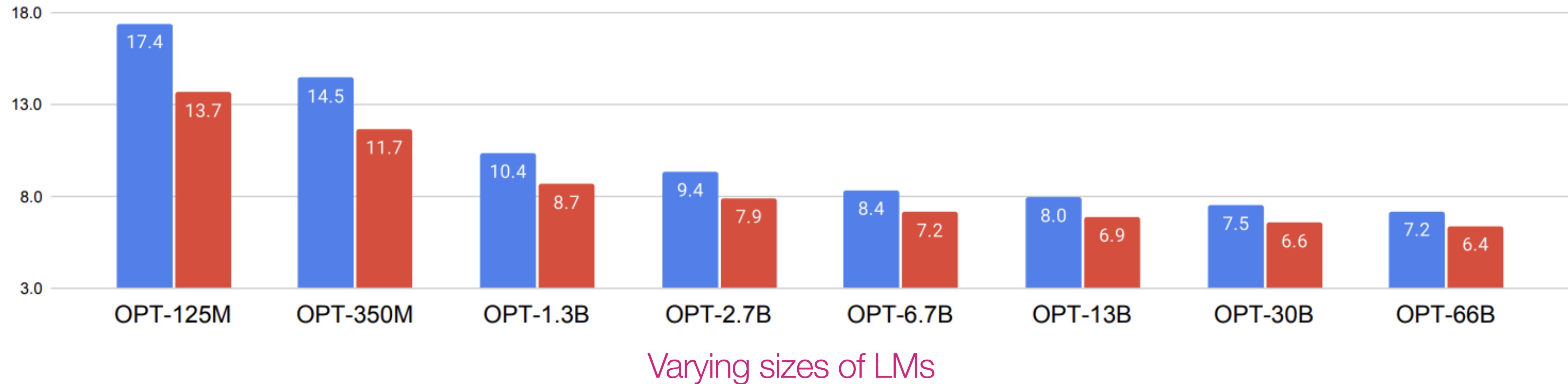
Ram et al. 2023. “In-Context Retrieval-Augmented Language Models”

Shi et al. 2023. “REPLUG: Retrieval-Augmented Black-Box Language Models”

Retrieval-in-context LM

Perplexity: The lower the better

■ No Retrieval ■ In-Context RALM (BM25)



Retrieval helps over all sizes of LMs

Graphs from Ram et al. 2023

Retrieval-in-context LM

Is $q=x$ necessary?

Retrieval-in-context LM

Is $q=x$ necessary?

x = Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Retrieval-in-context LM

Is $q=x$ necessary?

x = Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Retrieval

Retrieval-in-context LM

Is $q=x$ necessary?

x = Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Retrieval

The U.S. national team defeated Iran 1-0.

Retrieval-in-context LM

Is $q=x$ necessary?

x = Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Retrieval

The U.S. national team defeated Iran 1-0.

Does not cover “tokens that will come next”

Retrieval-in-context LM

Is $q=x$ necessary?

x = Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to



Retrieval

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



Retrieval

The U.S. national team defeated Iran 1-0.

Does not cover “tokens that will come next”

Retrieval-in-context LM

Is $q=x$ necessary?

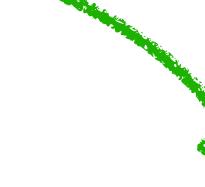
x = Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to



Retrieval

The U.S. national team defeated Iran 1-0.



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



Retrieval

FIFA World Cup 2026 will expand to 48 teams.

Does not cover “tokens that will come next”

Retrieval-in-context LM

Is $q=x$ necessary?

x = Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to

Team USA celebrates after winning its match against Iran at Al Thumama Stadium in Group B play of the FIFA World Cup 2022 on Nov. 29, 2022. (..) World Cup 2022 was the last with 32 teams, before the increase to



Retrieval

The U.S. national team defeated Iran 1-0.

Does not cover “tokens that will come next”

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

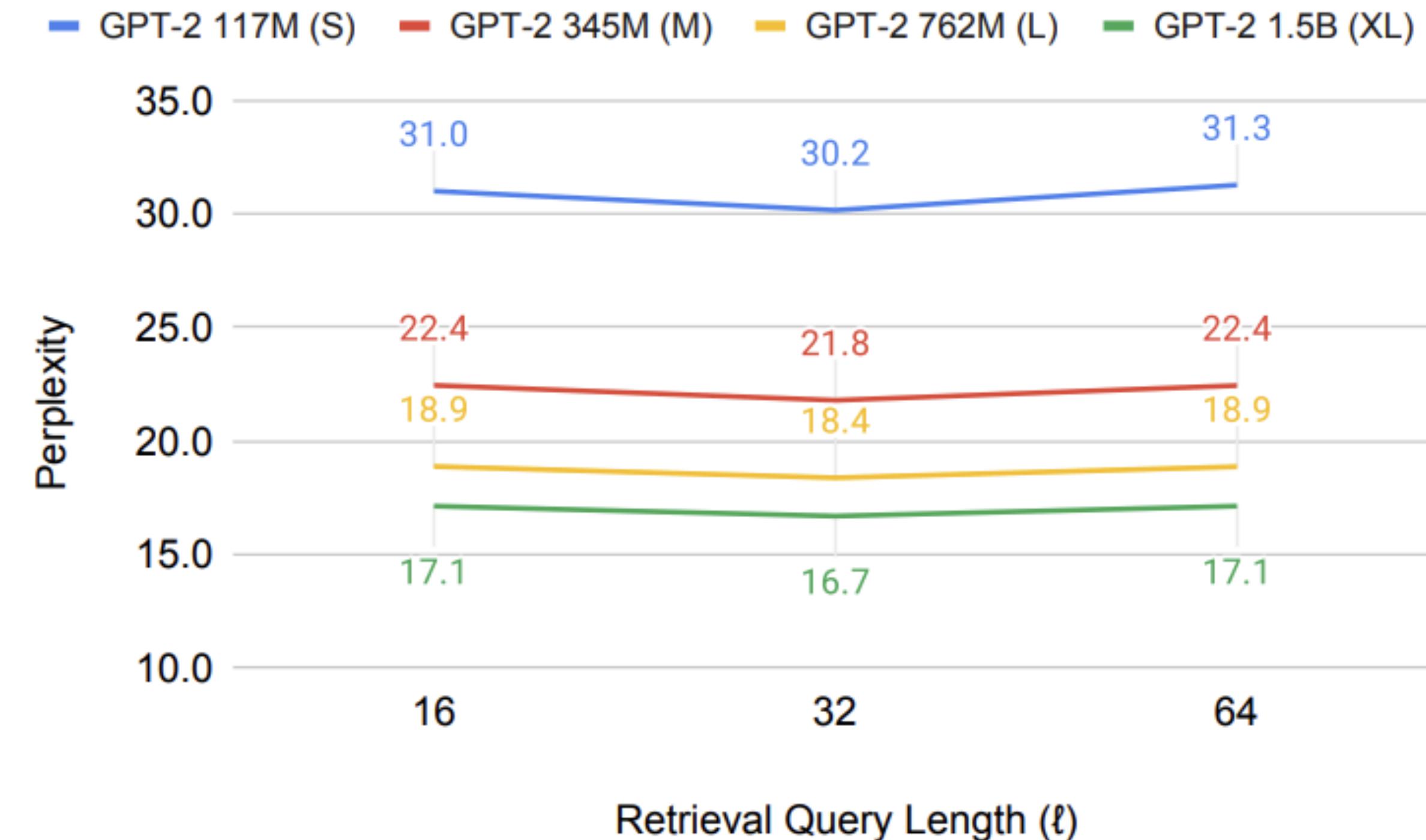


Retrieval

FIFA World Cup 2026 will expand to 48 teams.

more relevant to what will come next

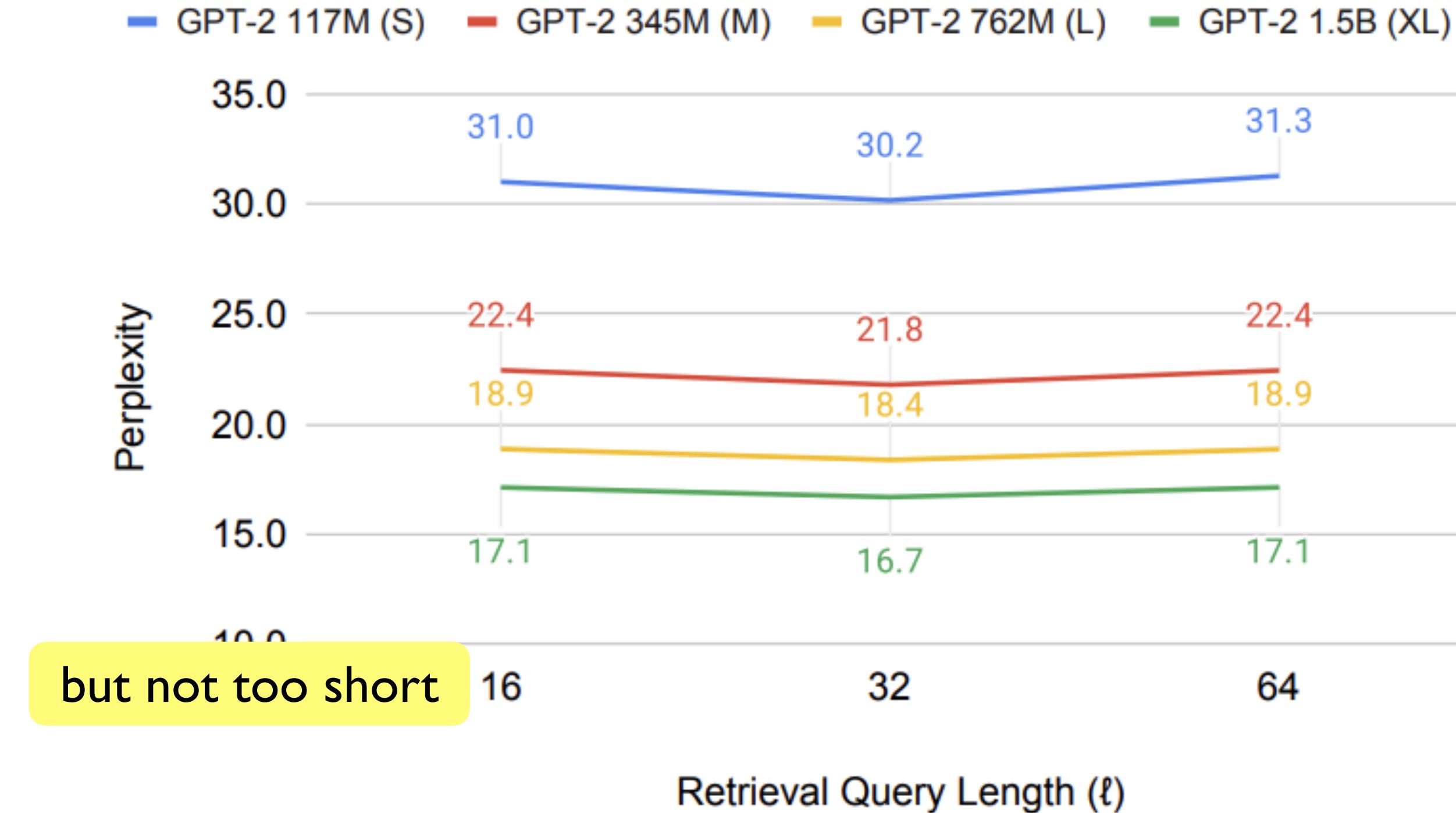
Retrieval-in-context LM



Shorter prefix (more recent tokens) as a query helps

Graphs from Ram et al. 2023

Retrieval-in-context LM



Shorter prefix (more recent tokens) as a query helps

Graphs from Ram et al. 2023

Retrieval-in-context LM

How frequent should retrieval be?

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament.

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with

LM



32 teams before the increase to 48 in the 2026 tournament.

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with



LM



32 teams before the increase to 48 in the 2026 tournament.

explained by retrieval

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with



LM



32 teams before the increase to 48 in the 2026 tournament.

explained by retrieval

not really covered

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with



LM



32 teams before the increase

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with



LM



32 teams before the increase

World Cup 2022 was the last with 32 teams before the increase



Retrieval



FIFA World Cup 2026 will expand to 48 teams.

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with



LM



32 teams before the increase

World Cup 2022 was the last with 32 teams before the increase



Retrieval



FIFA World Cup 2026 will expand to 48 teams. World Cup 2022 was the last with 32 teams, before the increase

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with



LM



32 teams before the increase

World Cup 2022 was the last with 32 teams before the increase



Retrieval



FIFA World Cup 2026 will expand to 48 teams. World Cup 2022 was the last with 32 teams, before the increase



LM



to 48 in the 2026 tournament.

Retrieval-in-context LM

How frequent should retrieval be?

World Cup 2022 was the last with



Retrieval



The 2022 FIFA World Cup (...) 32 national teams involved in the tournament. World Cup 2022 was the last with



LM



32 teams before the increase

World Cup 2022 was the last with 32 teams before the increase



Retrieval



FIFA World Cup 2026 will expand to 48 teams. World Cup 2022 was the last with 32 teams, before the increase



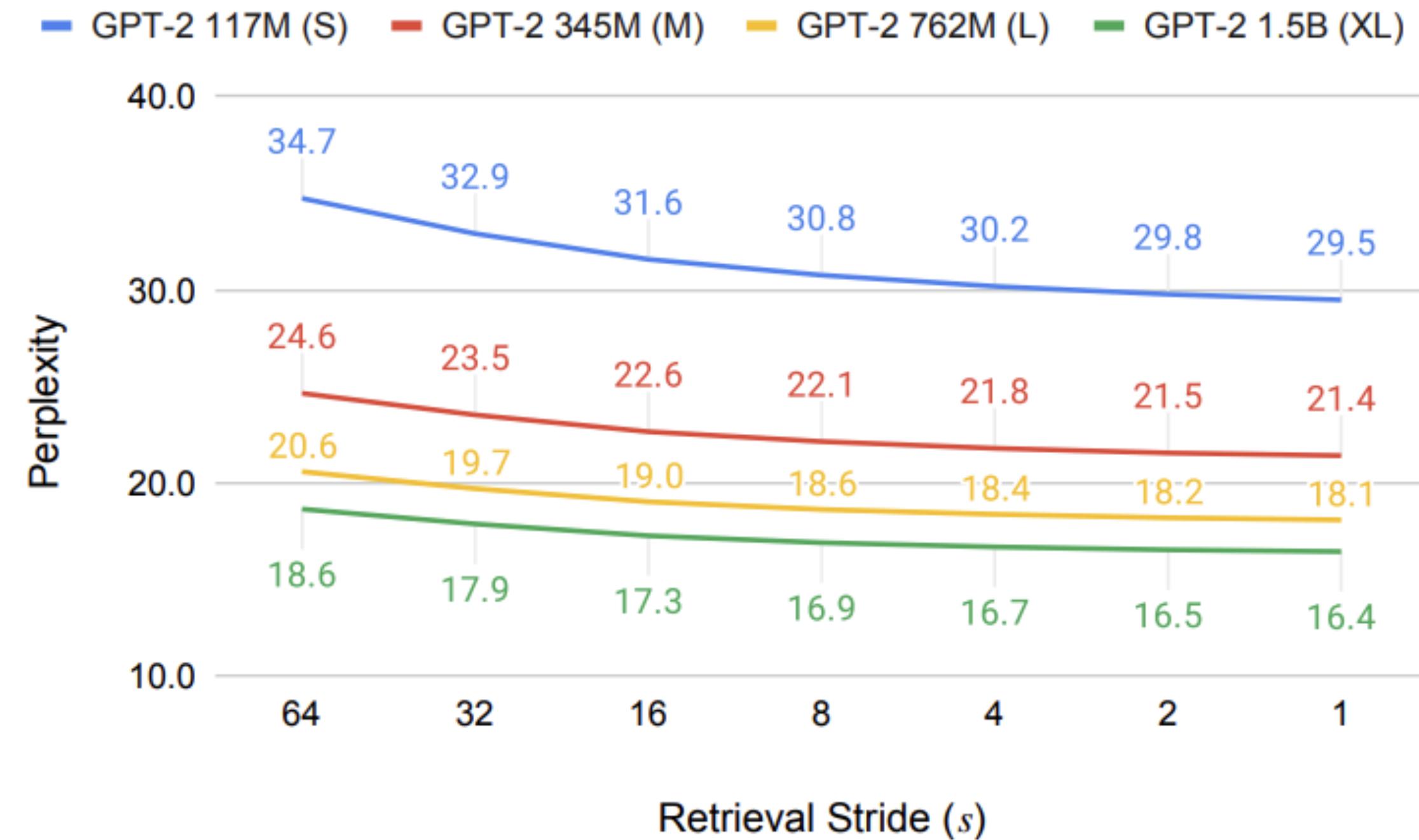
LM



to 48 in the 2026 tournament.

Retrieval results from a new query explain them!

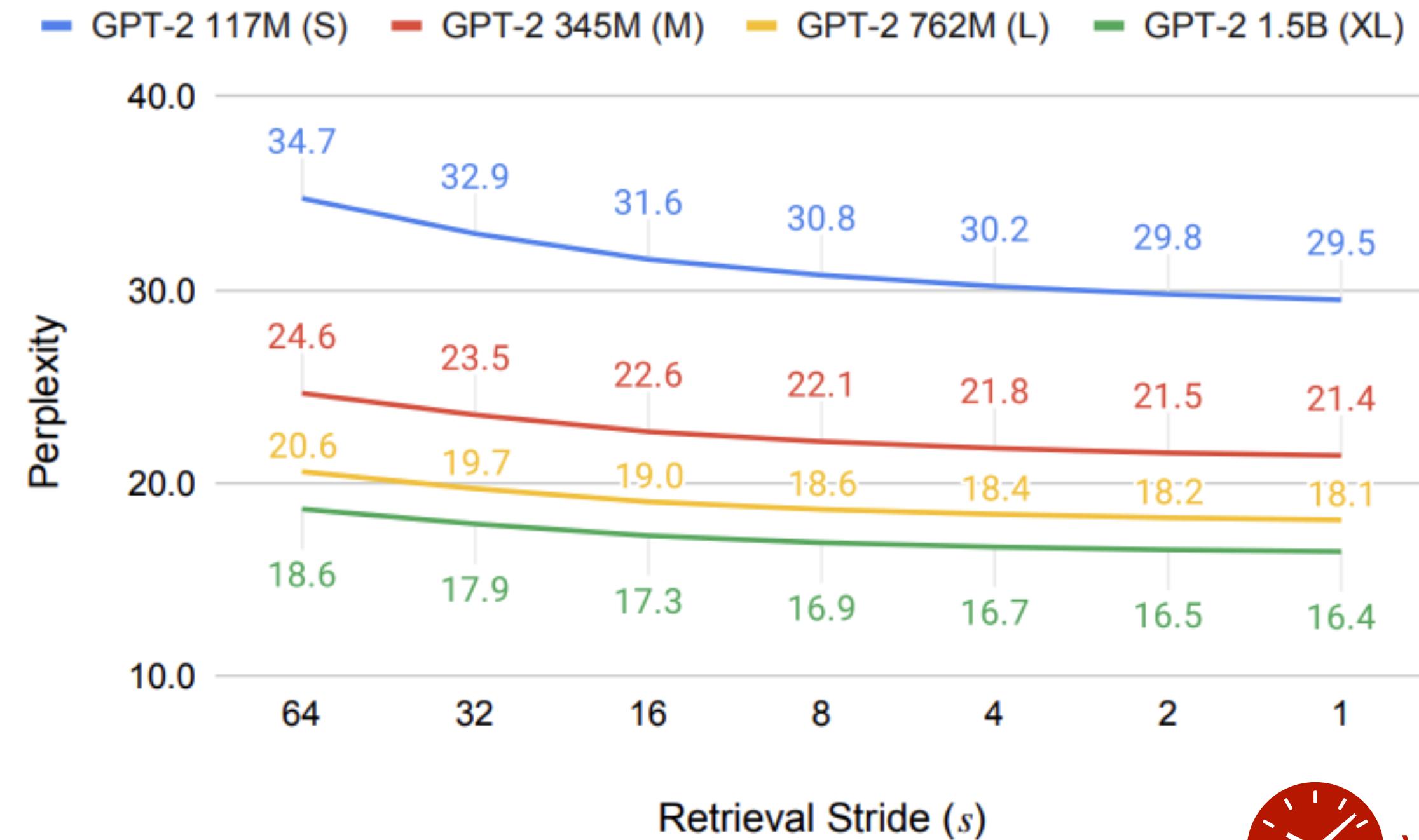
Retrieval-in-context LM



Retrieving more frequently helps

Graphs from Ram et al. 2023

Retrieval-in-context LM



Retrieving more frequently helps



with cost in inference time

Graphs from Ram et al. 2023

Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer ✓
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer ✓
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$) ✓
- Every token

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

Applying the same approach to LM raised new questions
which mattered less in prior work (e.g. REALM) with short inputs & short outputs

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

can be very inefficient to retrieve many text chunks, frequently

RETRO (Borgeaud et al. 2021)

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

RETRO (Borgeaud et al. 2021)

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

RETRO (Borgeaud et al. 2021)

- ✓ Incorporation in the “intermediate layer” instead of the “input” layer
→ designed for many chunks, frequently, more efficiently
- ✓ Scale the datastore (1.8T tokens)

RETRO (Borgeaud et al. 2021)

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

RETRO (Borgeaud et al. 2021)

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

x₁

x₂

x₃

RETRO (Borgeaud et al. 2021)

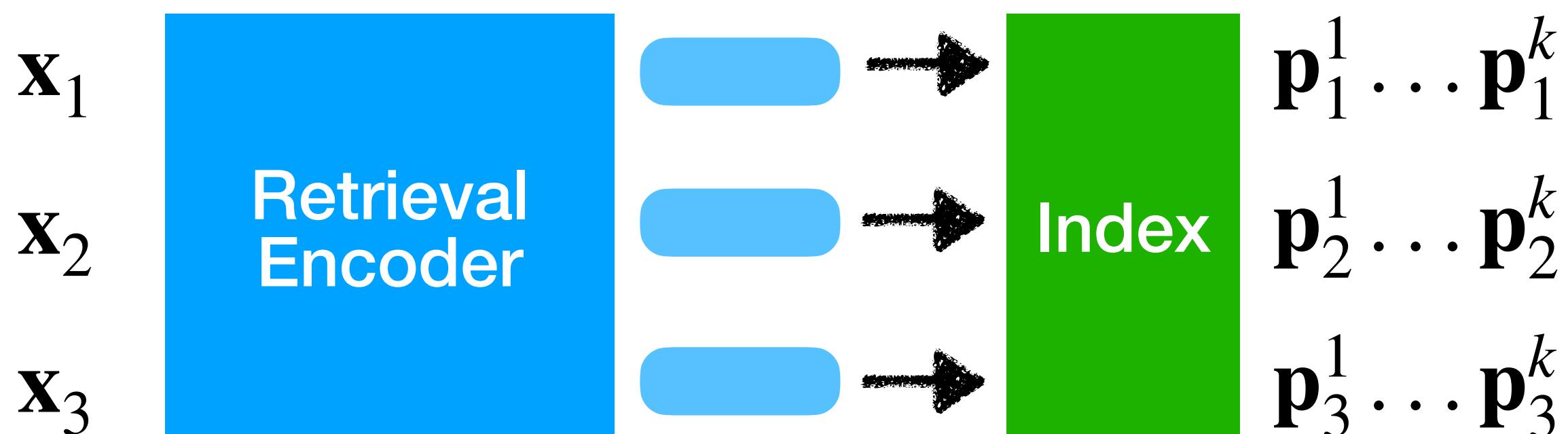
\mathbf{x} = World Cup 2022 was ~~the last with 32 teams~~, before the increase to

\mathbf{x}_1

\mathbf{x}_2

\mathbf{x}_3

(k chunks of text per split)



RETRO (Borgeaud et al. 2021)

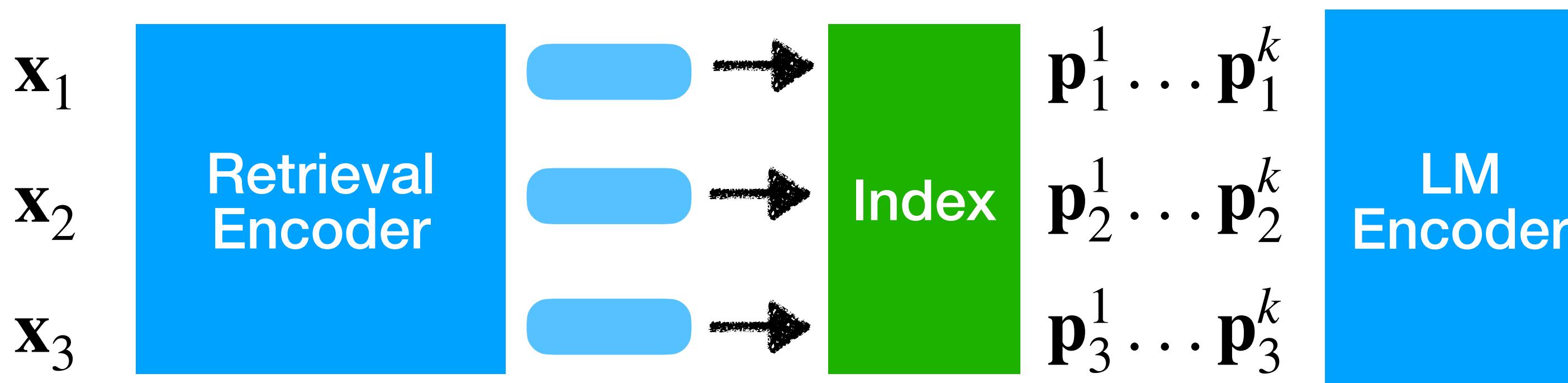
\mathbf{x} = World Cup 2022 was ~~the last with 32 teams~~, before the increase to

\mathbf{x}_1

\mathbf{x}_2

\mathbf{x}_3

(k chunks of text per split)



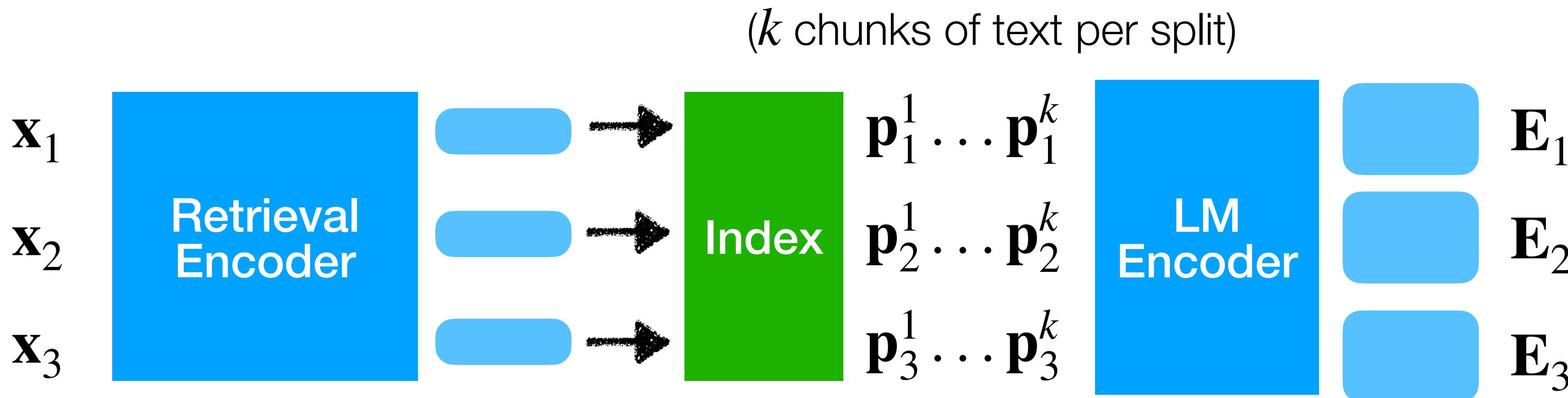
RETRO (Borgeaud et al. 2021)

\mathbf{x} = World Cup 2022 was~~the~~ last with 32 teams,~~before~~ before the increase to

\mathbf{x}_1

\mathbf{x}_2

\mathbf{x}_3



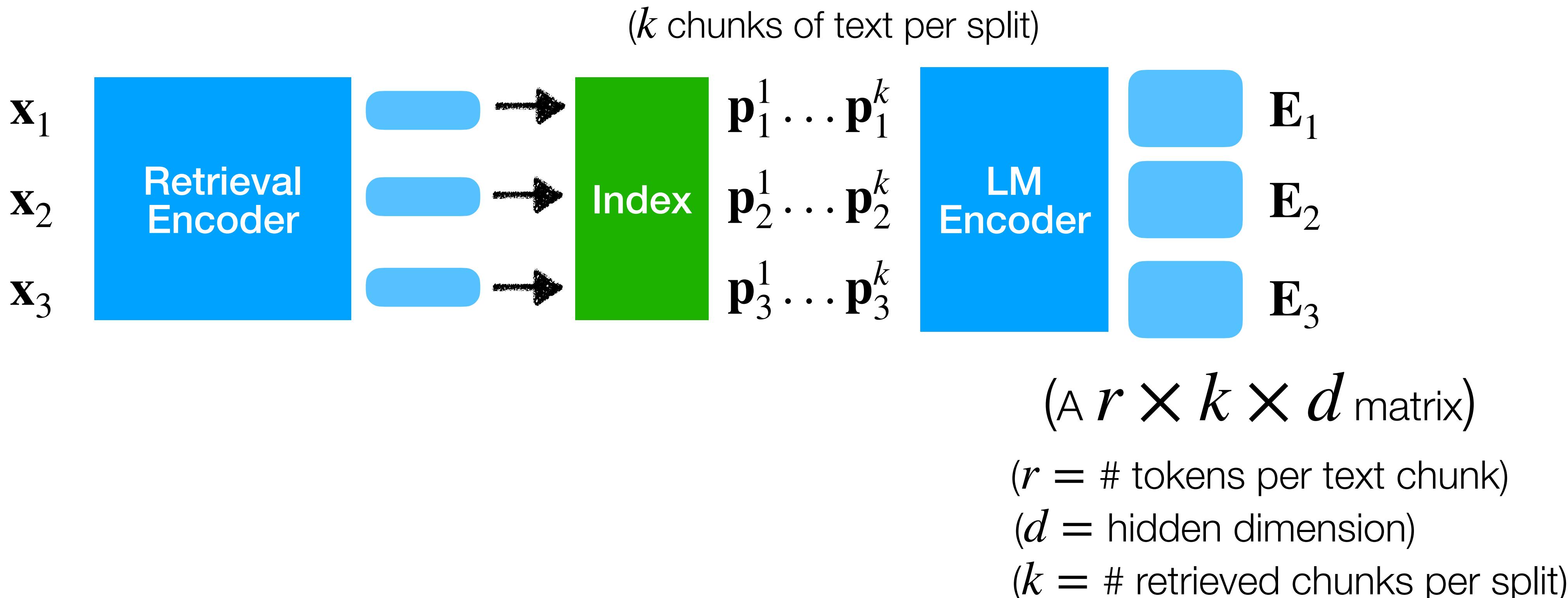
RETRO (Borgeaud et al. 2021)

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

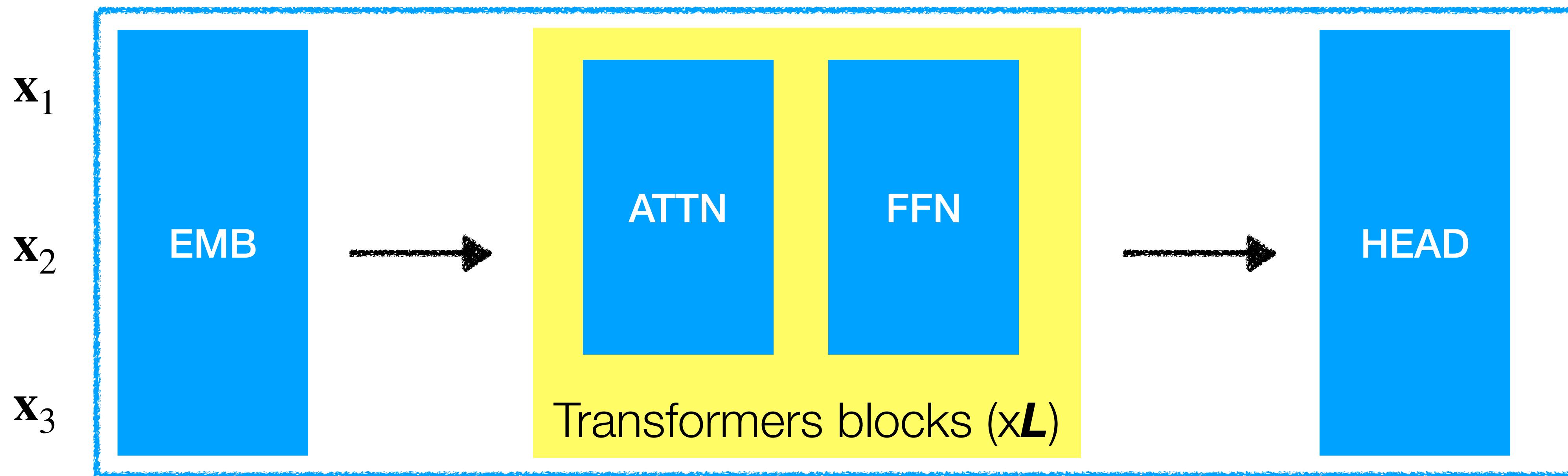
\mathbf{x}_1

\mathbf{x}_2

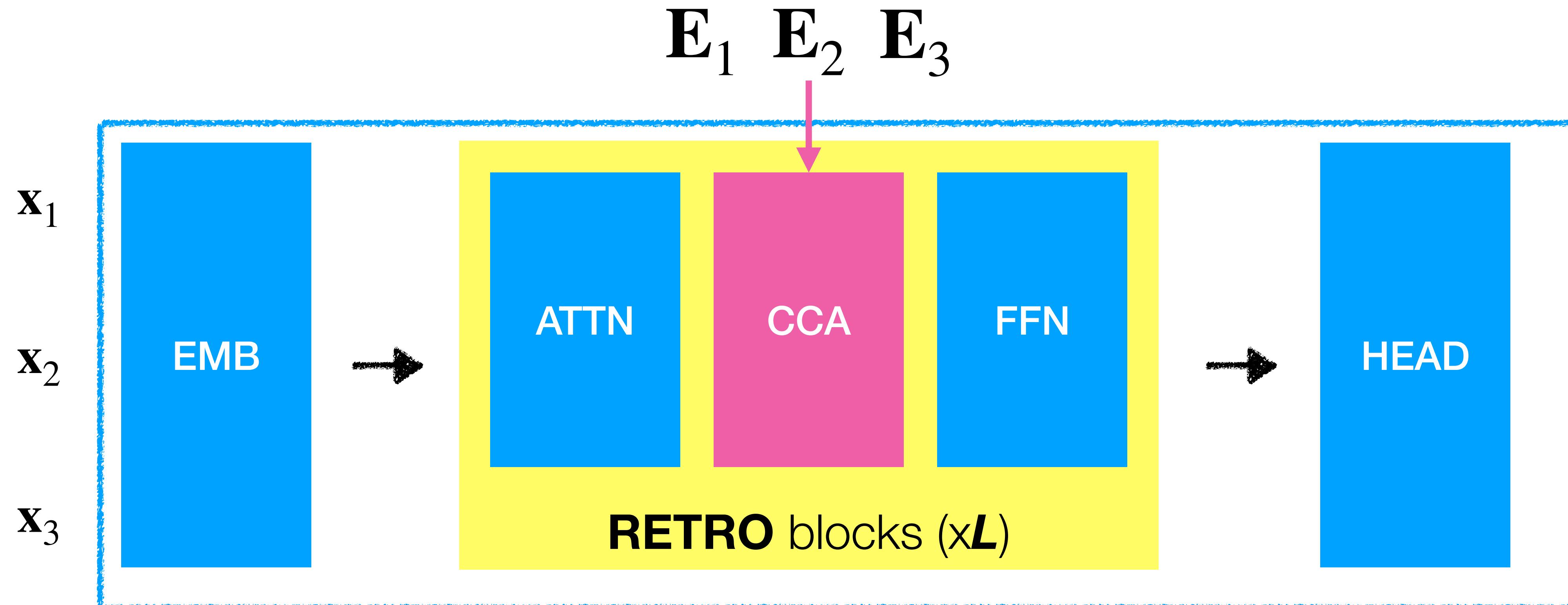
\mathbf{x}_3



Regular decoder

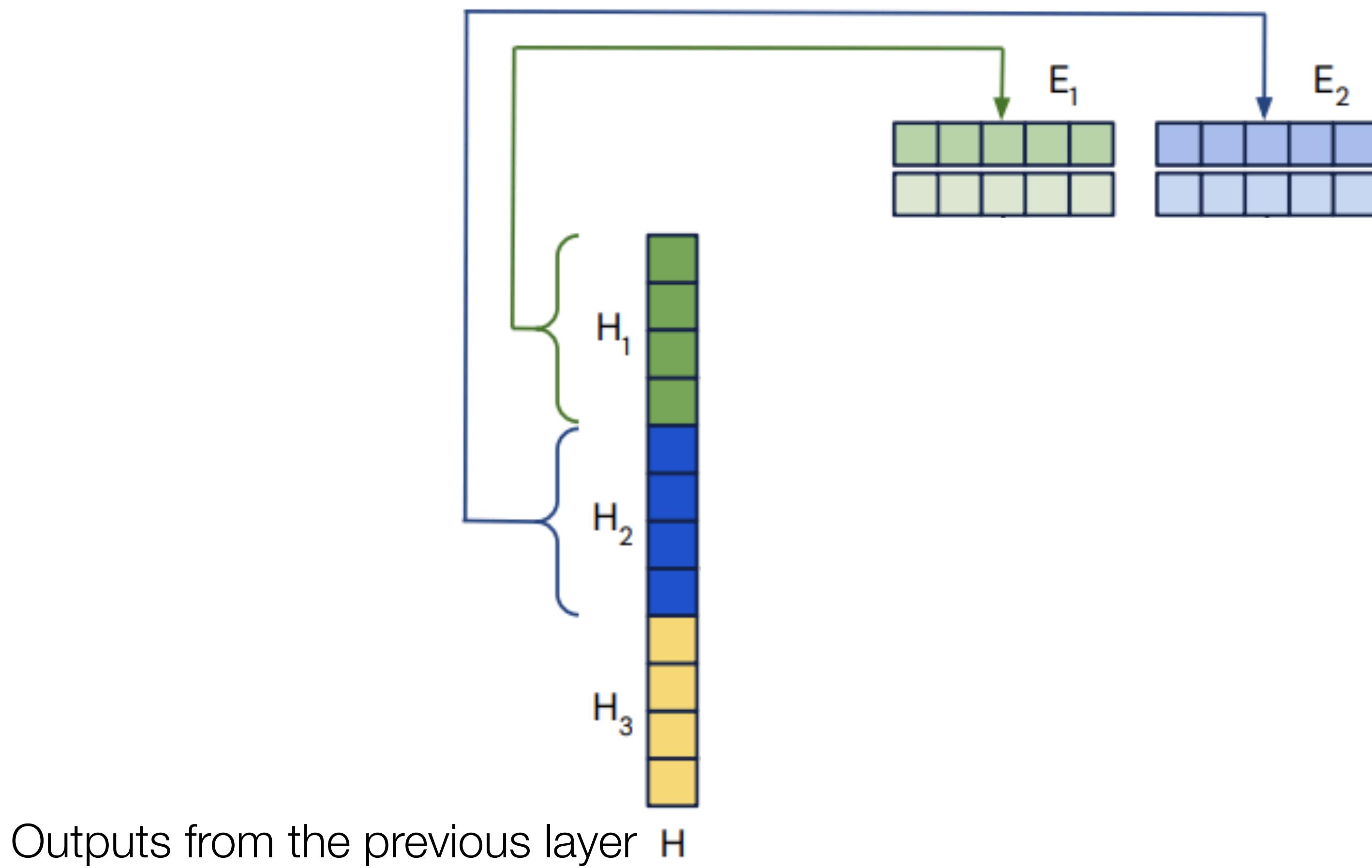


Decoder in RETRO

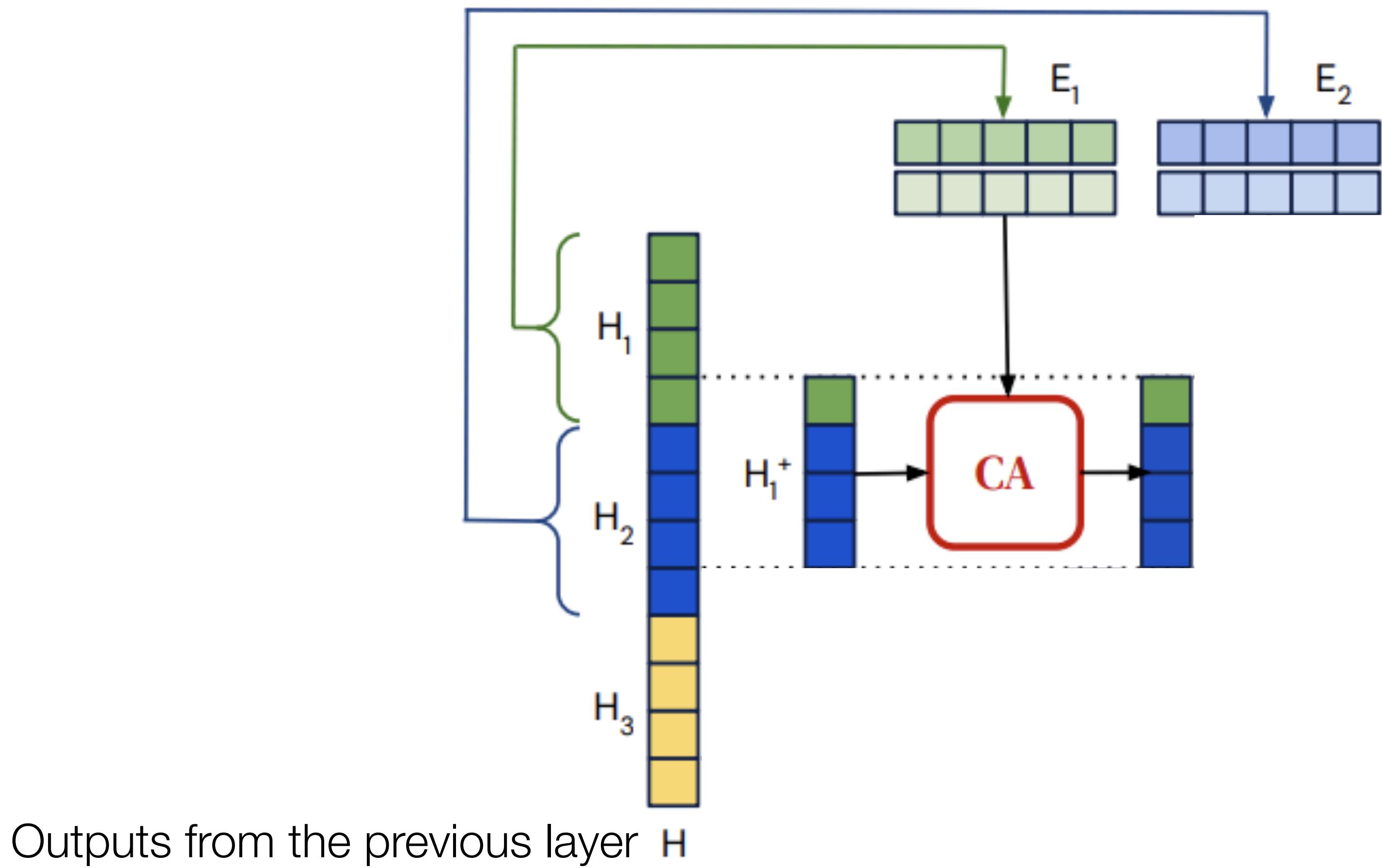


Chunked Cross Attention (CCA)

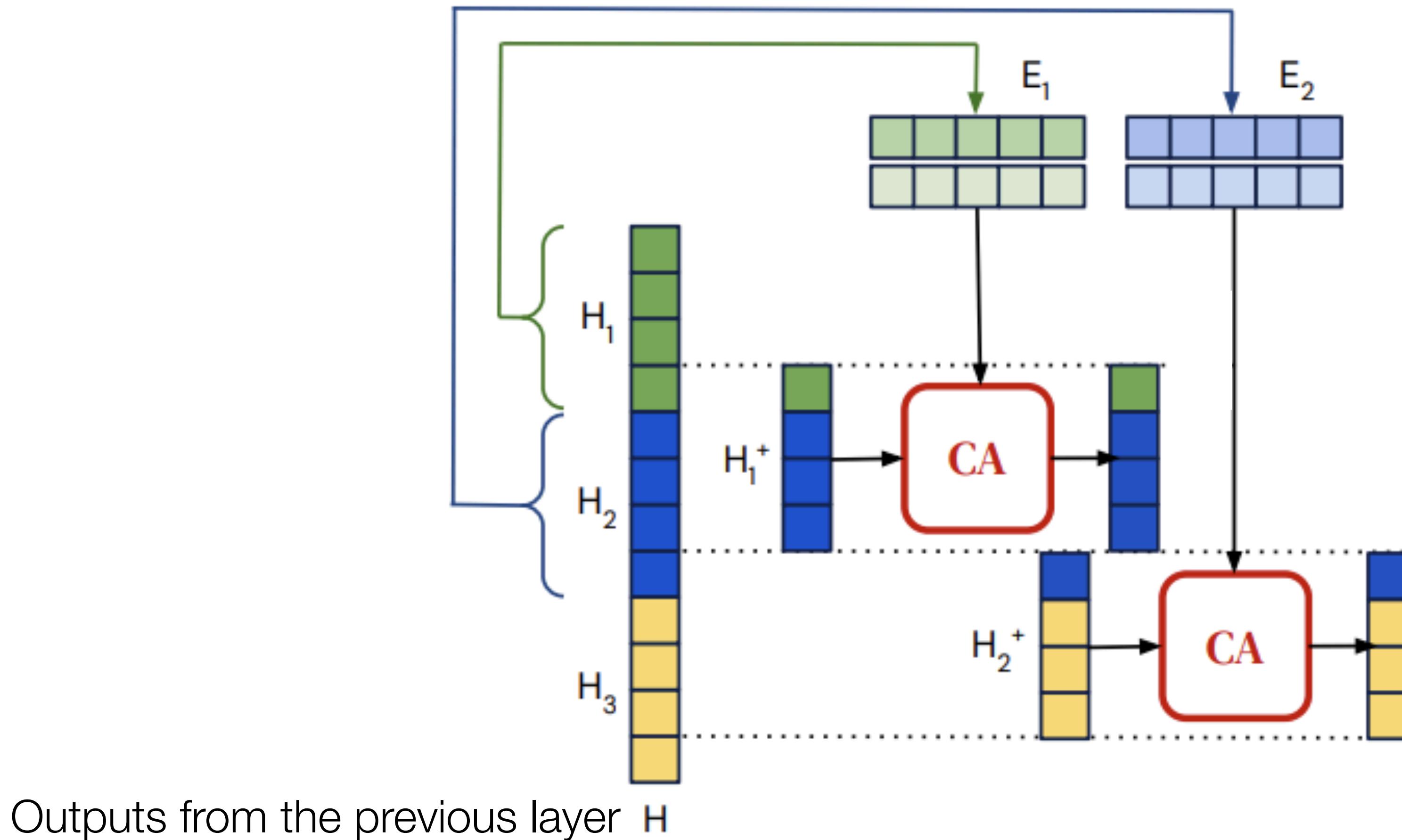
Chunked Cross Attention



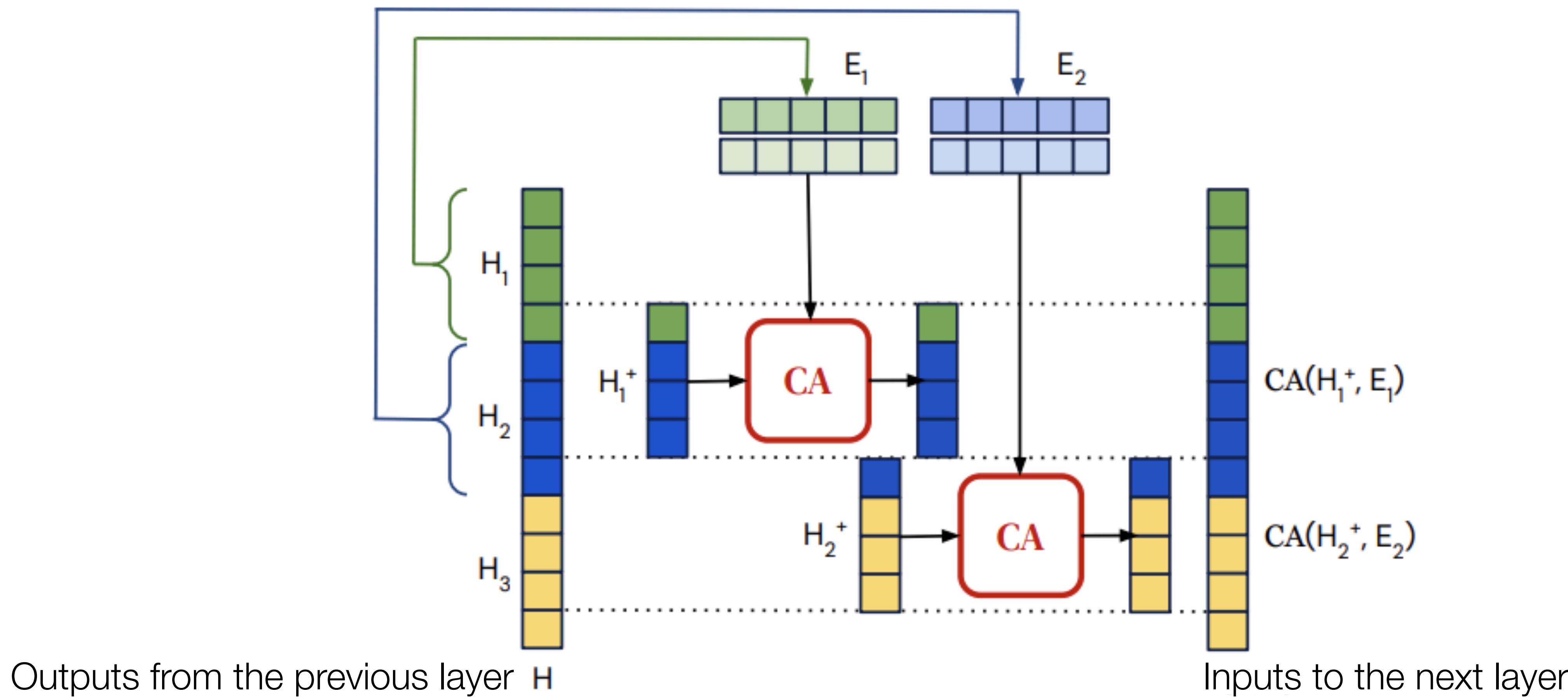
Chunked Cross Attention



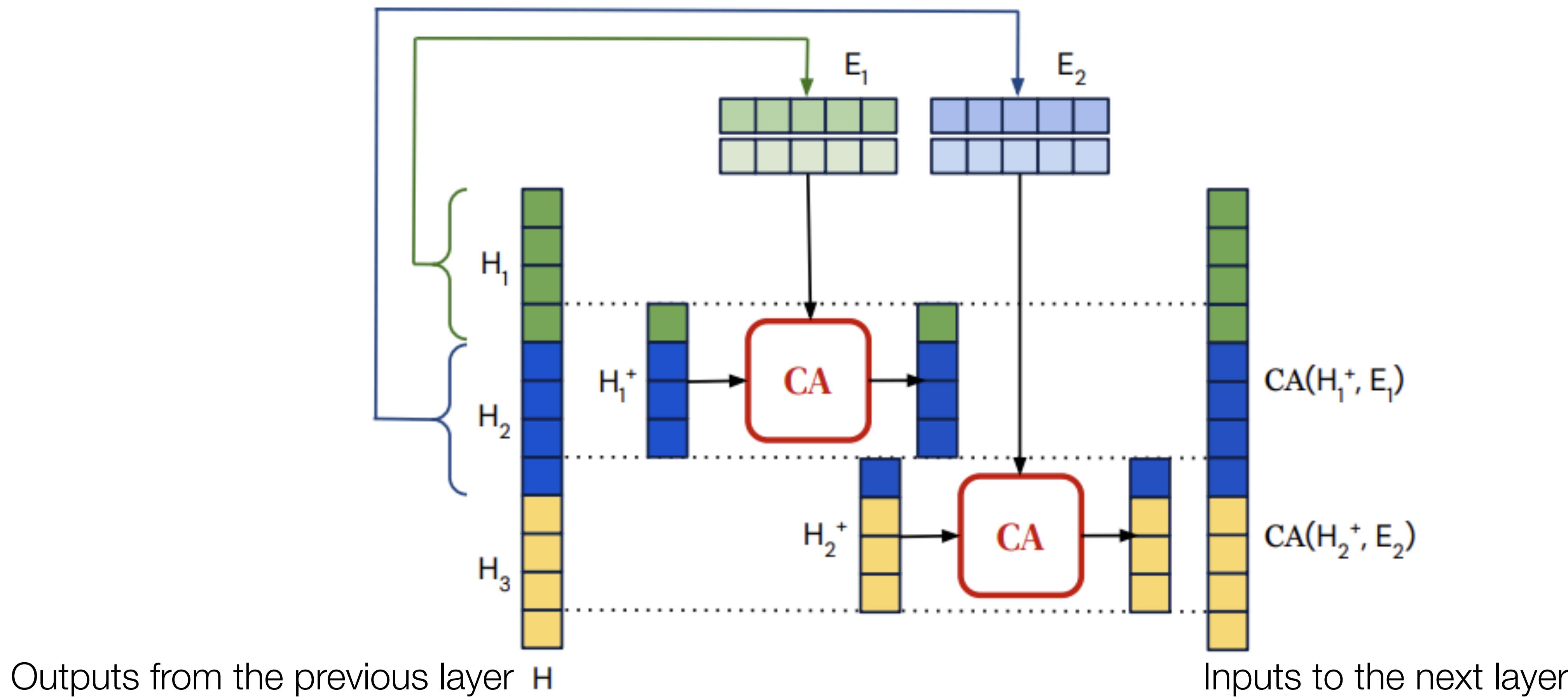
Chunked Cross Attention



Chunked Cross Attention

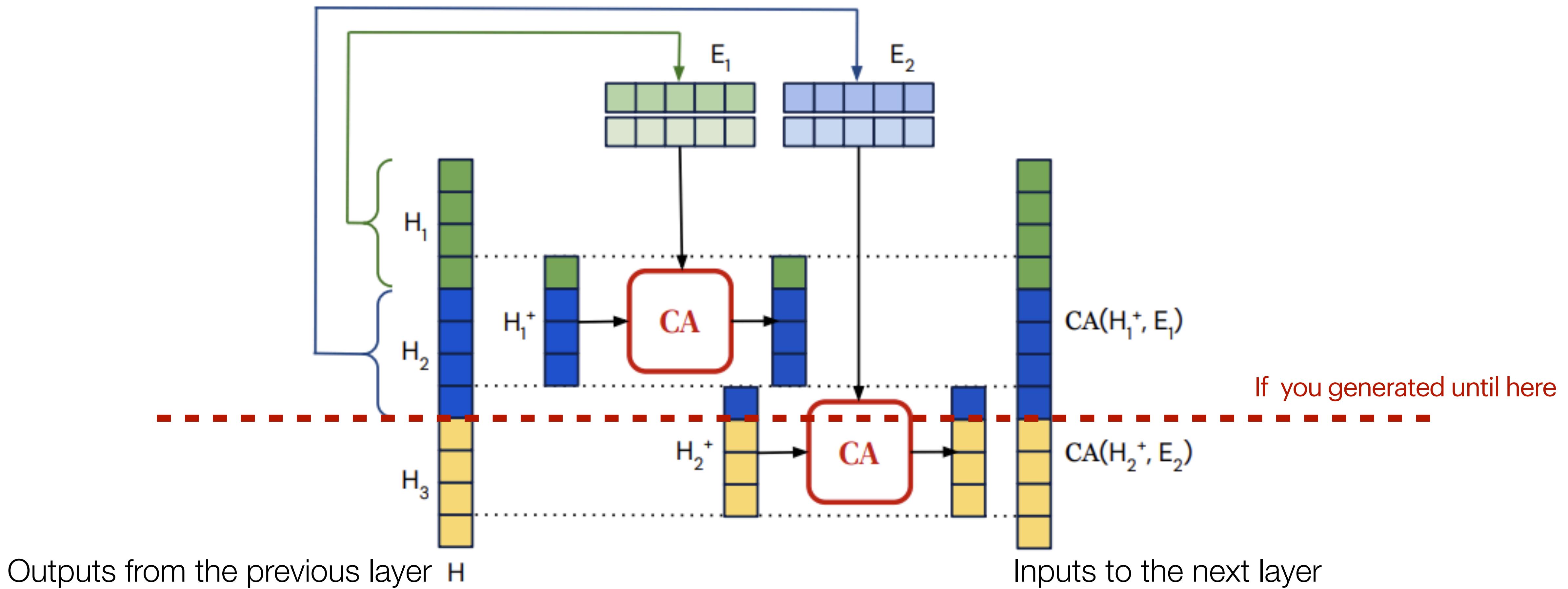


Chunked Cross Attention



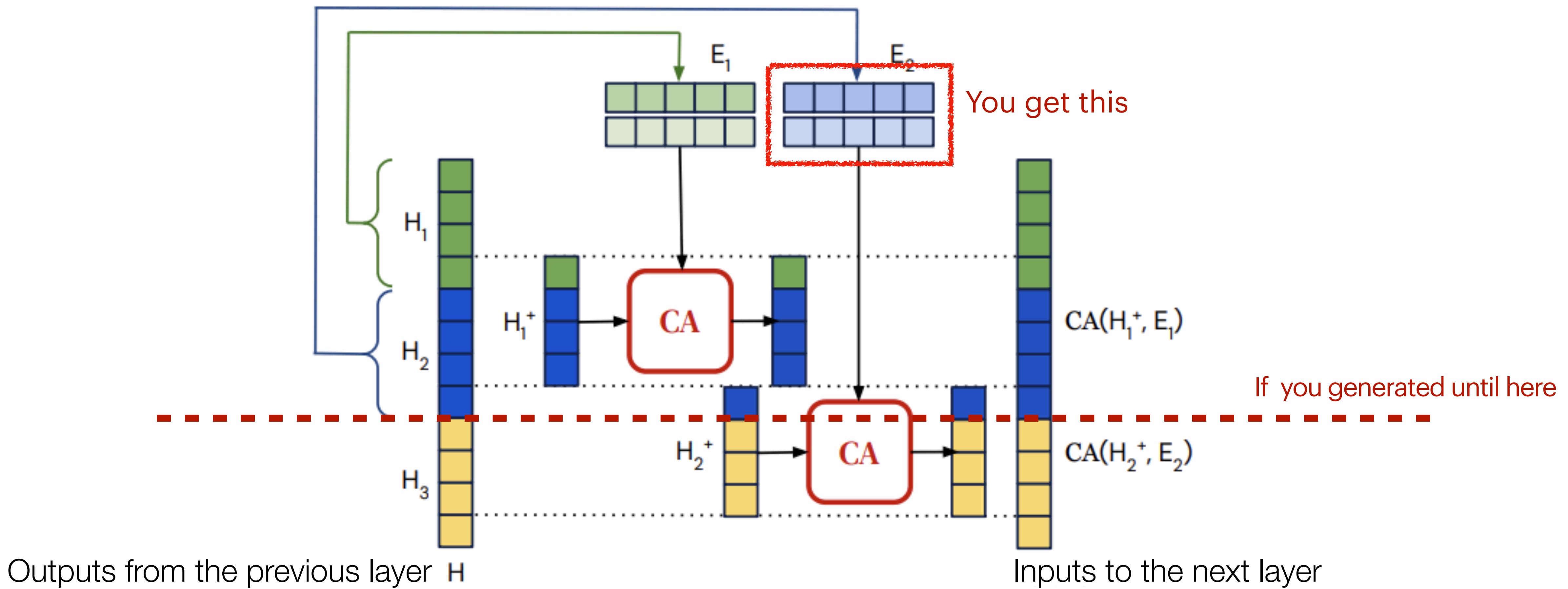
✓ Cross-attention can be computed *in parallel*

Chunked Cross Attention



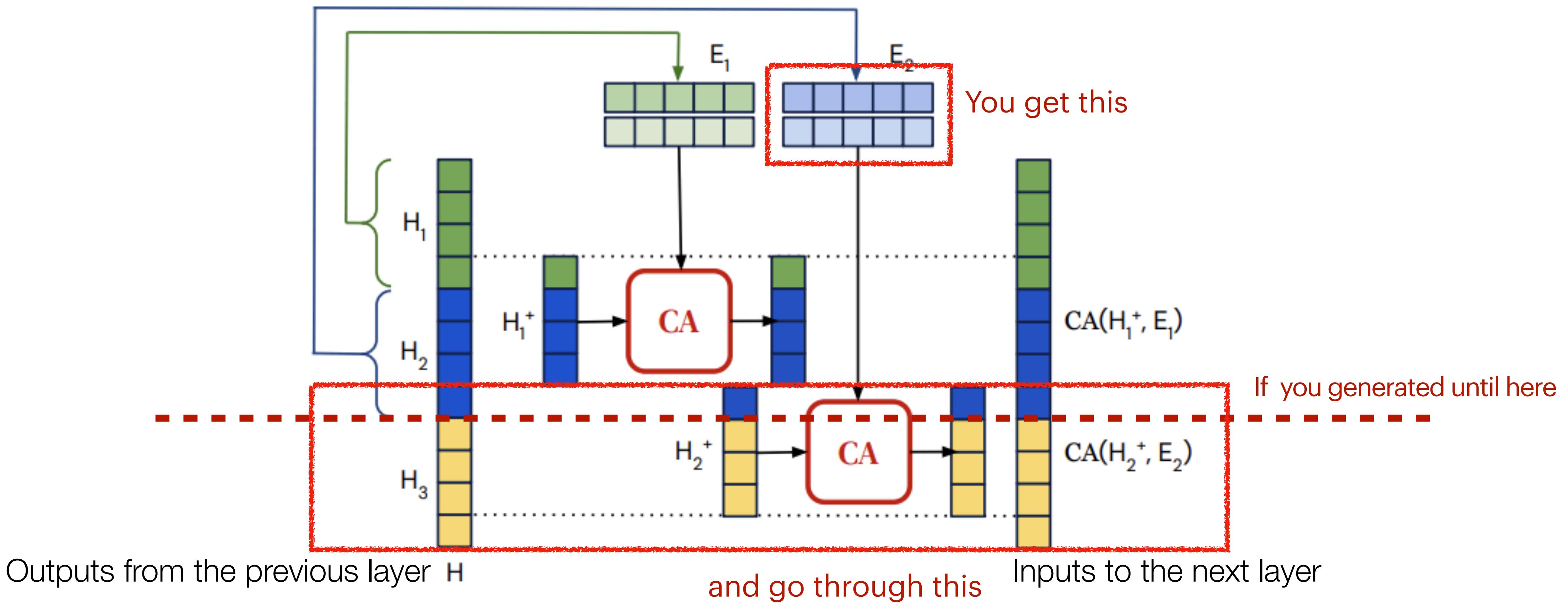
Cross-attention can be computed *in parallel*

Chunked Cross Attention



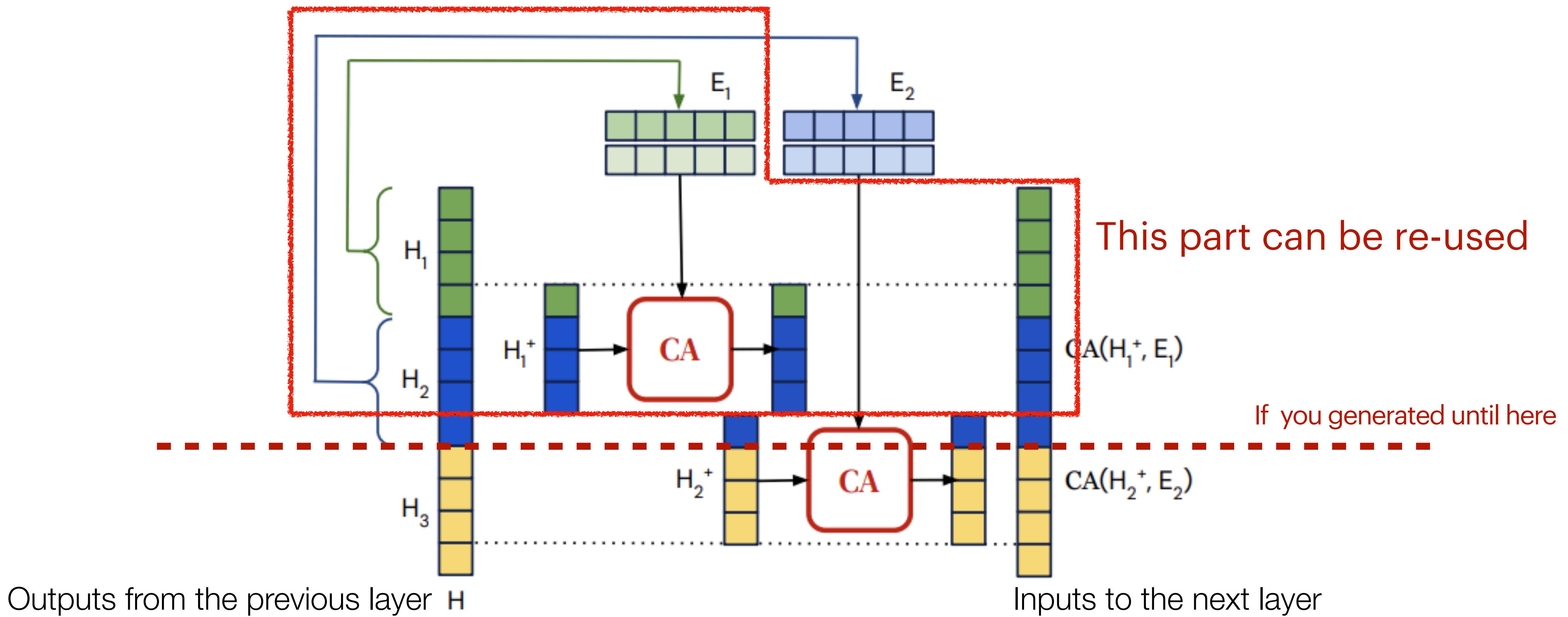
✓ Cross-attention can be computed *in parallel*

Chunked Cross Attention



✓ Cross-attention can be computed *in parallel*

Chunked Cross Attention



✓ Cross-attention can be computed *in parallel*

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
kNN-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
kNN-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

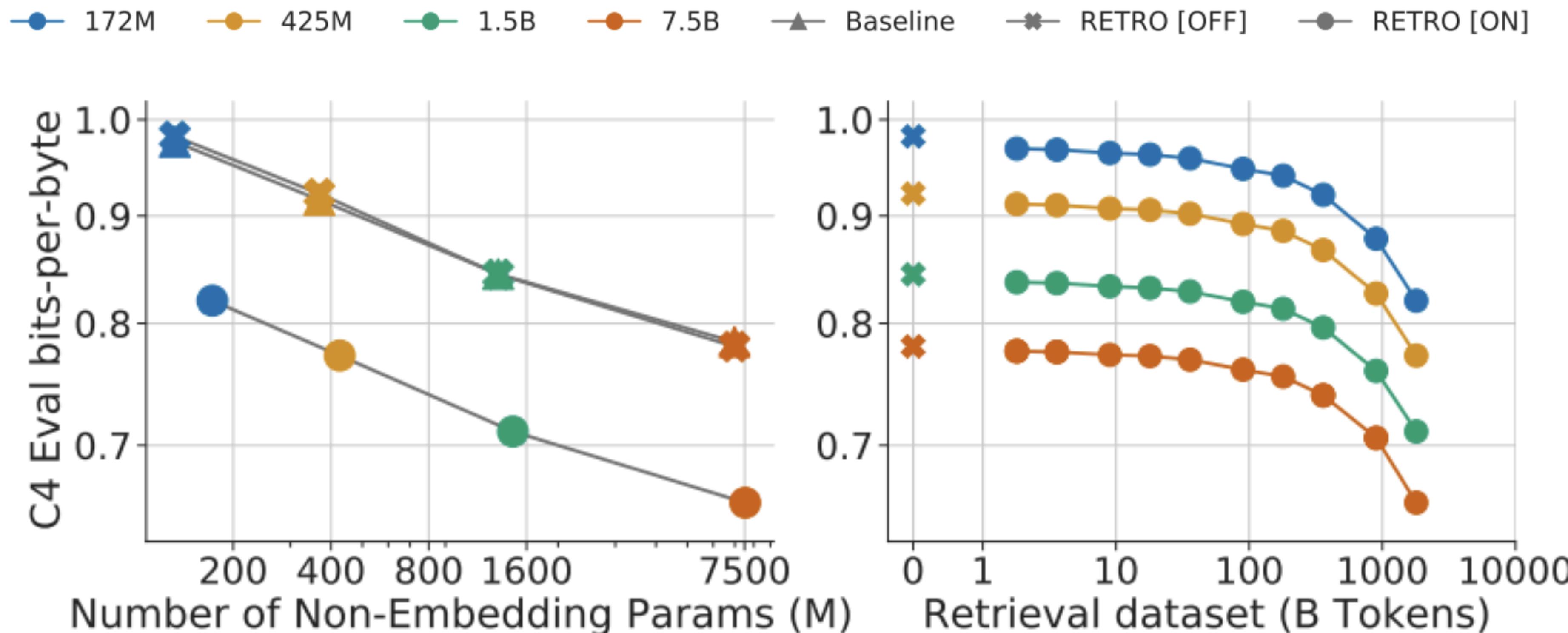
Results

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
kNN-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
kNN-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

Results



Gains are constant with model scale

The larger datastore is, the better

RETRO (Borgeaud et al. 2021)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

RETRO (Borgeaud et al. 2021)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers ✓
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

RETRO (Borgeaud et al. 2021)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers ✓
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$) ✓
- Every token

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

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	Input layer Intermediate layers	Every n tokens

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	Input layer Intermediate layers	Every n tokens



Can use many blocks, more frequently, more efficiently

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	Input layer Intermediate layers	Every n tokens



Can use many blocks, more frequently, more efficiently



Additional complexity; Can't be used without training (more in section 4)

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

What else?

kNN-LM (Khandelwal et al. 2020)

kNN-LM (Khandelwal et al. 2020)

- ✓ A different way of using retrieval, where the LM outputs a nonparametric distribution over every token in the data.

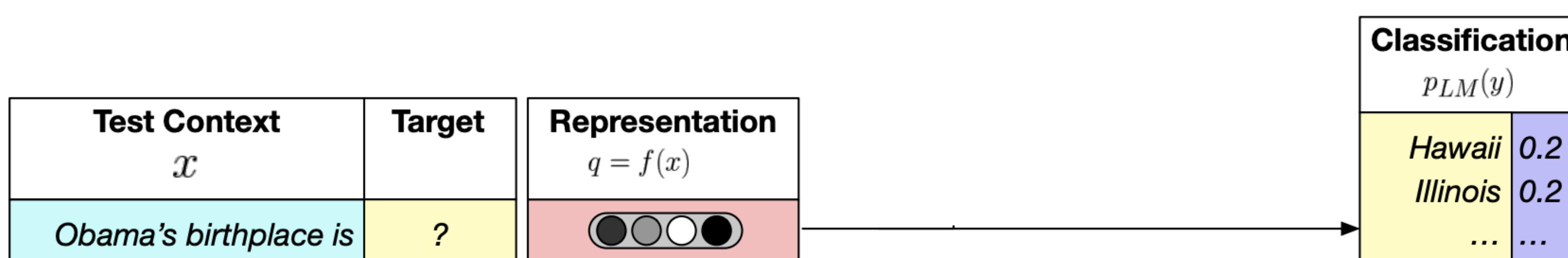
kNN-LM (Khandelwal et al. 2020)

- ✓ A different way of using retrieval, where the LM outputs a nonparametric distribution over every token in the data.
- ✓ Can be seen as an incorporation in the “output” layer

kNN-LM (Khandelwal et al. 2020)

Test Context	Target
x	
<i>Obama's birthplace is</i>	?

kNN-LM (Khandelwal et al. 2020)



kNN-LM (Khandelwal et al. 2020)

... Obama was senator for Illinois from 1997 to 2005, Barack is Married to Michelle and their first daughter, ... Obama was born in Hawaii, and graduated from Columbia University. ... Obama is a native of Hawaii,

Test Context	Target	Representation
x		$q = f(x)$
<i>Obama's birthplace is</i>	?	

kNN-LM (Khandelwal et al. 2020)

Training Contexts	Targets
c_i	v_i
<i>Obama was senator for</i>	<i>Illinois</i>
<i>Barack is married to</i>	<i>Michelle</i>
<i>Obama was born in</i>	<i>Hawaii</i>
...	...
<i>Obama is a native of</i>	<i>Hawaii</i>

... Obama was senator for Illinois from 1997 to 2005, Barack is Married to Michelle and their first daughter, ... Obama was born in Hawaii, and graduated from Columbia University. ... Obama is a native of Hawaii,

Test Context	Target	Representation
x		$q = f(x)$
<i>Obama's birthplace is</i>	?	

kNN-LM (Khandelwal et al. 2020)

The size of the datastore = # of tokens in the corpus (>1B)

Training Contexts	Targets
c_i	v_i
<i>Obama was senator for</i>	<i>Illinois</i>
<i>Barack is married to</i>	<i>Michelle</i>
<i>Obama was born in</i>	<i>Hawaii</i>
...	...
<i>Obama is a native of</i>	<i>Hawaii</i>

... Obama was senator for Illinois from 1997 to 2005, Barack is Married to Michelle and their first daughter, ... Obama was born in Hawaii, and graduated from Columbia University. ... Obama is a native of Hawaii,

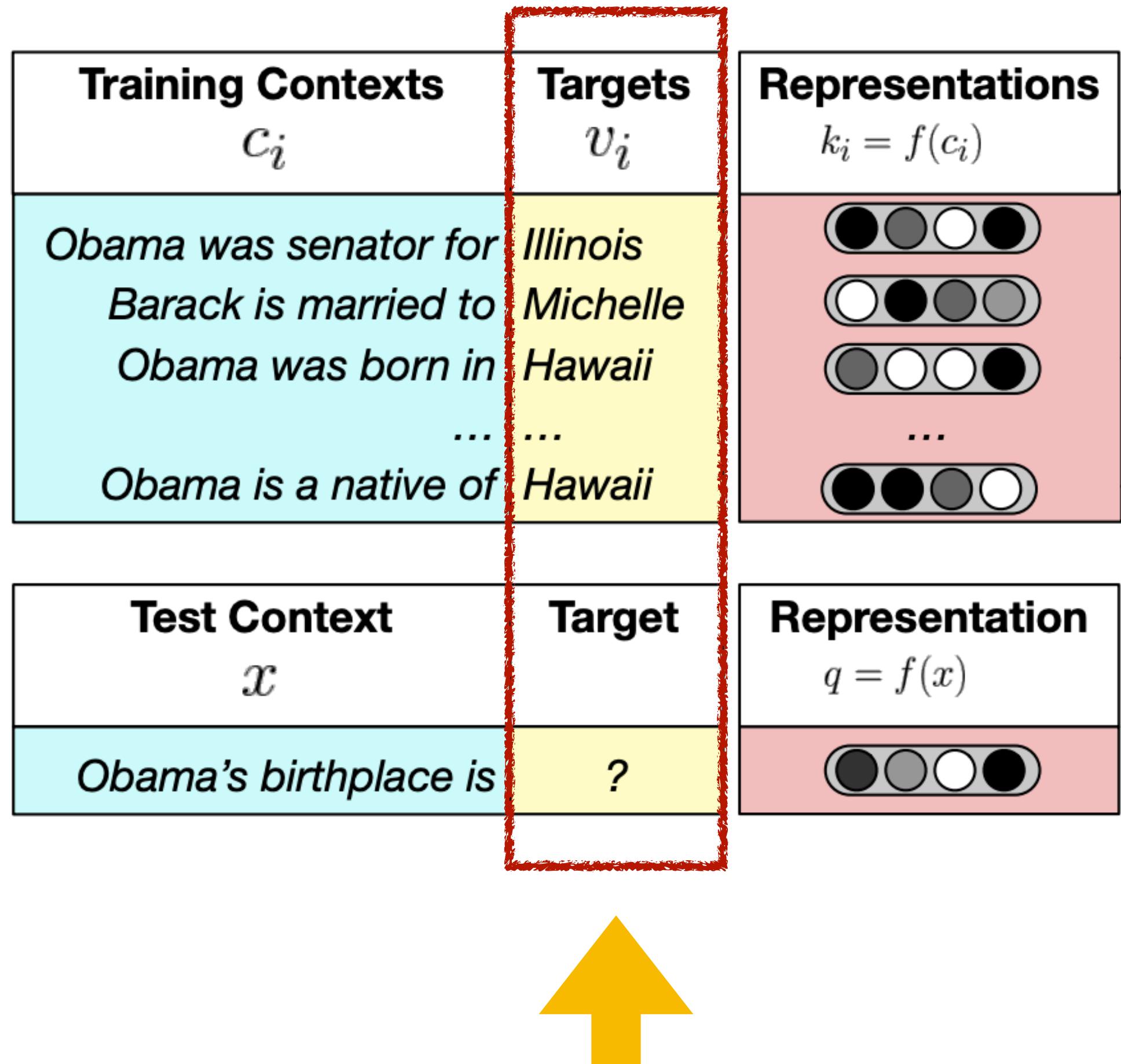
Test Context	Target	Representation
x		$q = f(x)$
<i>Obama's birthplace is</i>	?	

kNN-LM (Khandelwal et al. 2020)

Training Contexts	Targets	Representations
c_i	v_i	$k_i = f(c_i)$
<i>Obama was senator for</i>	<i>Illinois</i>	
<i>Barack is married to</i>	<i>Michelle</i>	
<i>Obama was born in</i>	<i>Hawaii</i>	
...
<i>Obama is a native of</i>	<i>Hawaii</i>	

Test Context	Target	Representation
x		$q = f(x)$
<i>Obama's birthplace is</i>	?	

kNN-LM (Khandelwal et al. 2020)



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

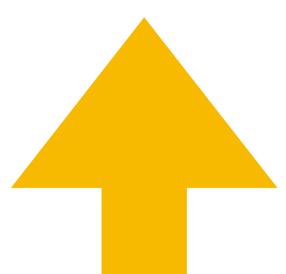
kNN-LM (Khandelwal et al. 2020)

Training Contexts	Targets	Representations
c_i	v_i	$k_i = f(c_i)$
<i>Obama was senator for</i>	<i>Illinois</i>	
<i>Barack is married to</i>	<i>Michelle</i>	
<i>Obama was born in</i>	<i>Hawaii</i>	
...
<i>Obama is a native of</i>	<i>Hawaii</i>	
Test Context	Target	Representation
x		$q = f(x)$
<i>Obama's birthplace is</i>	?	

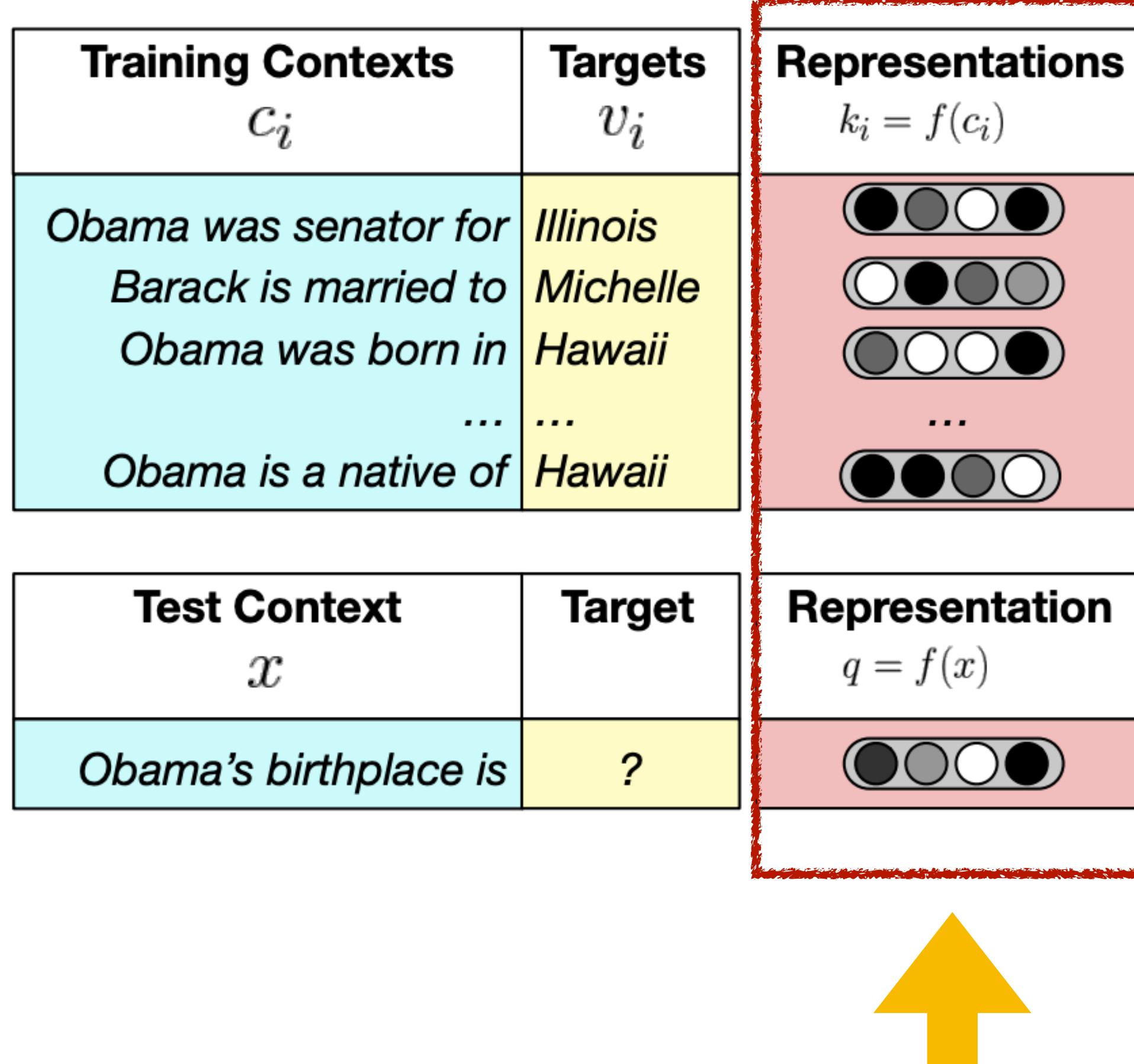
Which tokens in a datastore are close to the next token?

=

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



kNN-LM (Khandelwal et al. 2020)



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

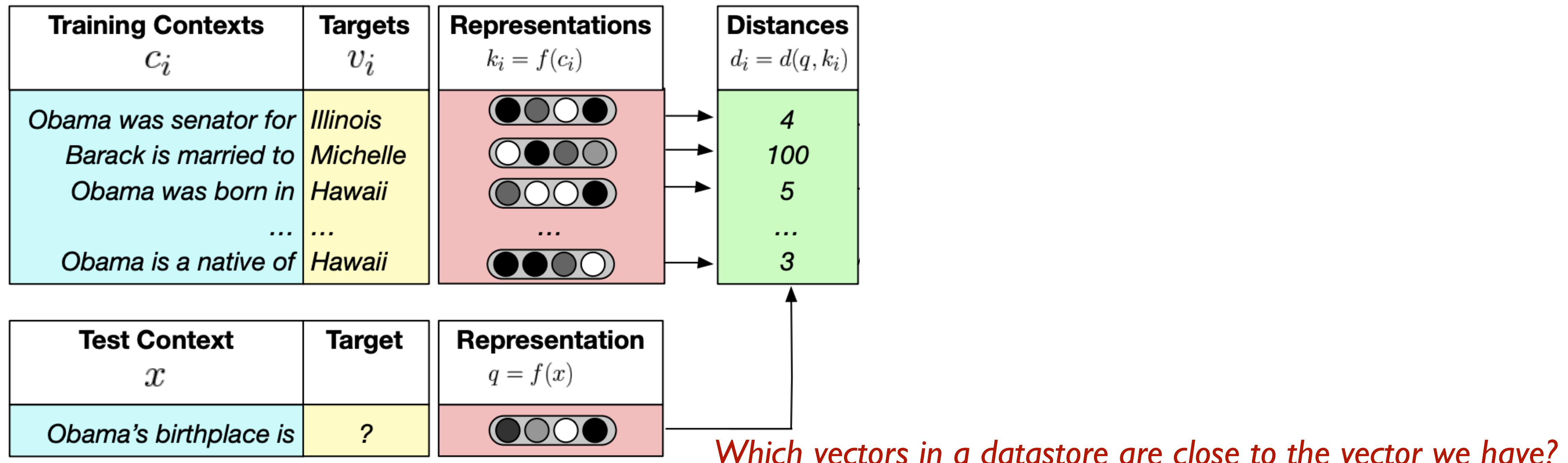
=

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

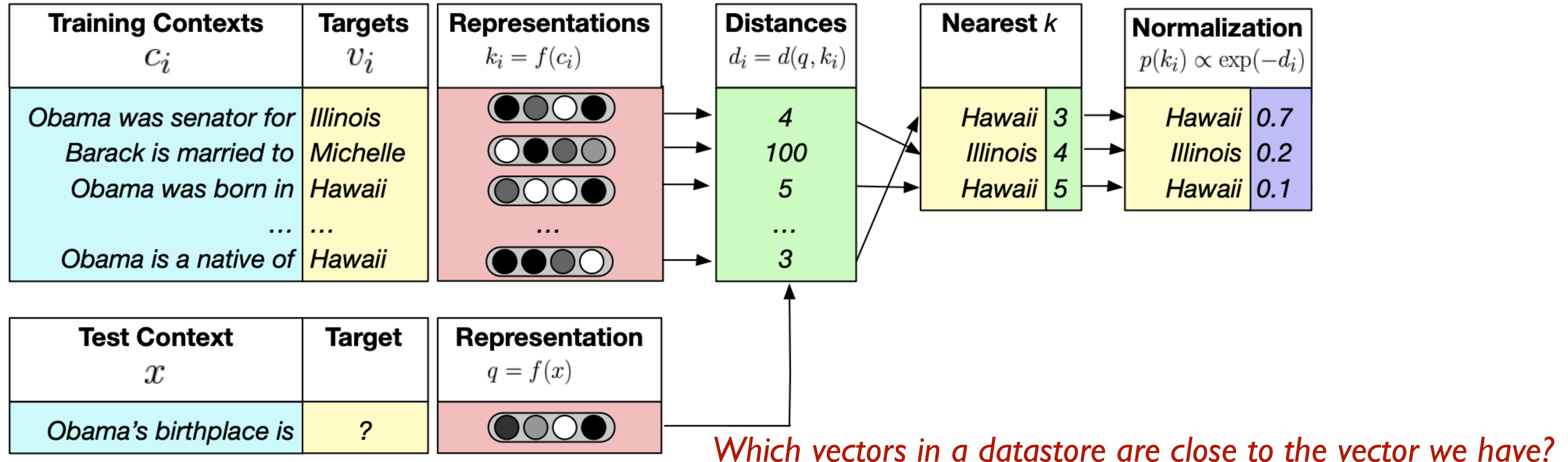
=

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

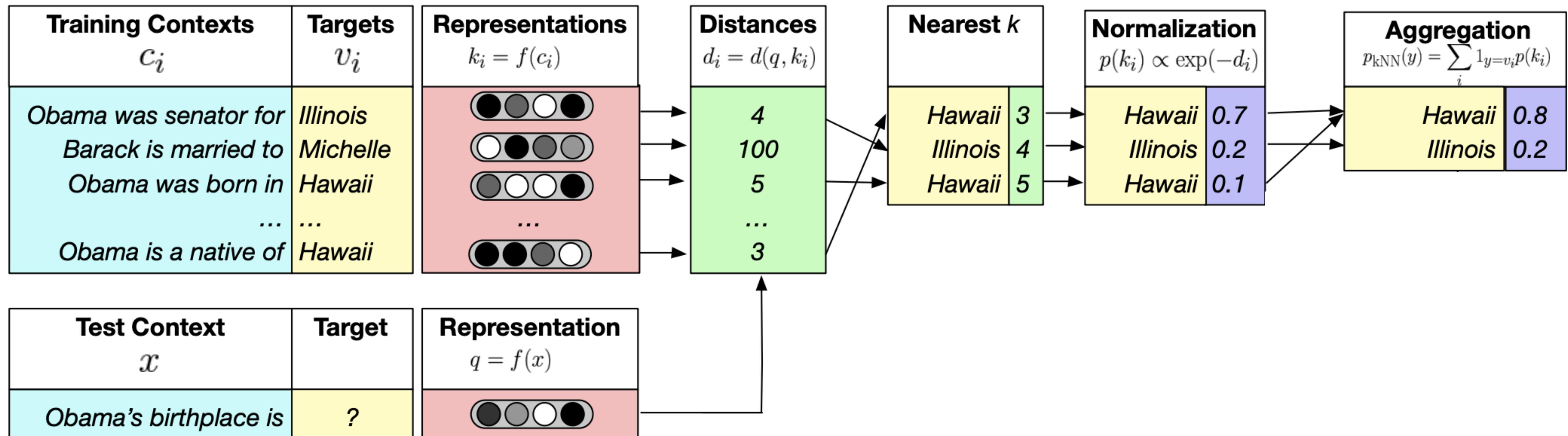
kNN-LM (Khandelwal et al. 2020)



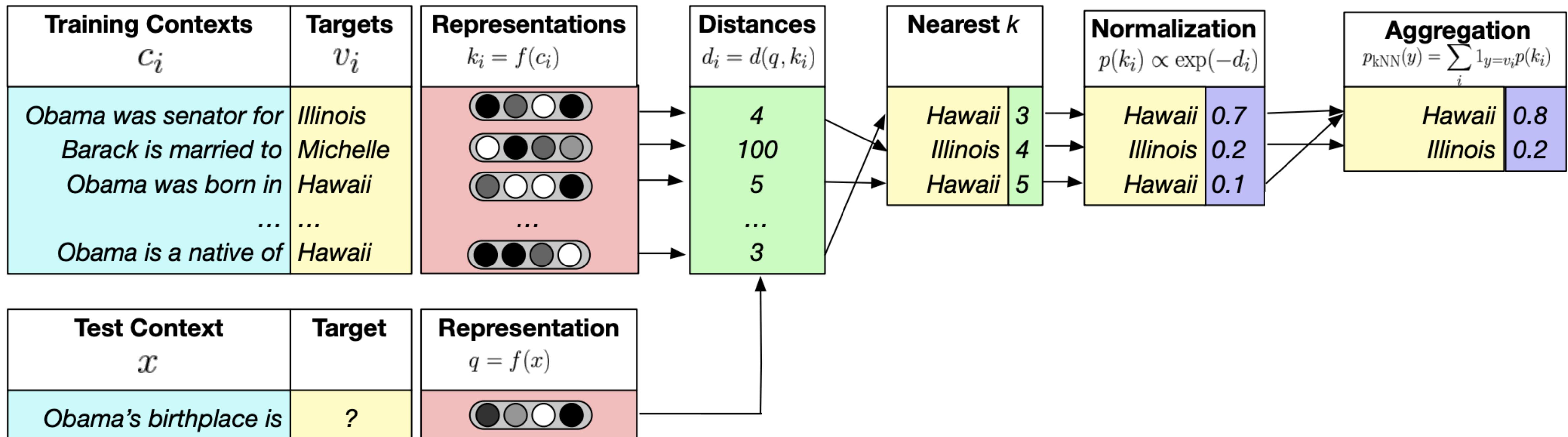
kNN-LM (Khandelwal et al. 2020)



kNN-LM (Khandelwal et al. 2020)

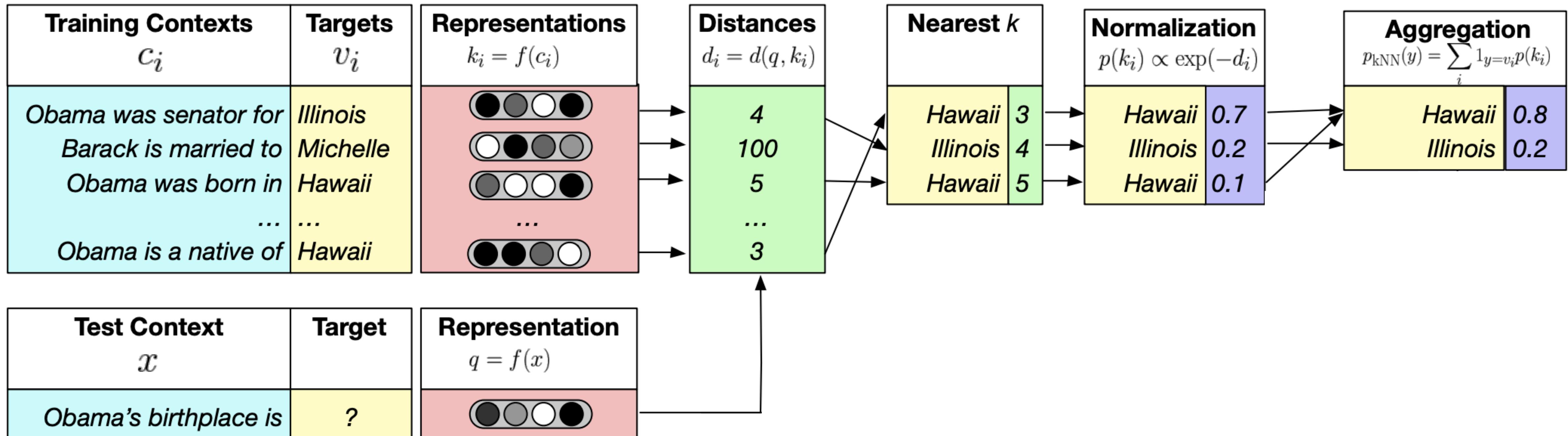


kNN-LM (Khandelwal et al. 2020)



$$P_{kNN}(y|x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v=y] \text{sim}(k, x)$$

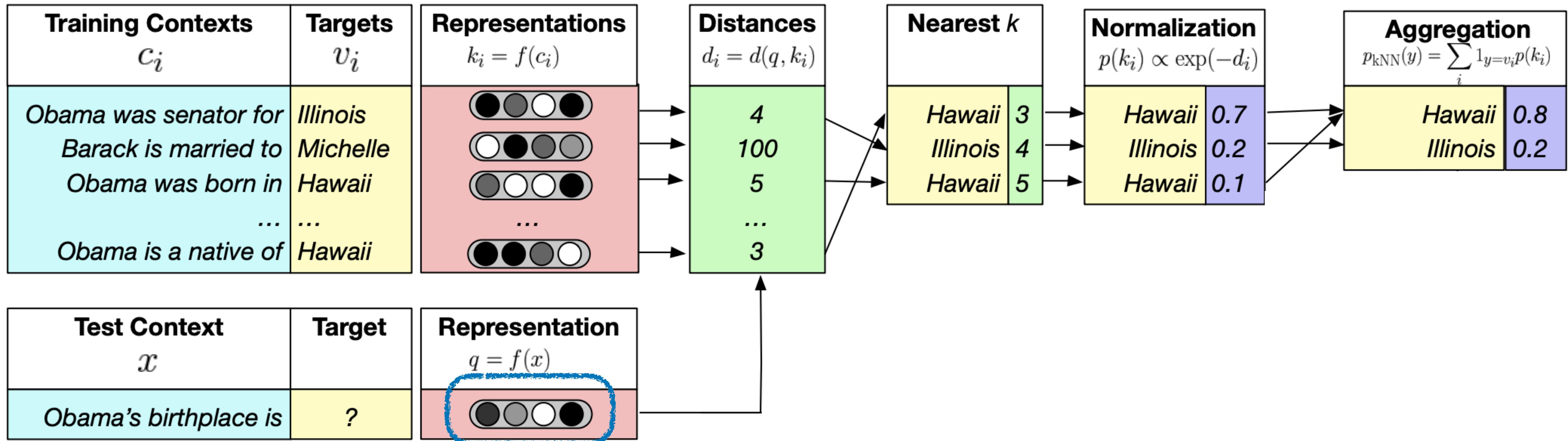
kNN-LM (Khandelwal et al. 2020)



$$P_{kNN}(y | x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \text{sim}(k, x)$$

$$\text{sim}(k, x) = \exp(-d(\text{Enc}(k), \text{Enc}(x)))$$

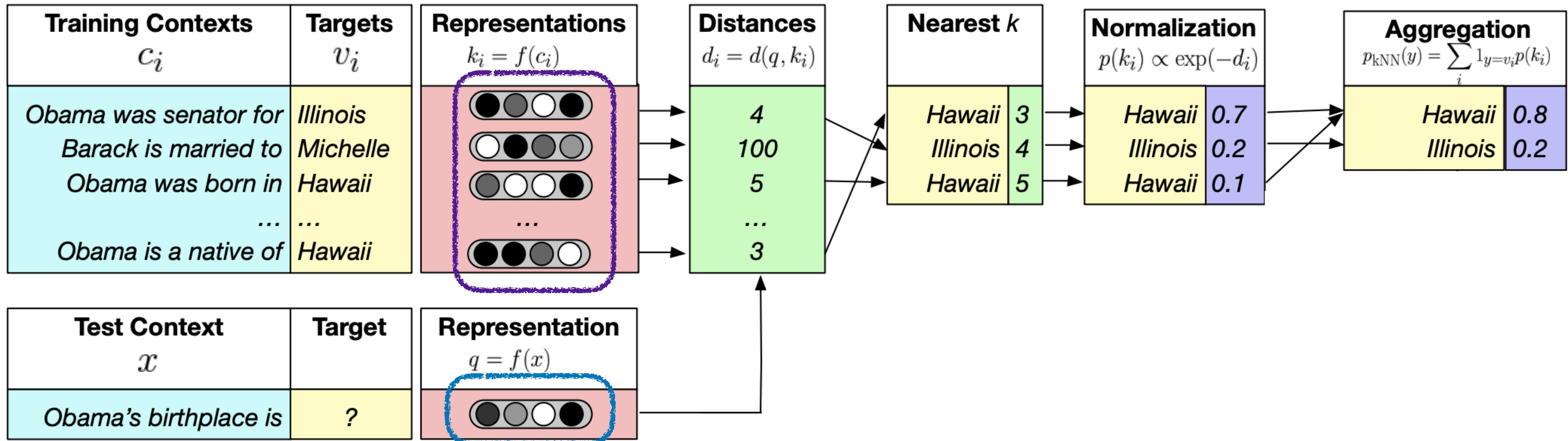
kNN-LM (Khandelwal et al. 2020)



$$P_{kNN}(y | x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \text{sim}(k, x)$$

$$\text{sim}(k, x) = \exp(-d(\text{Enc}(k), \underline{\text{Enc}(x)}))$$

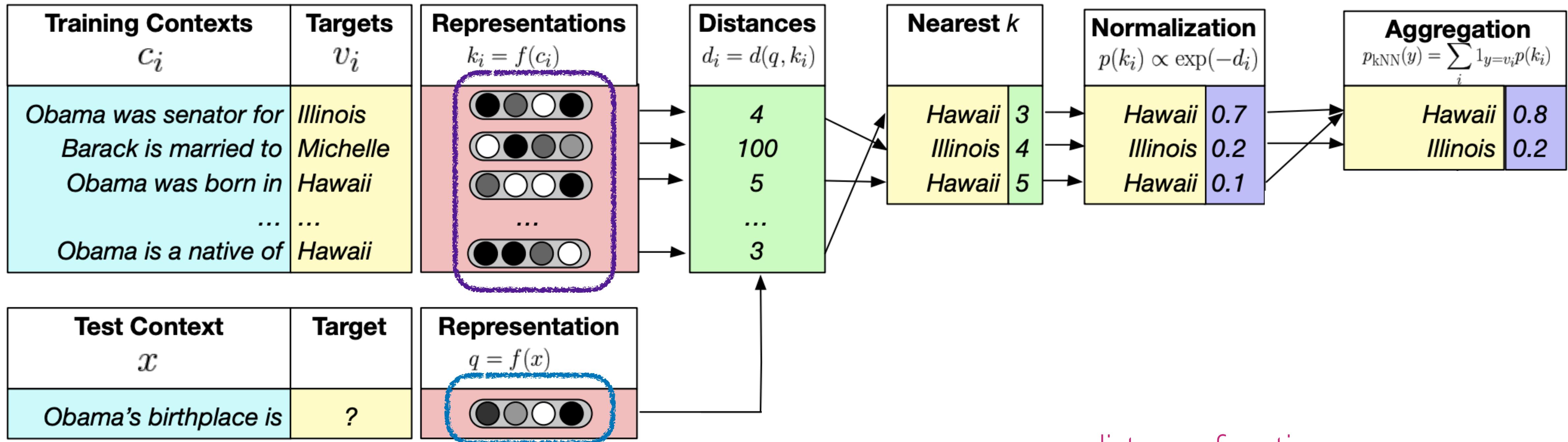
kNN-LM (Khandelwal et al. 2020)



$$P_{kNN}(y | x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \text{sim}(k, x)$$

$$\text{sim}(k, x) = \exp \left(-d(\underline{\text{Enc}}(k), \underline{\text{Enc}}(x)) \right)$$

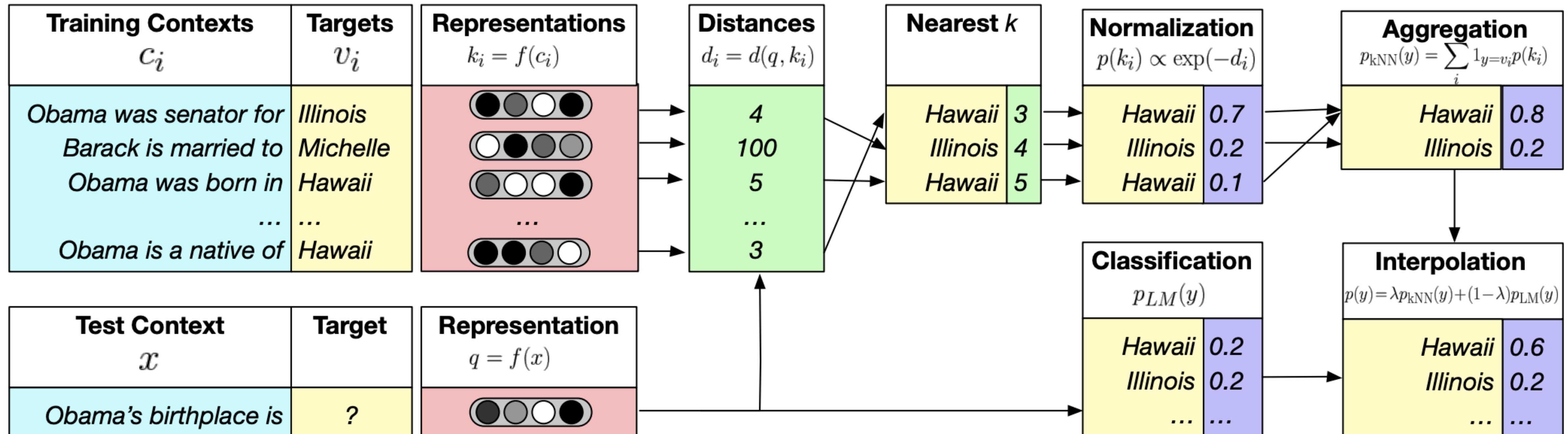
kNN-LM (Khandelwal et al. 2020)



$$P_{kNN}(y | x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \text{sim}(k, x)$$

$$\text{sim}(k, x) = \exp \left(-d(\underline{\text{Enc}(k)}, \underline{\text{Enc}(x)}) \right)$$

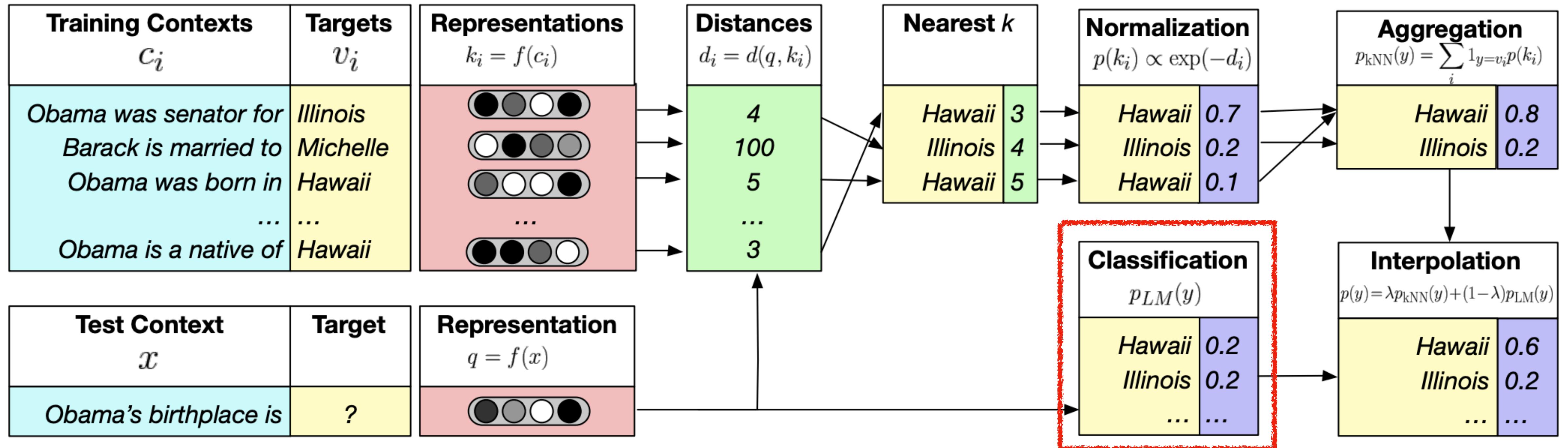
kNN-LM (Khandelwal et al. 2020)



$$P_{kNN-LM}(y | x) = (1 - \lambda)P_{LM}(y | x) + \lambda P_{kNN}(y | x)$$

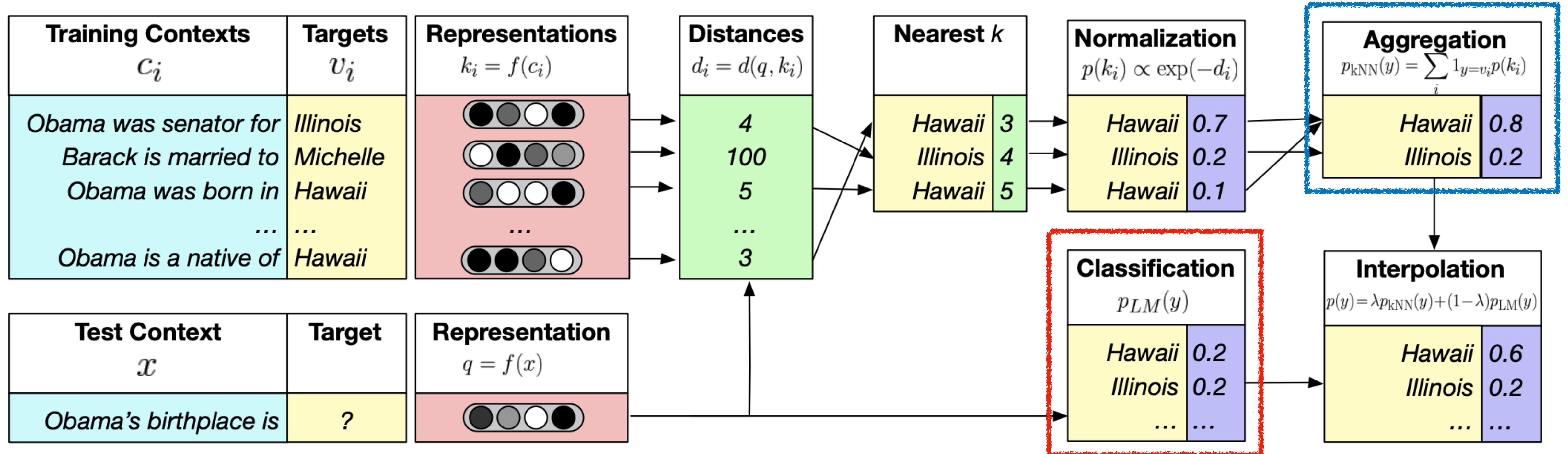
Later work, e.g., NPM (Min et al. 2023) removed interpolation (more in Section 4)

kNN-LM (Khandelwal et al. 2020)



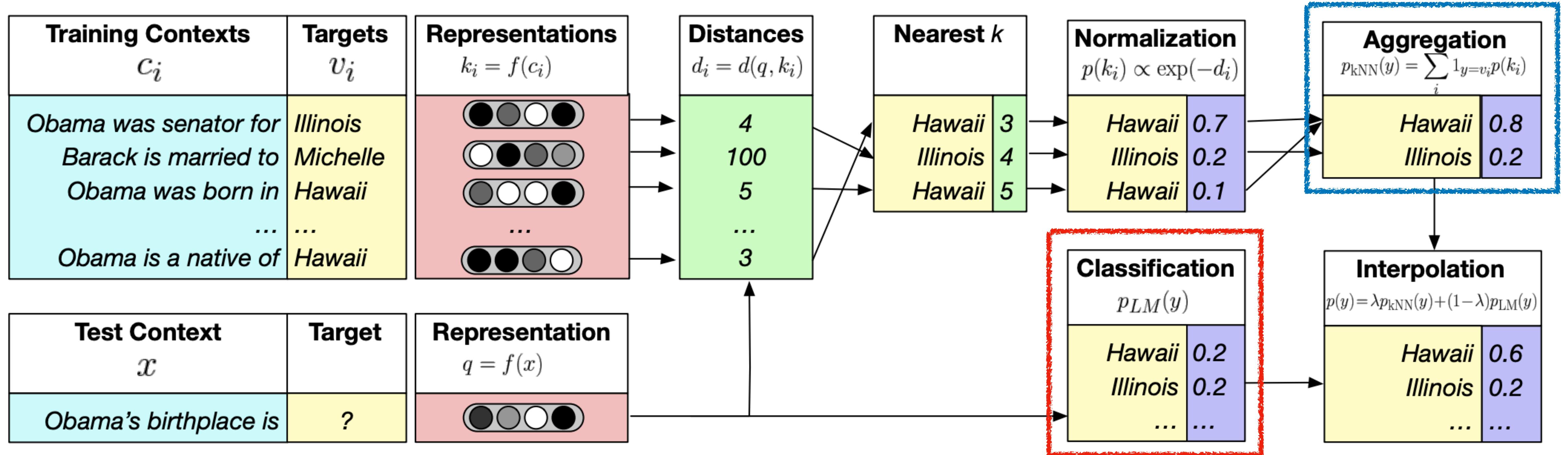
Later work, e.g., NPM (Min et al. 2023) removed interpolation (more in Section 4)

kNN-LM (Khandelwal et al. 2020)



Later work, e.g., NPM (Min et al. 2023) removed interpolation (more in Section 4)

kNN-LM (Khandelwal et al. 2020)



λ : hyperparameter

$$P_{kNN-LM}(y | x) = (1 - \lambda) \underline{P_{LM}(y | x)} + \lambda \underline{P_{kNN}(y | x)}$$

Later work, e.g., NPM (Min et al. 2023) removed interpolation (more in Section 4)

kNN-LM - why?

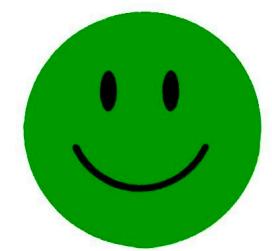
Training contexts	Targets
<i>10/10, would buy this item delivered broken. Very</i>	<i>cheap</i>
<i>To check the version of PyTorch, you can use</i>	<i>torch</i>
<i>You are permitted to bring a</i>	<i>torch</i>
<i>A group of infections ... one of the</i>	<i>torch</i>

kNN-LM - why?

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

kNN-LM - why?

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



kNN-LM - why?

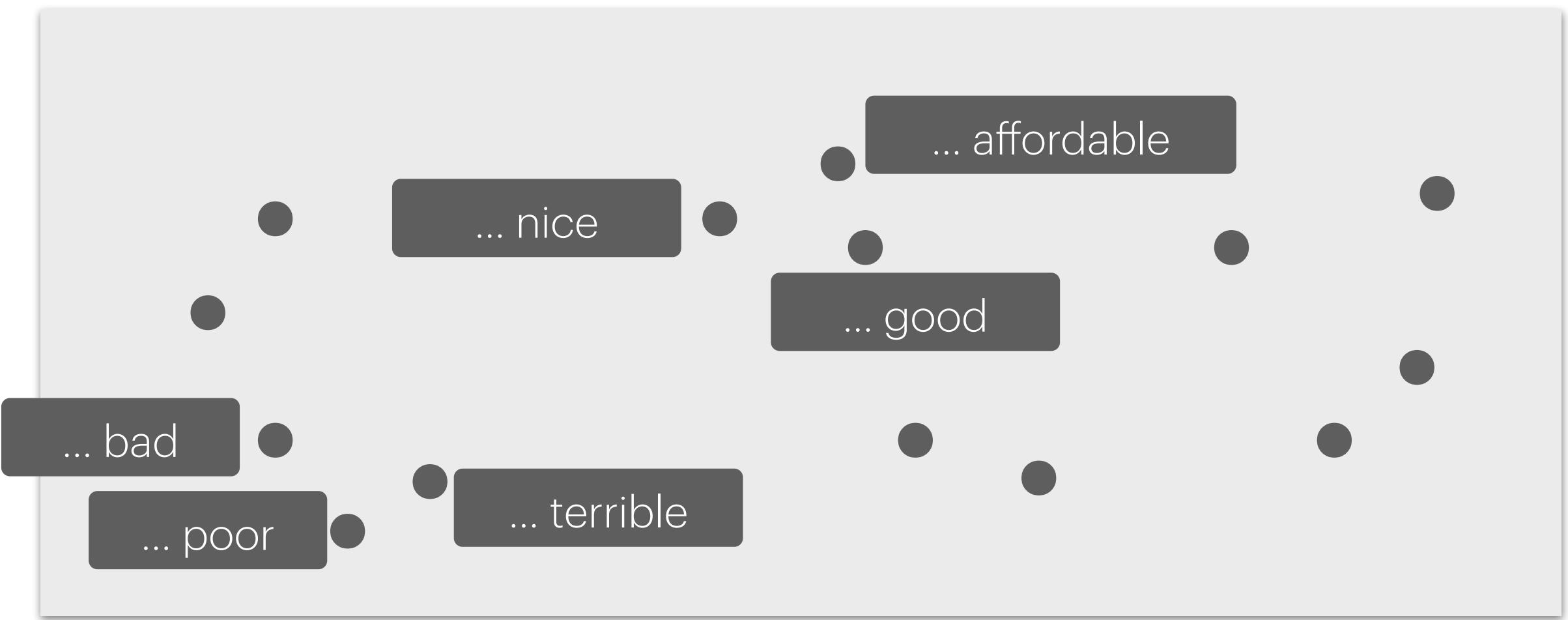
Training contexts	Targets
<i>10/10, would buy this item delivered broken. Very</i>	<i>cheap</i>
<i>To check the version of PyTorch, you can use</i>	<i>cheap</i>
<i>You are permitted to bring a</i>	<i>torch</i>
<i>A group of infections ... one of the</i>	<i>torch</i>



kNN-LM - why?

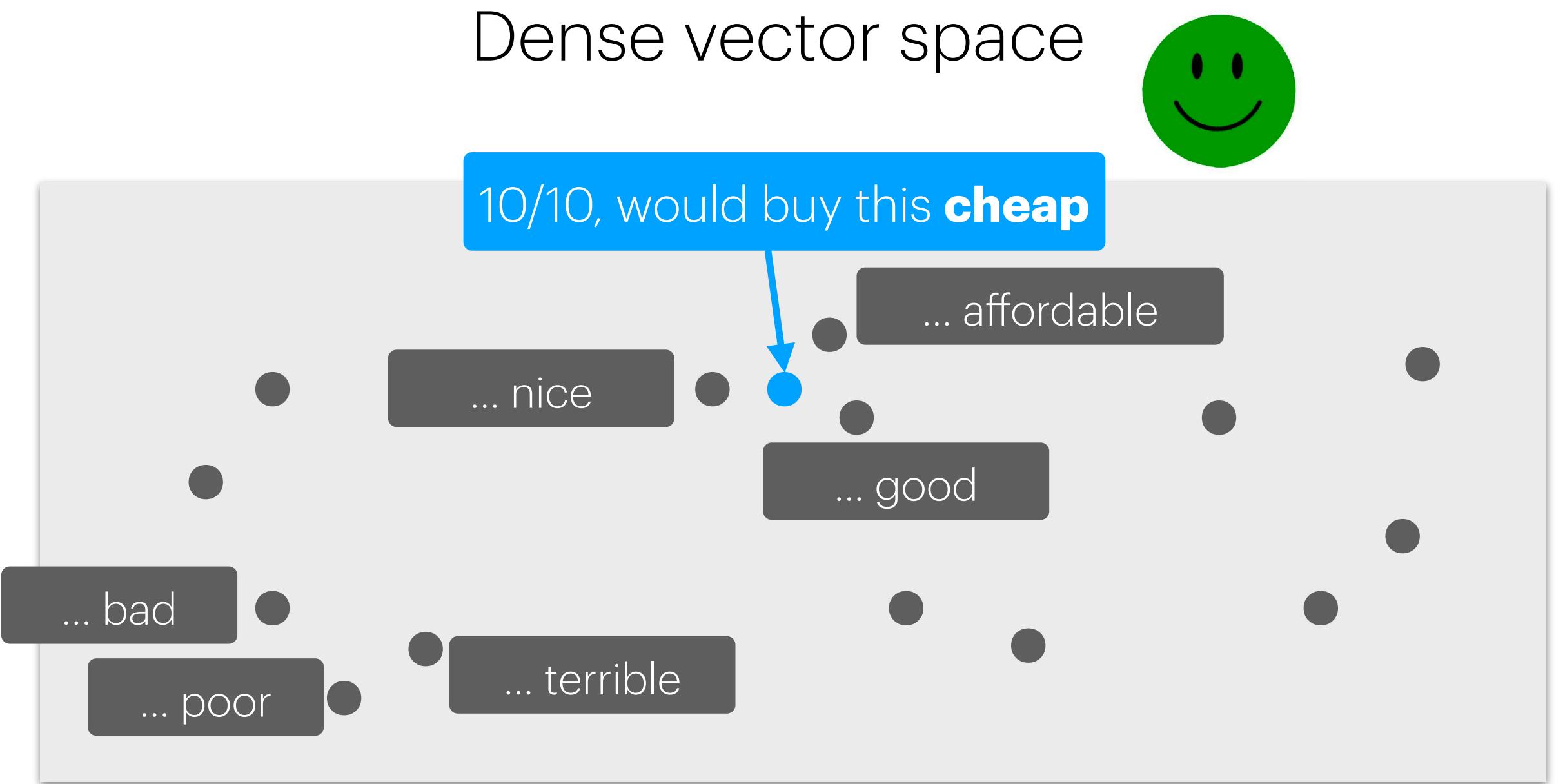
Dense vector space

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



kNN-LM - why?

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



kNN-LM - why?

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



Dense vector space



10/10, would buy this **cheap**

... nice

... affordable

... good

... bad

... poor

... terrible

Item delivered broken. Very **cheap**

kNN-LM - why?

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

kNN-LM - why?

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



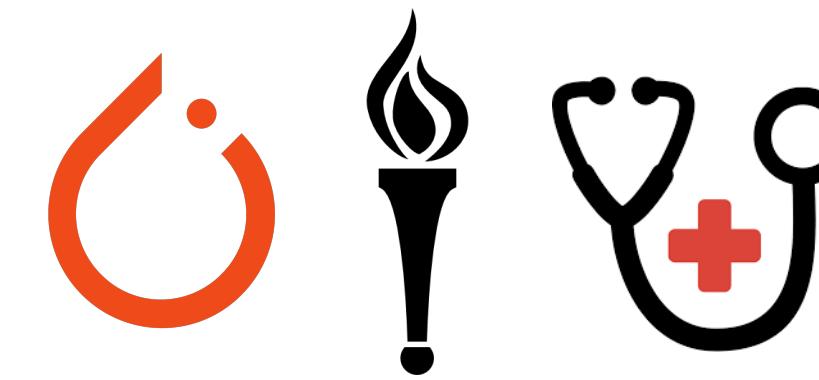
kNN-LM - why?

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



kNN-LM - why?

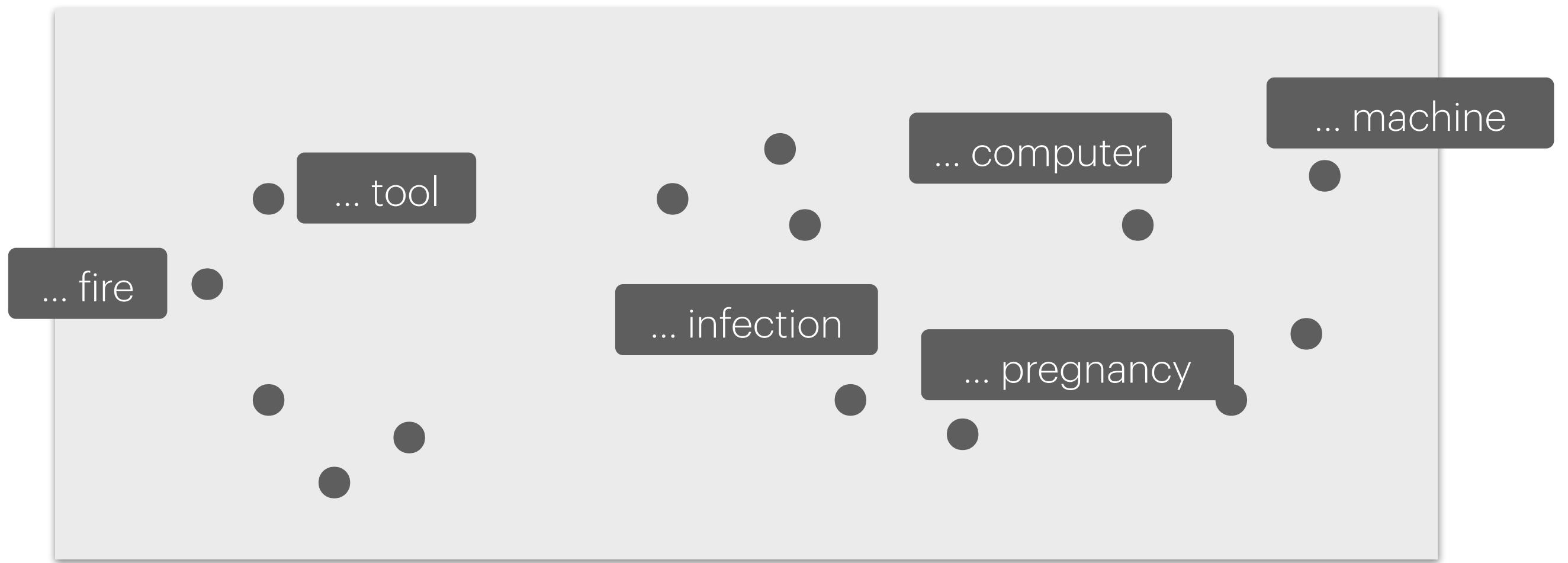
Training contexts	Targets
<i>10/10, would buy this item delivered broken. Very</i>	<i>cheap</i>
To check the version of PyTorch, you can use	<i>cheap</i>
You are permitted to bring a	<i>torch</i>
A group of infections ... one of the	<i>torch</i>
	<i>torch</i>



kNN-LM - why?

Dense vector space

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

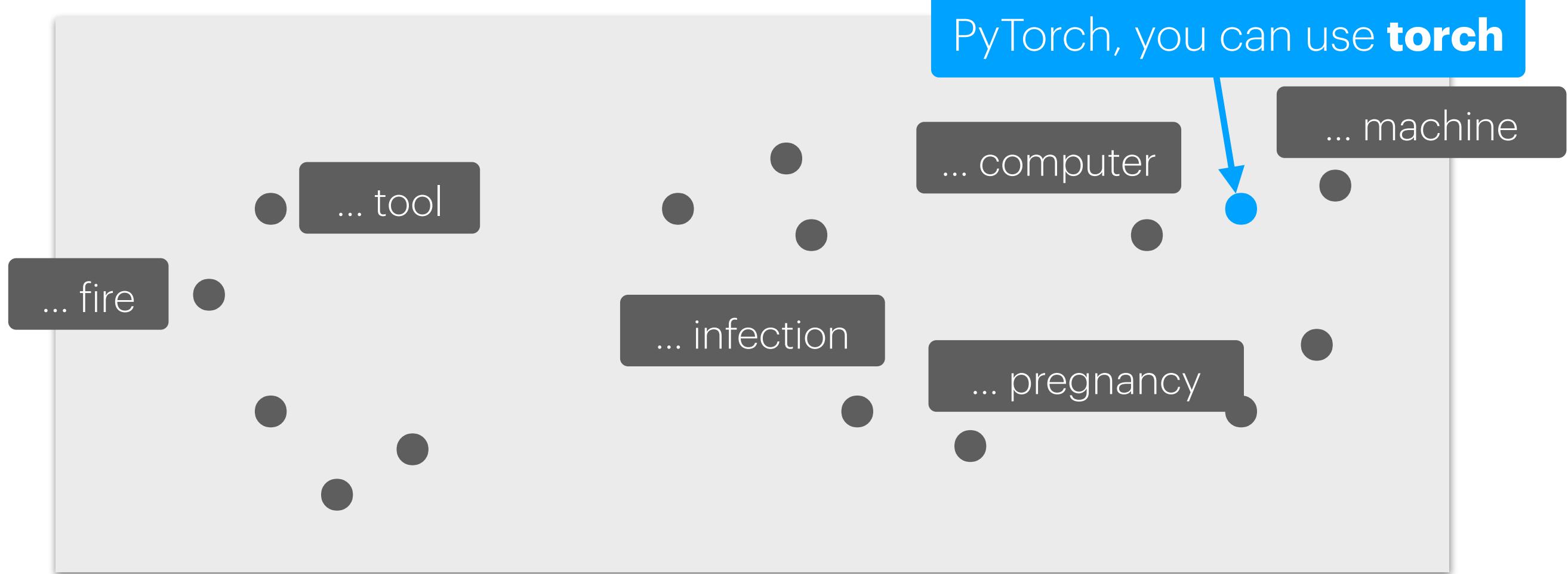


kNN-LM - why?

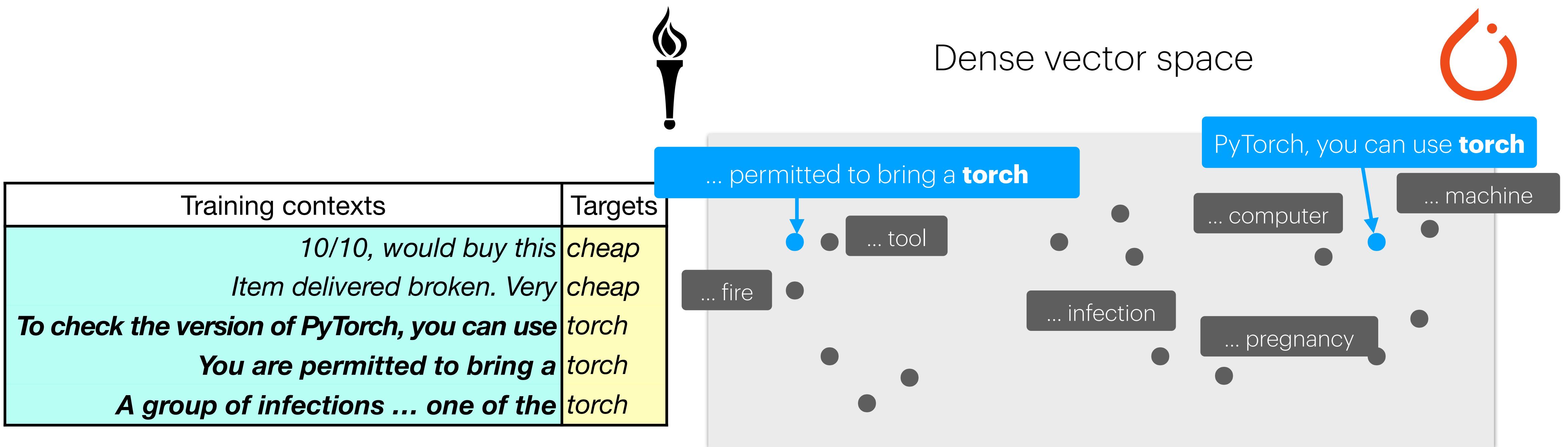
Dense vector space



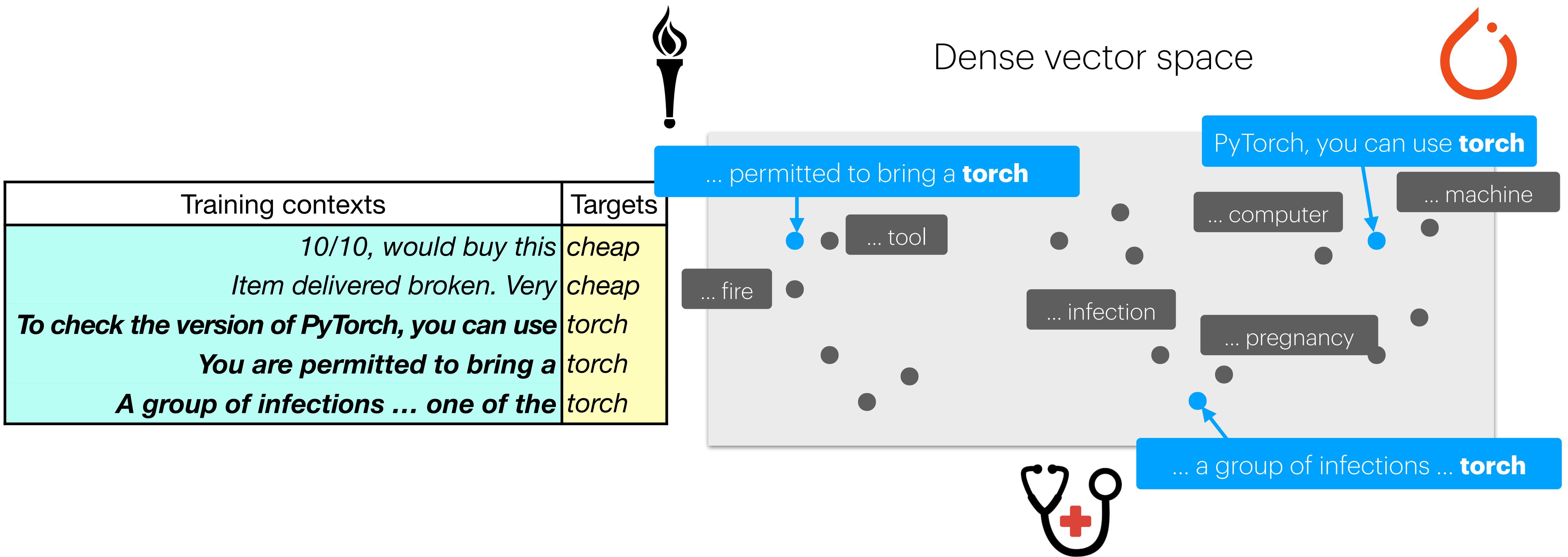
Training contexts	Targets
<i>10/10, would buy this item delivered broken. Very</i>	<i>cheap</i>
To check the version of PyTorch, you can use	<i>cheap</i>
You are permitted to bring a	<i>torch</i>
A group of infections ... one of the	<i>torch</i>
	<i>torch</i>



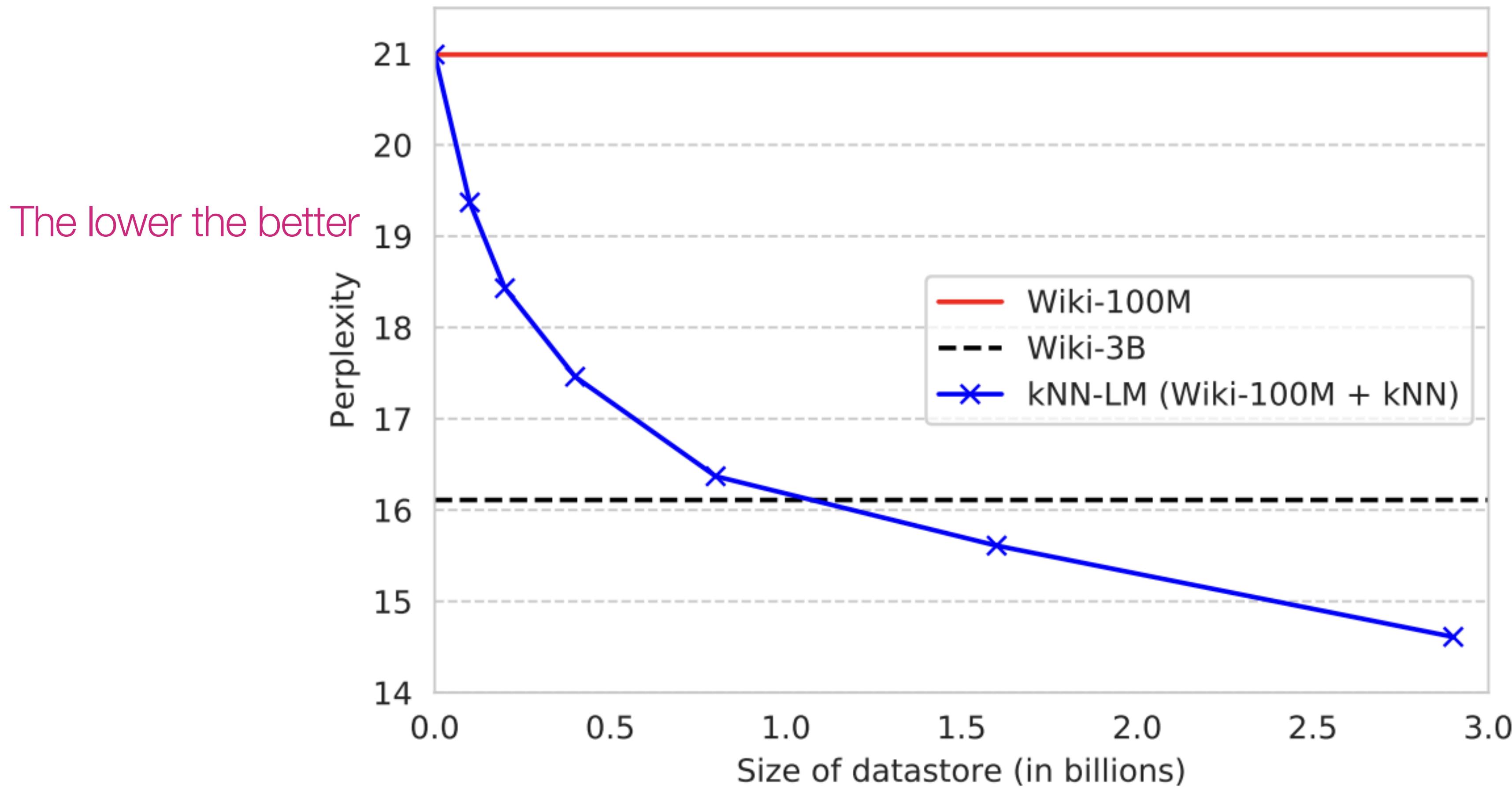
kNN-LM - why?



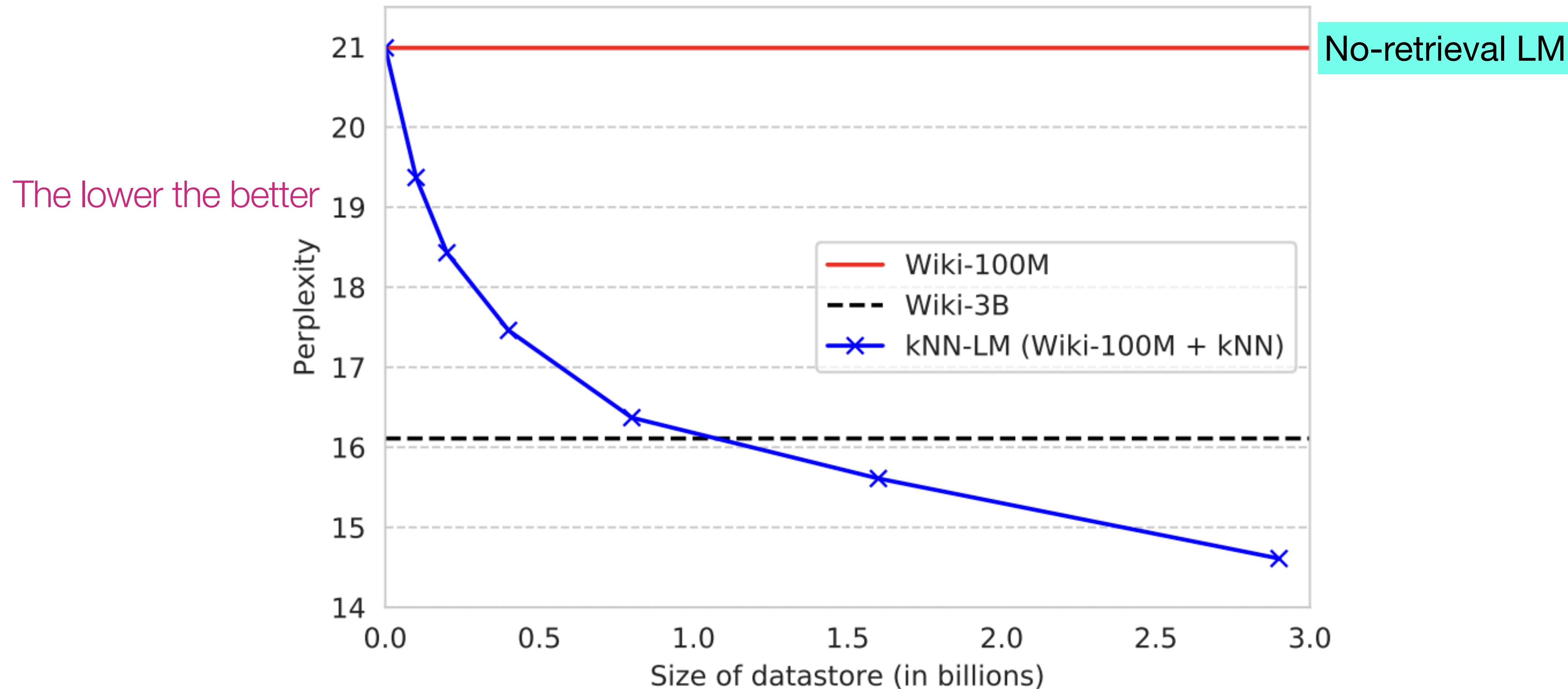
kNN-LM - why?



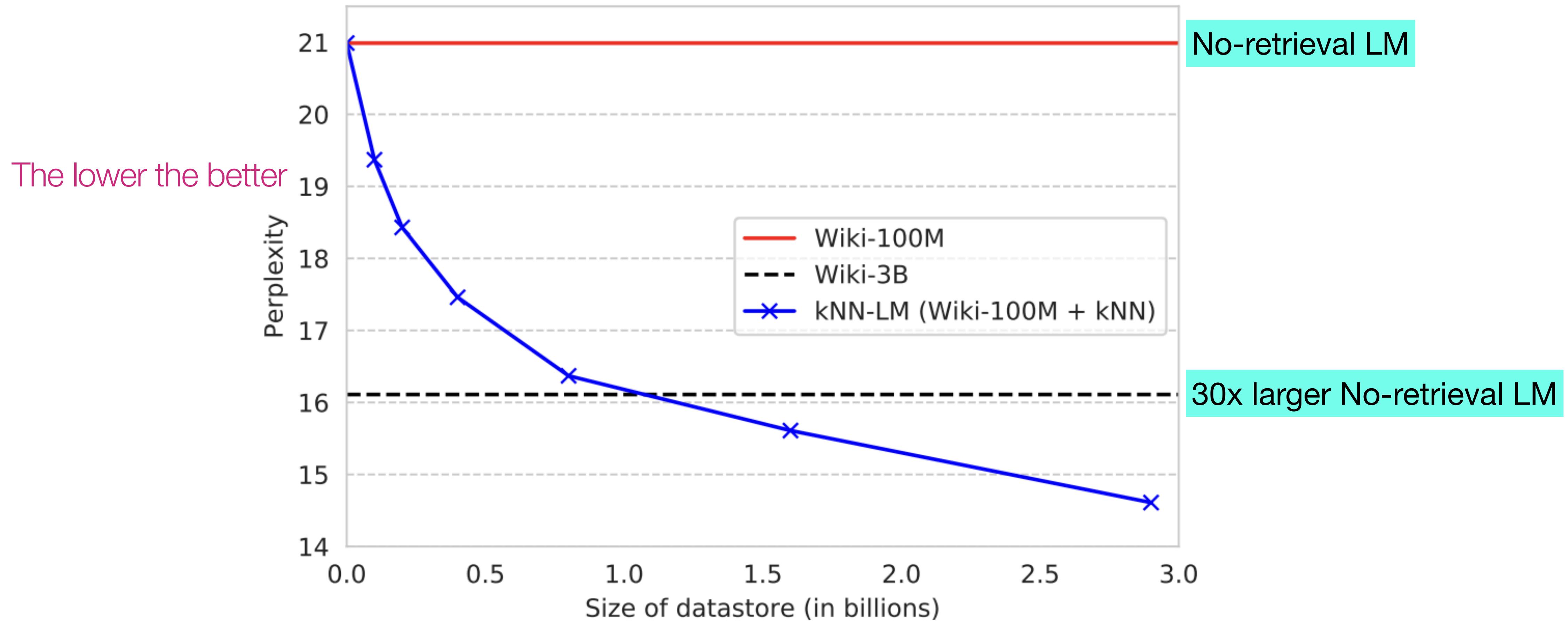
kNN-LM - results



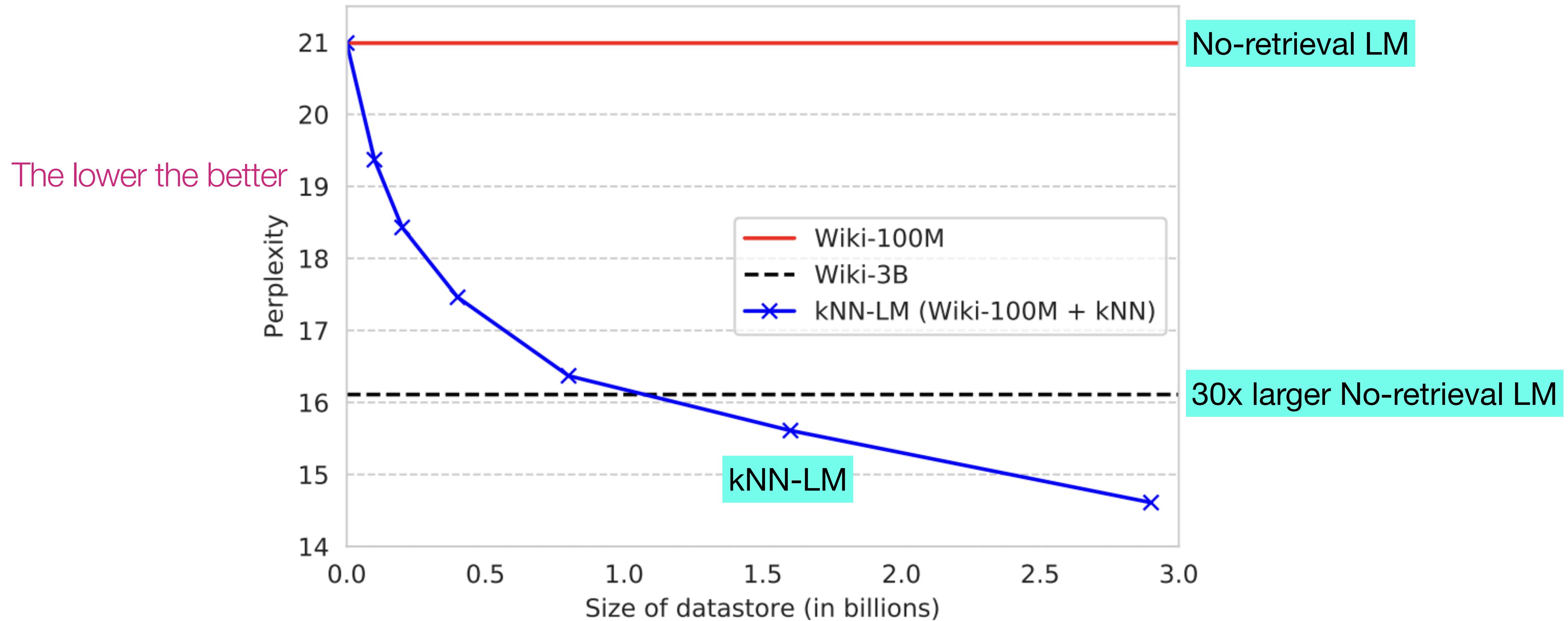
kNN-LM - results



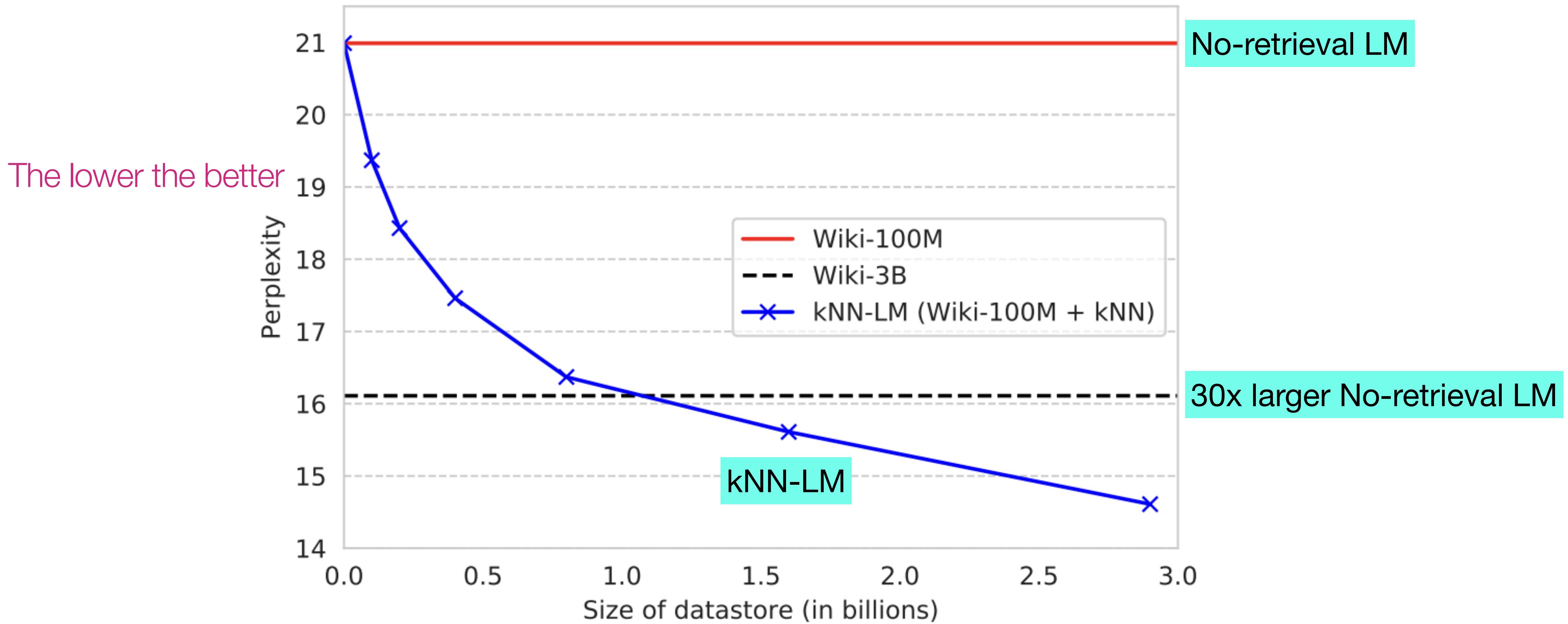
kNN-LM - results



kNN-LM - results

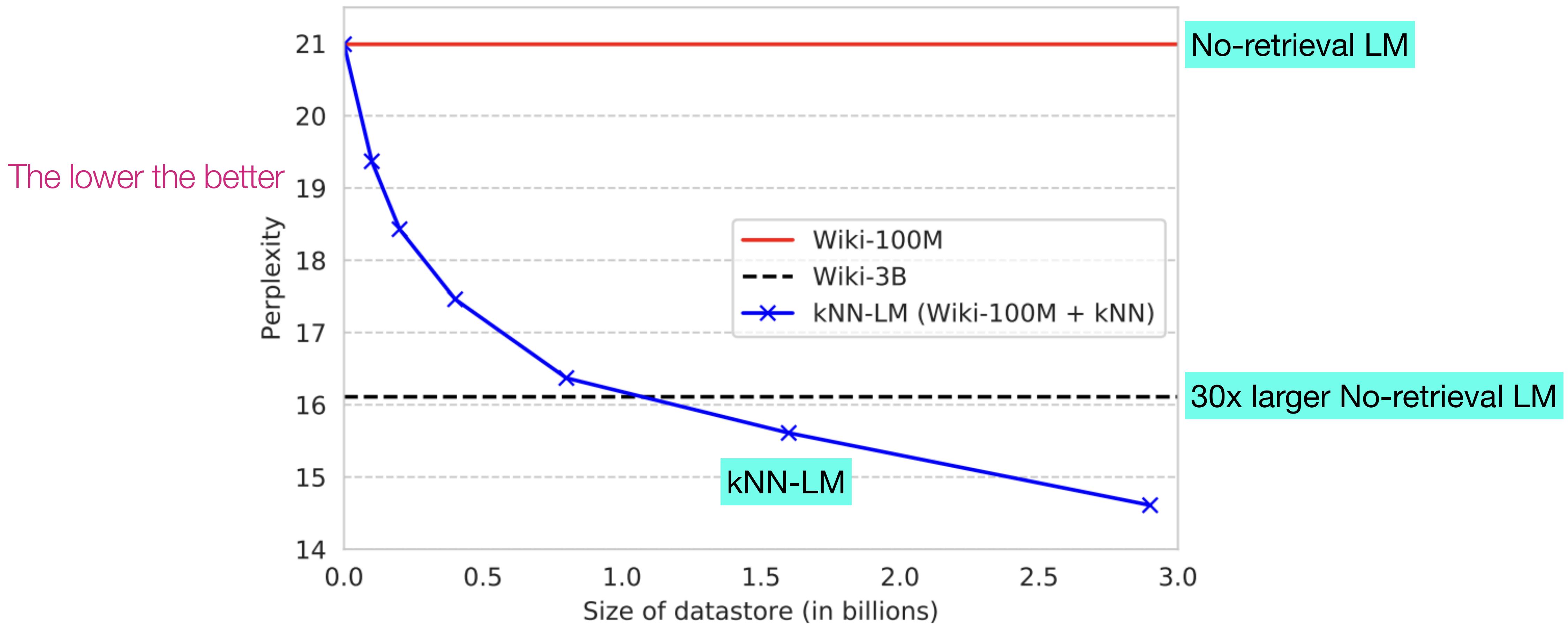


kNN-LM - results



Outperforms no-retrieval LM

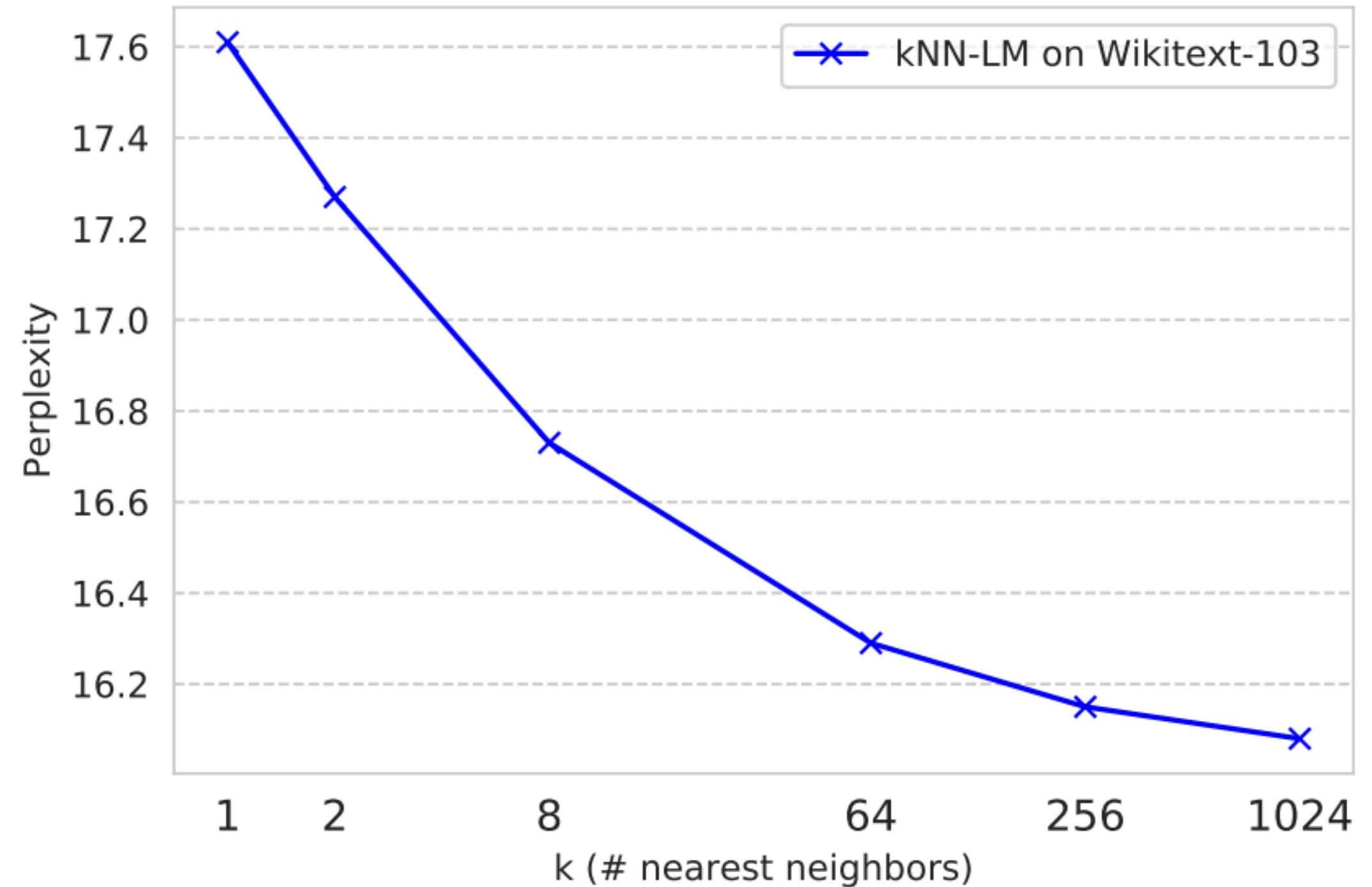
kNN-LM - results



Outperforms no-retrieval LM

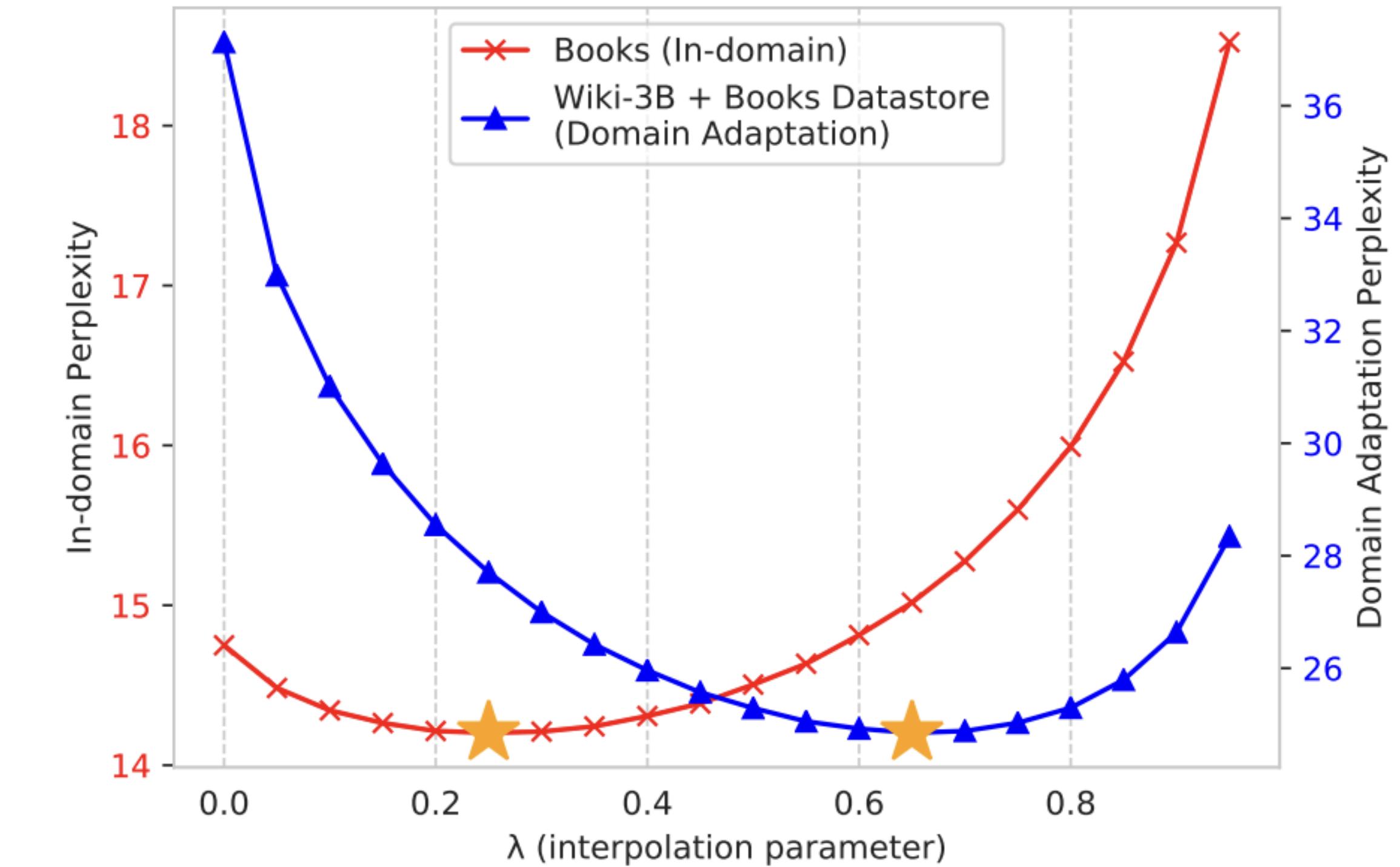
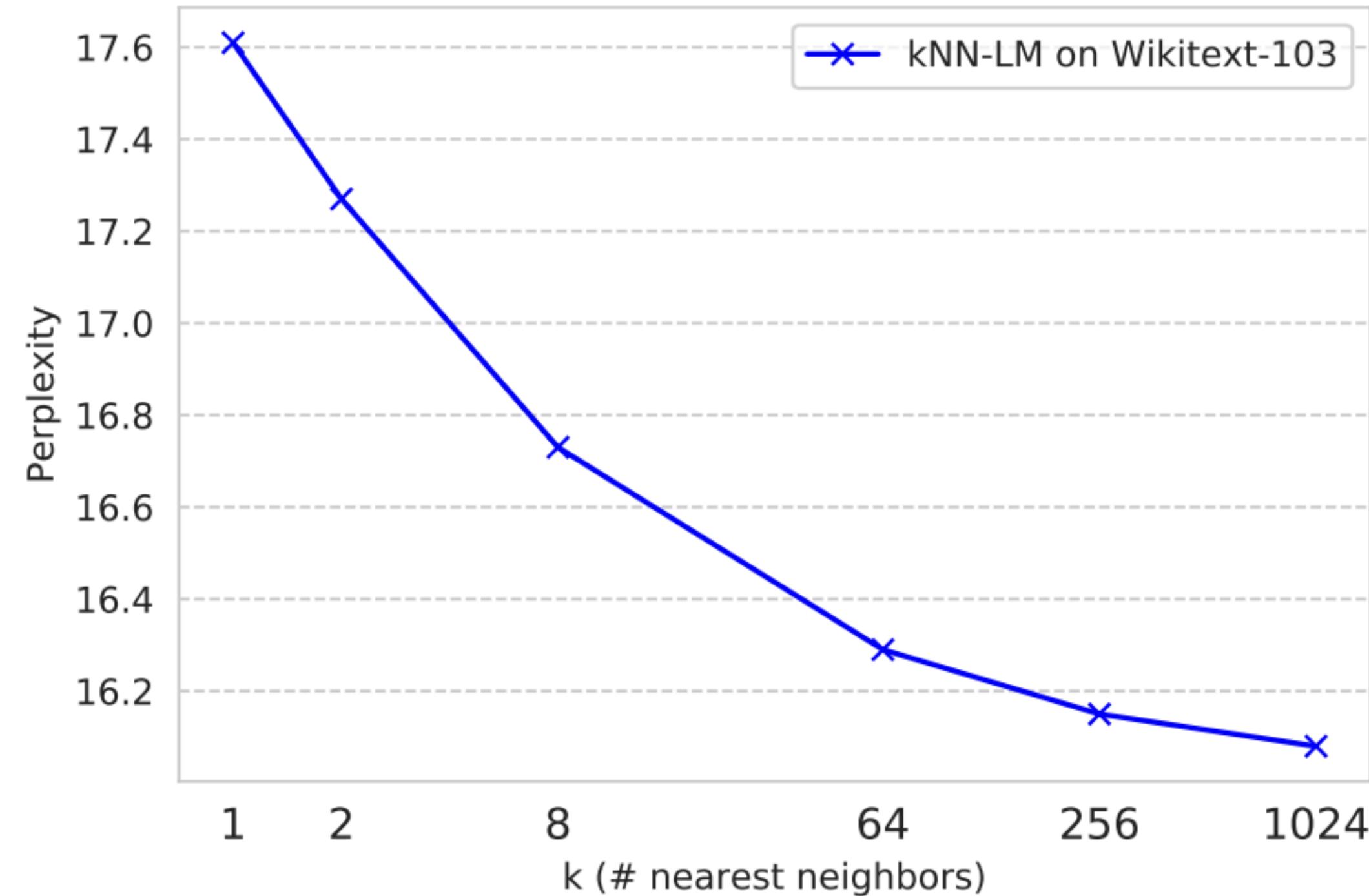
Better with bigger datastore

kNN-LM - results



Better with
bigger k

kNN-LM - results

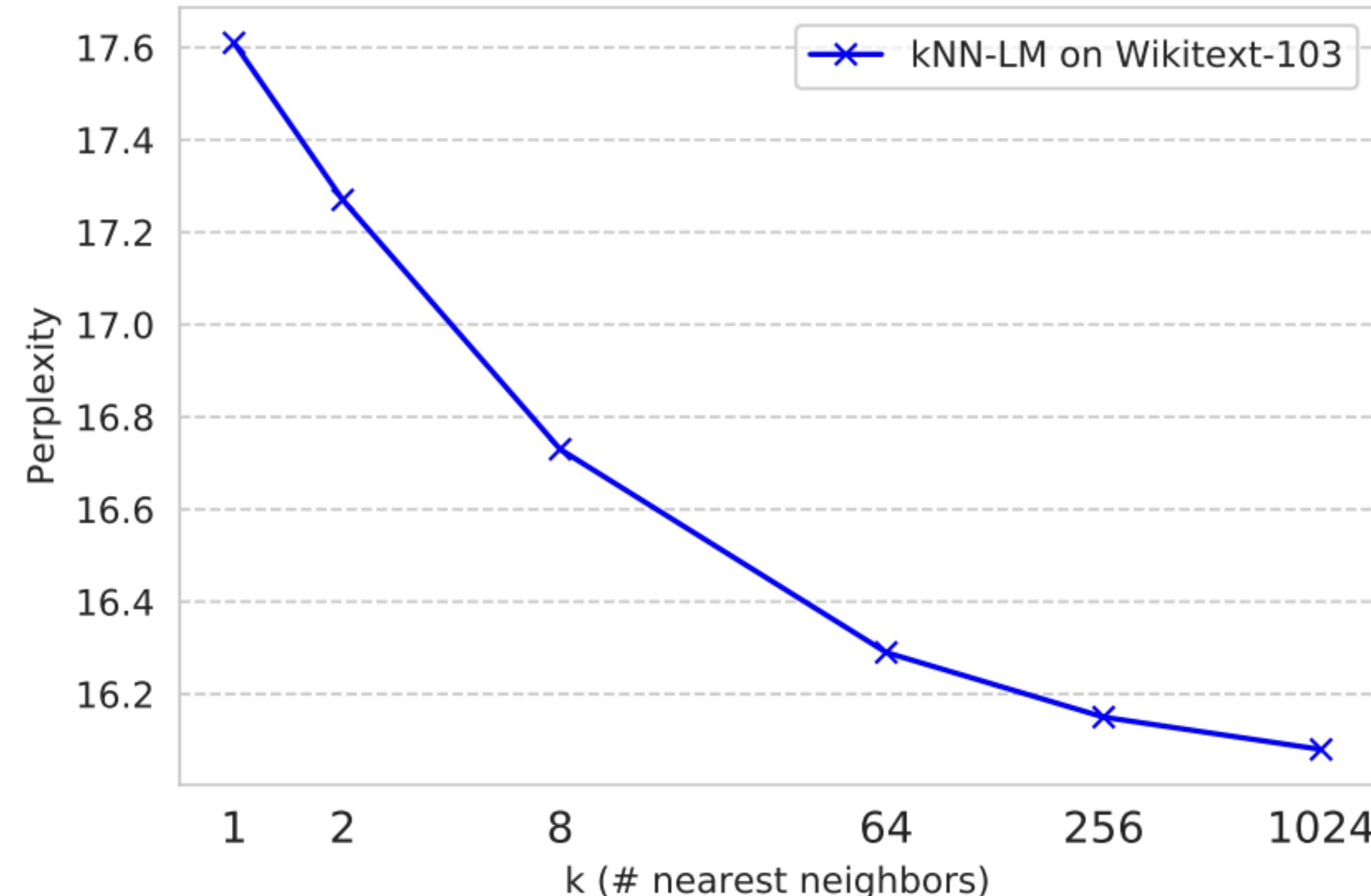


Better with
bigger k

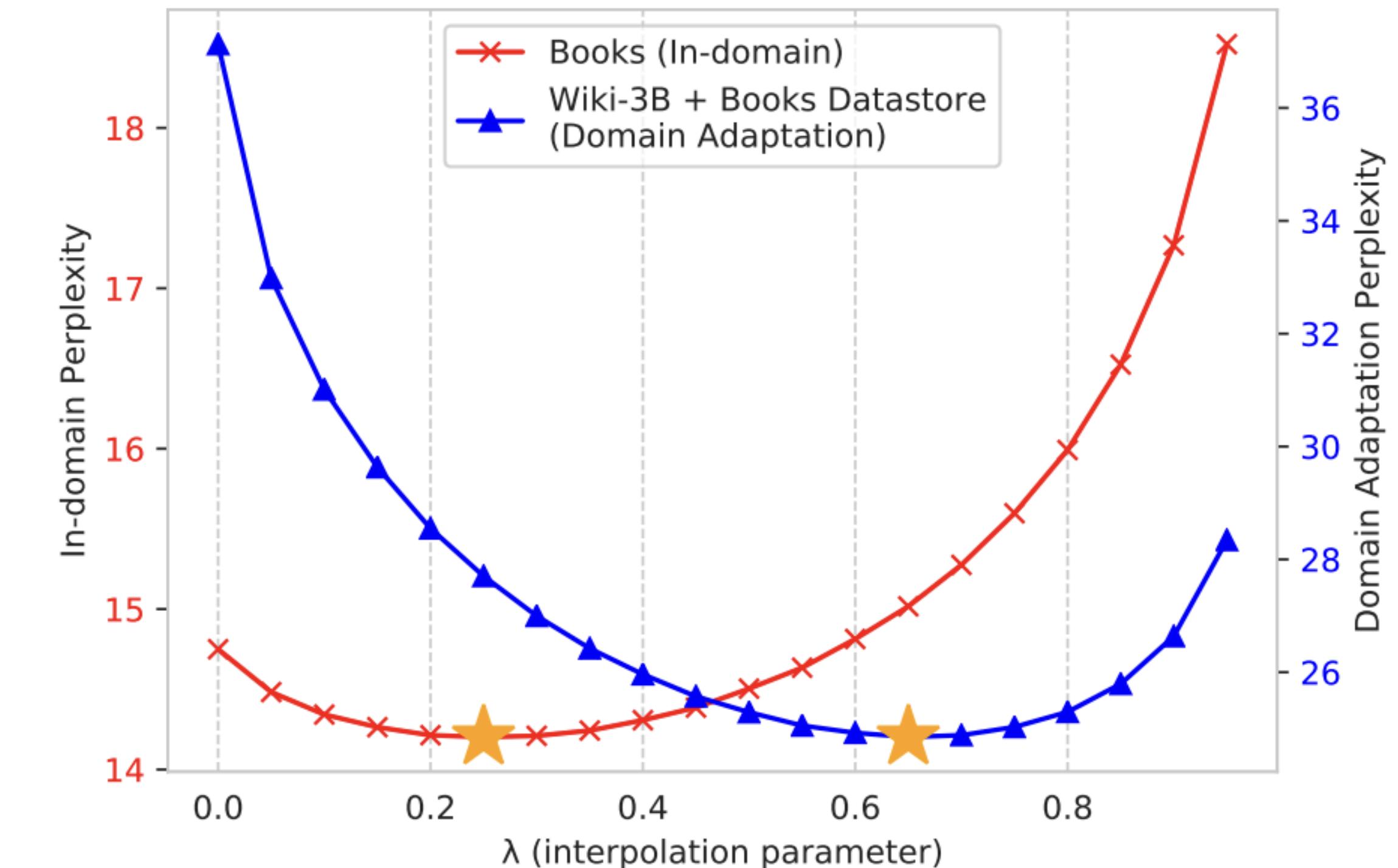
Helps more
out-of-domain

kNN-L

Can use in-domain datastore
even if parameters were not trained in-domain



Better with
bigger k



Helps more
out-of-domain

kNN-LM (Khandelwal et al. 2020)

What to retrieve?

- Chunks
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

kNN-LM (Khandelwal et al. 2020)

What to retrieve?

- Chunks
- **Tokens** ✓
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

kNN-LM (Khandelwal et al. 2020)

What to retrieve?

- Chunks
- **Tokens** ✓
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- **Output layer** ✓

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- Every token

kNN-LM (Khandelwal et al. 2020)

What to retrieve?

- Chunks
- **Tokens** ✓
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- **Output layer** ✓

When to retrieve?

- Once
- Every n tokens ($n > 1$)
- **Every token** ✓

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

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

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

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)

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)



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

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

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)



Datastore is expensive in space: given the same data, # text chunks vs. # tokens
No cross attention between input and retrieval results

Extensions

	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

Extensions

	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

It's fixed! Can we do adaptively?

Adaptive retrieval for efficiency

Adaptive retrieval of
text chunks
(following retrieve-in-context)

Adaptive retrieval of
tokens
(following kNN-LM)

Adaptive retrieval of chunks

- *Judge necessity*

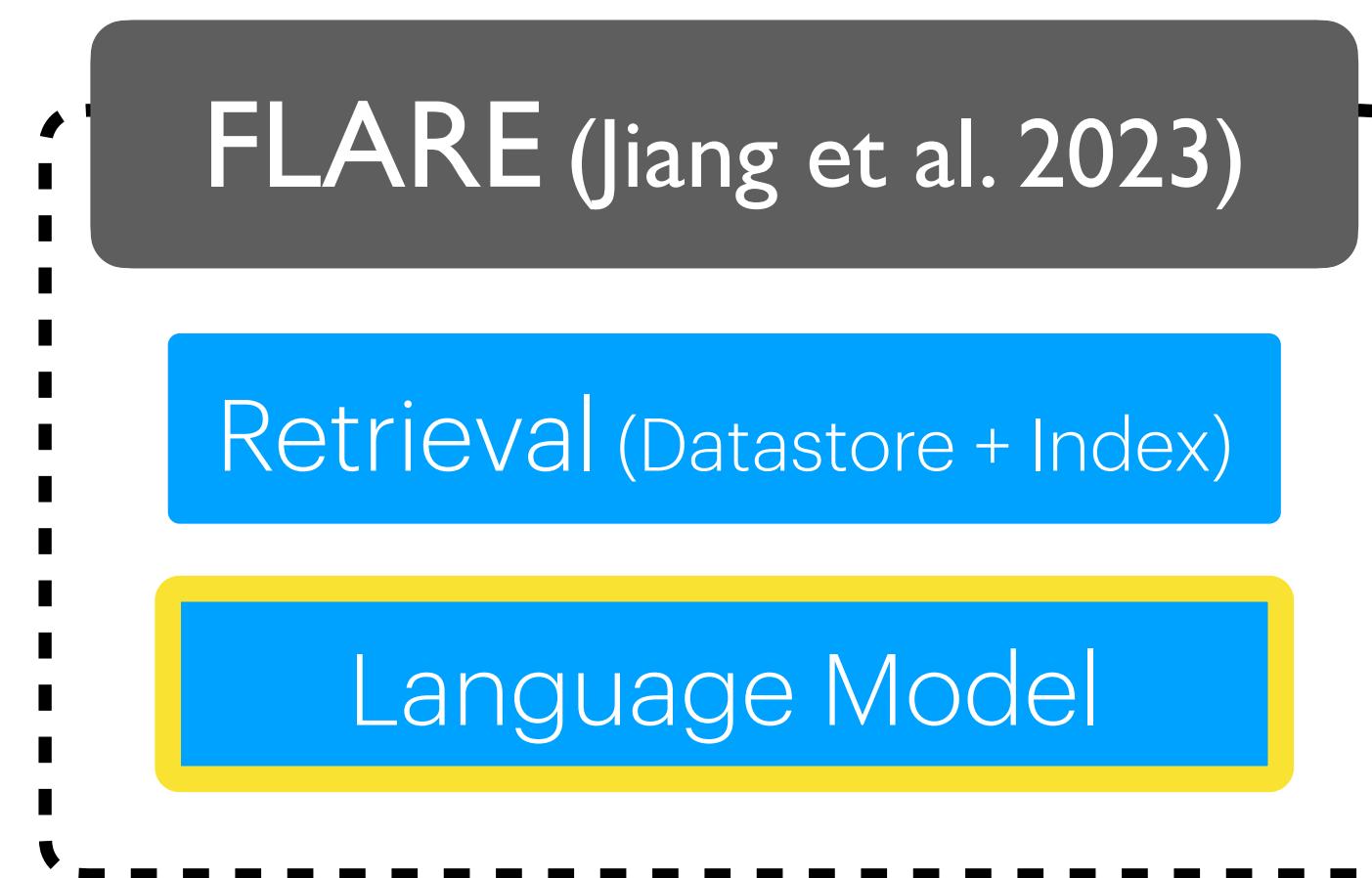
Input: Generate a summary about Joe Biden.



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.

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

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



I am confident!

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

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



I am confident!

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

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.

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

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.



Unsure...

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

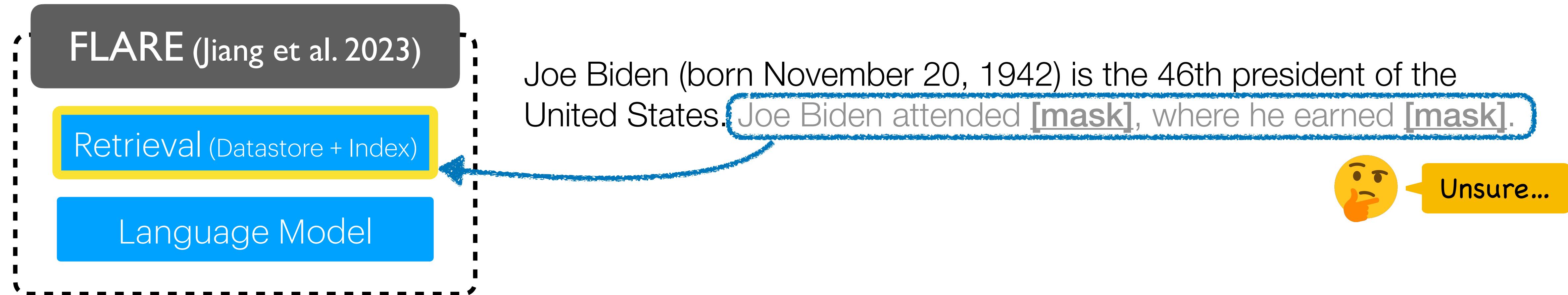
Joe Biden (born November 20, 1942) is the 46th president of the United States. Joe Biden attended [mask], where he earned [mask].



Adaptive retrieval of chunks

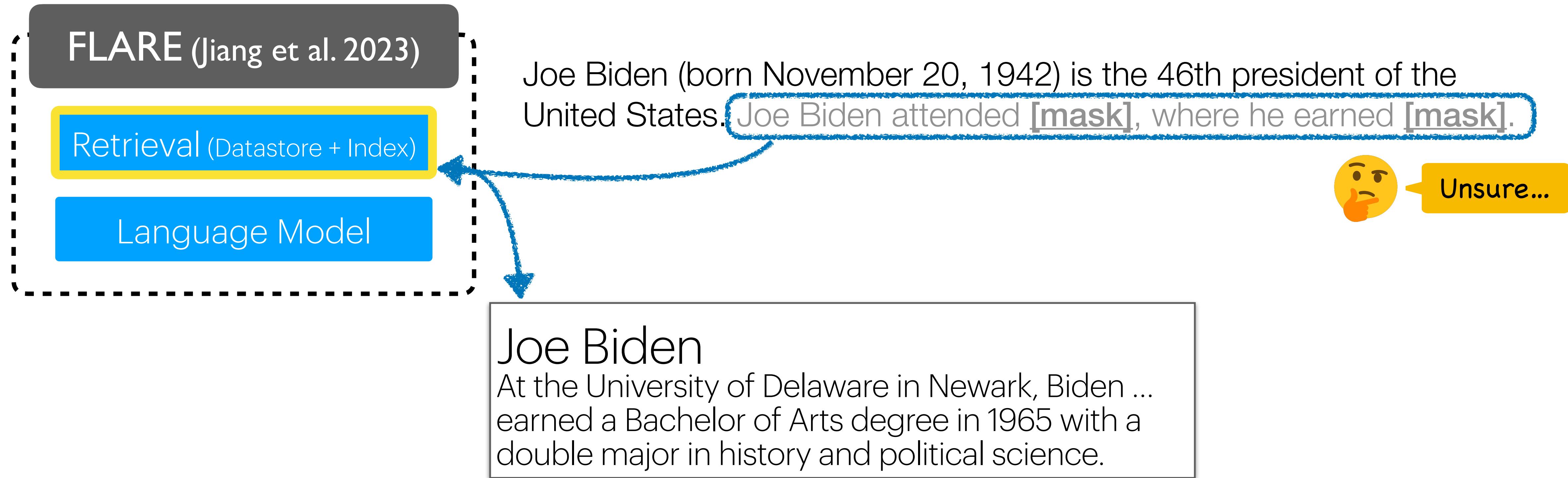
- Judge necessity

Input: Generate a summary about Joe Biden.



Adaptive retrieval of chunks

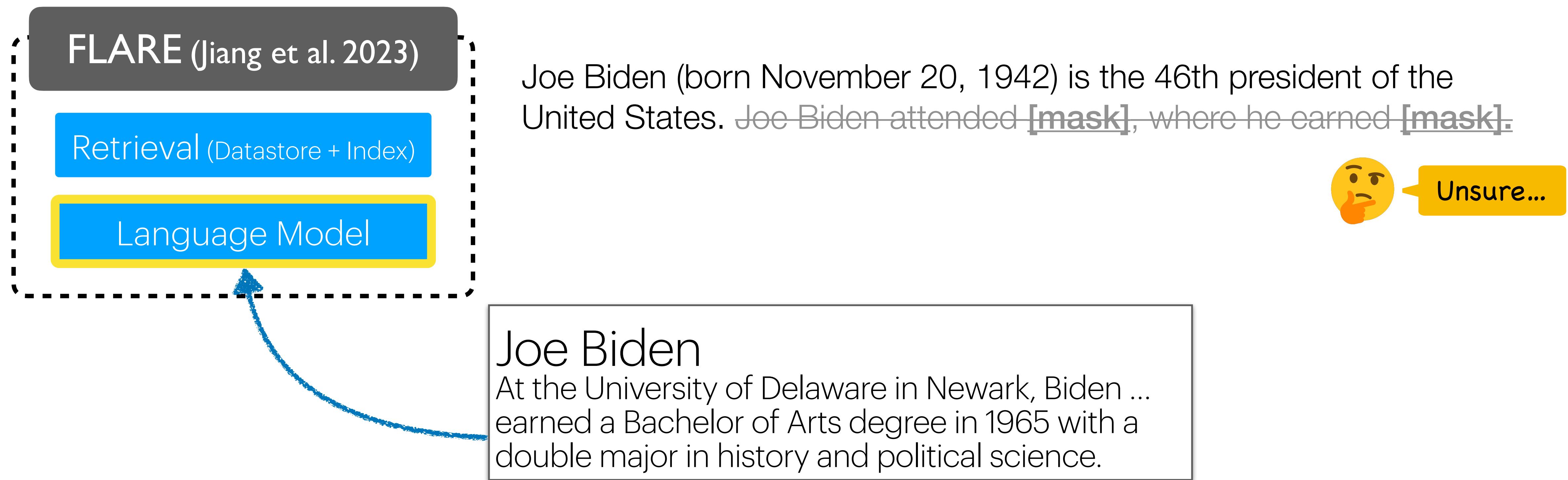
- Judge necessity



Adaptive retrieval of chunks

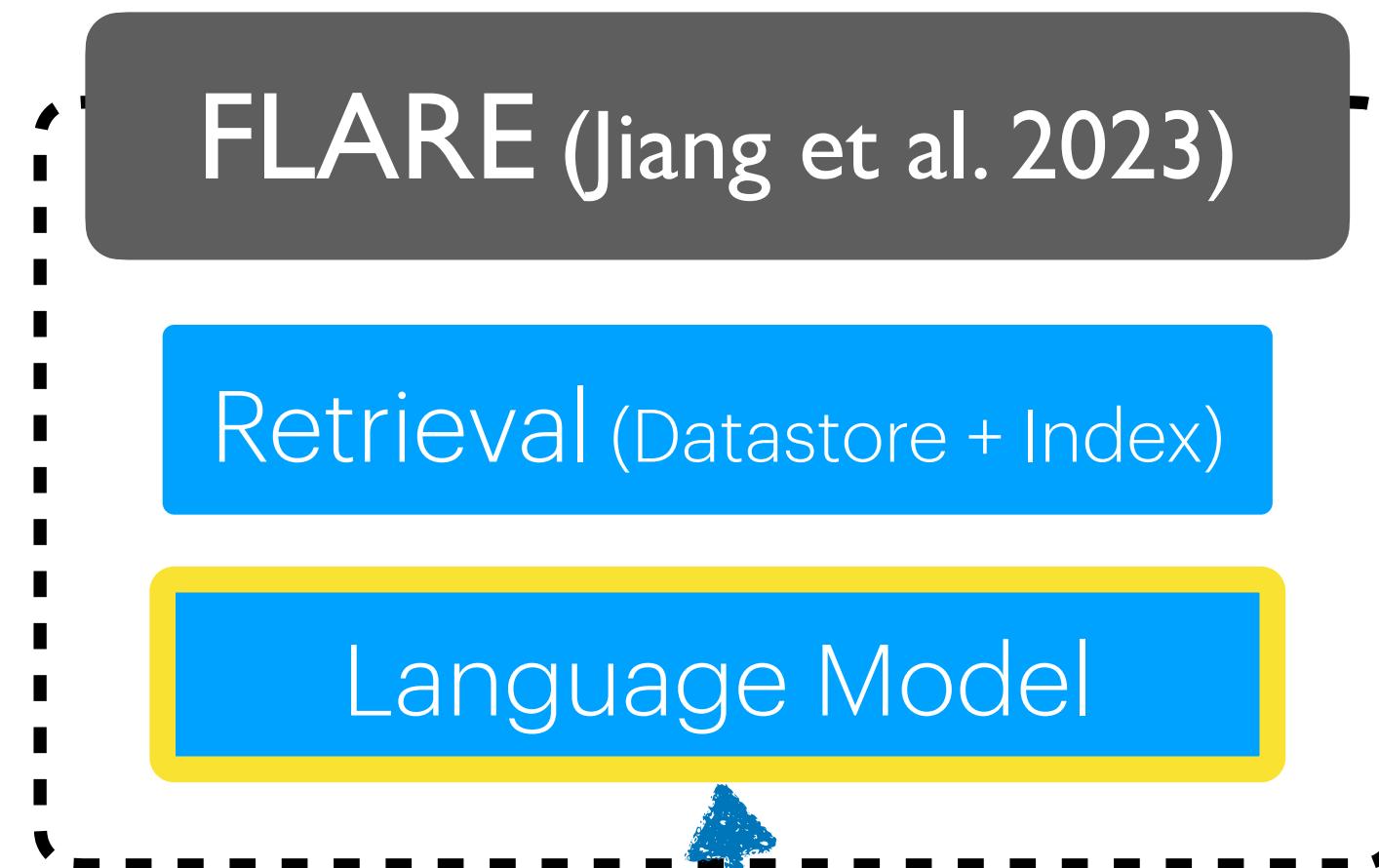
- Judge necessity

Input: Generate a summary about Joe Biden.



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.

Joe Biden
At the University of Delaware in Newark, Biden ...
earned a Bachelor of Arts degree in 1965 with a
double major in history and political science.

Adaptive retrieval of *tokens*

- *Judge necessity*

Adaptive retrieval of *tokens*

- *Judge necessity*

Joe Biden graduated from the University of Delaware .

Adaptive retrieval of *tokens*

- Judge necessity

retrieve retrieve retrieve retrieve retrieve retrieve retrieve
Joe Biden graduated from the University of Delaware .

Adaptive retrieval of *tokens*

- Judge necessity

retrieve retrieve retrieve retrieve retrieve retrieve retrieve
Joe Biden graduated from the University of Delaware .

retrieve LM LM retrieve retrieve retrieve LM
Joe Biden graduated from the University of Delaware .

Adaptive retrieval of *tokens*

- Judge necessity

retrieve retrieve retrieve retrieve retrieve retrieve retrieve
Joe Biden graduated from the University of Delaware .

retrieve LM LM retrieve retrieve retrieve LM
Joe Biden graduated from the University of Delaware .

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

Adaptive retrieval of *tokens*

- Judge necessity

retrieve retrieve retrieve retrieve retrieve retrieve retrieve
Joe Biden graduated from the University of Delaware .

retrieve LM LM retrieve retrieve retrieve LM
Joe Biden graduated from the University of Delaware .

$$P_{k\text{NN-LM}}(y|x) = \underbrace{(1 - \lambda(x))}_{\text{A function of the input } \mathbf{x}} P_{\text{LM}}(y|x) + \underbrace{\lambda(x)}_{\rightarrow \lambda = 0 \text{ if } \lambda < \gamma} P_{k\text{NN}}(y|x)$$

A function of the input \mathbf{x}

$$\rightarrow \lambda = 0 \text{ if } \lambda < \gamma$$

Adaptive retrieval of *tokens*

- *Use local info*

Adaptive retrieval of *tokens*

- *Use local info*

Training contexts	Targets
	<i>At the</i>
	<i>At the University</i>
<i>At the University of</i>	<i>of</i>
<i>At the University of Delaware</i>	<i>Delaware</i>
<i>At the University of Delaware in</i>	<i>in</i>
<i>At the University of Delaware in Newark</i>	<i>Newark</i>

Joe Biden graduated from

Adaptive retrieval of tokens

- Use local info

Training contexts	Targets
	At the
	At the University
At the University	of
At the University of	Delaware
At the University of Delaware	in
At the University of Delaware in Newark	Newark

retrieve
Joe Biden graduated from the

Adaptive retrieval of tokens

- Use local info

Training contexts	Targets
	<i>At the</i>
<i>At the University</i>	
<i>At the University of</i>	<i>of</i>
<i>At the University of Delaware</i>	<i>Delaware</i>
<i>At the University of Delaware in</i>	<i>in</i>
<i>At the University of Delaware in Newark</i>	<i>Newark</i>

retrieve retrieve
Joe Biden graduated from the University

Adaptive retrieval of tokens

- Use local info

Training contexts	Targets
	<i>At the</i>
<i>At the University</i>	<i>University</i>
<i>At the University of</i>	<i>of</i>
<i>At the University of Delaware</i>	<i>Delaware</i>
<i>At the University of Delaware in</i>	<i>in</i>
<i>At the University of Delaware in Newark</i>	<i>Newark</i>

Joe Biden graduated from the University of

retrieve retrieve retrieve



Adaptive retrieval of tokens

- Use local info

Training contexts	Targets
	<i>At the</i>
	<i>At the</i>
<i>At the University</i>	<i>University</i>
	<i>of</i>
<i>At the University of</i>	<i>Delaware</i>
<i>At the University of Delaware</i>	<i>in</i>
<i>At the University of Delaware in</i>	<i>Newark</i>

Joe Biden graduated from the University of Delaware.

retrieve retrieve retrieve retrieve



Adaptive retrieval of tokens

- Use local info

Training contexts	Targets
	At the
	At the University
At the University	of
At the University of	Delaware
At the University of Delaware	in
At the University of Delaware in Newark	

retrieve
Joe Biden graduated from the

Adaptive retrieval of tokens

- Use local info

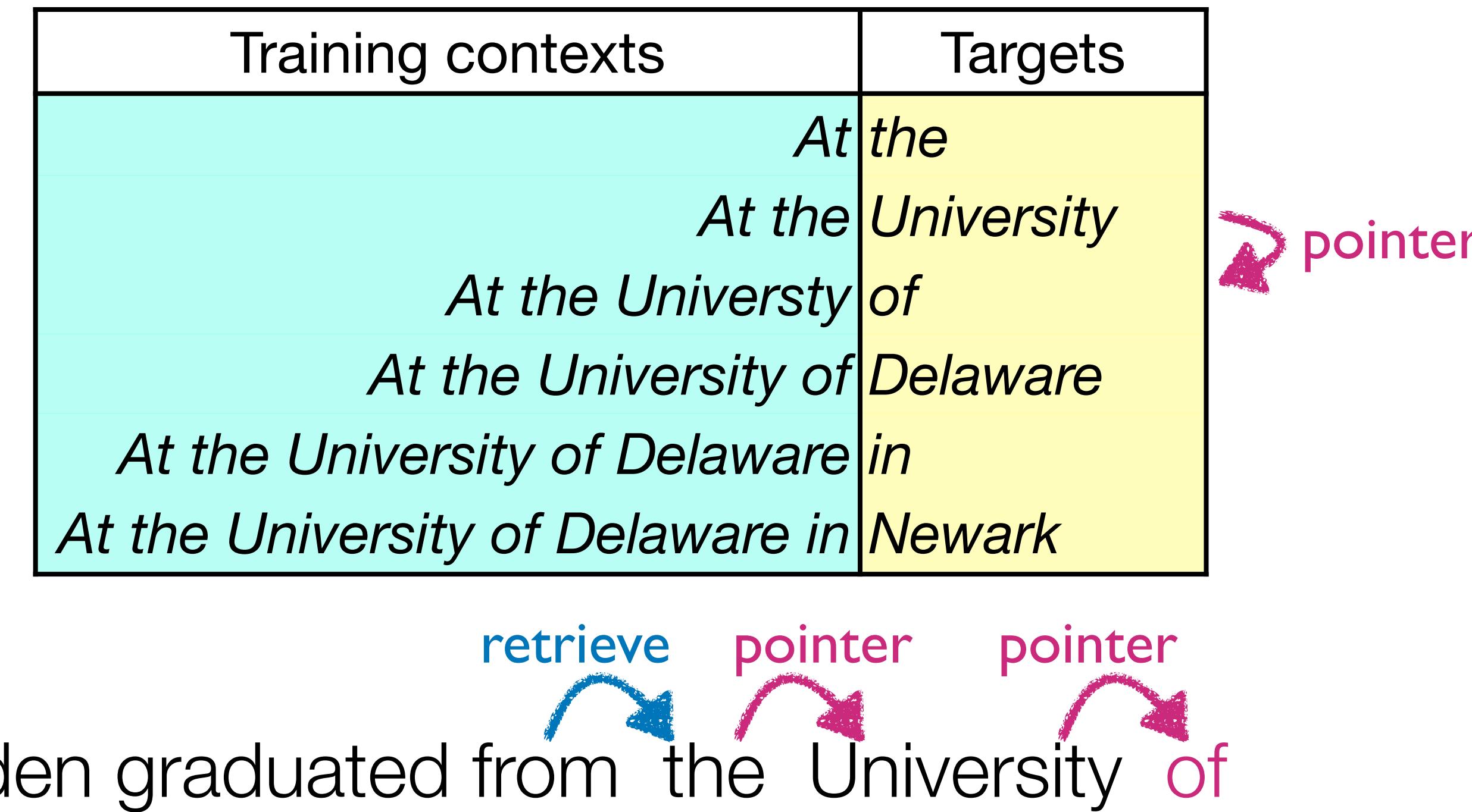
Training contexts	Targets
	<i>At</i> <i>the</i>
	<i>At the</i> <i>University</i>
	<i>At the University</i> <i>of</i>
	<i>At the University of</i> <i>Delaware</i>
<i>At the University of Delaware</i> <i>in</i>	<i>in</i>
<i>At the University of Delaware in Newark</i>	<i>Newark</i>

retrieve pointer

Joe Biden graduated from the University

Adaptive retrieval of tokens

- Use local info



Adaptive retrieval of tokens

- Use local info

Training contexts	Targets
	<i>At the</i>
	<i>At the</i>
<i>At the University</i>	<i>University</i>
<i>At the University of</i>	<i>of</i>
<i>At the University of Delaware</i>	<i>Delaware</i>
<i>At the University of Delaware in</i>	<i>in</i>
<i>At the University of Delaware in Newark</i>	<i>Newark</i>

Joe Biden graduated from the University of Delaware.

retrieve pointer pointer pointer

Adaptive retrieval of tokens

- Use local info

Training contexts	Targets
	<i>At the</i>
	<i>At the</i>
<i>At the University</i>	<i>University</i>
<i>At the University of</i>	<i>of</i>
<i>At the University of Delaware</i>	<i>Delaware</i>
<i>At the University of Delaware in</i>	<i>in</i>
<i>At the University of Delaware in Newark</i>	<i>Newark</i>

Joe Biden graduated from the University of Delaware.



Retrieve once, and save other searches!

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)

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)



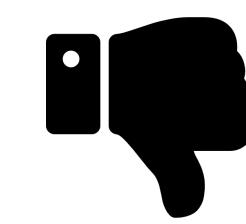
More efficient

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)



More efficient



Decision may not always be optimal

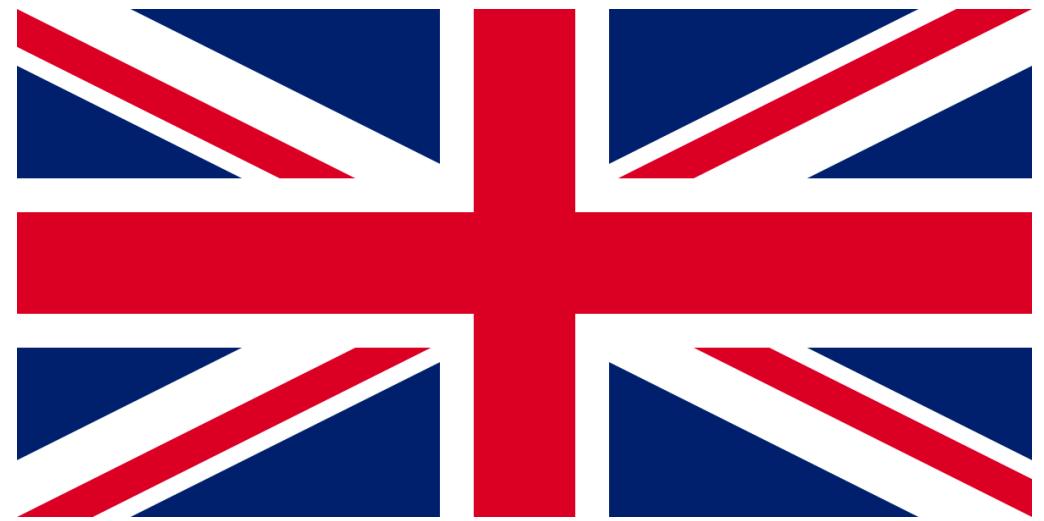
Summary

	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)

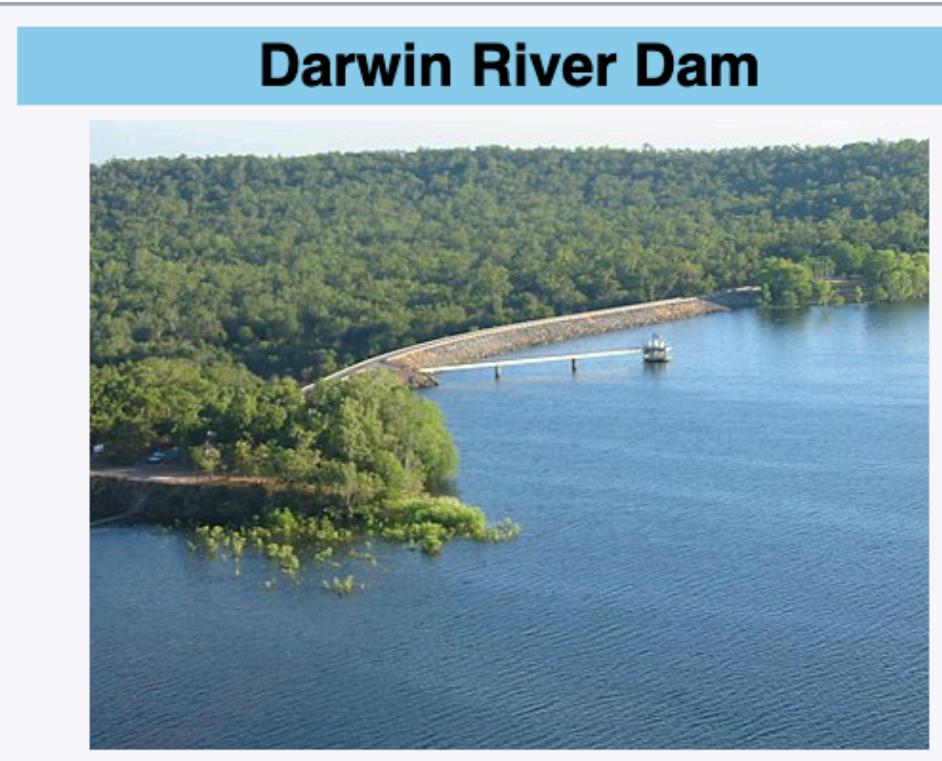
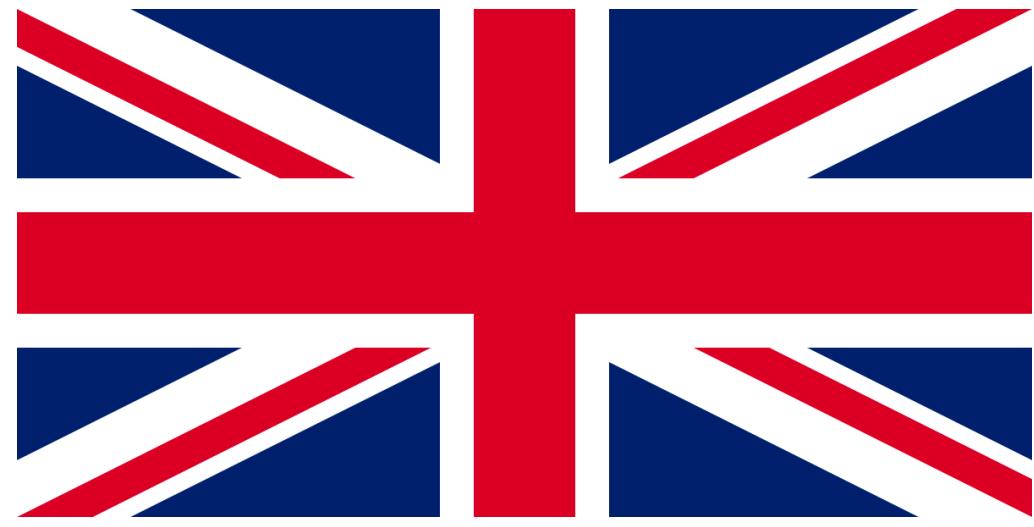
What else beyond text chunks and tokens?

Entities as Experts (Fevry et al. 2020)

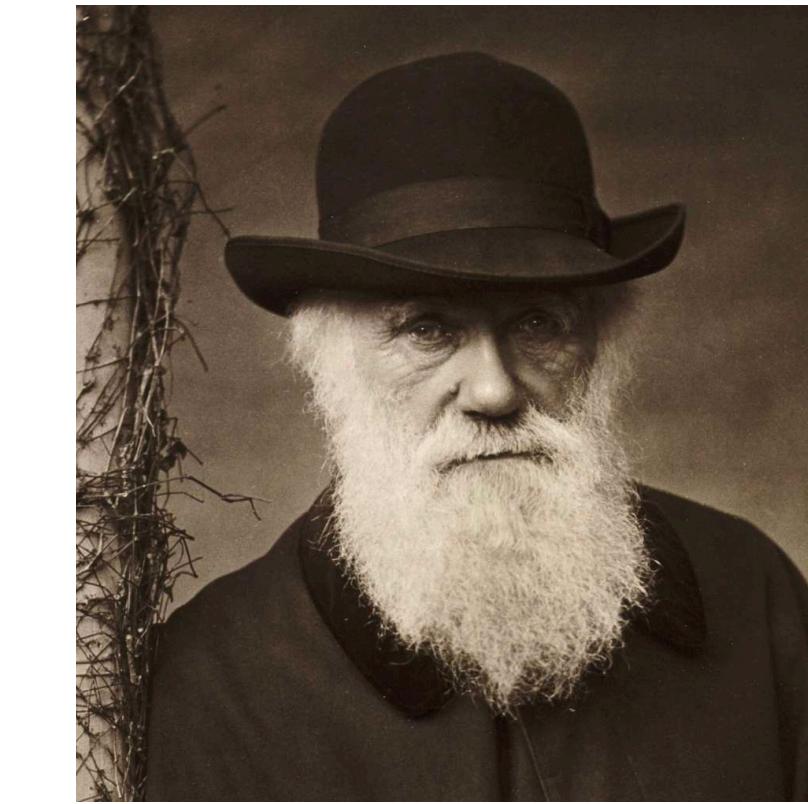
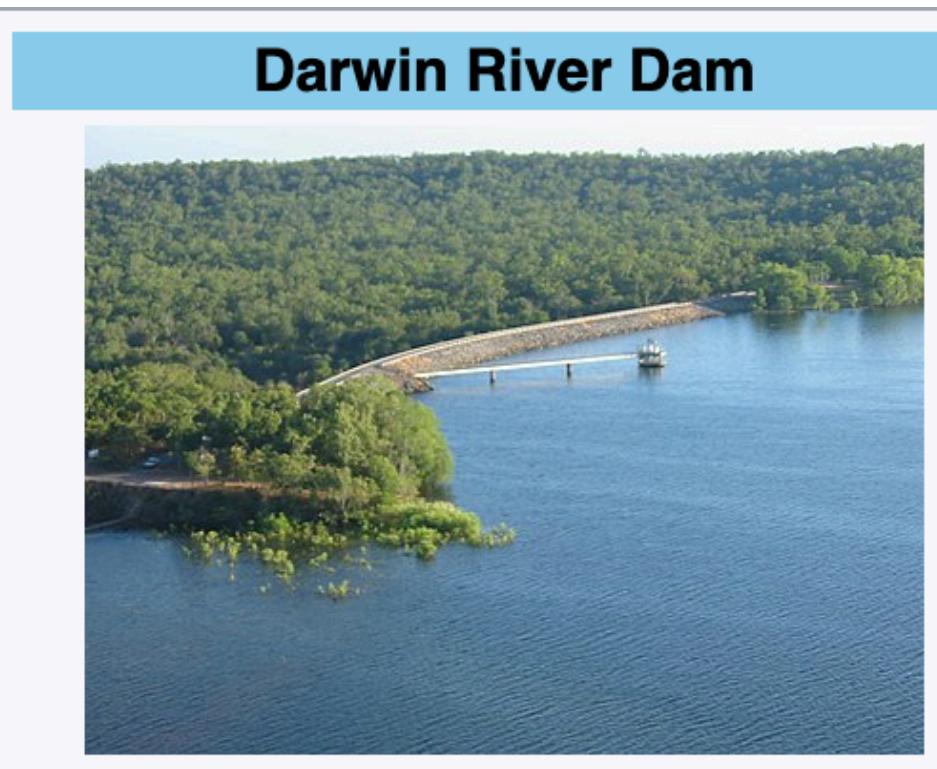
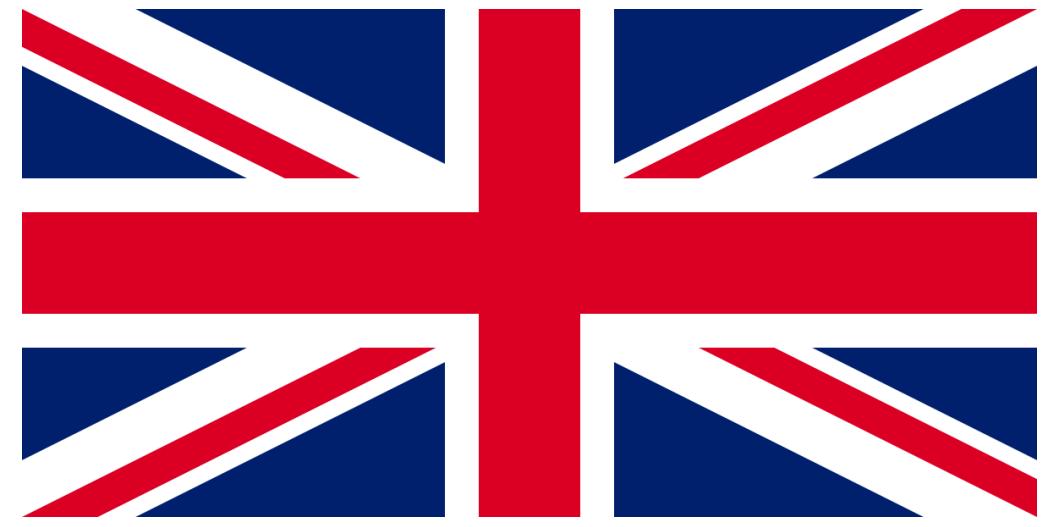
Entities as Experts (Fevry et al. 2020)



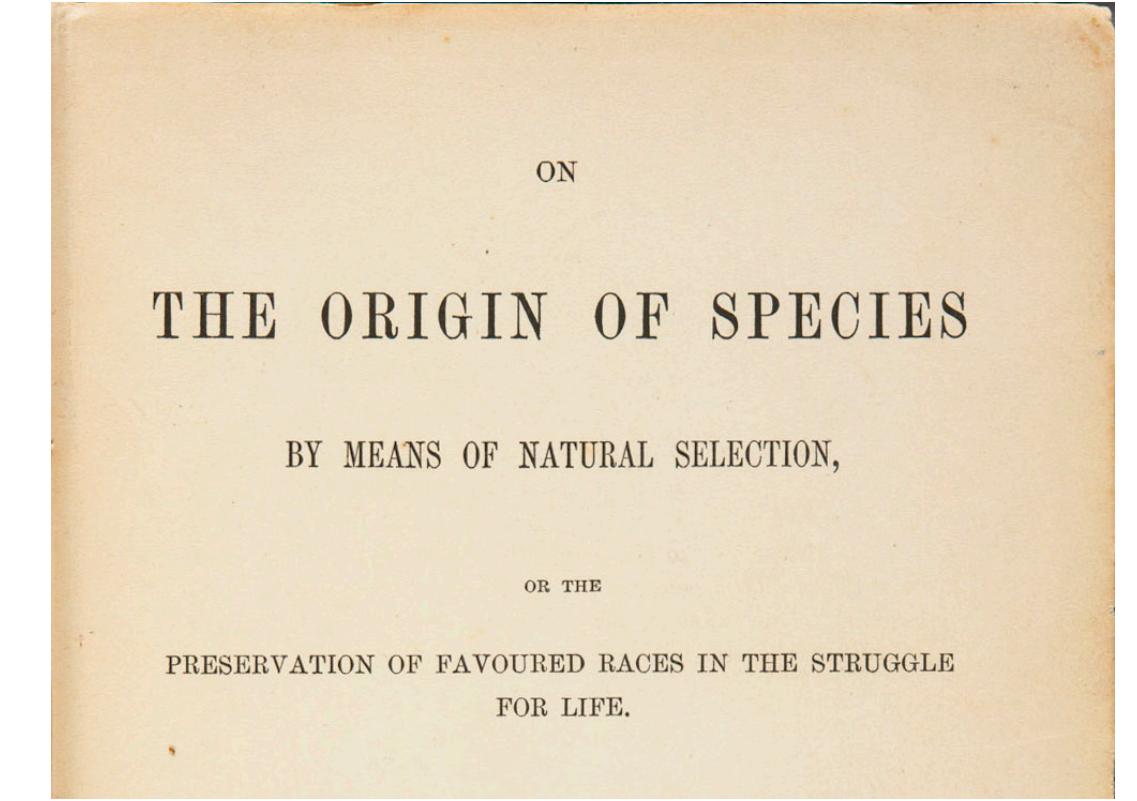
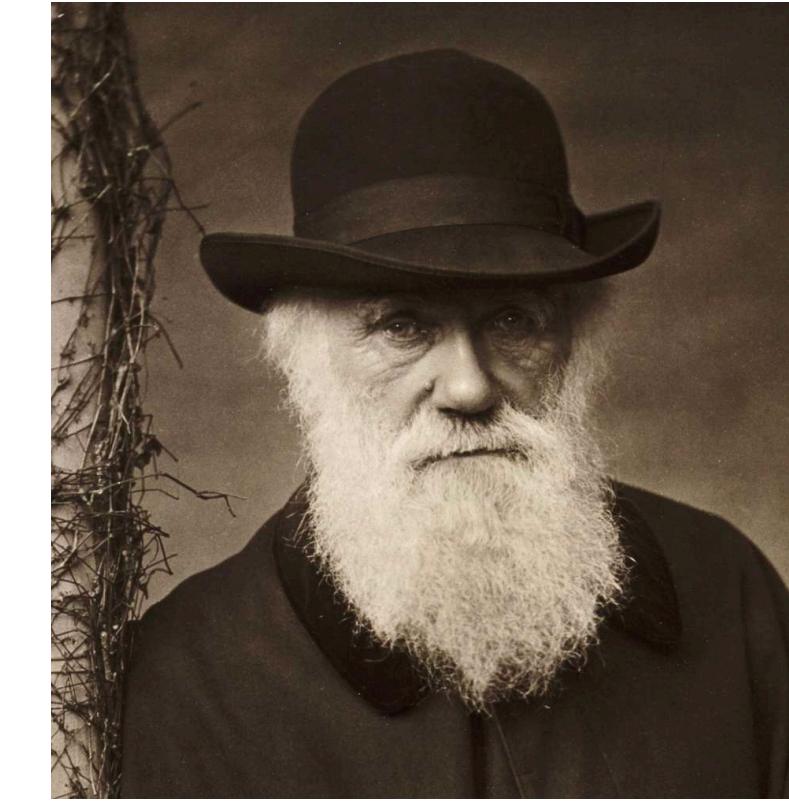
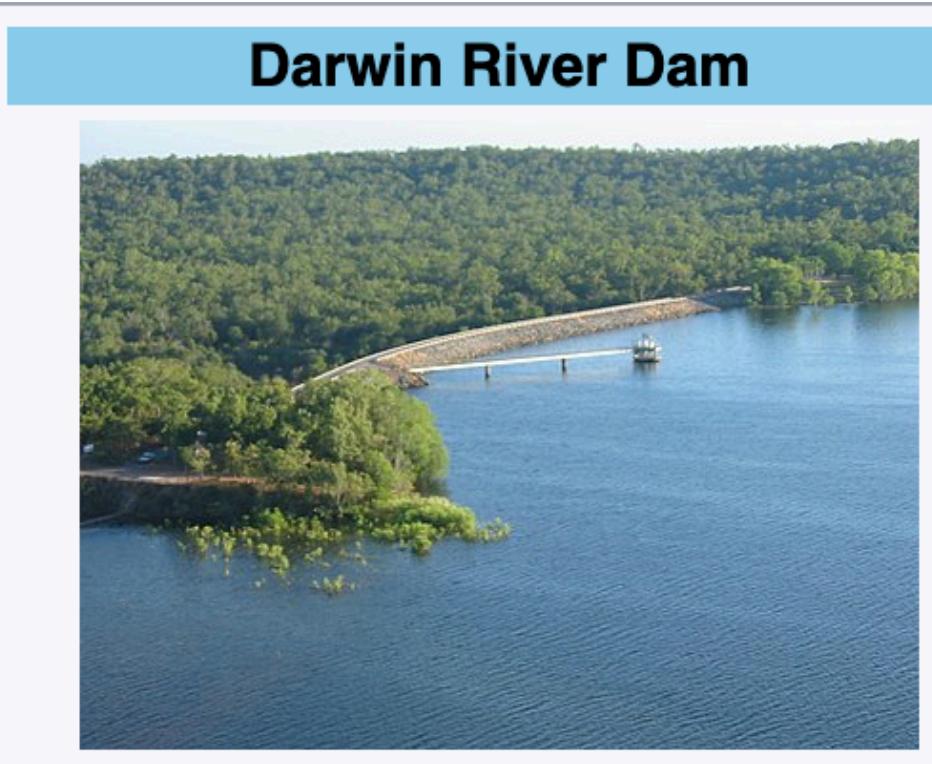
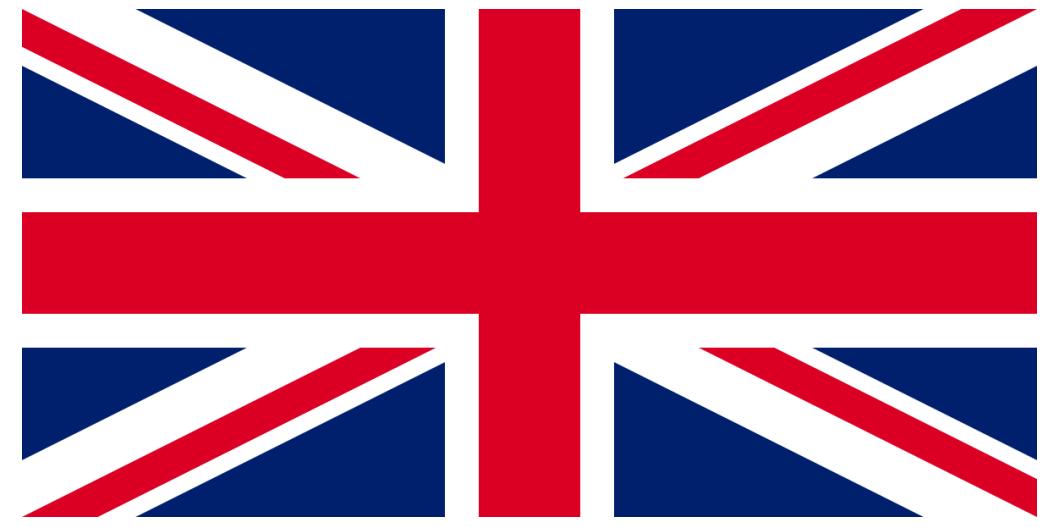
Entities as Experts (Fevry et al. 2020)



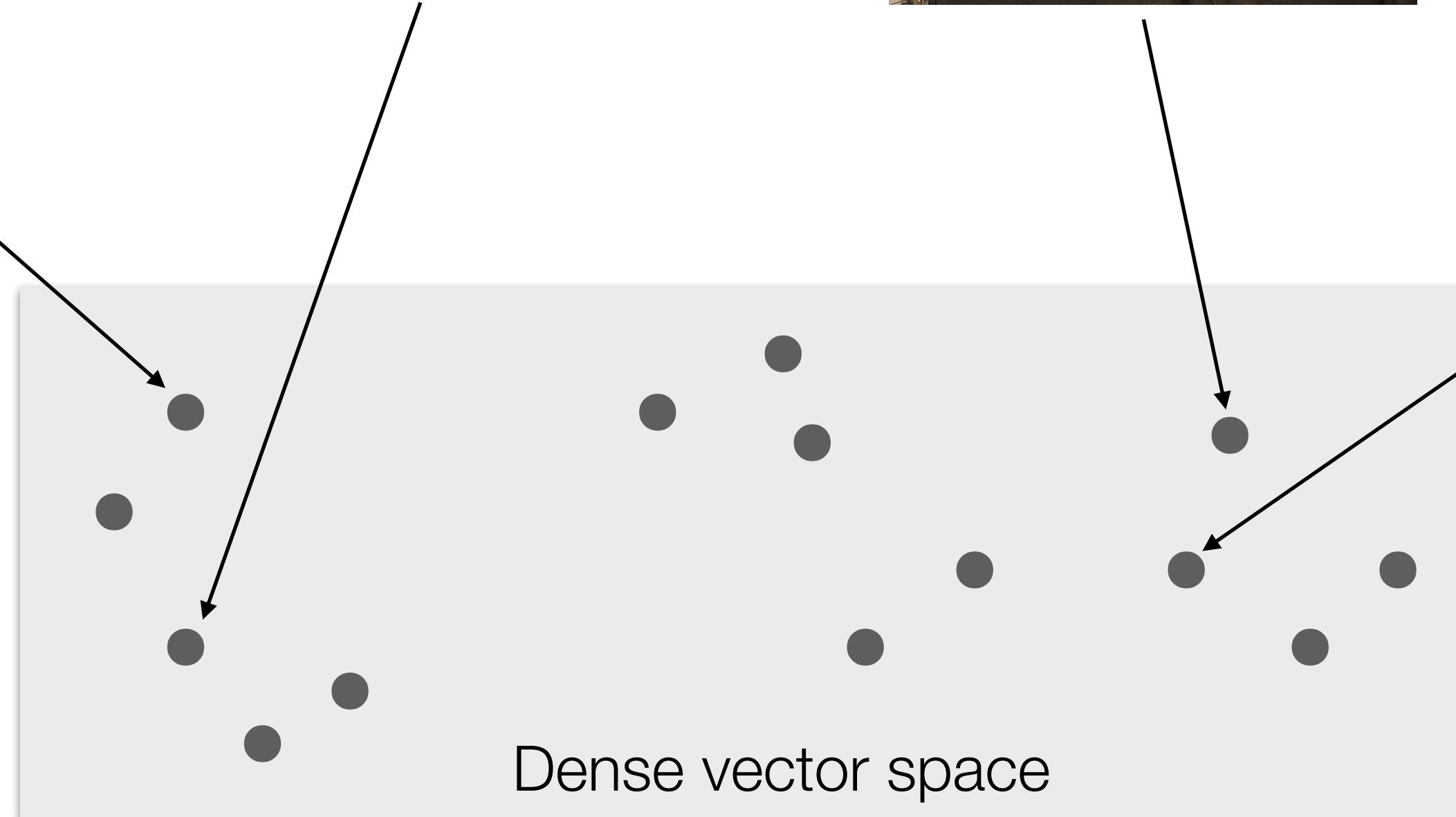
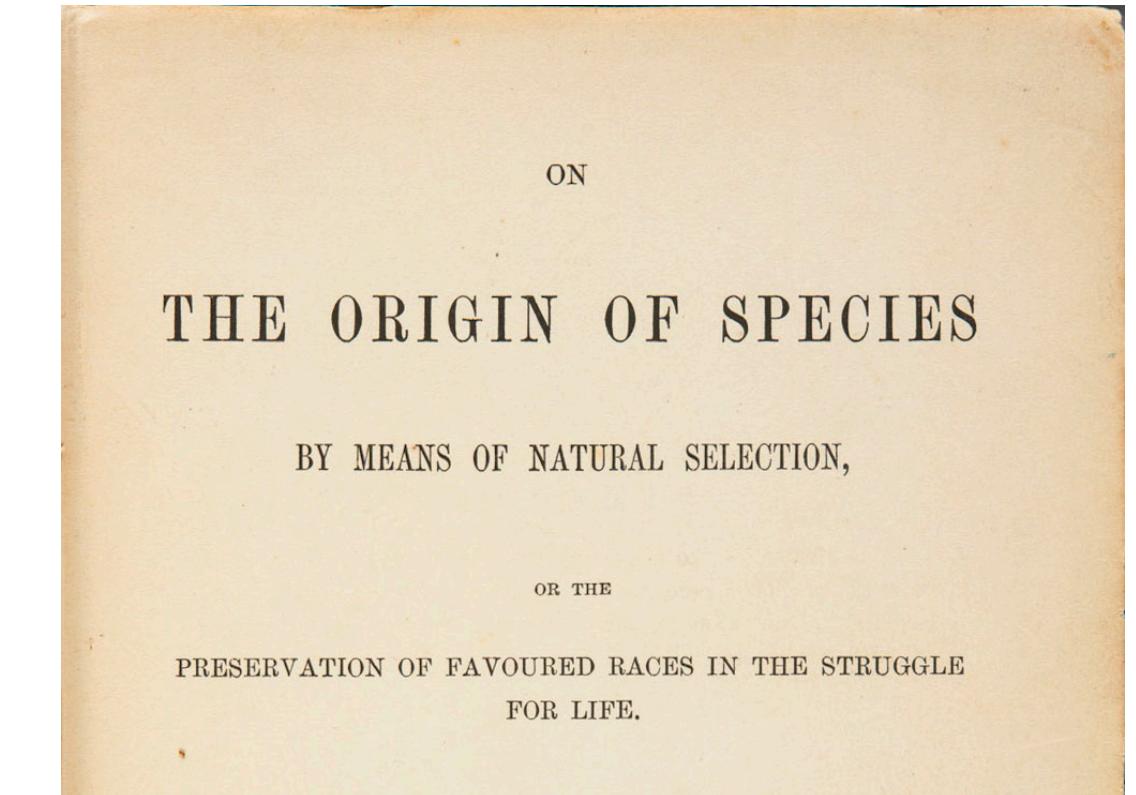
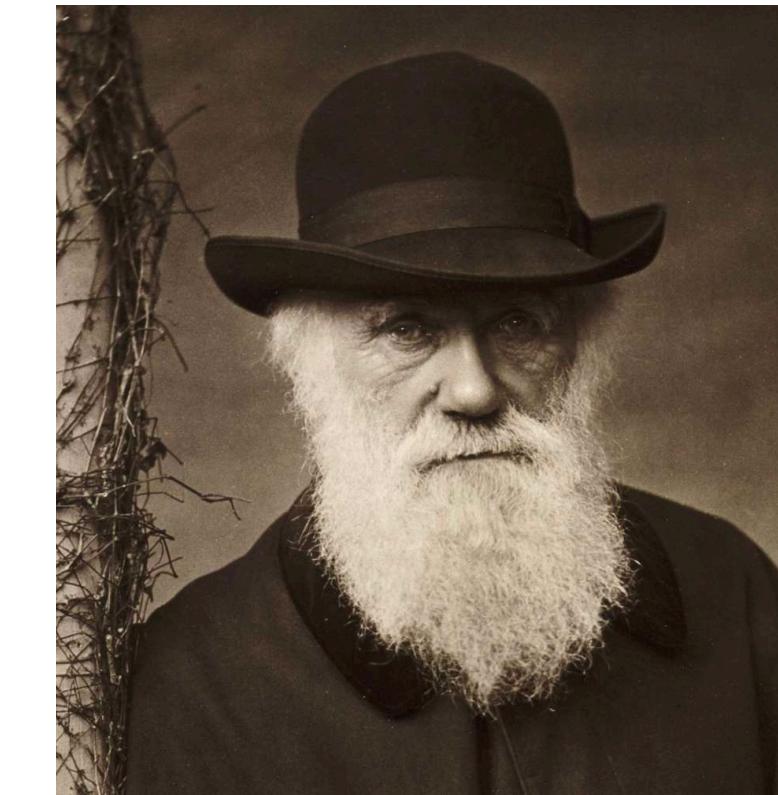
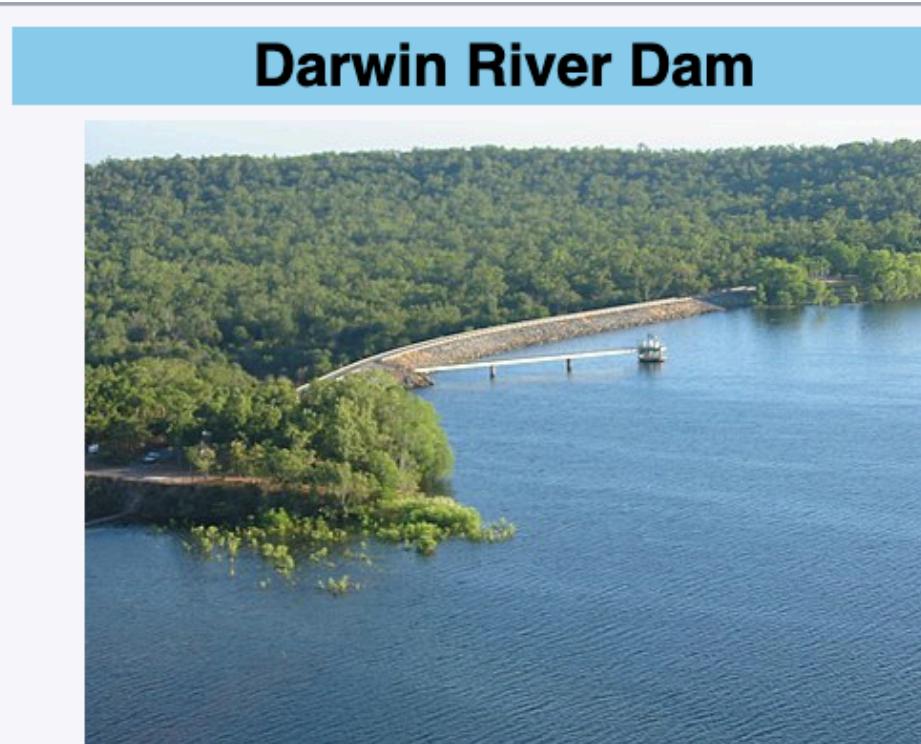
Entities as Experts (Fevry et al. 2020)



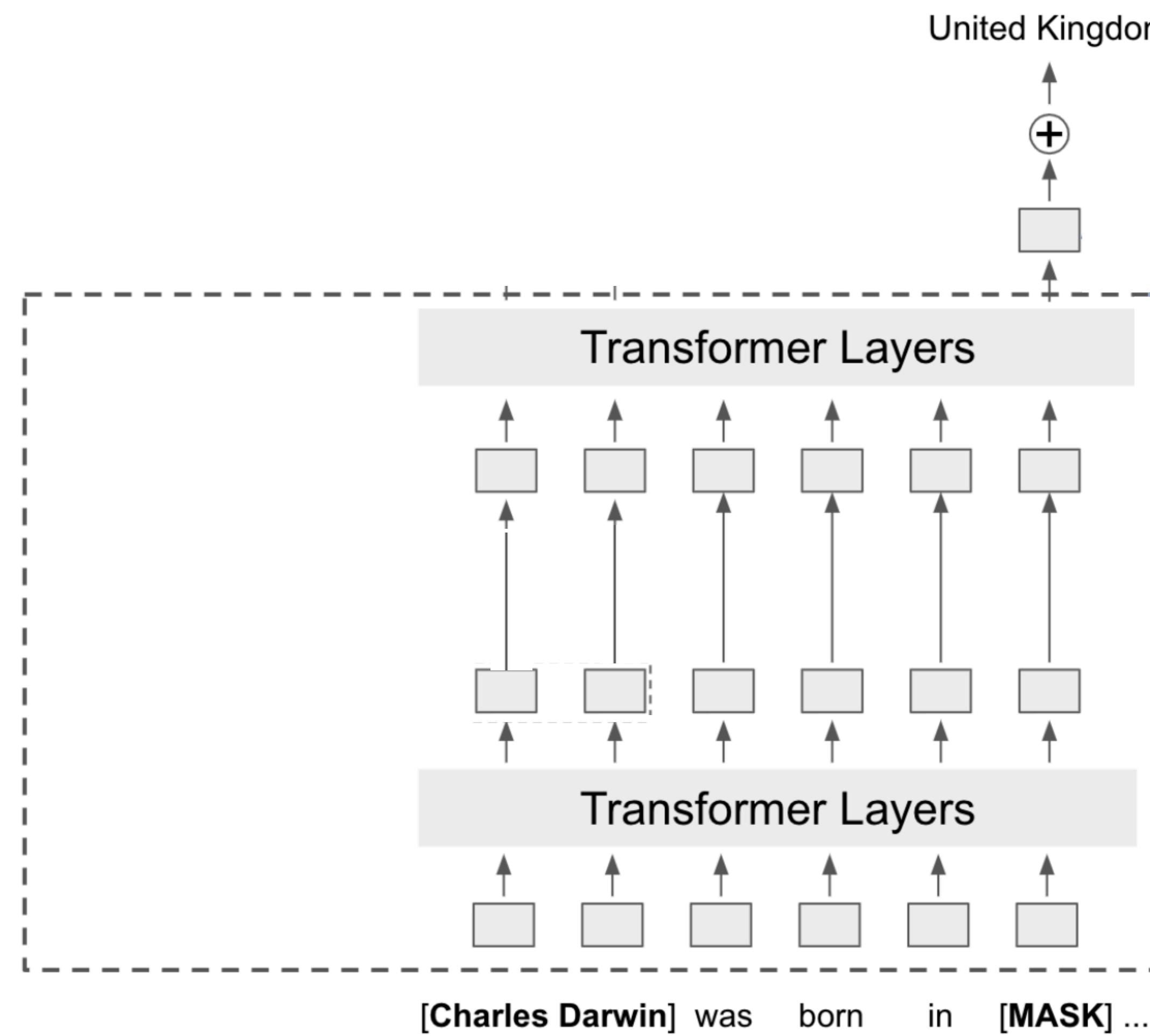
Entities as Experts (Fevry et al. 2020)



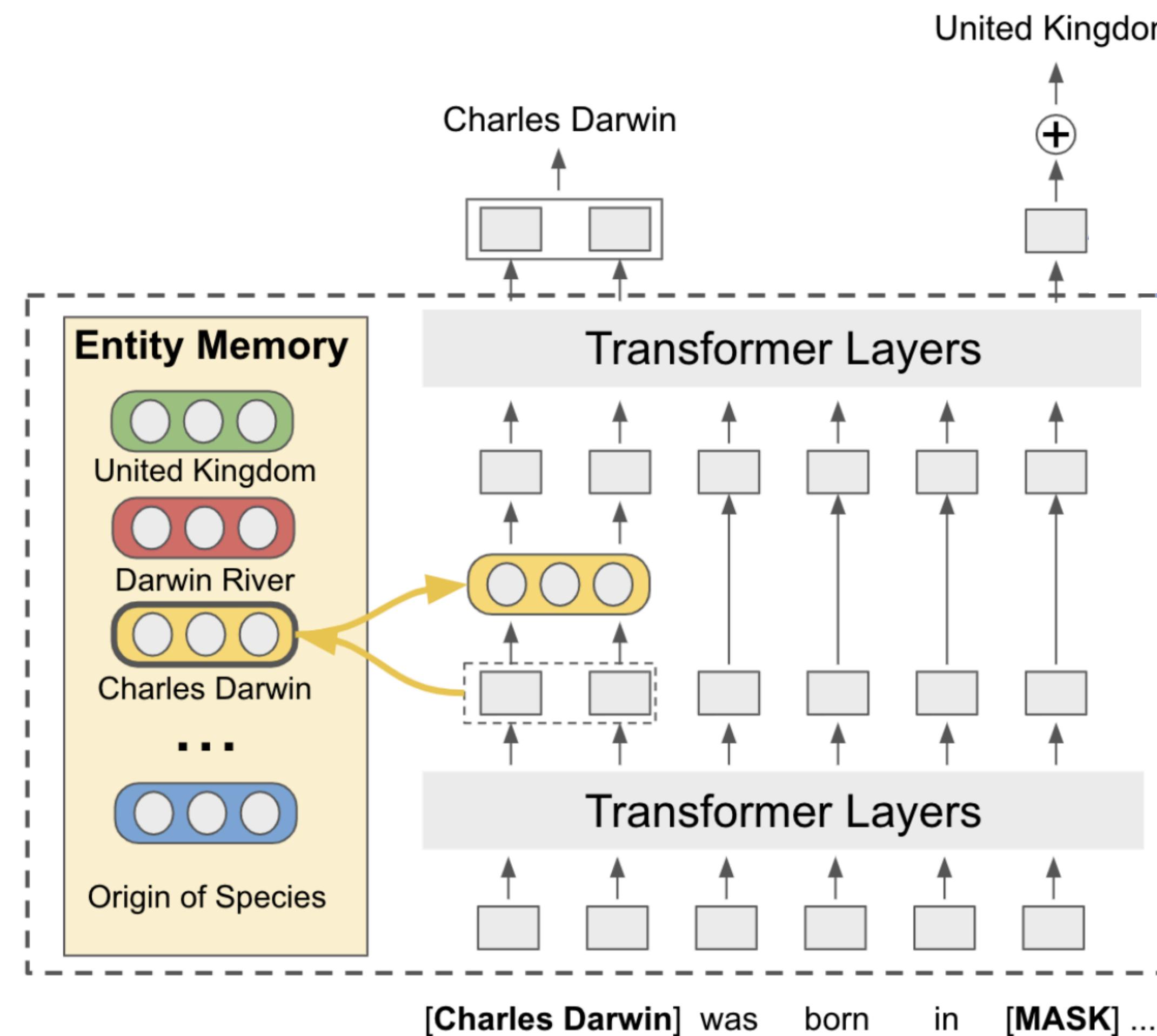
Entities as Experts (Fevry et al. 2020)



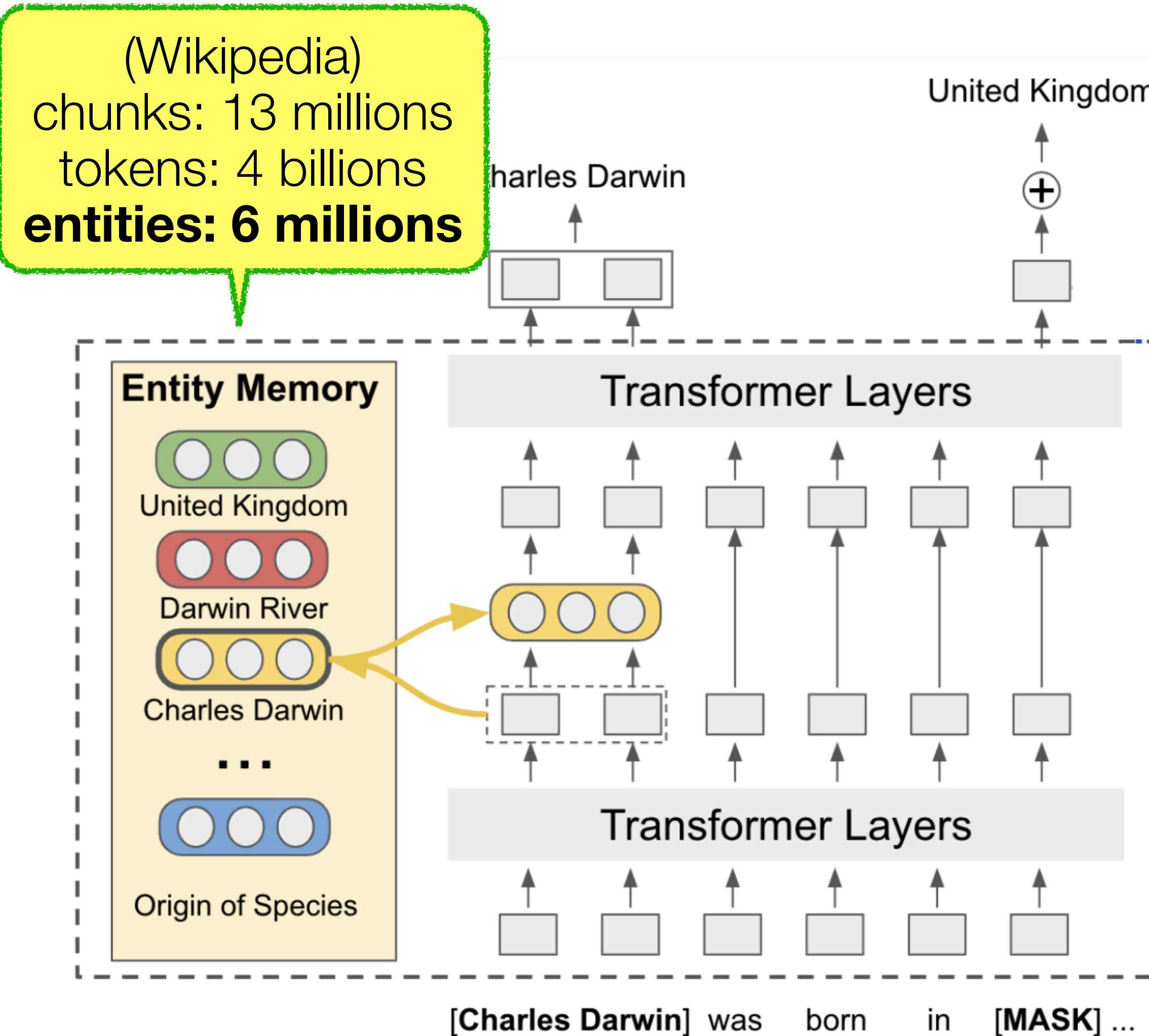
Entities as Experts (Fevry et al. 2020)



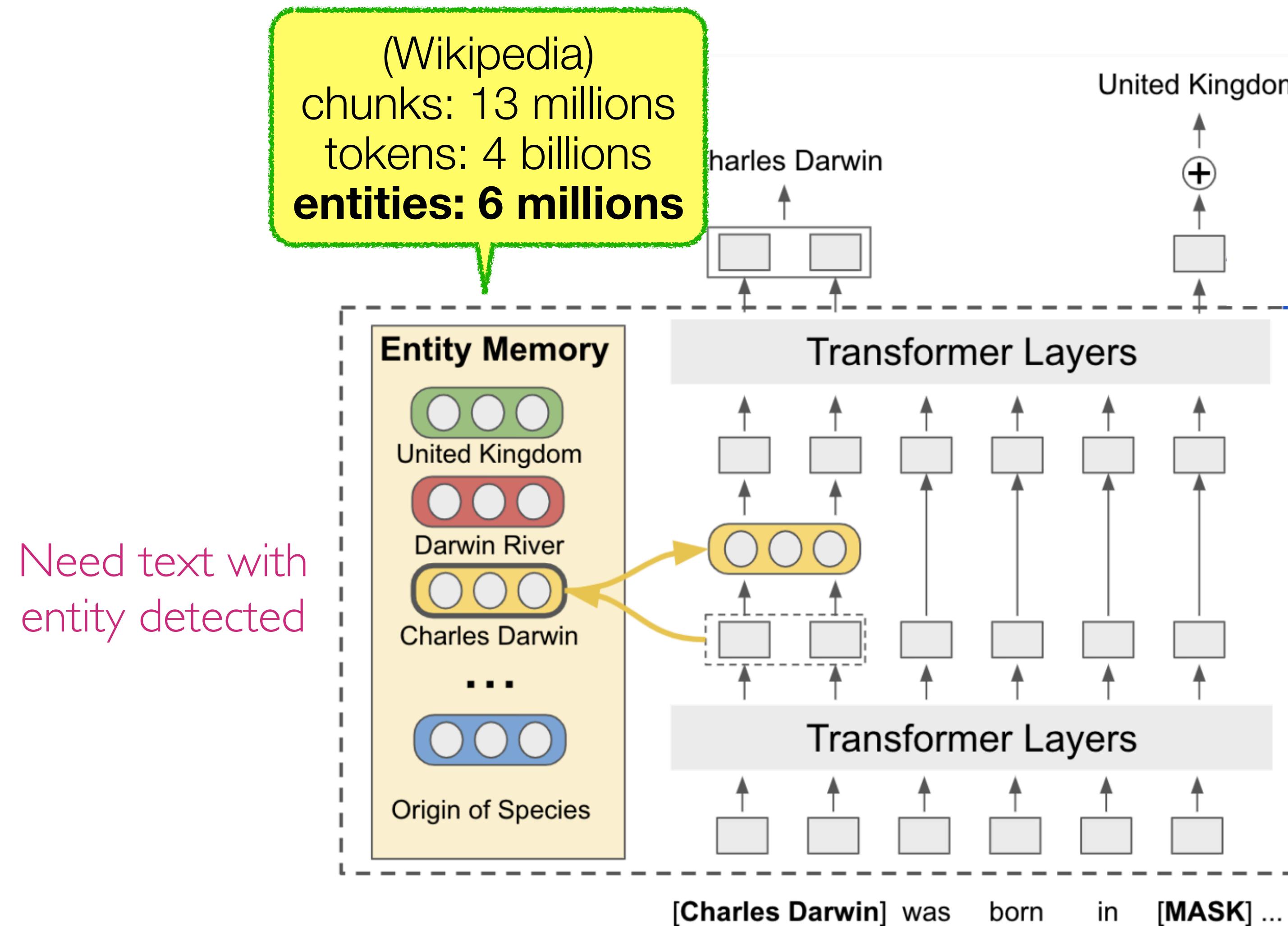
Entities as Experts (Fevry et al. 2020)



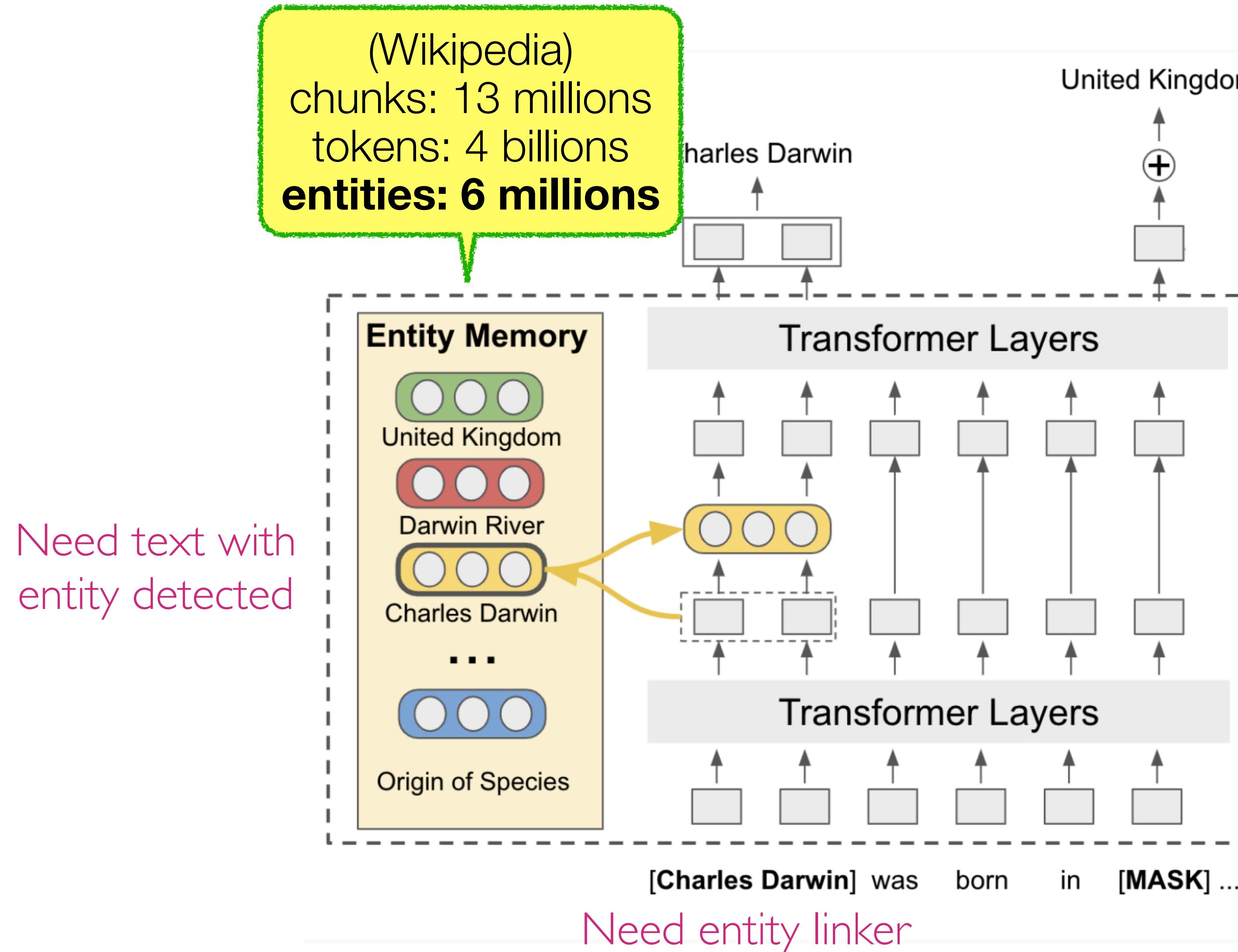
Entities as Experts (Fevry et al. 2020)



Entities as Experts (Fevry et al. 2020)



Entities as Experts (Fevry et al. 2020)

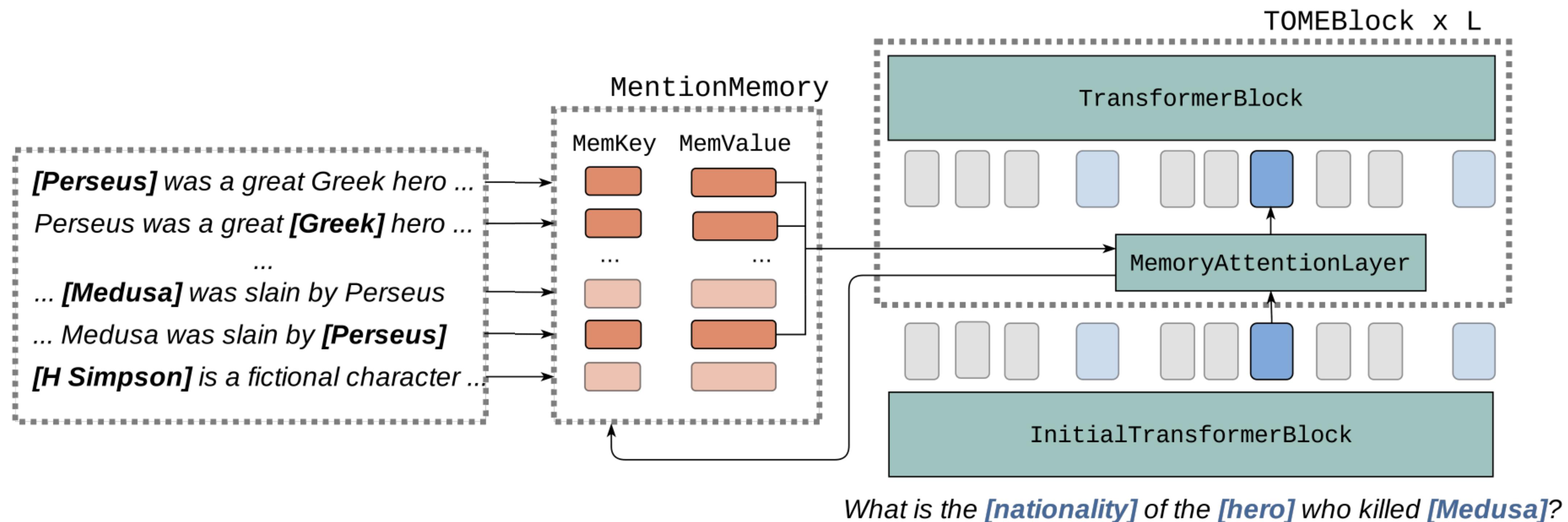


Mention Memory (de Jong et al. 2022)

One vector per entity → One vector per entity *mention*

Mention Memory (de Jong et al. 2022)

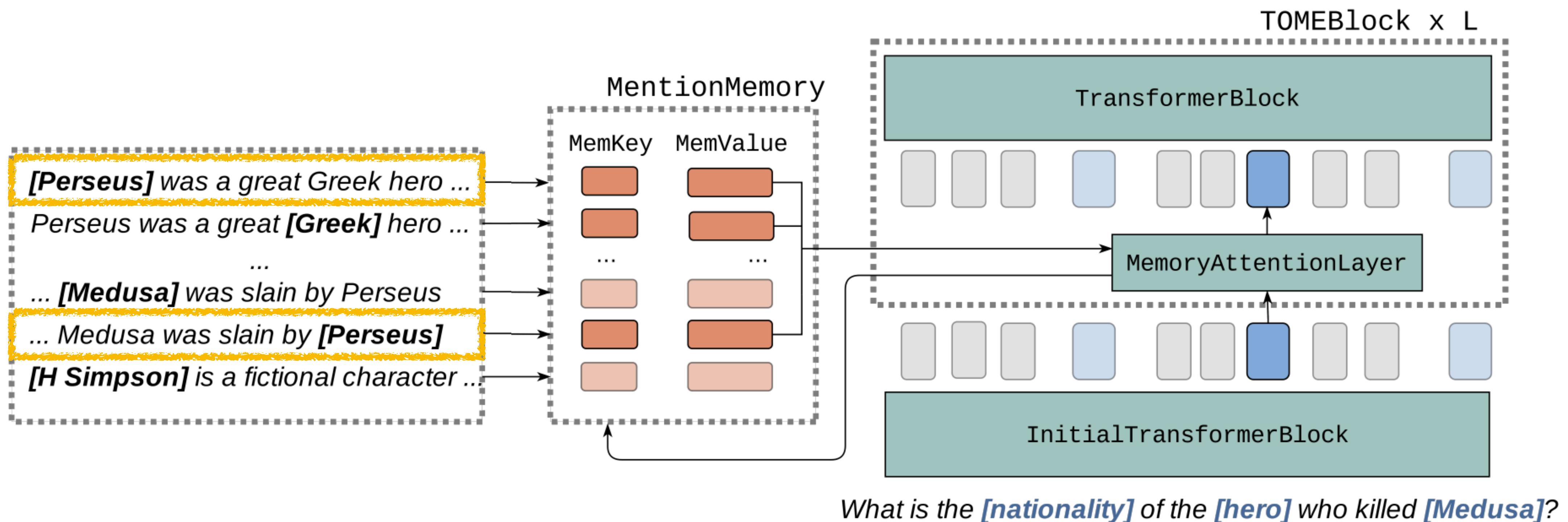
One vector per entity → One vector per entity mention



de Jong et al. 2022. "Mention Memory:
incorporating textual knowledge into Transformers through entity mention attention"

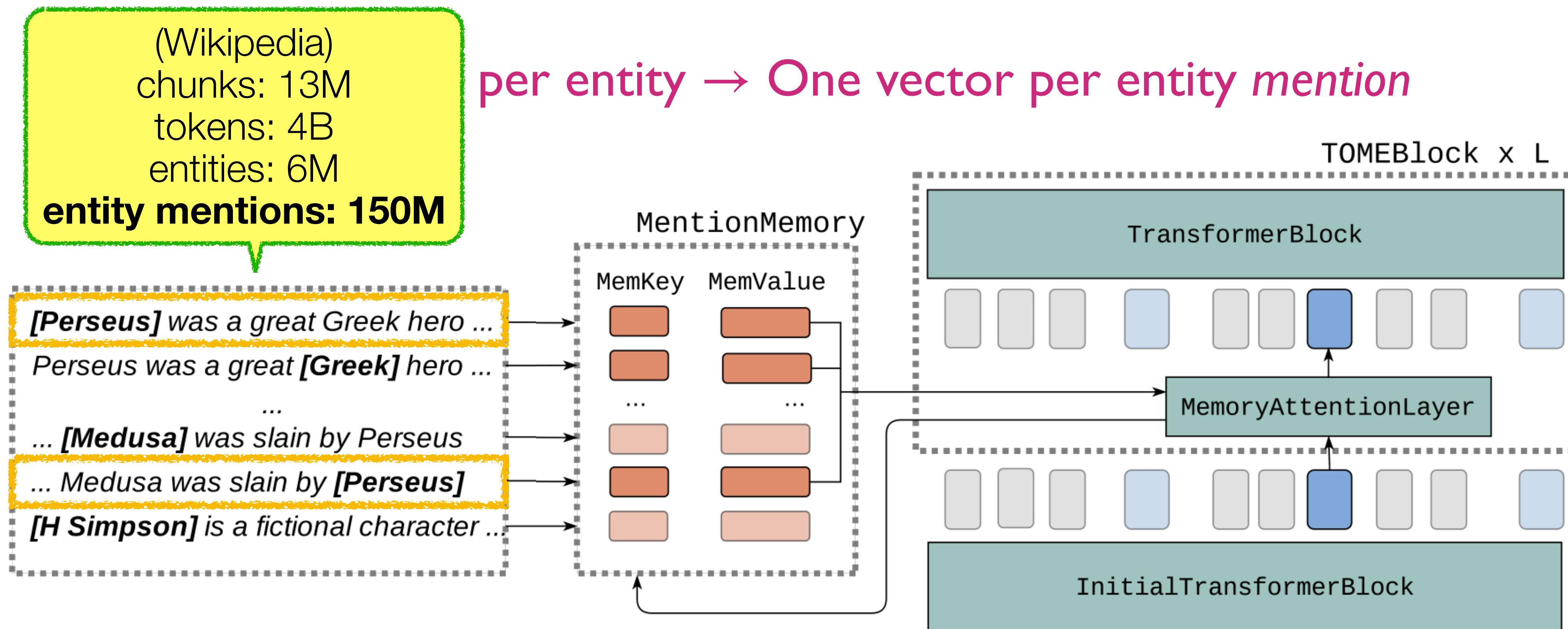
Mention Memory (de Jong et al. 2022)

One vector per entity → One vector per entity mention



de Jong et al. 2022. "Mention Memory:
incorporating textual knowledge into Transformers through entity mention attention"

Mention Memory (de Jong et al. 2022)



What is the **[nationality]** of the **[hero]** who killed **[Medusa]**?

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)
Entities as Experts (Fevry et al. 2020), Mention Memory (de Jong et al. 2022)	Entities or entity mentions	Intermediate layers	Every entity mentions

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)
Entities as Experts (Fevry et al. 2020), Mention Memory (de Jong et al. 2022)	Entities or entity mentions	Intermediate layers	Every entity mentions



Most effective for entity-centric tasks & space-efficient

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)
Entities as Experts (Fevry et al. 2020), Mention Memory (de Jong et al. 2022)	Entities or entity mentions	Intermediate layers	Every entity mentions



Most effective for entity-centric tasks & space-efficient



Additional entity detection required

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)
Entities as Experts (Fevry et al. 2020), Mention Memory (de Jong et al. 2022)	Entities or entity mentions	Intermediate layers	Every entity mentions

All models retrieve from the external text

Summary

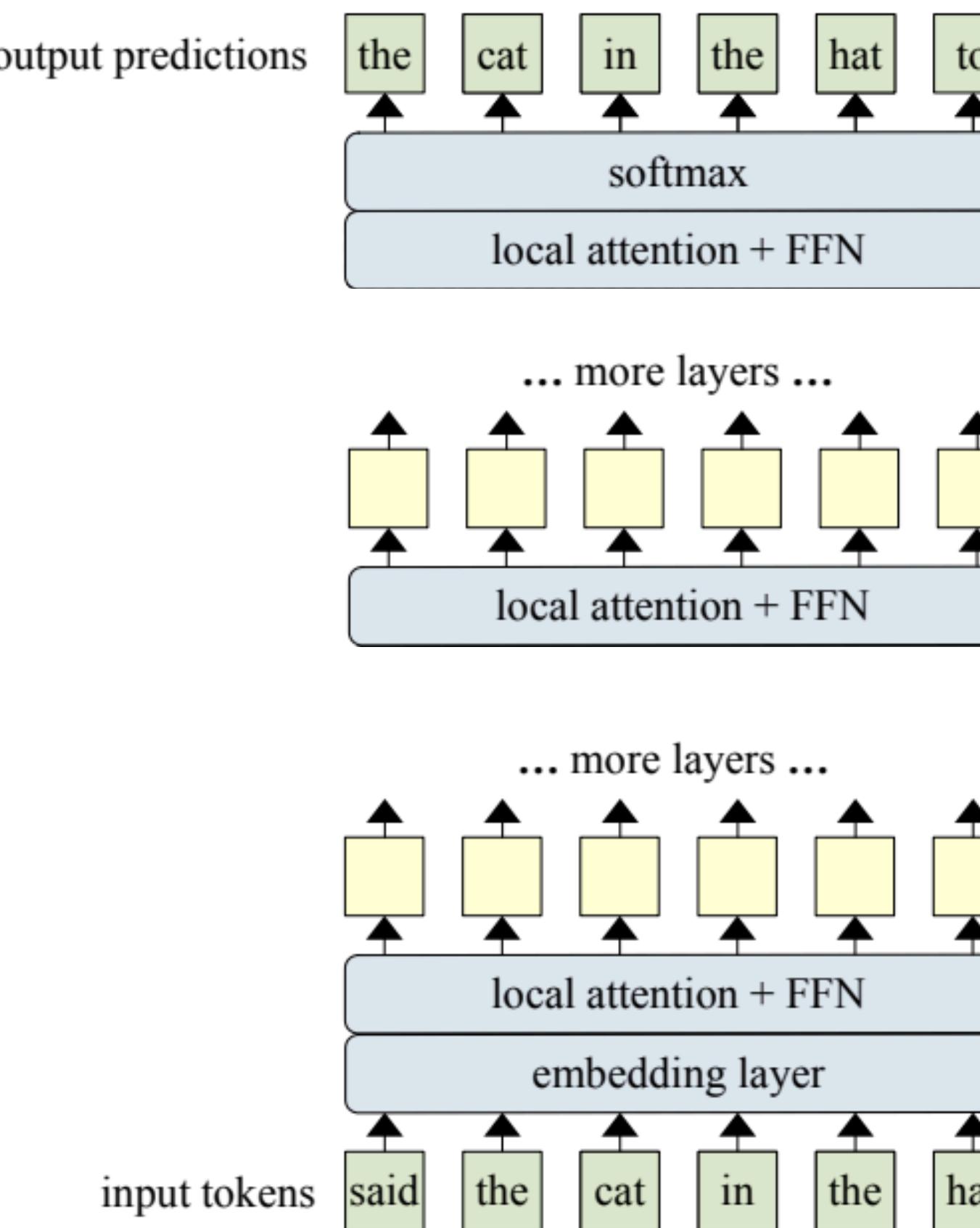
	What do retrieve?	How to use retrieval?	When to retrieve?
REALM (Guu et al 2020)	Text chunks	Input layer	Once
Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)	Text chunks	Input layer	Every n tokens
RETRO (Borgeaud et al. 2021)	Text chunks	Intermediate layers	Every n tokens
kNN-LM (Khandelwal et 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)
Entities as Experts (Fevry et al. 2020), Mention Memory (de Jong et al. 2022)	Entities or entity mentions	Intermediate layers	Every entity mentions

*All models retrieve from the external text
What else can we do with these models?*

Retrieval for long-range LM

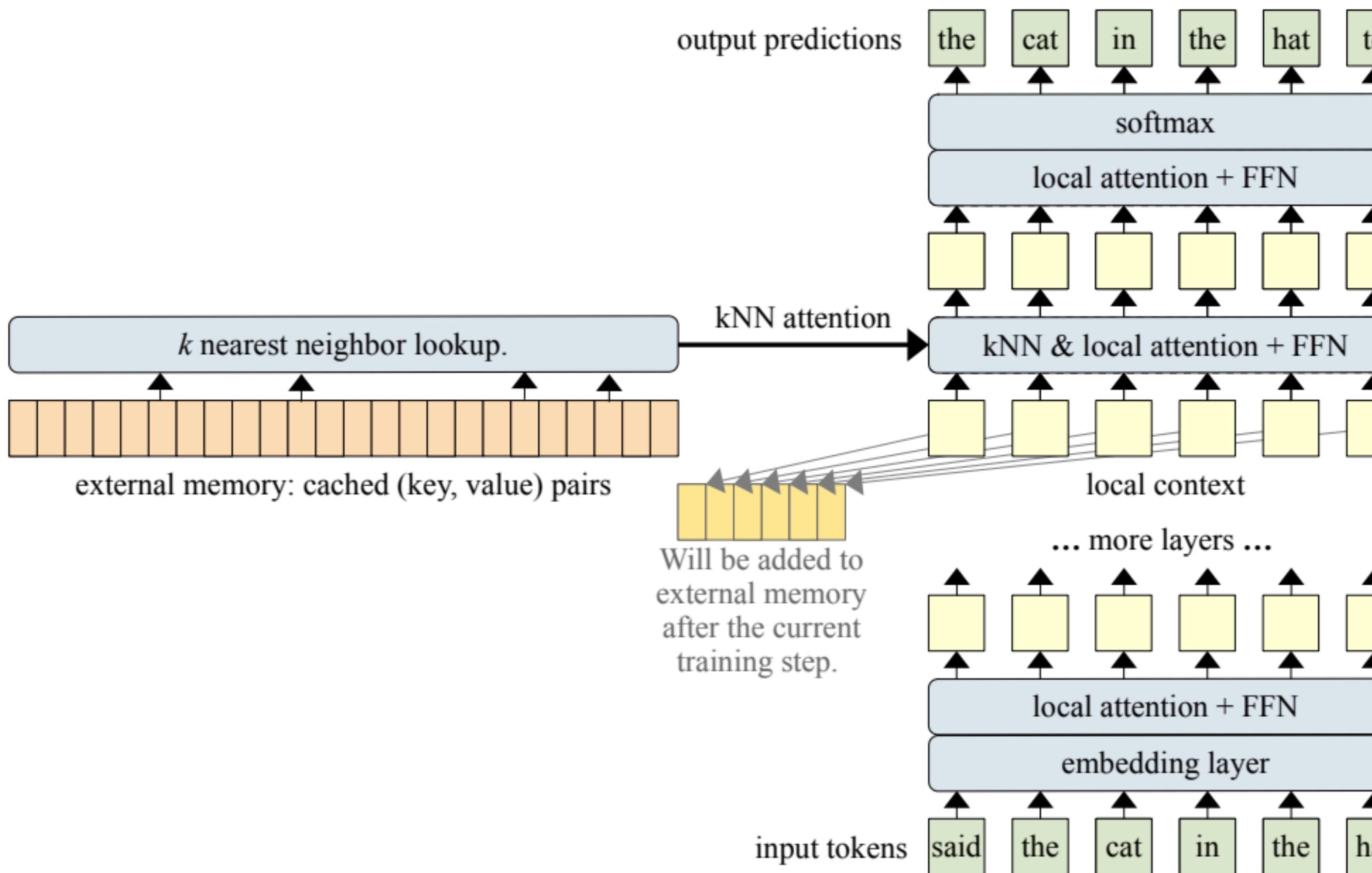
Wu et al. 2022. Memorizing Transformers (**Figure source**)
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval

Retrieval for long-range LM



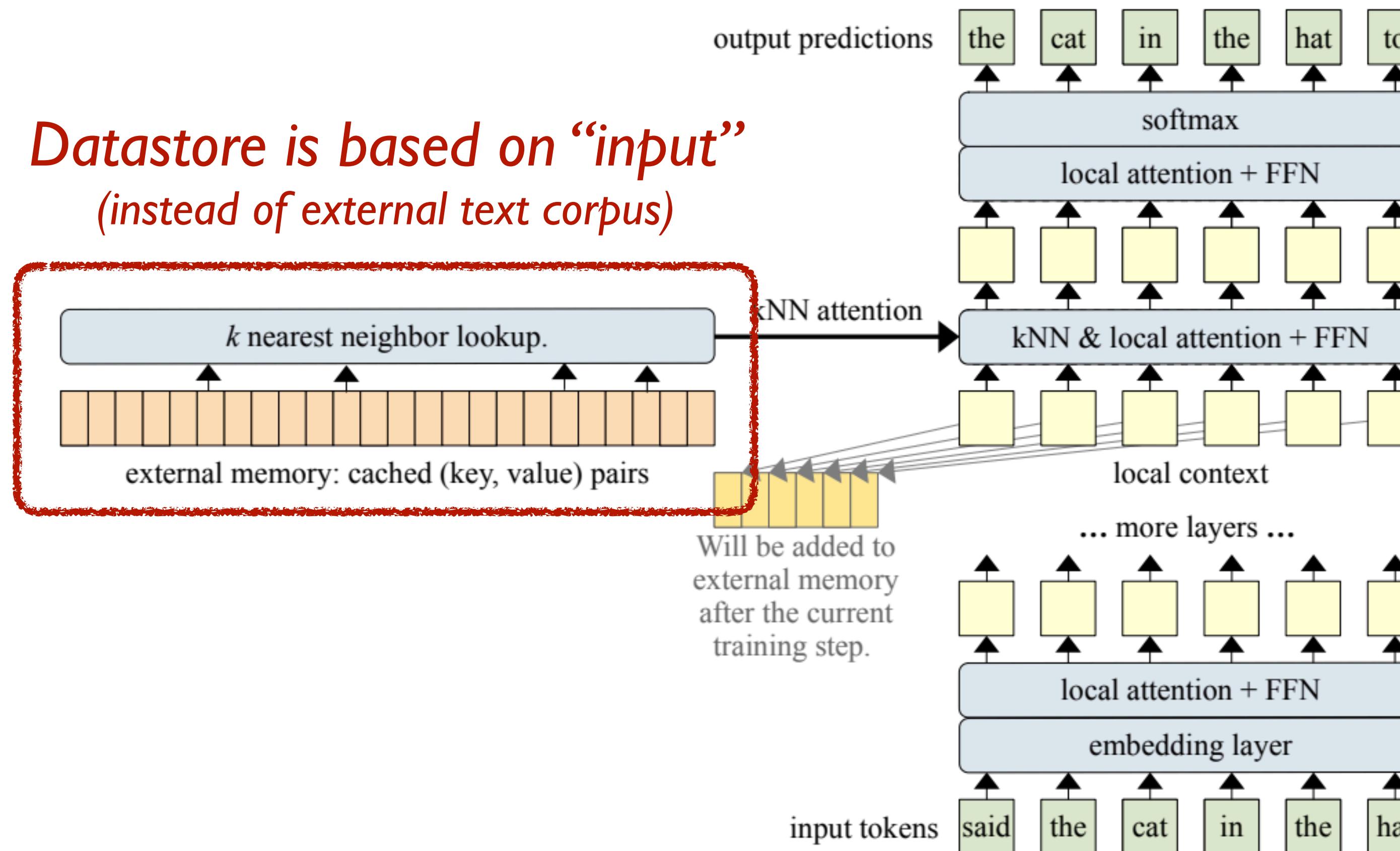
Wu et al. 2022. Memorizing Transformers (**Figure source**)
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval

Retrieval for long-range LM



Wu et al. 2022. Memorizing Transformers (**Figure source**)
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval

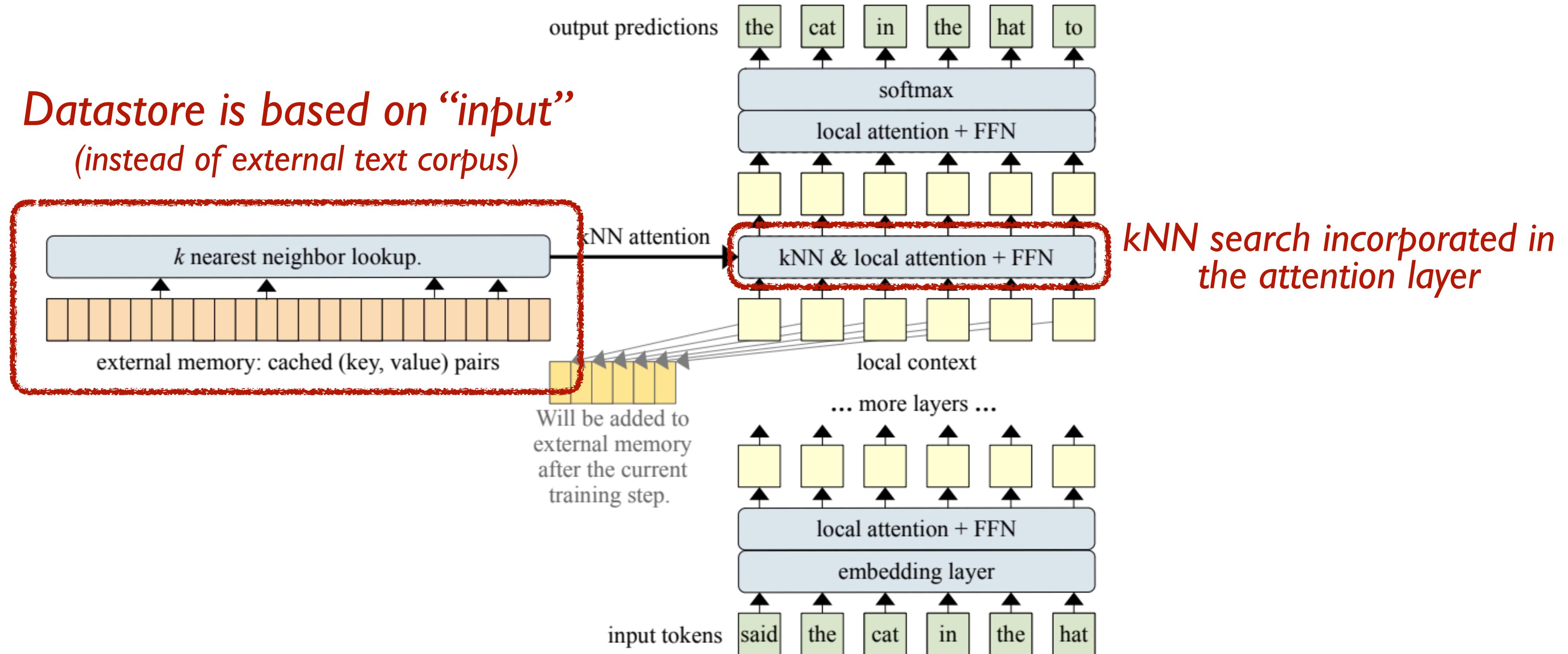
Retrieval for long-range LM



Wu et al. 2022. Memorizing Transformers (**Figure source**)
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval

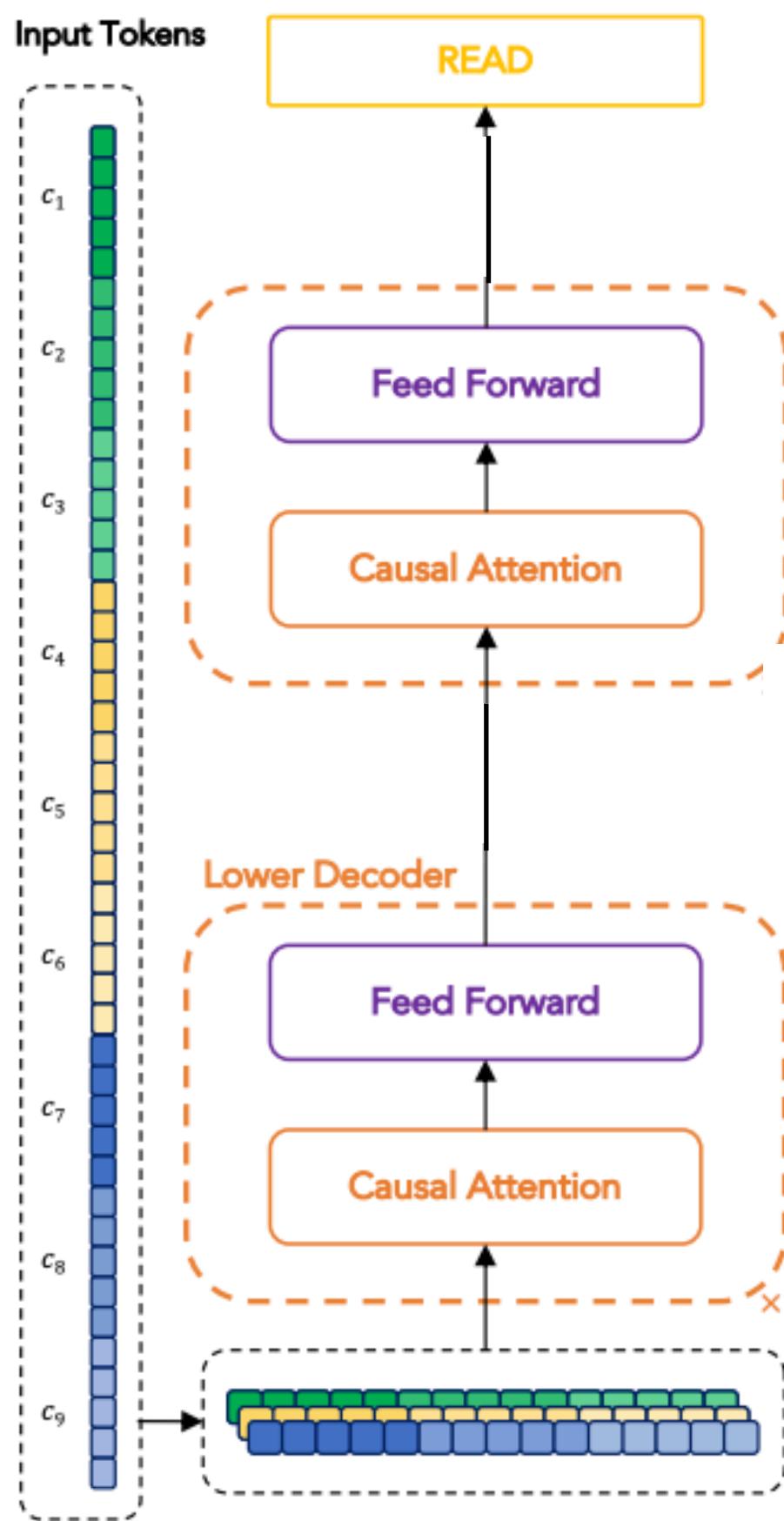
Retrieval for long-range LM

*Datastore is based on “input”
(instead of external text corpus)*



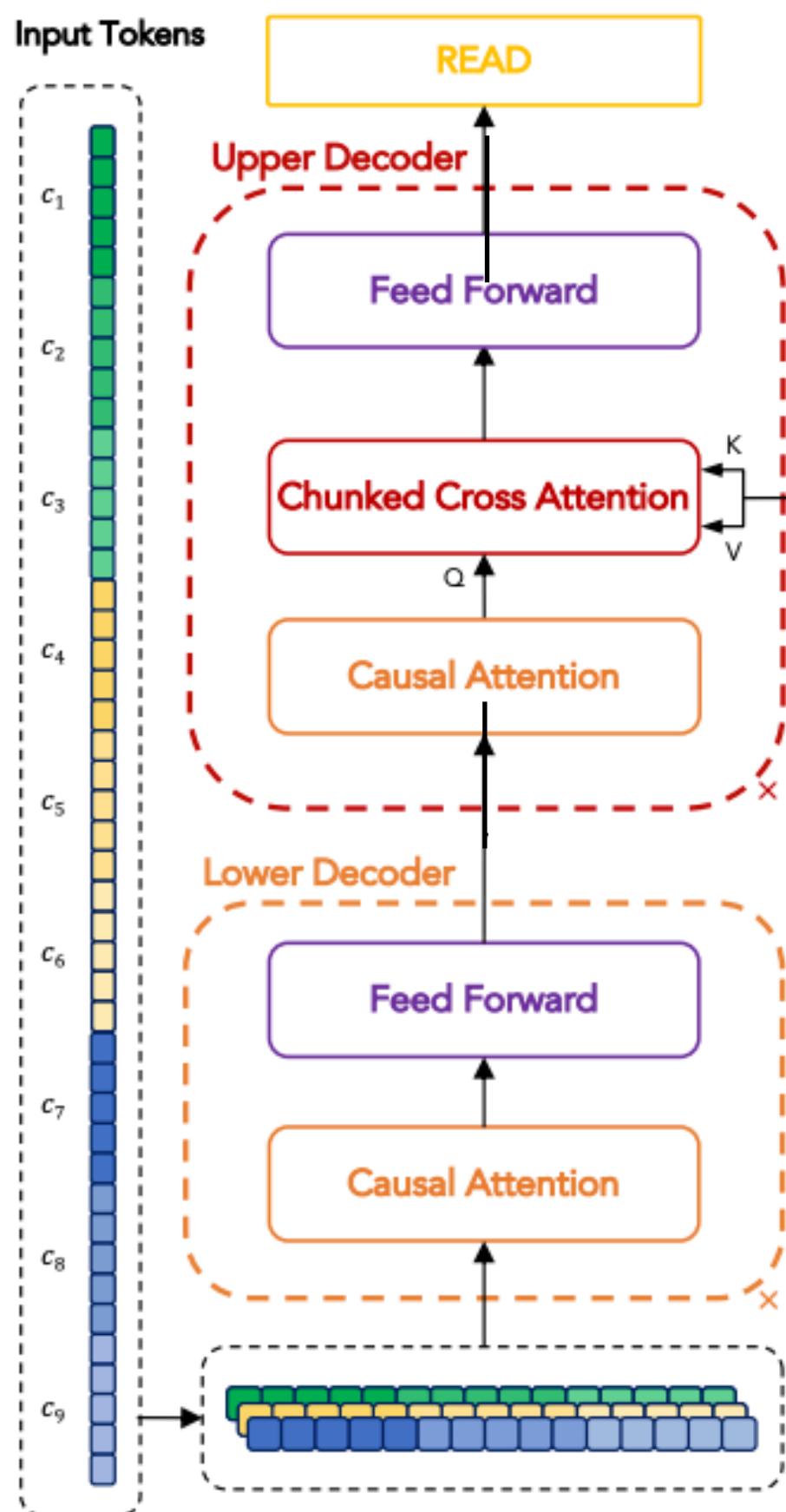
Wu et al. 2022. Memorizing Transformers (**Figure source**)
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval

Retrieval for long-range LM



Wu et al. 2022. Memorizing Transformers
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval (**Figure source**)

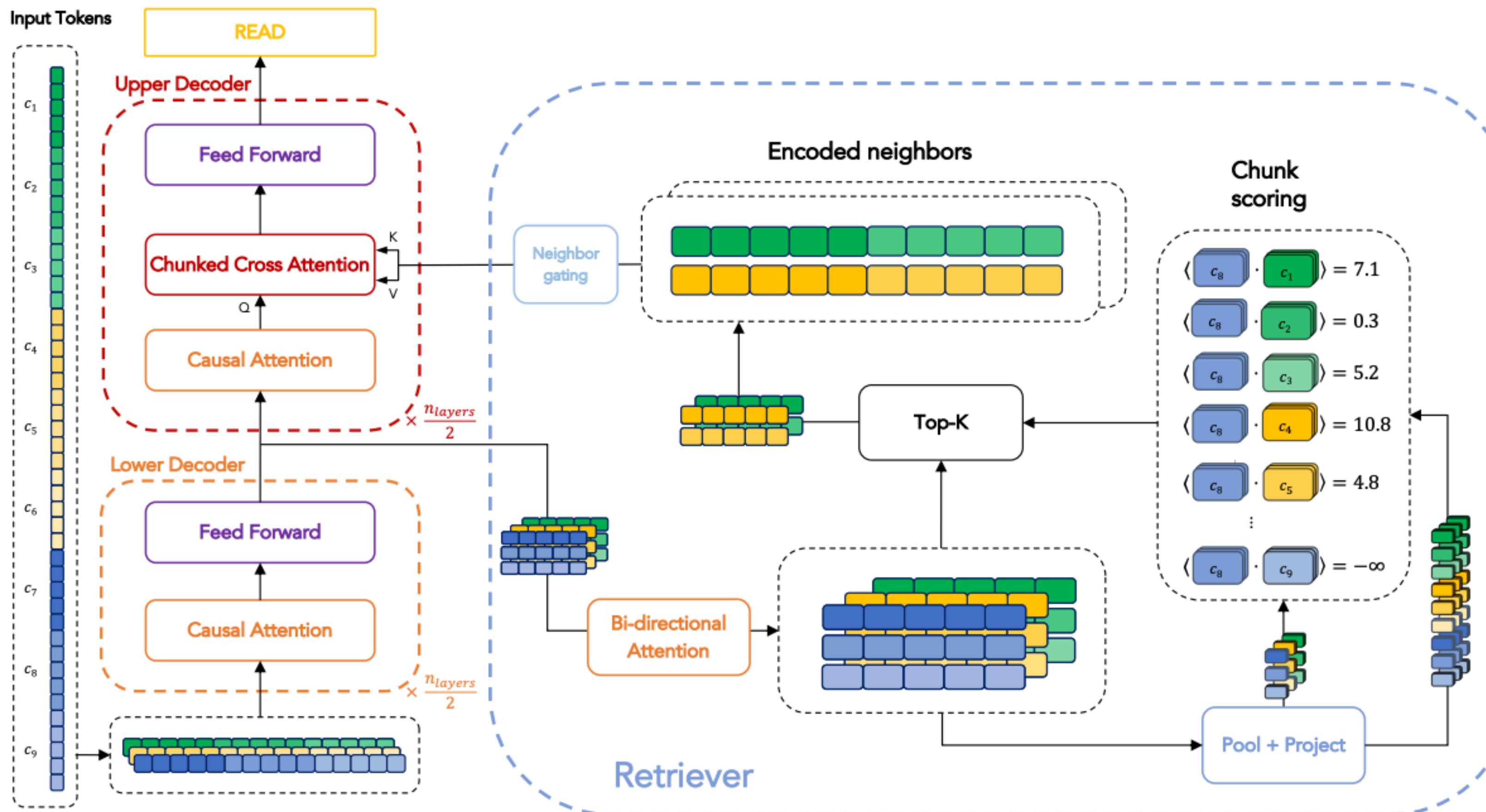
Retrieval for long-range LM



Chunked Cross Attention

Wu et al. 2022. Memorizing Transformers
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval (**Figure source**)

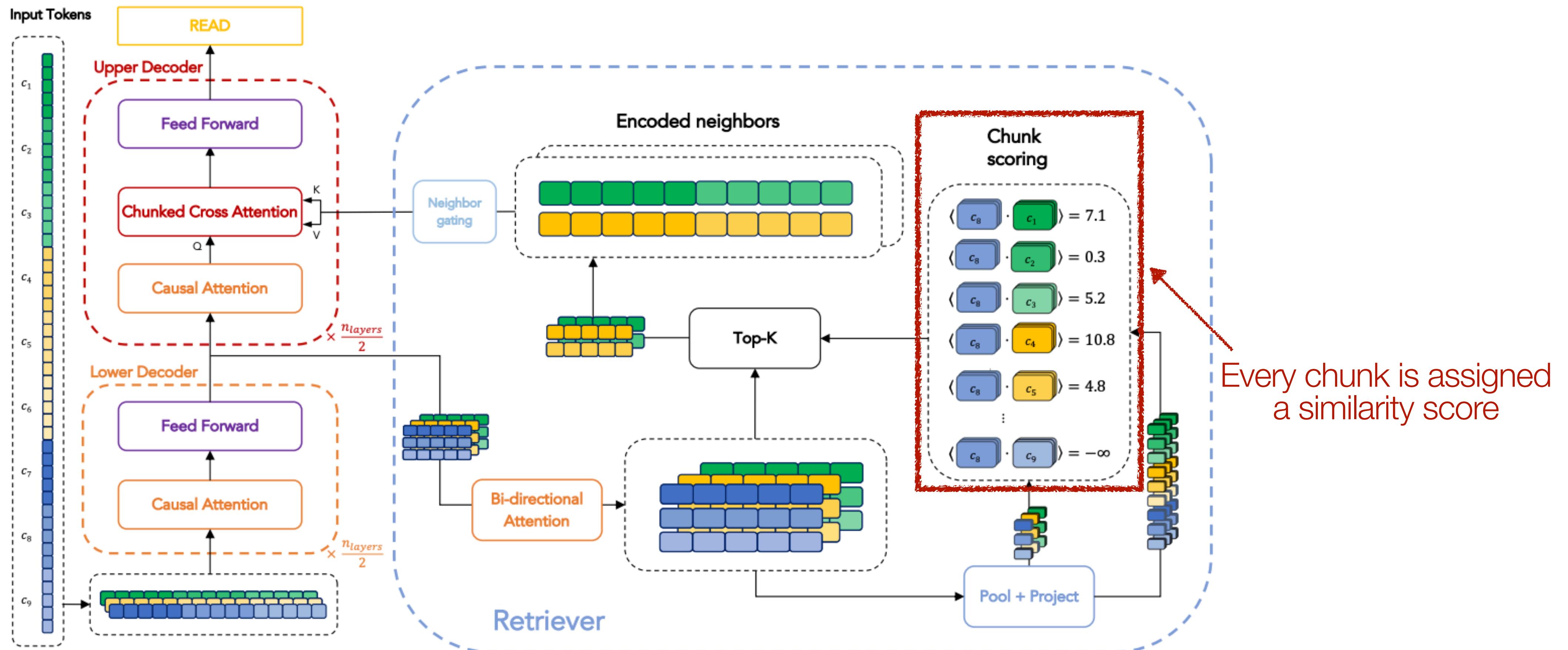
Retrieval for long-range LM



Wu et al. 2022. Memorizing Transformers

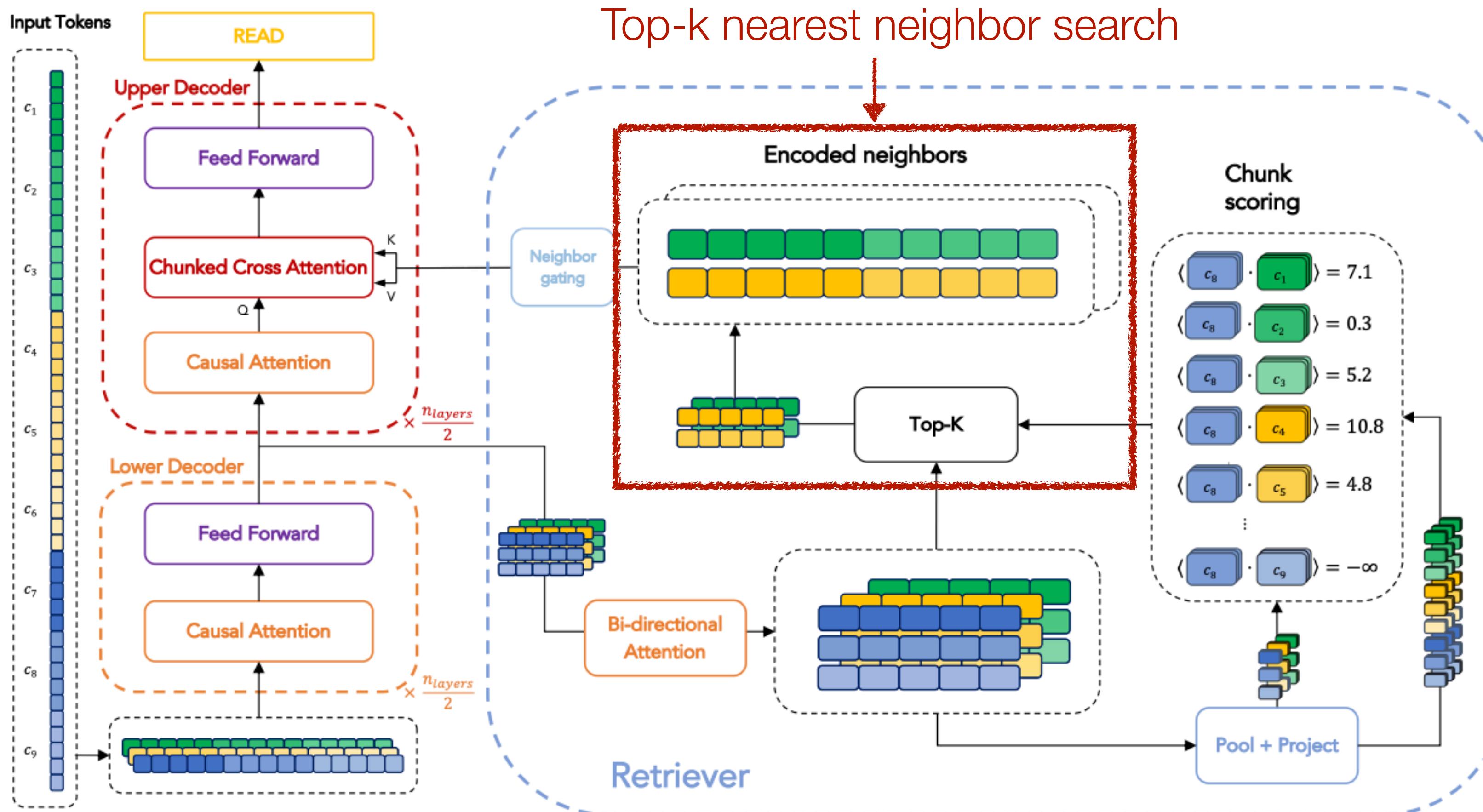
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
 Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval (**Figure source**)

Retrieval for long-range LM



Wu et al. 2022. Memorizing Transformers
 Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
 Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval (**Figure source**)

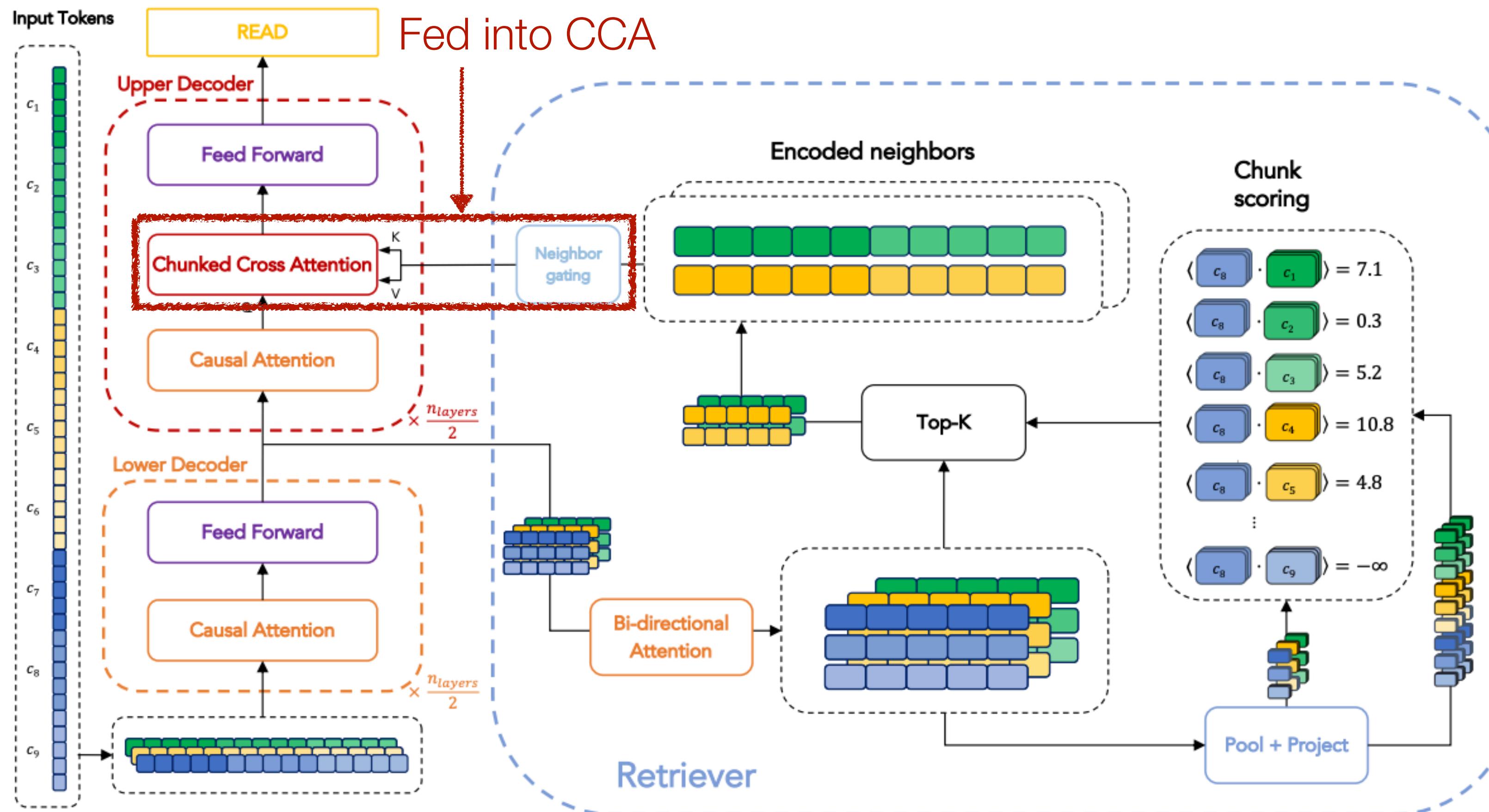
Retrieval for long-range LM



Wu et al. 2022. Memorizing Transformers

Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
 Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval (**Figure source**)

Retrieval for long-range LM



Wu et al. 2022. Memorizing Transformers

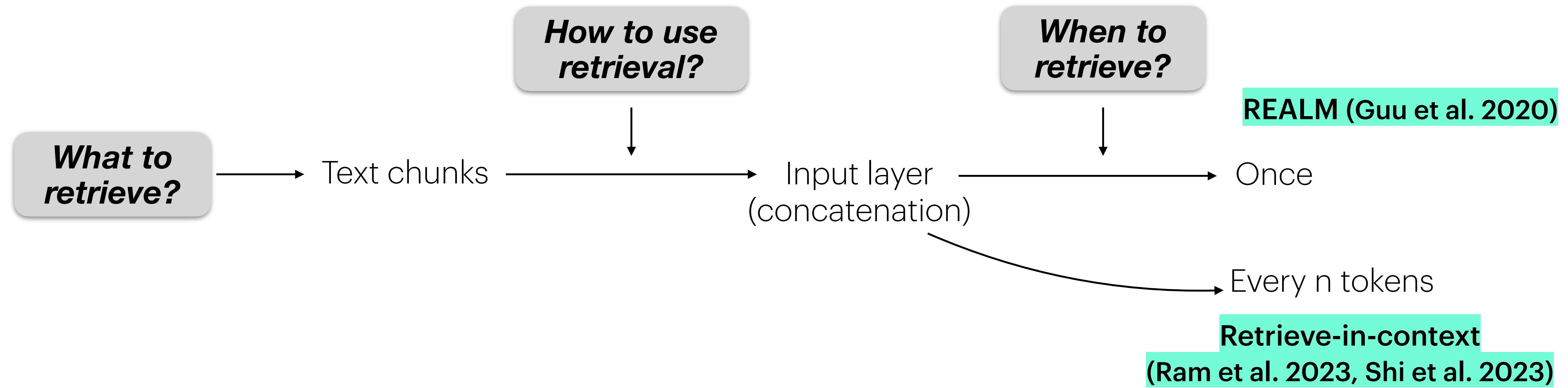
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
 Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval (**Figure source**)

Summary

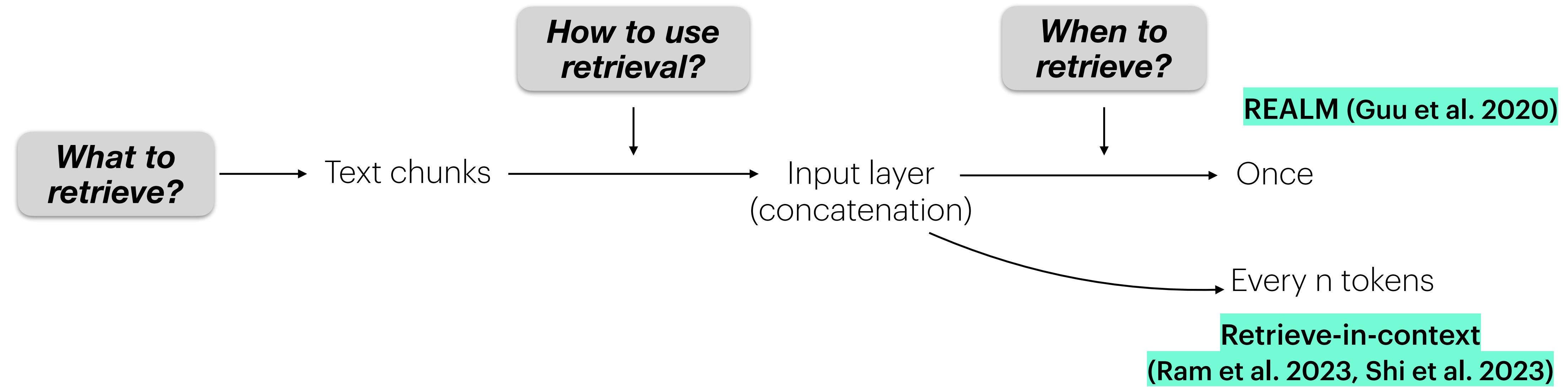
	What do retrieve?	How to use retrieval?	When to retrieve?
REALM (Guu et al 2020)	Text chunks	Input layer	Once
Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)	Text chunks	Input layer	Every n tokens
RETRO (Borgeaud et al. 2021)	Text chunks	Intermediate layers	Every n tokens
kNN-LM (Khandelwal et 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)
Entities as Experts (Fevry et al. 2020), Mention Memory (de Jong et al. 2022)	Entities or entity mentions	Intermediate layers	Every entity mentions
Wu et al. 2022, Bertsch et al. 2023, Rubin & Berant. 2023	Text chunks from the input	Intermediate layers	Once or every n tokens

Wrapping up

Wrapping up

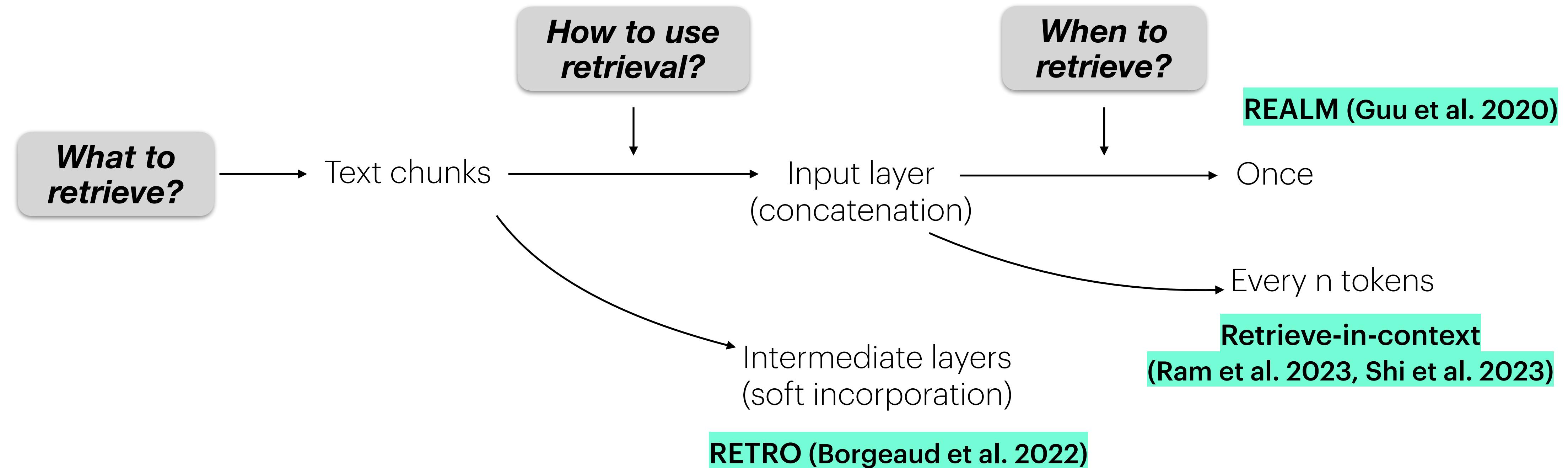


Wrapping up

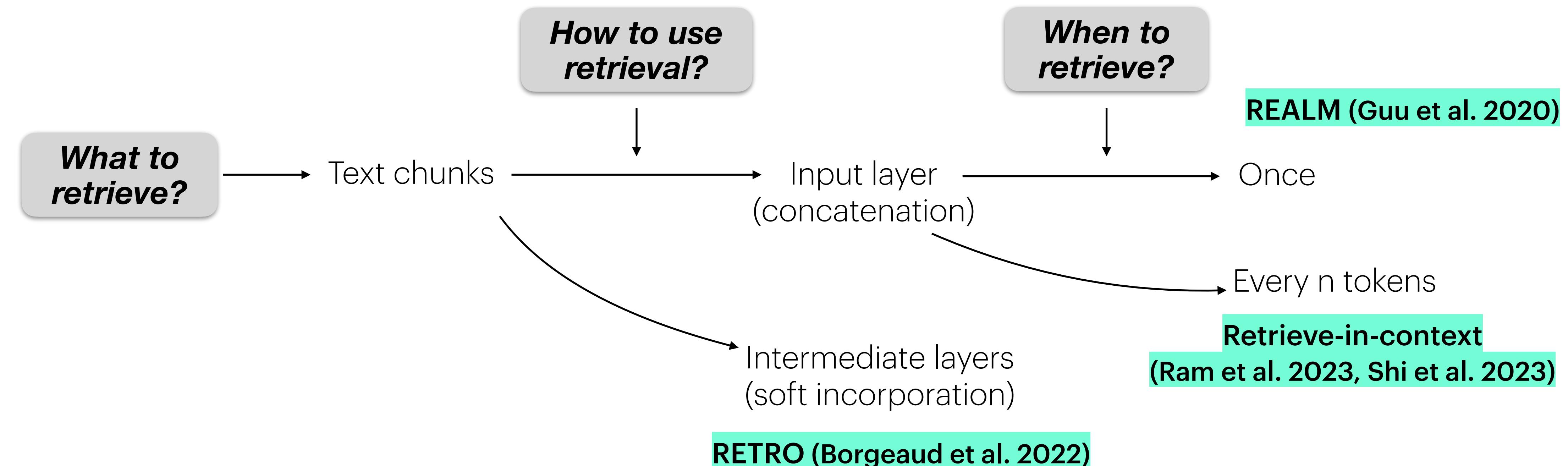


More frequent retrieval = better in performance, but slower

Wrapping up

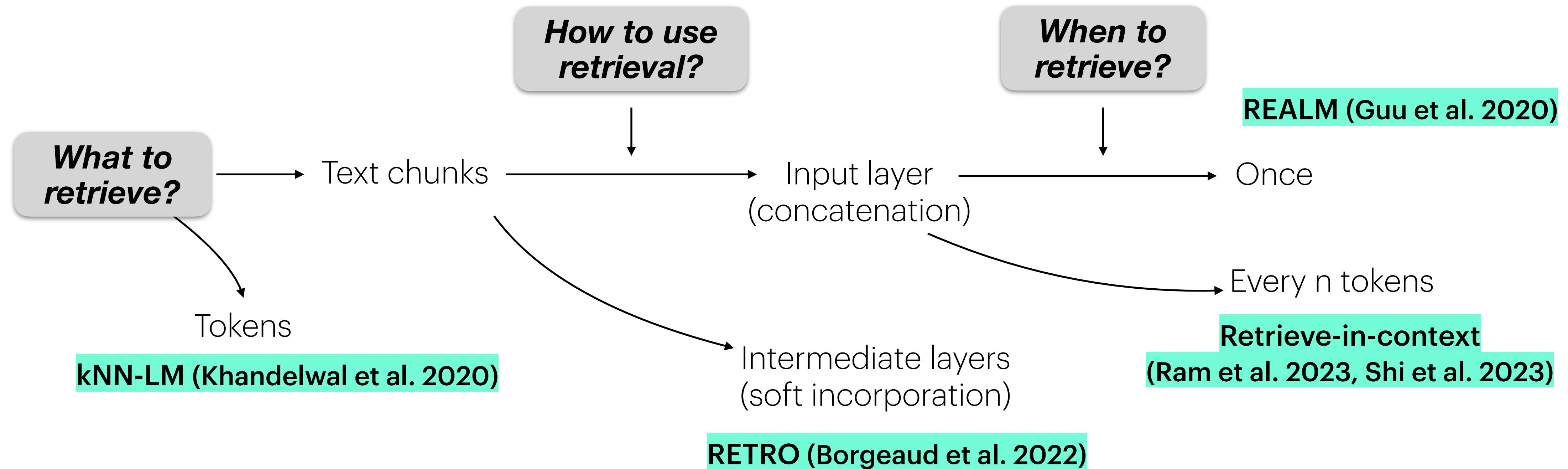


Wrapping up

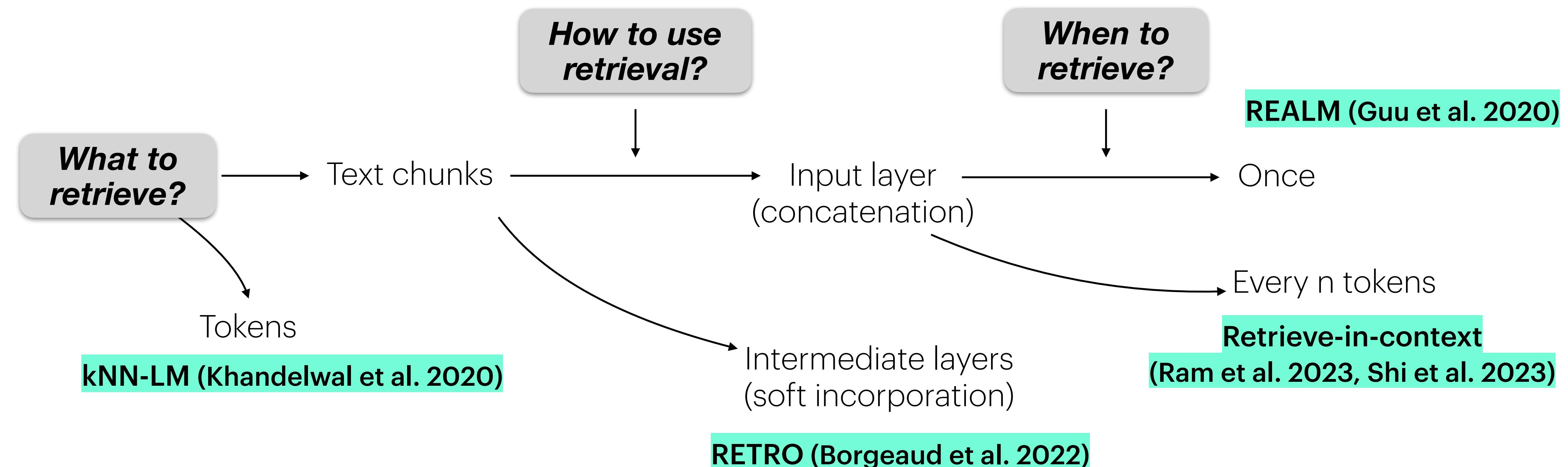


- Input layer: Simple but can be slower
- Intermediate layers: More complex (need training) but can be designed to be more efficient

Wrapping up

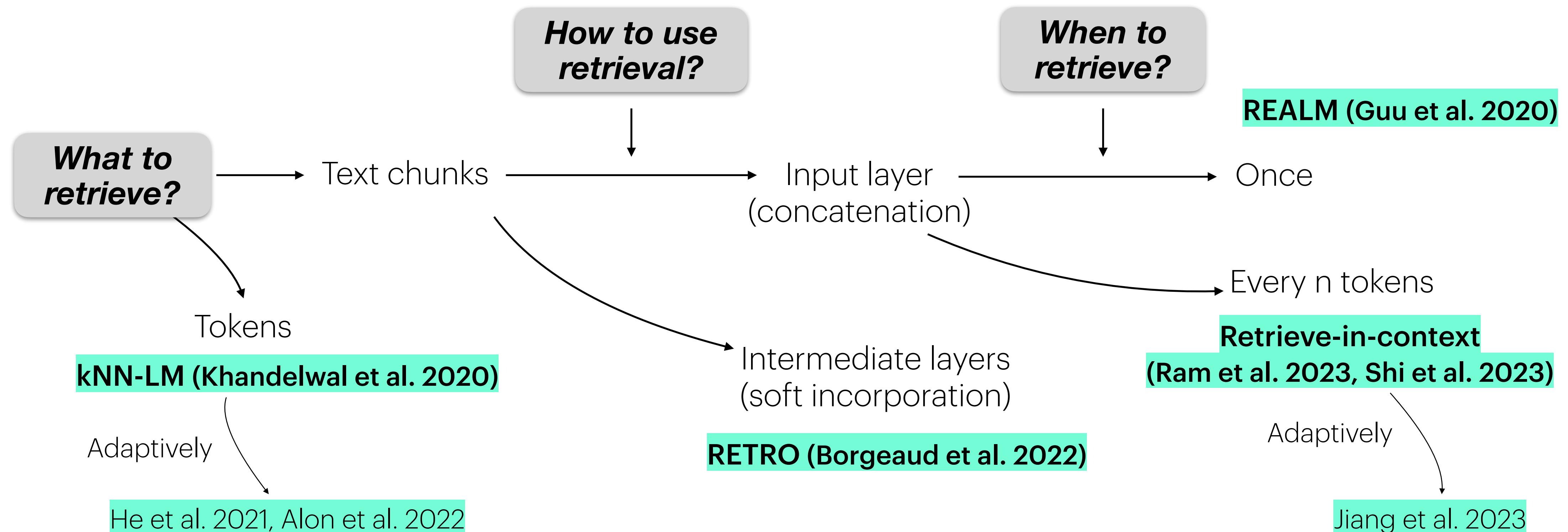


Wrapping up



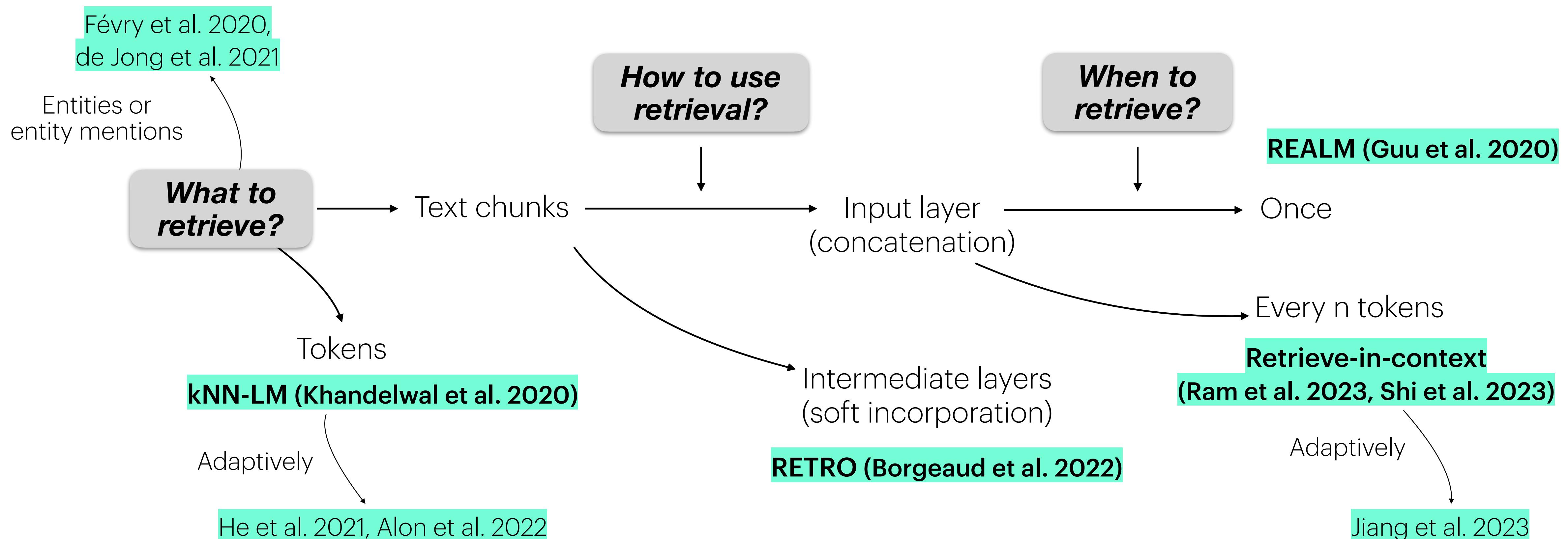
- Text blocks: Datastore can be space-efficient, more computation
- Tokens: More fine-grained, compute-efficient, but datastore can be space-expensive

Wrapping up



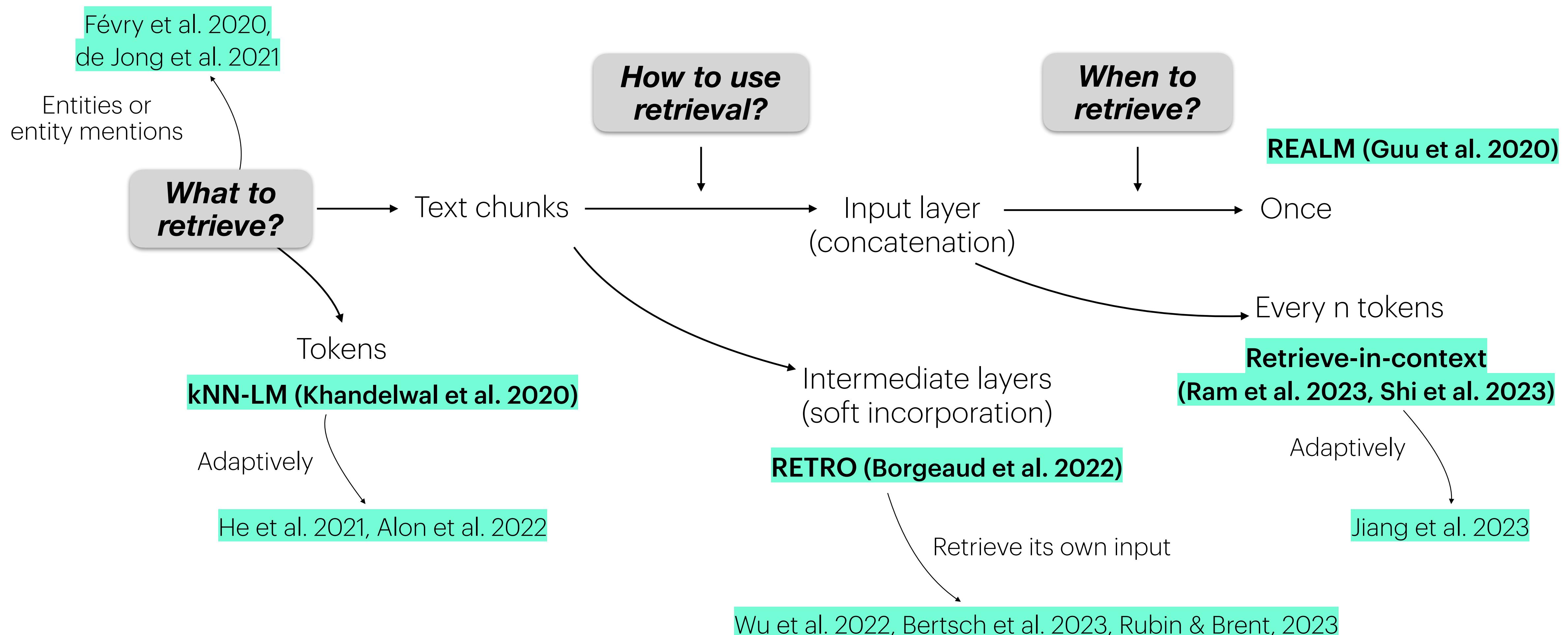
Adaptive retrieval can improve efficiency

Wrapping up



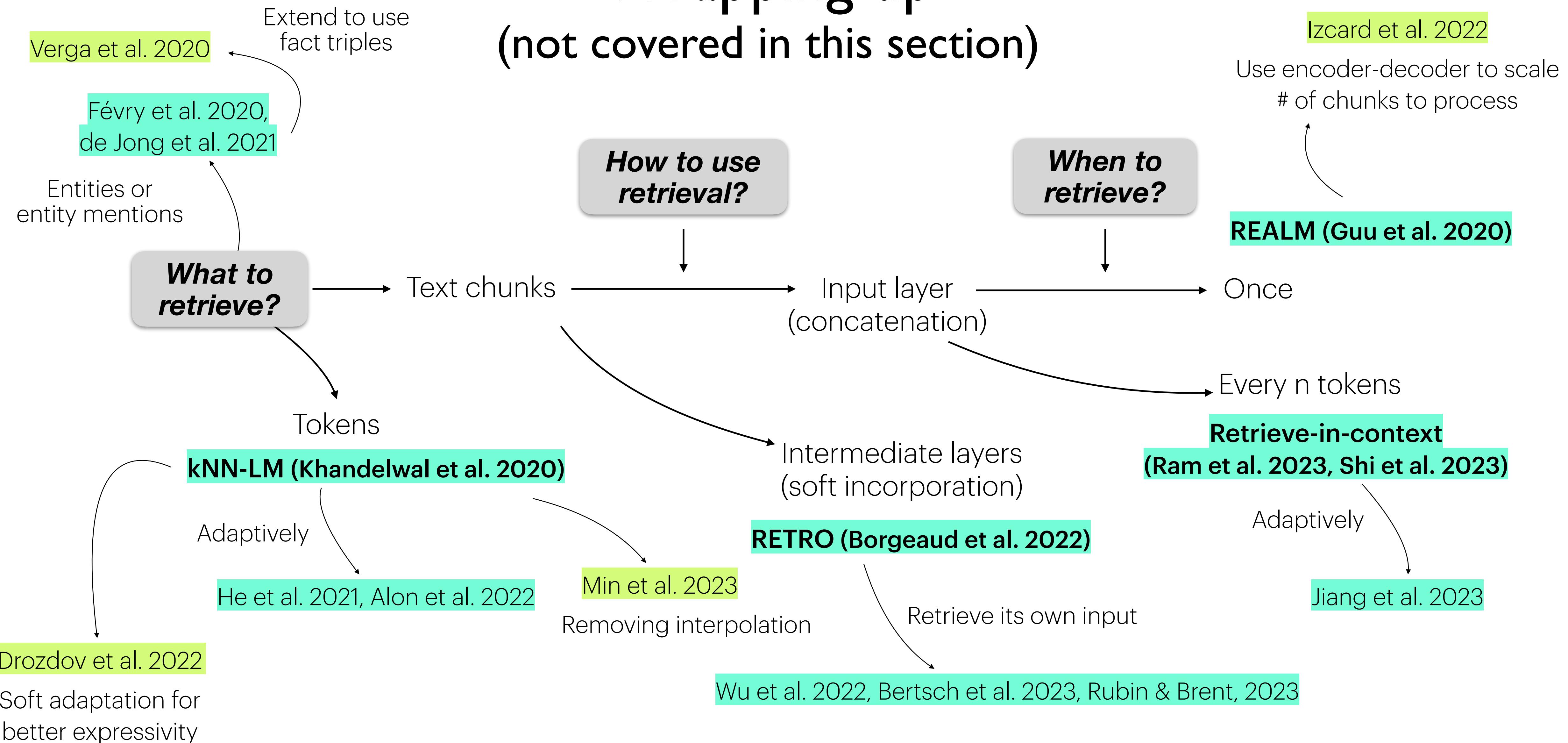
Entities or entity mentions instead of every token or chunk

Wrapping up

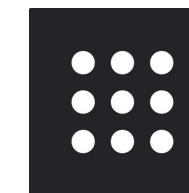
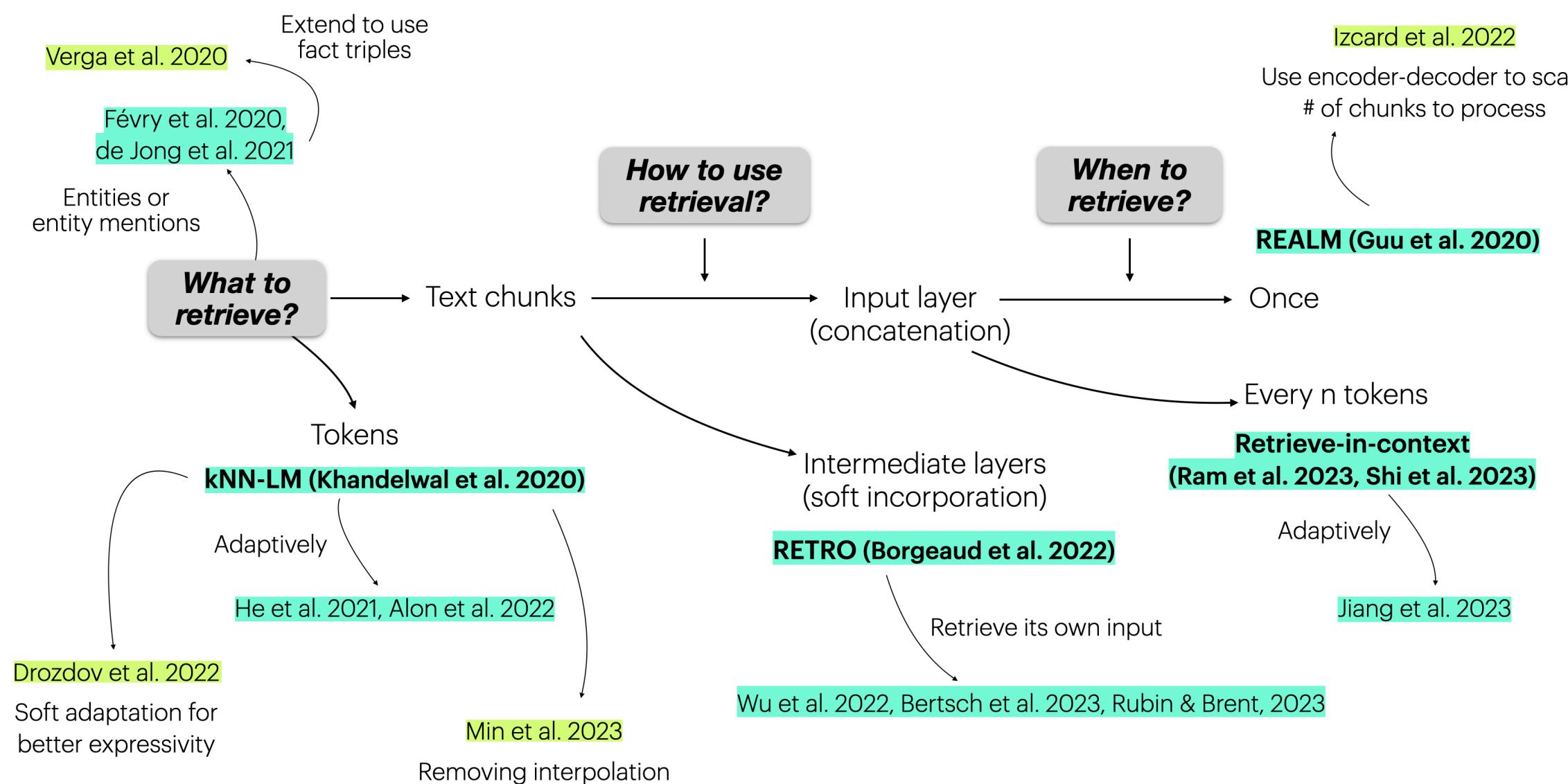


We can use a similar approach for long-sequence modeling

Wrapping up (not covered in this section)



Wrapping up



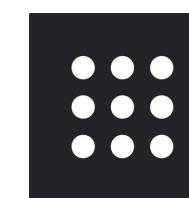
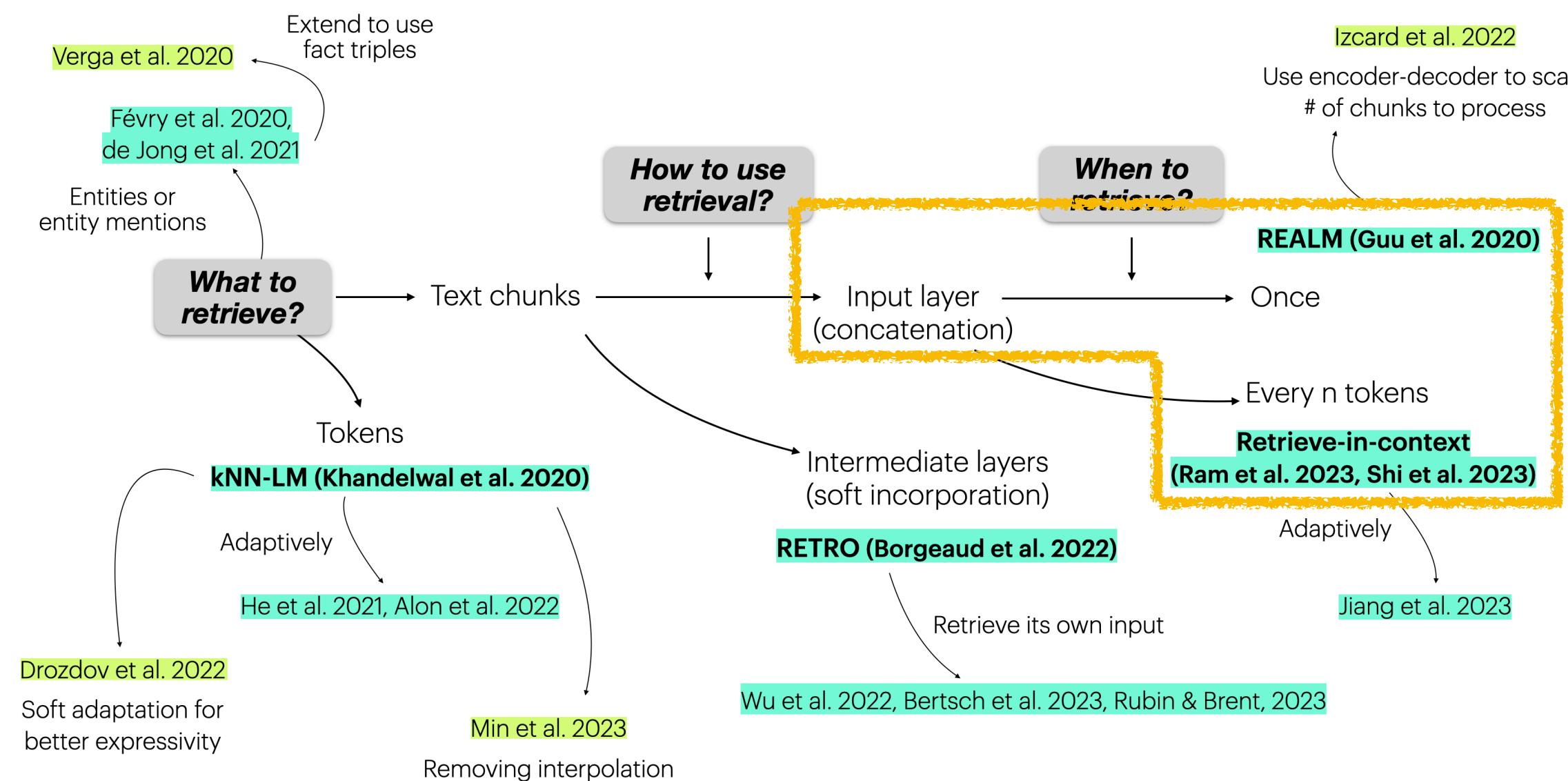
Perplexity

WebGPT



Chat GPT Extension

Wrapping up



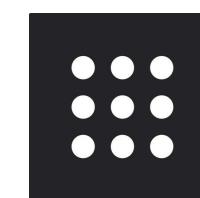
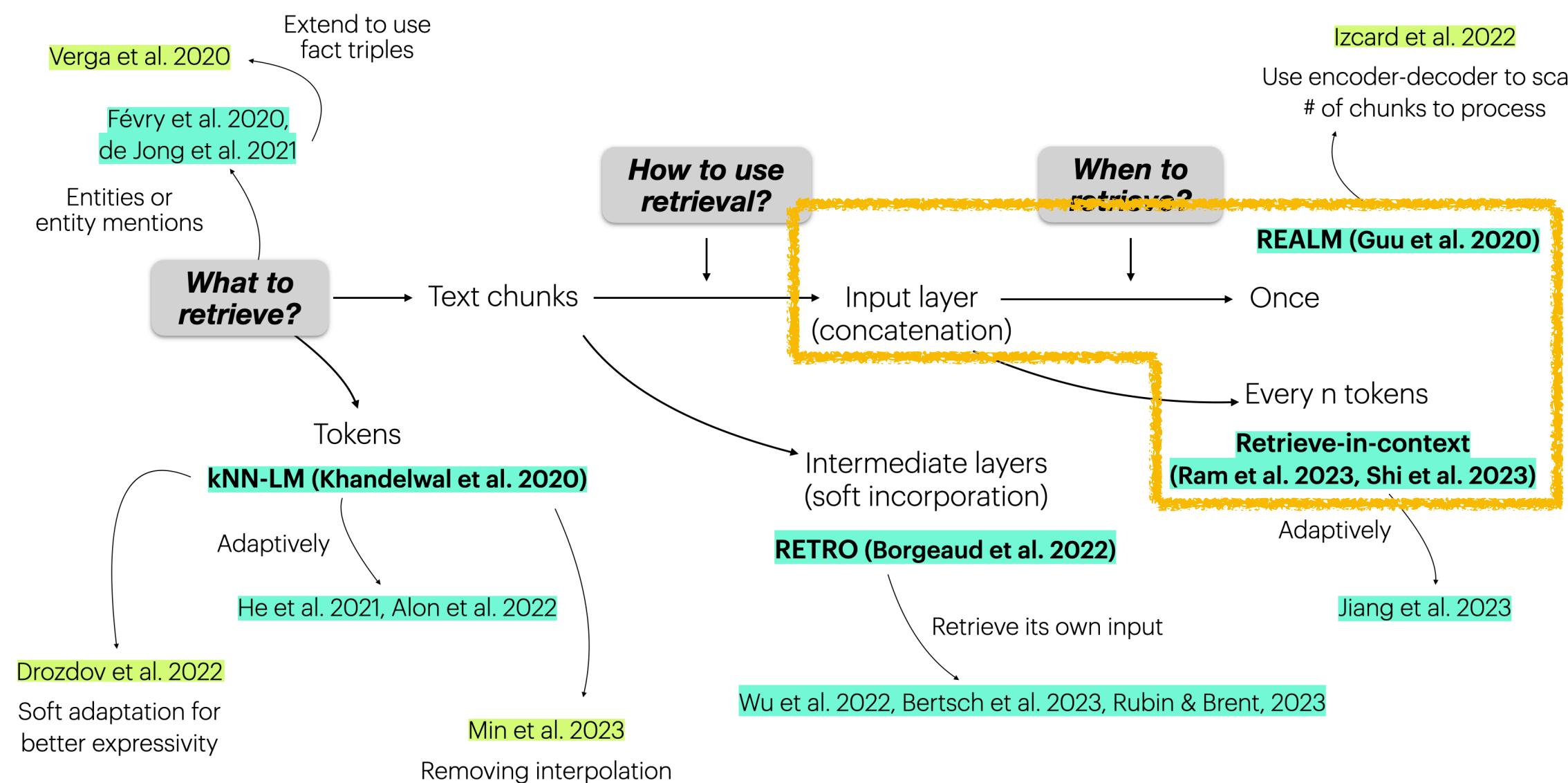
Perplexity

WebGPT



Chat GPT Extension

Wrapping up



Perplexity

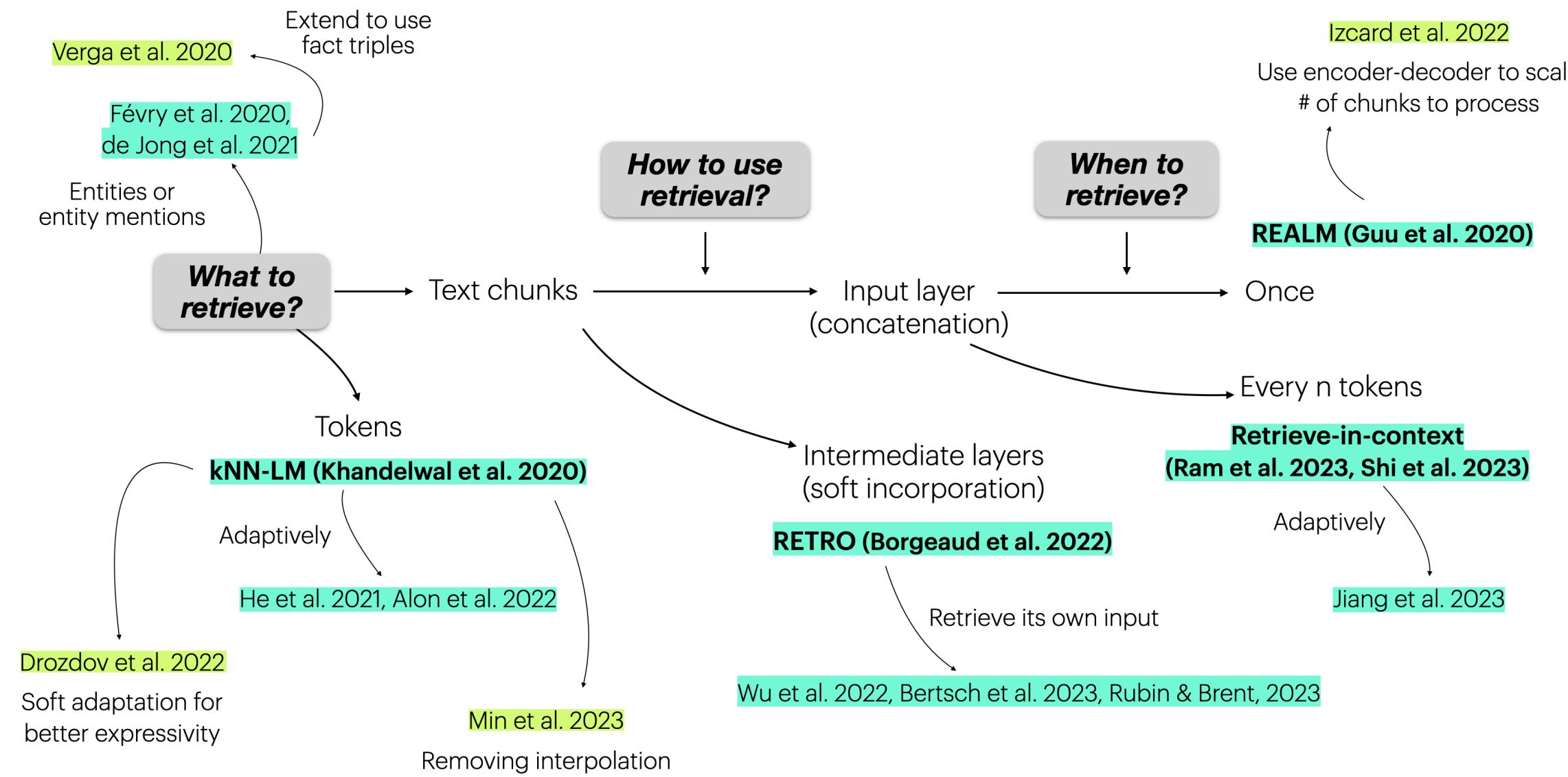
WebGPT



Chat GPT Extension

Still largely under-explored!

Wrapping up



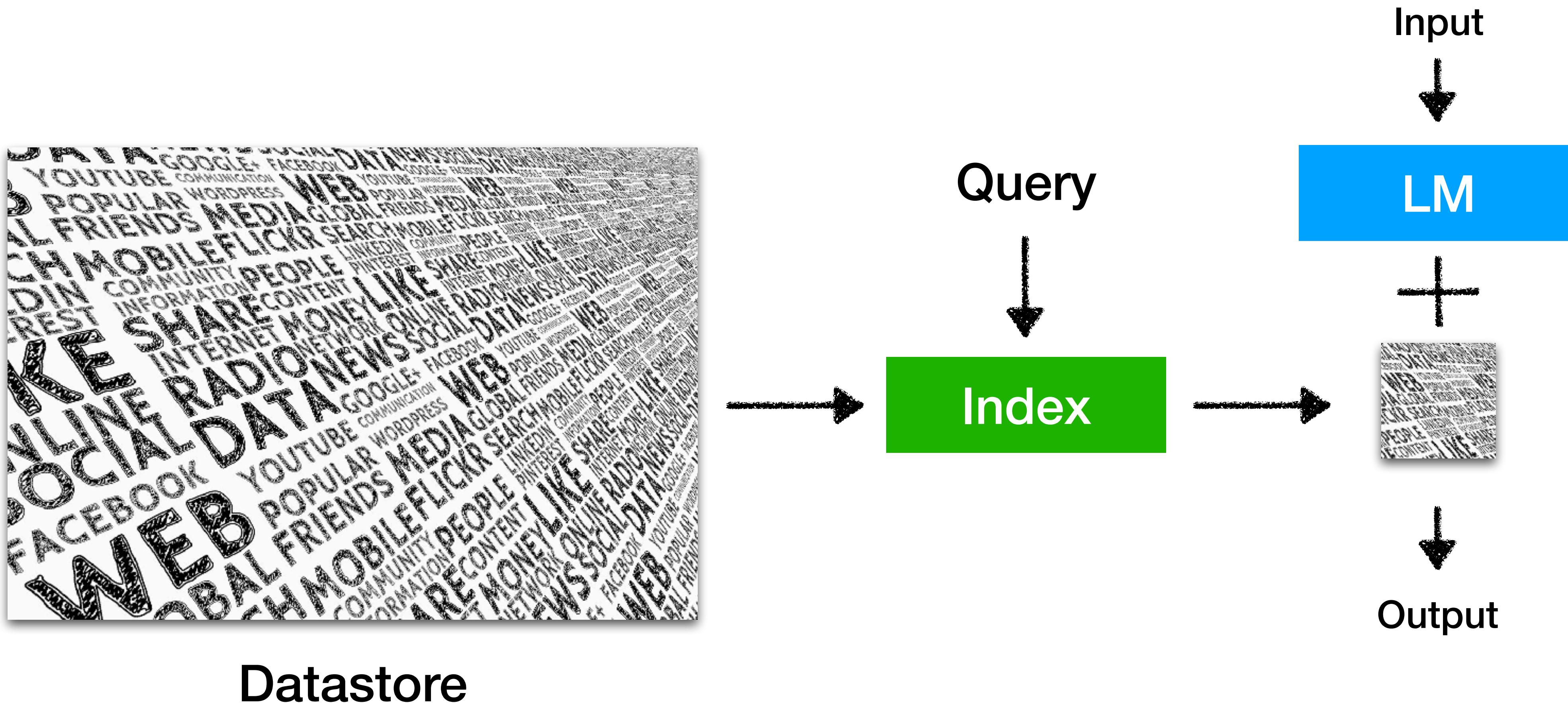
We didn't cover anything about training →

Section 4!

We briefly saw some results but not extensively
on downstream tasks → **Section 5!**

Section 4: Retrieval-based LMs: Training

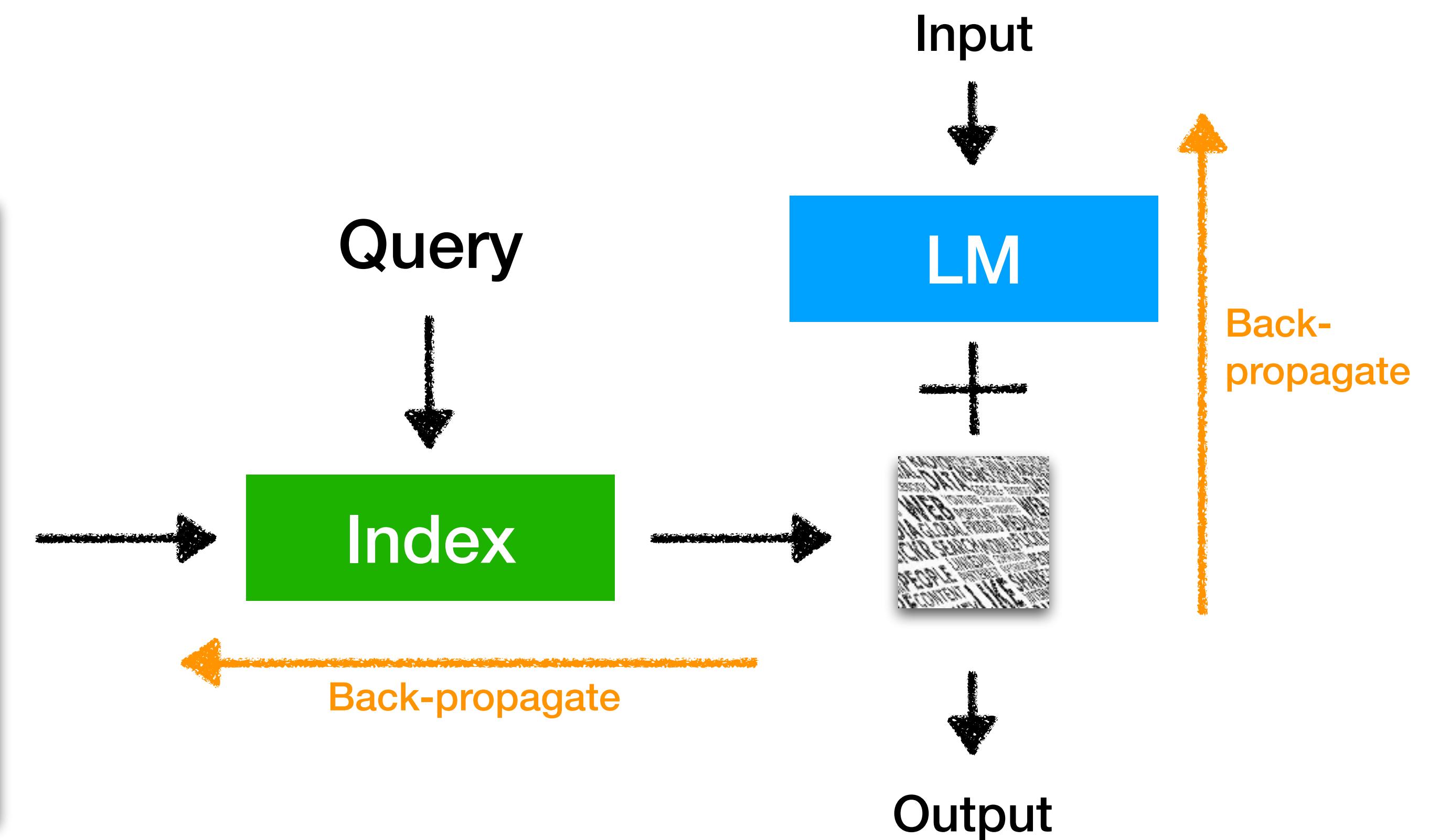
Retrieval-based LMs



Training retrieval-based LMs



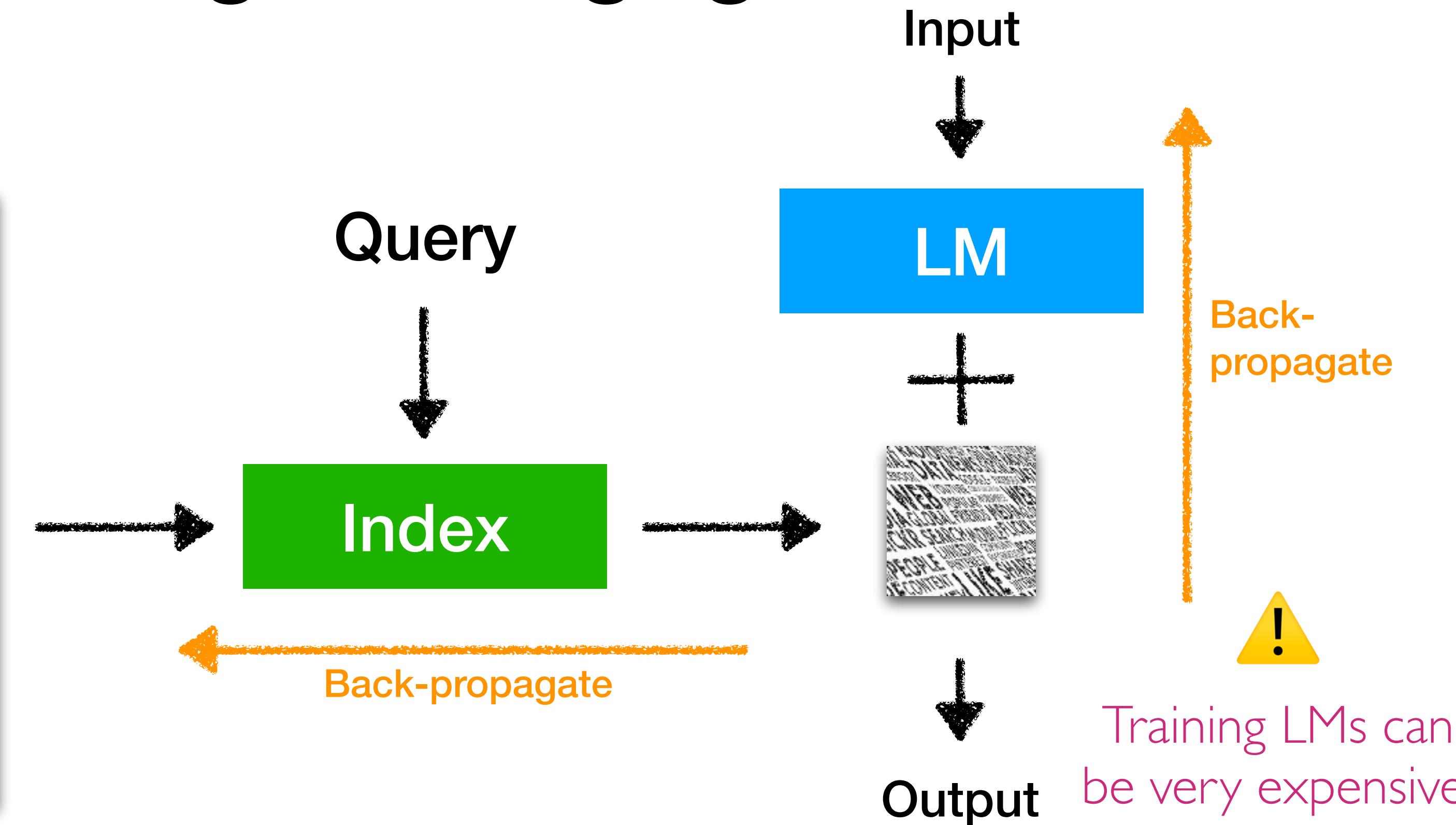
Datastore



Why is training challenging?



Datastore

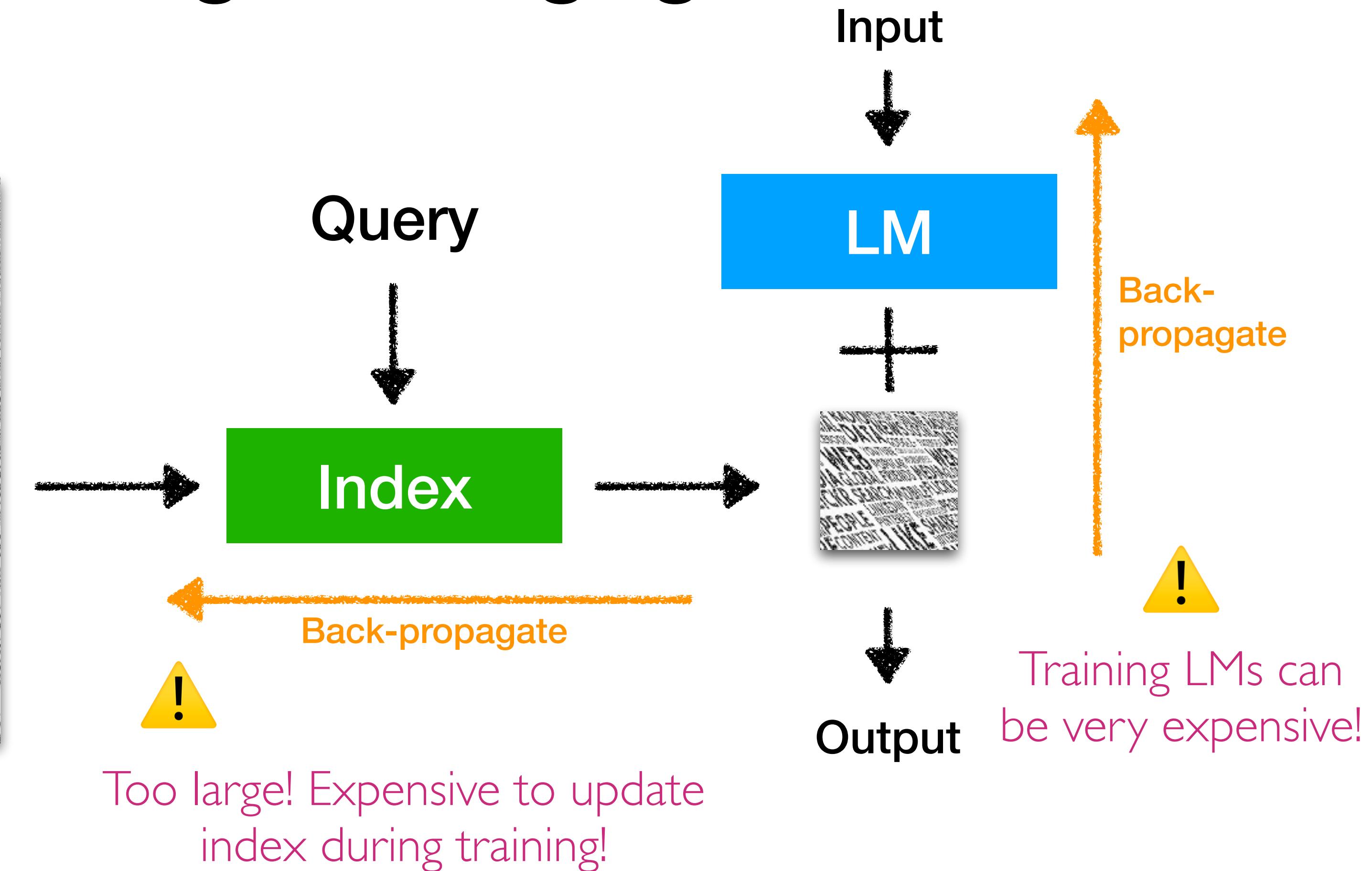


Training LMs can
be very expensive!

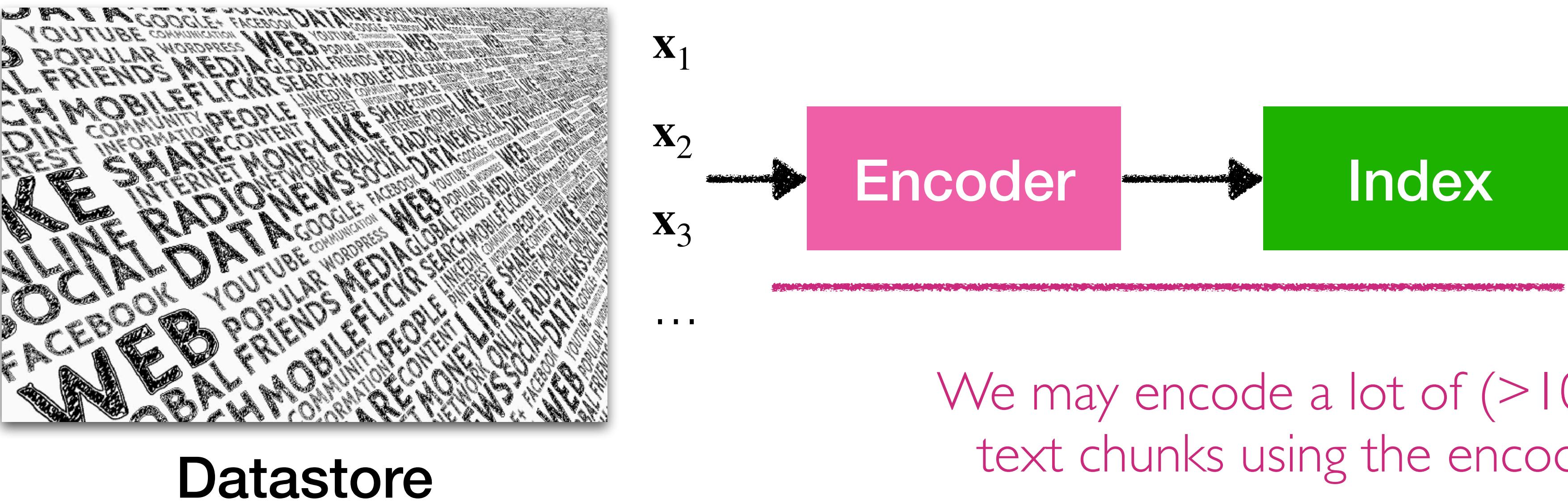
Why is training challenging?



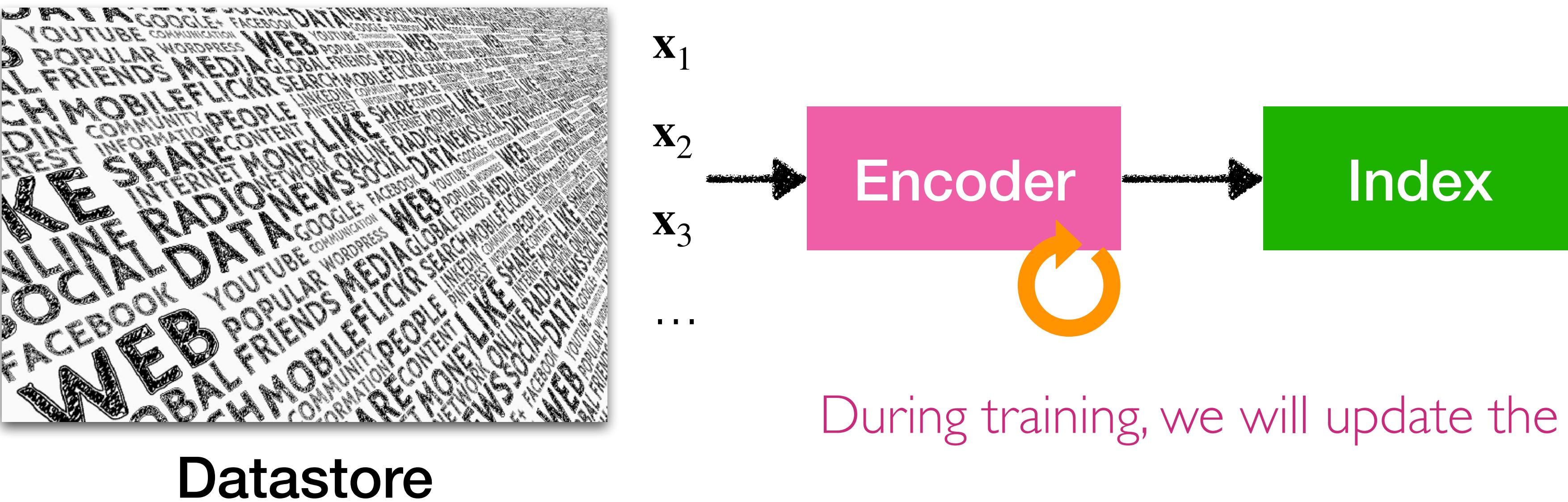
Datastore



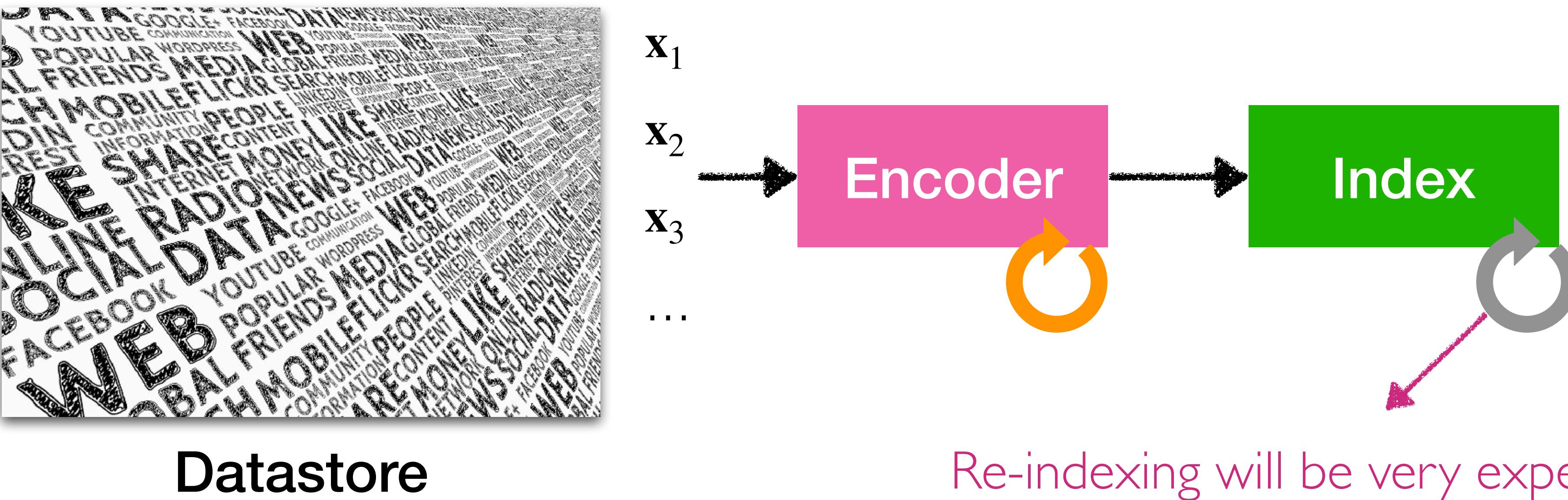
Challenges of updating retrieval models



Challenges of updating retrieval models



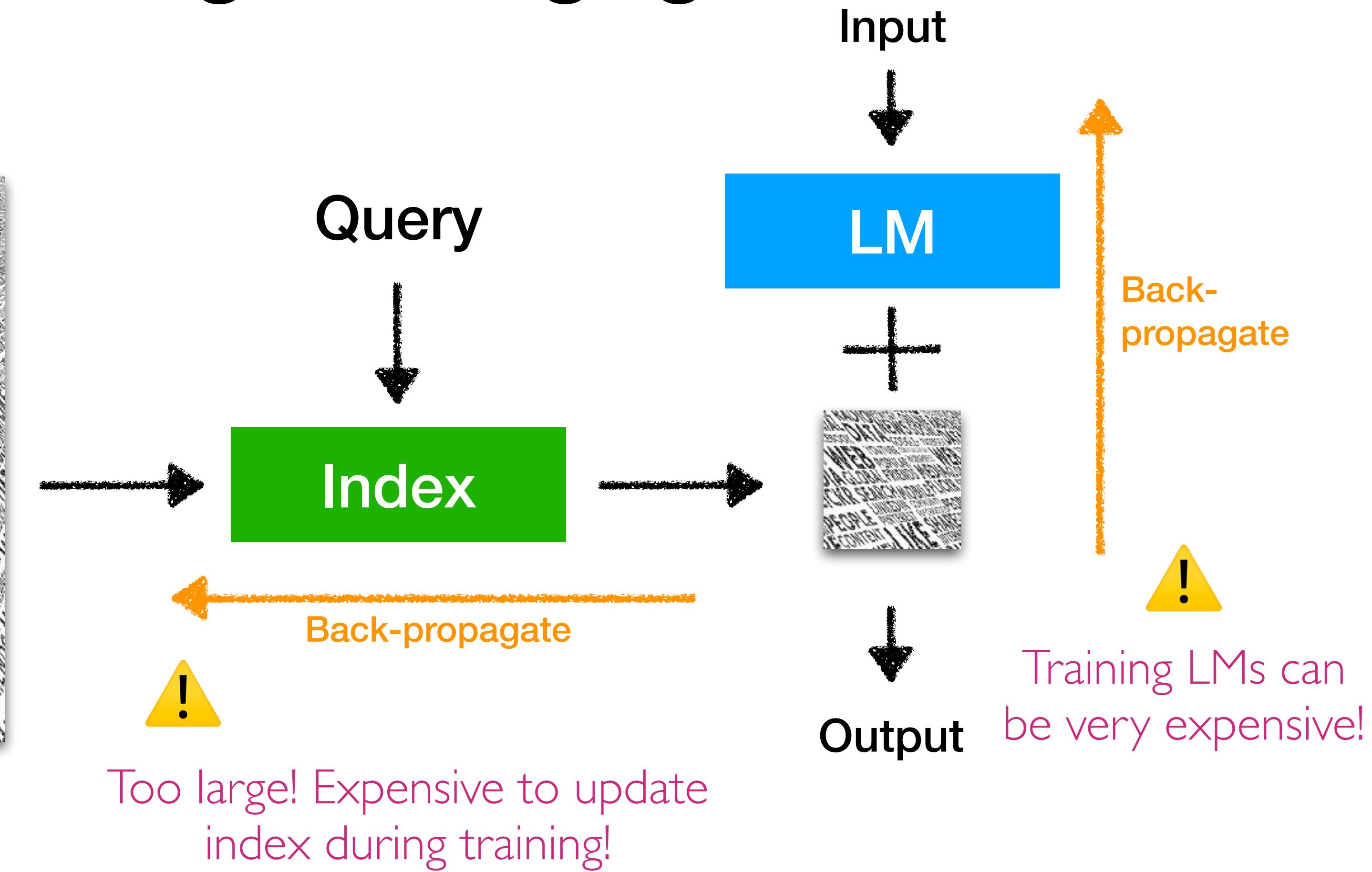
Challenges of updating retrieval models



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 methods for retrieval-based LMs

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

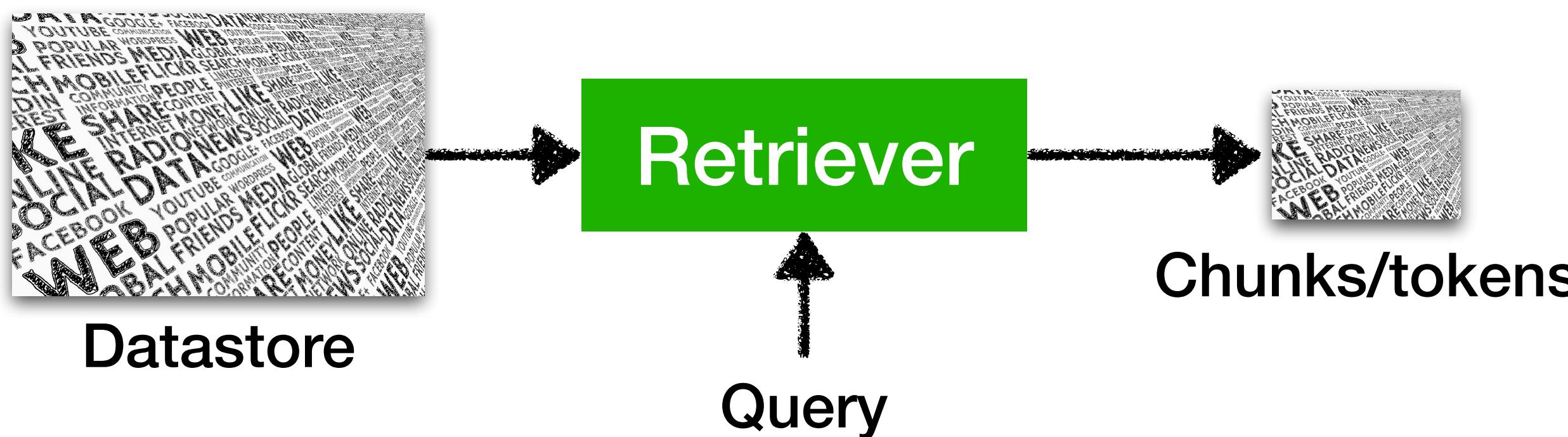
Independent training

Retrieval models and language models are trained **independently**

- Training language models



- Training retrieval models



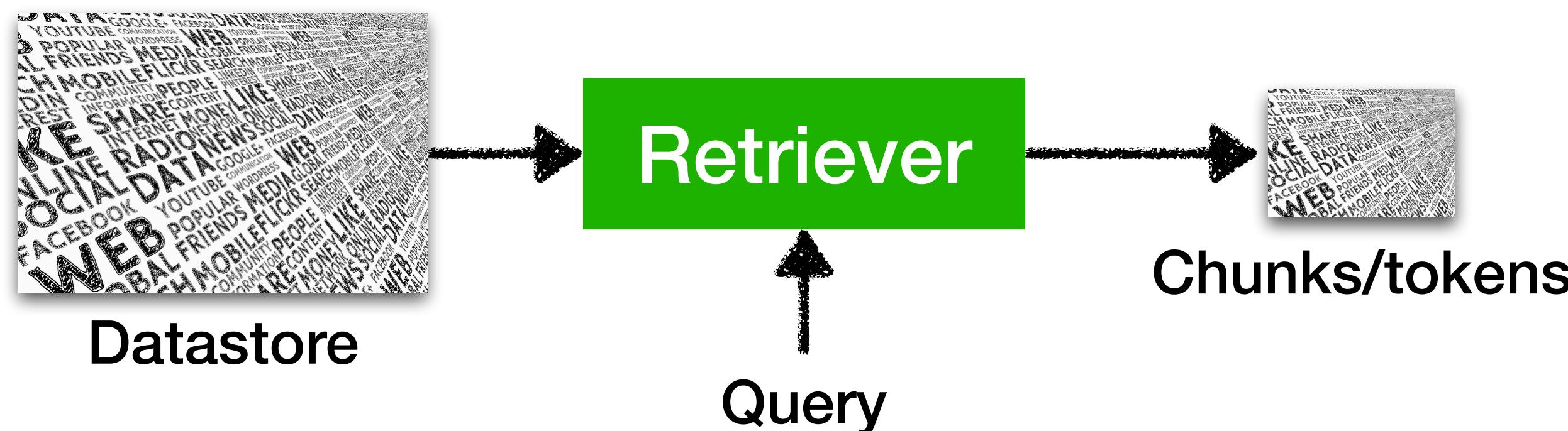
Independent training

Retrieval models and language models are trained **independently**

- Training language models



- Training retrieval models



Training language models



Minimize $-\log P_{\text{LM}}(y | x)$

Training language models



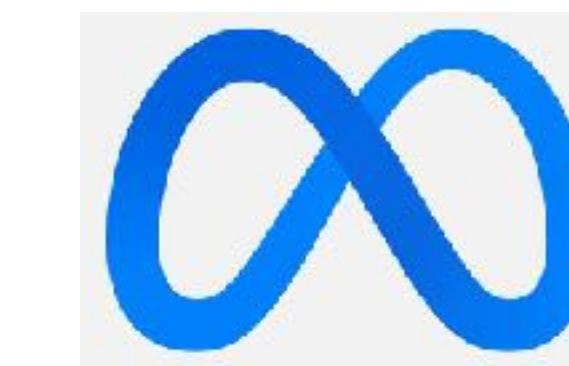
Minimize $-\log P_{\text{LM}}(y | x)$



GPT



PaLM



LLaMA



GPT-J

.....

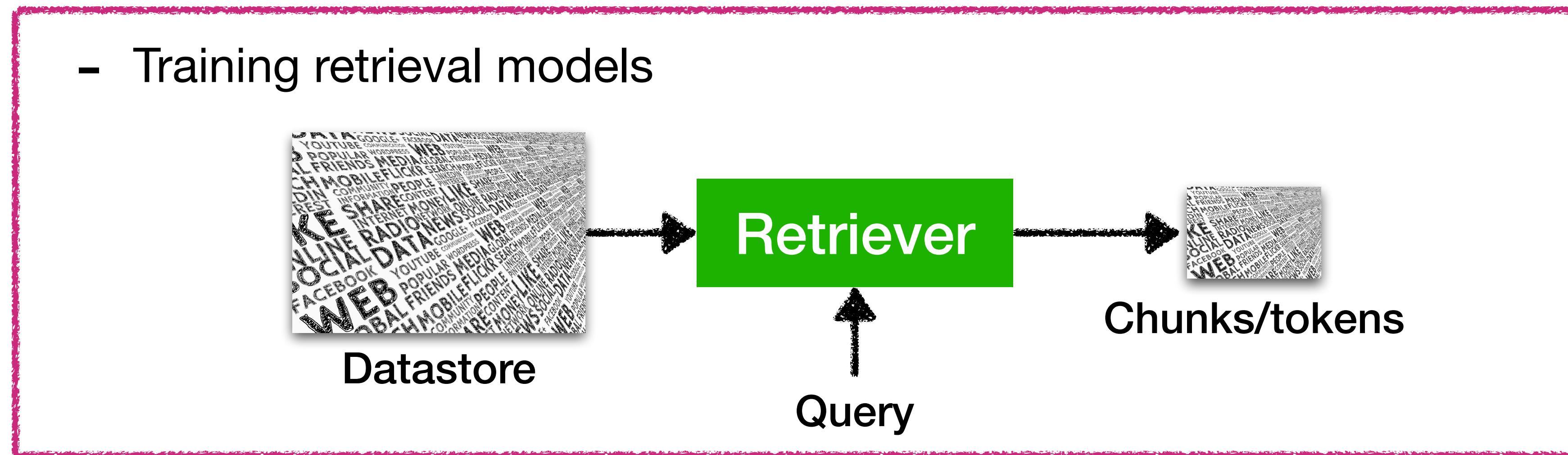
Independent training

Retrieval models and language models are trained **independently**

- Training language models



- Training retrieval models



Sparse retrieval models: TF-IDF / BM25

In 1997, Apple merged with NeXT,
and Steve Jobs became CEO of ...

[0, 0, 0.4, 0, 0.8, 0.7, ...]

Jobs returned to Apple as CEO
after the company's acquisition ...

[0, 1.2, 0.4, 0, 0.8, 0, ...]

Text chunks

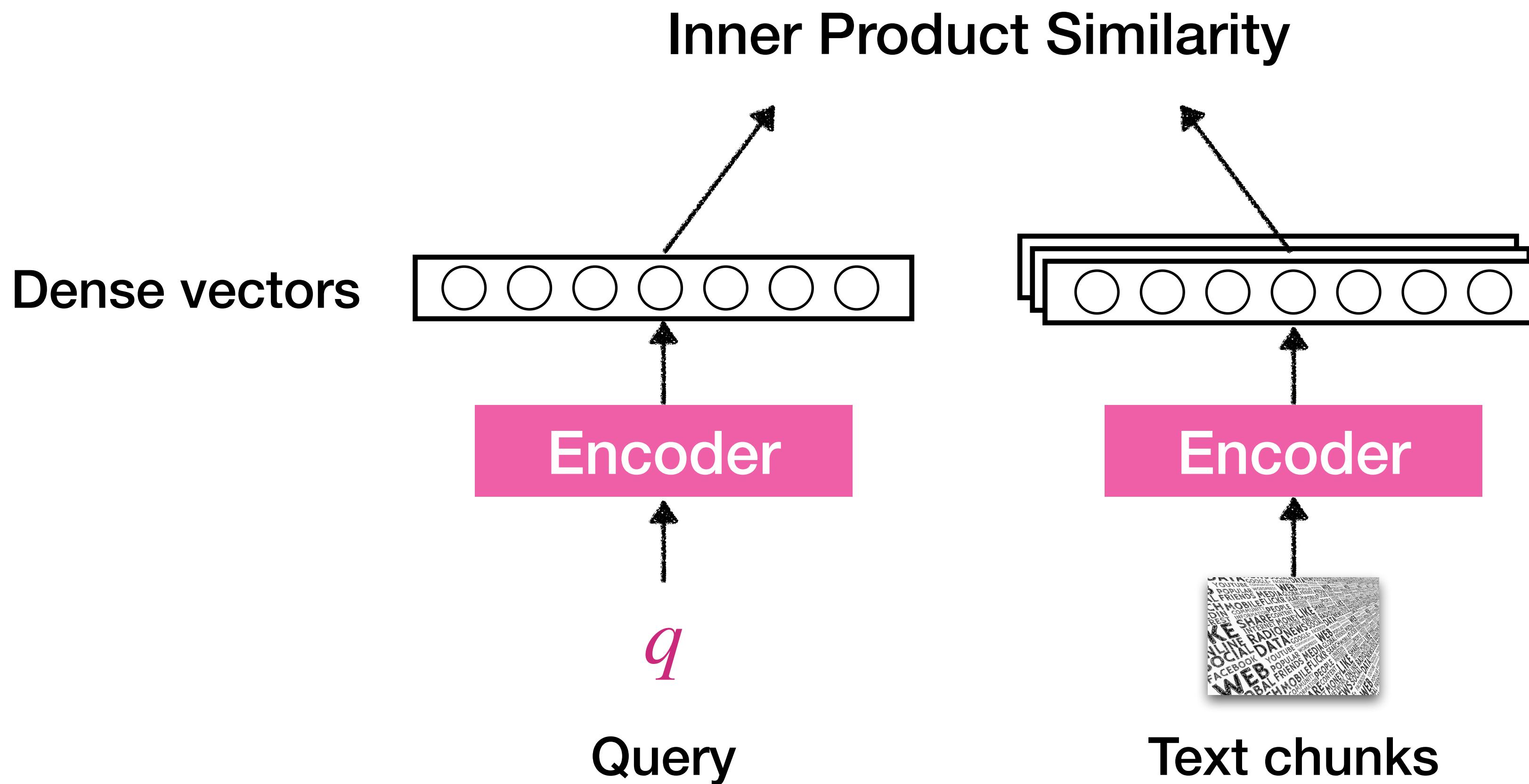
Sparse vectors



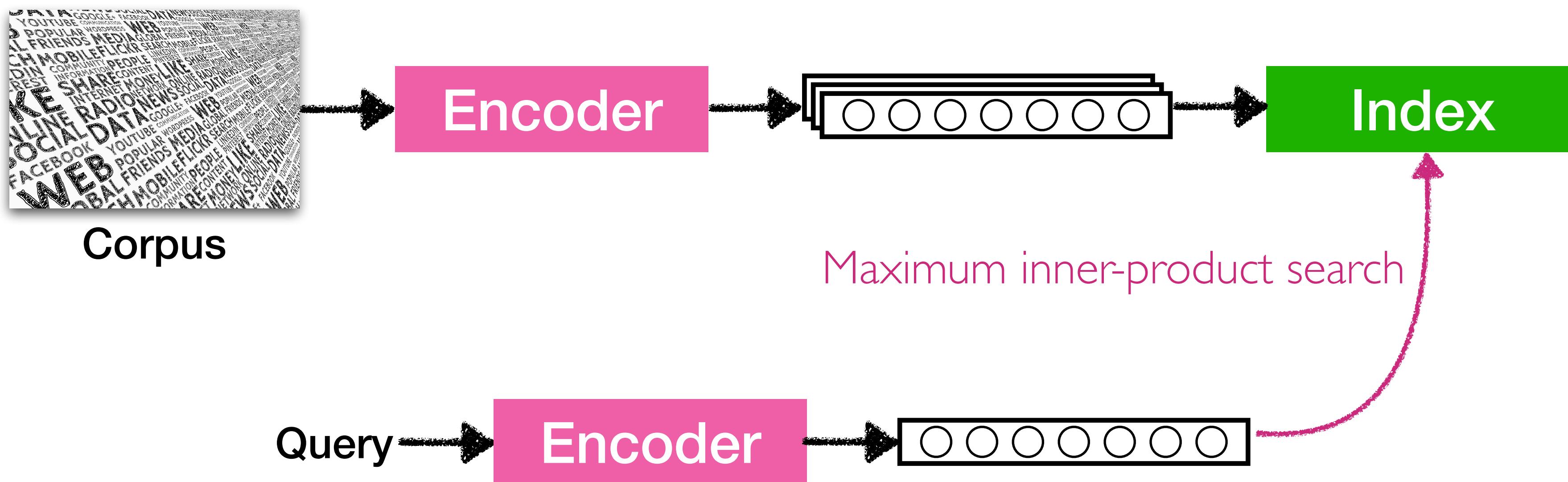
Lexical overlap

No training needed!

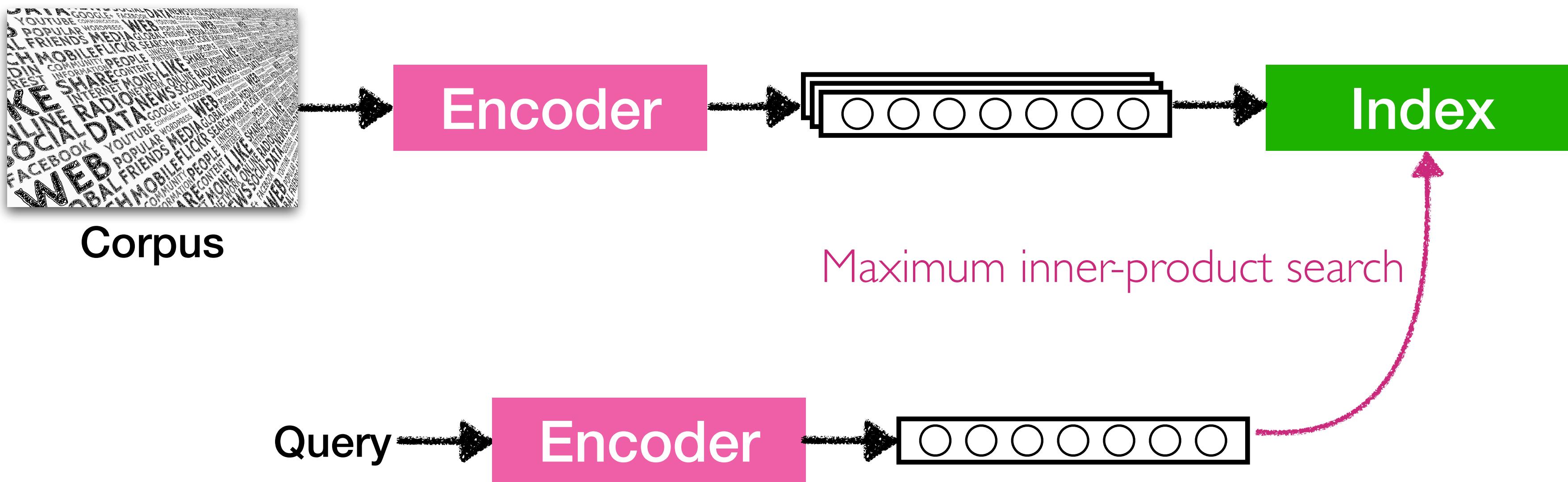
Dense retrieval models: DPR (Karpukhin et al. 2020)



Dense retrievers: Inference

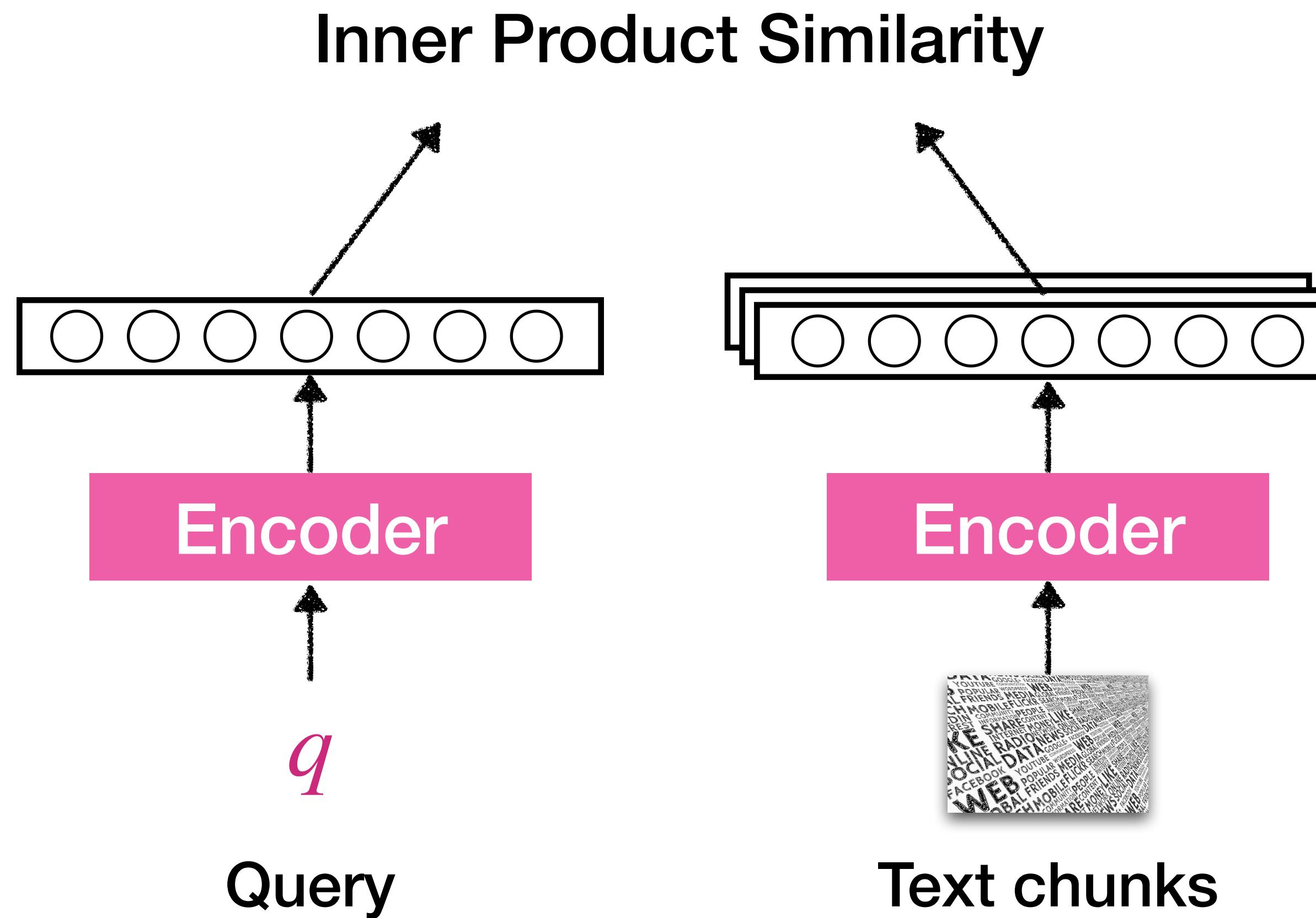


Dense retrievers: Inference

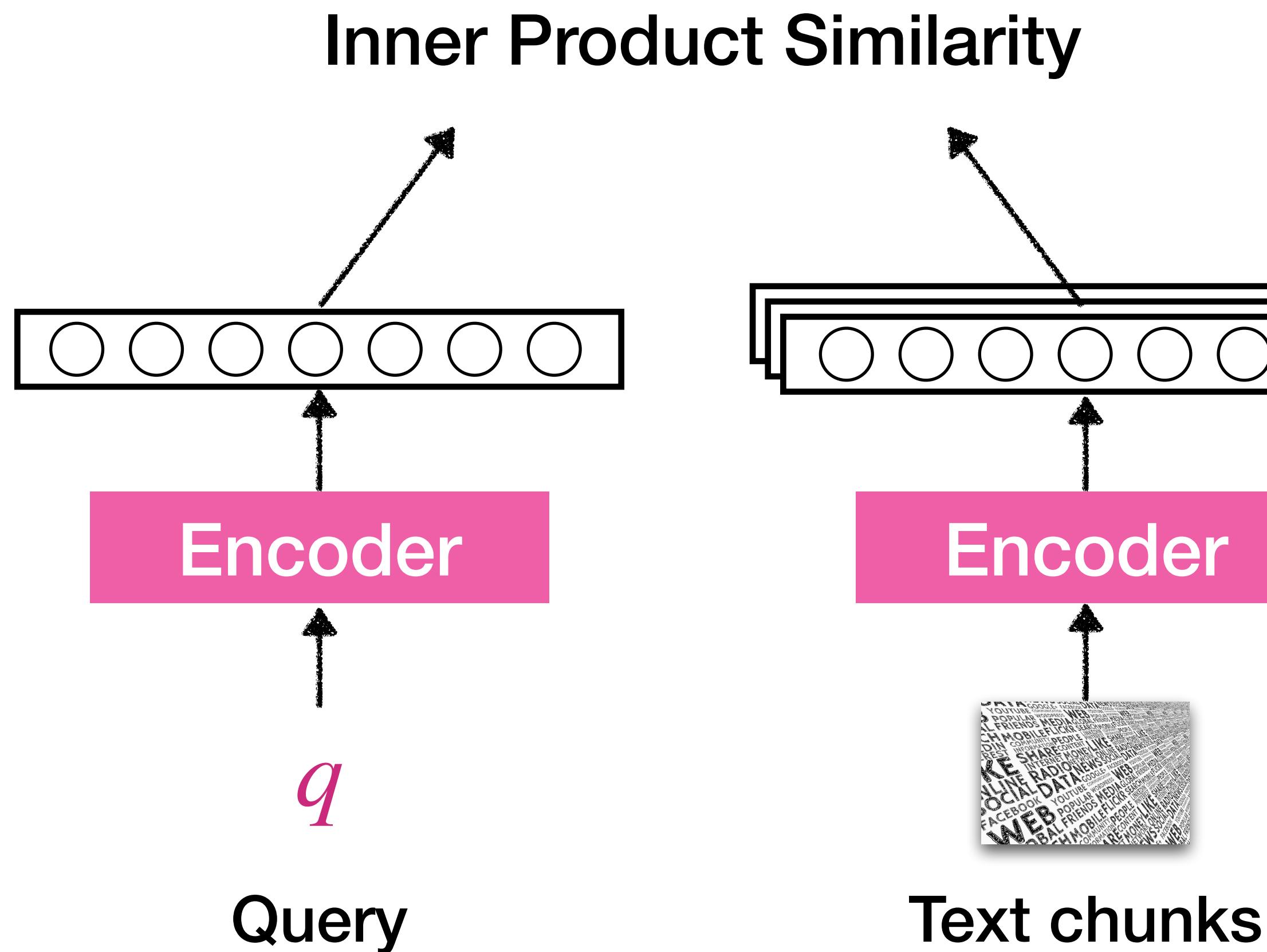


How to train dense retrieval models?

Training dense retrieval models: DPR

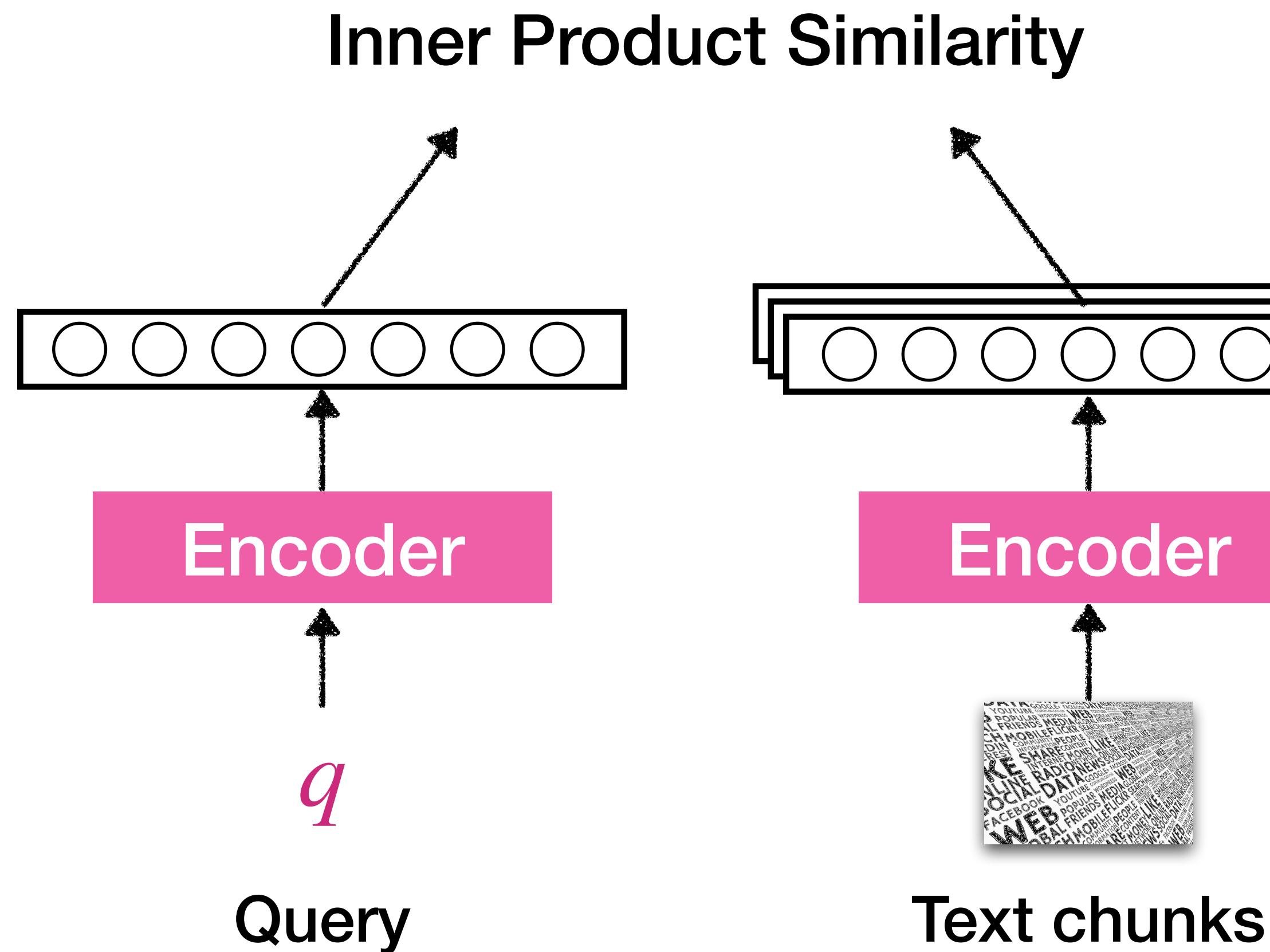


Training dense retrieval models: DPR



$$L(q, p^+, p_1^-, p_2^-, \dots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^n \exp(\text{sim}(q, p_j^-))}$$

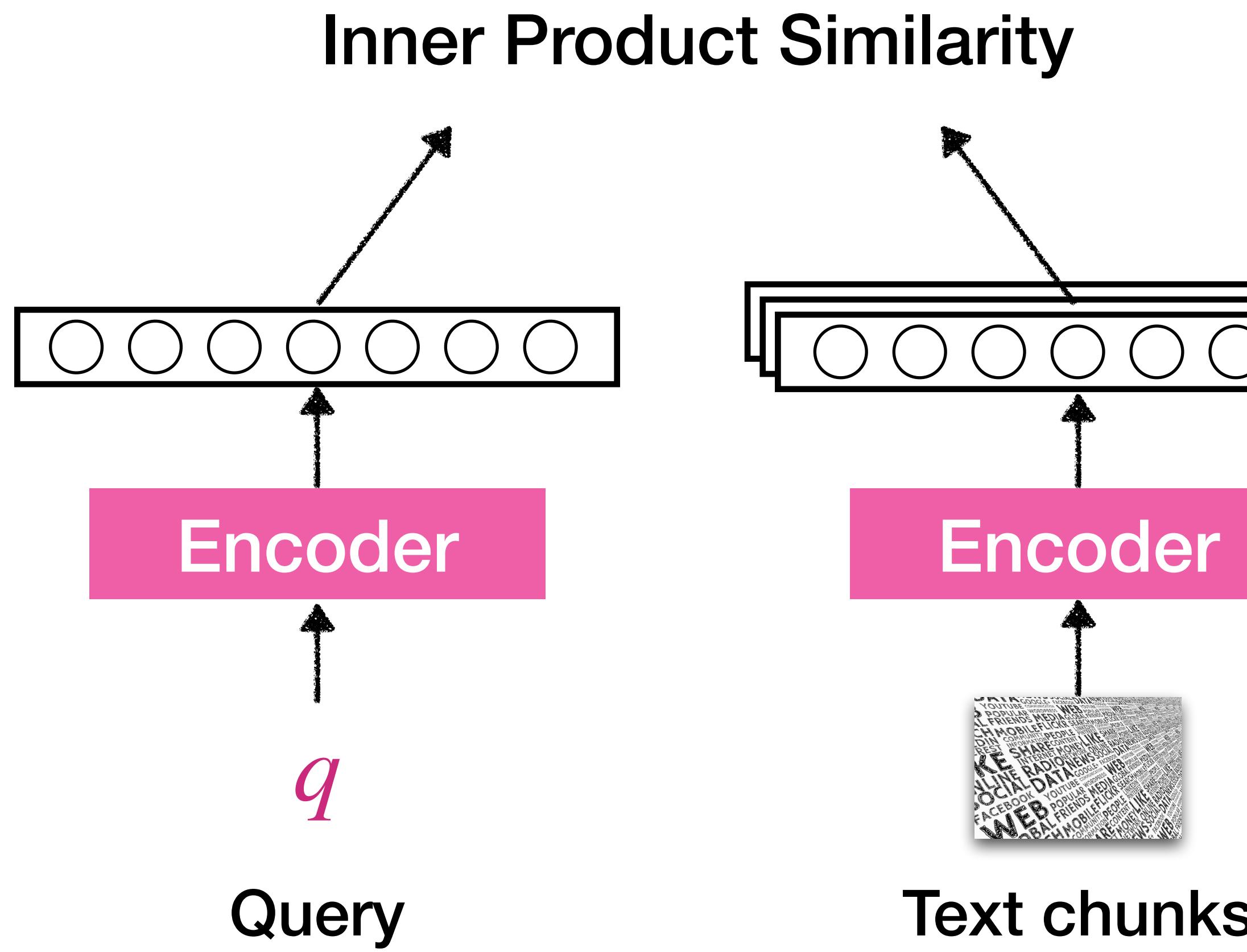
Training dense retrieval models: DPR



$$L(q, p^+, p_1^-, p_2^-, \dots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^n \exp(\text{sim}(q, p_j^-))}$$

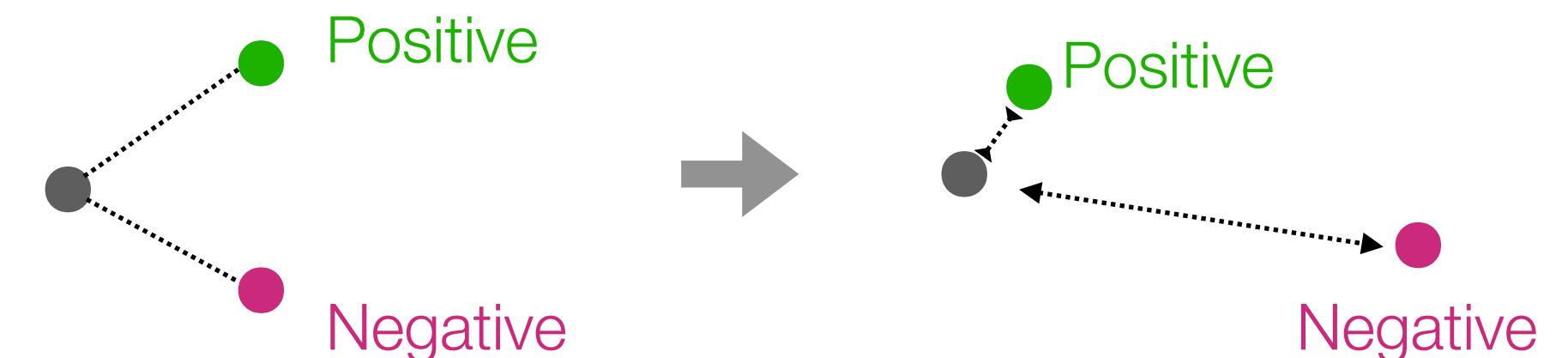
Contrastive learning

Training dense retrieval models: DPR

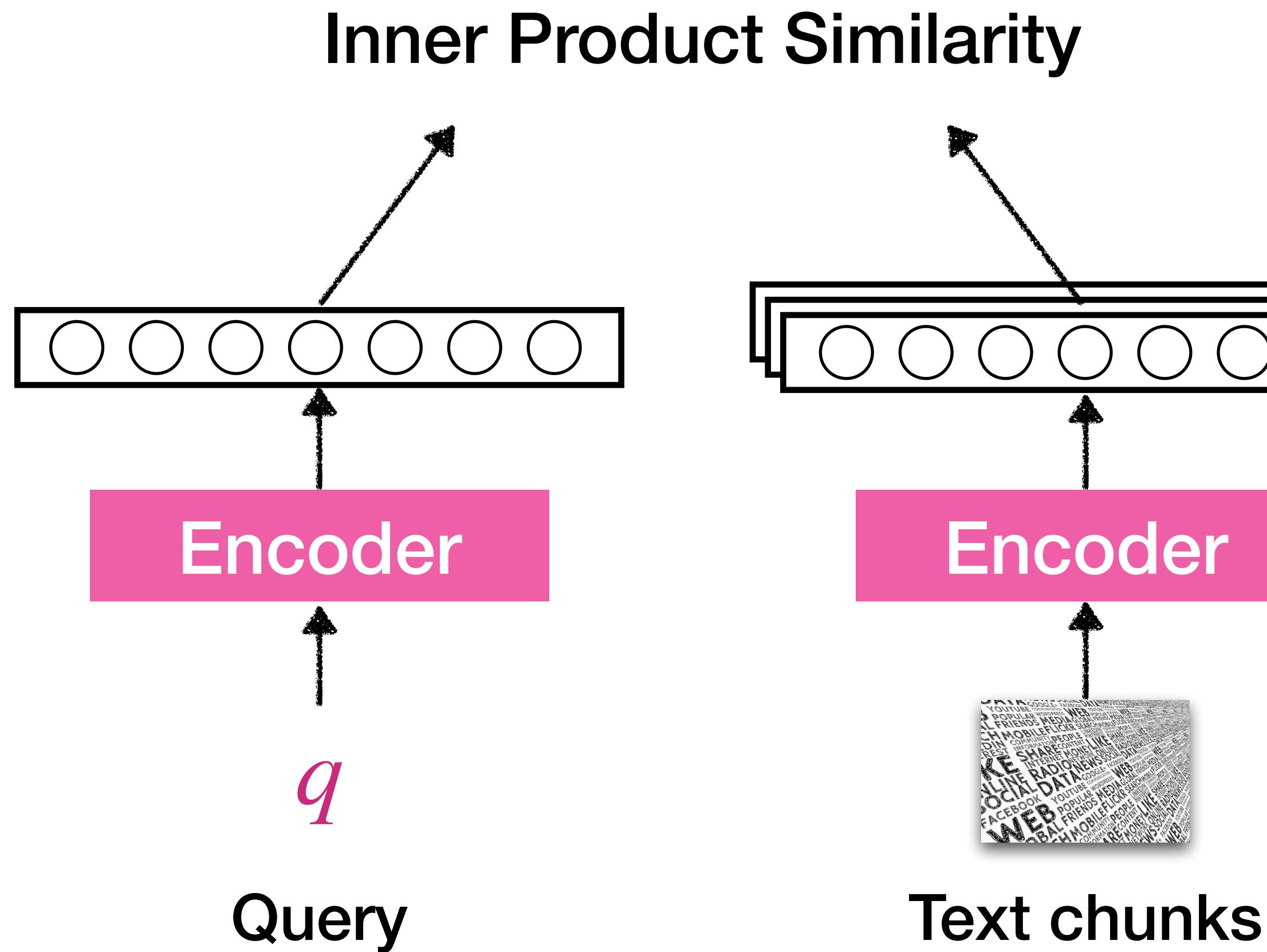


$$L(q, p^+, p_1^-, p_2^-, \dots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^n \exp(\text{sim}(q, p_j^-))}$$

Contrastive learning



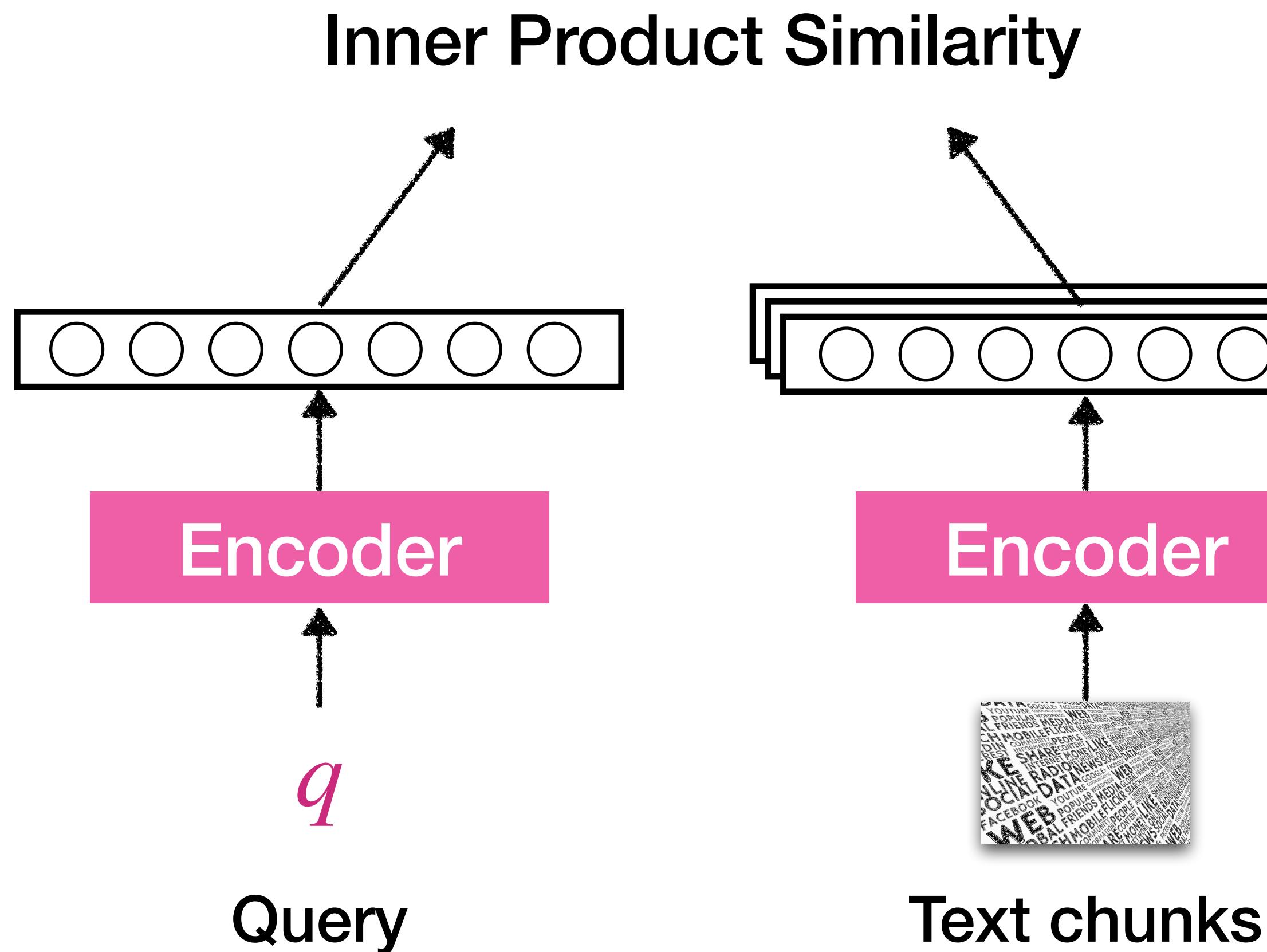
Training dense retrieval models: DPR



$$L(q, p^+, p_1^-, p_2^-, \dots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^n \exp(\text{sim}(q, p_j^-))}$$

Positive passage

Training dense retrieval models: DPR



Negative passages
Too expensive to consider all negatives!

$$L(q, p^+, p_1^-, p_2^-, \dots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^n \exp(\text{sim}(q, p_j^-))}$$

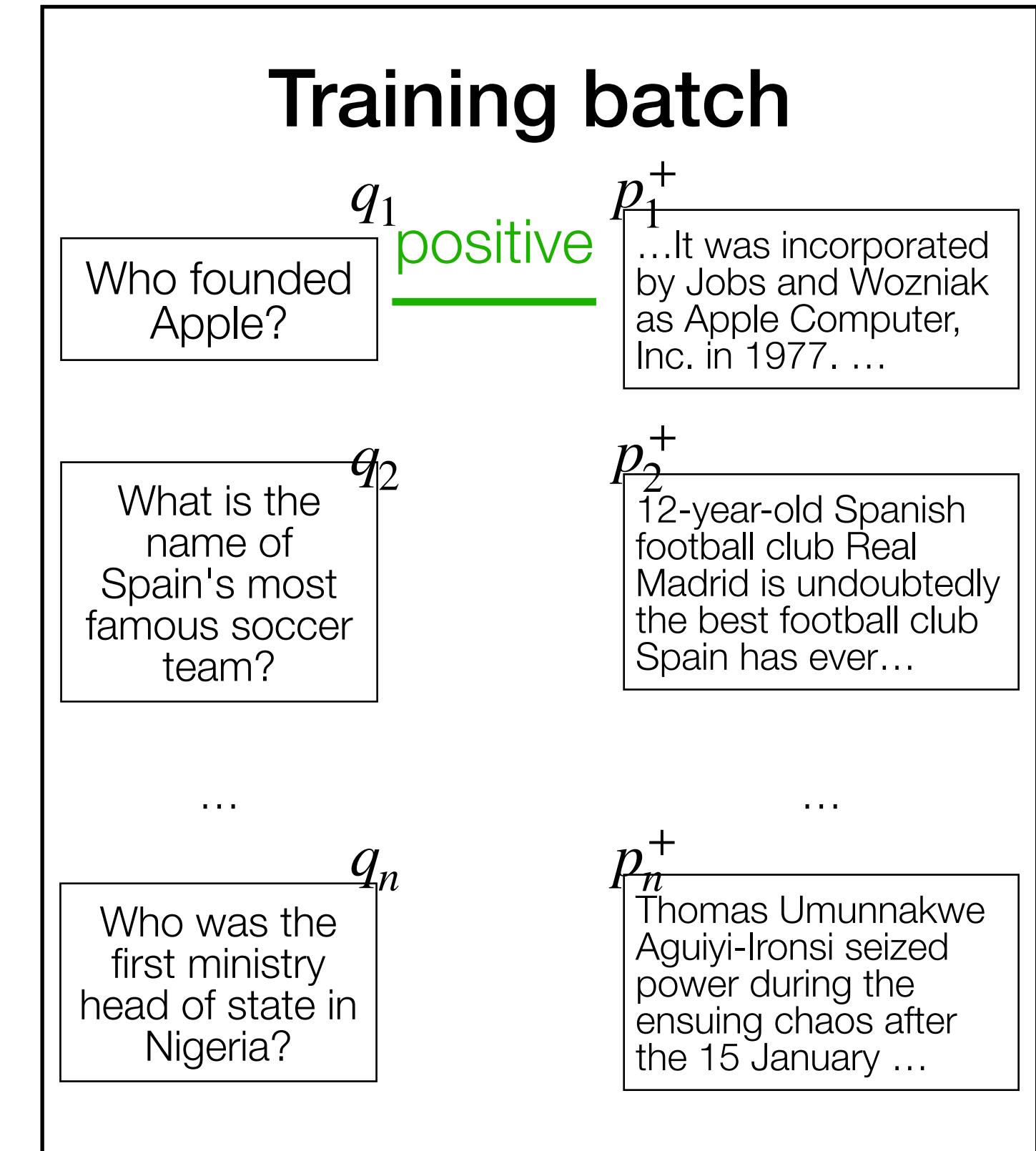
Training with “in-batch” negatives

$$L(q, p^+, p_1^-, p_2^-, \dots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^n \exp(\text{sim}(q, p_j^-))}$$

Training batch	
q_1	p_1^+ ...It was incorporated by Jobs and Wozniak as Apple Computer, Inc. in 1977. ...
q_2	p_2^+ 12-year-old Spanish football club Real Madrid is undoubtedly the best football club Spain has ever...
...	...
q_n	p_n^+ Thomas Umunnakwe Aguiyi-Ironsi seized power during the ensuing chaos after the 15 January ...

Training with “in-batch” negatives

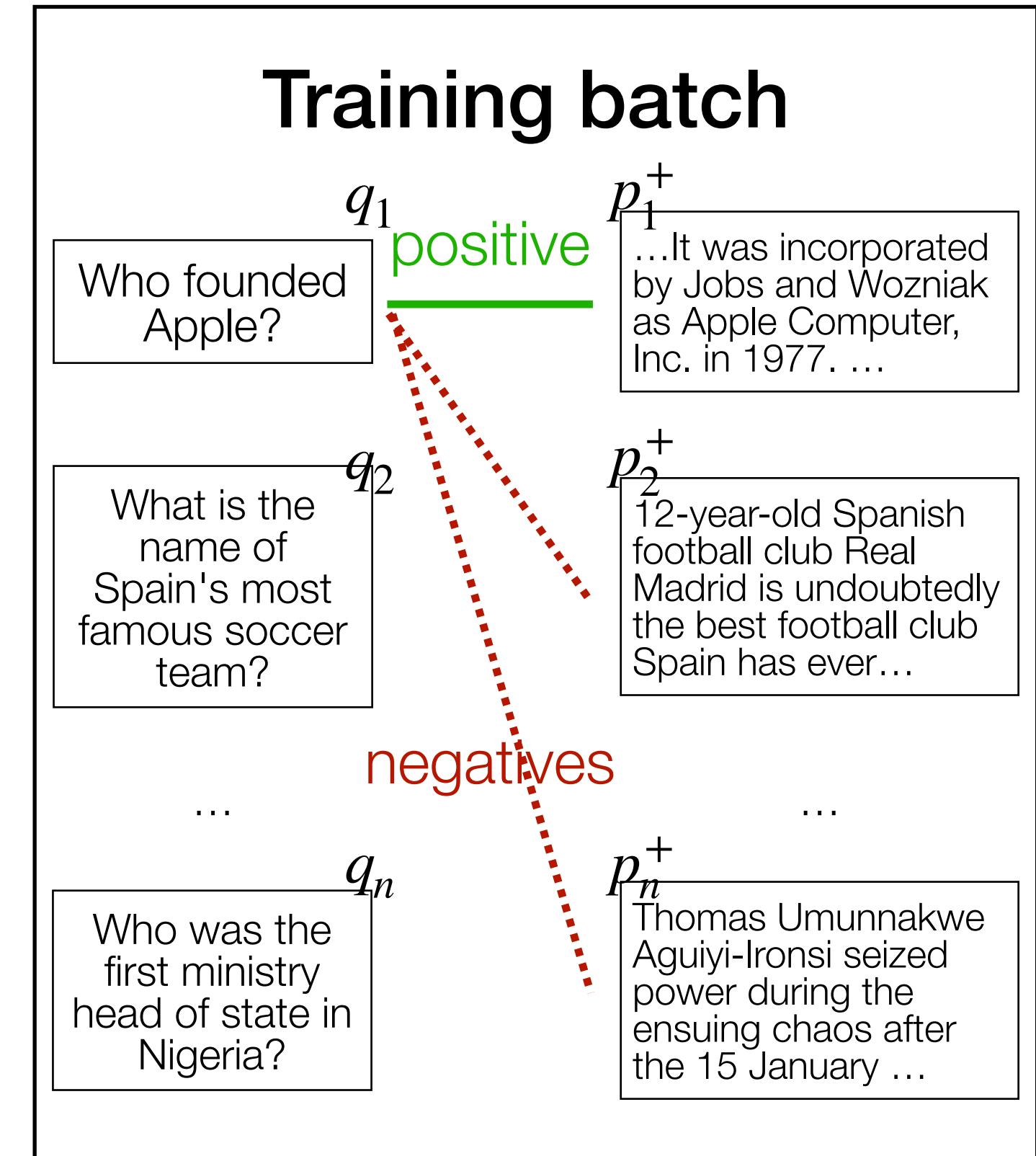
$$L(q, p^+, p_1^-, p_2^-, \dots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^n \exp(\text{sim}(q, p_j^-))}$$



Training with “in-batch” negatives

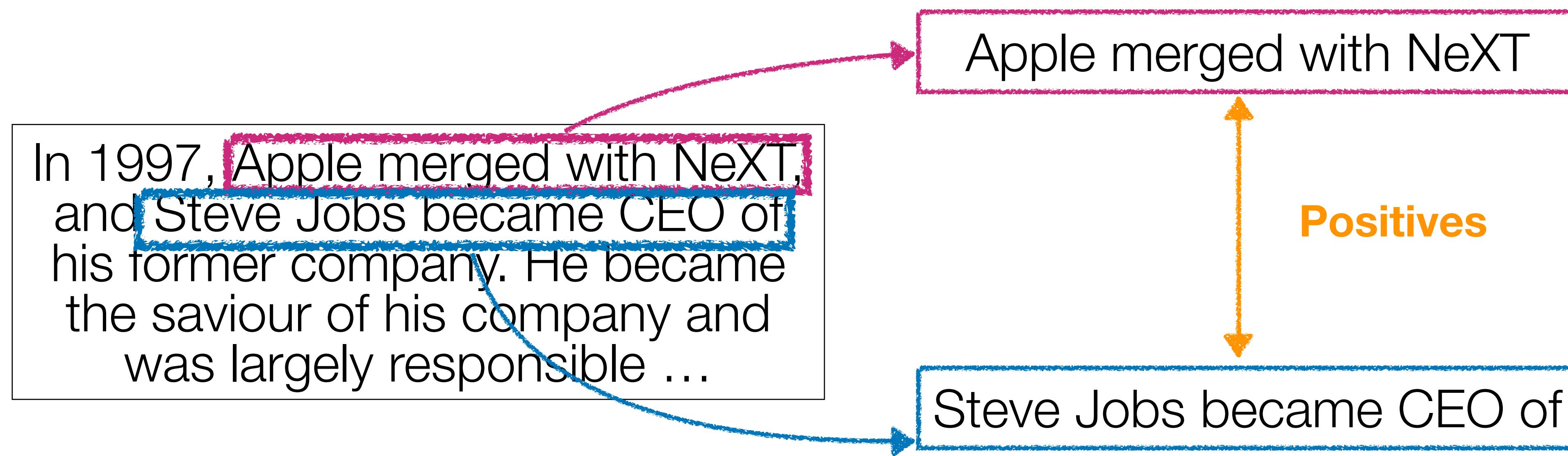
$$L(q, p^+, p_1^-, p_2^-, \dots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^n \exp(\text{sim}(q, p_j^-))}$$

Back-propagation to all in-batch negatives!



Contriever (Izacard et al. 2022)

Independent Cropping



Unsupervised dense retrieval model!

Retrieval-in-context in LM (Ram et al. 2023)

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

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



Retrieval Model



FIFA World Cup 2026 will expand to 48 teams. World Cup 2022 was the last with 32 teams, before the increase to



LM



48 in the 2026 tournament.

Retrieval-in-context in LM (Ram et al. 2023)

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

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



Retrieval Model

BM25, DPR, Contriever, ...



FIFA World Cup 2026 will expand to 48 teams. World Cup 2022 was the last with 32 teams, before the increase to



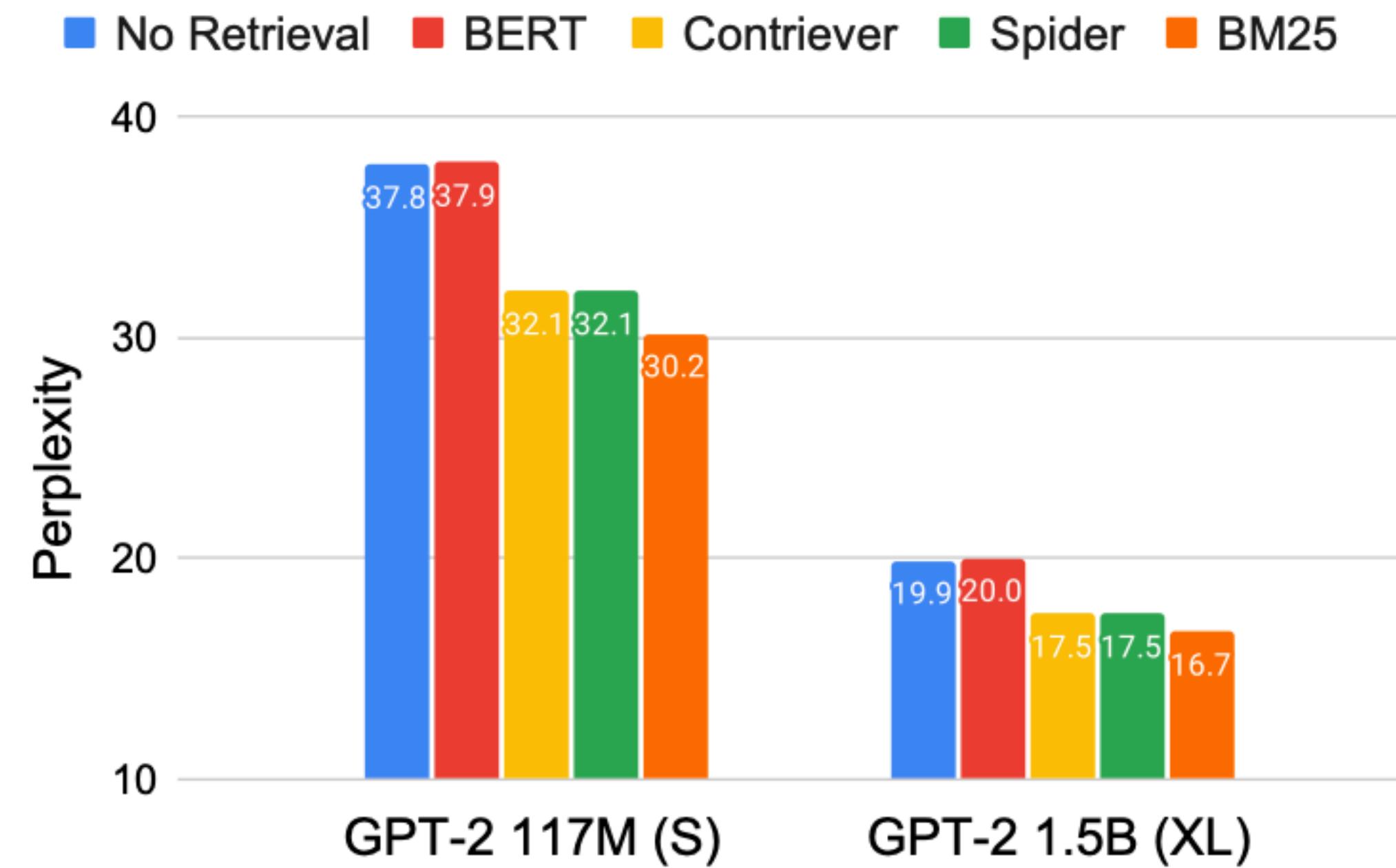
LM

GPT, OPT, LLaMA, ...



48 in the 2026 tournament.

Retrieval-in-context in LM



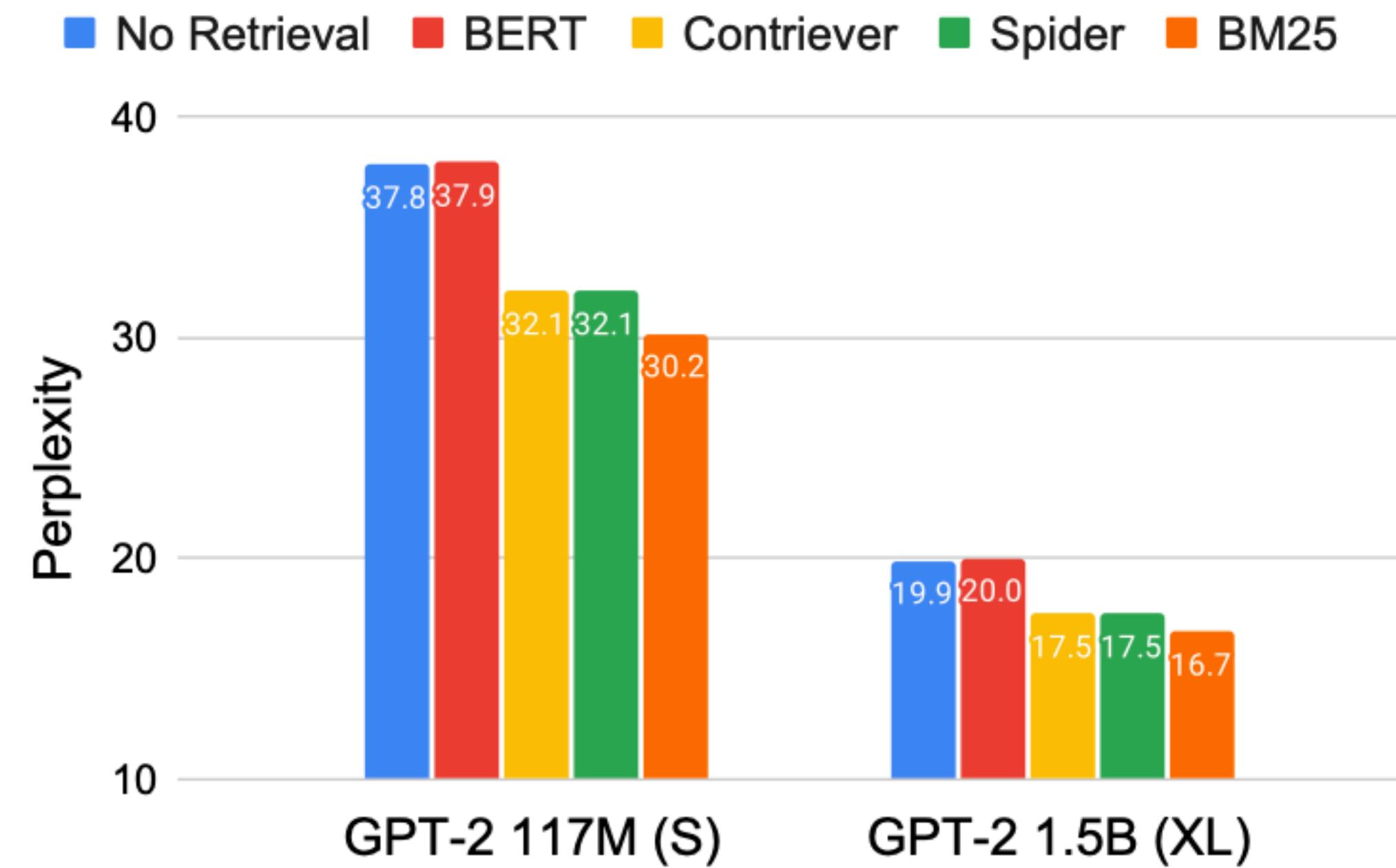
Better retrieval model



Better **retrieval-based LMs**

Better base LMs

Retrieval-in-context in LM



Better retrieval model



Better **retrieval-based LMs**

Better base LMs

Each component can be improved separately

kNN-LM (Khandelwal et al. 2020)

Inference

$$P_{k\text{NN}}(y \mid x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \exp(-d(\text{Enc}(k), \text{Enc}(x)))$$

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

kNN-LM (Khandelwal et al. 2020)

Inference

Re-use the LM encoder. No training needed!

$$P_{k\text{NN}}(y \mid x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \exp(-d(\text{Enc}(k), \text{Enc}(x)))$$

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

kNN-LM (Khandelwal et al. 2020)

Inference

Re-use the LM encoder. No training needed!

$$P_{k\text{NN}}(y|x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \exp(-d(\text{Enc}(k), \text{Enc}(x)))$$

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

Training

Minimize $-\log P_{\text{LM}}(y|x)$

Independent training

-  Work with off-the-shelf models (no extra training required)
-  Each part can be improved independently

Independent training

-  Work with off-the-shelf models (no extra training required)
-  Each part can be improved independently
-  LMs are not trained to leverage retrieval
-  Retrieval models are not optimized for LM tasks/domains

Training methods for retrieval-based LMs

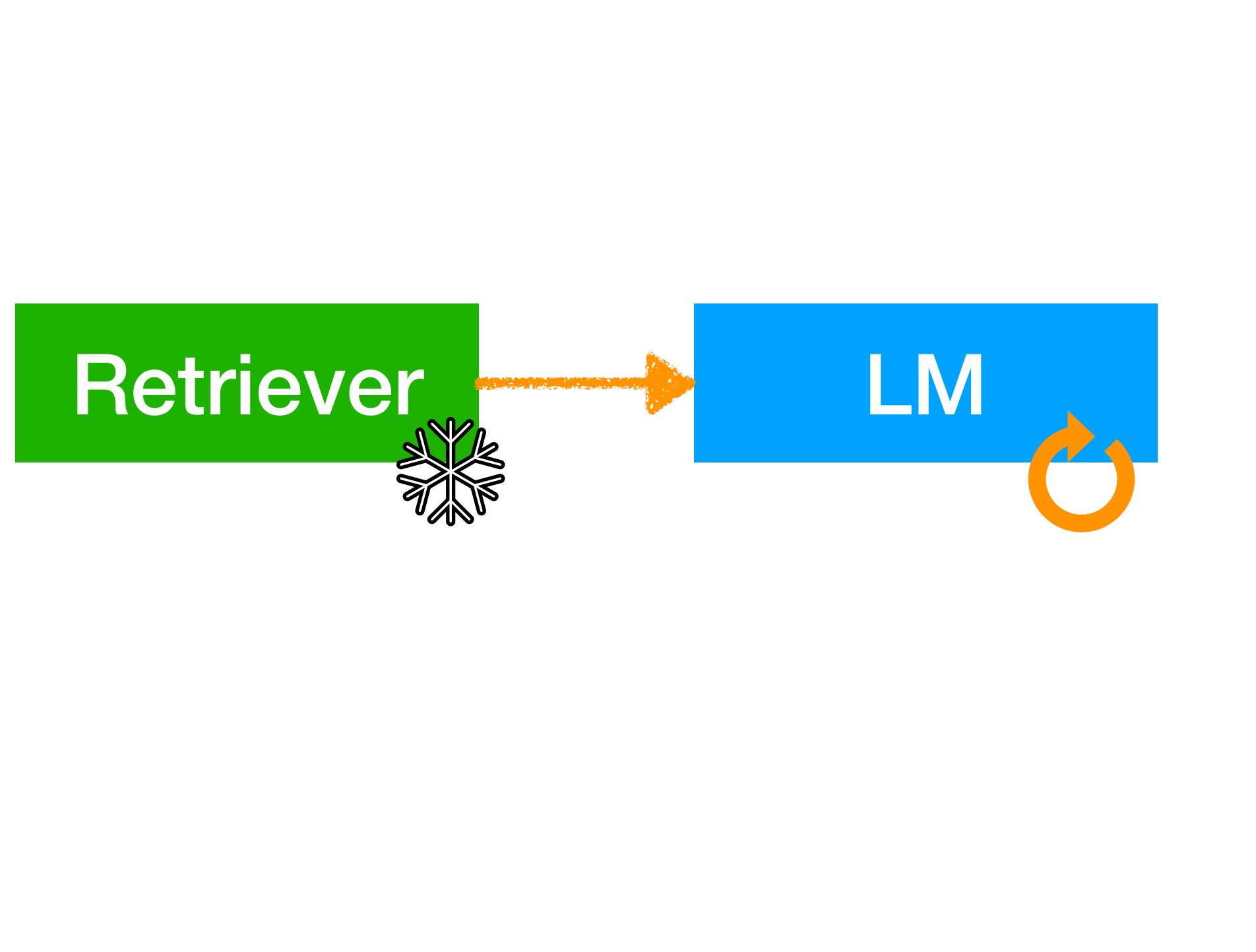
- Independent training
- **Sequential training**
- Joint training w/ asynchronous index update
- Joint training w/ in-batch approximation

Sequential training

- One component is first trained independently and then fixed
- The other component is trained with an objective that depends on the first one

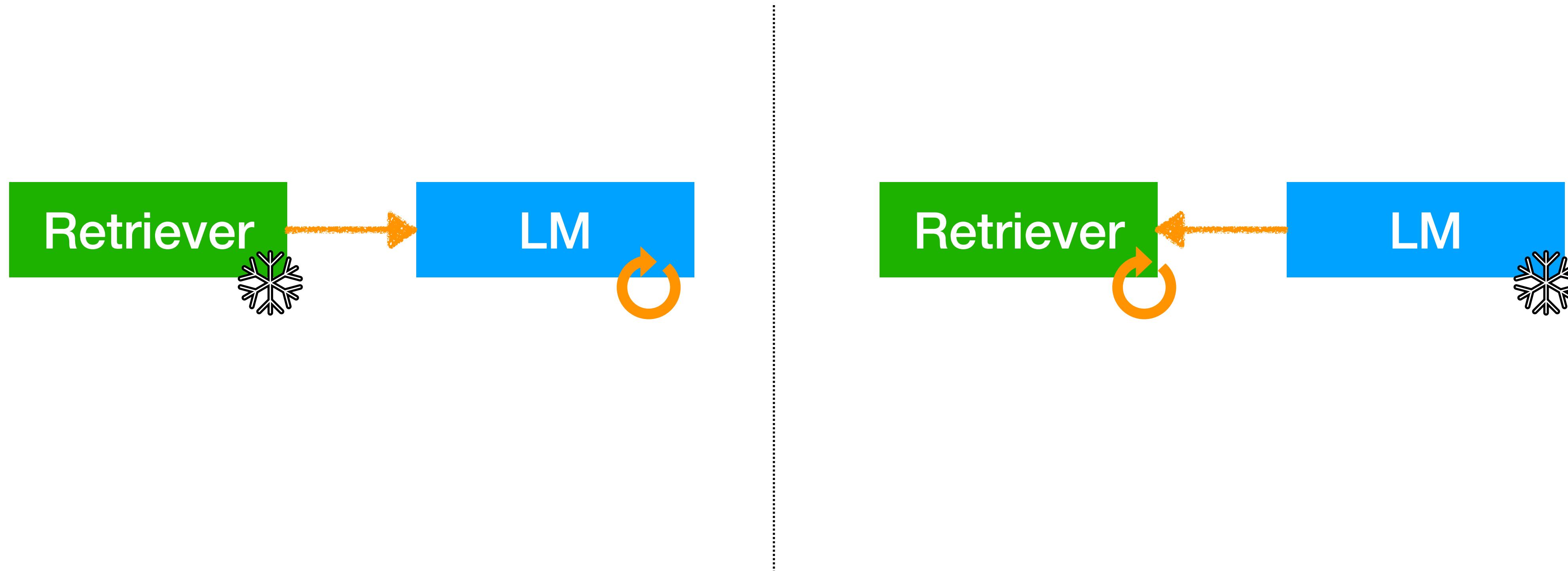
Sequential training

- One component is first trained independently and then fixed
- The other component is trained with an objective that depends on the first one



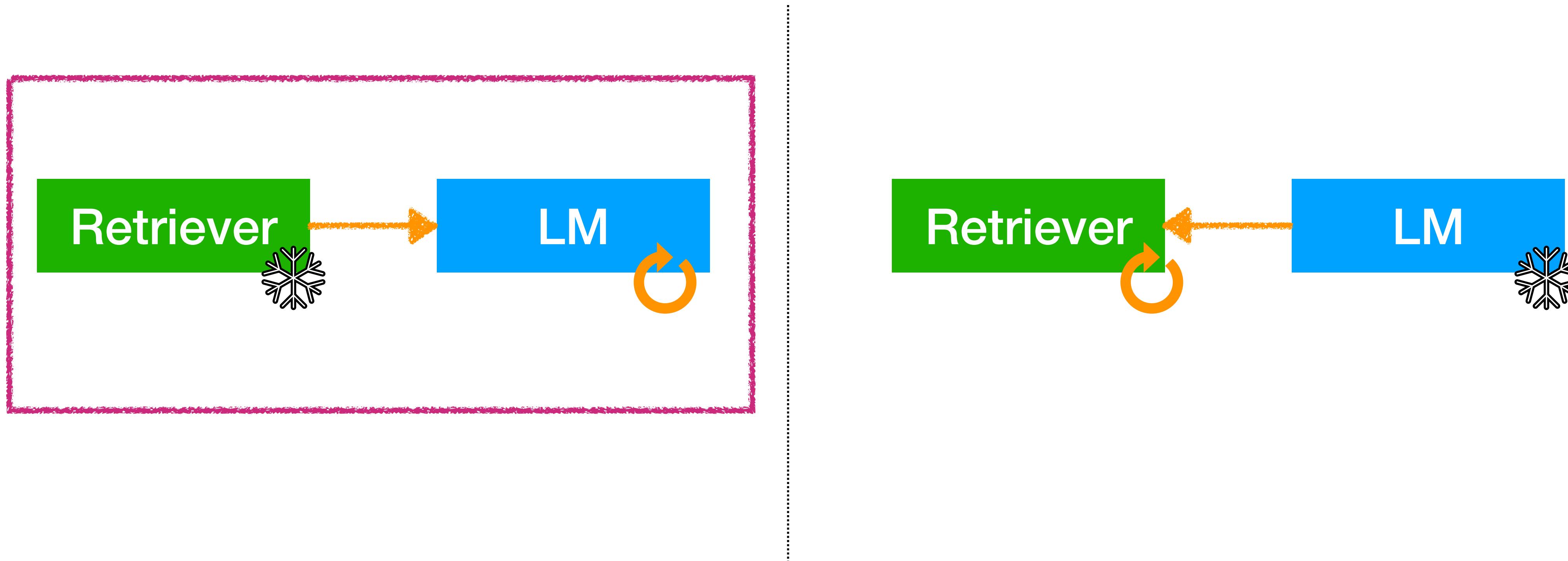
Sequential training

- One component is first trained independently and then fixed
- The other component is trained with an objective that depends on the first one



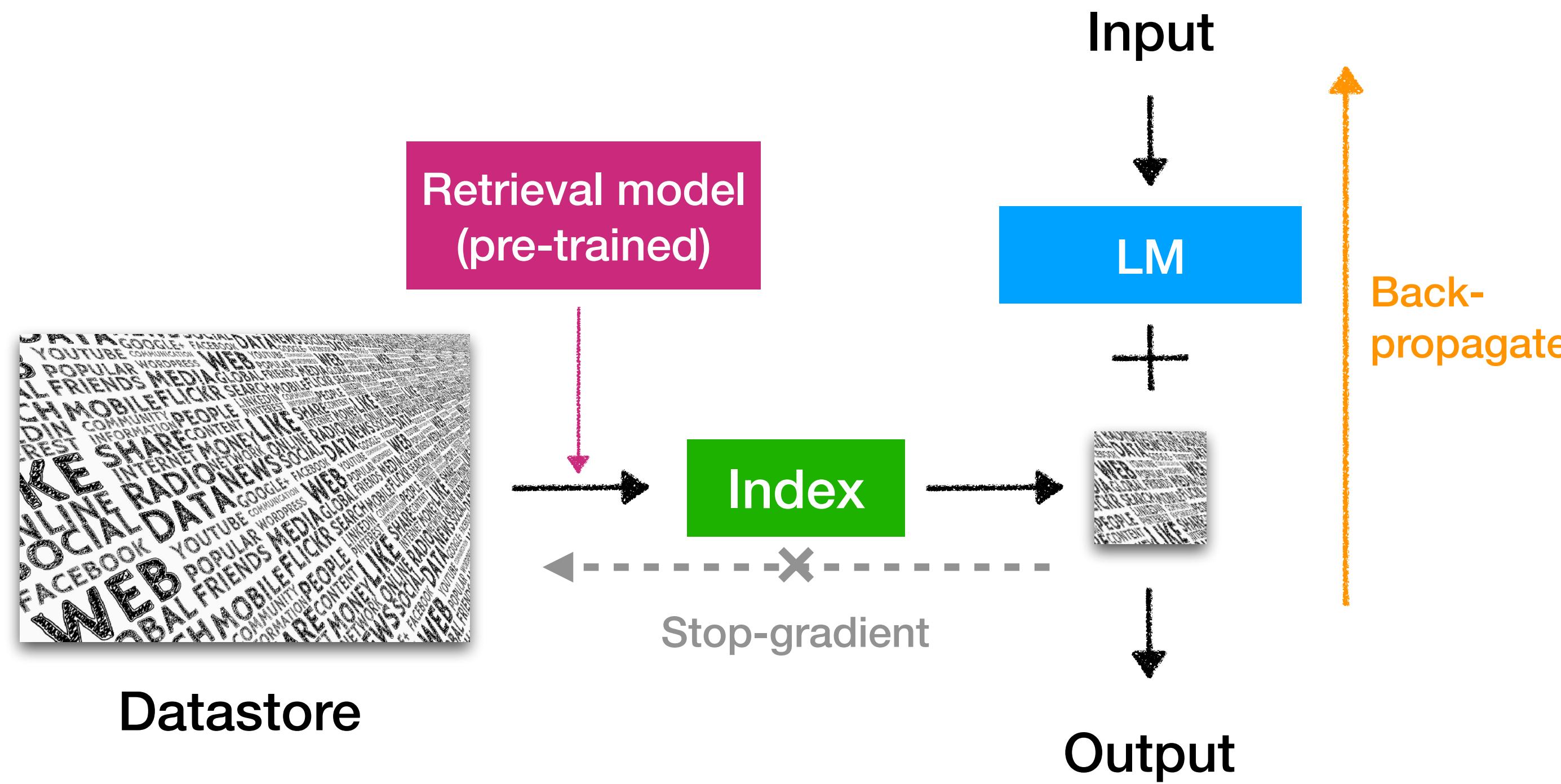
Sequential training

- One component is first trained independently and then fixed
- The other component is trained with an objective that depends on the first one



Sequential training

- Retrieval models are first trained independently and then fixed
- Language models are trained with an objective that depends on the retrieval



RETRO (Borgeaud et al. 2021)

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

x₁

x₂

x₃

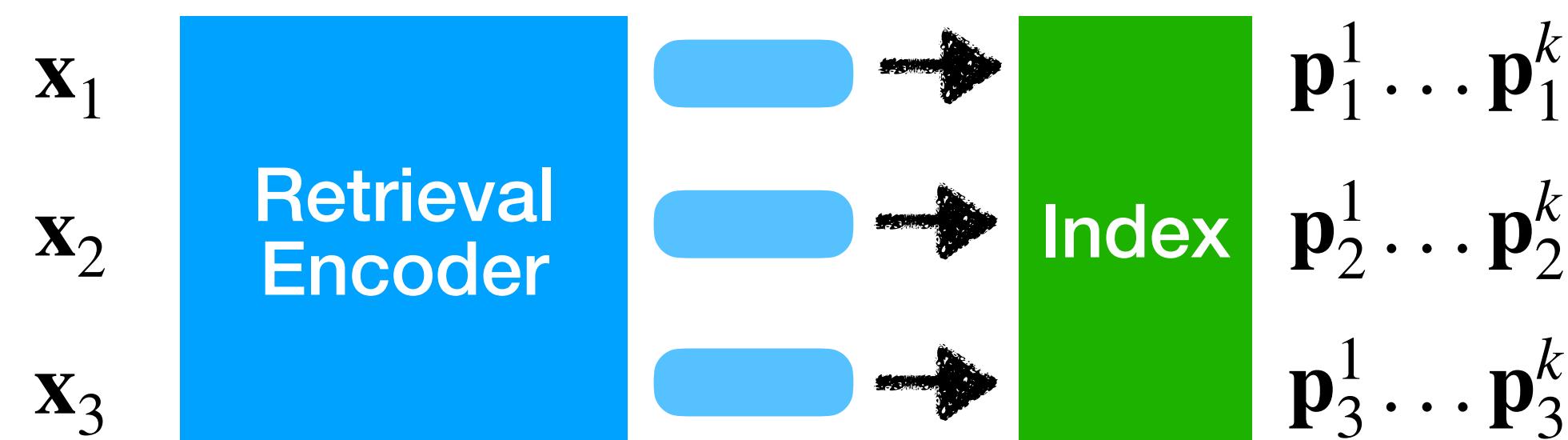
RETRO (Borgeaud et al. 2021)

\mathbf{x} = World Cup 2022 was ~~the last with 32 teams~~, before the increase to

\mathbf{x}_1

\mathbf{x}_2

\mathbf{x}_3



RETRO (Borgeaud et al. 2021)

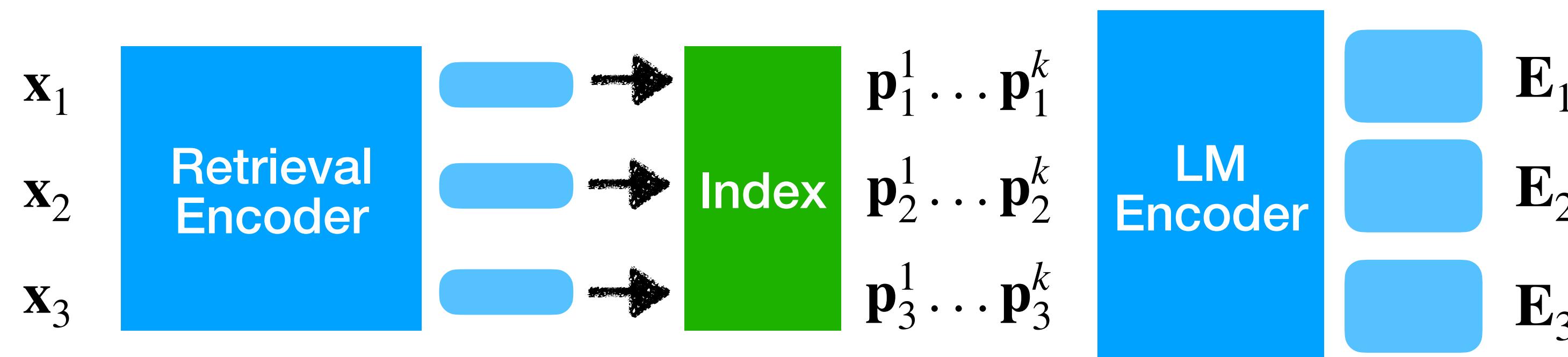
\mathbf{x} = World Cup 2022 was~~the~~ last with 32 teams,~~before~~ before the increase to

\mathbf{x}_1

\mathbf{x}_2

\mathbf{x}_3

(k chunks of text per split)



RETRO (Borgeaud et al. 2021)

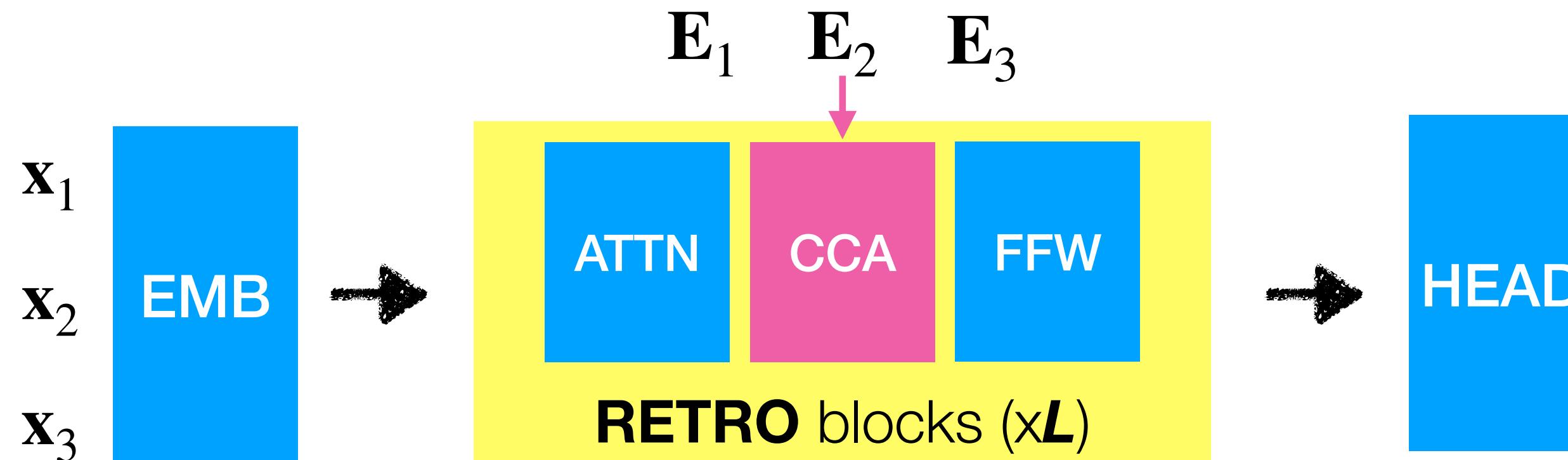
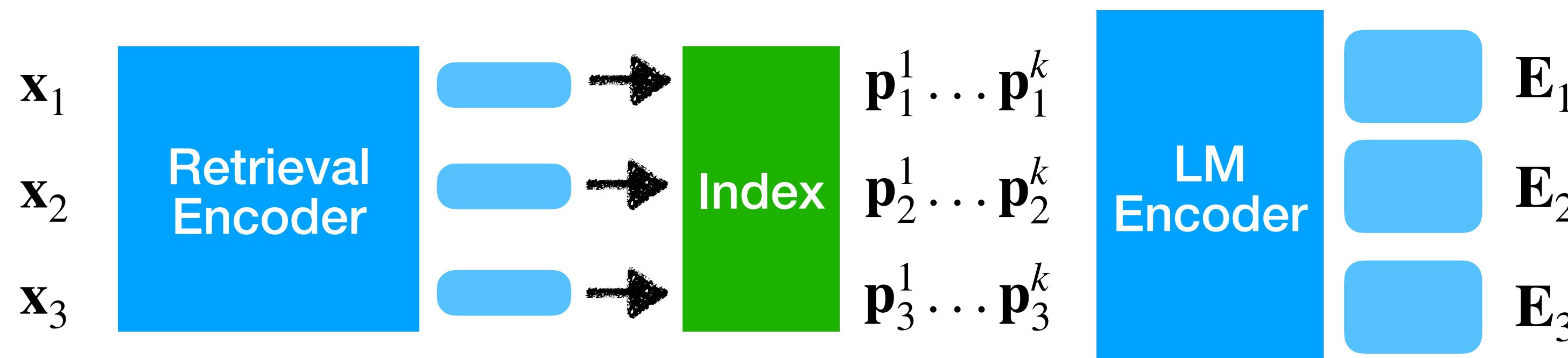
~~x = World Cup 2022 was the last with 32 teams, before the increase to~~

\mathbf{x}_1

\mathbf{x}_2

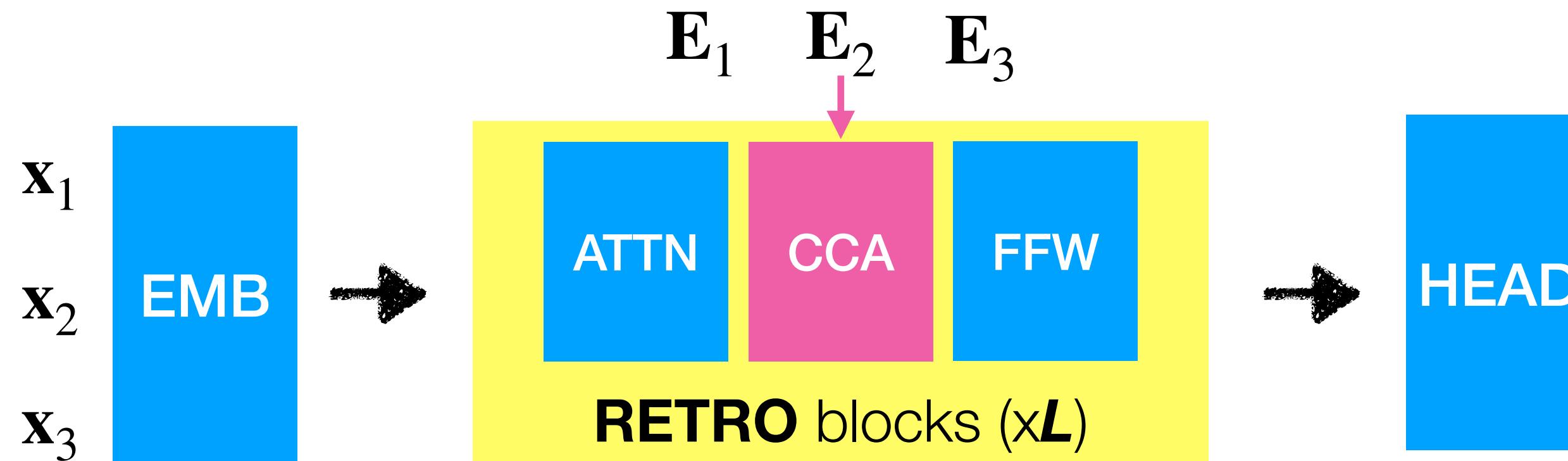
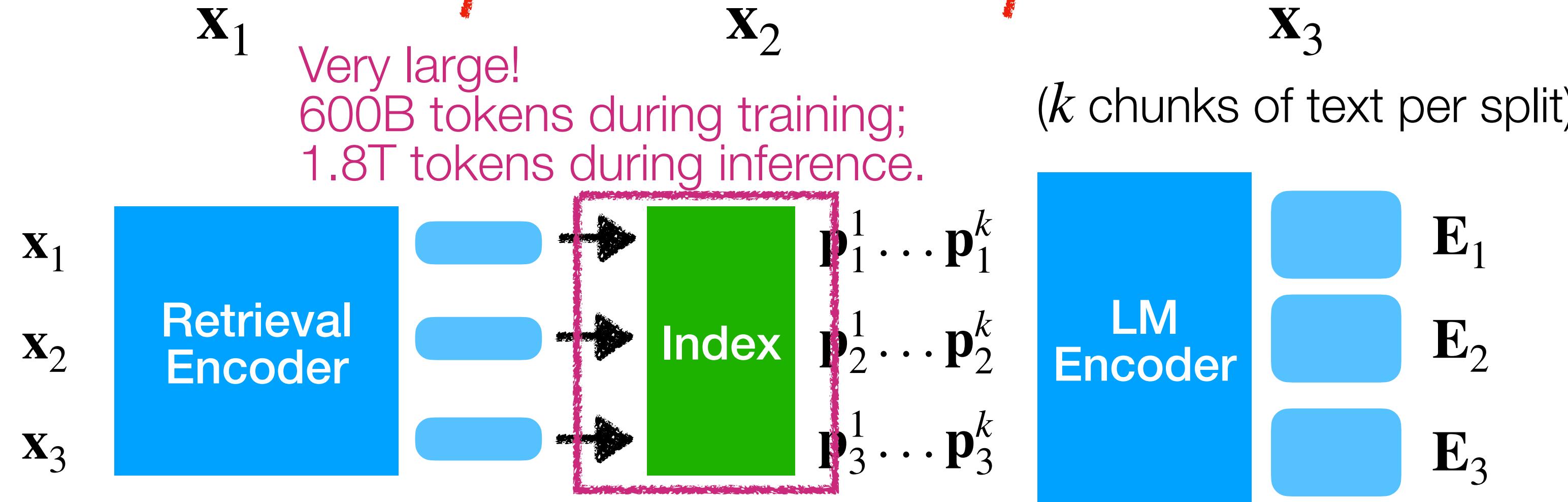
\mathbf{x}_3

(k chunks of text per split)

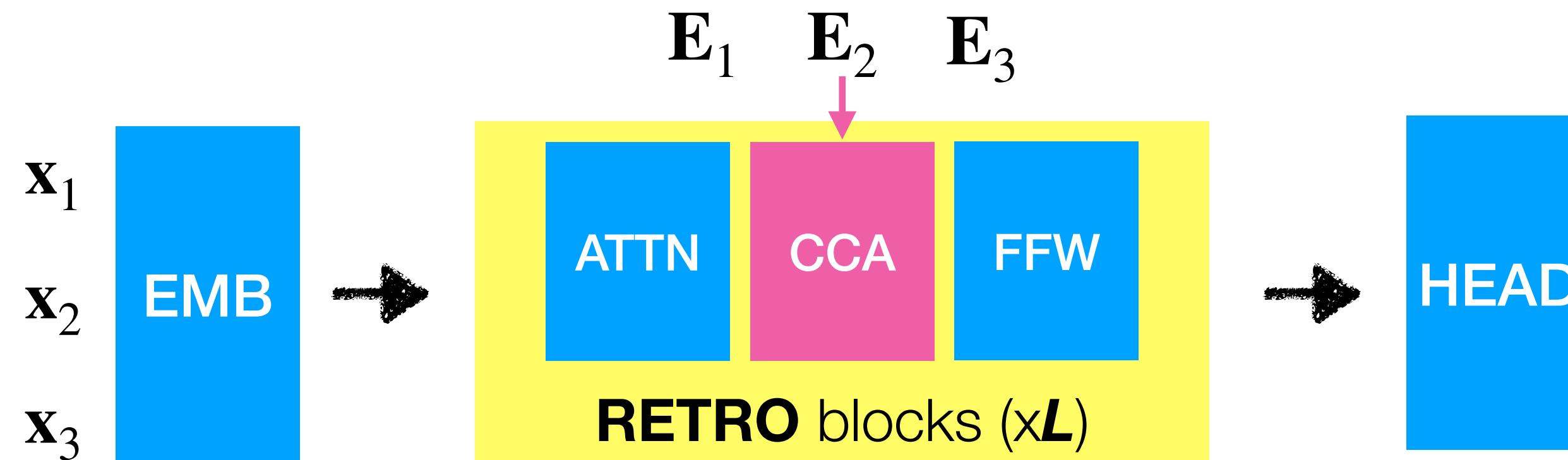
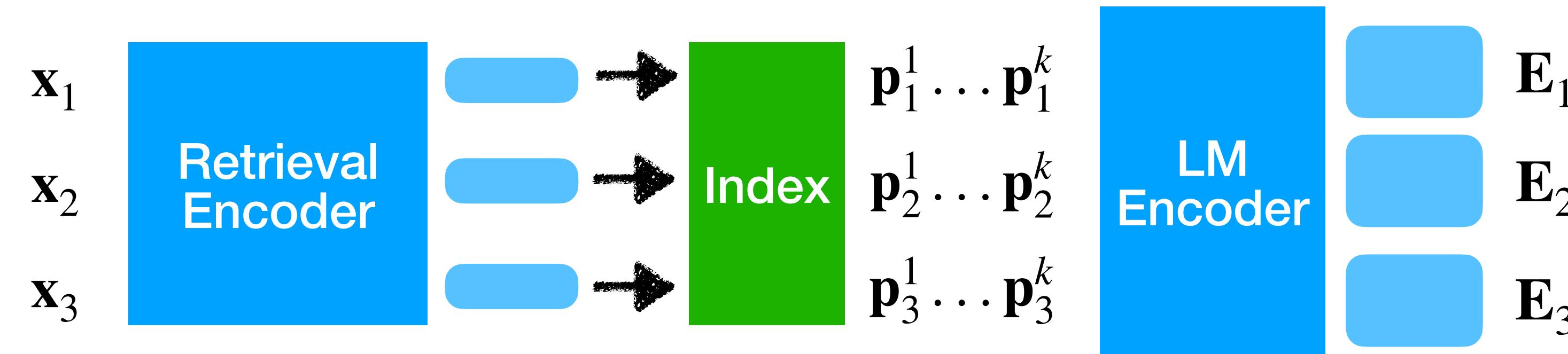


RETRO (Borgeaud et al. 2021)

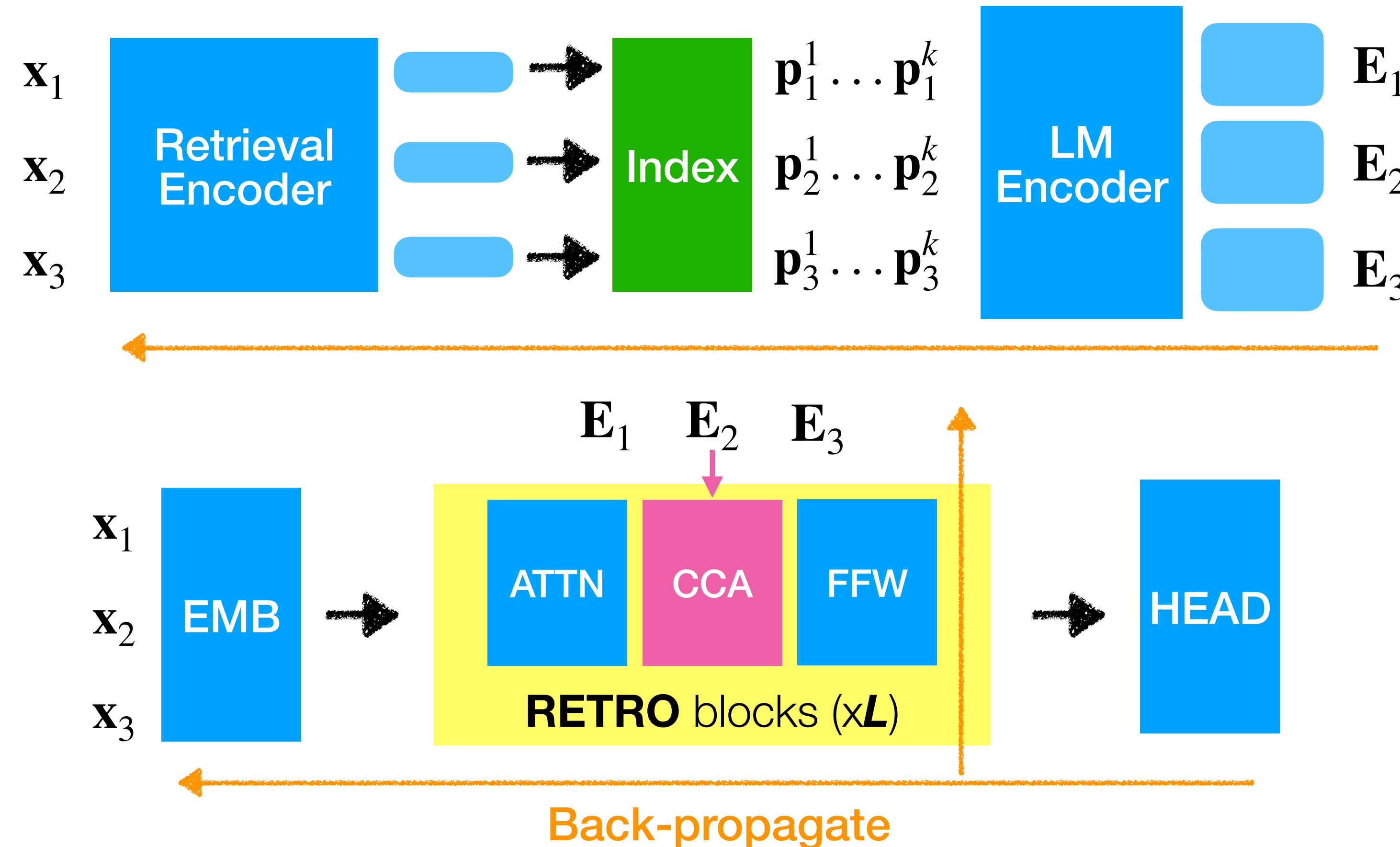
\mathbf{x} = World Cup 2022 was~~the~~ the last with 32 teams, before the increase to



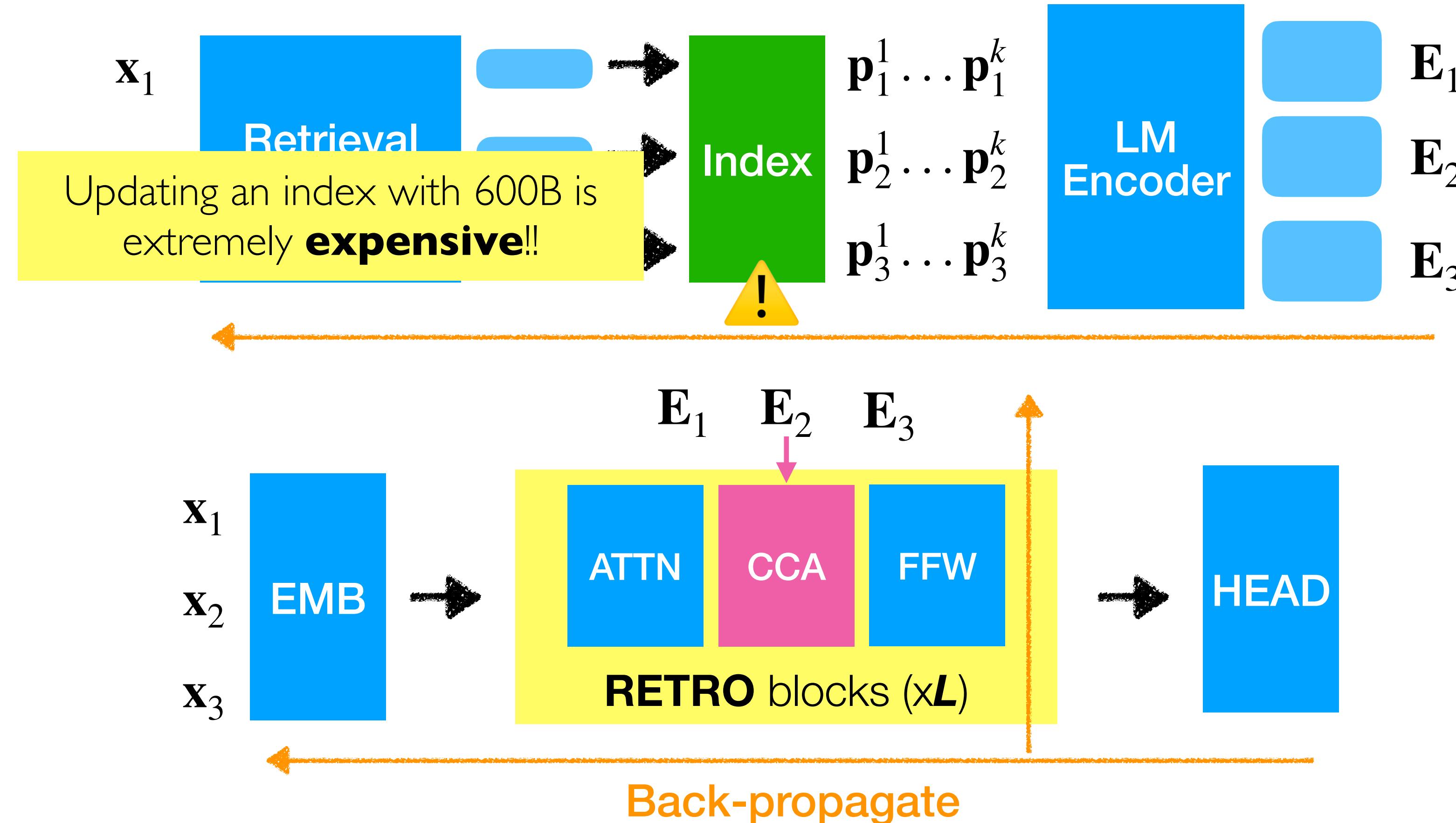
RETRO:Training



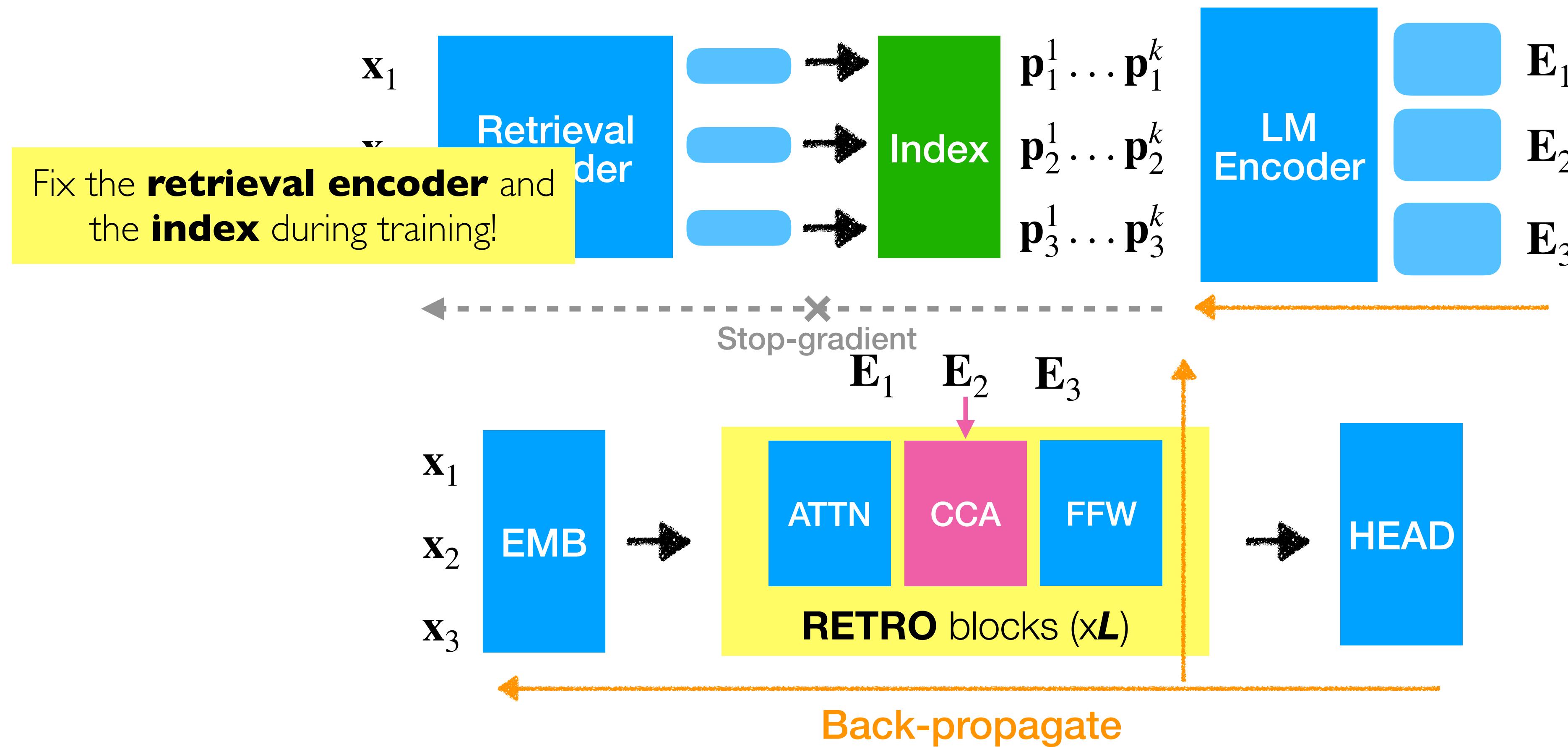
RETRO:Training



RETRO: Training

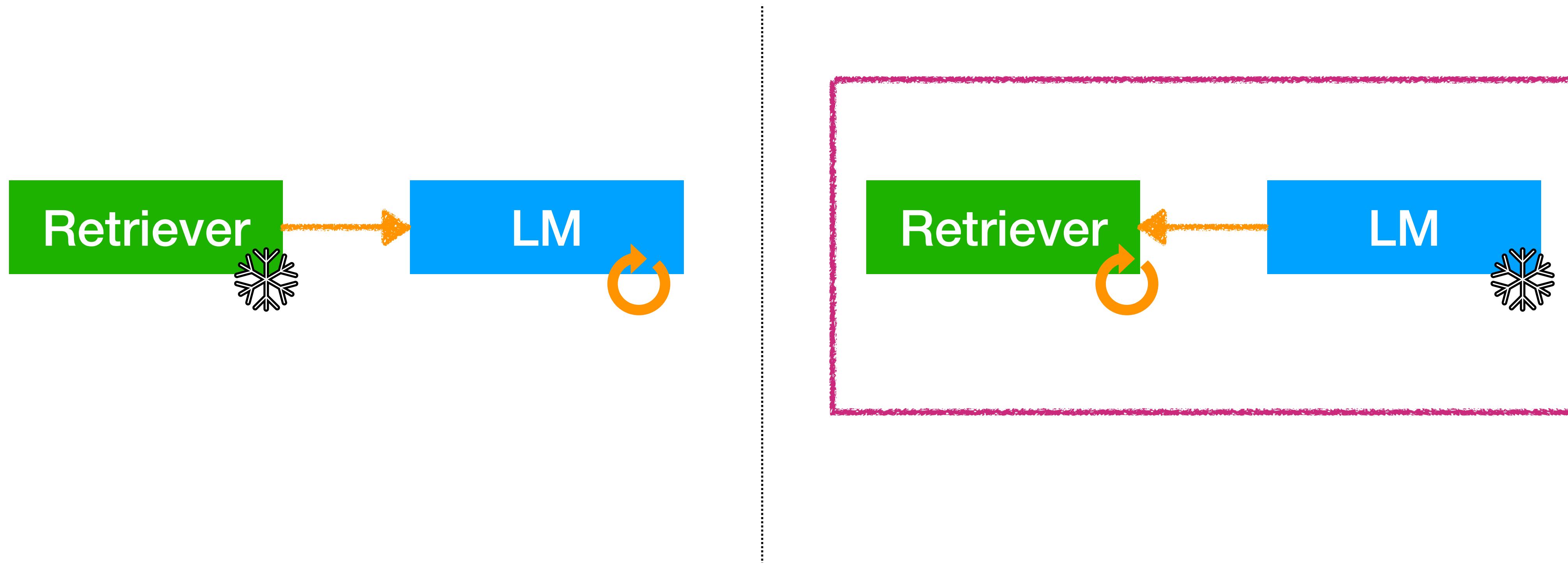


RETRO: Training



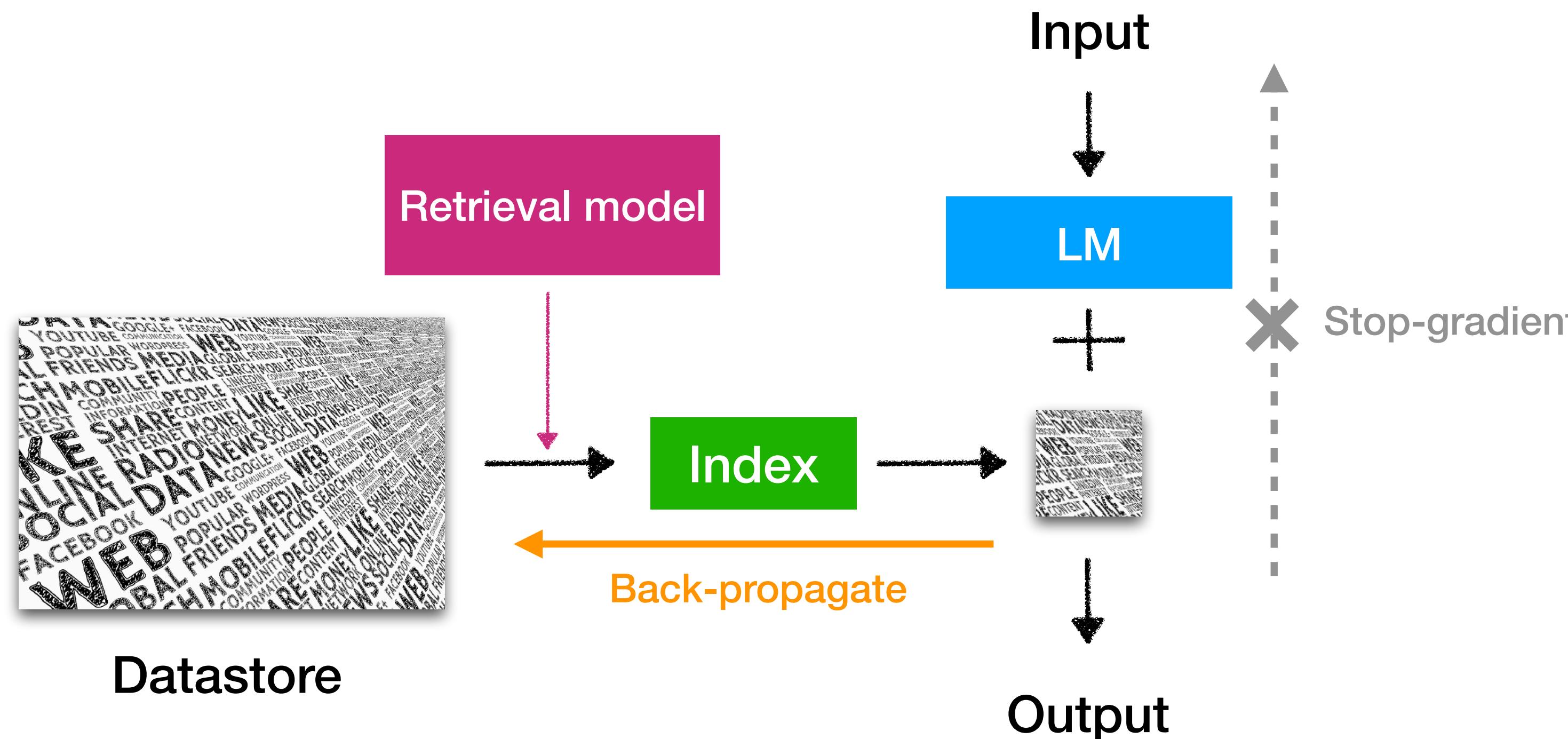
Sequential training

- One component is first trained independently and then fixed
- The other component is trained with an objective that depends on the first one

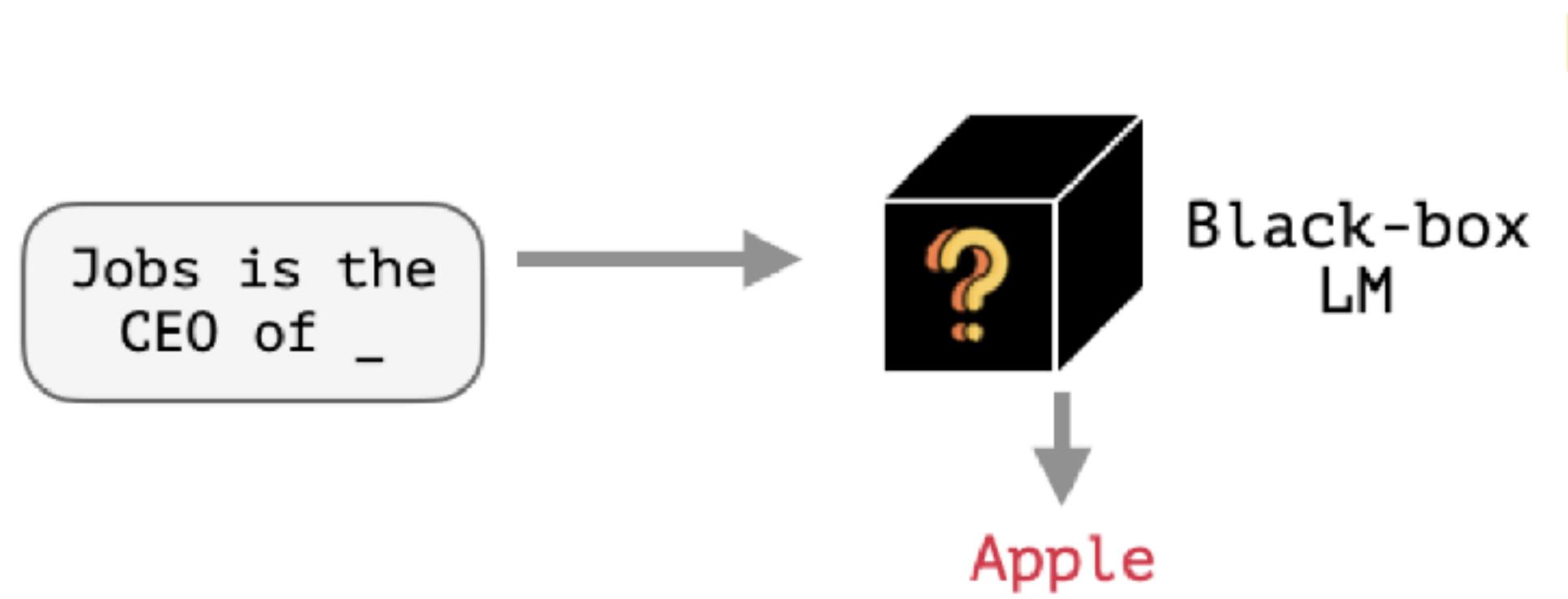


Sequential training

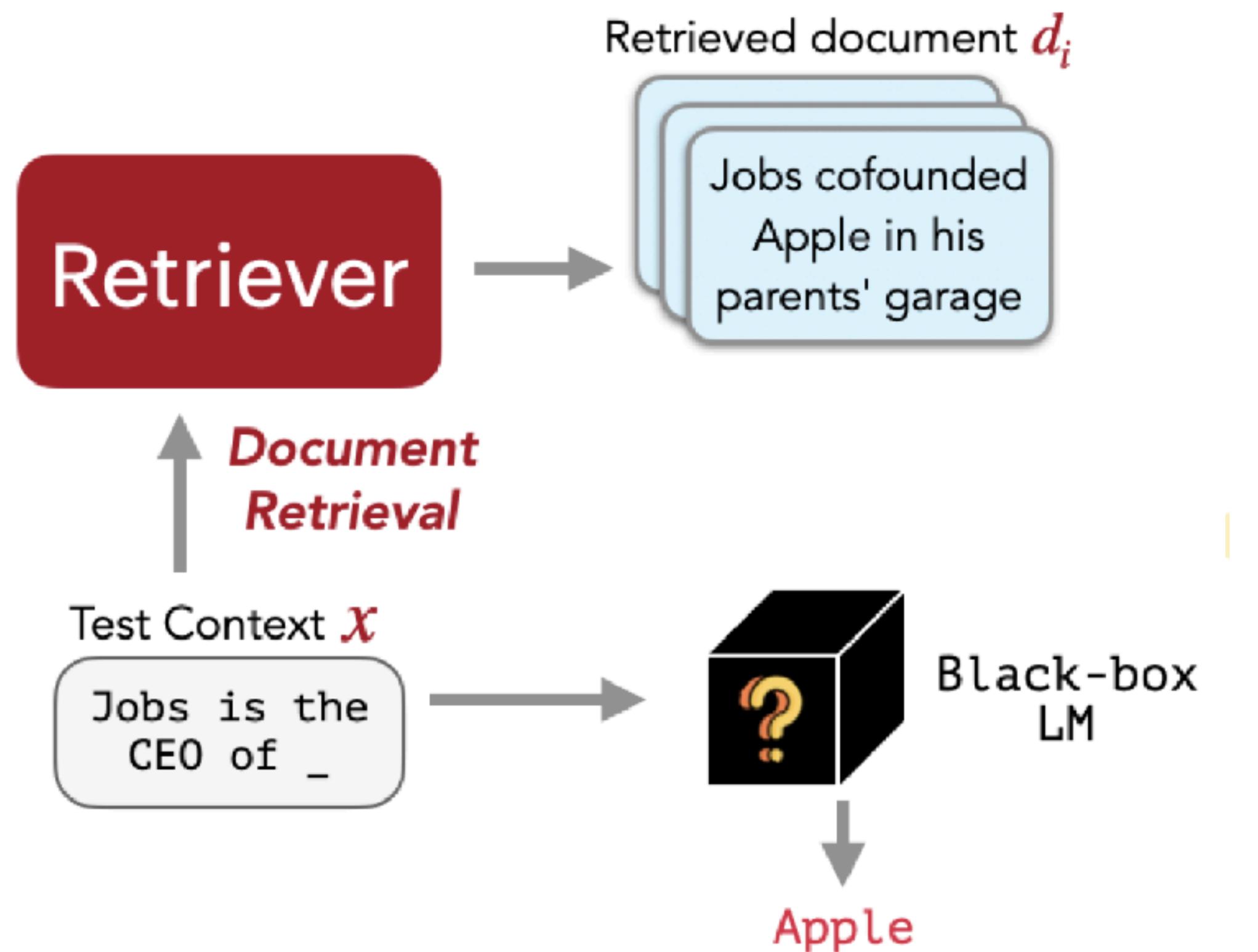
- Language models are first trained independently and then fixed
- Retrieval models are trained/fine-tuned with supervisions from LMs



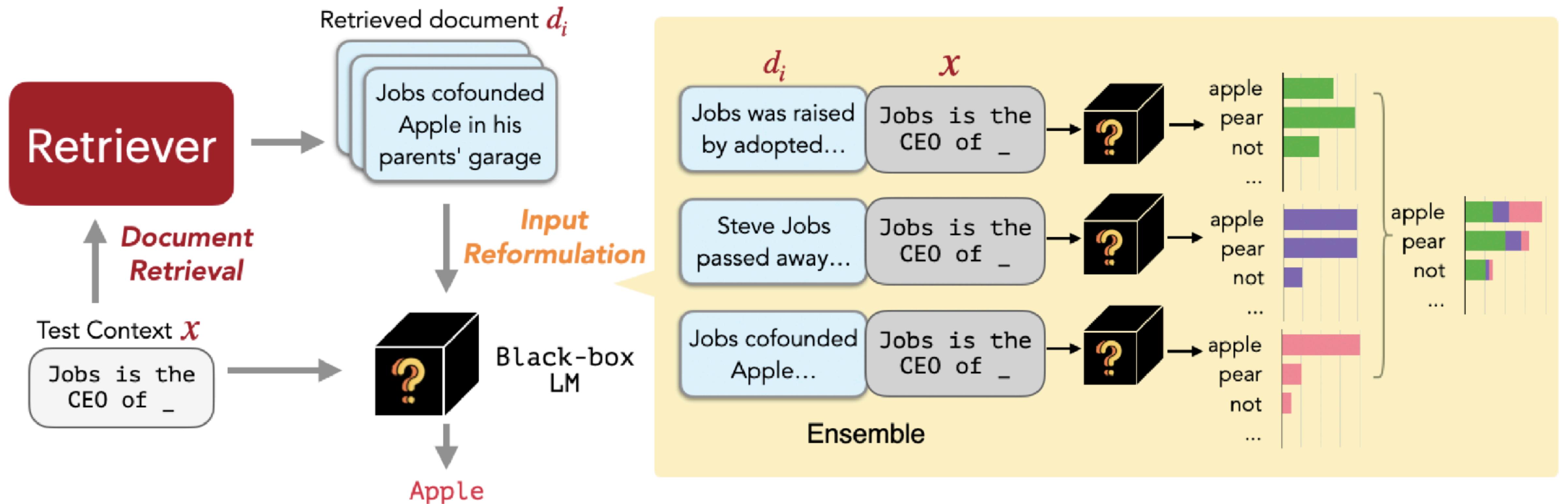
REPLUG (Shi et al. 2023)



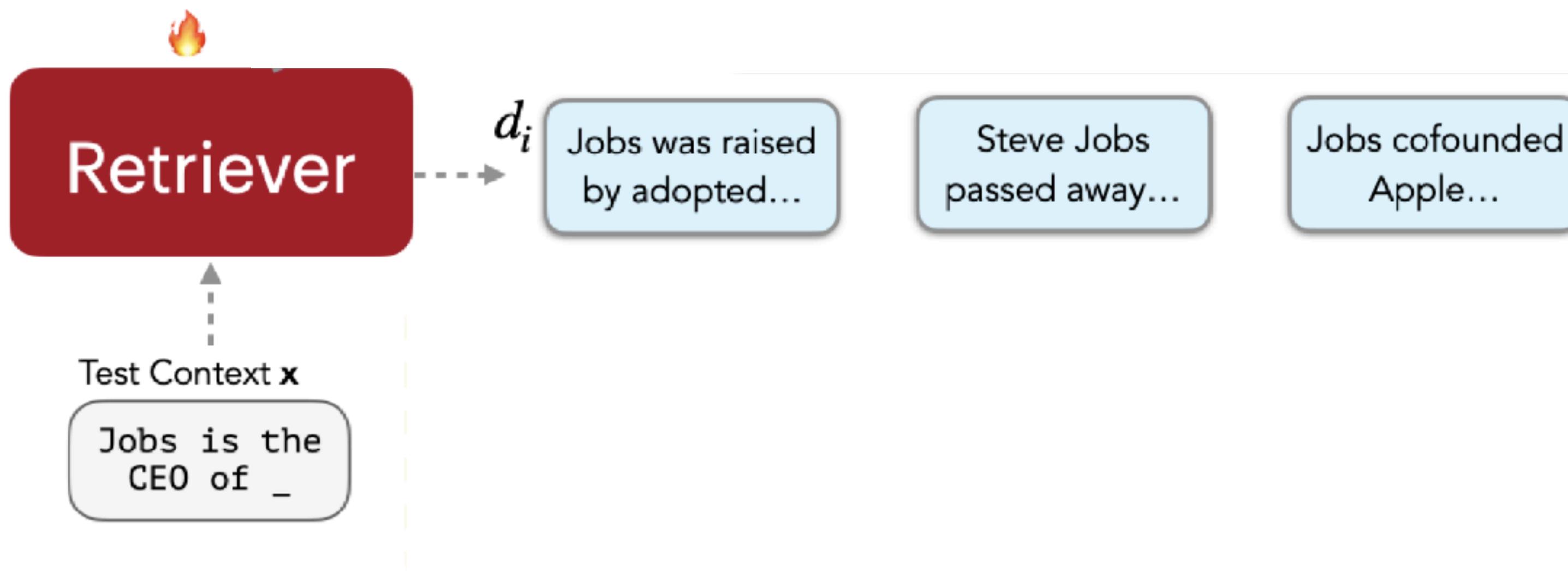
REPLUG (Shi et al. 2023)



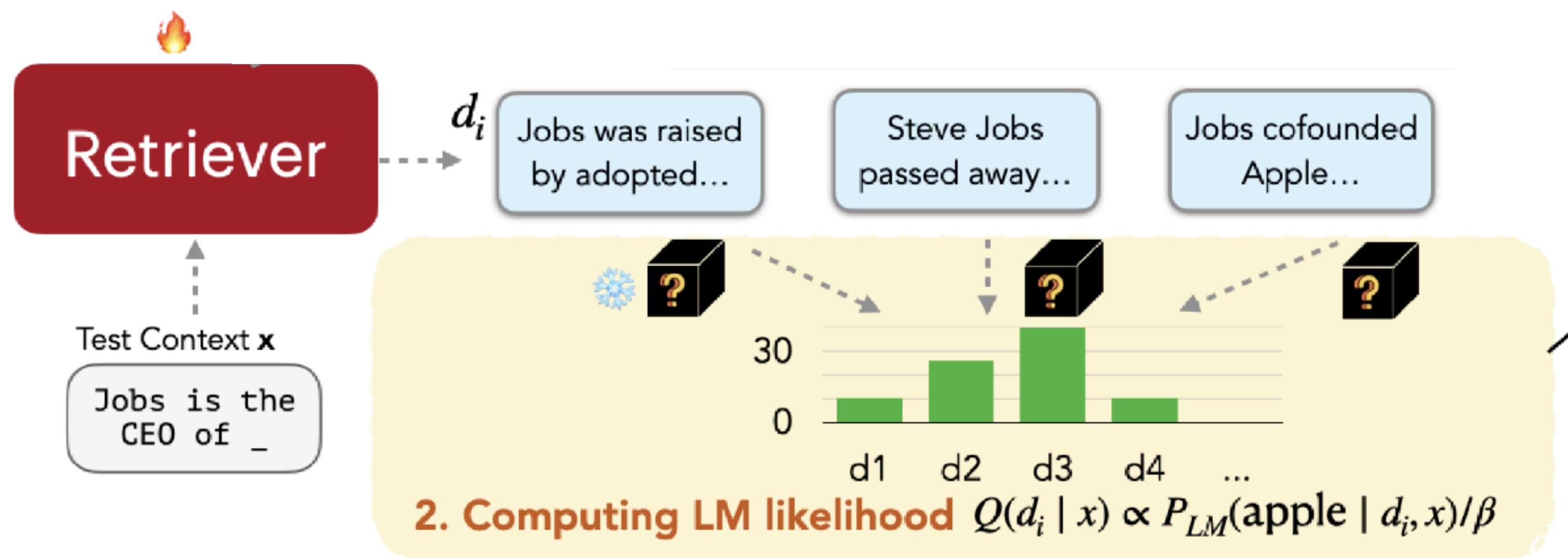
REPLUG (Shi et al. 2023)



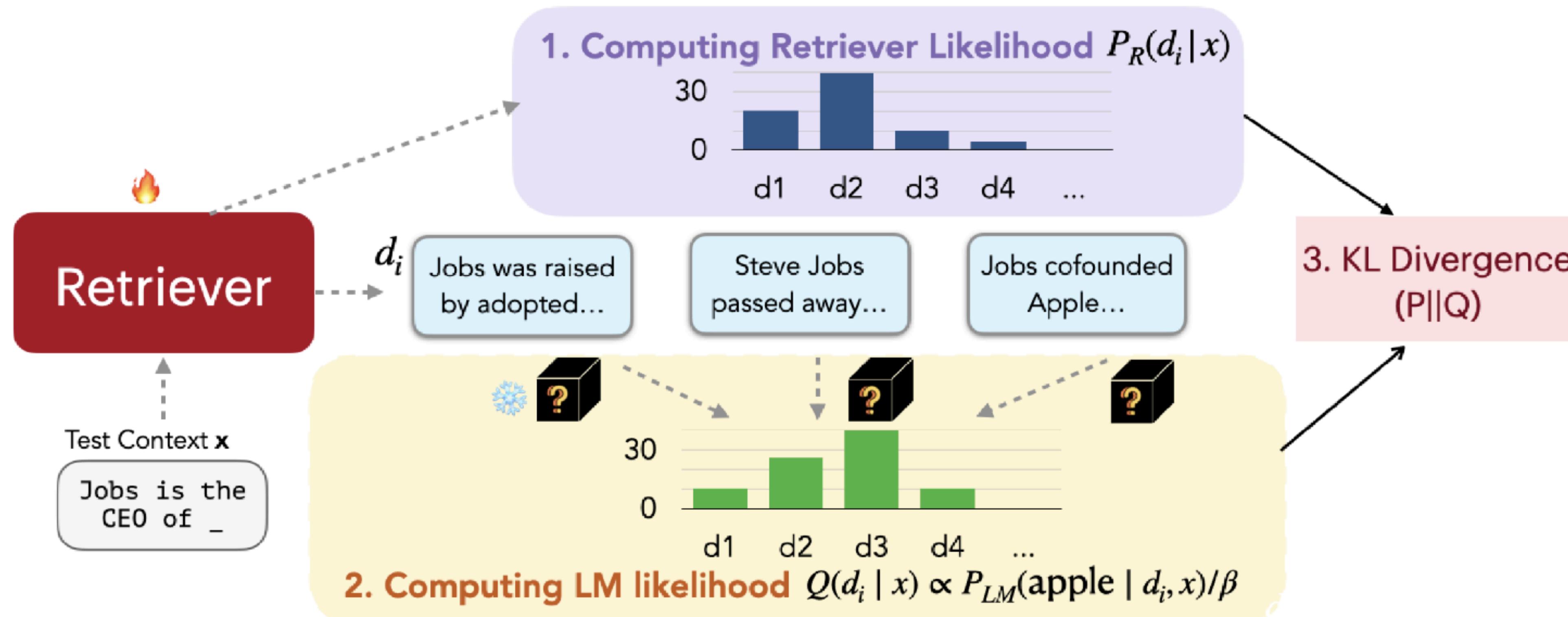
REPLUG LSR (LM-Supervised Retrieval)



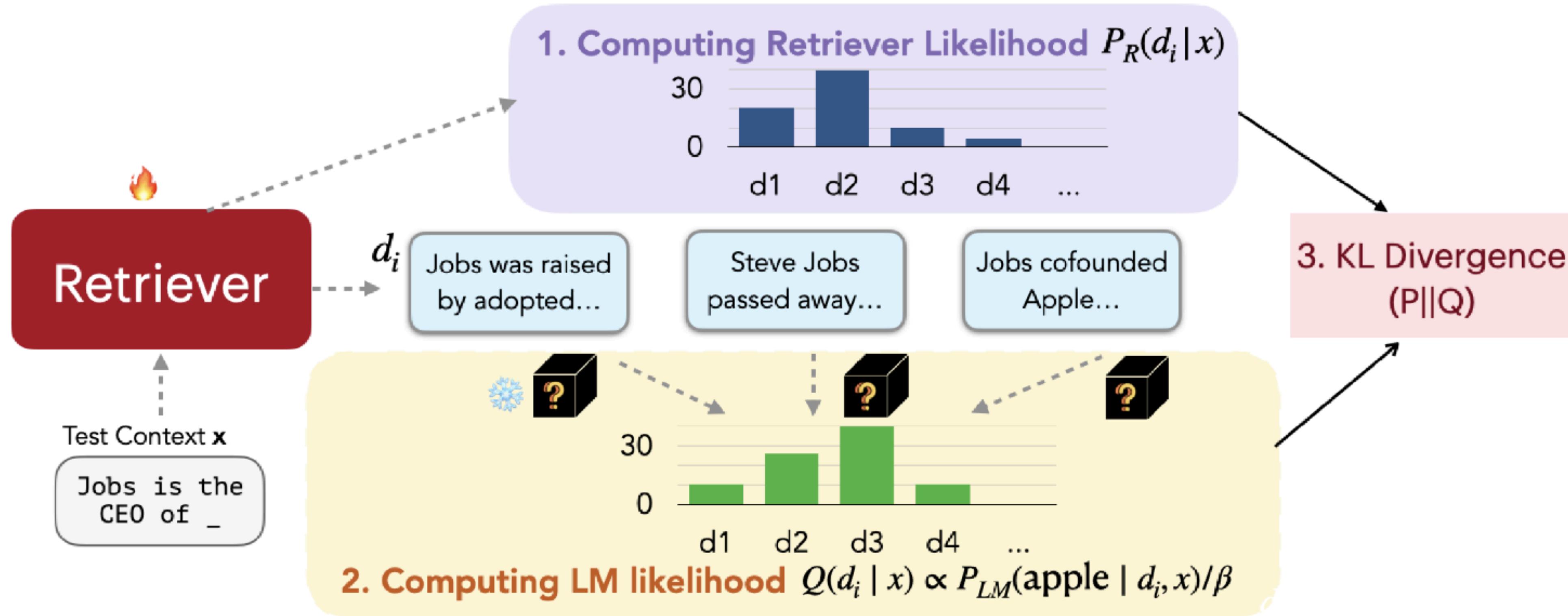
REPLUG LSR (LM-Supervised Retrieval)



REPLUG LSR (LM-Supervised Retrieval)



REPLUG LSR (LM-Supervised Retrieval)



Updating retrieval encoder → Retrieval Index becomes “stale”

How to deal with this issue? We will talk about it soon!

“Asynchronous update”

REPLUG results

Bits per byte (BPB): The lower the better

Model		# Parameters	Original
GPT-2	Small	117M	1.33
	Medium	345M	1.20
	Large	774M	1.19
	XL	1.5B	1.16
GPT-3 (black-box)	Ada	350M	1.05
	Babbage	1.3B	0.95
	Curie	6.7B	0.88
	Davinci	175B	0.80

REPLUG results

With Contriever, “**independent training**”

Model		# Parameters	Original	+ REPLUG	Gain %
GPT-2	Small	117M	1.33	1.26	5.3
	Medium	345M	1.20	1.14	5.0
	Large	774M	1.19	1.15	3.4
	XL	1.5B	1.16	1.09	6.0
GPT-3 (black-box)	Ada	350M	1.05	0.98	6.7
	Babbage	1.3B	0.95	0.90	5.3
	Curie	6.7B	0.88	0.85	3.4
	Davinci	175B	0.80	0.77	3.8

REPLUG results

Fine-tuning Contriever with
LM-supervised training
“Sequential training”

Model		# Parameters	Original	+ REPLUG	Gain %	+ REPLUG LSR	Gain %
GPT-2	Small	117M	1.33	1.26	5.3	1.21	9.0
	Medium	345M	1.20	1.14	5.0	1.11	7.5
	Large	774M	1.19	1.15	3.4	1.09	8.4
	XL	1.5B	1.16	1.09	6.0	1.07	7.8
GPT-3 (black-box)	Ada	350M	1.05	0.98	6.7	0.96	8.6
	Babbage	1.3B	0.95	0.90	5.3	0.88	7.4
	Curie	6.7B	0.88	0.85	3.4	0.82	6.8
	Davinci	175B	0.80	0.77	3.8	0.75	6.3

Sequential training

-  Work with off-the-shelf components (either a large index or a powerful LM)
-  LMs are trained to effectively leverage retrieval results
-  Retrievers are trained to provide text that helps LMs the most
-  One component is still fixed and not trained

Sequential training

-  Work with off-the-shelf components (either a large index or a powerful LM)
-  LMs are trained to effectively leverage retrieval results
-  Retrievers are trained to provide text that helps LMs the most
-  One component is still fixed and not trained

Let's jointly train retrieval models and LMs!

Q & A



coffee break

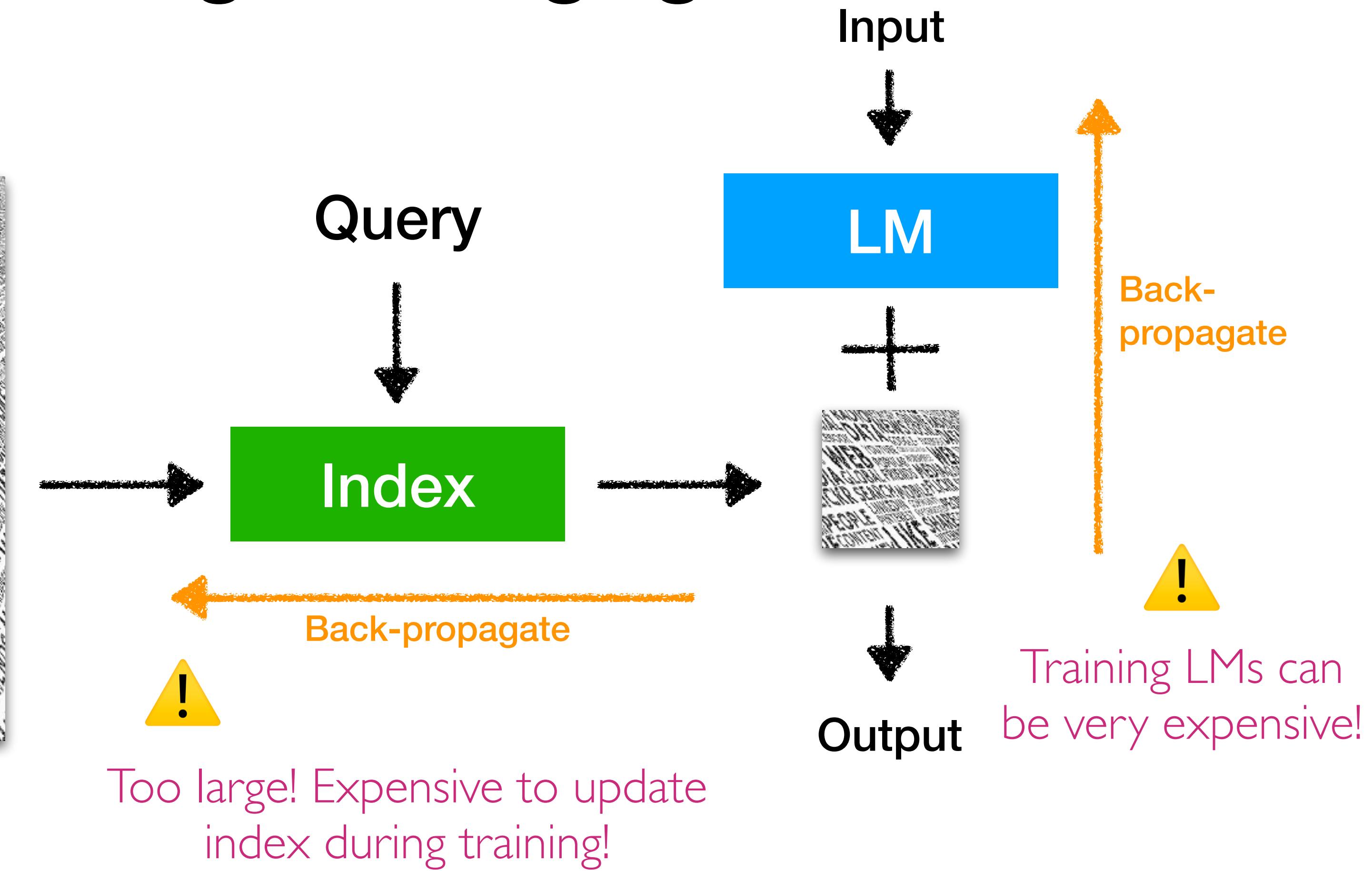
We'll be back at 4:00pm!

Section 4: Retrieval-based LMs: Training (cont'd)

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

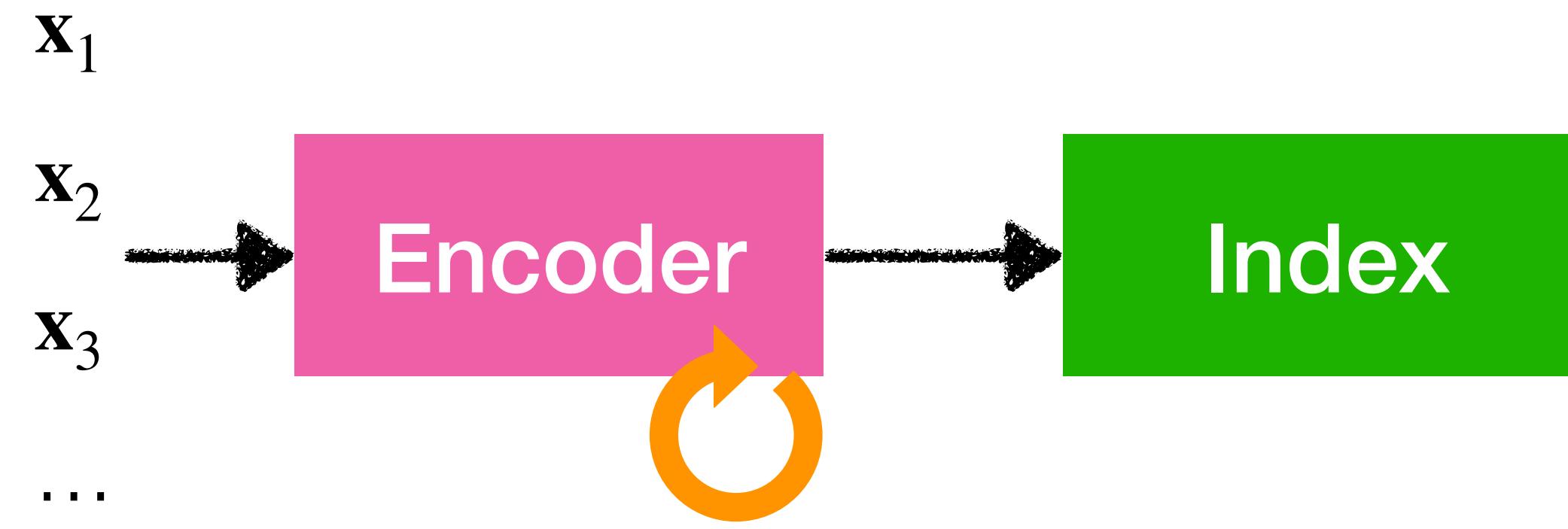
Training methods for retrieval-based LMs

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

Challenges of updating retrieval models

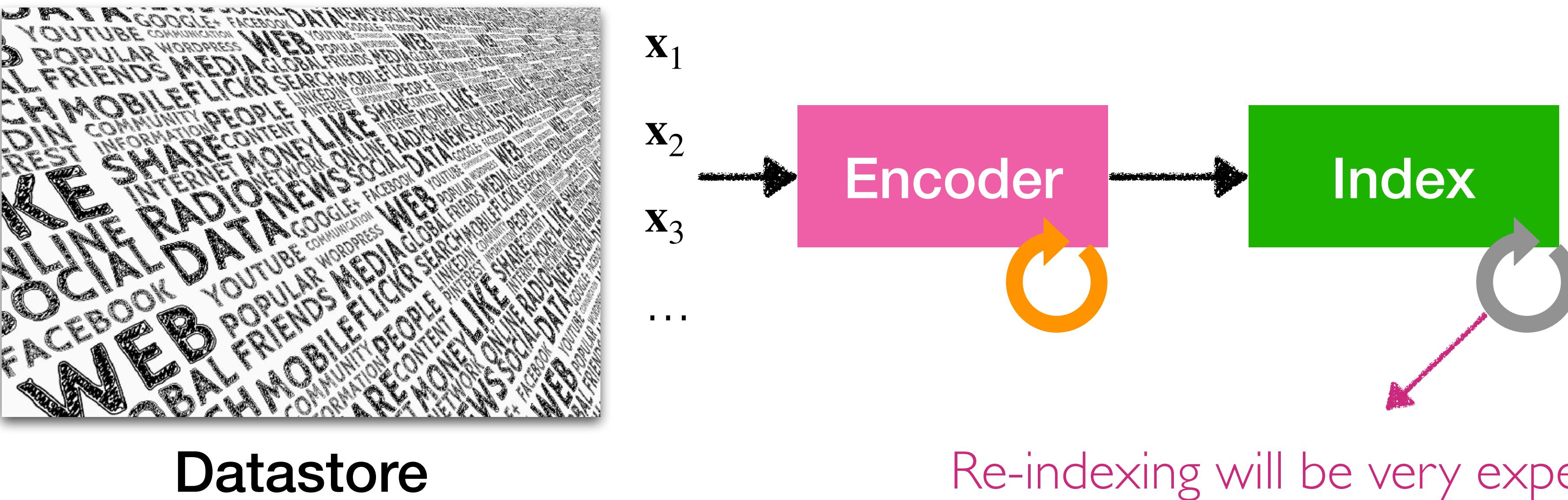


Datastore



During training, we will update the encoder

Challenges of updating retrieval models

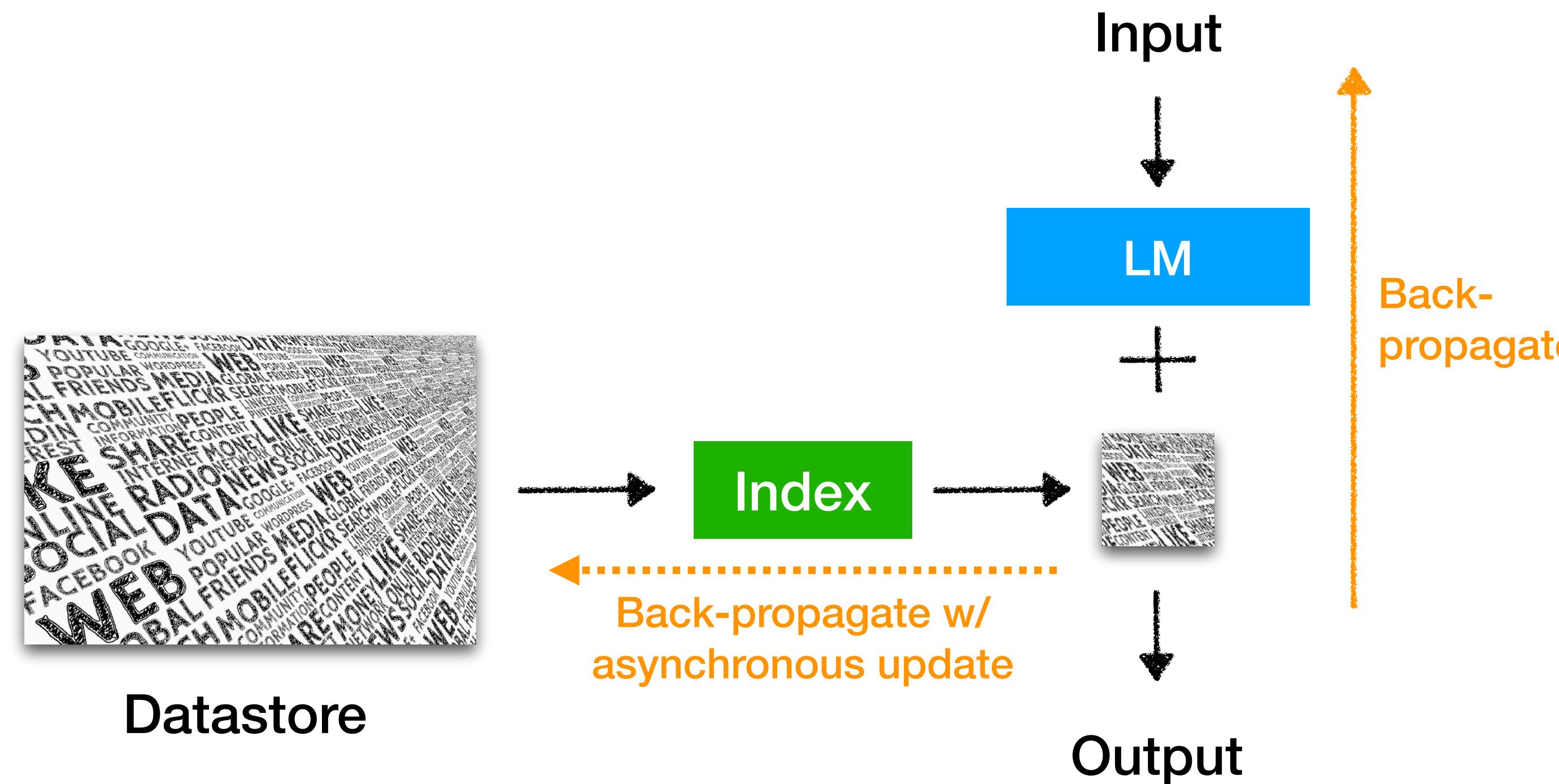


Training methods for retrieval-based LMs

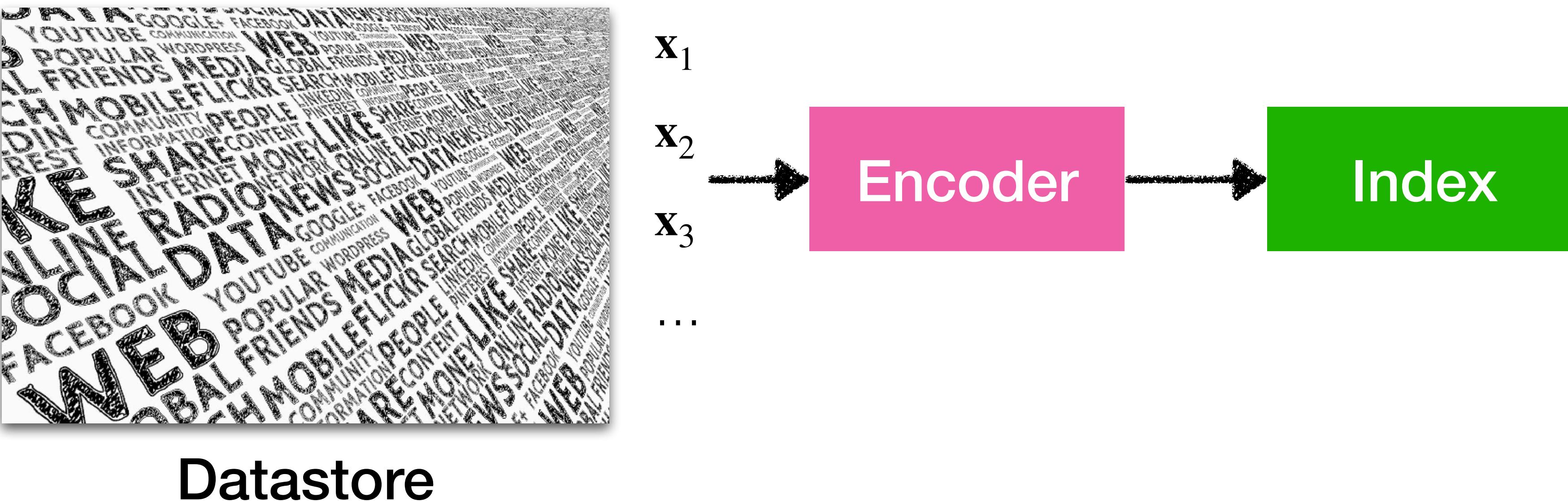
- Independent training
- Sequential training
- **Joint training w/ asynchronous index update**
- Joint training w/ in-batch approximation

Joint training w/ asynchronous index update

- Retrieval models and language models are trained jointly
- Allow the index to be “**stale**”; rebuild the retrieval index every T steps



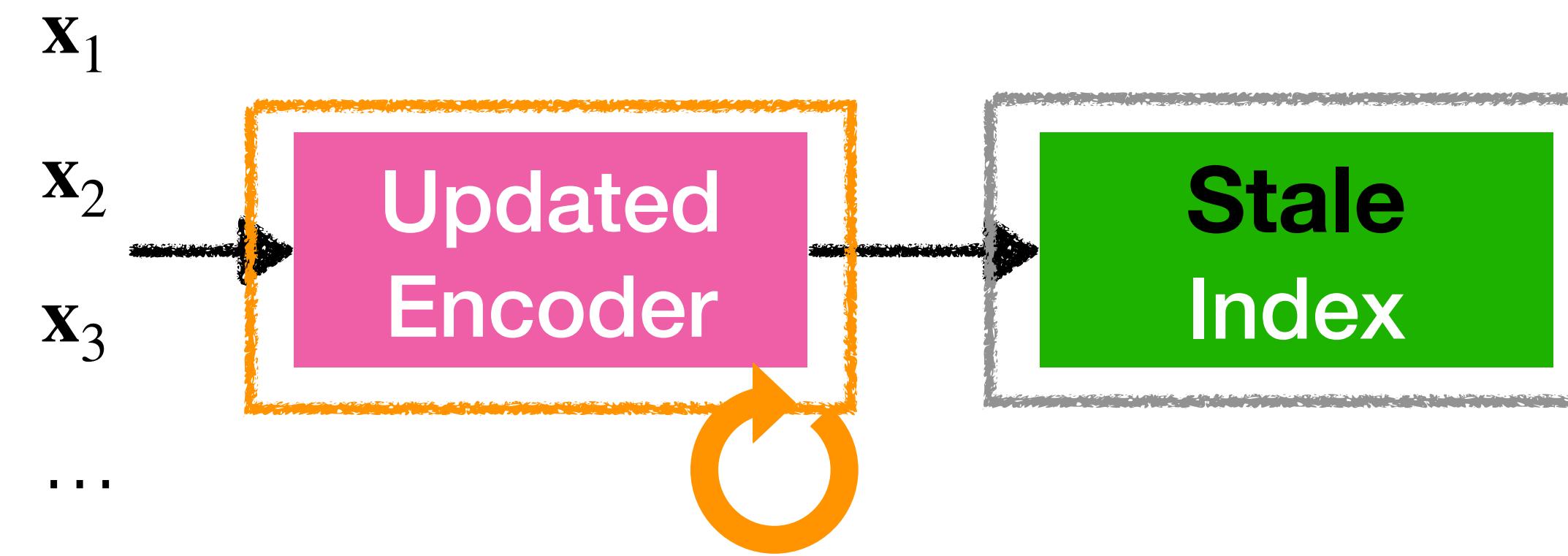
Asynchronous index update



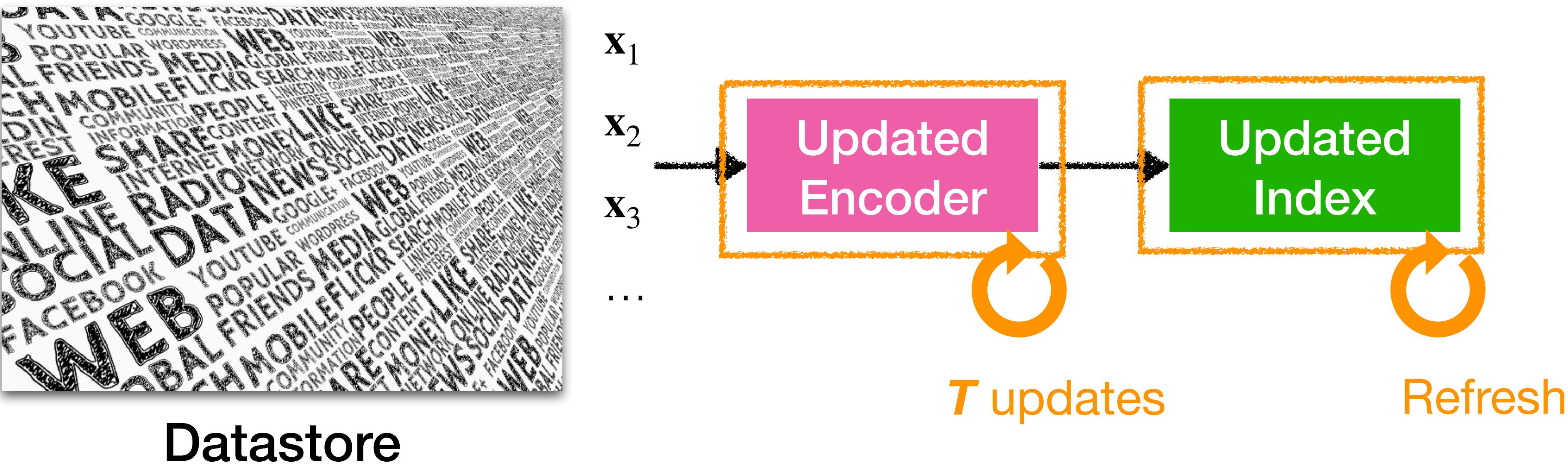
Asynchronous index update



Datastore

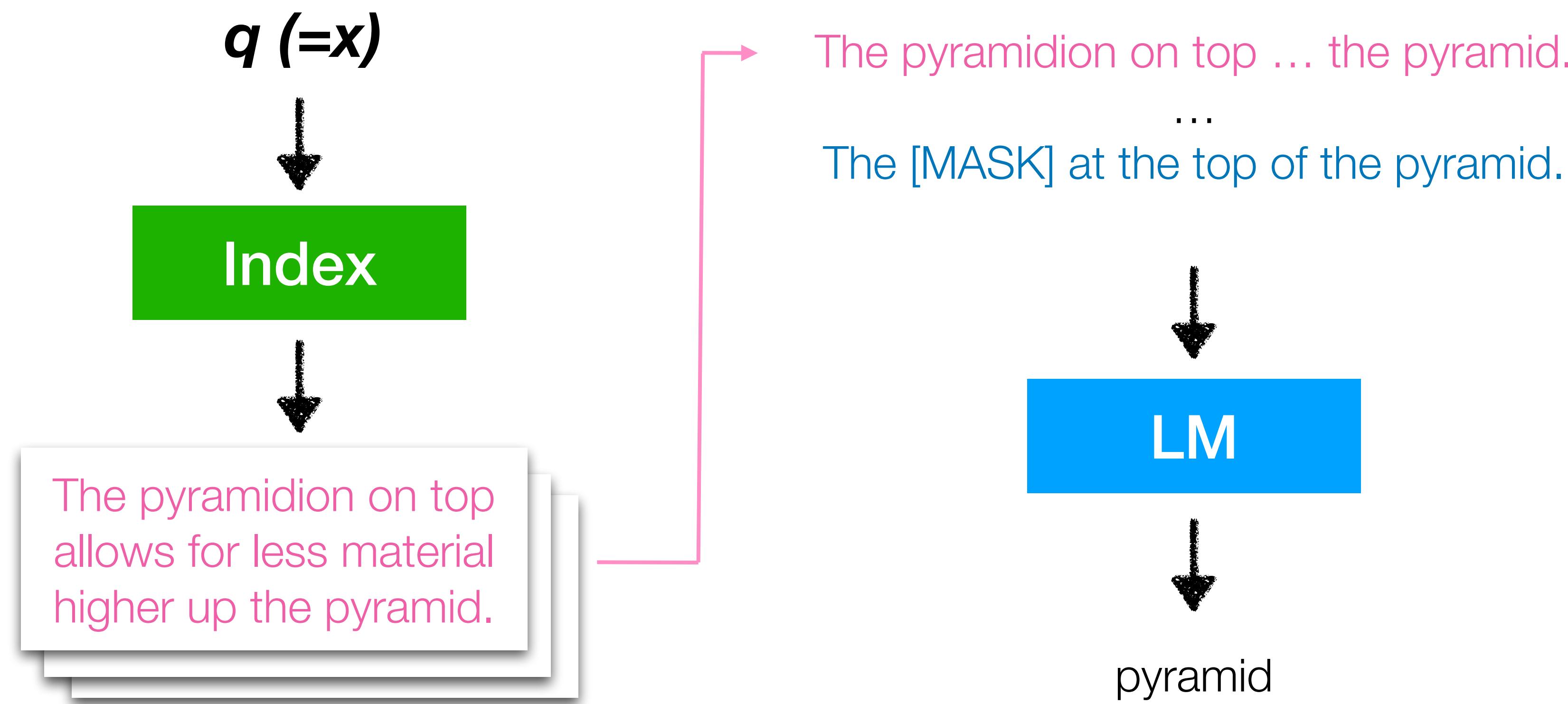


Asynchronous index update



REALM (Guu et al. 2020)

x = The [MASK] at the top of the pyramid.

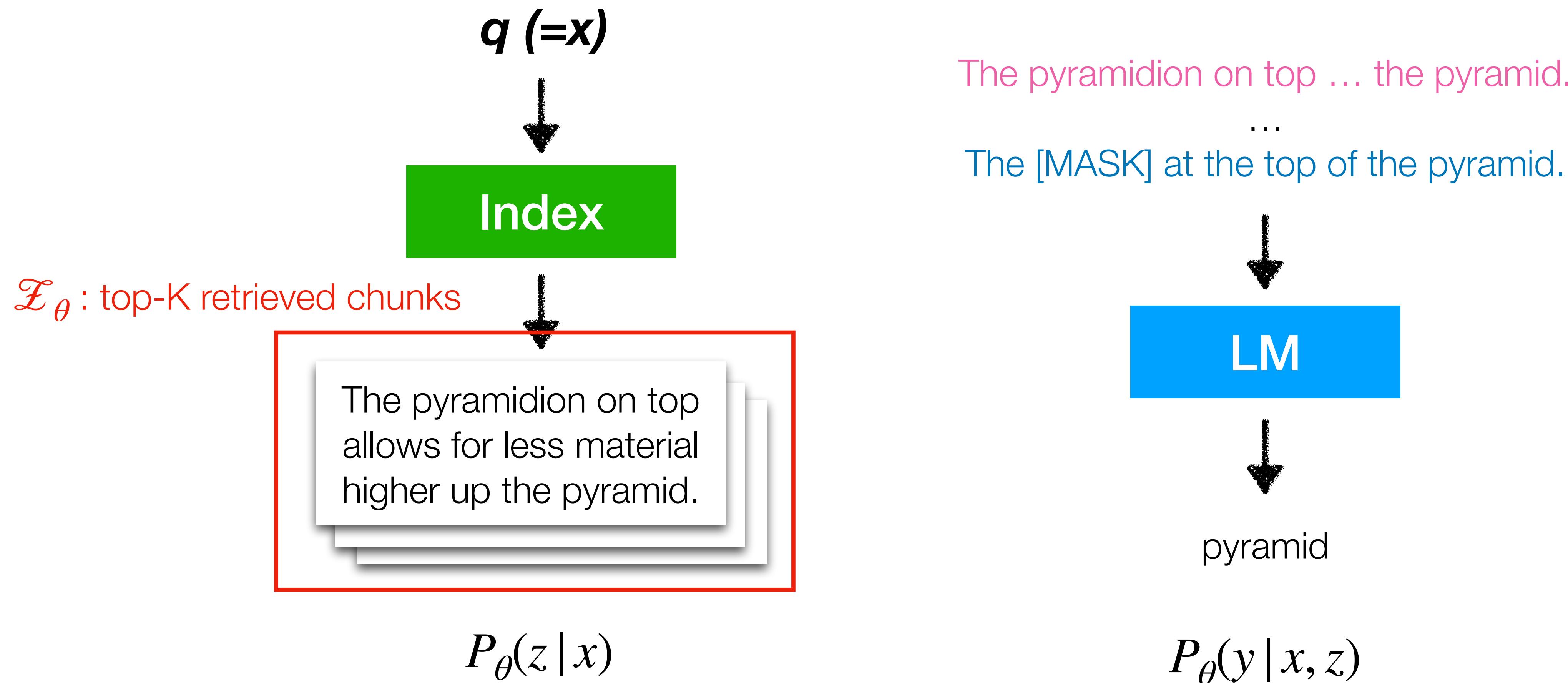


$$P(z | x)$$

$$P(y | x, z)$$

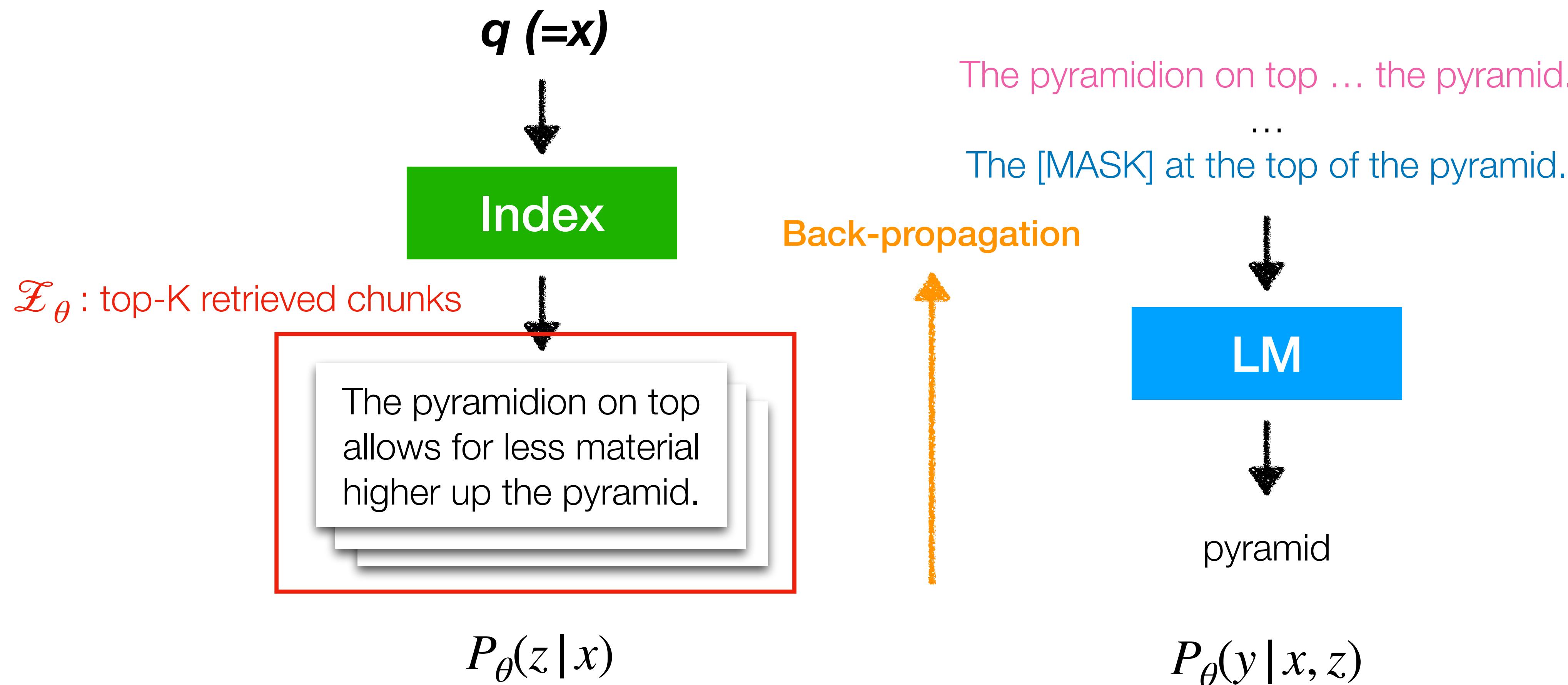
REALM: Training

Objective: maximize $\sum_{z \in \mathcal{Z}_\theta} P_\theta(z | q) P_\theta(y | q, z)$



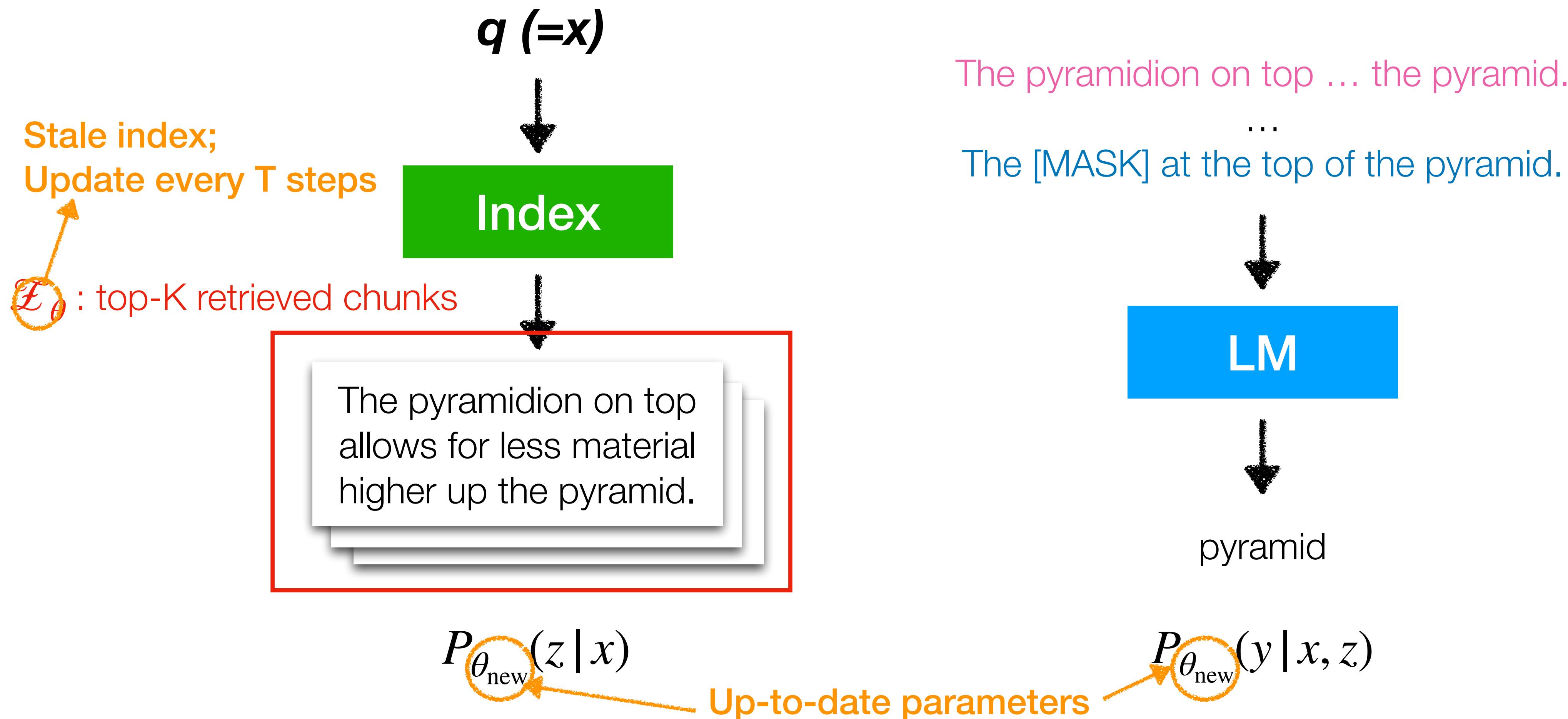
REALM: Training

Objective: maximize $\sum_{z \in \mathcal{Z}_\theta} P_\theta(z | q)P_\theta(y | q, z)$



REALM: Training

Objective: maximize $\sum_{z \in \mathcal{Z}_\theta} P_\theta(z | q) P_\theta(y | q, z)$



REALM: Index update rate

How often should we update the retrieval index?

- Frequency too high: expensive
- Frequency too slow: out-dated

REALM: Index update rate

How often should we update the retrieval index?

- Frequency too high: expensive
- Frequency too slow: out-dated

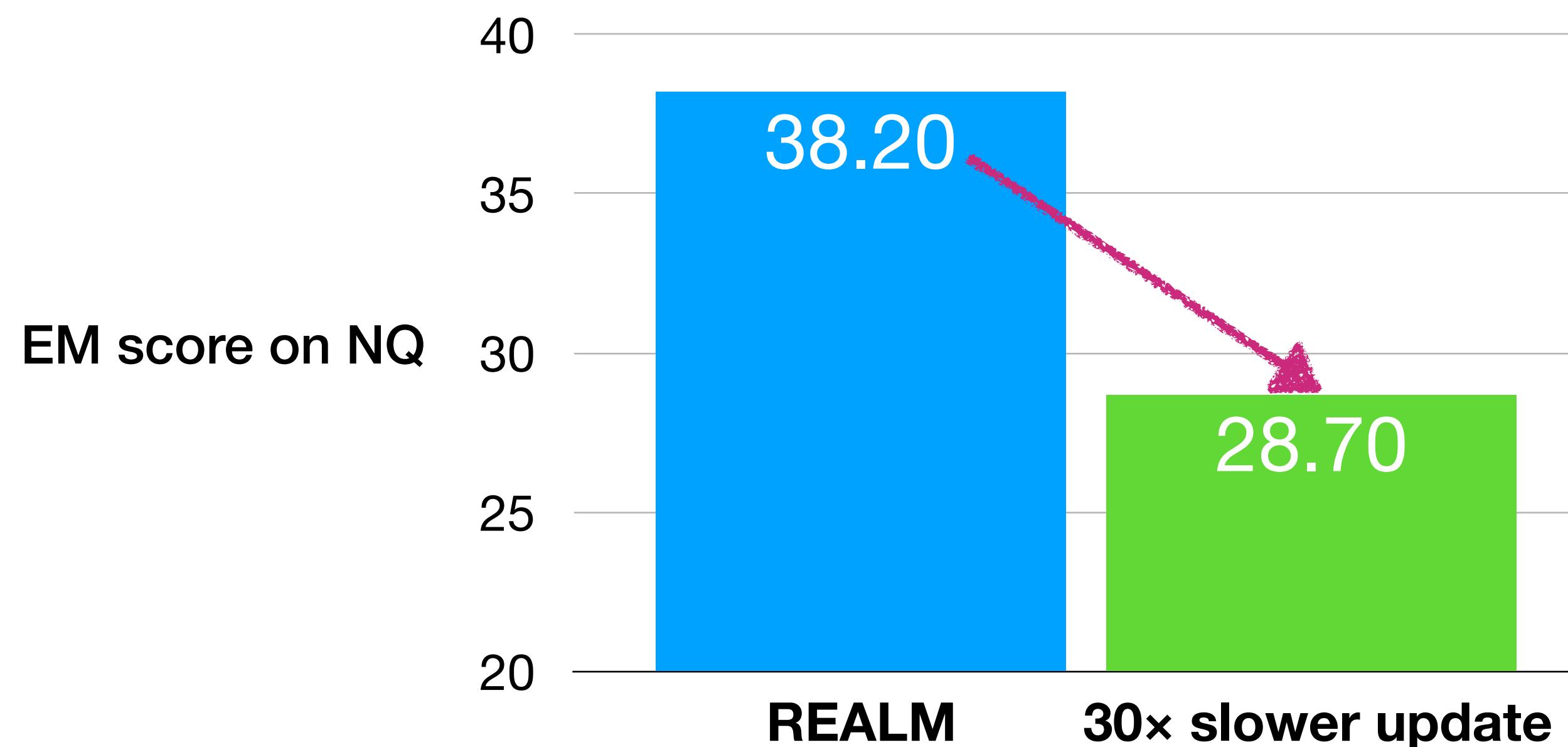
REALM: updating the index every 500 training steps

REALM: Index update rate

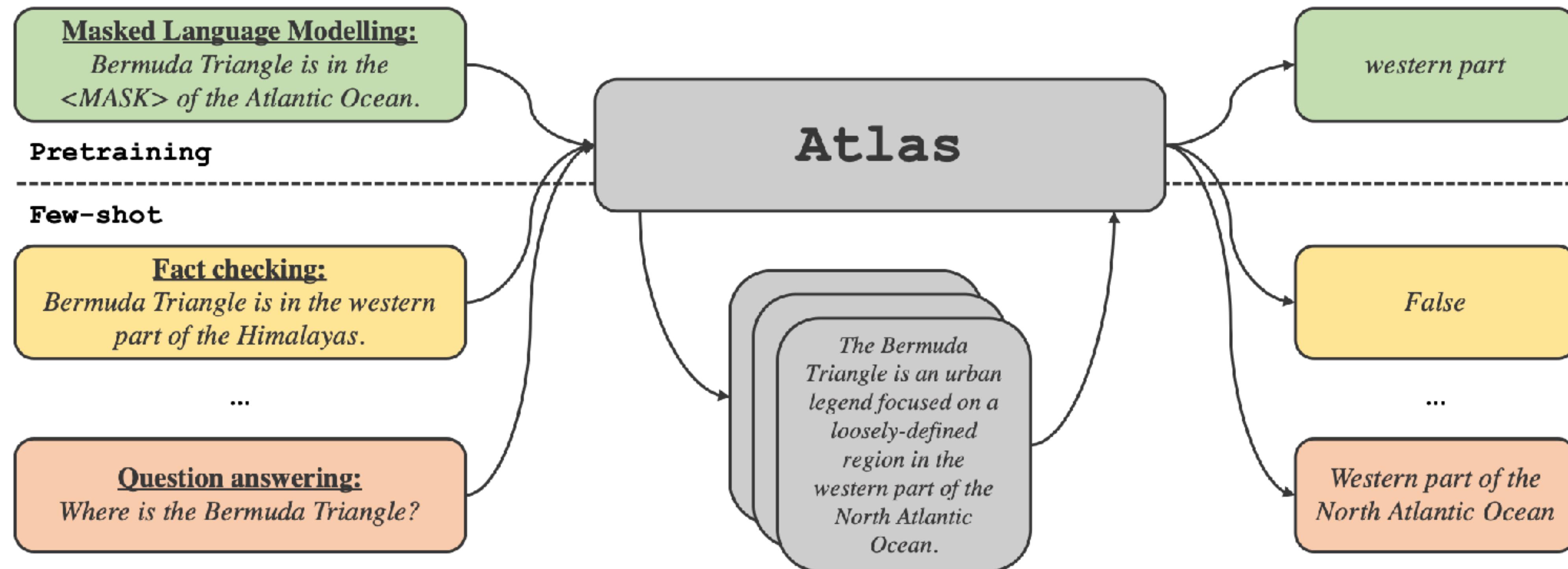
How often should we update the retrieval index?

- Frequency too high: expensive
- Frequency too slow: out-dated

REALM: updating the index every 500 training steps



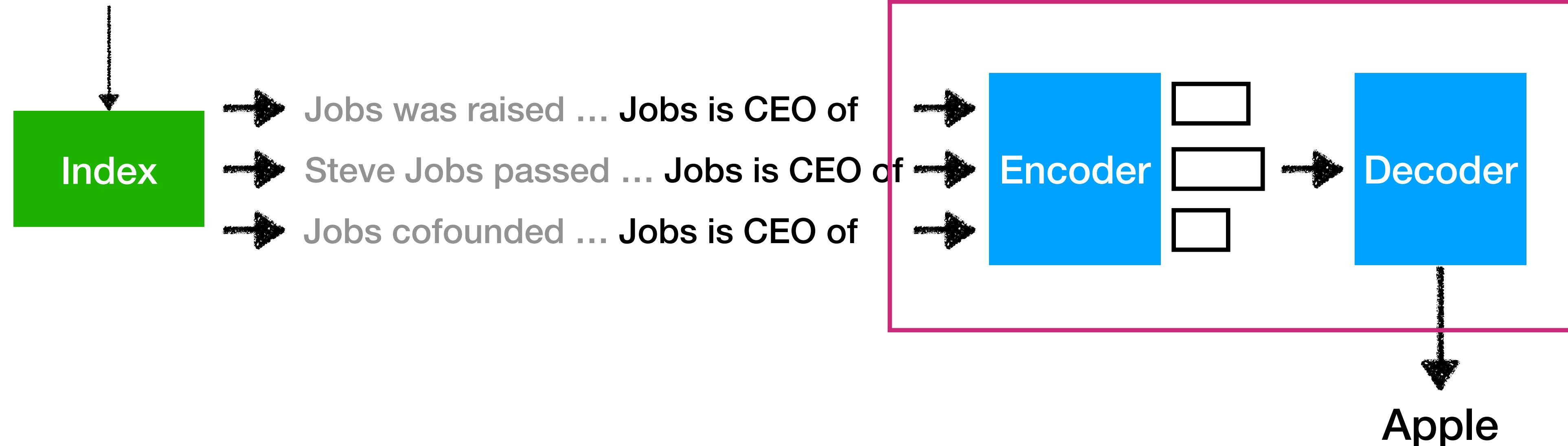
Atlas (Izacard et al. 2022)



Atlas (Izacard et al. 2022)

Retrieval-based encoder-decoder model

Jobs is CEO of _

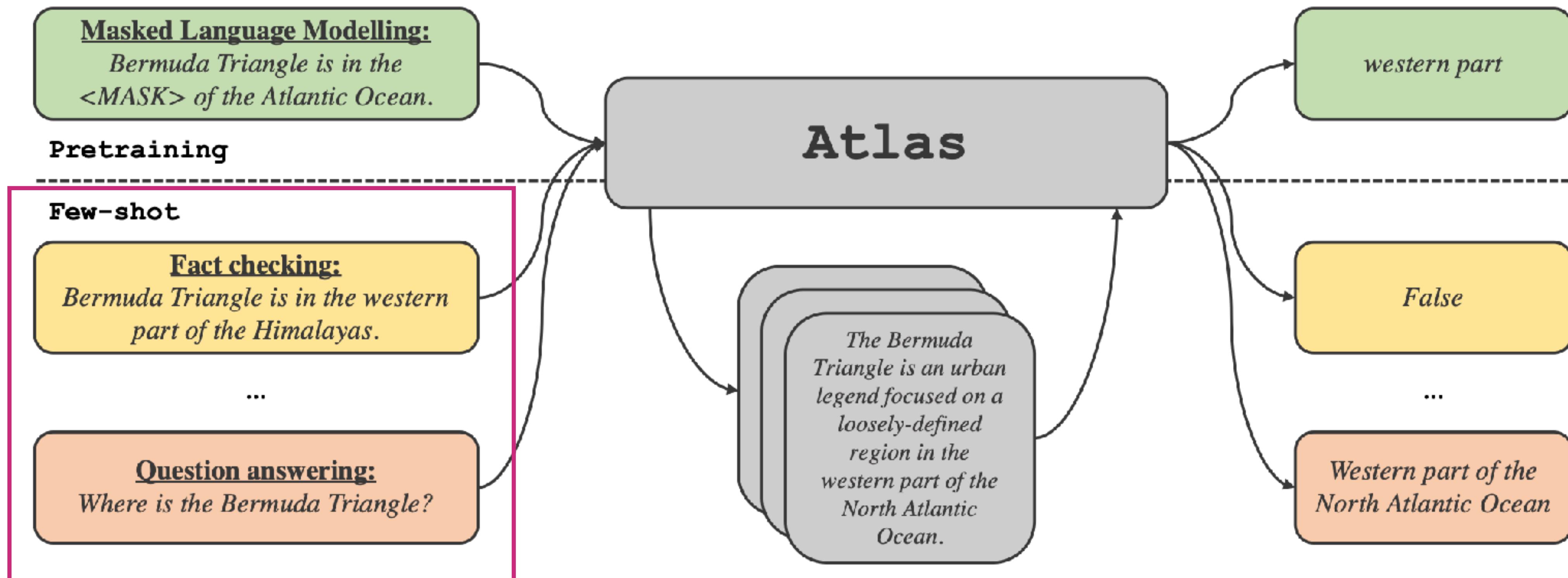


Atlas (Izacard et al. 2022)

Jobs is CEO of _



Atlas (Izacard et al. 2022)



Adapted to a lot of downstream tasks! (Section 5)

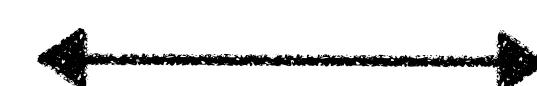
Atlas: Retriever training

Perplexity Distillation

Retrieve the text that can help LM encoders improve perplexity

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

How likely each document is retrieved



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

How much each document improves the ppl

Atlas: Retriever training

Similarity based on retrieval encoder

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

How likely each document is retrieved

KL Divergence

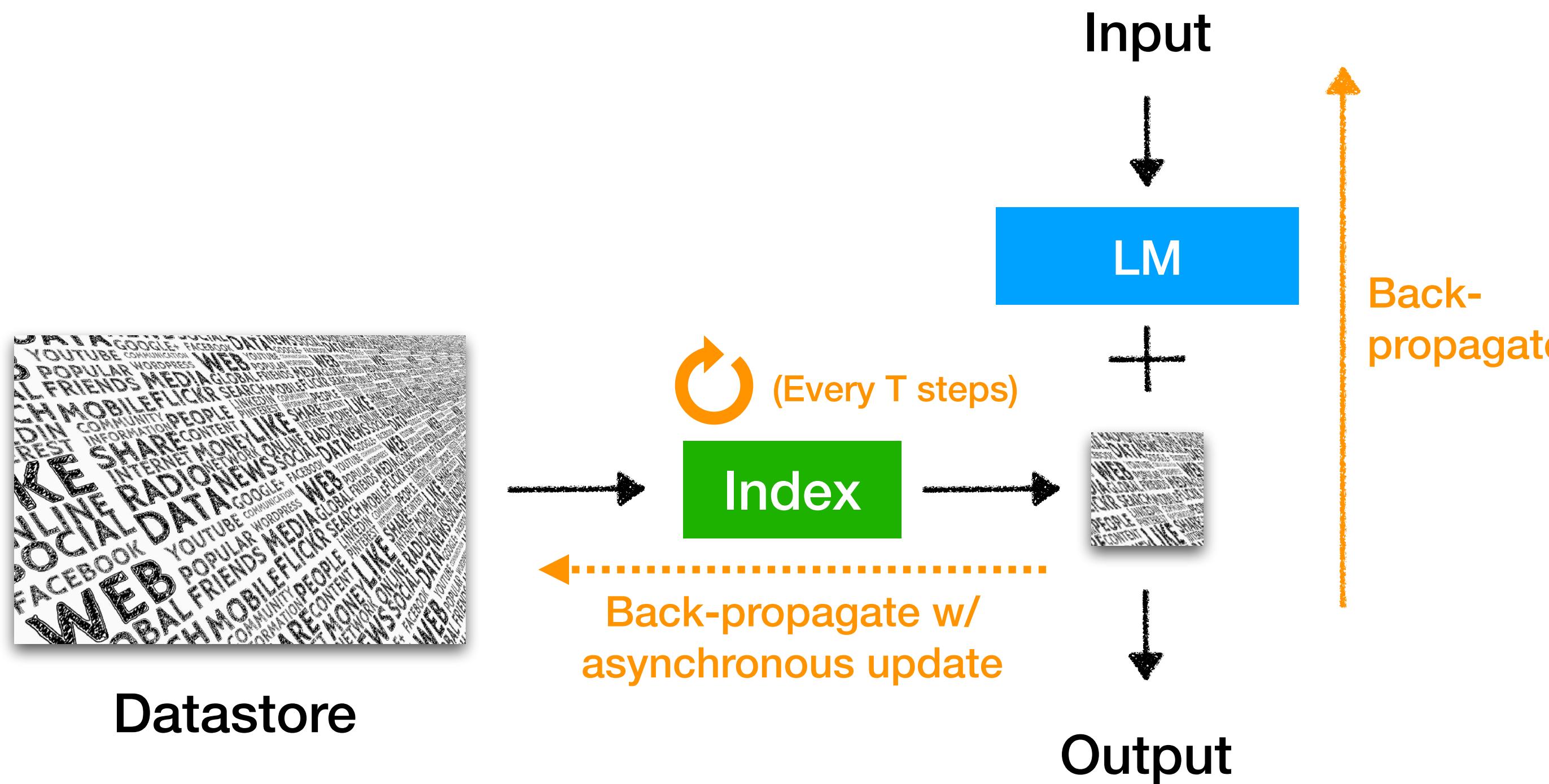
Prob of the gold labels if augmenting this text chunk

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

How much each document improves the ppl

Perplexity Distillation

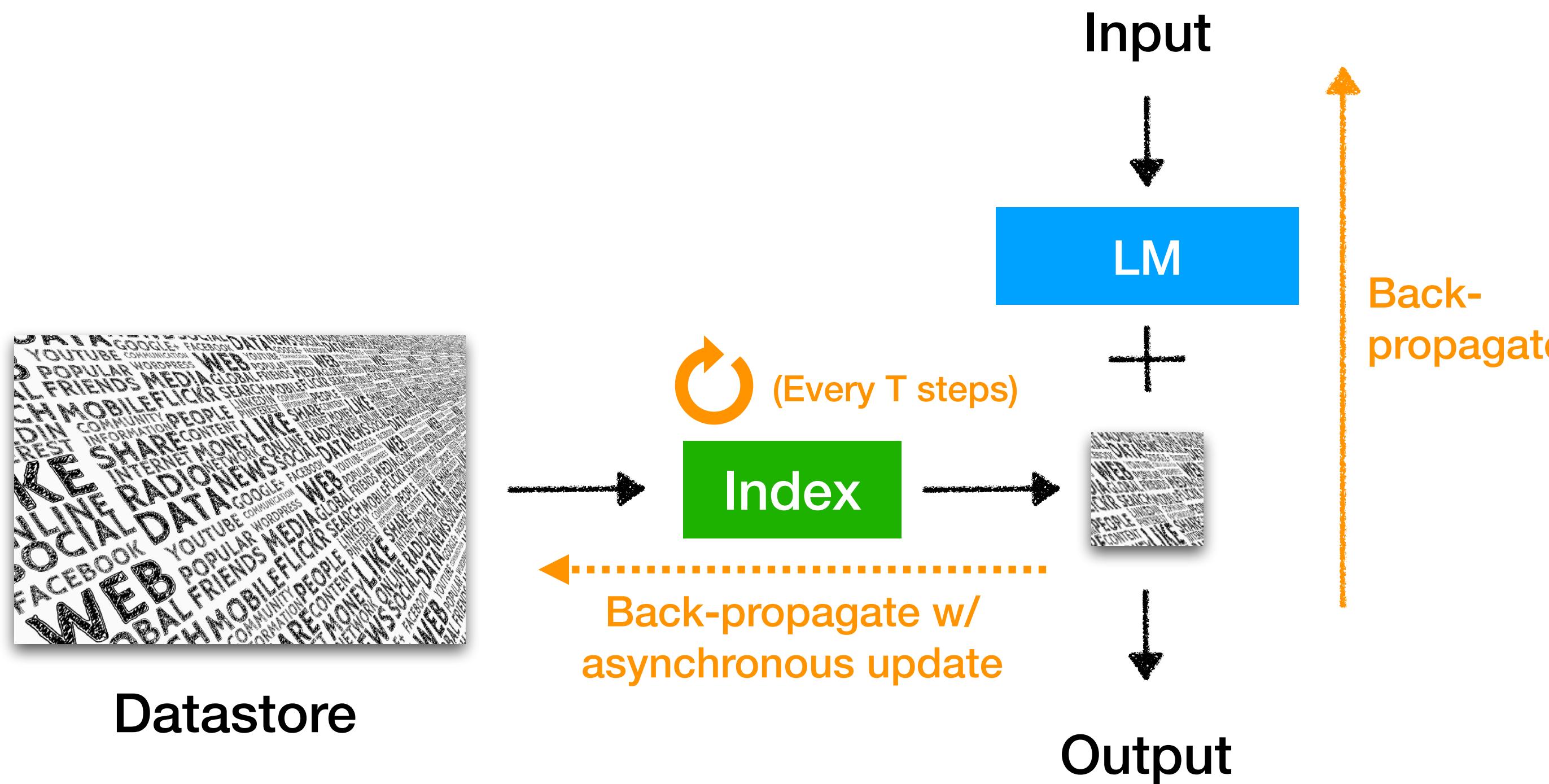
Atlas: Asynchronous index update



Update the index every T steps

30% overhead for asynchronous update on Wikipedia

Atlas: Asynchronous index update



Update the index every T steps

30% overhead for asynchronous update on Wikipedia

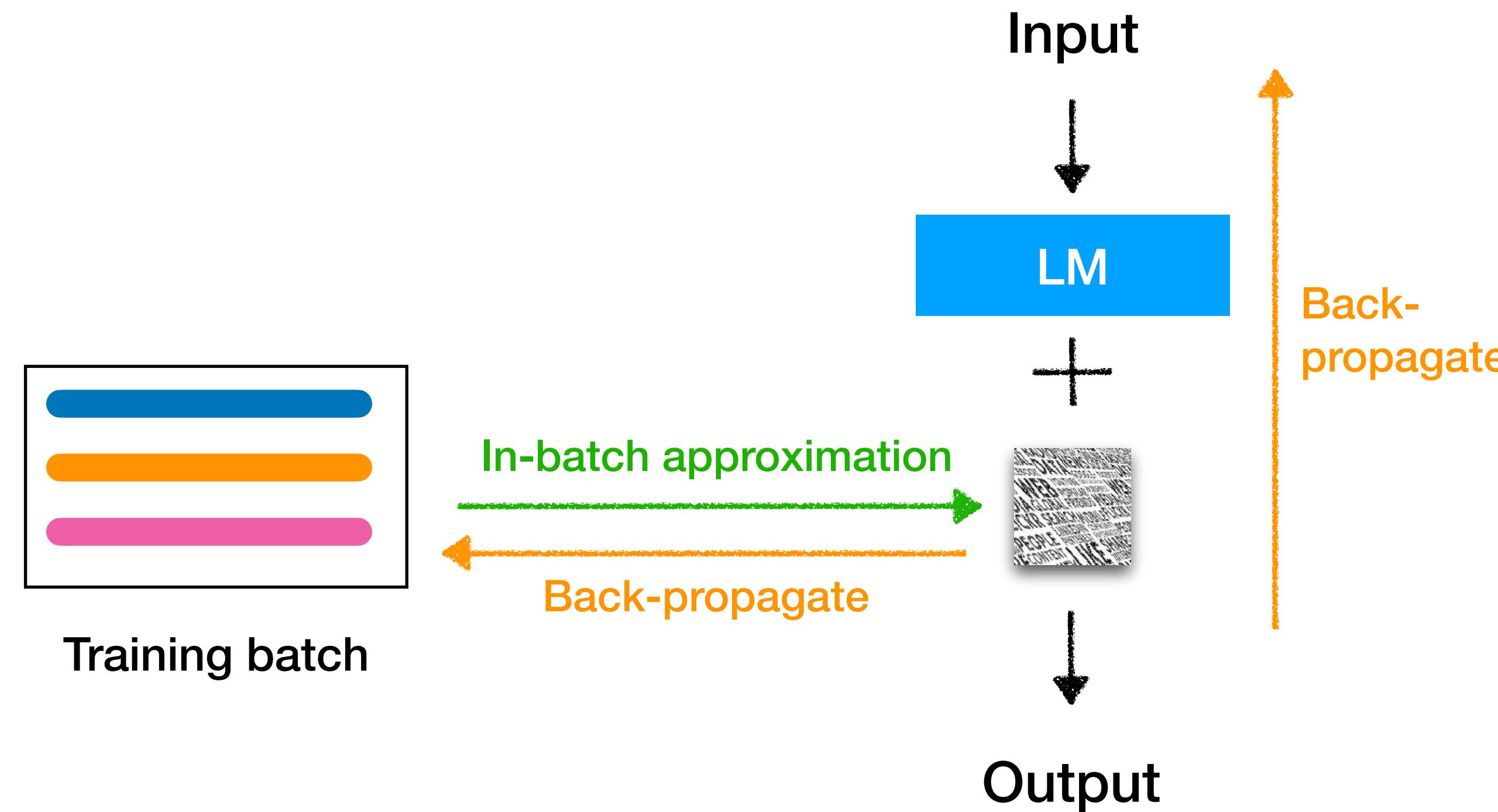
How can we get rid of this?

Training methods for retrieval-based LMs

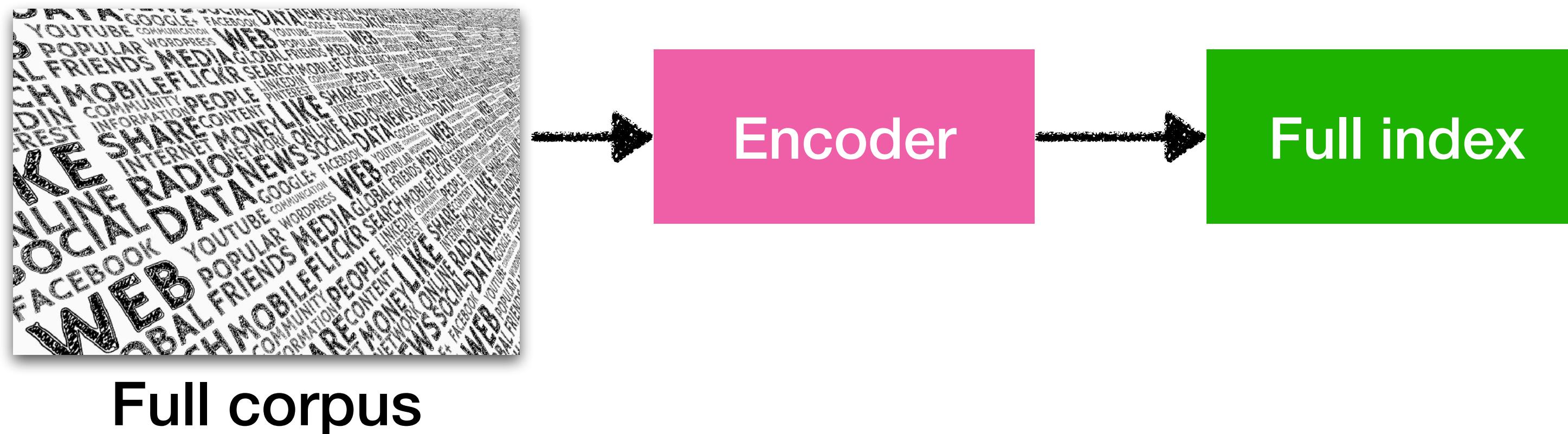
- Independent training
- Sequential training
- Joint training w/ asynchronous index update
- **Joint training w/ in-batch approximation**

Joint training w/ in-batch approximation

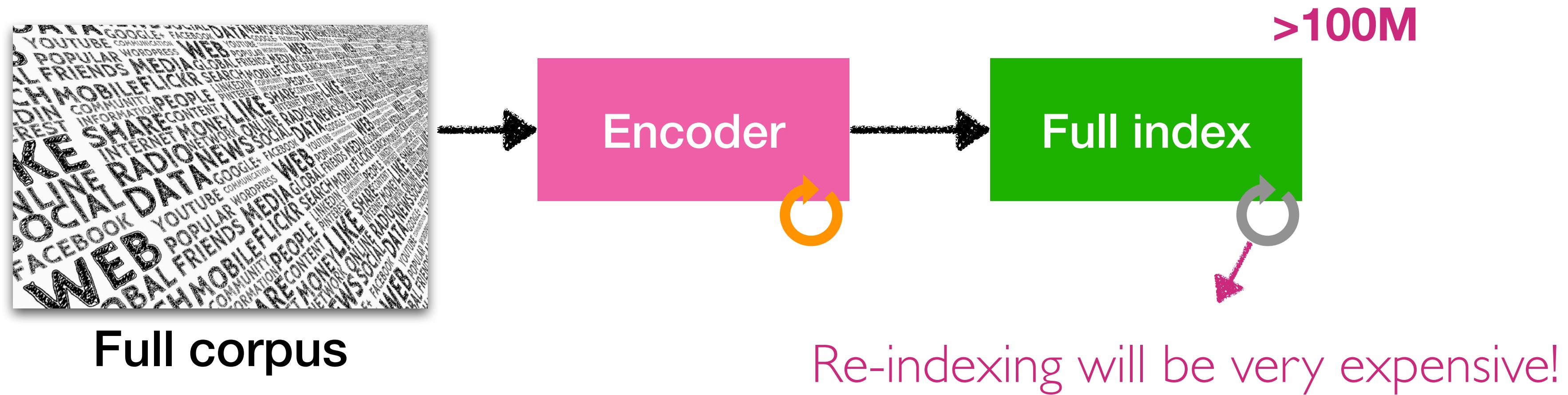
- Retrieval models and language models are trained jointly
- Use “in-batch index” instead of full index



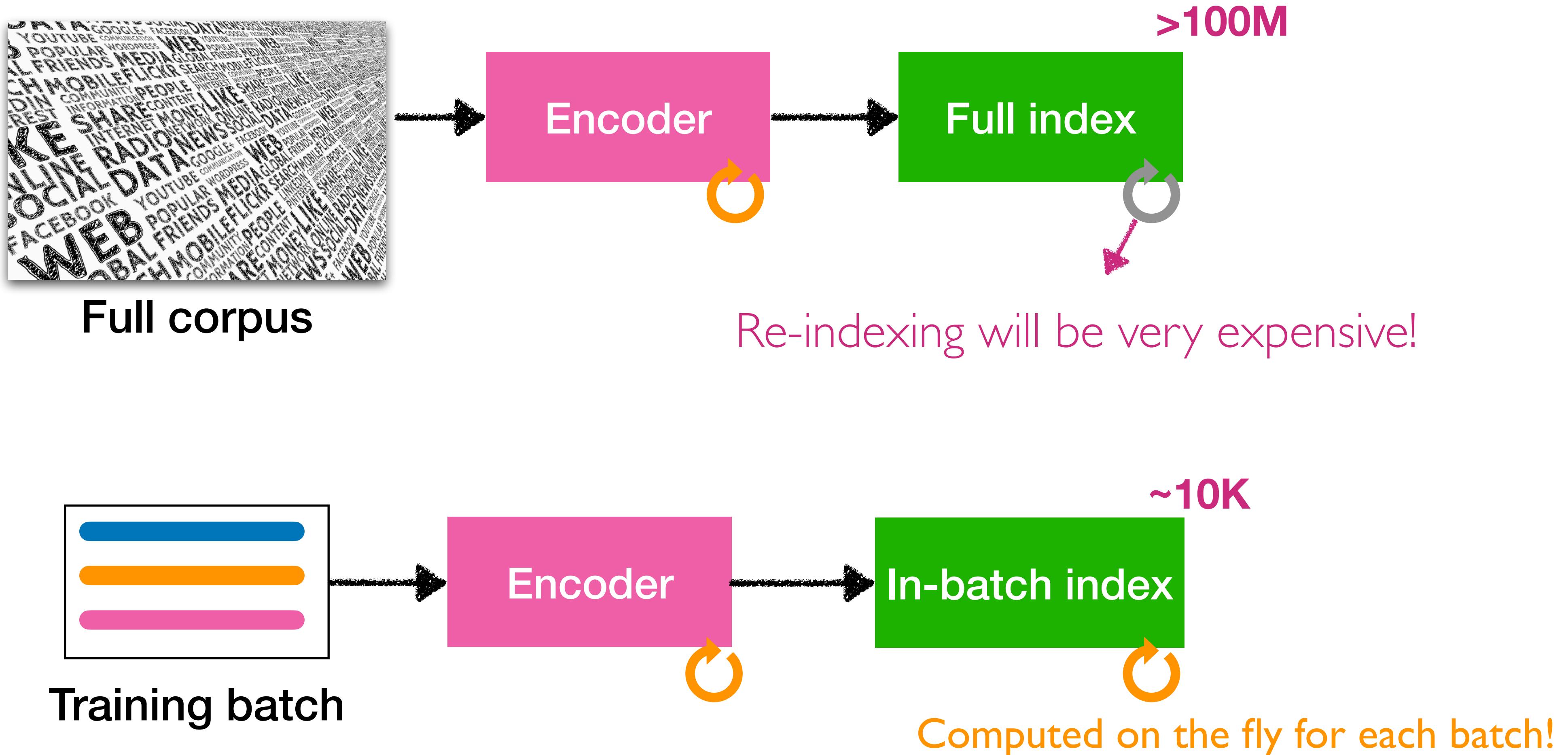
In-batch approximation



In-batch approximation



In-batch approximation

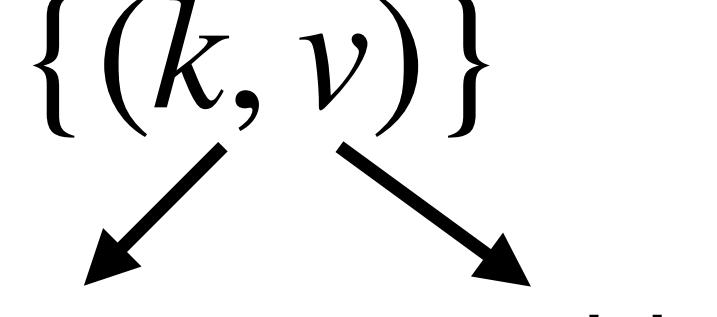


TRIME:Training with in-batch memory (Zhong et al. 2022)

Similar to kNN-LM

Datastore $\mathcal{D} = \{(k, v)\}$

context (chunk) next token



Keys	Values
<i>10/10, would buy this</i>	<i>cheap</i>
<i>Item delivered broken. Very</i>	<i>cheap</i>
<i>To check the version of PyTorch, you can use</i>	<i>torch</i>
<i>You are permitted to bring a</i>	<i>torch</i>
<i>A group of infections ... one of the</i>	<i>torch</i>

TRIME:Training with in-batch memory (Zhong et al. 2022)

Similar to kNN-LM

Datastore $\mathcal{D} = \{(k, v)\}$

context (chunk) next token

Keys	Values
<i>10/10, would buy this</i>	<i>cheap</i>
<i>Item delivered broken. Very</i>	<i>cheap</i>
<i>To check the version of PyTorch, you can use</i>	<i>torch</i>
<i>You are permitted to bring a</i>	<i>torch</i>
<i>A group of infections ... one of the</i>	<i>torch</i>

Inference

$$P(y | x) \propto \exp(E^\top f(x)) + \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \exp(-d(\text{Enc}(k), \text{Enc}(x)))$$

output embedding
(same as standard LMs)

datastore
(very large!)

TRIME: Training with in-batch memory (Zhong et al. 2022)

Similar to kNN-LM

Datastore $\mathcal{D} = \{(k, v)\}$

context (chunk) next token

Keys	Values
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

Inference

$$P(y | x) \propto \exp(E^\top f(x)) + \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \exp(-d(\text{Enc}(k), \text{Enc}(x)))$$

output embedding
(same as standard LMs)

datastore
(very large!)

1. Aligning the output representations with static embeddings

2. Aligning input chunks with all the chunks in datastore that share the same next token

TRIME:Training with in-batch memory (Zhong et al. 2022)

$$P(y | x) \propto \exp(E^\top f(x)) + \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \exp(-d(\text{Enc}(k), \text{Enc}(x)))$$

Jobs become CEO of Apple \longleftrightarrow He moves to Apple

Positive chunks \rightarrow pull together

Jobs become CEO of Apple \longleftrightarrow She works at Microsoft

Negative chunks \rightarrow push away

TRIME:Training with in-batch memory (Zhong et al. 2022)

$$P(y | x) \propto \exp(E^\top f(x)) + \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \exp(-d(\text{Enc}(k), \text{Enc}(x)))$$

Very large!

Jobs become CEO of Apple \longleftrightarrow He moves to Apple

Positive chunks \rightarrow pull together

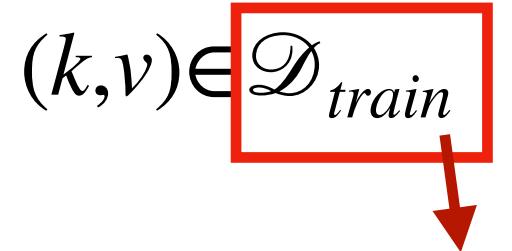
Jobs become CEO of Apple \longleftrightarrow She works at Microsoft

Negative chunks \rightarrow push away

TRIME: Training

Key idea: build a temporary index from same training batch on the fly

$$P(y \mid x) \propto \exp(E^\top f(x)) + \sum_{(k,v) \in \mathcal{D}_{train}} \mathbb{I}[v = y] (-d(\text{Enc}(k), \text{Enc}(x)))$$



In-batch approximation

(built from in-batch examples on the fly)

TRIME: Training

Key idea: build a temporary index from same training batch on the fly

$$P(y \mid x) \propto \exp(E^\top f(x)) + \sum_{(k,v) \in \mathcal{D}_{train}} \mathbb{I}[v = y] (-d(\text{Enc}(k), \text{Enc}(x)))$$

In-batch approximation

(built from in-batch examples on the fly)

We can back-propagate to all the representations in datastore \mathcal{D}_{train} !

TRIME: Full index vs. in-batch index



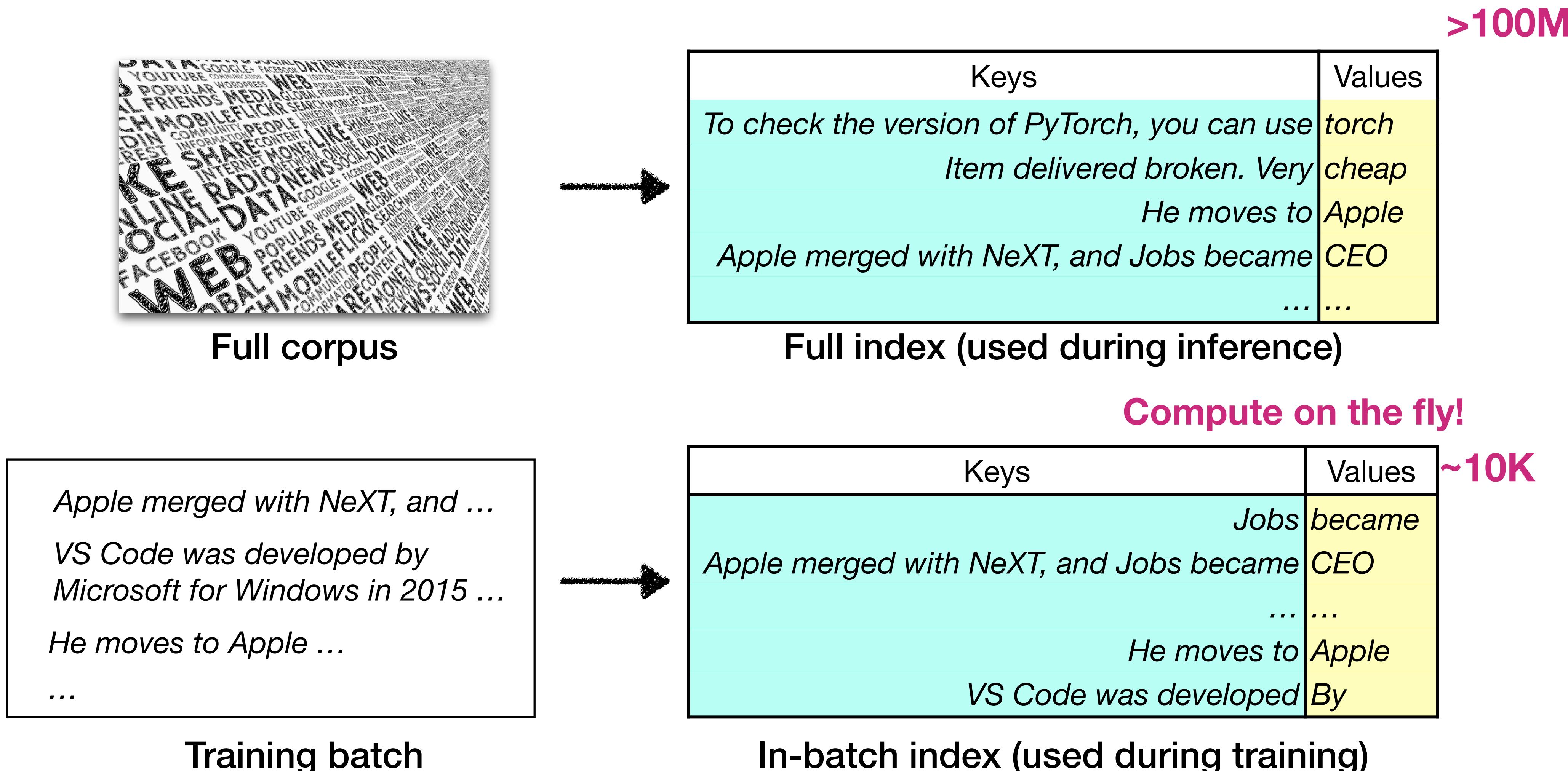
Full corpus



Keys	Values
<i>To check the version of PyTorch, you can use</i>	<i>torch</i>
<i>Item delivered broken. Very</i>	<i>cheap</i>
<i>He moves to</i>	<i>Apple</i>
<i>Apple merged with NeXT, and Jobs became</i>	<i>CEO</i>
<i>...</i>	<i>...</i>

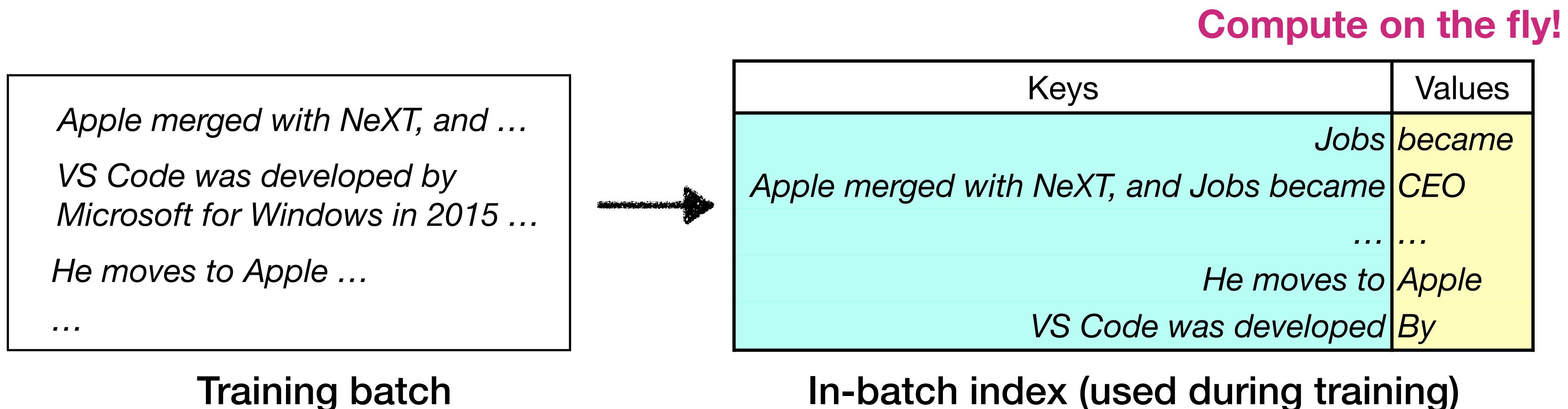
Full index (used during inference)

TRIME: Full index vs. in-batch index

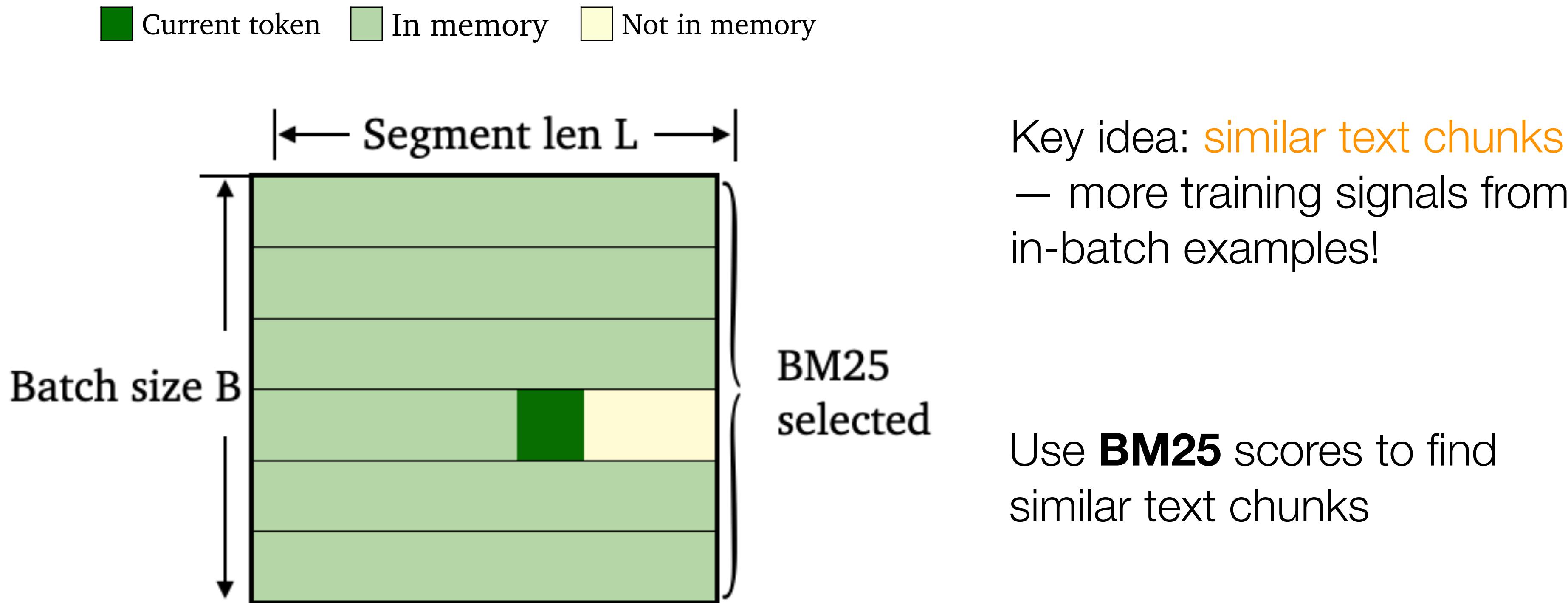


TRIME: Full index vs. in-batch index

How to batch training data —
so we can have good in-batch examples?

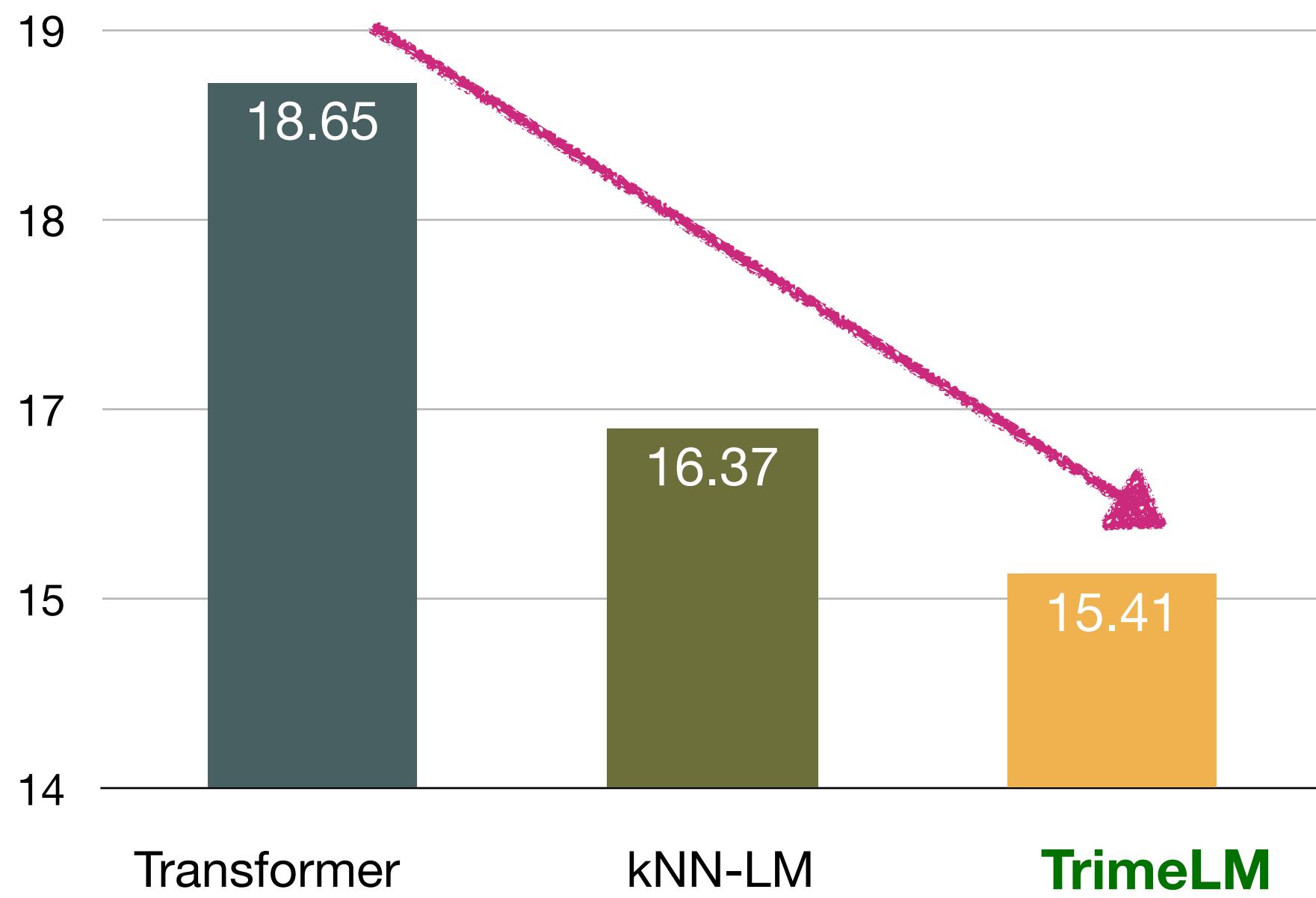


TRIME: Data batching strategy



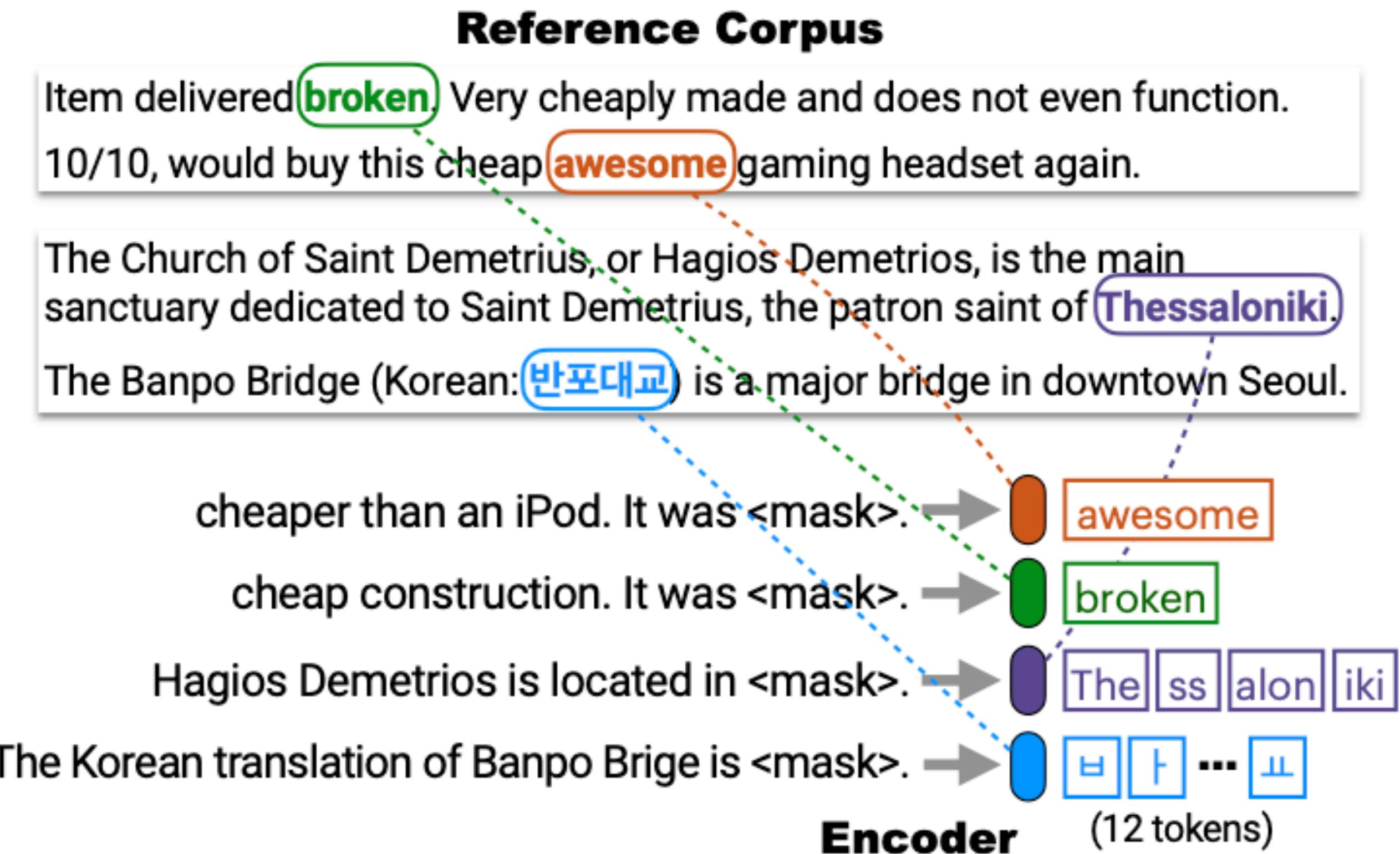
TRIME: Results

Perplexity: The lower the better

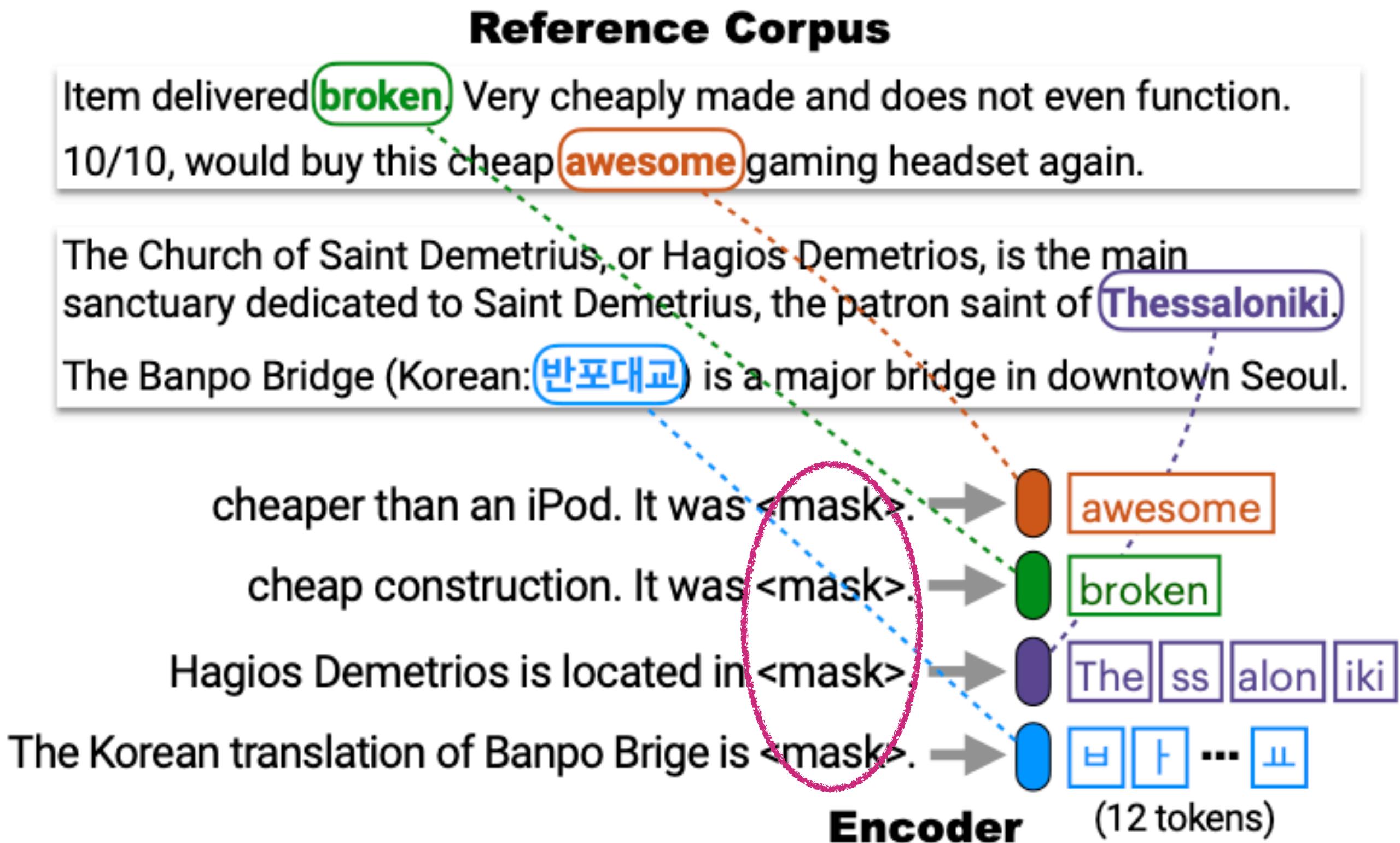


Perplexity on Wikitext-103

NPM: Nonparametric masked LMs (Min et al. 2023)

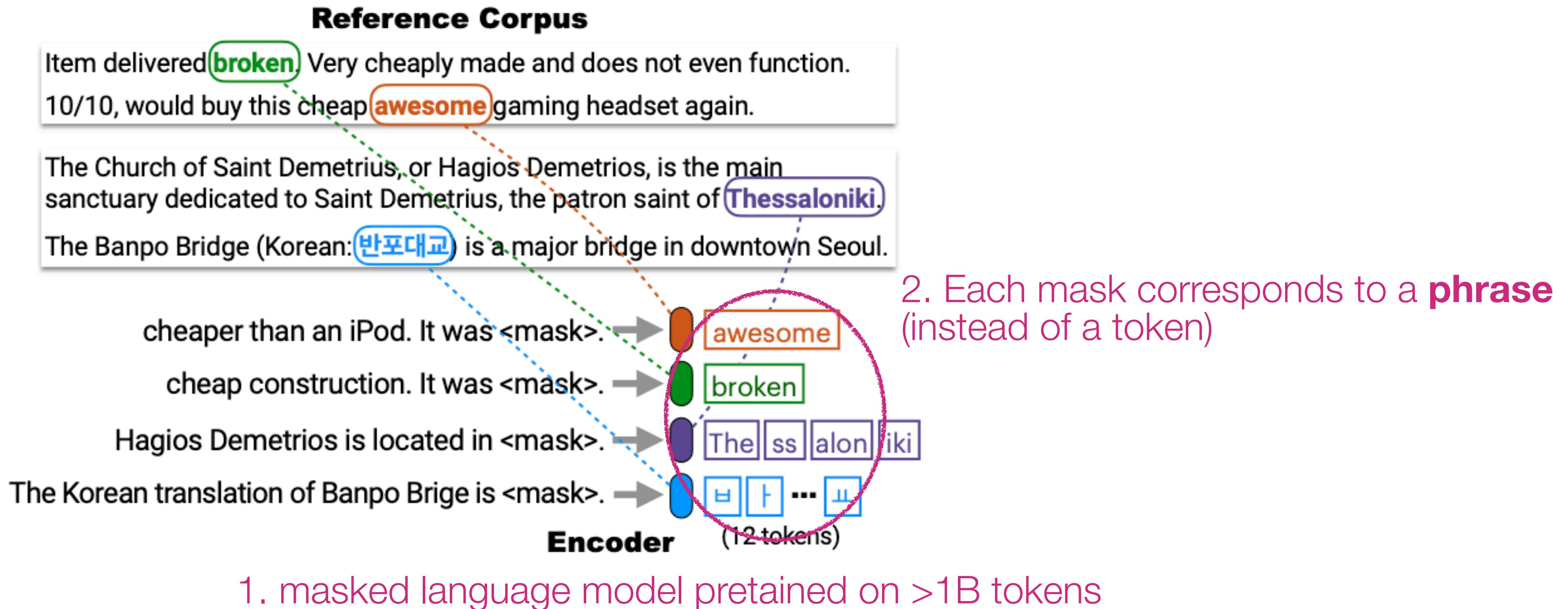


NPM: Nonparametric masked LMs (Min et al. 2023)

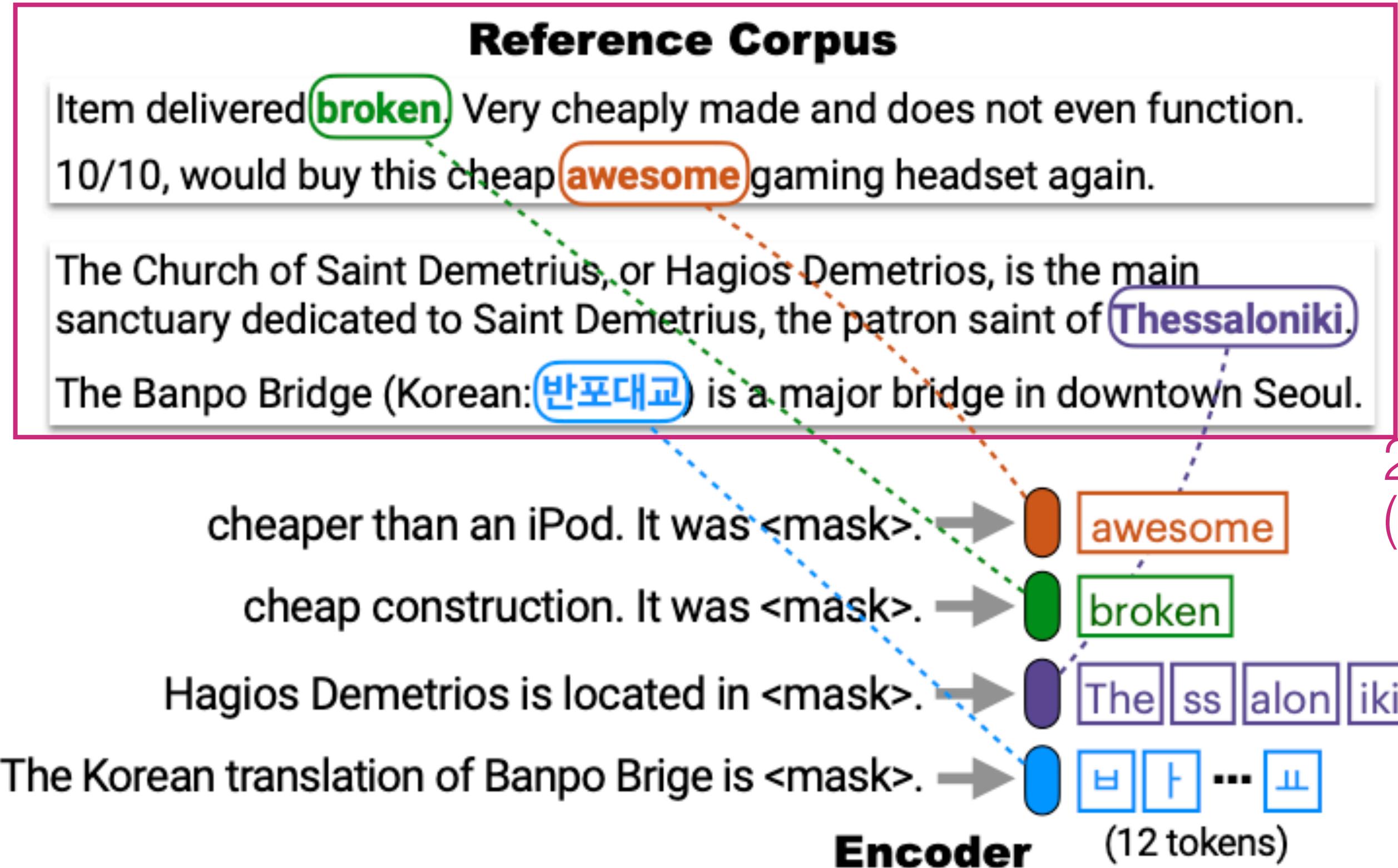


1. **masked** language model pretrained on >1B tokens

NPM: Nonparametric masked LMs (Min et al. 2023)



NPM: Nonparametric masked LMs (Min et al. 2023)



1. masked language model pertained on >1B tokens
2. Each mask corresponds to a phrase (instead of a token)
3. During inference, predictions are made **purely** according to retrieval results

NPM: Nonparametric masked LMs (Min et al. 2023)

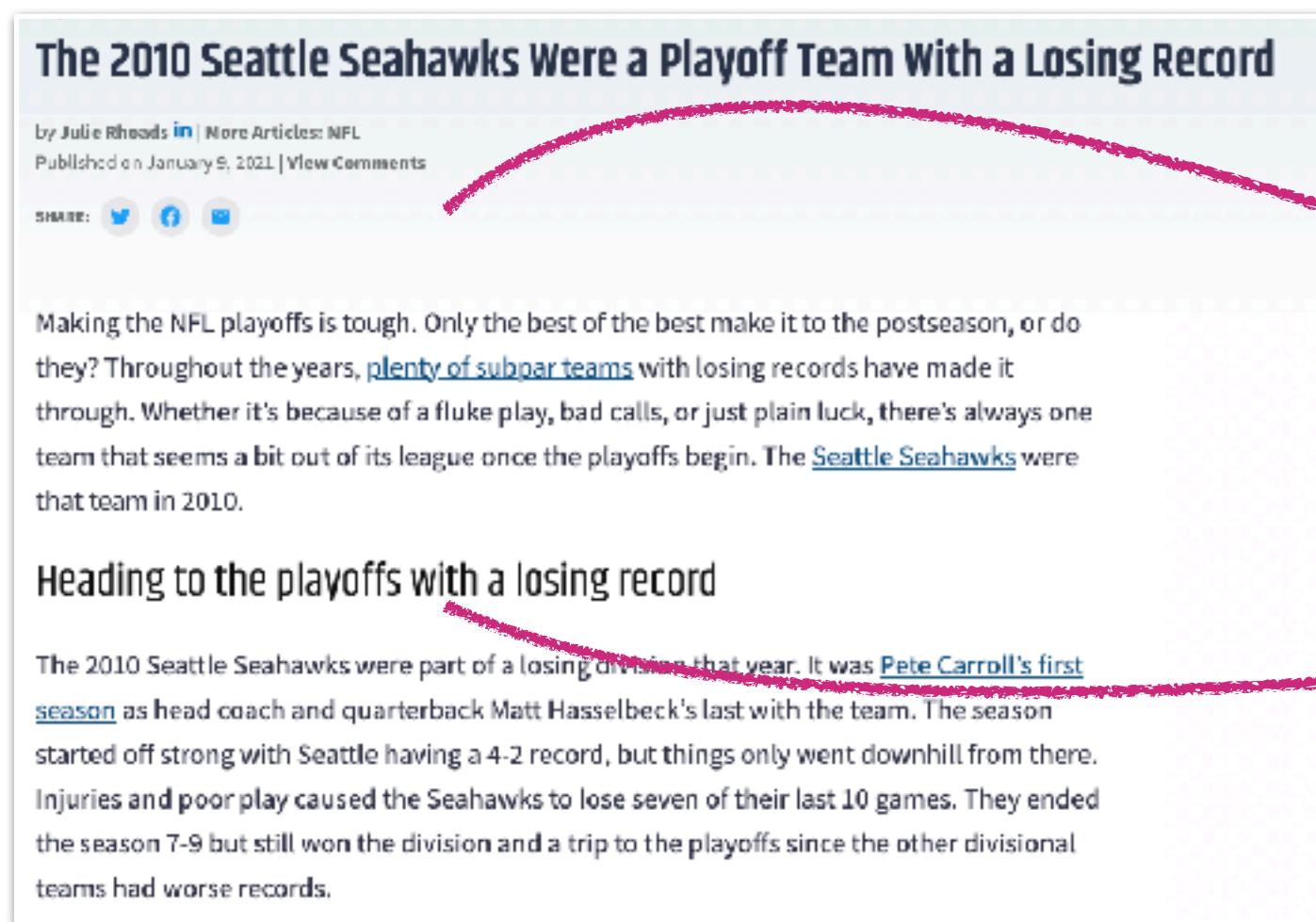
Key challenges

1. How to approximate the full retrieval index during training
2. How to get training signals (positive/negatives) from the index approximation

In-batch approximation with same-doc batching

NPM:Training

1. Sample sequences from the **same** document



In the 2010 NFL season, the Seattle Seahawks made history by making it into the playoffs despite having a 7-9 record.

... against the Seattle Seahawks as a member of (...) In the 2010 season, the Seahawks became the first team in NFL history to ...

For simplicity, we assume 2 sequences in a batch

NPM:Training

2. Identify co-occurring spans

The 2010 Seattle Seahawks Were a Playoff Team With a Losing Record

By Julie Rheads | More Articles: NFL
Published on January 9, 2021 | View Comments

SHARE:

Making the NFL playoffs is tough. Only the best of the best make it to the postseason, or do they? Throughout the years, [plenty of subpar teams](#) with losing records have made it through. Whether it's because of a fluke play, bad calls, or just plain luck, there's always one team that seems a bit out of its league once the playoffs begin. The [Seattle Seahawks](#) were that team in 2010.

Heading to the playoffs with a losing record

The 2010 Seattle Seahawks were part of a losing division that year. It was [Pete Carroll's first season](#) as head coach and quarterback Matt Hasselbeck's last with the team. The season started off strong with Seattle having a 4-2 record, but things only went downhill from there. Injuries and poor play caused the Seahawks to lose seven of their last 10 games. They ended the season 7-9 but still won the division and a trip to the playoffs since the other divisional teams had worse records.

In the 2010 NFL season, **the Seattle Seahawks** made history by making it into the playoffs despite having a 7-9 record.

... against **the Seattle Seahawks** as a member of (...) In the 2010 season, the Seahawks became the first team in NFL history to ...

For simplicity, we assume 2 sequences in a batch

NPM:Training

3. One is “positive” for the other

The 2010 Seattle Seahawks Were a Playoff Team With a Losing Record

By Julie Rhead [in](#) | More Articles: NFL
Published on January 9, 2021 | View Comments

SHARE:

Making the NFL playoffs is tough. Only the best of the best make it to the postseason, or do they? Throughout the years, [plenty of subpar teams](#) with losing records have made it through. Whether it's because of a fluke play, bad calls, or just plain luck, there's always one team that seems a bit out of its league once the playoffs begin. The [Seattle Seahawks](#) were that team in 2010.

Heading to the playoffs with a losing record

The 2010 Seattle Seahawks were part of a losing division that year. It was [Pete Carroll's first season](#) as head coach and quarterback Matt Hasselbeck's last with the team. The season started off strong with Seattle having a 4-2 record, but things only went downhill from there. Injuries and poor play caused the Seahawks to lose seven of their last 10 games. They ended the season 7-9 but still won the division and a trip to the playoffs since the other divisional teams had worse records.

In the 2010 NFL season, _____ made history by making it into the playoffs despite having a 7-9 record.

positive

... against **the Seattle Seahawks** as a member of (...) In the 2010 season, the Seahawks became the first team in NFL history to ...

For simplicity, we assume 2 sequences in a batch

NPM:Training

4. The others are “negatives”

The 2010 Seattle Seahawks Were a Playoff Team With a Losing Record

By Julie Rhead [in](#) | More Articles: NFL
Published on January 9, 2021 | View Comments

SHARE:

Making the NFL playoffs is tough. Only the best of the best make it to the postseason, or do they? Throughout the years, [plenty of subpar teams](#) with losing records have made it through. Whether it's because of a fluke play, bad calls, or just plain luck, there's always one team that seems a bit out of its league once the playoffs begin. The [Seattle Seahawks](#) were that team in 2010.

Heading to the playoffs with a losing record

The 2010 Seattle Seahawks were part of a losing division that year. It was [Pete Carroll's first season](#) as head coach and quarterback Matt Hasselbeck's last with the team. The season started off strong with Seattle having a 4-2 record, but things only went downhill from there. Injuries and poor play caused the Seahawks to lose seven of their last 10 games. They ended the season 7-9 but still won the division and a trip to the playoffs since the other divisional teams had worse records.

In the 2010 NFL season, _____ made history by making it into the playoffs despite having a 7-9 record.

positive

... against **the Seattle Seahawks** as a member of (...) In the 2010 season, the Seahawks became the first team in NFL history to ...

negatives

For simplicity, we assume 2 sequences in a batch

Beyond lexical clues?

In TRIME and NPM, retrieval models are trained to use *lexical* information

Positives: **co-occurring** tokens/spans

Can we do more than that?

RPT: Retrieval-pretrained transformer

(Rubin and Berant 2023)

Reference score $P(\text{"Apple"} \mid \text{"Jobs become CEO of"}, \underline{\text{"NeXT merged with ..."}})$

Reference chunk

RPT: Retrieval-pretrained transformer

(Rubin and Berant 2023)

Reference score $P(\text{"Apple"} \mid \text{"Jobs become CEO of"}, \text{"NeXT merged with ..."})$

Reference chunk

$P(\text{"Apple"} \mid \text{"Jobs become CEO of"}, \text{"He joined his former ..."}) > \text{Reference score}$

Positive chunks

RPT: Retrieval-pretrained transformer

(Rubin and Berant 2023)

Reference score $P(\text{"Apple"} \mid \text{"Jobs become CEO of"}, \text{"NeXT merged with ..."})$

Reference chunk

$P(\text{"Apple"} \mid \text{"Jobs become CEO of"}, \text{"He joined his former ..."}) > \text{Reference score}$

Positive chunks

$P(\text{"Apple"} \mid \text{"Jobs become CEO of"}, \text{"Jobs was raised ..."}) < \text{Reference score}$

Negative chunks

Joint training

-  End-to-end trained – each component is optimized
-  Good performance
-  Training is more complicated
(async update, overhead, data batching, etc)
-  Train-test discrepancy still remains

Summary

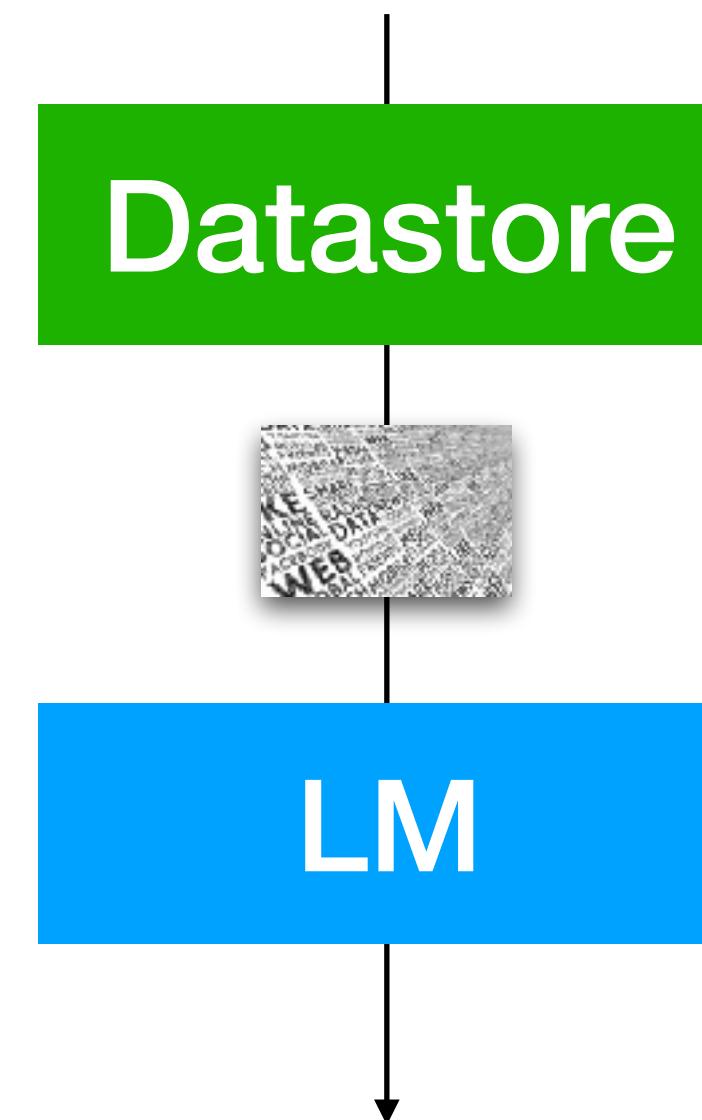
Training method	+	-
Independent training (Ram et al 2023; Khandelwal et al 2020)		
Sequential training (Borgeaud et al 2021; Shi et al 2023)	<ul style="list-style-type: none">* Easy to implement: off-the-shelf models* Easy to improve: sub-module can be separately improved	<ul style="list-style-type: none">* Models are not end-to-end trained — suboptimal performance
Joint training: async update (Guu et al 2020; Izacard et al 2022)	<ul style="list-style-type: none">* End-to-end trained — very good performance!	<ul style="list-style-type: none">* Training may be complicated (overhead, batching methods, etc)* Train-test discrepancy still remains
Joint training: in-batch approx (Zhong et al 2022; Min et al 2023; Rubin and Berant 2023)		

How do retrieval-based language models perform on downstream tasks? → **Section 5!**

Section 5: Applications

Downstream adaptation of retrieval-based LMs

The capital city of Ontario is __



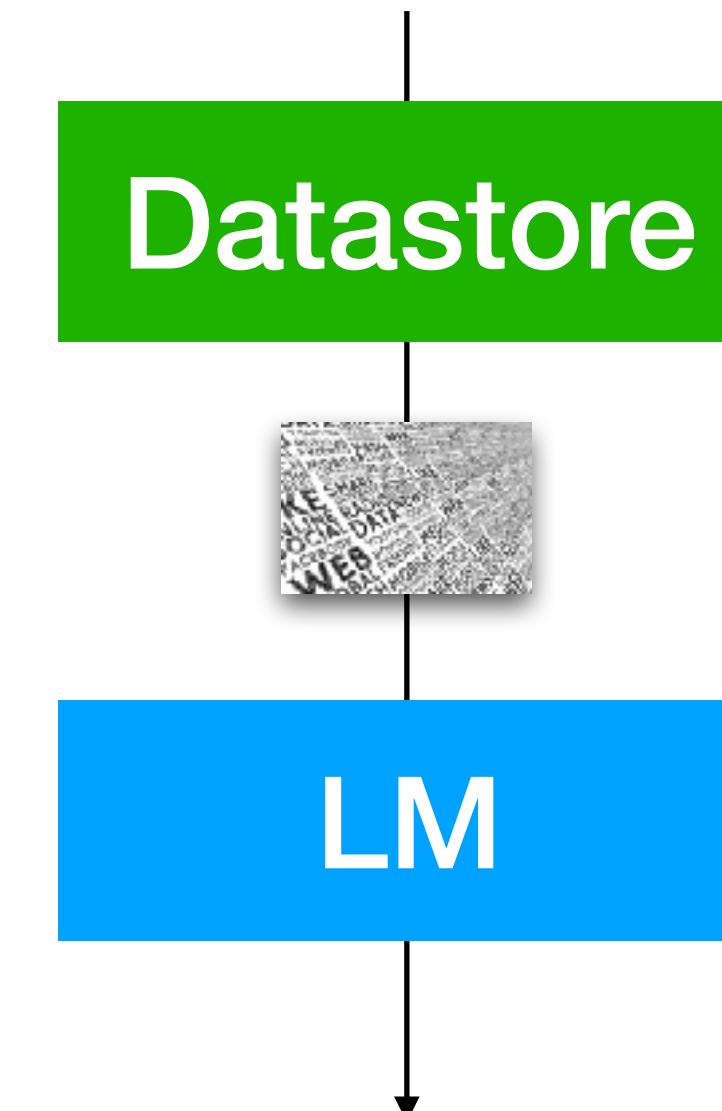
Toronto, which is known for ...

Downstream adaptation of retrieval-based LMs

What are the **tasks**?

Open-domain QA

What is the capital of Ontario?



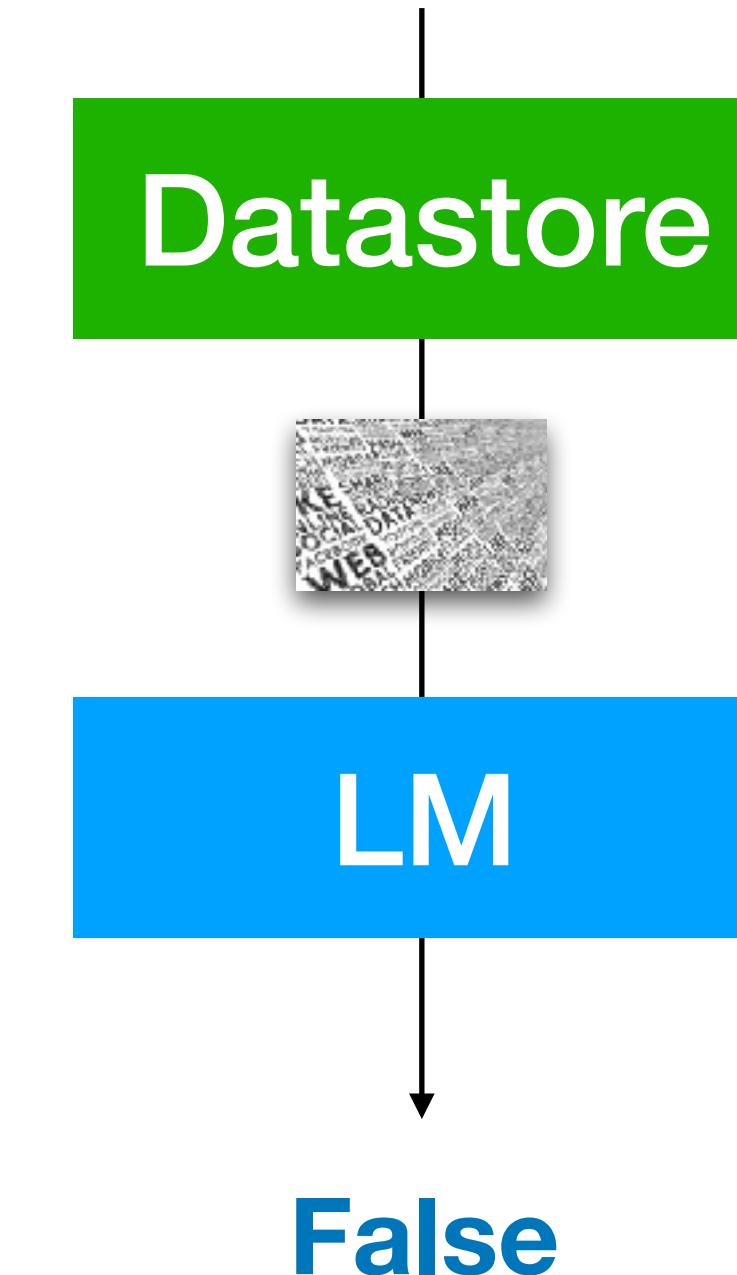
Toronto

Downstream adaptation of retrieval-based LMs

What are the **tasks**?

Fact verification

Ottawa is the Ontario state capital.



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

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)

Summarization

FLARE (Jiang et al, 2023)

Machine translation

kNN-MT (Khandelwal et al., 2020)
TRIME-MT (Zhong et al., 2022)

Code & proof generation

DocPrompting (Zhou et al., 2023)
Natural Prover
(Welleck et al., 2022)

NLI

kNN-Prompt (Shi et al., 2022)
NPM (Min et al., 2023)

Sentiment analysis

kNN-Prompt (Shi et al., 2022)
NPM (Min et al., 2023)

Commonsense reasoning

Raco (Yu et al, 2022)

More general NLP tasks

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)

Summarization

FLARE (Jiang et al, 2023)

Machine translation

kNN-MT (Khandelwal et al., 2020)
TRIME-MT (Zhong et al., 2022)

Code & proof generation

DocPrompting (Zhou et al., 2023)
Natural Prover
(Welleck et al., 2022)

NLI

kNN-Prompt (Shi et al., 2022)
NPM (Min et al., 2023)

Sentiment analysis

kNN-Prompt (Shi et al., 2022)
NPM (Min et al., 2023)

Commonsense reasoning

Raco (Yu et al, 2022)

More generations

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)

Summarization

FLARE (Jiang et al, 2023)

Machine translation

kNN-MT (Khandelwal et al., 2020)
TRIME-MT (Zhong et al., 2022)

Code & proof generation

DocPrompting (Zhou et al., 2023)
Natural Prover
(Welleck et al., 2022)

NLI

kNN-Prompt (Shi et al., 2022)
NPM (Min et al., 2023)

Sentiment analysis

kNN-Prompt (Shi et al., 2022)
NPM (Min et al., 2023)

Commonsense reasoning

Raco (Yu et al, 2022)

More classifications

Two key questions for downstream adaptations

How can we adapt a retrieval-based LM for a task?

When should we use a retrieval-based LM?

How to adapt a retrieval-based LM for a task

What are the **tasks**?

- Open-domain QA
- Other knowledge-intensive tasks
- Sentiment analysis
- Code generation

...

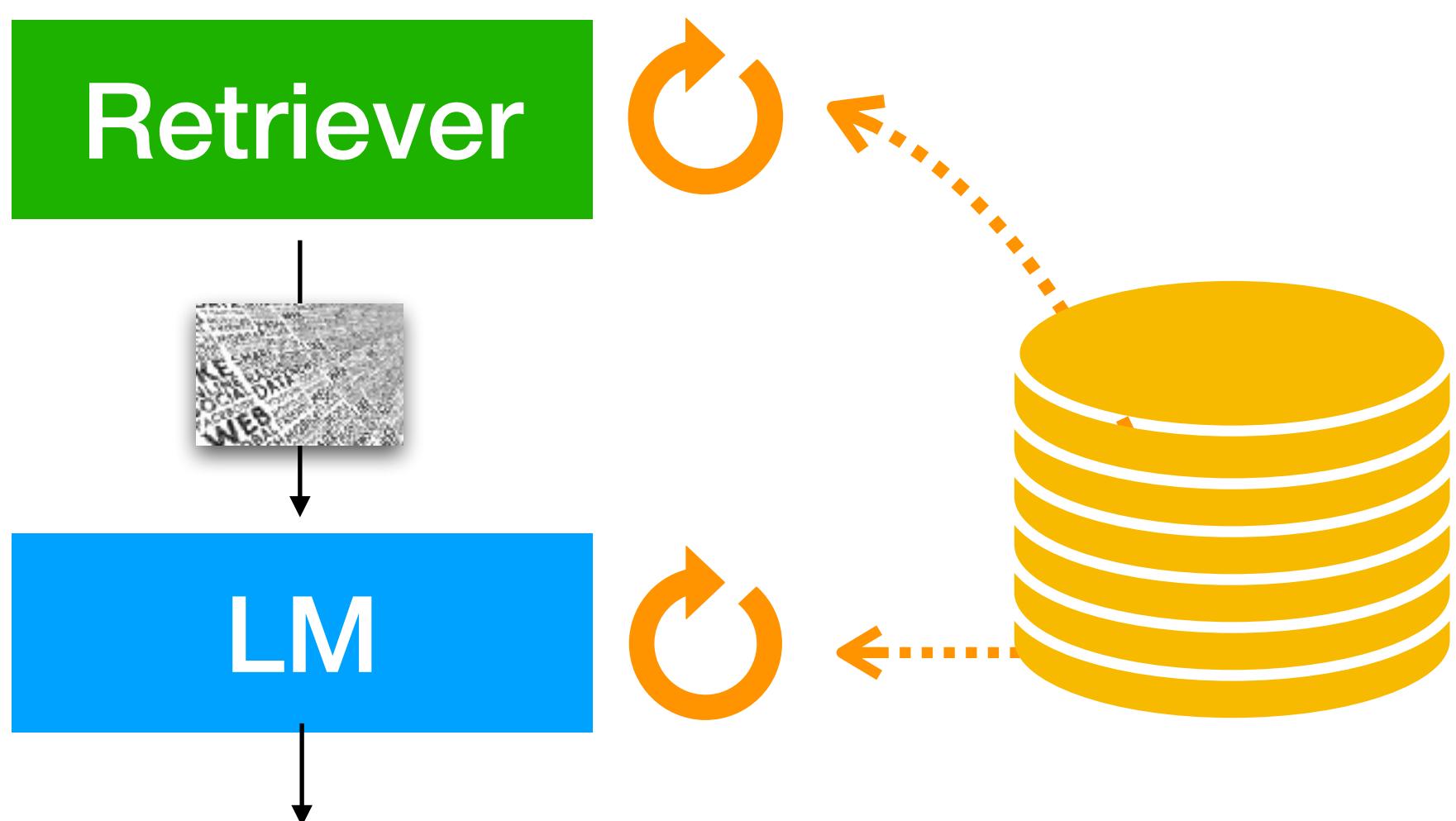
How to **adapt**?

- Fine-tuning
- Reinforcement learning
- Prompting

How to adapt a retrieval-based LM for a task

Fine-tuning (+RL)

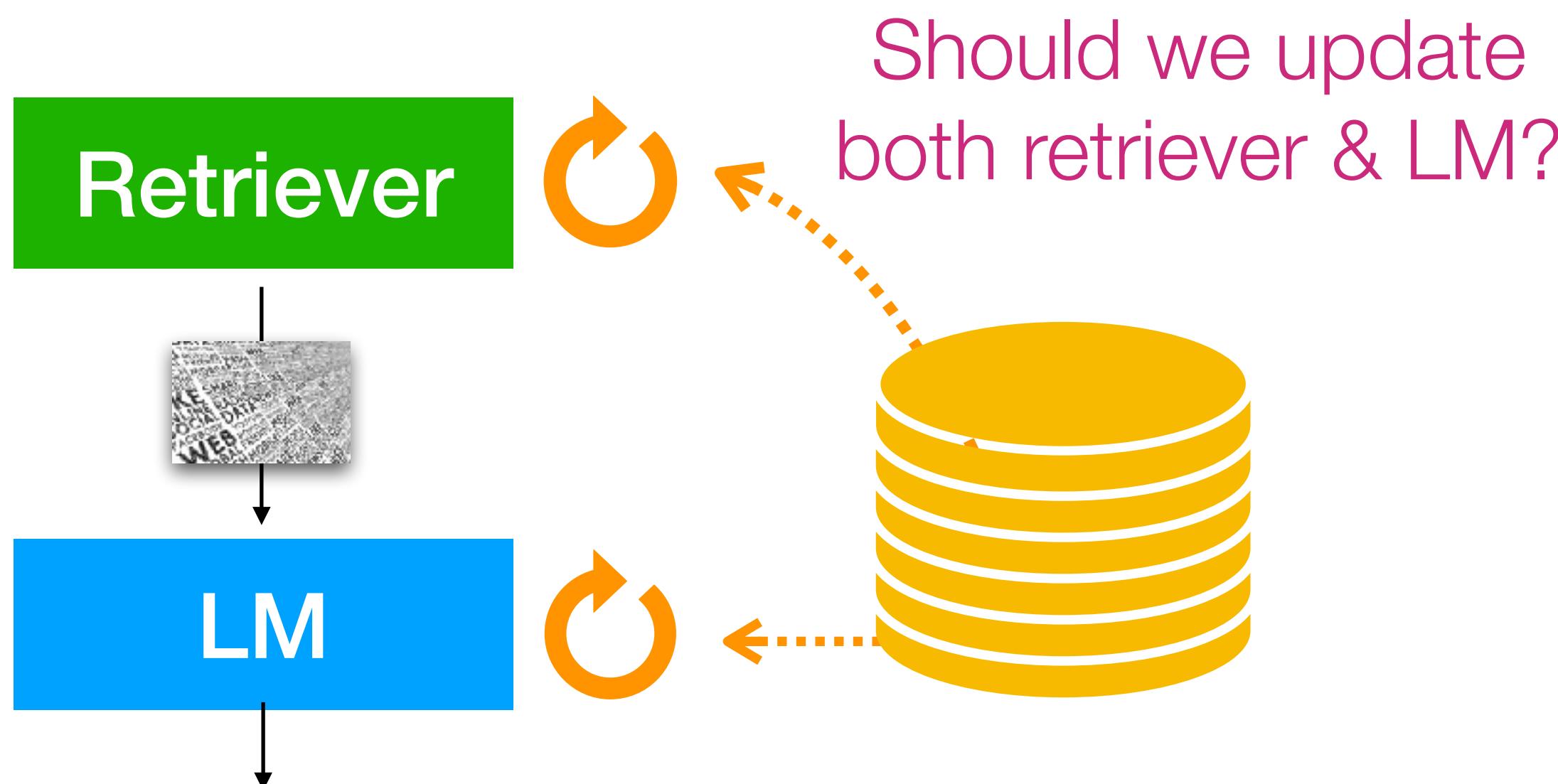
Training LM and / or retriever
on task-data & data store



How to adapt a retrieval-based LM for a task

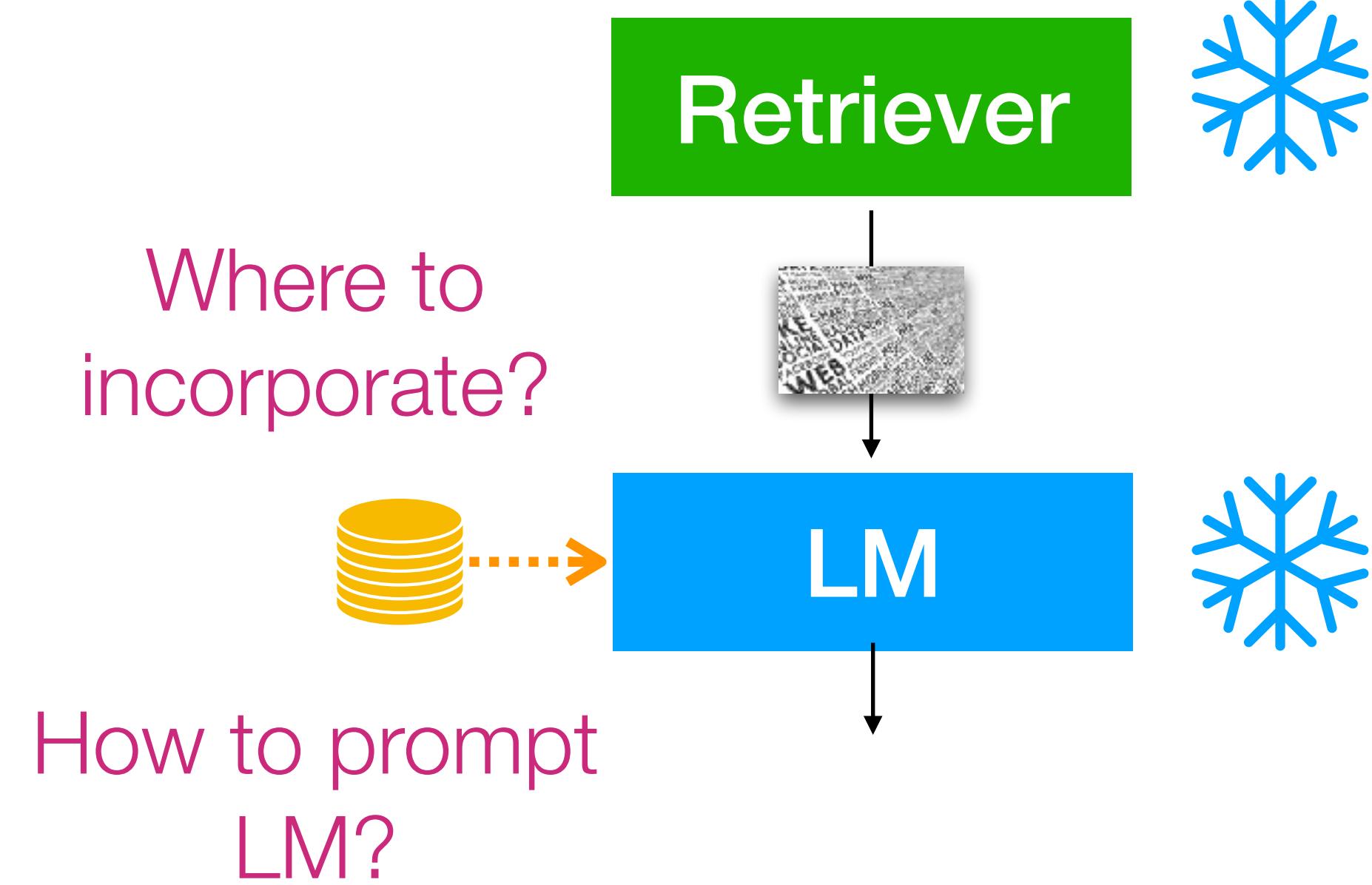
Fine-tuning (+RL)

Training LM and / or retriever
on task-data & data store



Prompting

Prompt a frozen LM with
retrieved knowledge



How to adapt a retrieval-based LM for a task

What are the **tasks**?

- Open-domain QA
- Other knowledge-intensive tasks
- Sentiment analysis
- Code generation
- ...

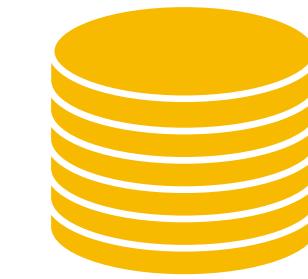
How to **adapt**?

- Fine-tuning
- Reinforcement learning
- Prompting

What is **data store**?



Wikipedia



Training data



Code documentation

When to use a retrieval-based LM

Long-tail

knowledge
update

Verifiability

Parameter-
efficiency

Effectiveness of retrieval-based LMs

Long-tail

knowledge
update

Verifiability

Parameter-
efficiency

Q: Is Toronto really
cold during winter?



Effectiveness of retrieval-based LMs

Long-tail

knowledge
update

Verifiability

Parameter-
efficiency

Q: Where is Toronto
Zoo located?



1361A Old Finch Avenue,
in Scarborough, Ontario



Effectiveness of retrieval-based LMs

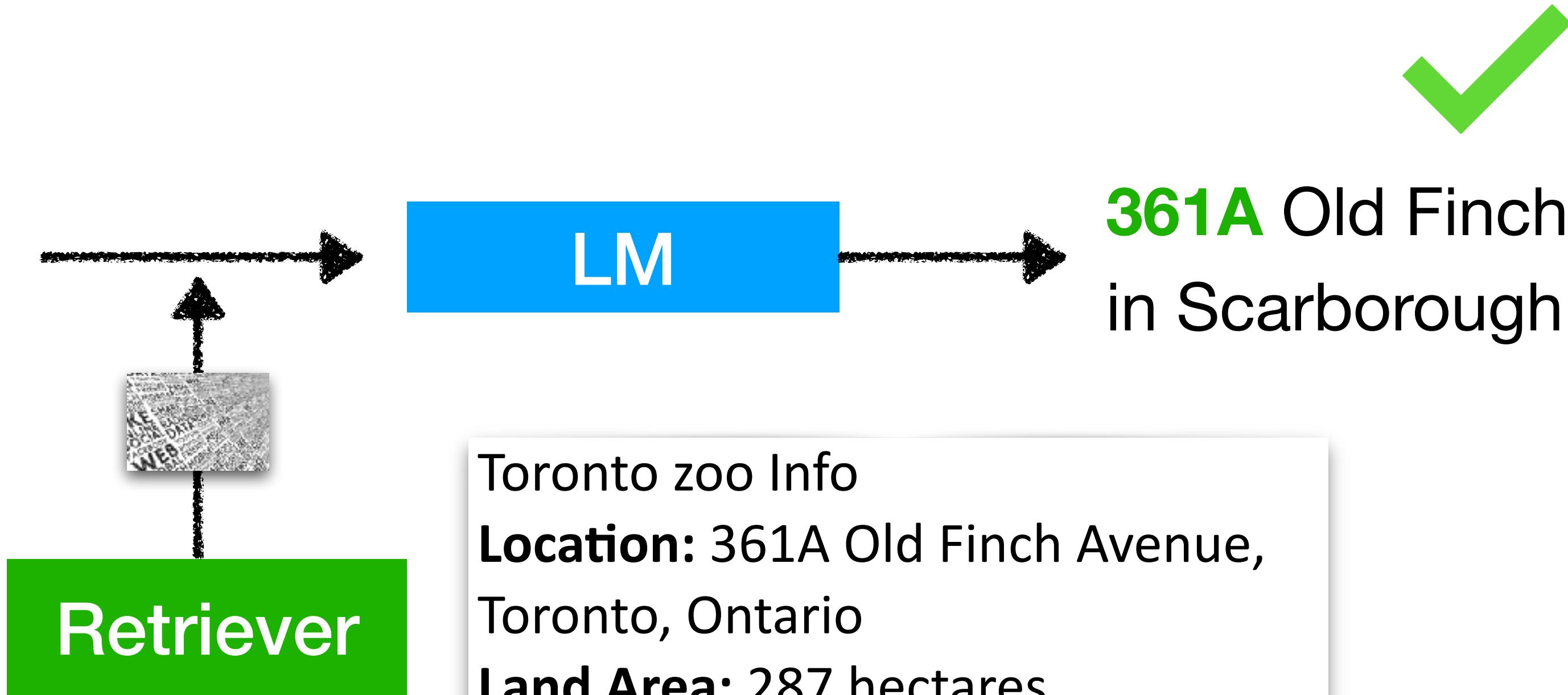
Long-tail

knowledge update

Verifiability

Parameter-efficiency

Q: Where is Toronto Zoo located?



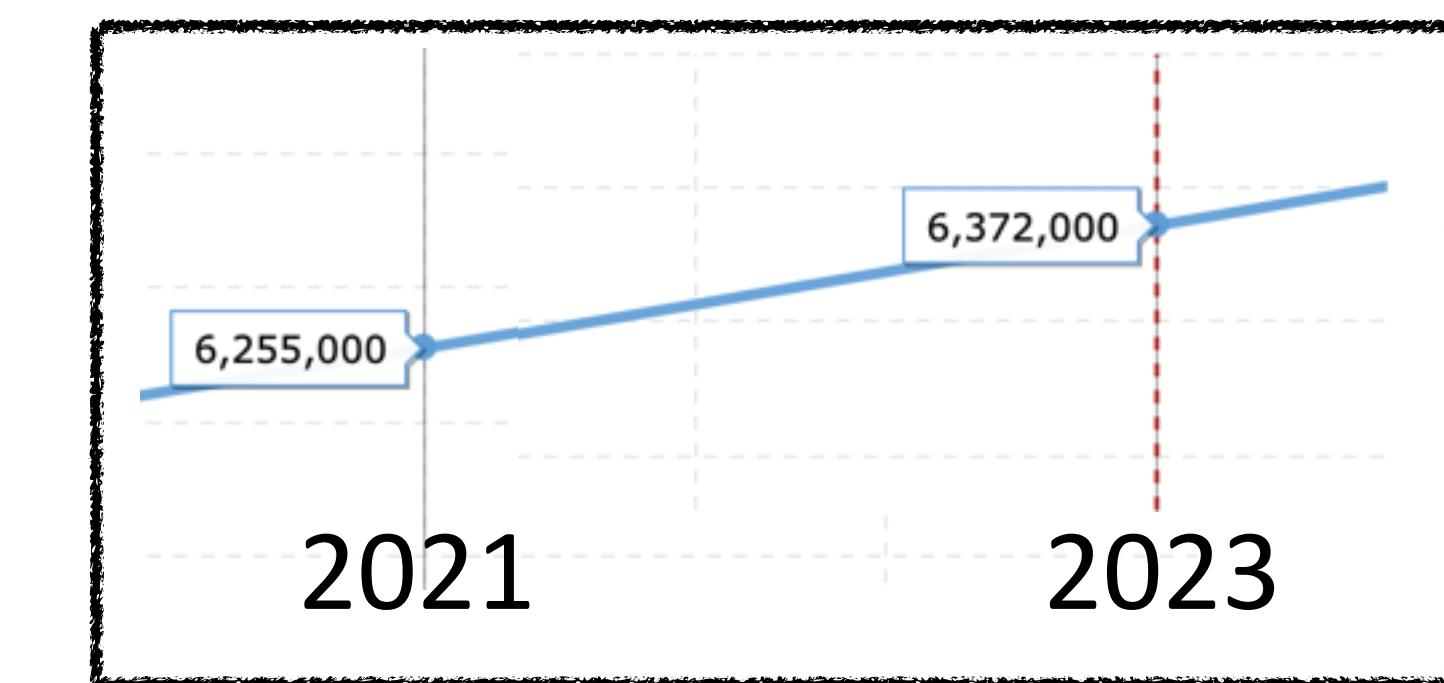
Effectiveness of retrieval-based LMs



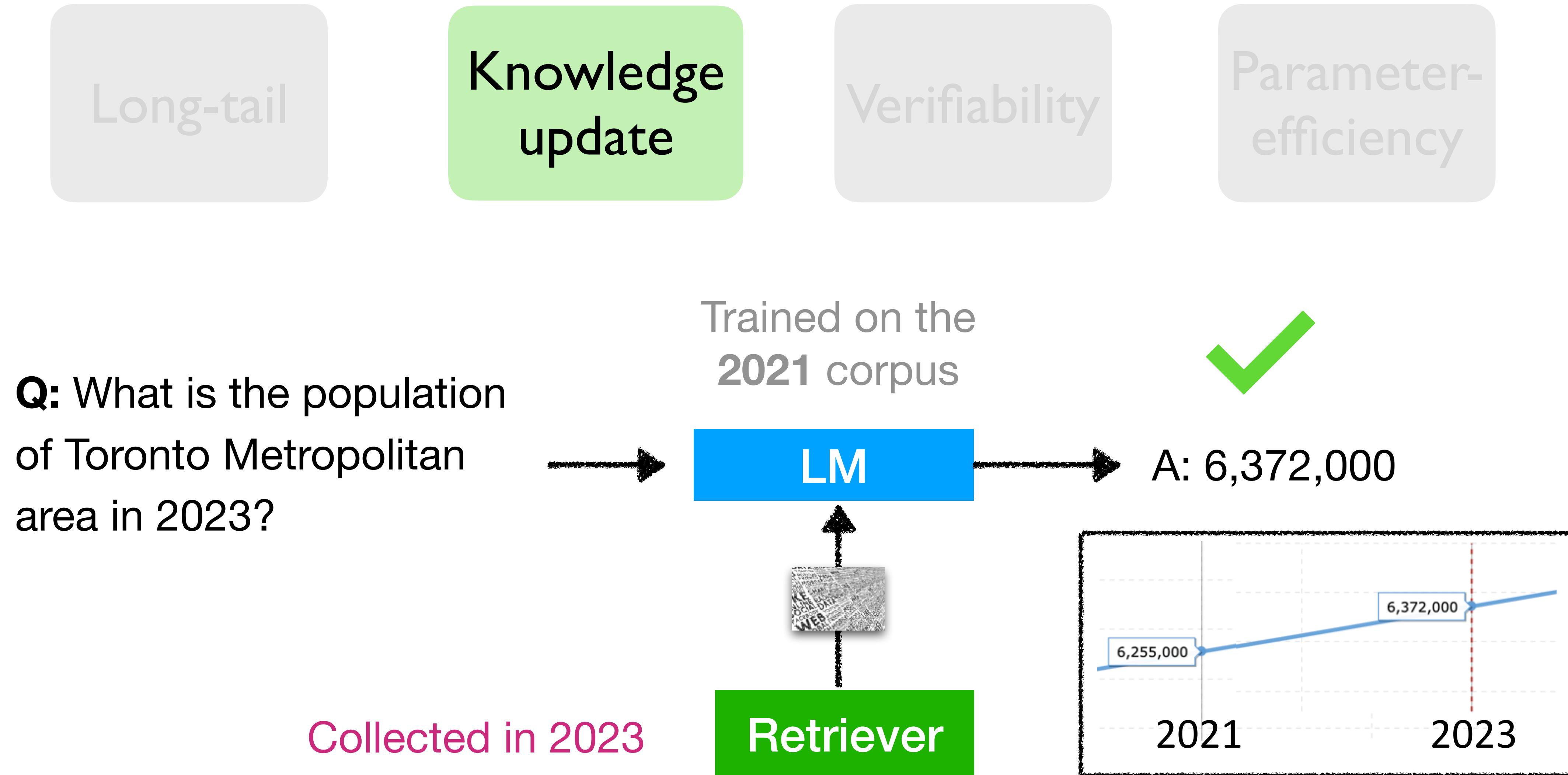
Q: What is the population
of Toronto Metropolitan
area in 2023?



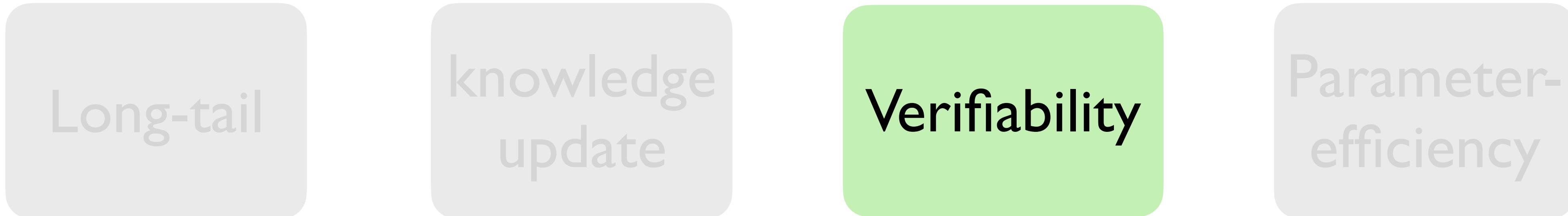
Trained on the
2021 corpus



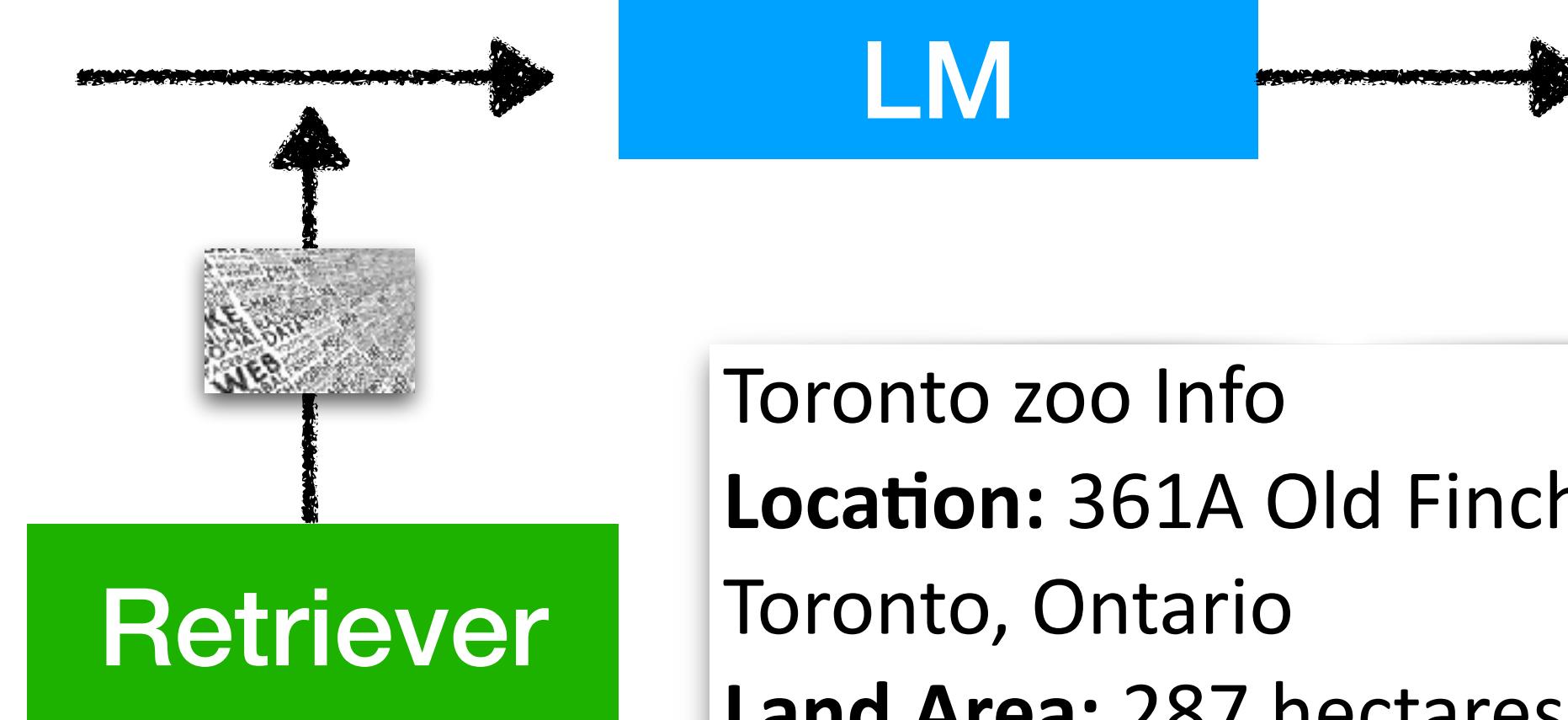
Effectiveness of retrieval-based LMs



Effectiveness of retrieval-based LMs



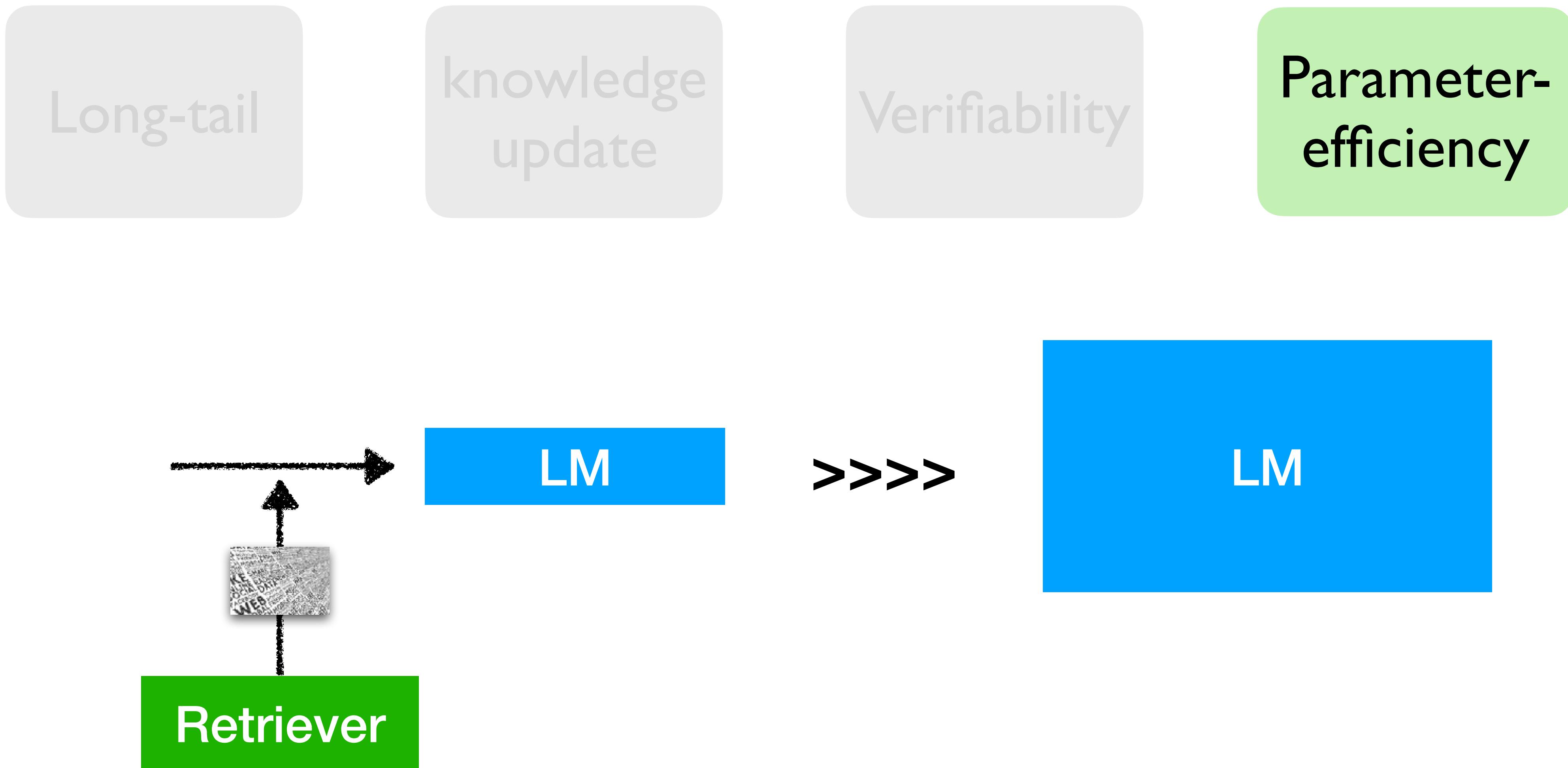
Q: Where is Toronto Zoo located?



Toronto zoo Info
Location: 361A Old Finch Avenue, Toronto, Ontario
Land Area: 287 hectares



Effectiveness of retrieval-based LMs



Two key questions for downstream adaptations

How can we adapt a retrieval-based LM for a task?

When should we use a retrieval-based LM?

Downstream adaptation of retrieval-based LMs

What are the **tasks**?

- Open-domain QA
- Other knowledge-intensive tasks
- General NLU
- Language Modeling & other generation tasks

How to **adapt**?

- **Fine-tuning**
- Reinforcement learning
- Prompting

What is **data store**?

- Unlabeled Wikipedia / CC
- Web (Google / Bing Search Results)
- Training data

Adapting retrieval-based LMs for tasks

Fine-tuning

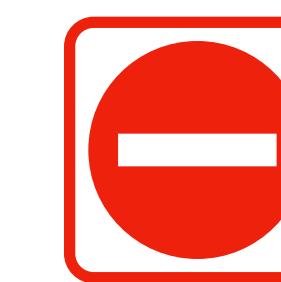
Training LM and / or retriever
on task-data & data store



Adapting retrieval-based LMs for tasks

Fine-tuning

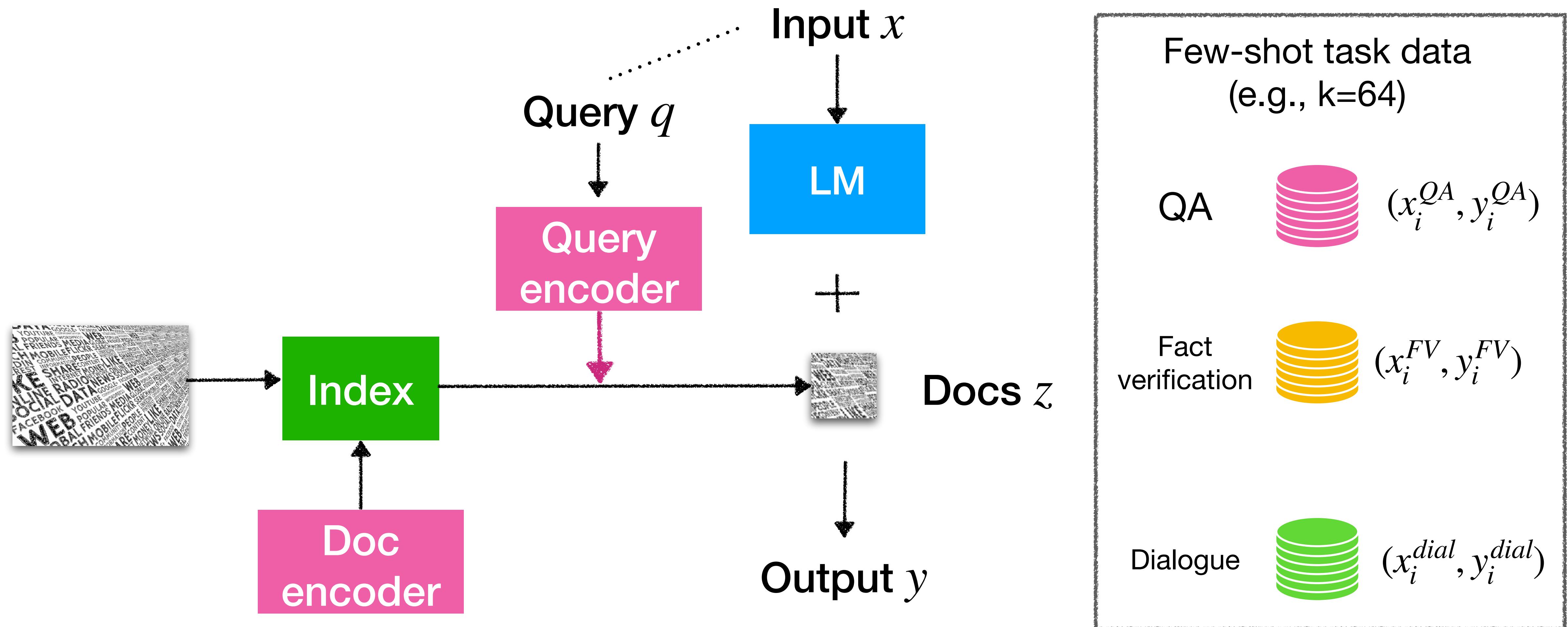
Training LM and / or retriever
on task-data & data store



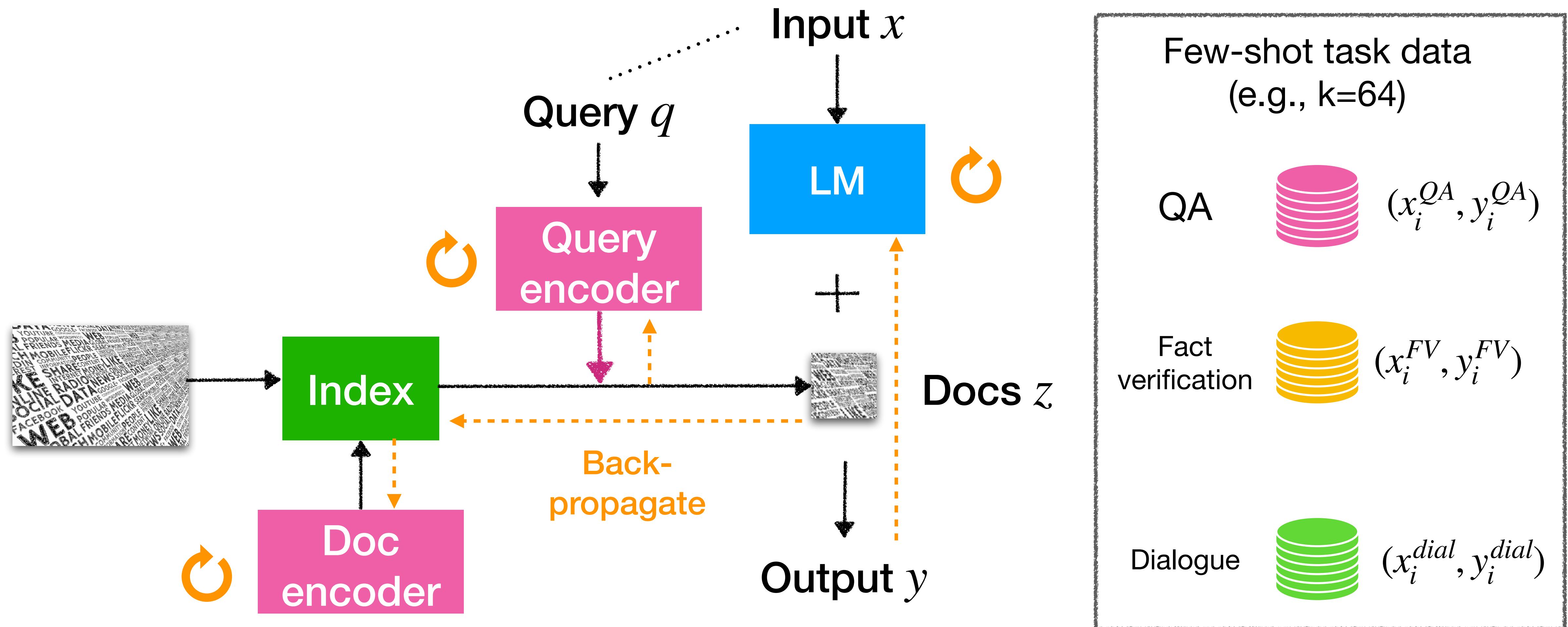
Costs of retrieval-based LM
training (Section 4)

Independent training (DPR)
Asynchronous updates (REALM)
...

ATLAS (Izacard et al., 2022; Section 4)

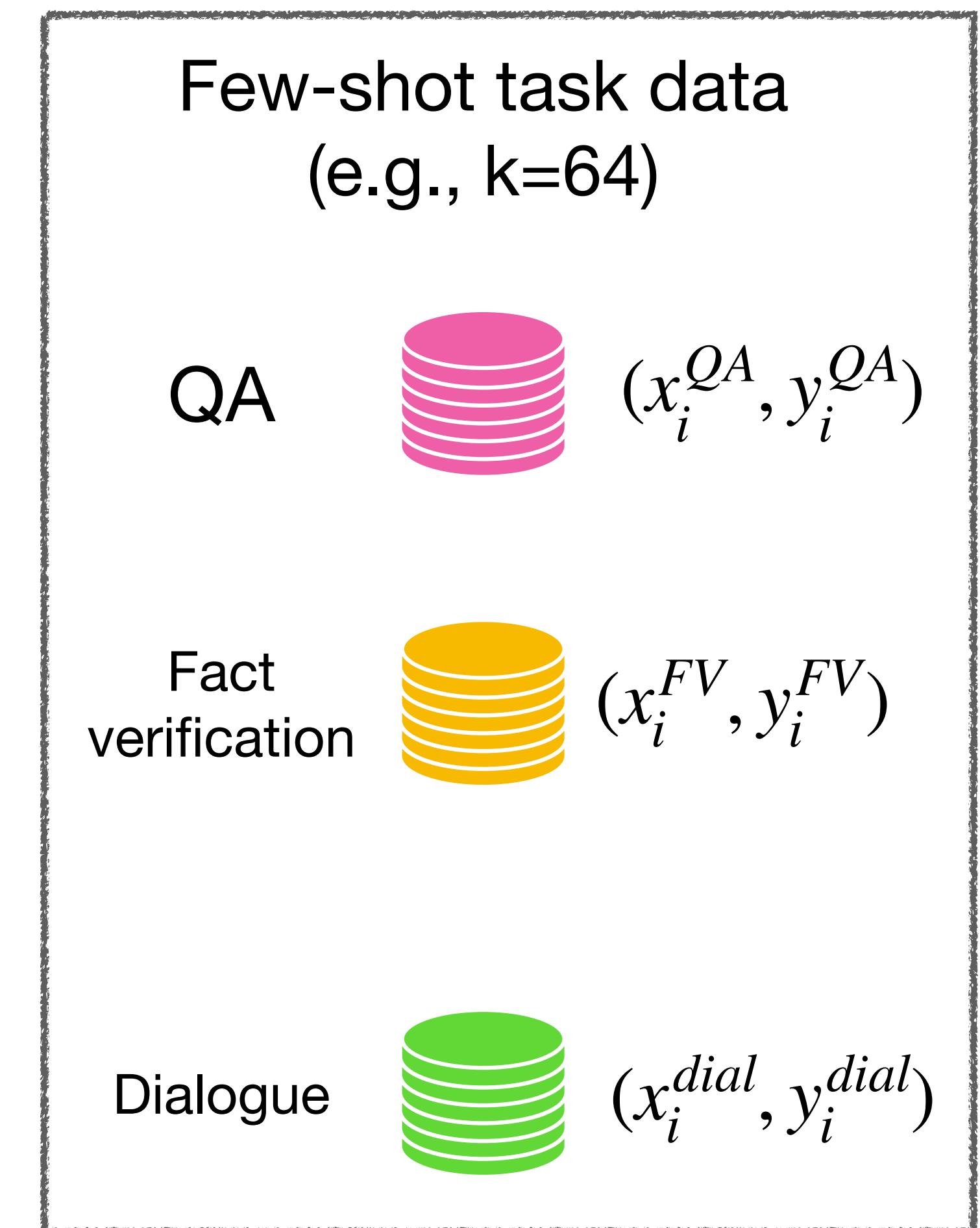
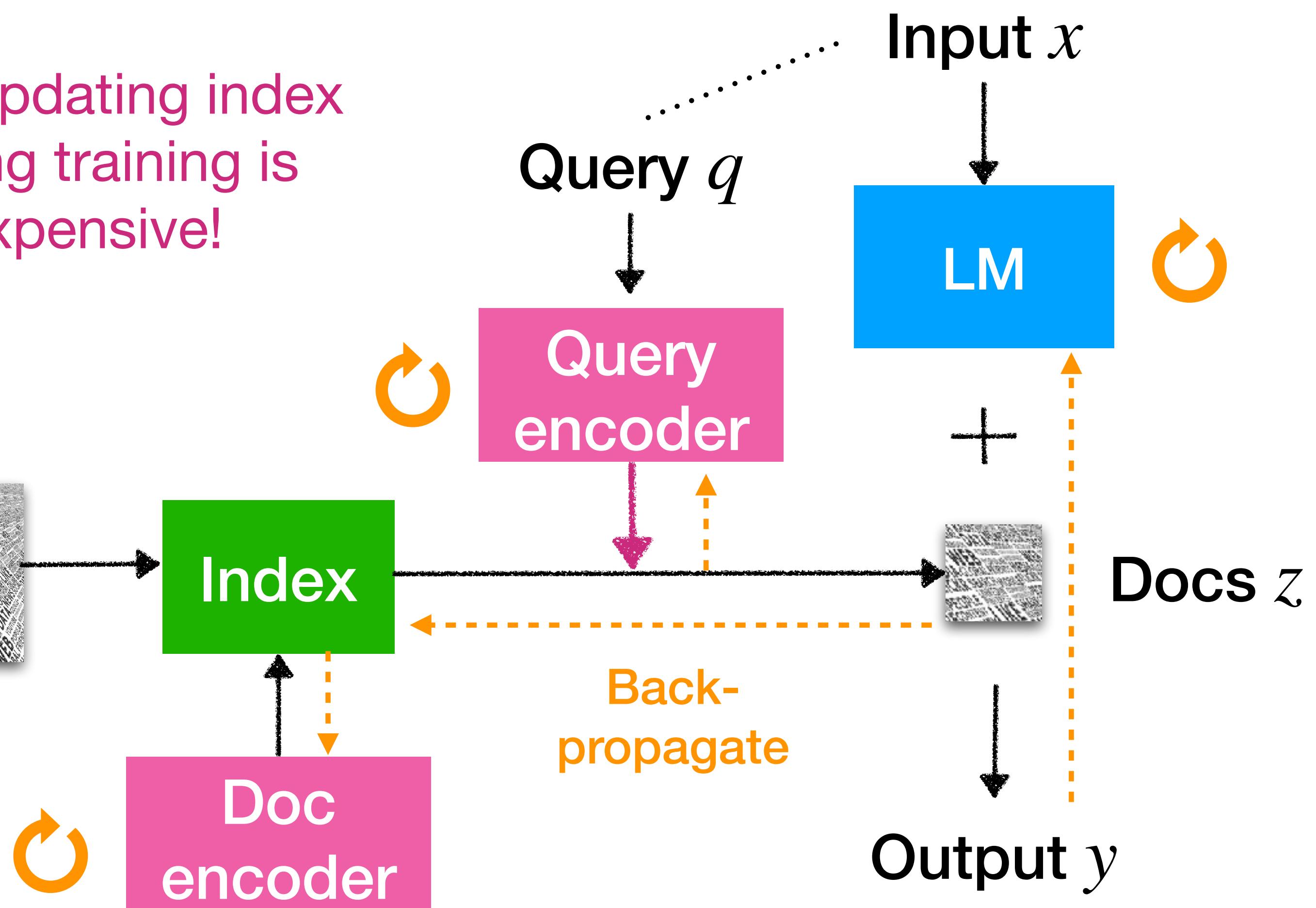


ATLAS (Izacard et al., 2022; Section 4)

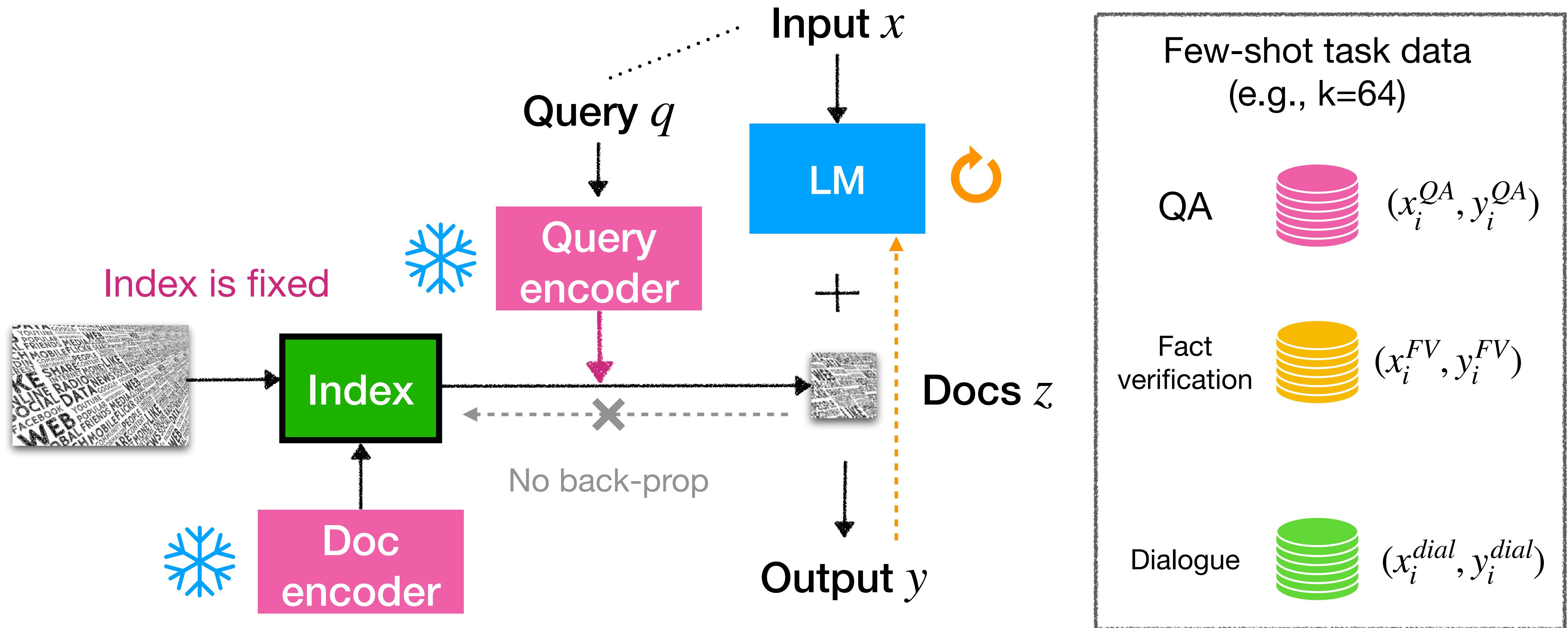


ATLAS (Izacard et al., 2022; Section 4)

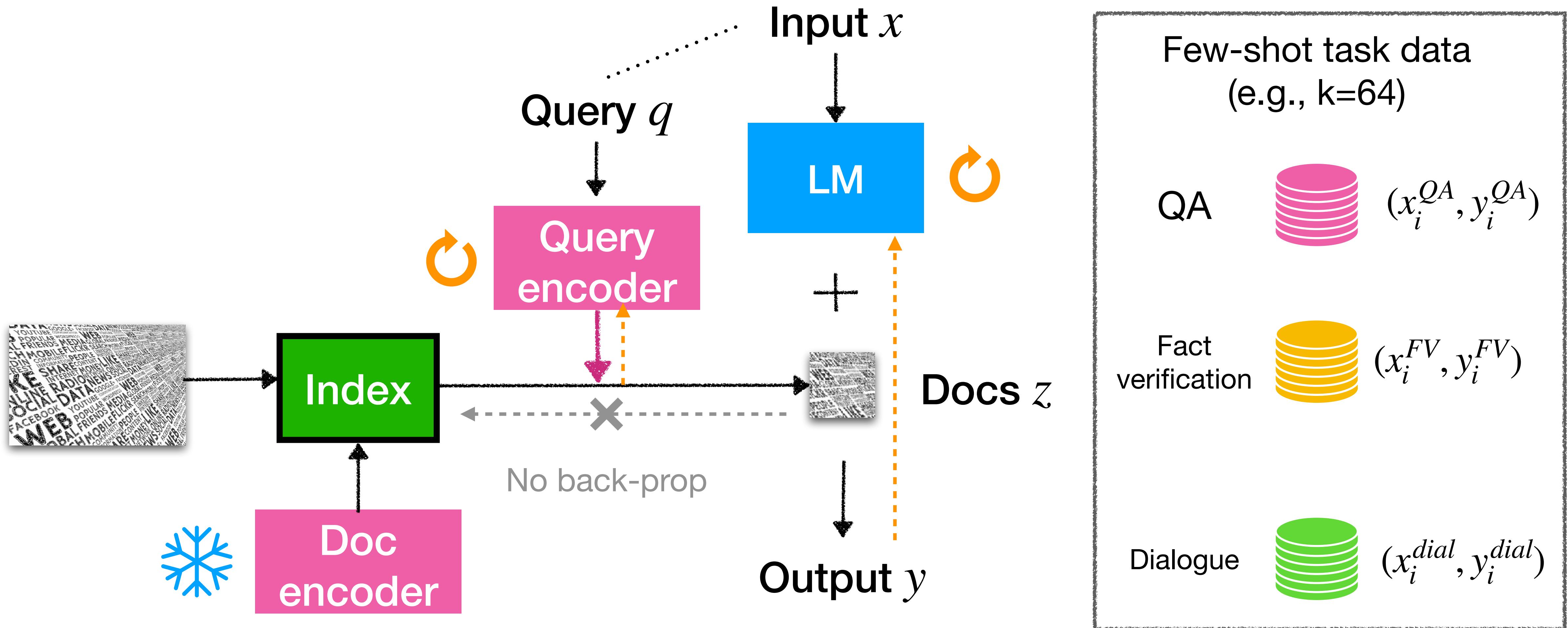
Fully updating index during training is expensive!



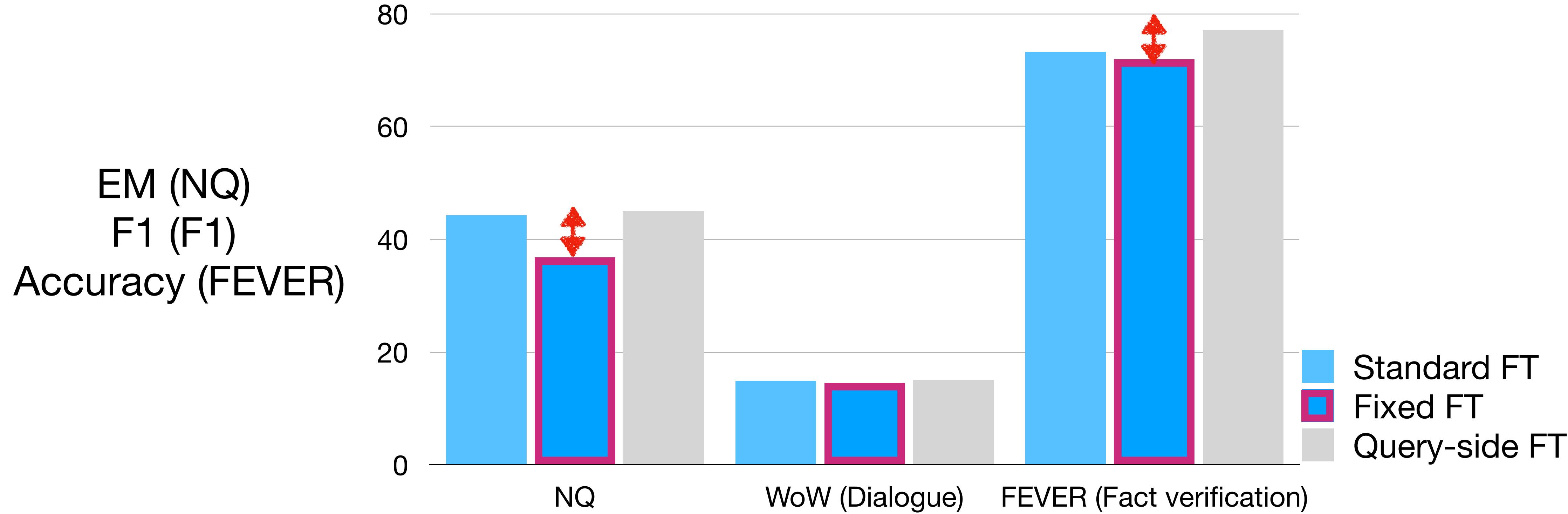
ATLAS: Fixed retrieval with fine-tuned LM



ATLAS: Query-side fine-tuning

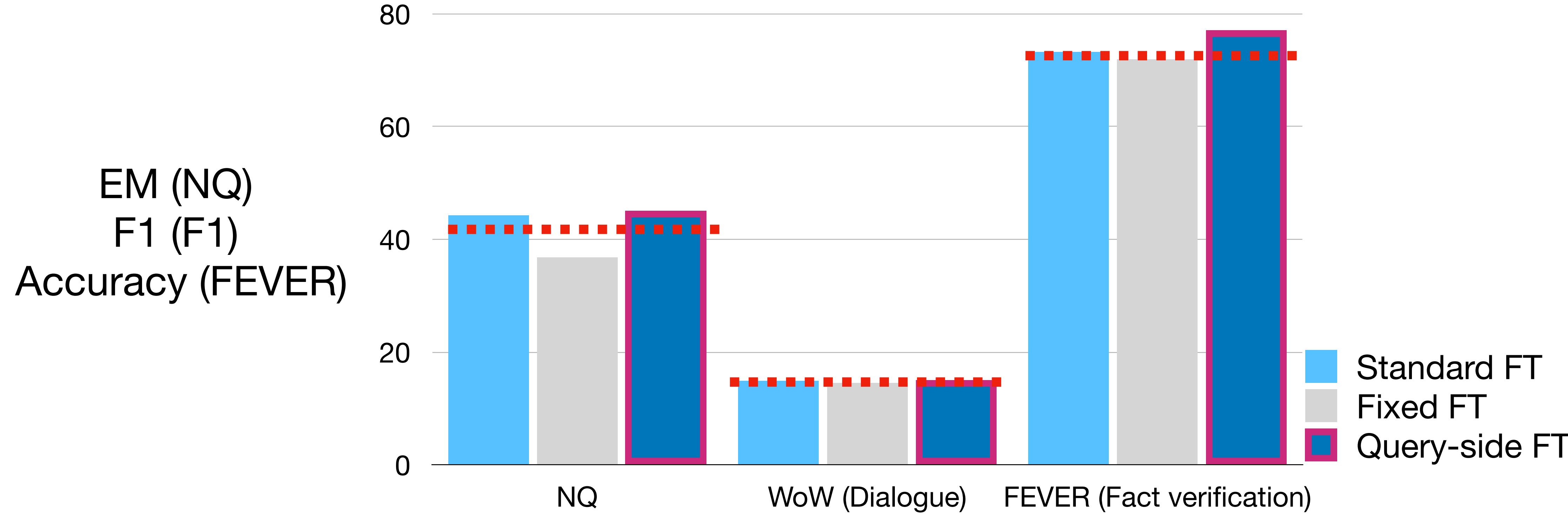


Ablations of efficient retrieval training



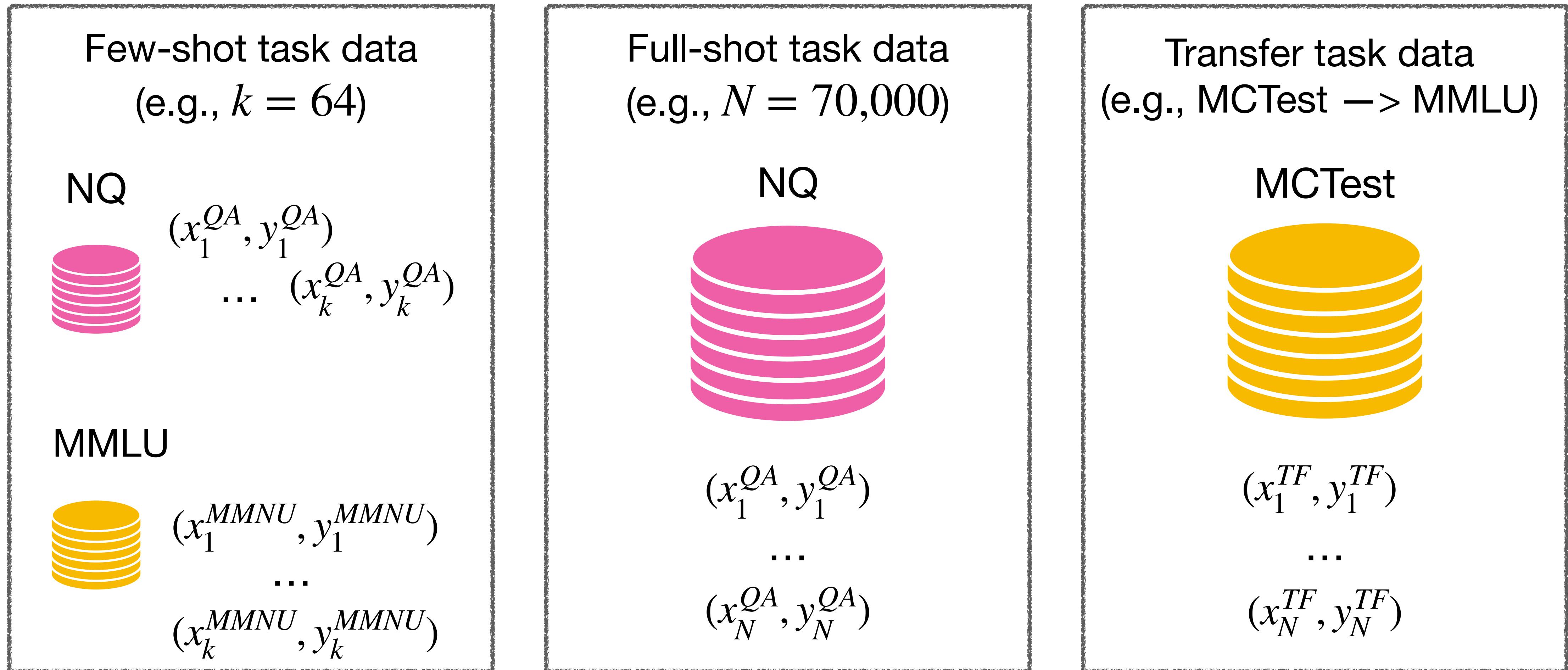
Fixed FT shows large performance drop on QA.

Ablations of efficient retrieval training



Query-side fine-tuning matches or outperforms full fine-tuning

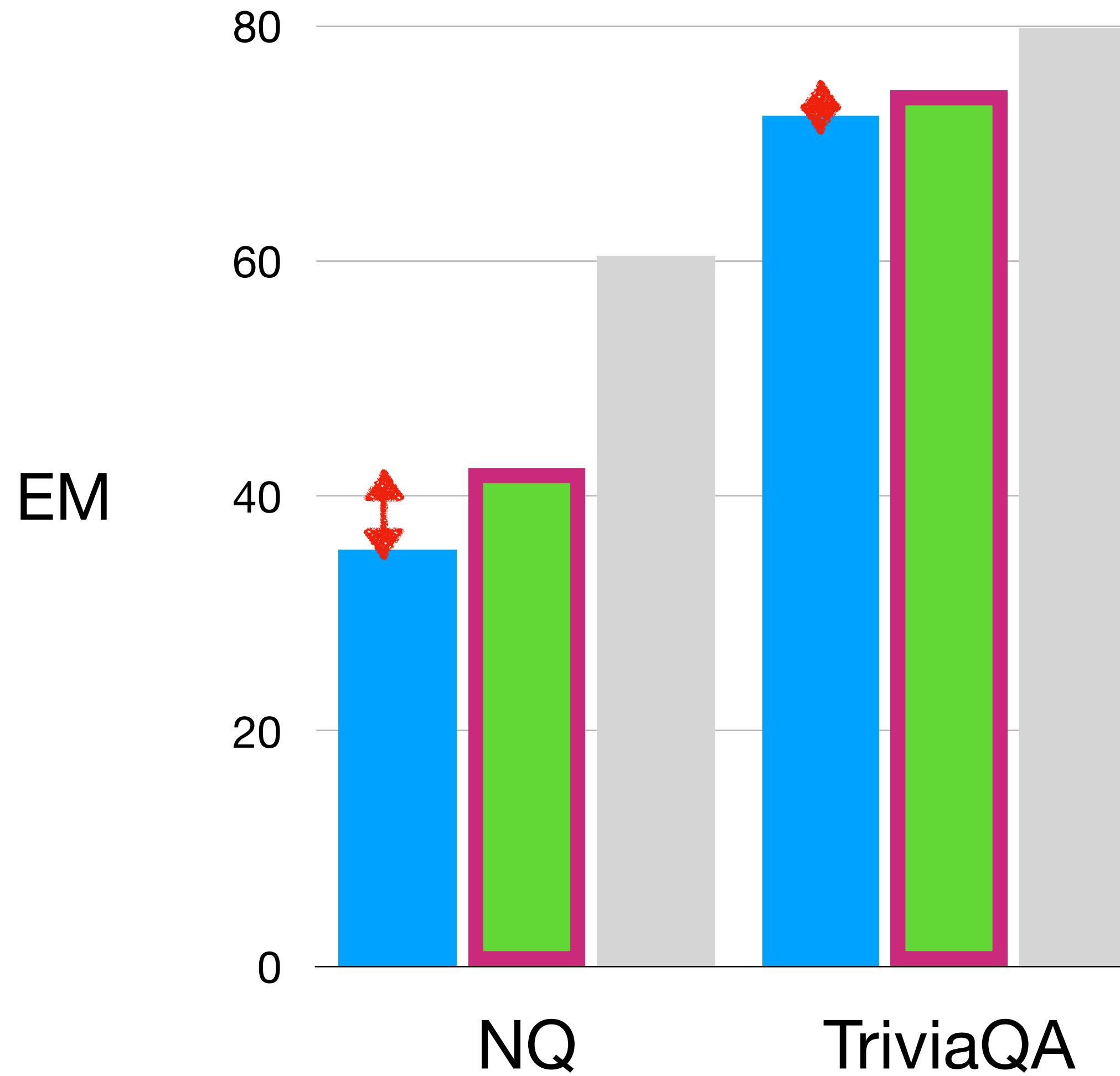
ATLAS: Few-shot v.s. full v.s. transfer setups



Kwiatkowski et al. 2019. “Natural Questions: A Benchmark for Question Answering Research”

Hendrycks et al. 2021. “Measuring Massive Multitask Language Understanding”

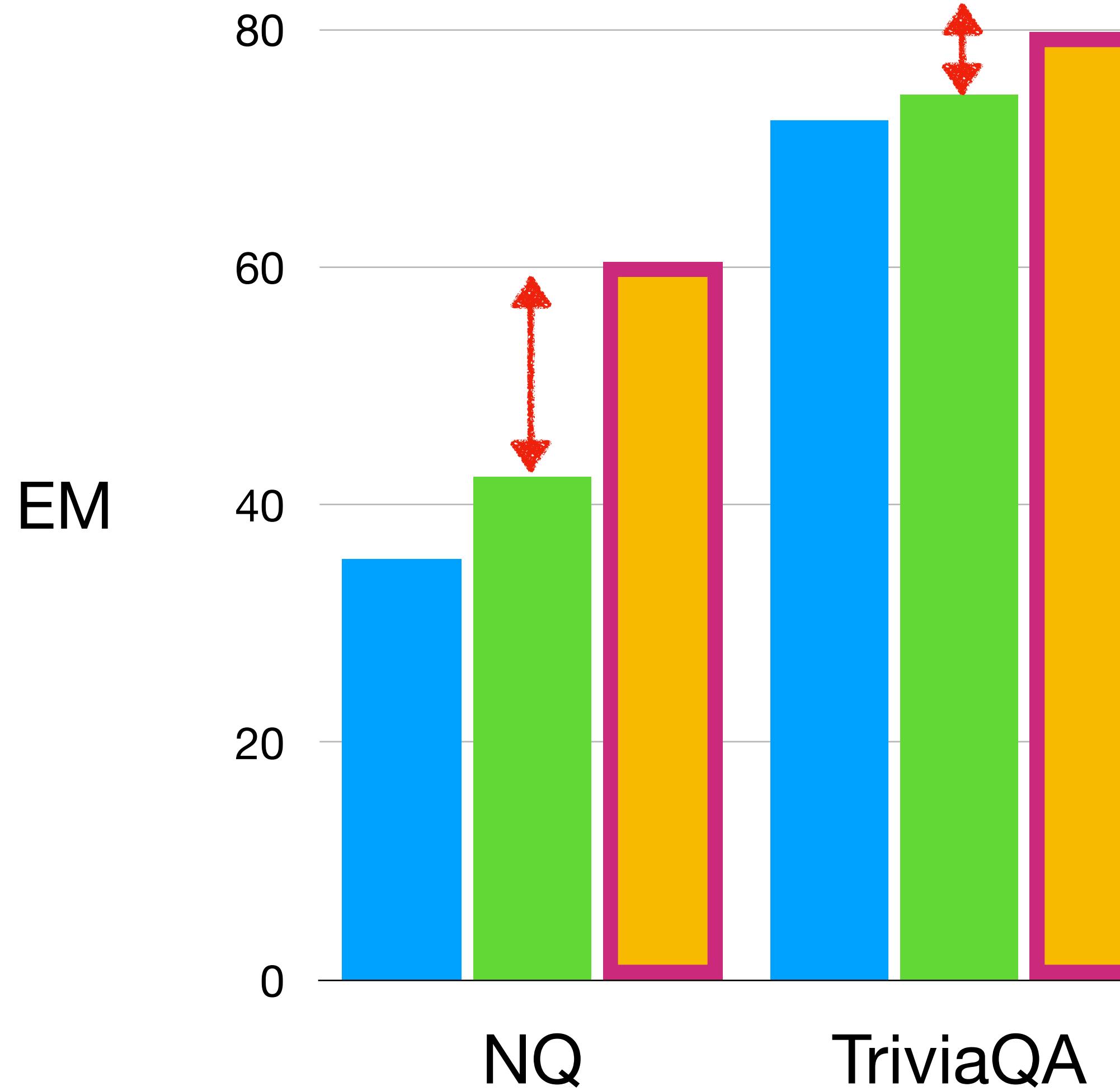
Task results



On QA, ATLAS largely outperforms other LLMs in few-shot

- Chinchilla (70B)
- ATLAS (Few; 11B)
- ATLAS (Full; 11B)

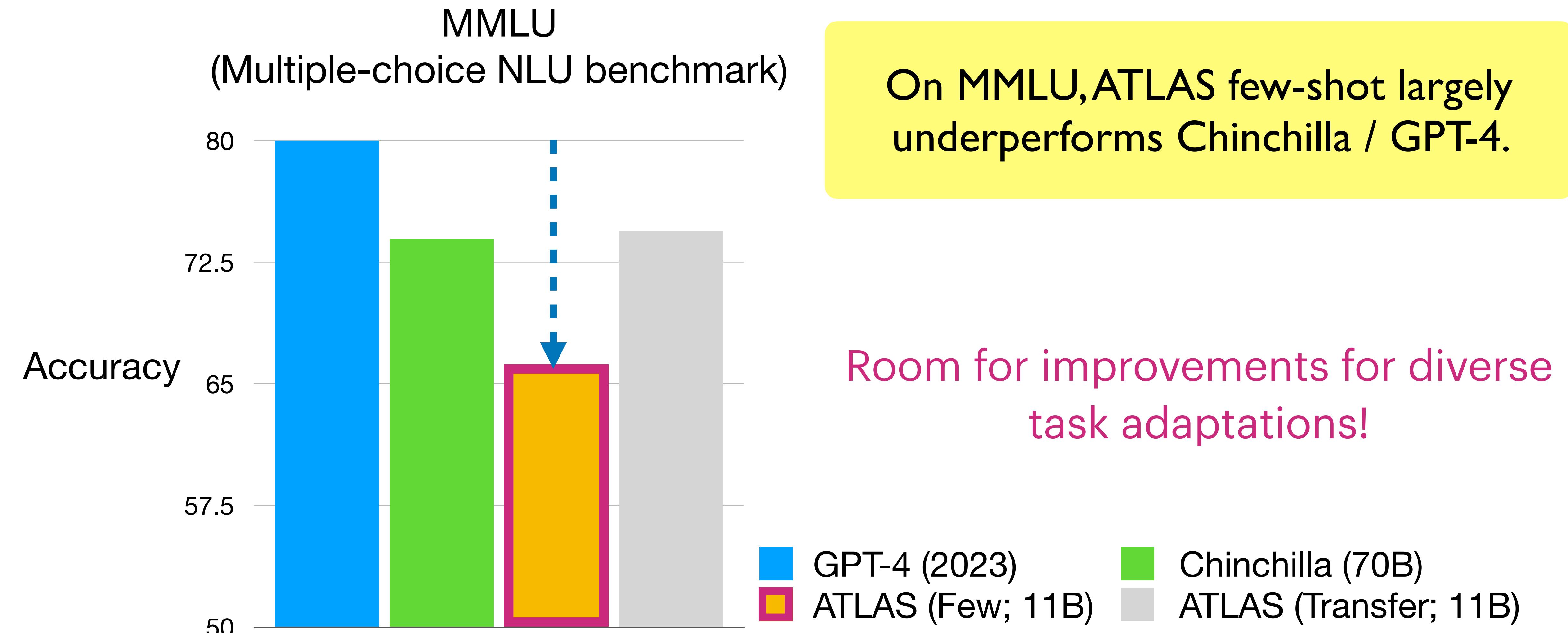
Task results



Full-shot fine-tuning further improves performance

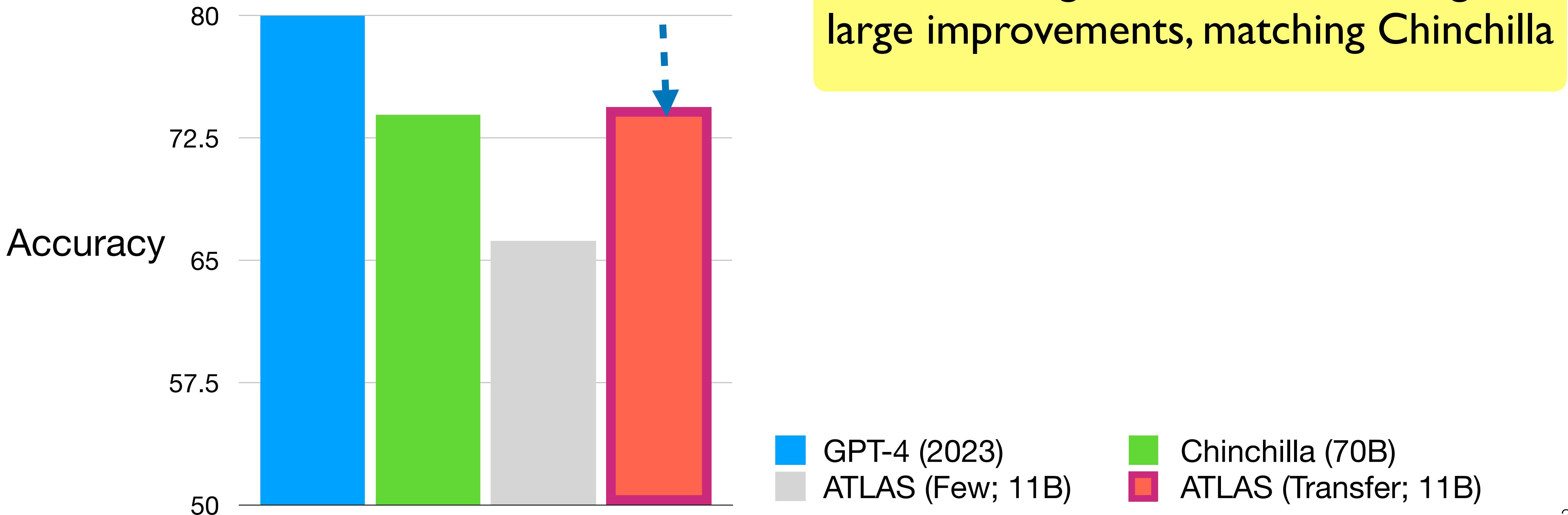
- Chinchilla (70B)
- ATLAS (Few; 11B)
- ATLAS (Full; 11B)

Task results



Task results

MMLU
(Multiple-choice NLU benchmark)



Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC

Fine-tuning for QA & knowledge-intensive tasks often gives strong performance (*even in few-shot*)

Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC

Fine-tuning a retriever for a task matters!

Downstream adaptation of retrieval-based LMs

What are the **tasks**?

- Open-domain QA
- Other knowledge-intensive tasks
- General NLU
- Language Modeling & other generation tasks

How to **adapt**?

- Fine-tuning
- **Reinforcement learning**
- Prompting

What is **data store**?

- Unlabeled Wikipedia / CC
- Web (Google / Bing Search Results)
- Training data

GopherCite (Menick et al., 2022)

User

What kind of animal is Scooby from Scooby Doo?

GopherCite

A Great Dane dog.

GopherCite (Menick et al., 2022)

User

What kind of animal is Scooby from Scooby Doo?

GopherCite

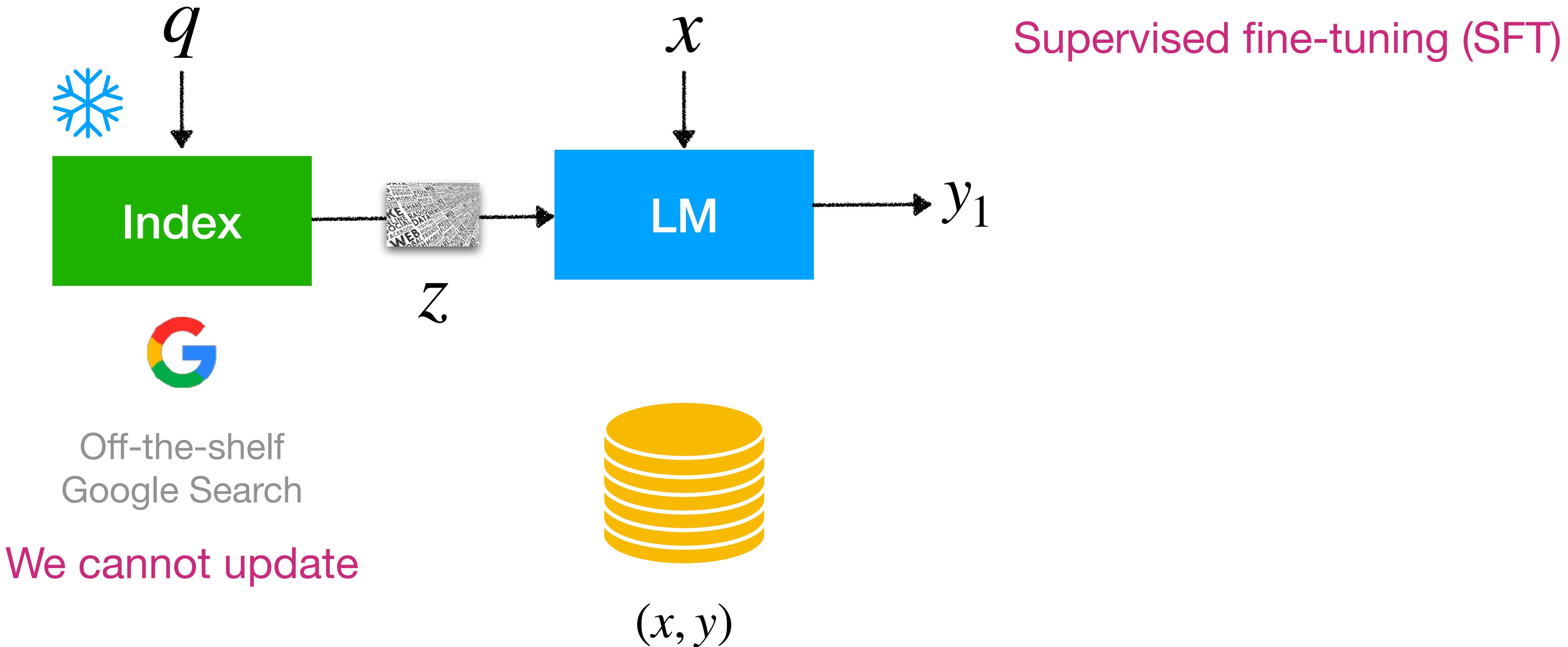
A Great Dane dog.

Wikipedia Page: Scooby-Doo

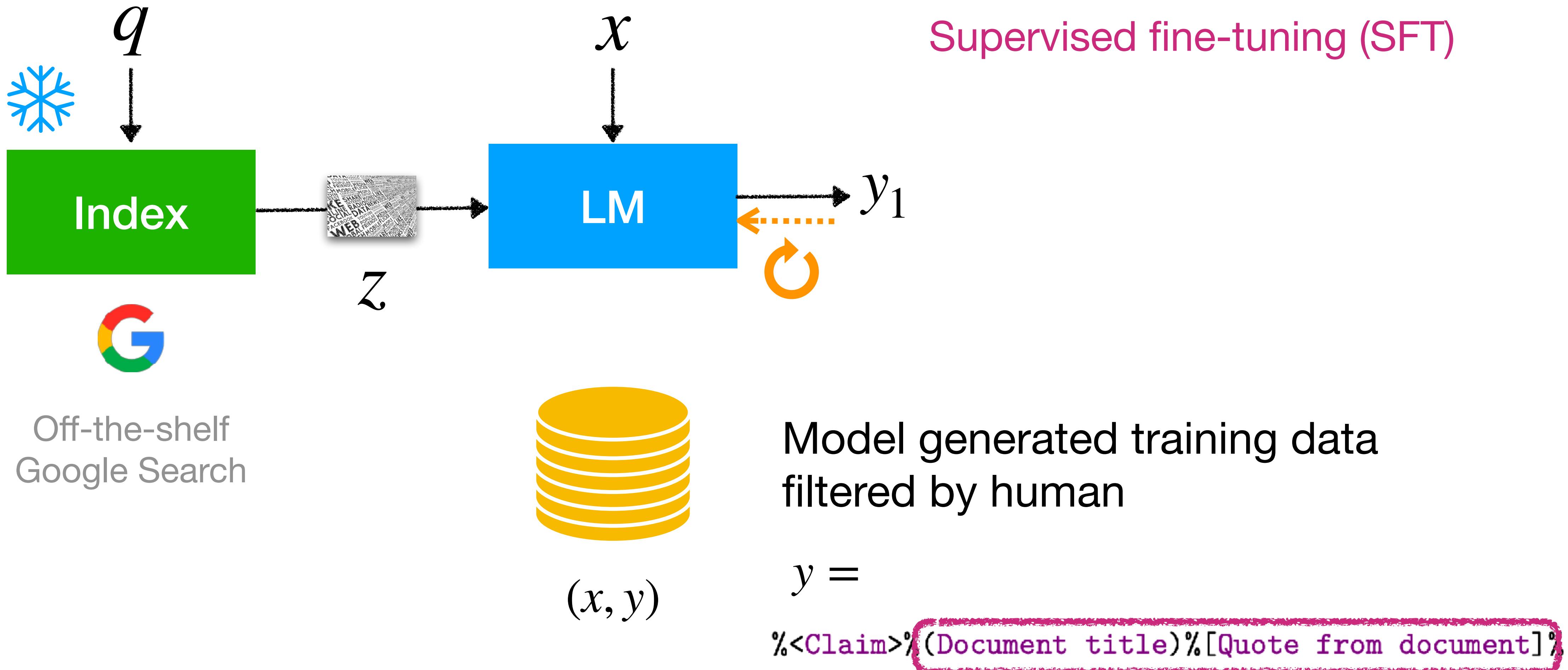
This Saturday-morning cartoon series featured teenagers Fred Jones, Daphne Blake, Velma Dinkley, and Shaggy Rogers, and their talking Great Dane named Scooby-Doo.

Extract and generate a quote to support an answer

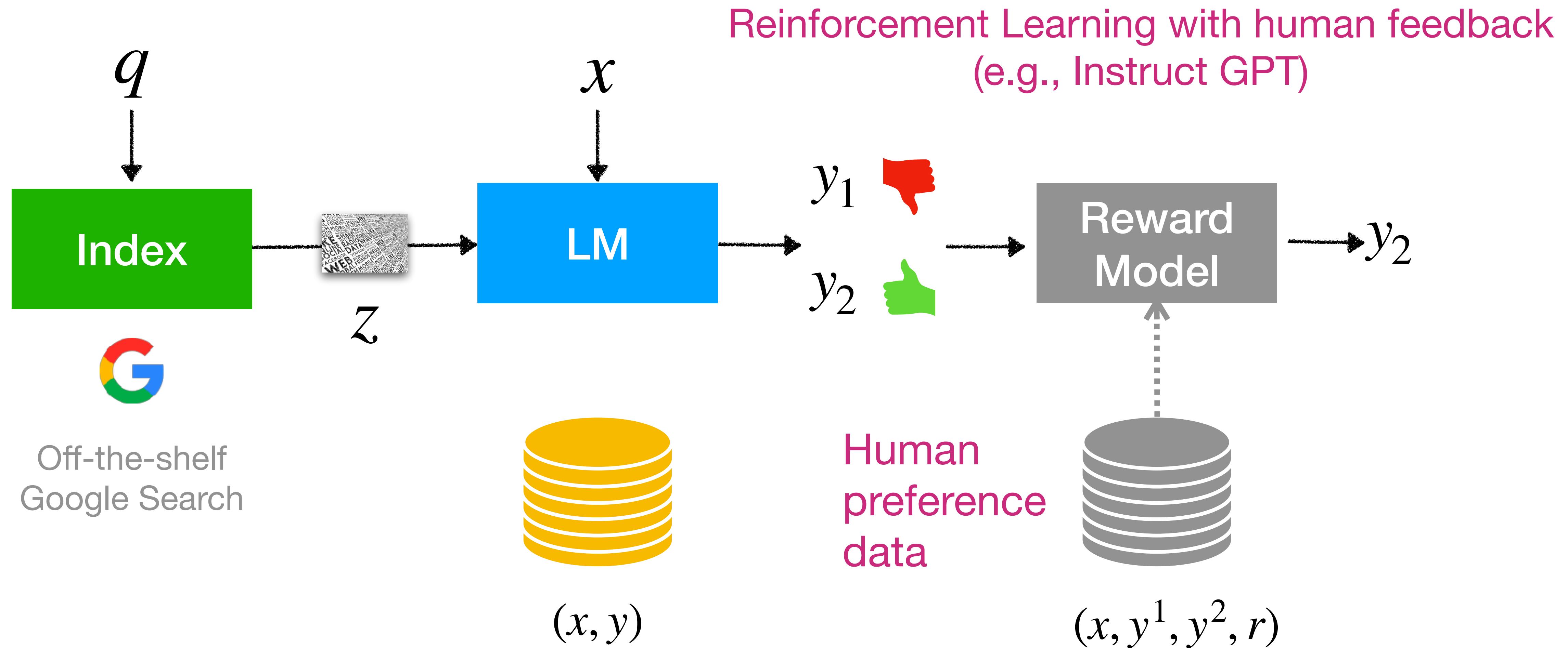
GopherCite: RLHF for answering with verified quotes



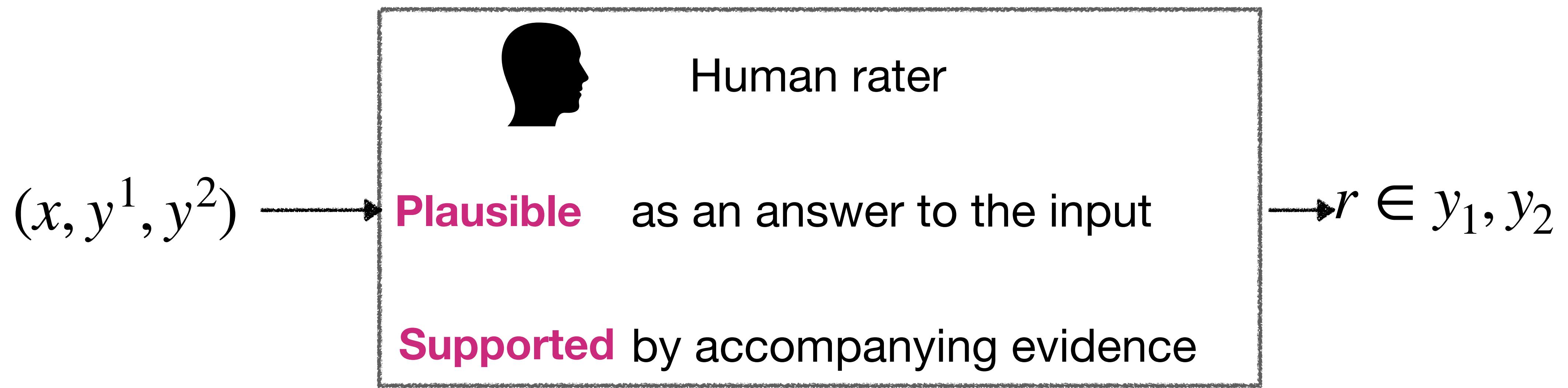
GopherCite: RLHF for answering with verified quotes



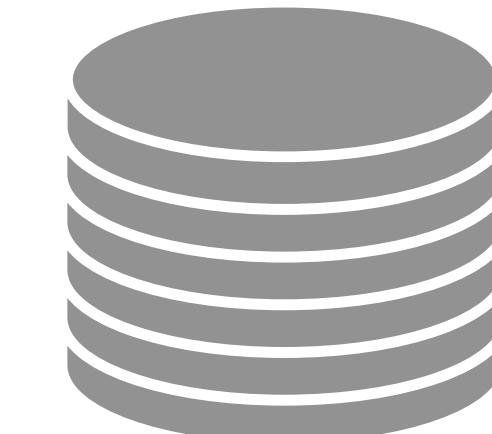
GopherCite: RLHF for answering with verified quotes



GopherCite: RLHF for answering with verified quotes



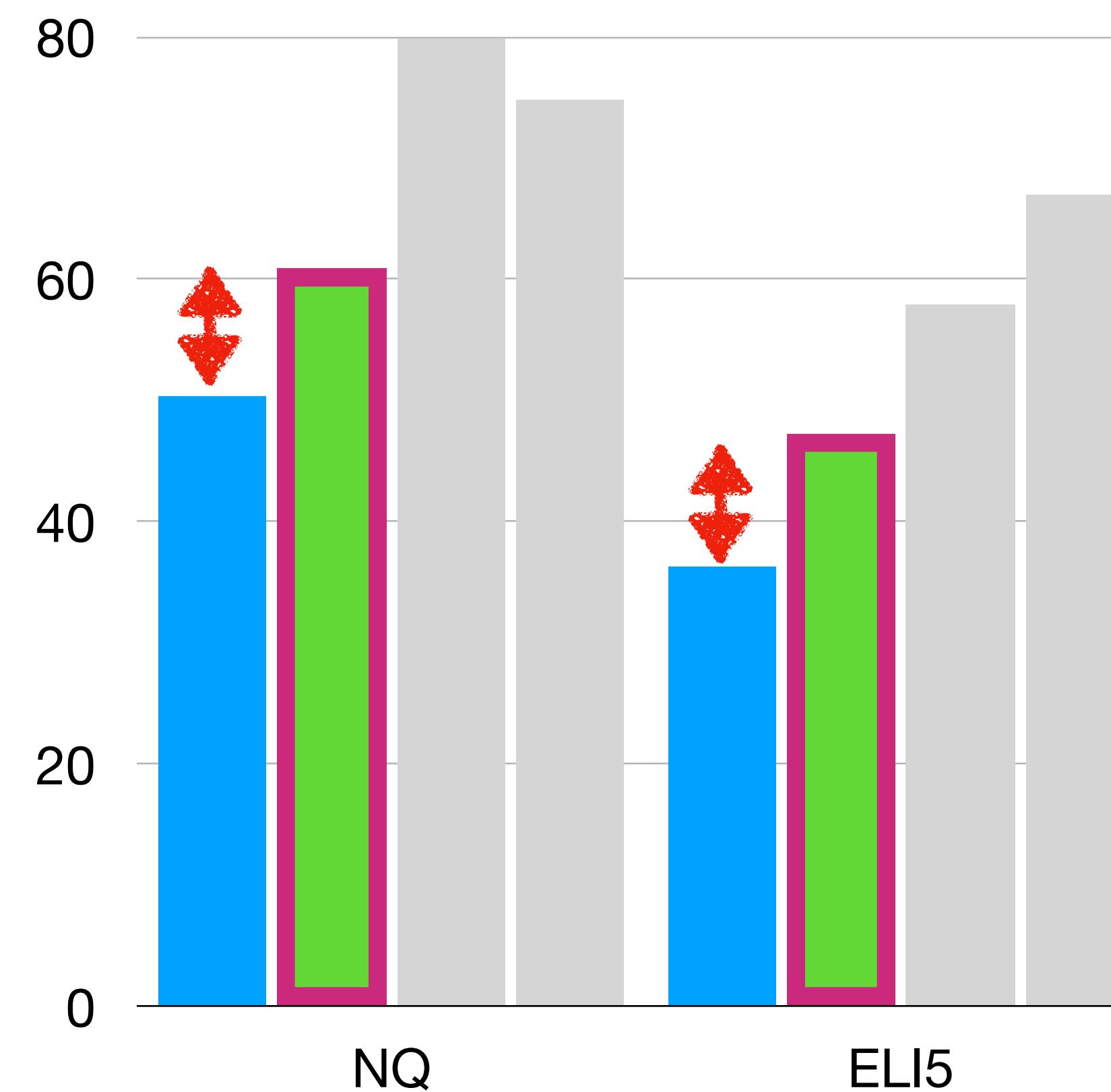
33k Human preference data



(x, y^1, y^2, r)

Effects of RL

S&P Natural Questions

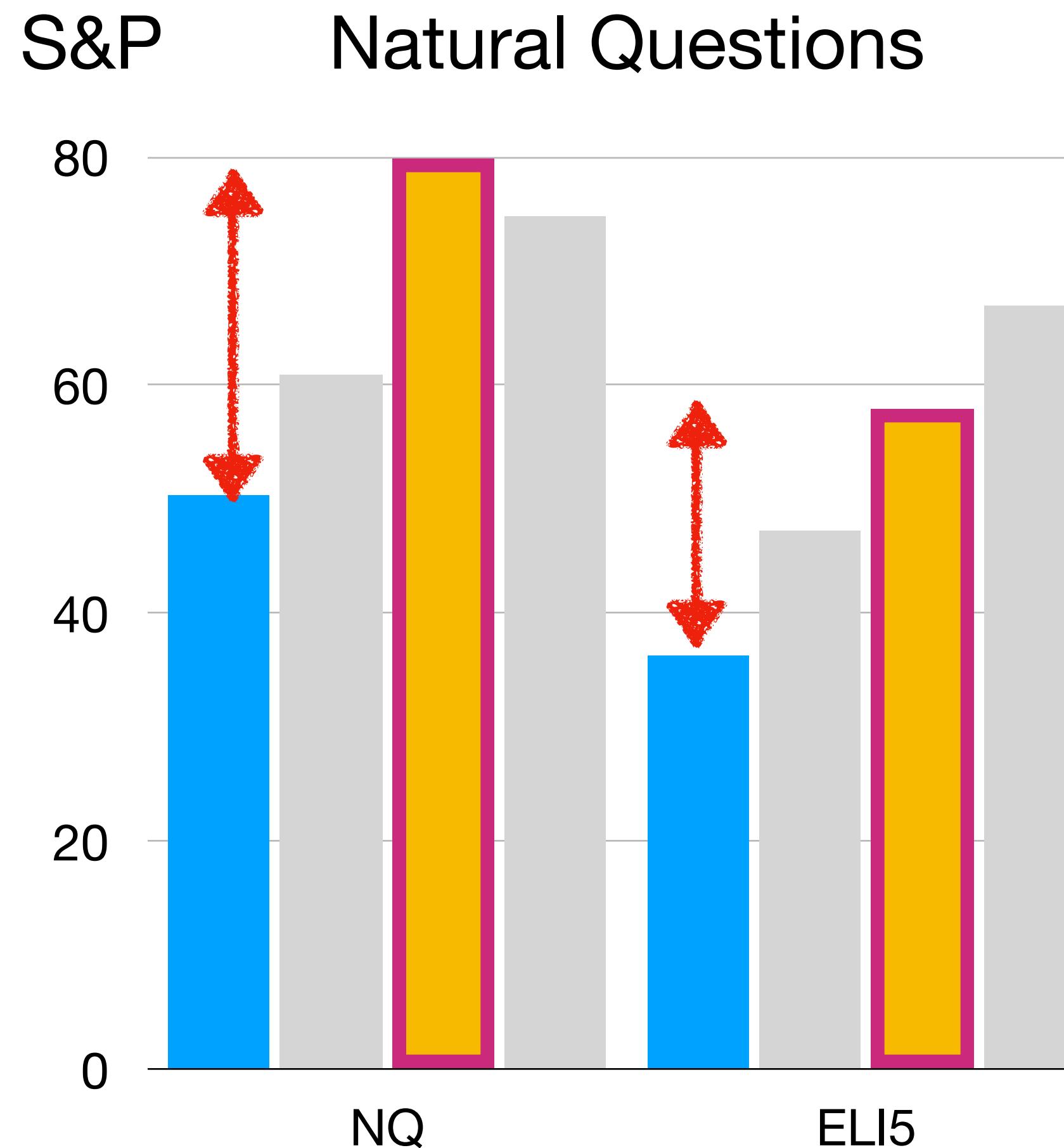


RL w/ human feedback improves the quality of top 1 generations

FT (best 1) RL (best 1)
FT (best 64) RL (best 64)

(S&P = Supported & Plausible)

Effects of RL



Sampling & reranking many generations
using a reward model gives gains from Top 1

FT (best 1) RL (best 1)
FT (best 64) RL (best 64)

Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC
GopherCite (Menick et al., 2022), also WebGPT (Nakano et al., 2021)	Open-domain QA, Long-form QA	Fine-tuning + RL (LM)	Google Search Results

Benefit of **fine-tuning**



Customizable



Competitive w/ more data



Requiring training

Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC
GopherCite (Menick et al., 2022), also WebGPT (Nakano et al., 2021)	Open-domain QA, Long-form QA	Fine-tuning + RL (LM)	Google Search Results

Benefit of **RL**



Better alignment with user preferences



Requiring additional data collection (preference)

Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC
GopherCite (Menick et al., 2022), also WebGPT (Nakano et al., 2021)	Open-domain QA, Long-form QA	Fine-tuning + RL (LM)	Google Search Results

What if we cannot train LMs for downstream tasks?
(e.g., lack of computational resources / proprietary LM ... etc)

Downstream adaptation of retrieval-based LMs

What are the **tasks**?

- Open-domain QA
- Other knowledge-intensive tasks
- General NLU
- Language Modeling & other generation tasks

How to **adapt**?

- Fine-tuning
- Reinforcement learning
- **Prompting**

What is **data store**?

- Wikipedia
- Web (Google / Bing Search Results)
- Training data

Prompting

k -shot instances ($k=0, 32 \dots$ etc)



Q: who Is the US president

A: Joe Biden

##

Q: What is the capital of US?

A: Washington DC.

##

Q: what is the Ontario capital?

A:

Doesn't require LM training on tasks!

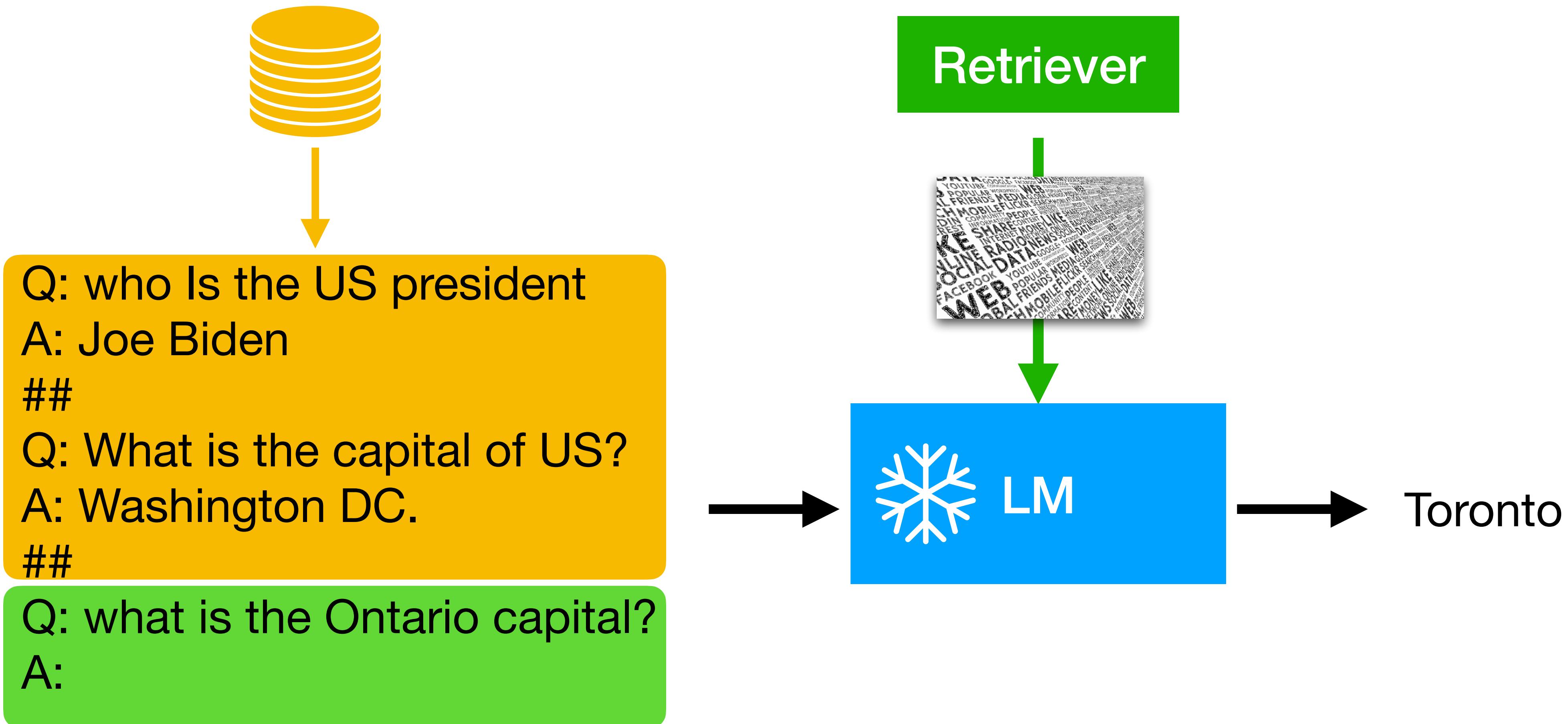
Training instances (demonstrations)



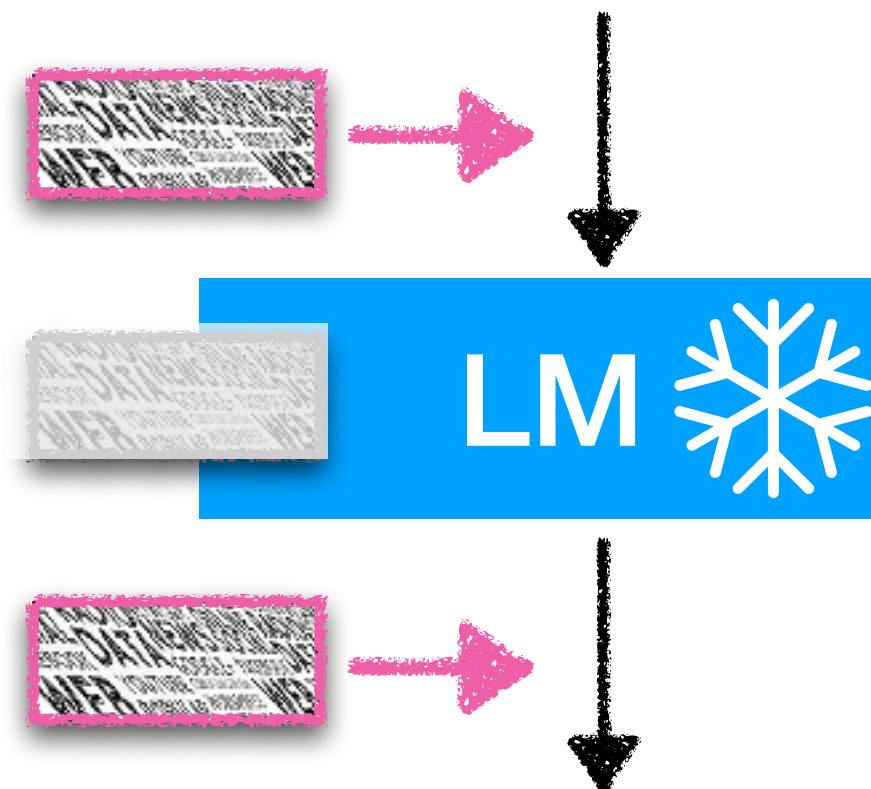
Test instances

Retrieval-based prompting

k -shot instances ($k=0, 32 \dots$ etc)



Design choice of retrieval-based Prompting



Input space:

Incorporate retrieved context in input space

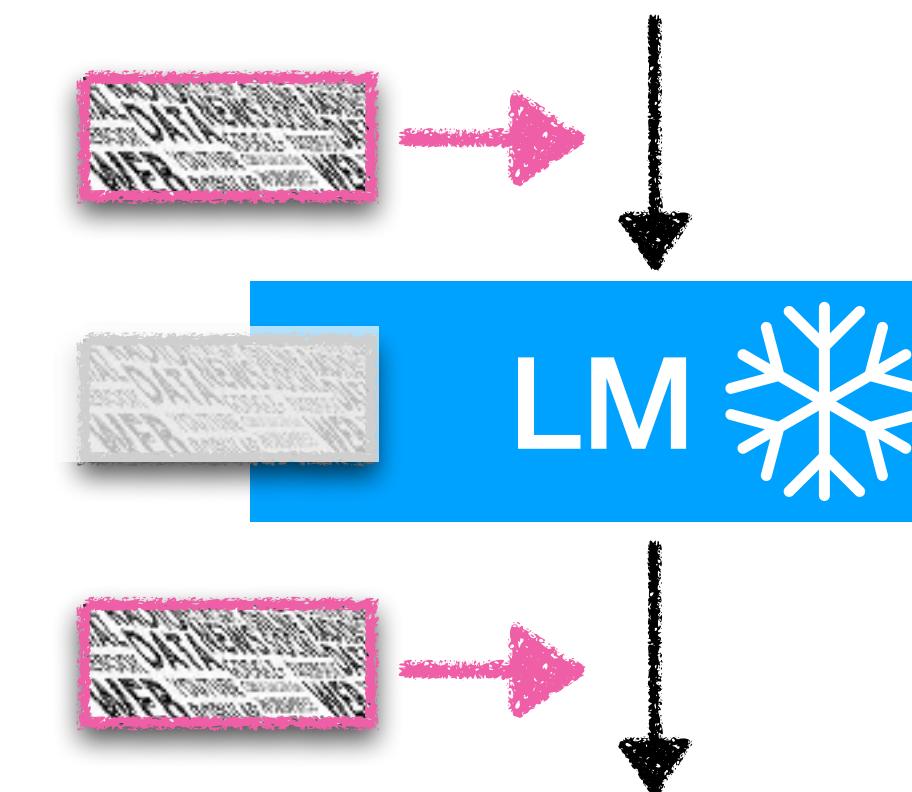
Intermediate layers:

N/A

Output space:

Interpolate token probability distributions in output space

Design choice of retrieval-based Prompting



Extending kNN-LM for downstream tasks

Input space:
Incorporate retrieved context in input space

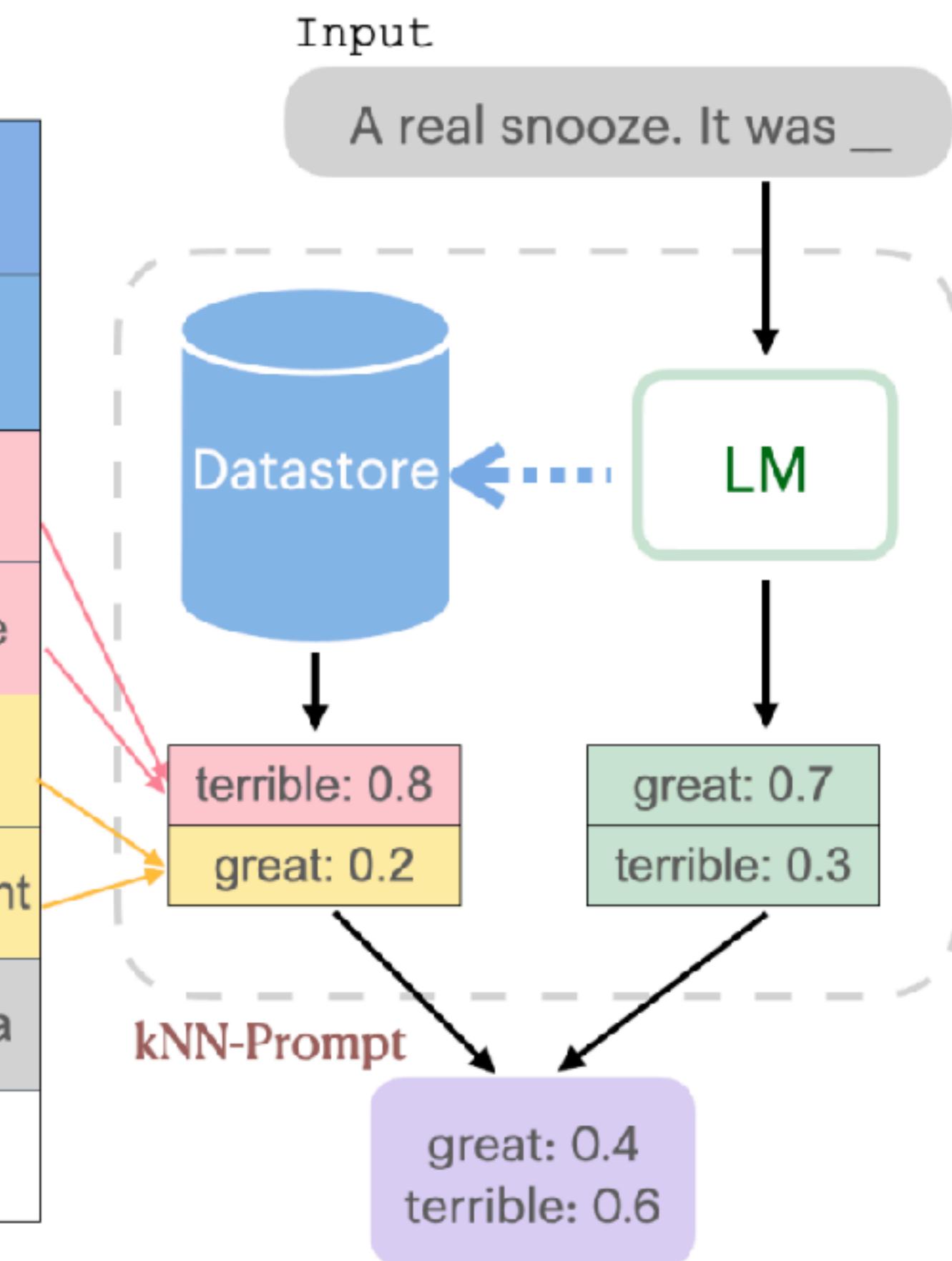
Intermediate layers:
N/A

Output space:
Interpolate token probability distributions in output space

kNN-Prompt (Shi et al., 2022)

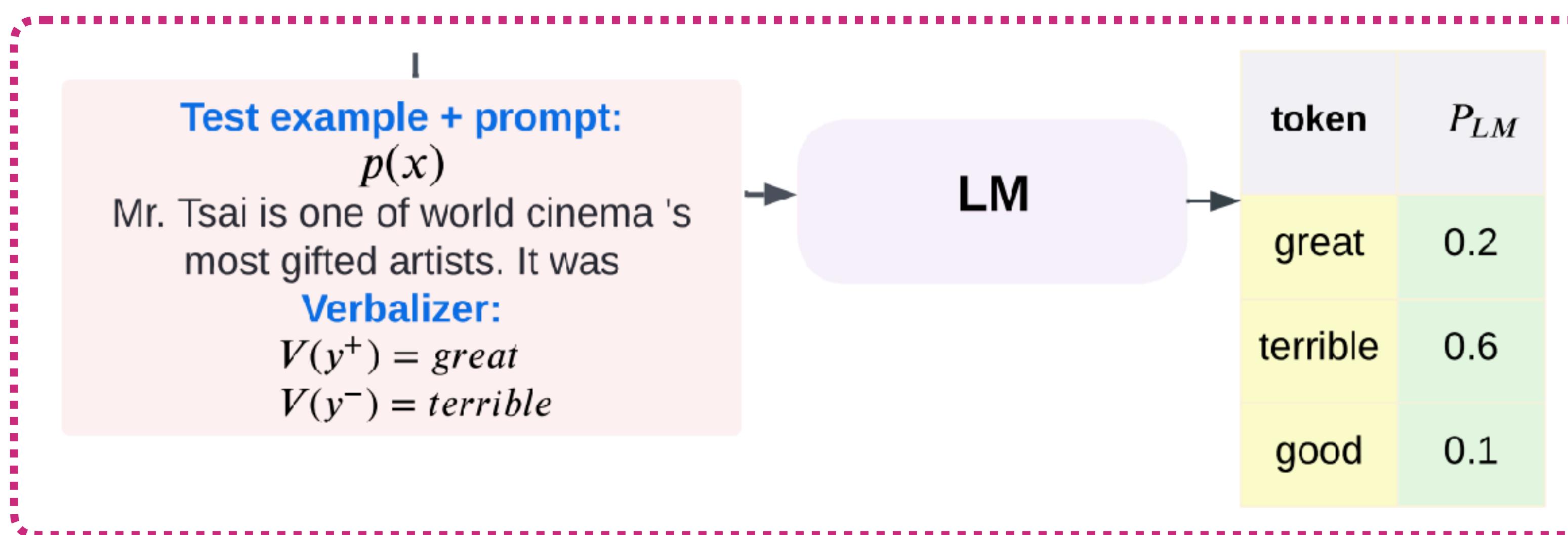
kNN LM with fuzzy verbalizers
for zero-/few-shot **classification**

Datastore	
Leftward contexts	Next token
The thriller is a real snooze. The director can't	silly
It is seriously a real snooze-fest. The acting is	terrible
The character and world design was	great
Five great movies that give us	excellent
This is junk food	cinema
...	...



kNN-Prompt (Shi et al., 2022)

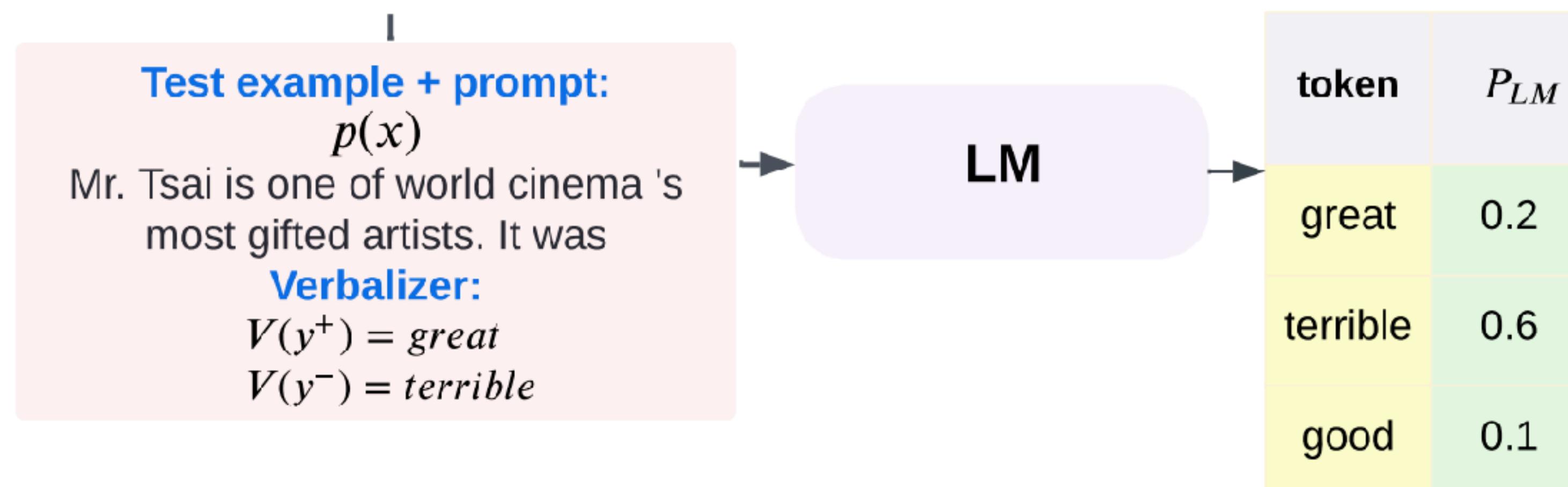
LM predicts next token



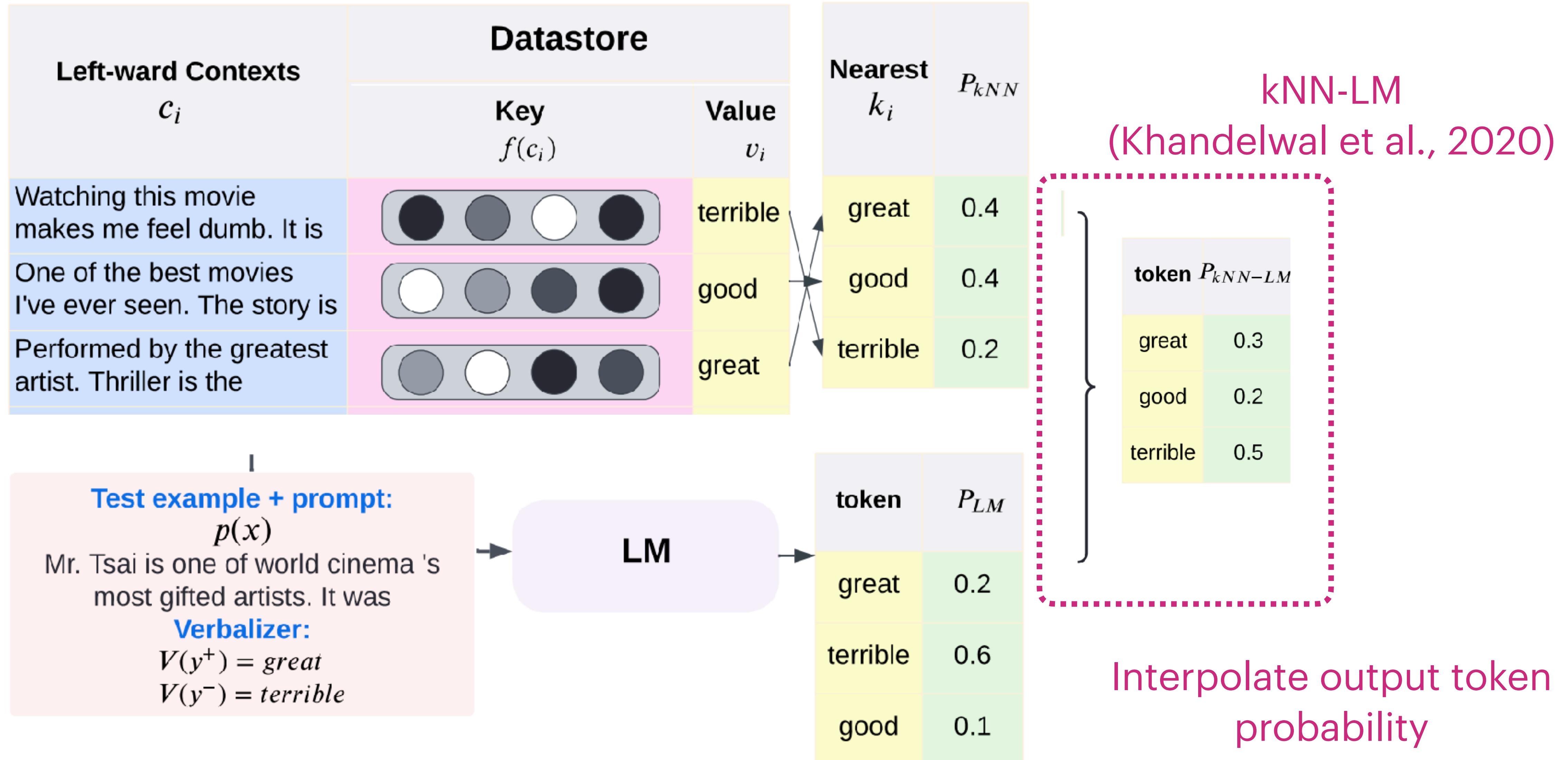
kNN-Prompt (Shi et al., 2022)

Left-ward Contexts		Datastore		Nearest k_i	P_{kNN}
c_i		Key $f(c_i)$	Value v_i		
Watching this movie makes me feel dumb. It is			terrible	great	0.4
One of the best movies I've ever seen. The story is			good	good	0.4
Performed by the greatest artist. Thriller is the			great	terrible	0.2

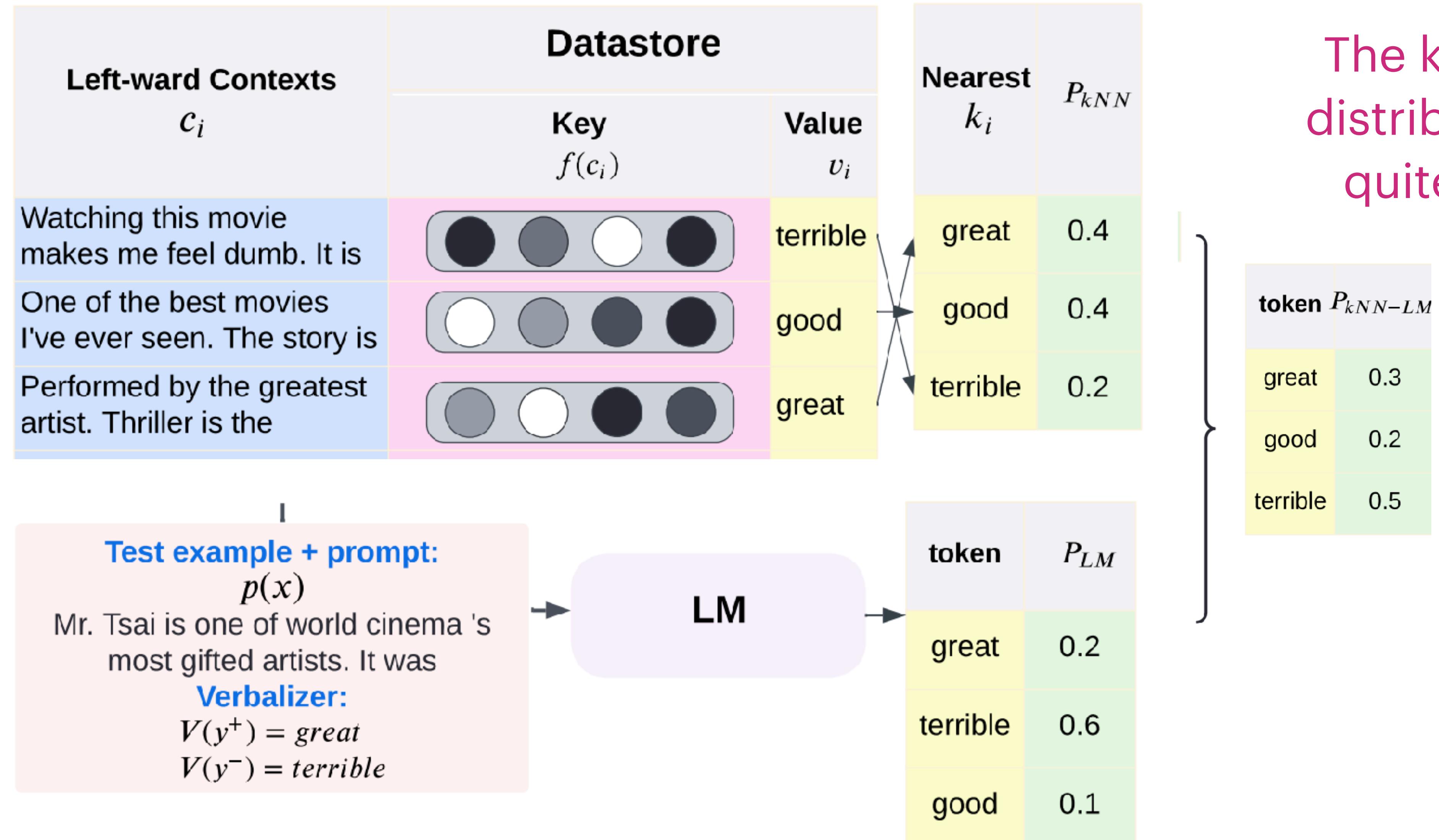
kNN predicts next tokens



kNN-Prompt (Shi et al., 2022)

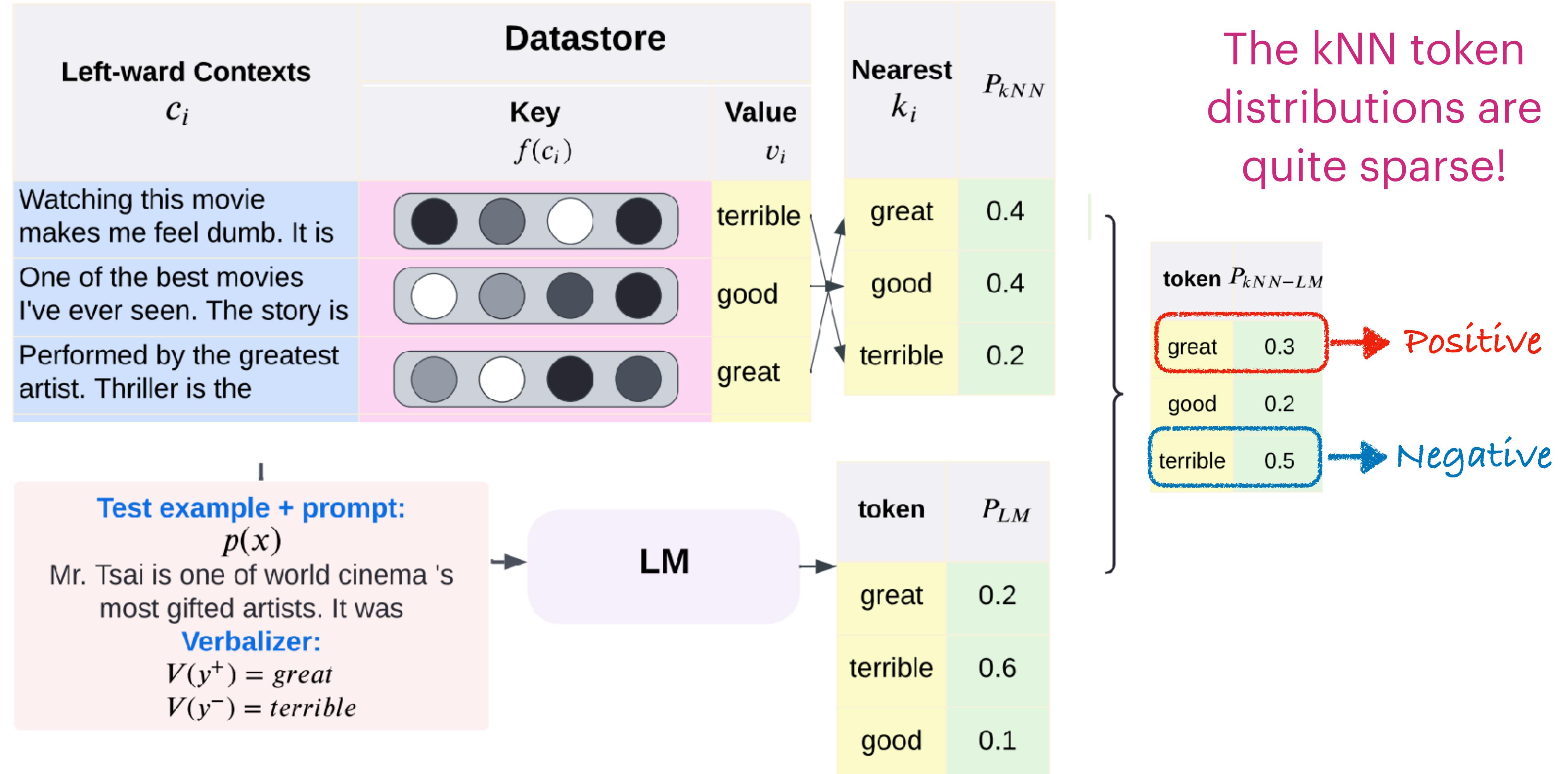


kNN-Prompt (Shi et al., 2022)

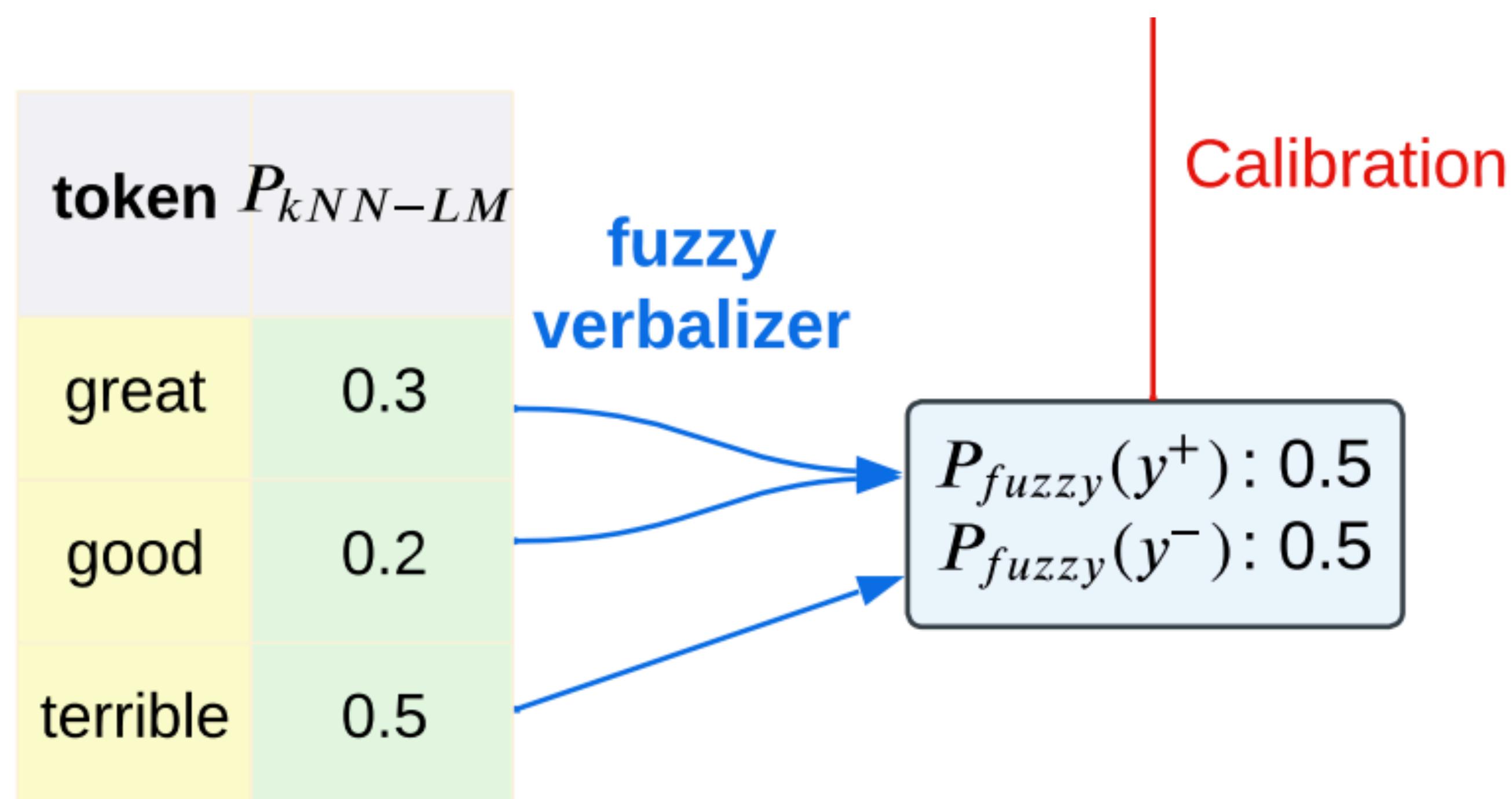


The kNN token distributions are quite sparse!

kNN-Prompt (Shi et al., 2022)



kNN-Prompt (Shi et al., 2022)

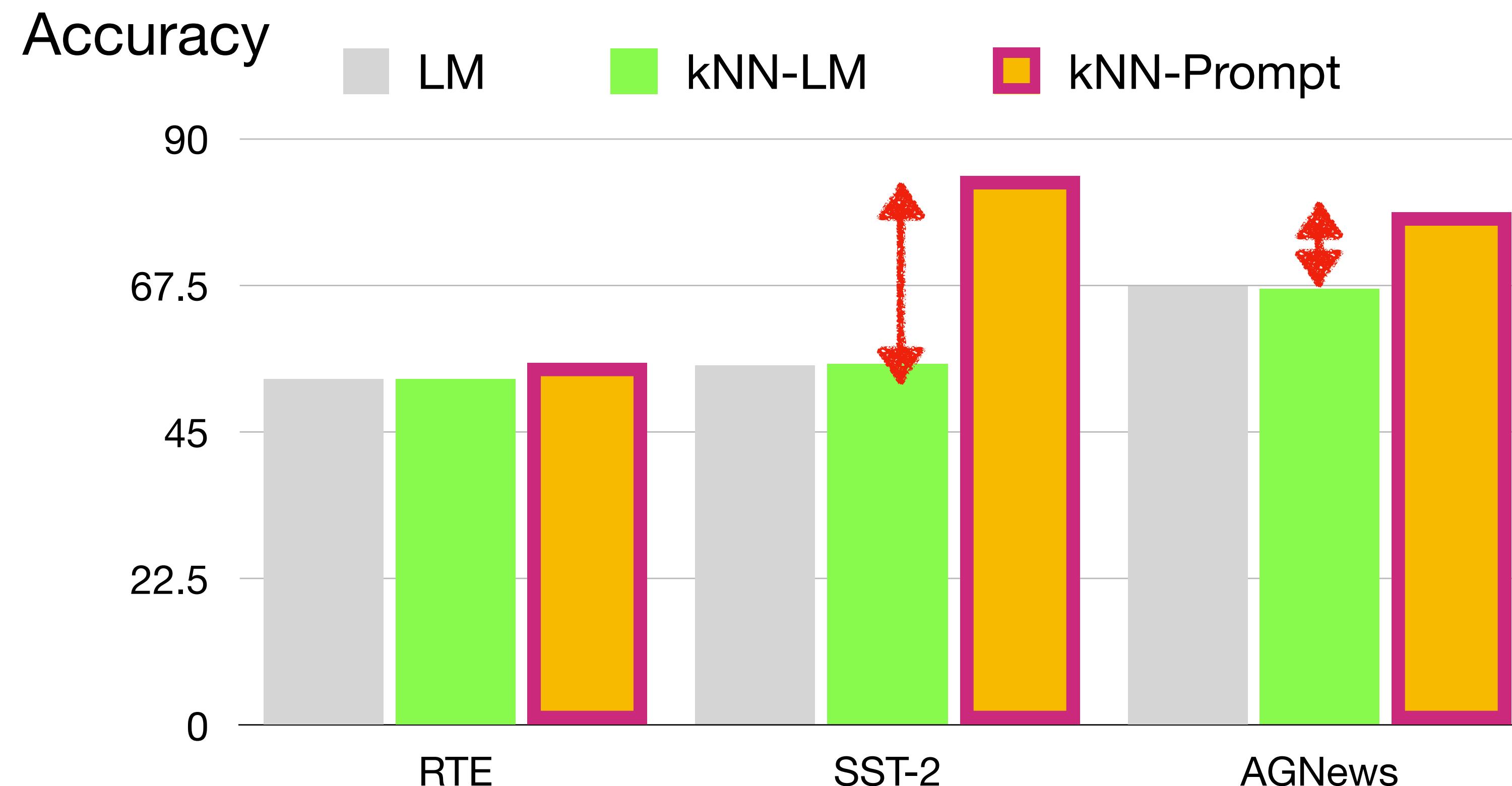


Fuzzy verbalizer maps token probability to target class labels

$$P_{FV}(y \mid x) \propto \sum_{v_i \in \mathcal{N}(v)} P(v_i \mid p(x))$$

Find similar tokens using GloVe & ConceptNet

Results on diverse classification tasks



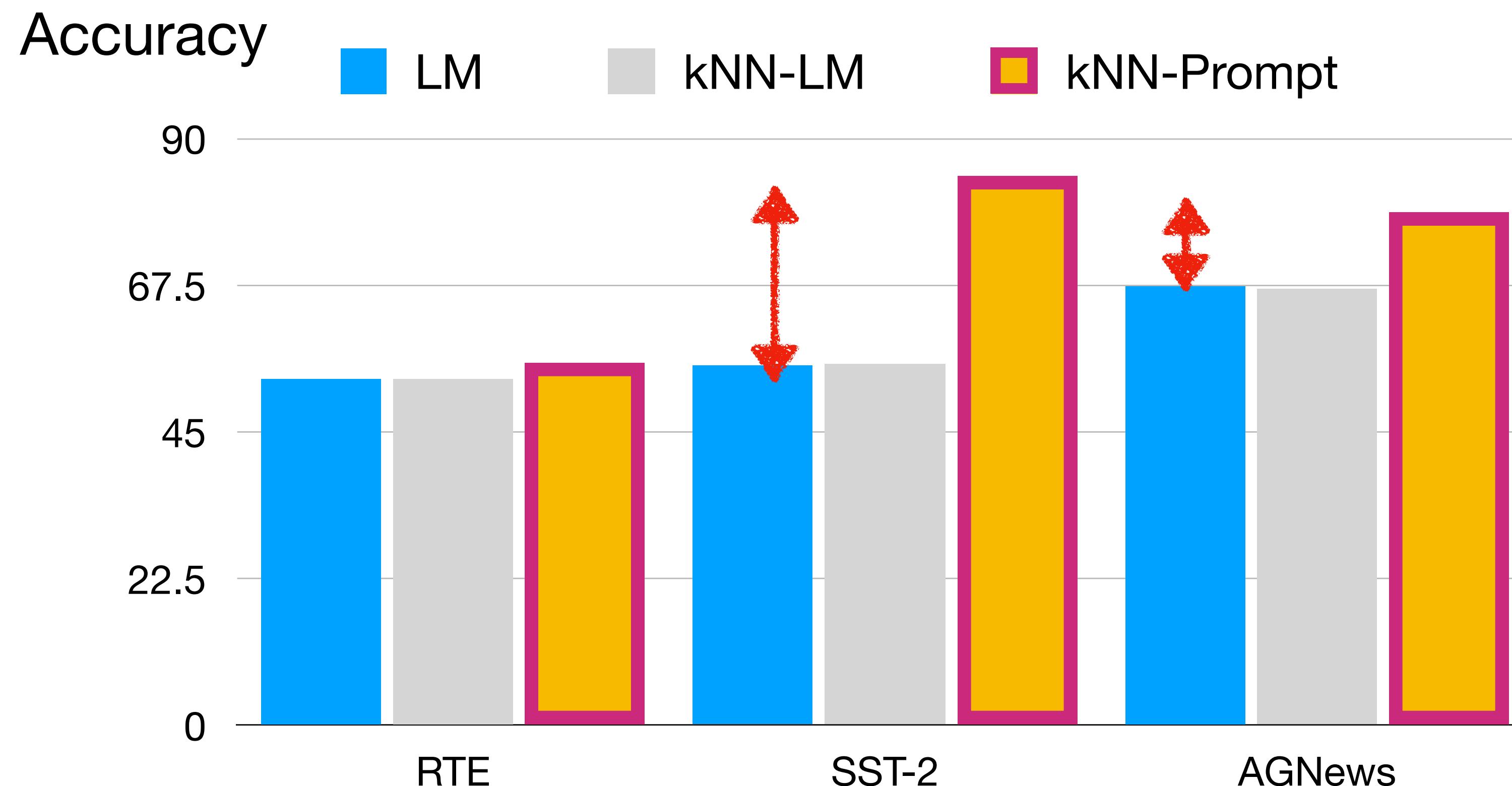
NLI
/ entailment

Sentiment
analysis

Topic
classification

Significant gains from
kNN-LM

Results on diverse classification tasks



kNN-prompt largely
outperforms vanilla LM in
zero-shot classification

NLI
/ entailment

Sentiment
analysis

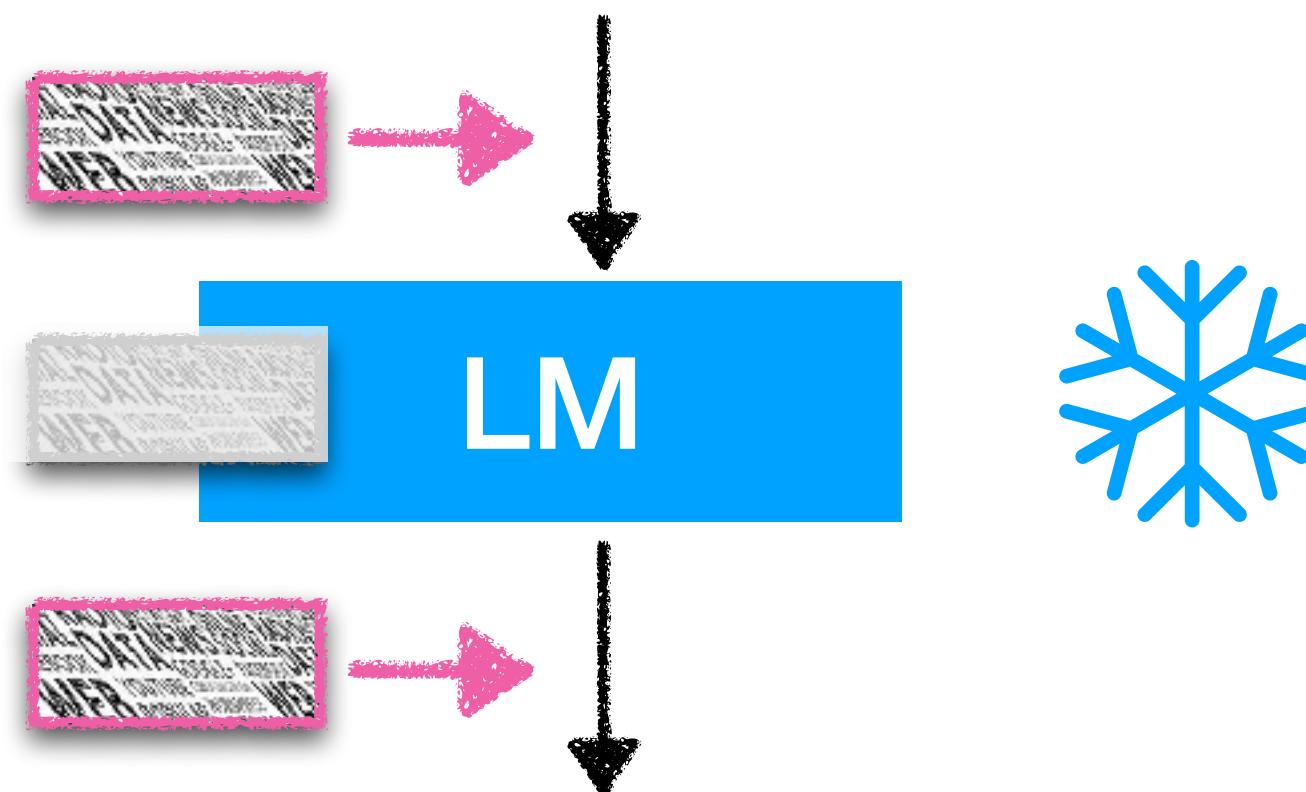
Topic
classification

Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC
GopherCite (Menick et al., 2022)	Open-domain QA, Long-form QA	Fine-tuning + RL (LM)	Google Search Results
kNN-prompt (Shi et al., 2022)	Classification	Prompting (output)	Wikipedia Pile

Retrieval-based LMs are effective in general NLU tasks!

Retrieval-based Prompting

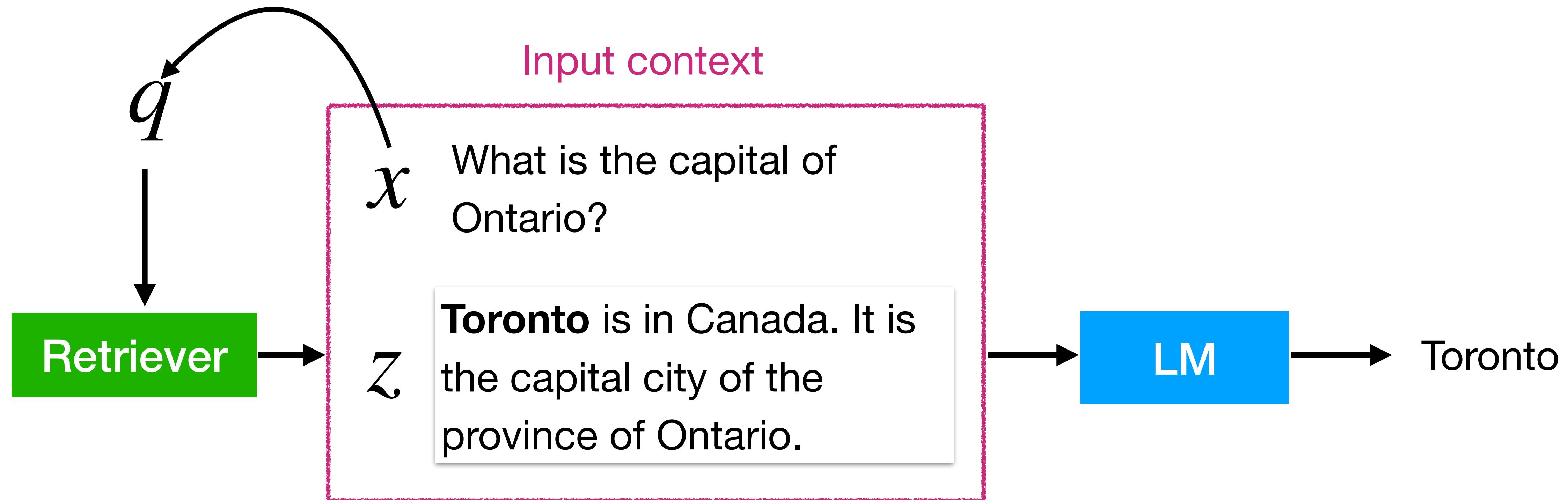


Input space:
Incorporate retrieved context in input space

Intermediate layers:
N/A

Output space:
Interpolate token probability
distributions in output space

Retrieval-in-context

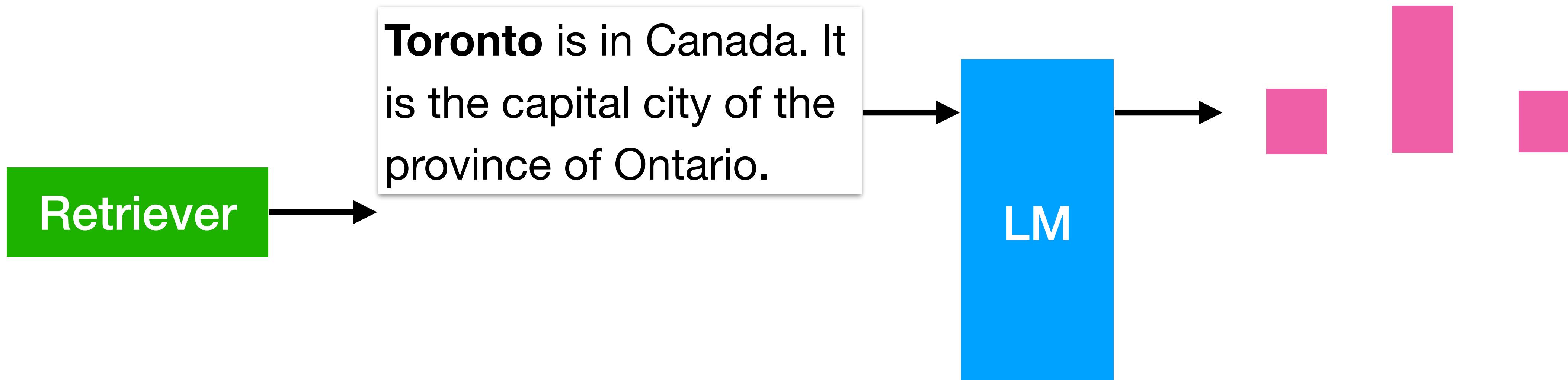


(Shi et al., 2023; Ram et al., 2022; Mallen et al., 2022; Yu et al., 2022; Press et al., 2022; *inter alia*)

REPLUG (Shi et al., 2023; Section 3&4)

✗ What is the capital of Ontario?

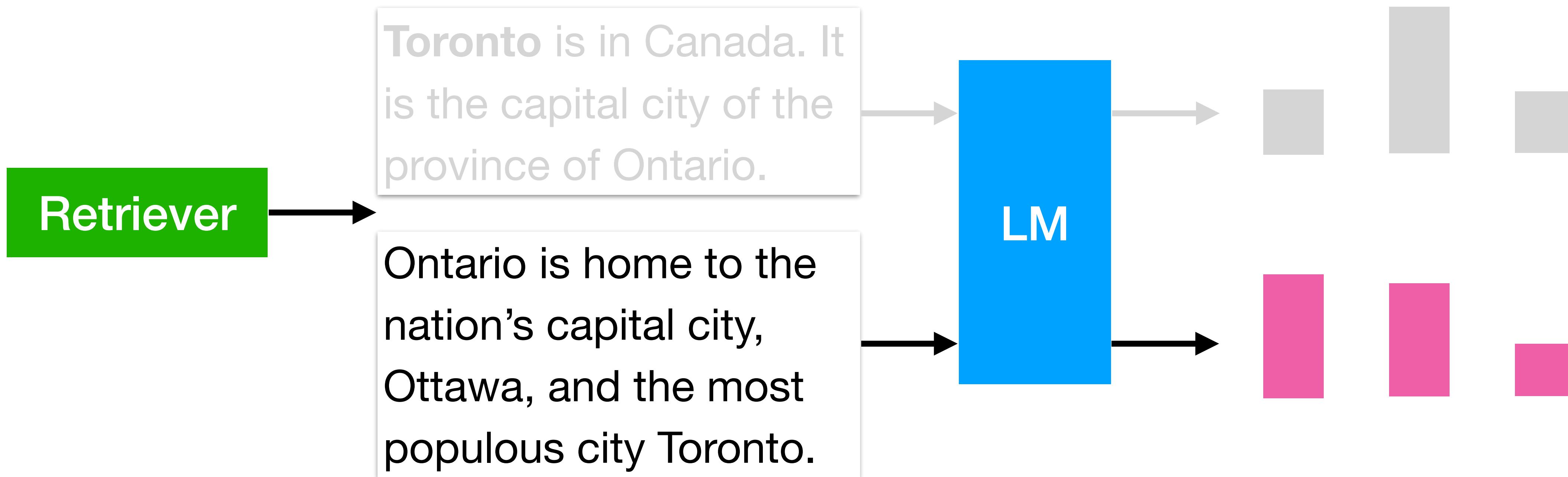
Ottawa **Toronto** Ontario



REPLUG (Shi et al., 2023; Section 3&4)

✗ What is the capital of Ontario?

Ottawa Toronto Ontario



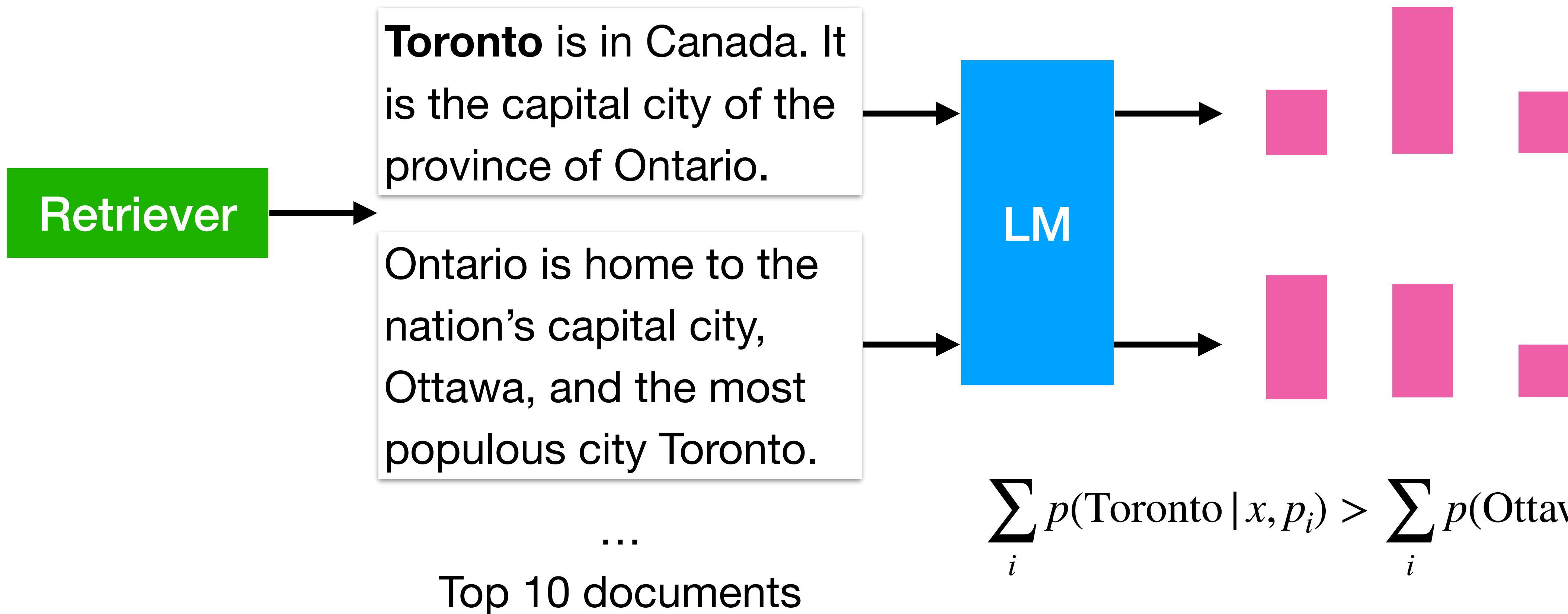
...

Top 10 documents

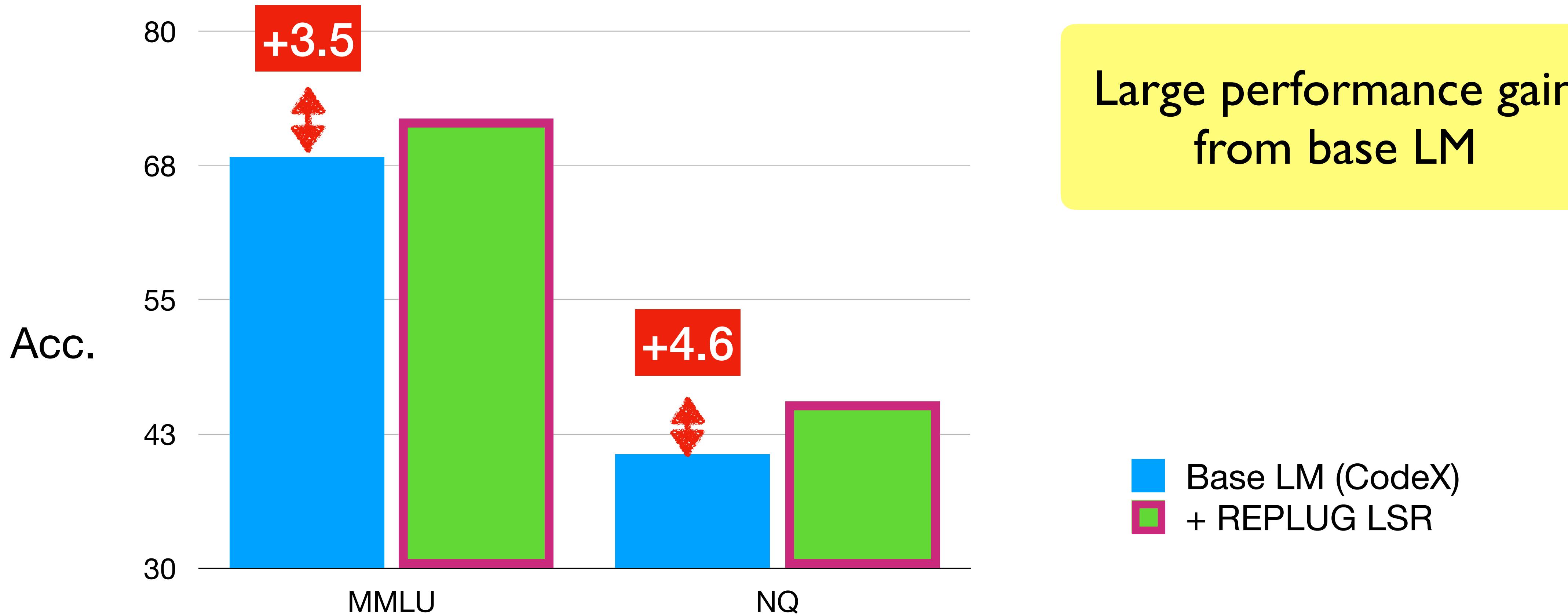
REPLUG (Shi et al., 2023; Section 3&4)

✗ What is the capital of Ontario?

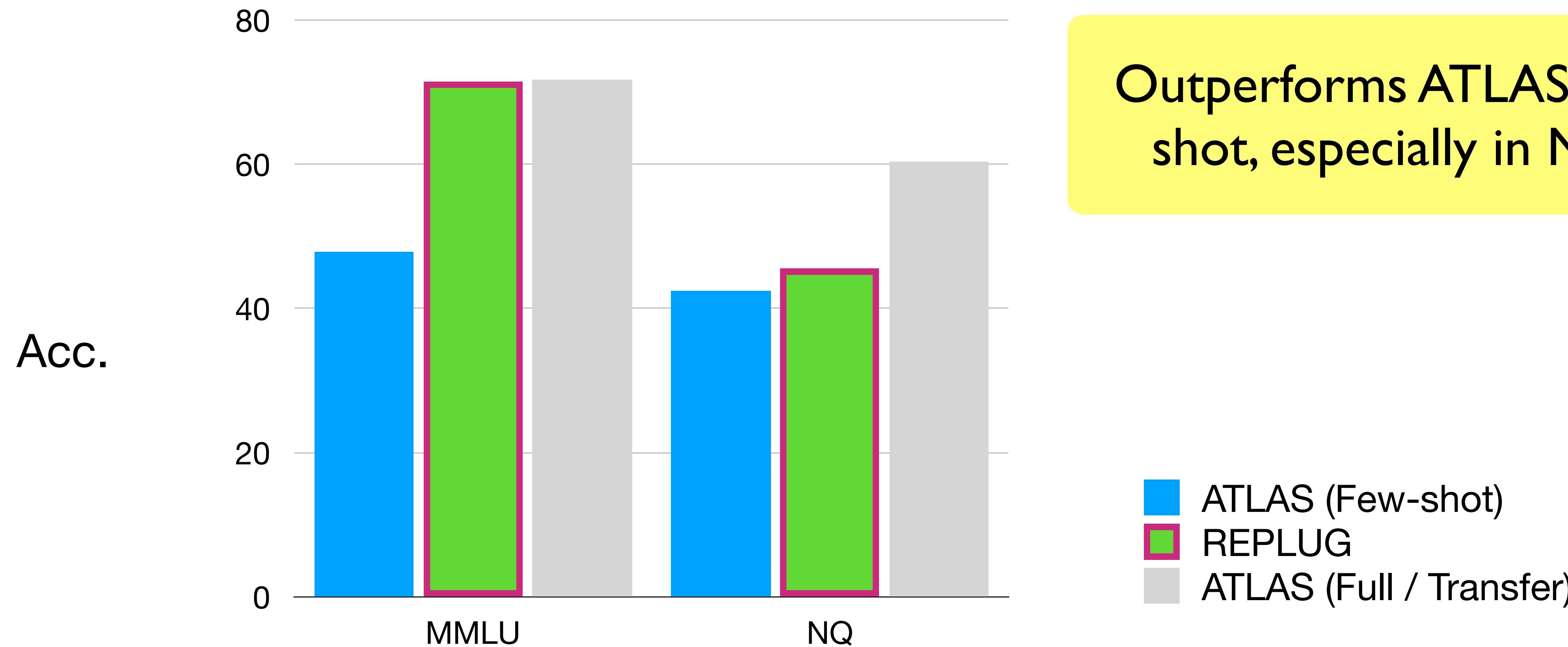
Ottawa **Toronto** Ontario



REPLUG: Results on QA & MMLU

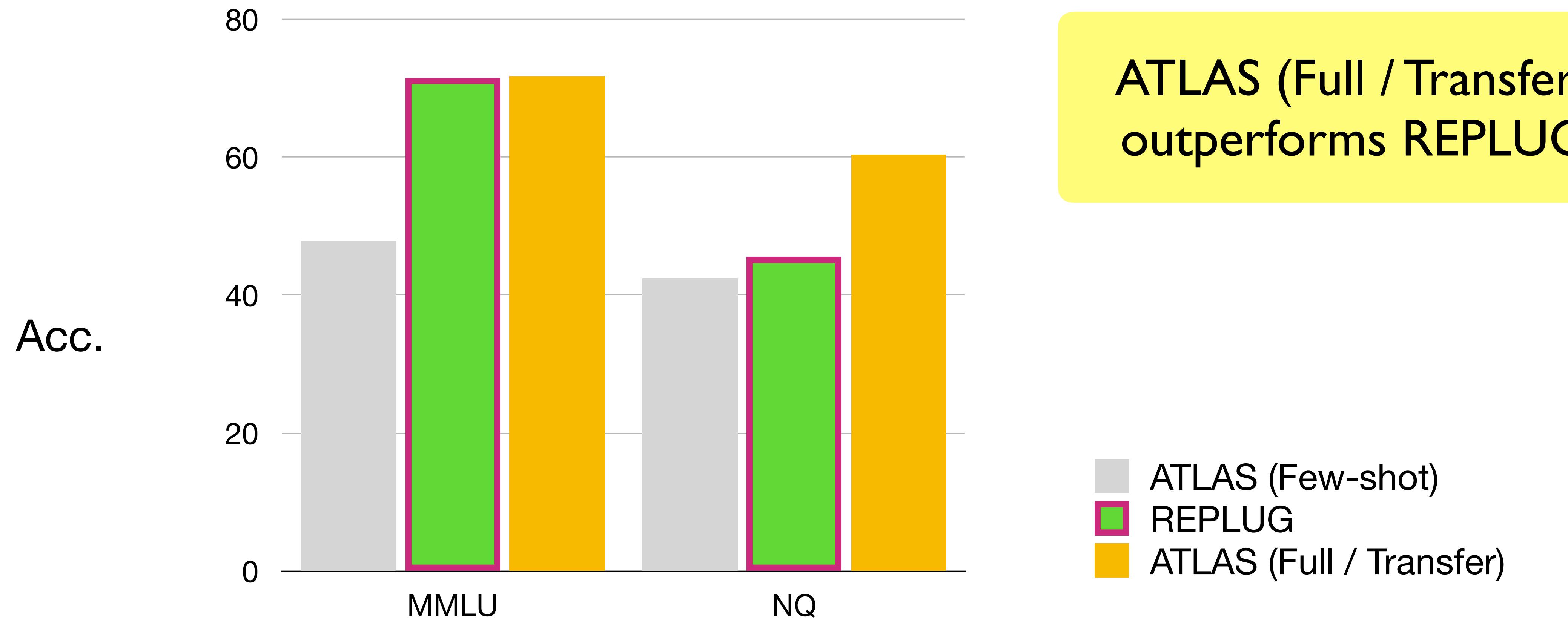


REPLUG: Comparison with ATLAS



Outperforms ATLAS in few-shot, especially in MMLU

REPLUG: Comparison with ATLAS



Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC
GopherCite (Menick et al., 2022)	Open-domain QA, Long-form QA	Fine-tuning + RL (LM)	Google Search Results
kNN-prompt (Shi et al., 2022)	Classification	Prompting (output)	Wikipedia CC
REPLUG (Shi et al., 2023)	Knowledge-intensive	Prompting (input)	Wikipedia CC

Benefit of **retrieval-based prompting**



No training & strong performance



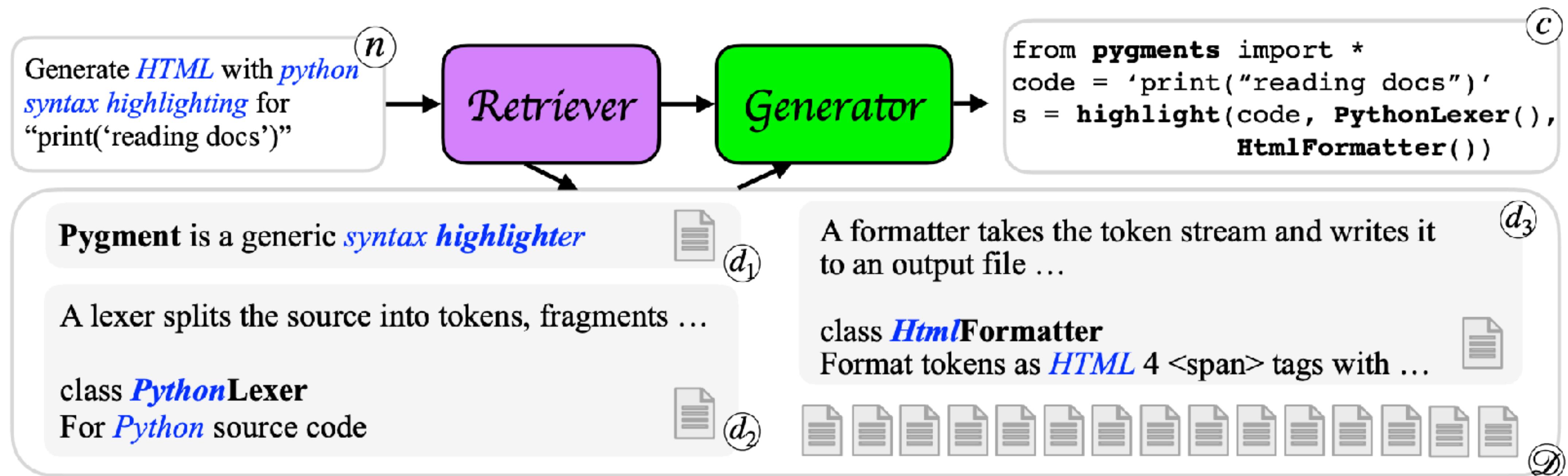
Hard to control, underperforming full FT model

Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC
GopherCite (Menick et al., 2022)	Open-domain QA, Long-form QA	Fine-tuning + RL (LM)	Google Search Results
kNN-prompt (Shi et al., 2022)	Classification	Prompting (output)	Wikipedia CC
REPLUG (Shi et al., 2023)	Knowledge-intensive	Prompting (input)	Wikipedia CC

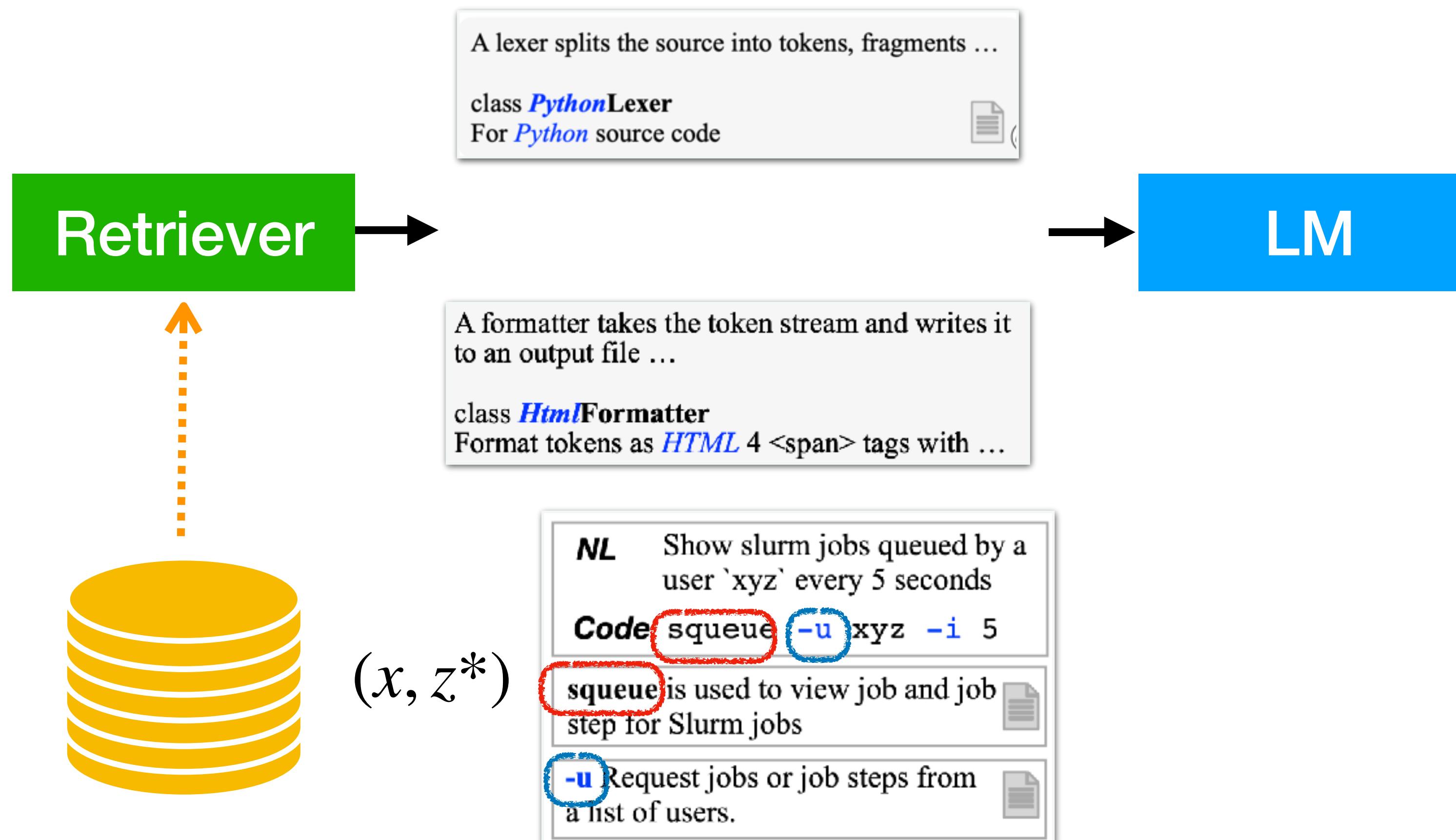
What can be other types of datastores?

DocPrompting (Zhou et al., 2023)

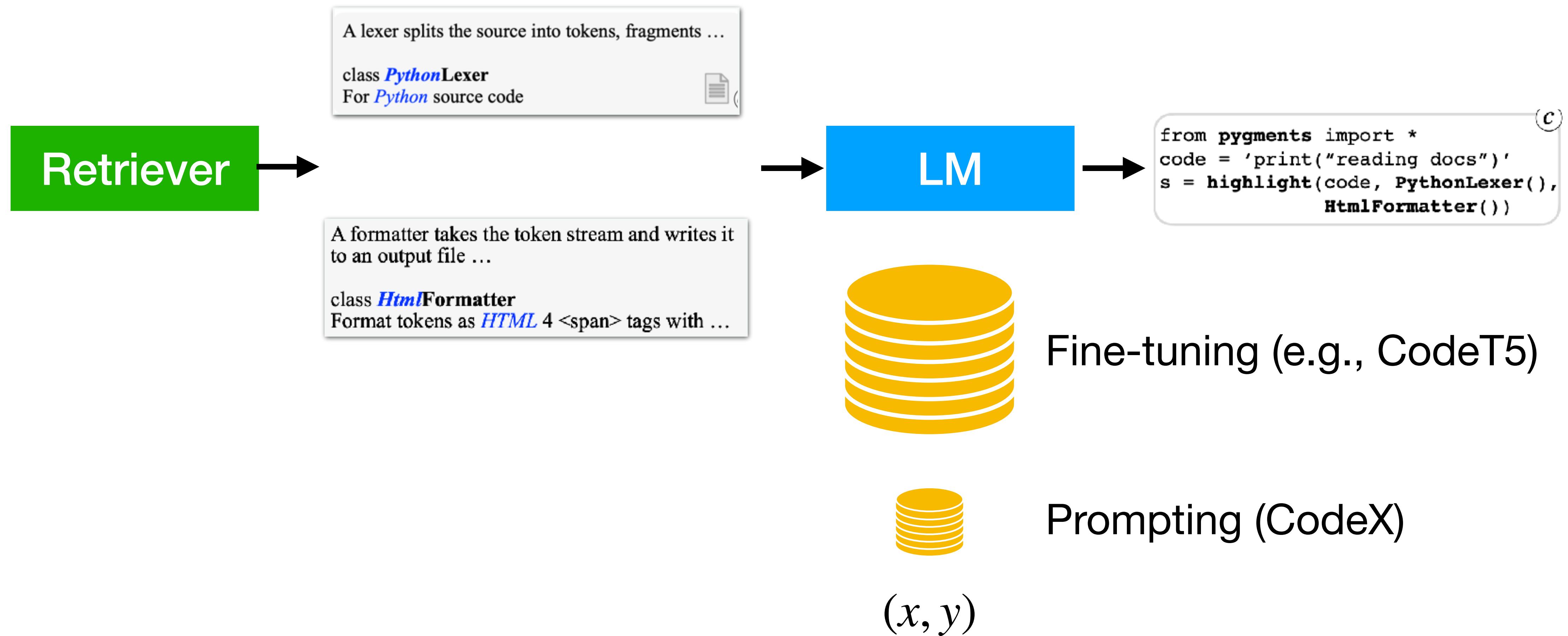


Retrieve **code documentations** about related functions

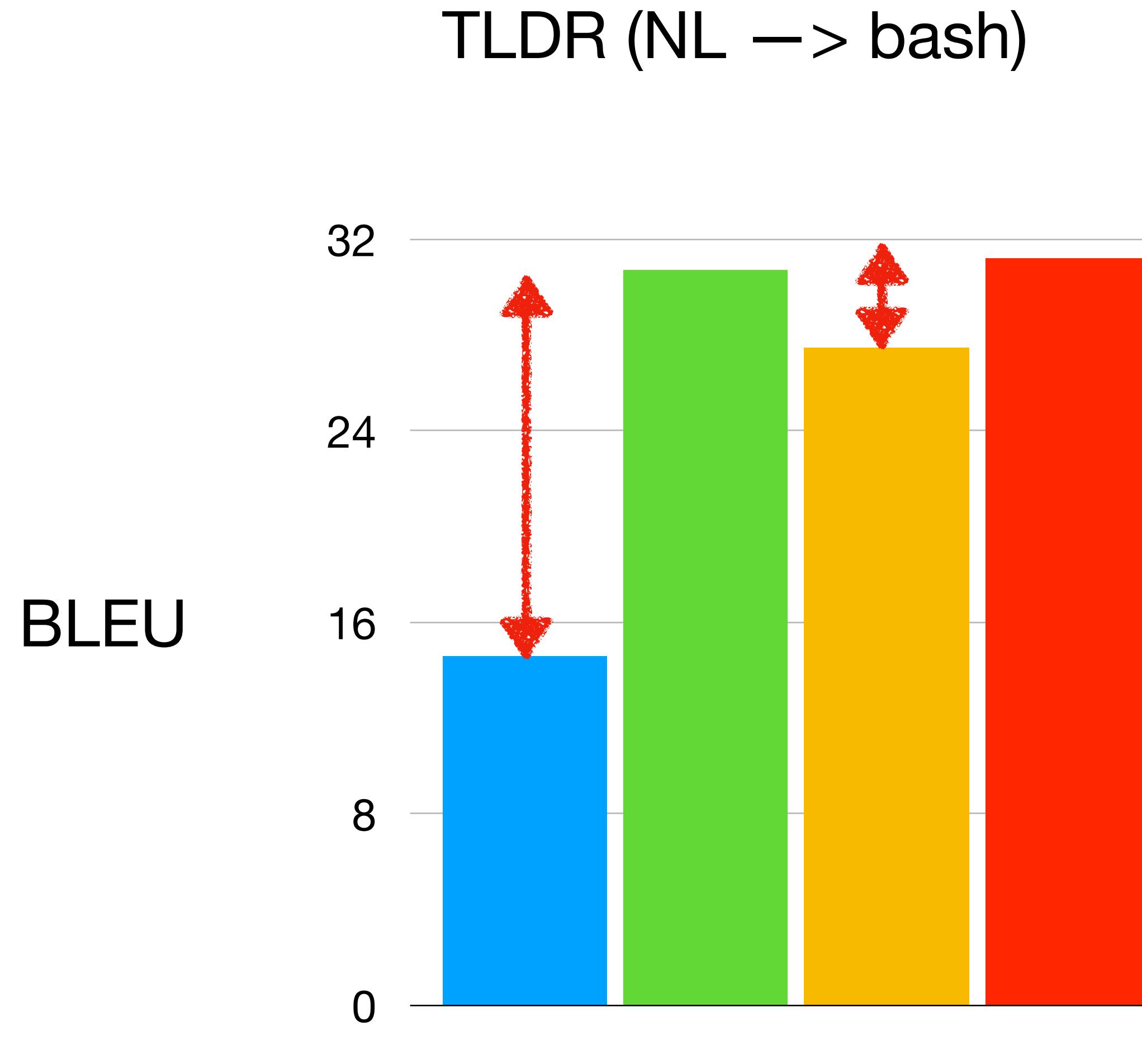
DocPrompting (Zhou et al., 2023)



DocPrompting (Zhou et al., 2023)

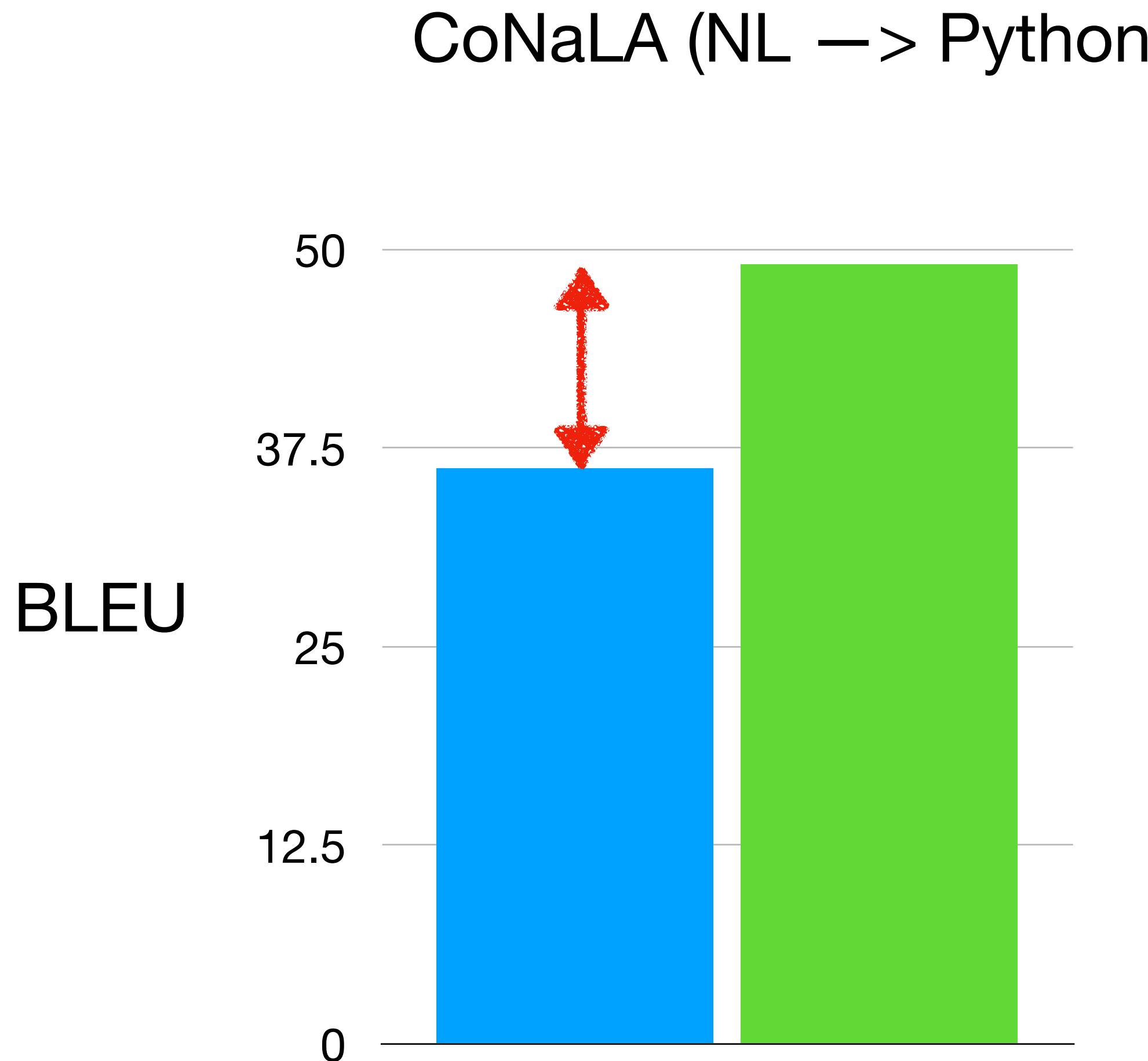


DocPrompting (Zhou et al., 2023)



Large gain given by DocPrompting
for both CodeT5 & CodeX

DocPrompting (Zhou et al., 2023)



Room for improvement in the retrieval component

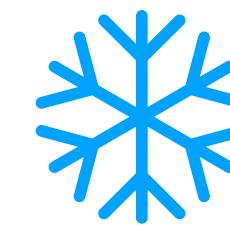
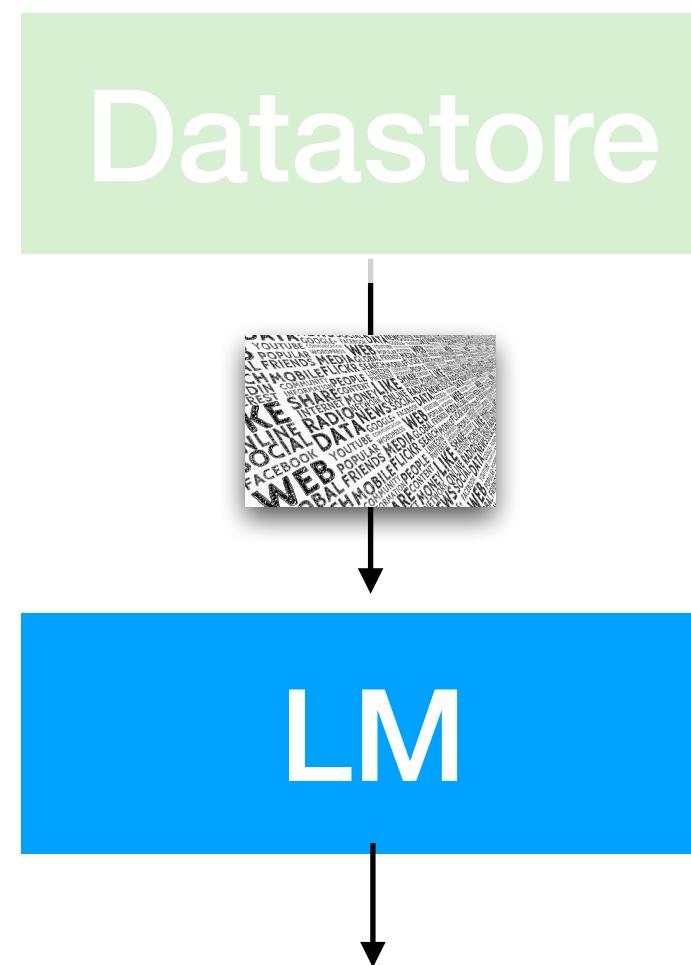
Active research in OOD / Zero-shot retrieval!
(BEIR; Thakur et al., 2021)

- + DocPrompting
- + DocPrompting (Oracle)

Summary of downstream adaptations

	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia CC
GopherCite (Menick et al., 2022)	Open-domain QA, Long-form QA	Fine-tuning + RL (LM)	Google Search Results
kNN-prompt (Shi et al., 2022)	Classification	Prompting (output)	Wikipedia CC
REPLUG (Shi et al., 2023)	Knowledge-intensive	Prompting (input)	Wikipedia CC
DocPrompting (Zhou et al., 2023)	Code Generation	Fine-tuning (DS & LM), Prompting (Input)	Code documentations

How to adapt a retrieval-based LM for a task

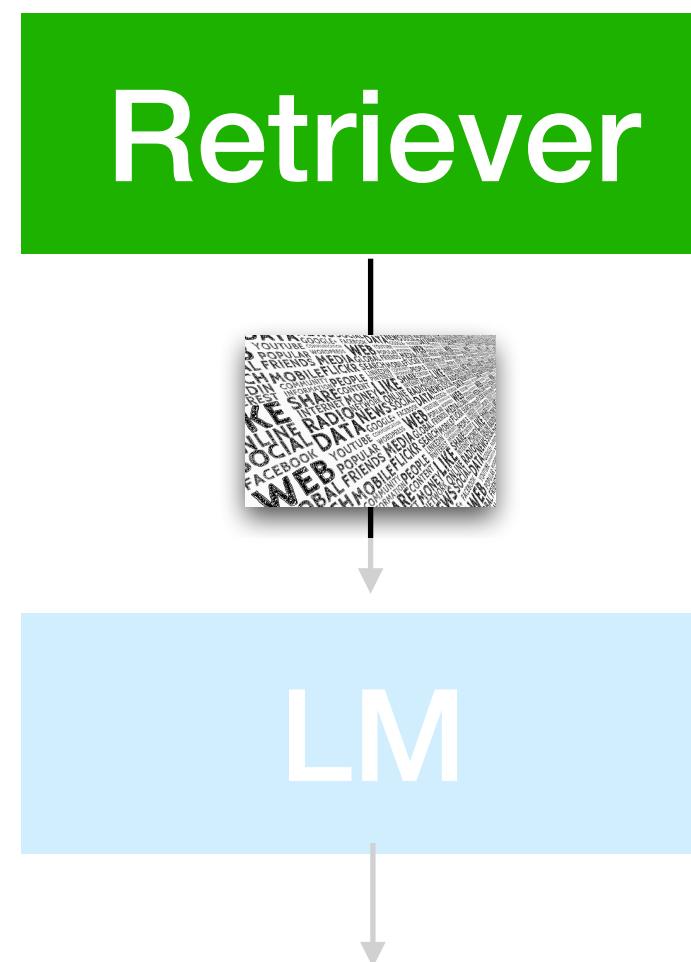


Retrieval-based prompting is easy and simple; no need to train but has higher variance



Fine-tuning (+ RL) requires training but less variance & is competitive with more data

How to adapt a retrieval-based LM for a task



Training a **retriever** on downstream tasks helps

Datastore can be diverse (also in [Section 6](#)) while challenges remain in OOD retrieval

Two key questions for downstream adaptations

How can we adapt a retrieval-based LM for a task?

When should we use a retrieval-based LM?

When to use a retrieval-based LM

Long-tail

knowledge
update

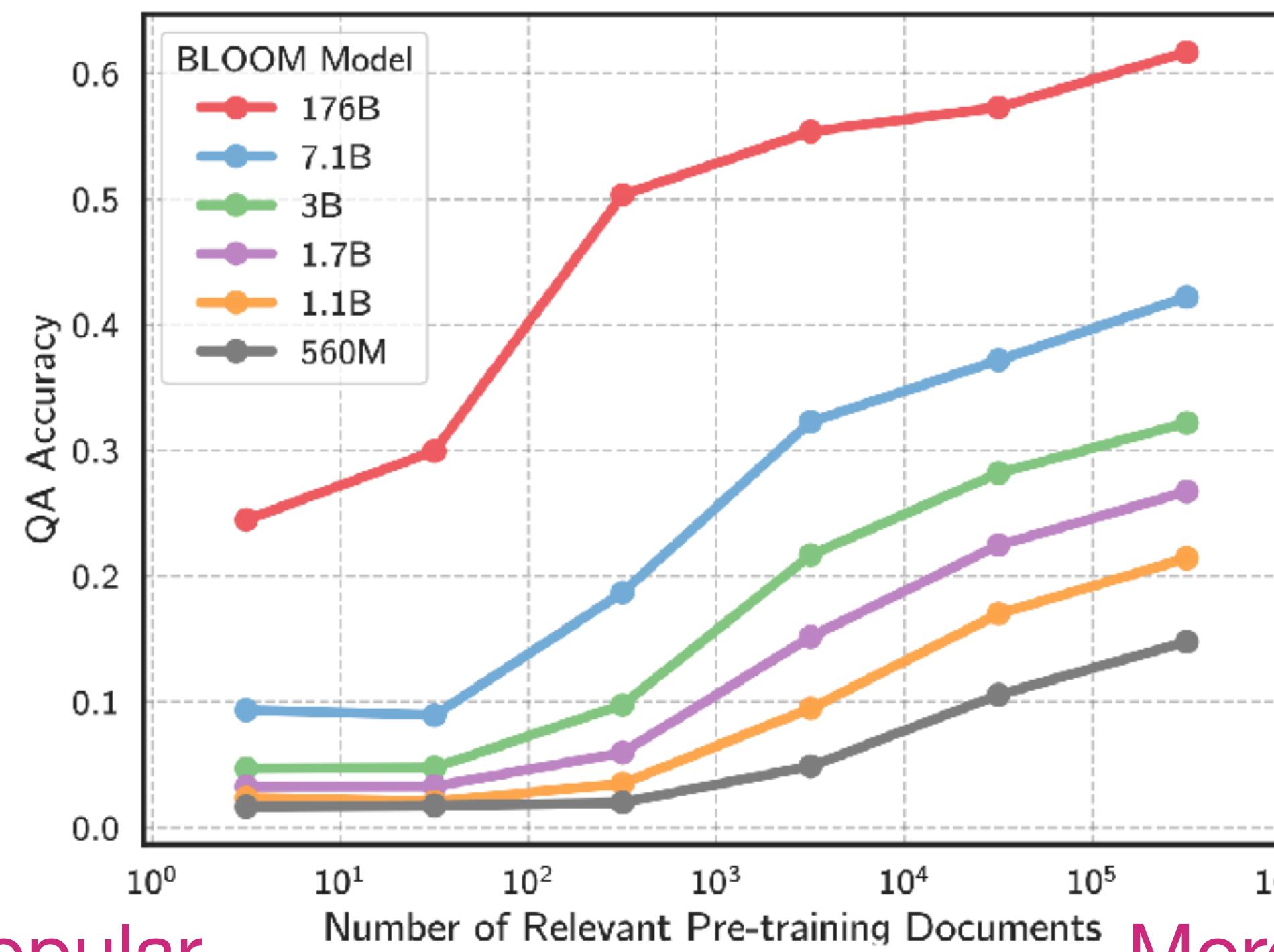
Verifiability

Parameter-
efficiency

Key effectiveness in downstream tasks

Long-tail

LLMs often struggle in **long-tail/less frequent entities**



<— less popular

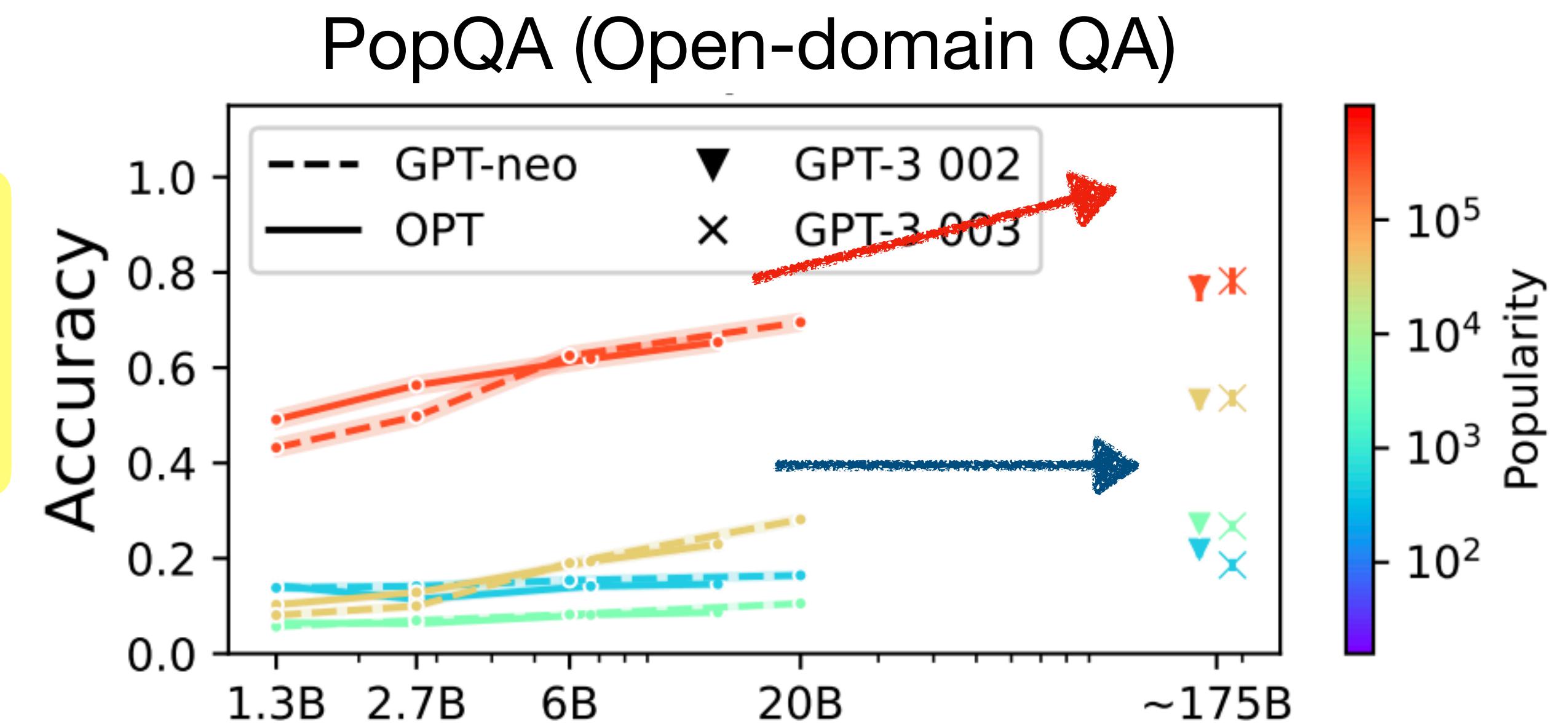
More popular —>

Key effectiveness in downstream tasks

Long-tail

Scaling LLMs only helps for **popular knowledge**; for long tail, scaling gives marginal performance improvements

Performance on less popular questions (blue) doesn't improve over scale

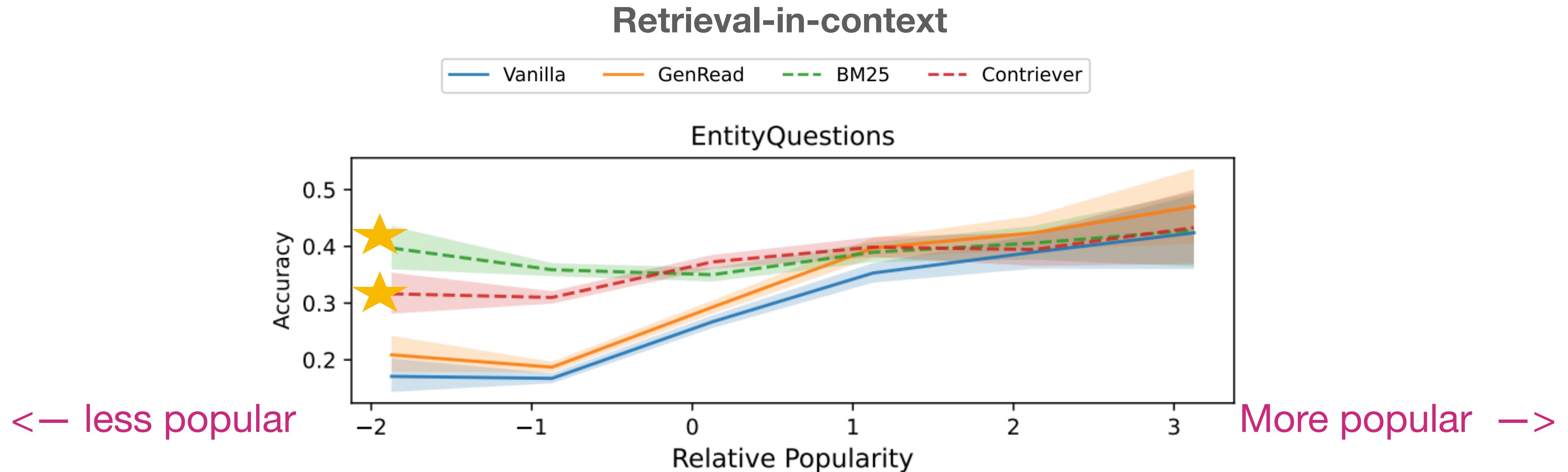


Mallen* and Asai* et al. 2023. “When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories”

Key effectiveness in downstream tasks

Long-tail

Retrieval gives large performance gain in such **long-tail**



Mallen* and Asai* et al. 2023. “When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories”

Key effectiveness in downstream tasks

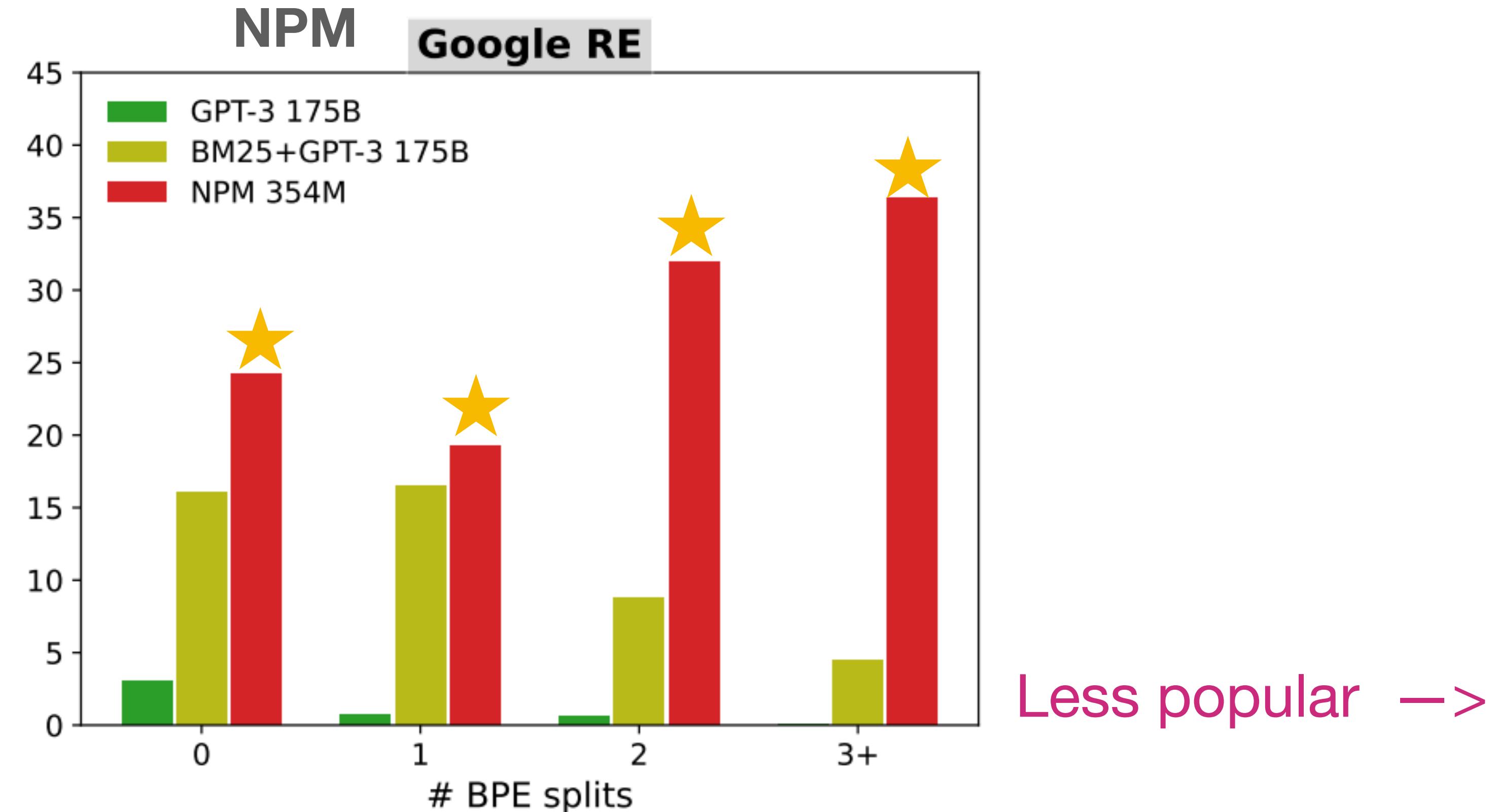
Long-tail

Retrieval gives large performance gain in such **long-tail**

Google RE

Joshua Mathiot died in [MASK].

<— more popular

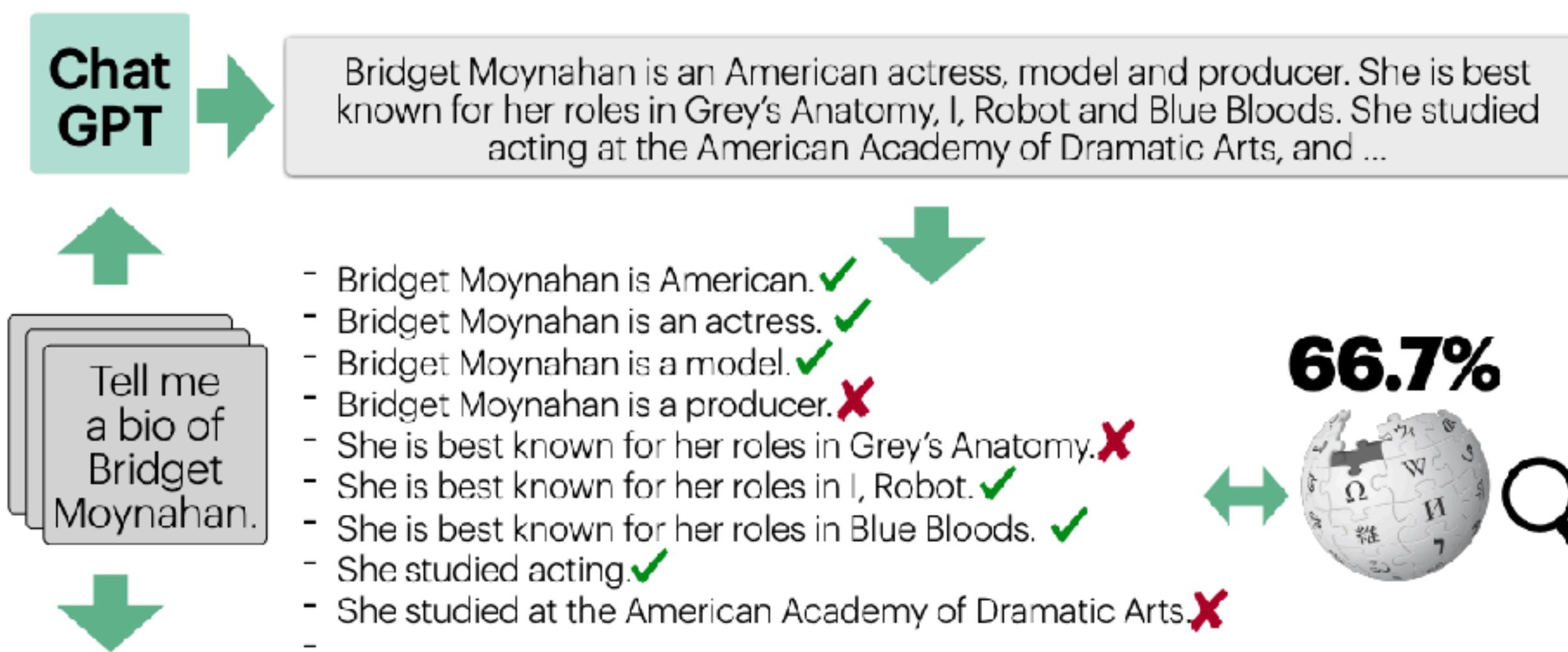


Less popular —>

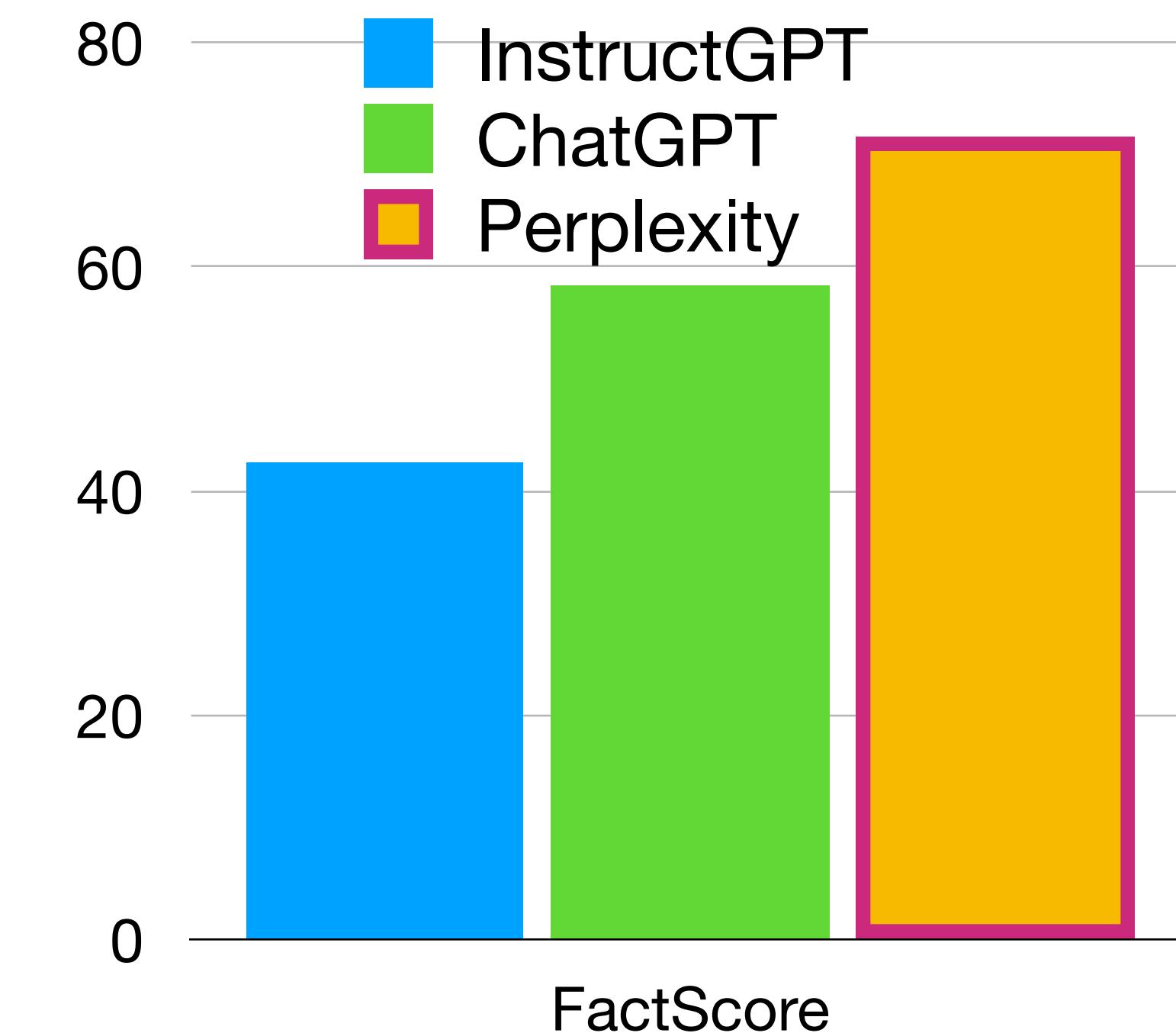
Key effectiveness in downstream tasks

Long-tail

Largely reduce hallucinations in **long-form generations**



FactScore



Key effectiveness in downstream tasks

Update

Standard LLMs need to be **trained again** to adapt to evolving world knowledge

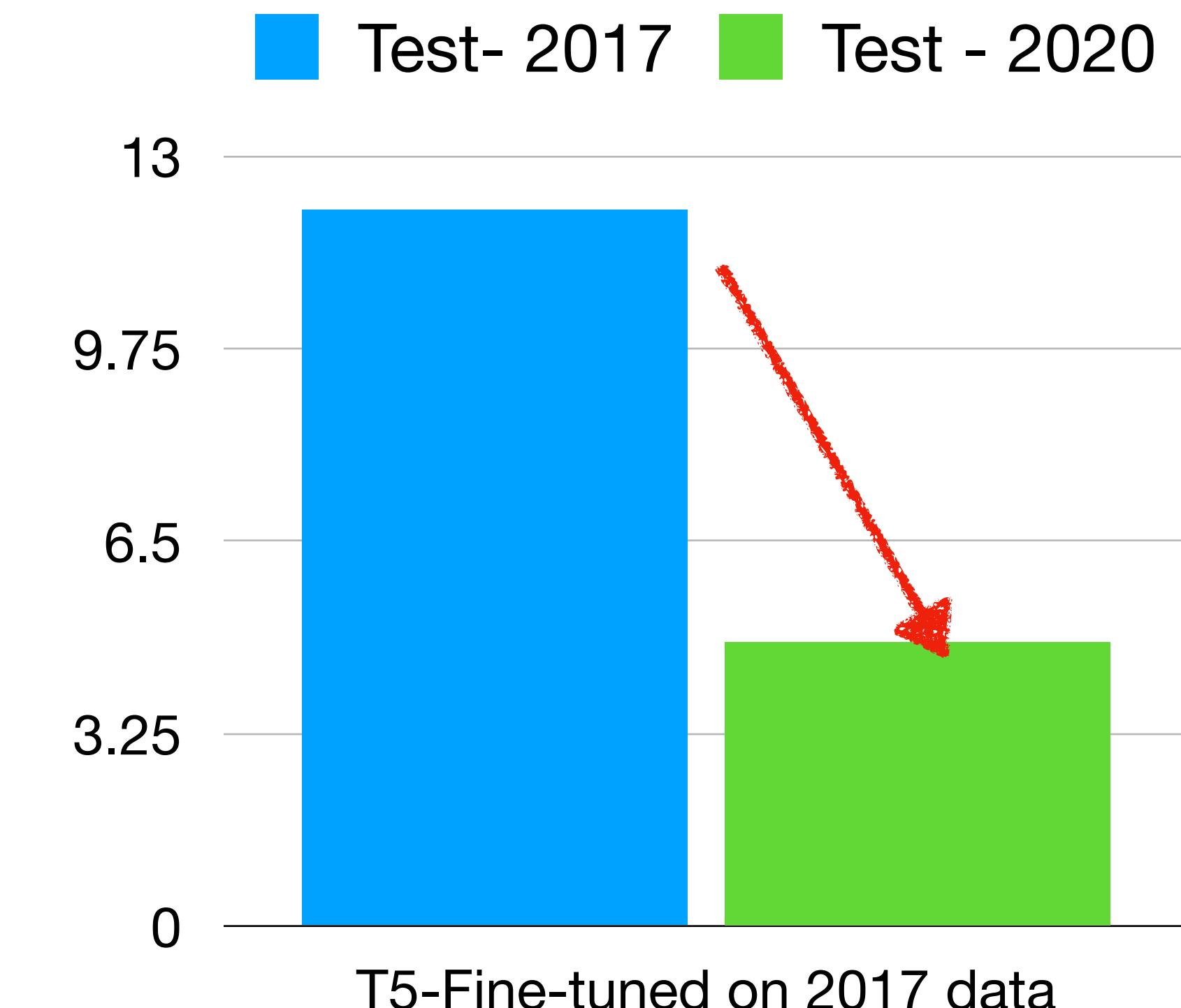
Temp LAMA

2012	Cristiano Ronaldo plays for _X_.	Real Madrid
2019	Cristiano Ronaldo plays for _X_.	Juventus FC

Huge performance drop when test knowledge needs to be updated

Izacard et al. 2022. “Few-shot learning with retrieval augmented language models”

Dhingra et al. 2022. “Time-Aware language models as temporal knowledge bases”



Key effectiveness in downstream tasks

Update

Swapping the knowledge corpus to **accommodate temporal changes** without additional training.

Temp LAMA

2012	Cristiano Ronaldo plays for _X_.	Real Madrid
2019	Cristiano Ronaldo plays for _X_.	Juventus FC

Swapping test datastore only retains strong performance



Izacard et al. 2022. "Few-shot learning with retrieval augmented language models"

Dhingra et al. 2022. "Time-Aware language models as temporal knowledge bases"

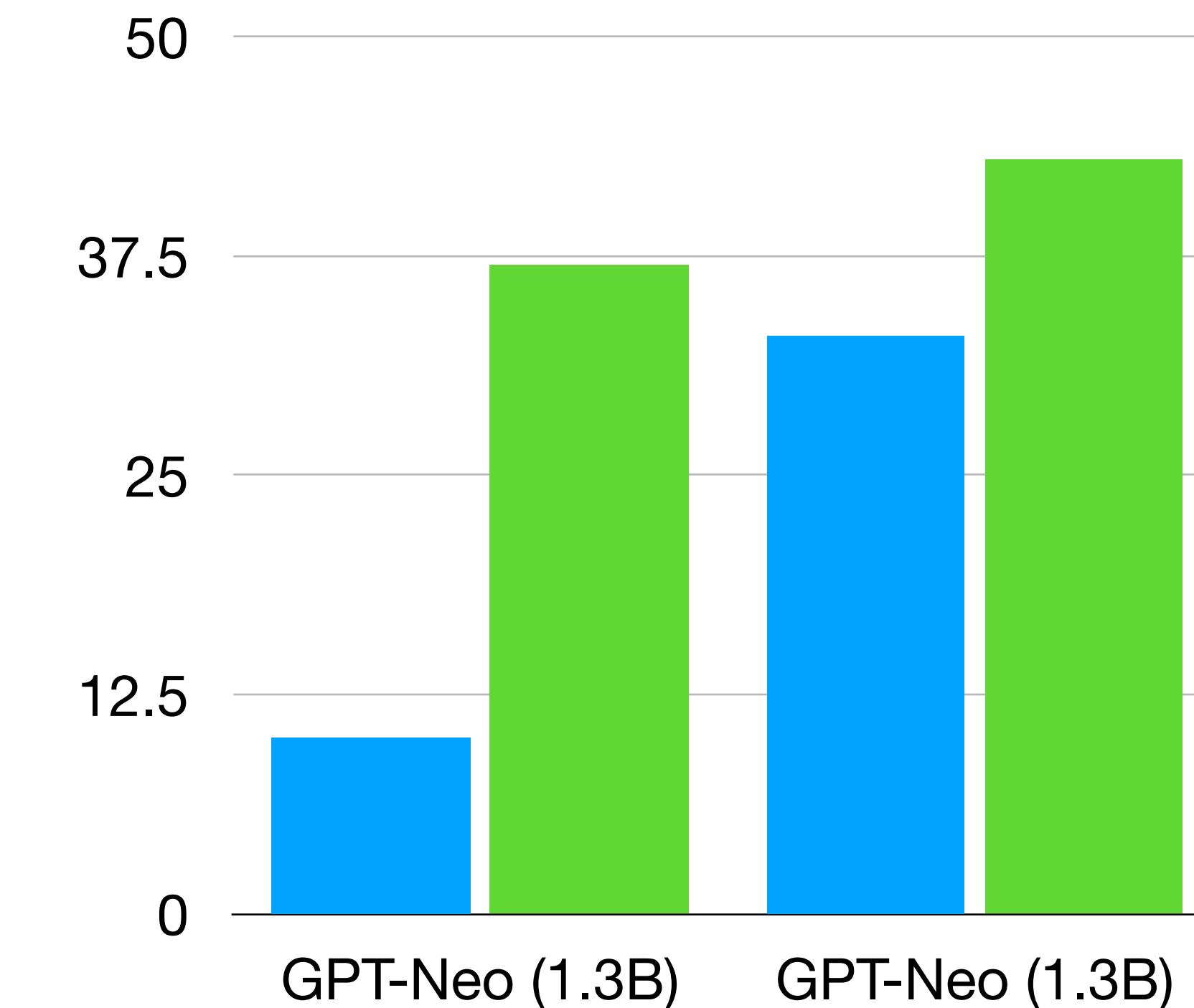
Key effectiveness in downstream tasks

Parameter-
efficiency

Much smaller LMs with retrieval can outperform much larger LMs in fact completions.

Retrieval + GPT-Neo 1.3B outperforms vanilla GPT3 on PopQA

w/o retrieval
w/ Contriever



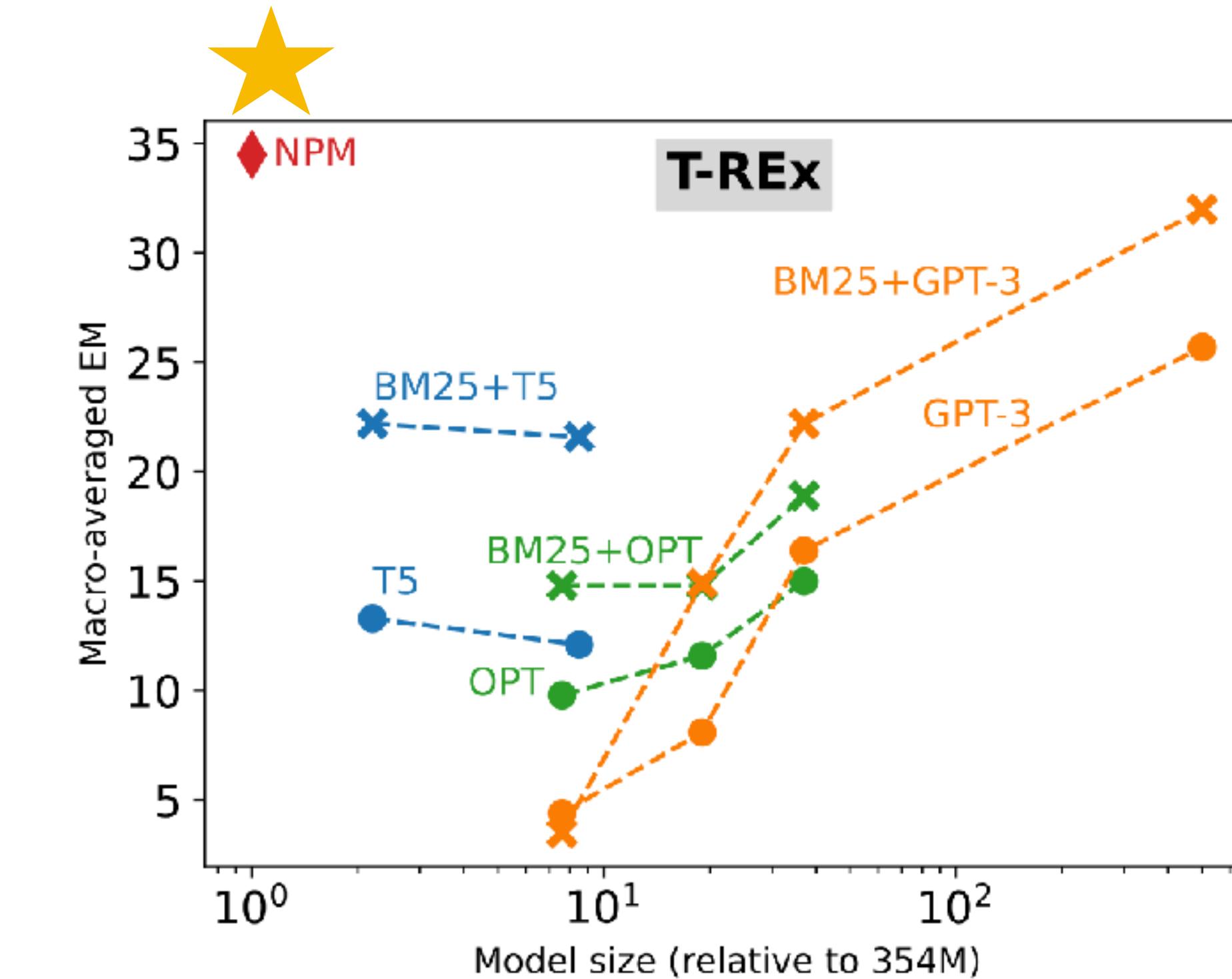
Mallen* and Asai* et al. 2023. “When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories”

Key effectiveness in downstream tasks

Parameter-
efficiency

NPM (354 M) outperforms
GPT-3 on T-Rex.

Much smaller LMs with retrieval can outperform much larger LMs in fact completions.



Min et al. 2023. “Nonparametric Masked Language Modeling”
Petroni et al. 2019. “Language Models as Knowledge Bases?”

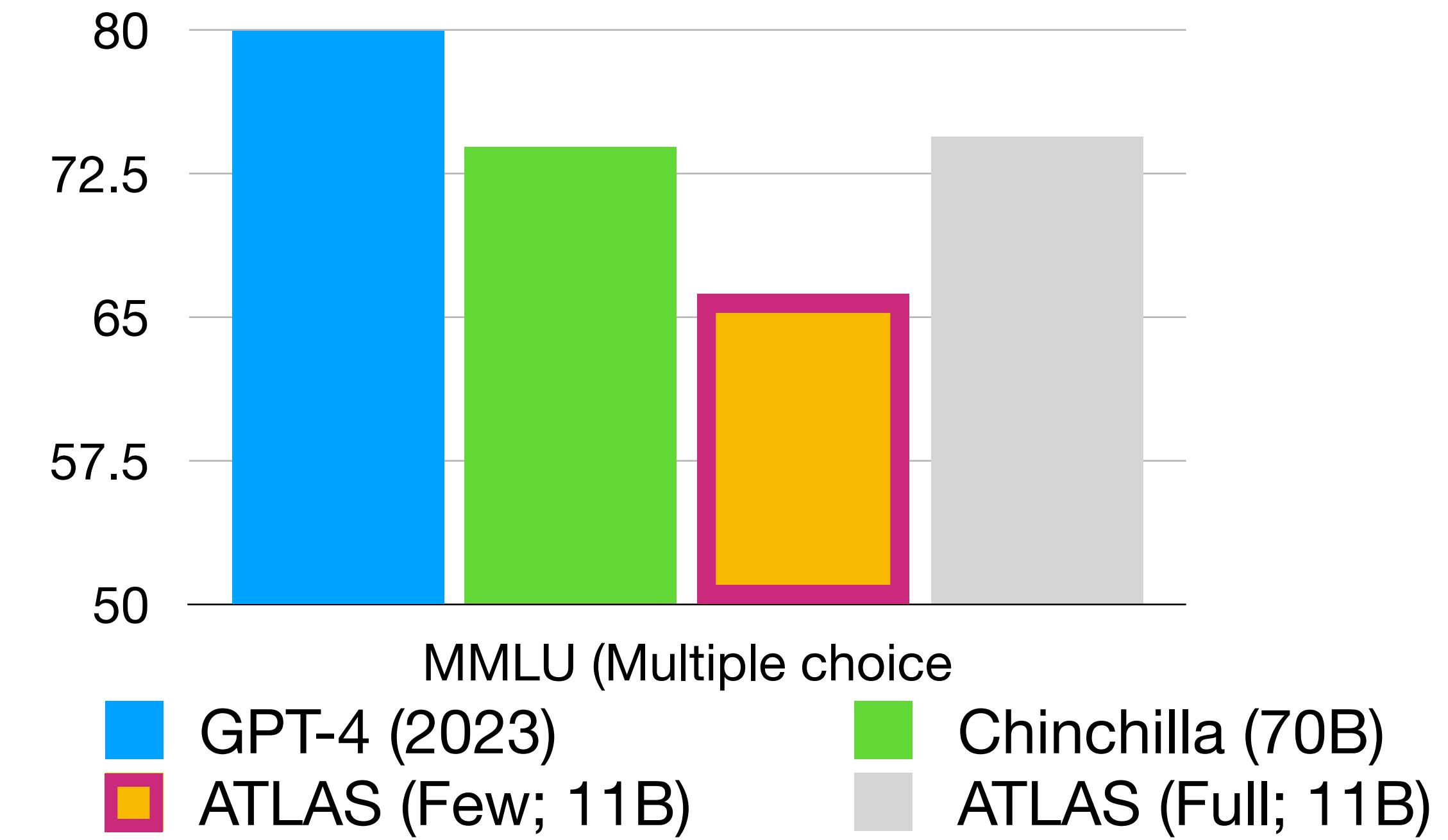
Key effectiveness in downstream tasks

Parameter-
efficiency

Much smaller LMs with retrieval can outperform much larger LMs in fact completions.

Room for improvements for diverse task adaptations!

Izacard et al. 2022. “Few-shot learning with retrieval augmented language models”



Key effectiveness in downstream tasks

Verifiability

Human and model can reliably assess the **factuality of the generations** using the retrieved evidence.

Why is it sometimes hard to eat after not eating for a while?

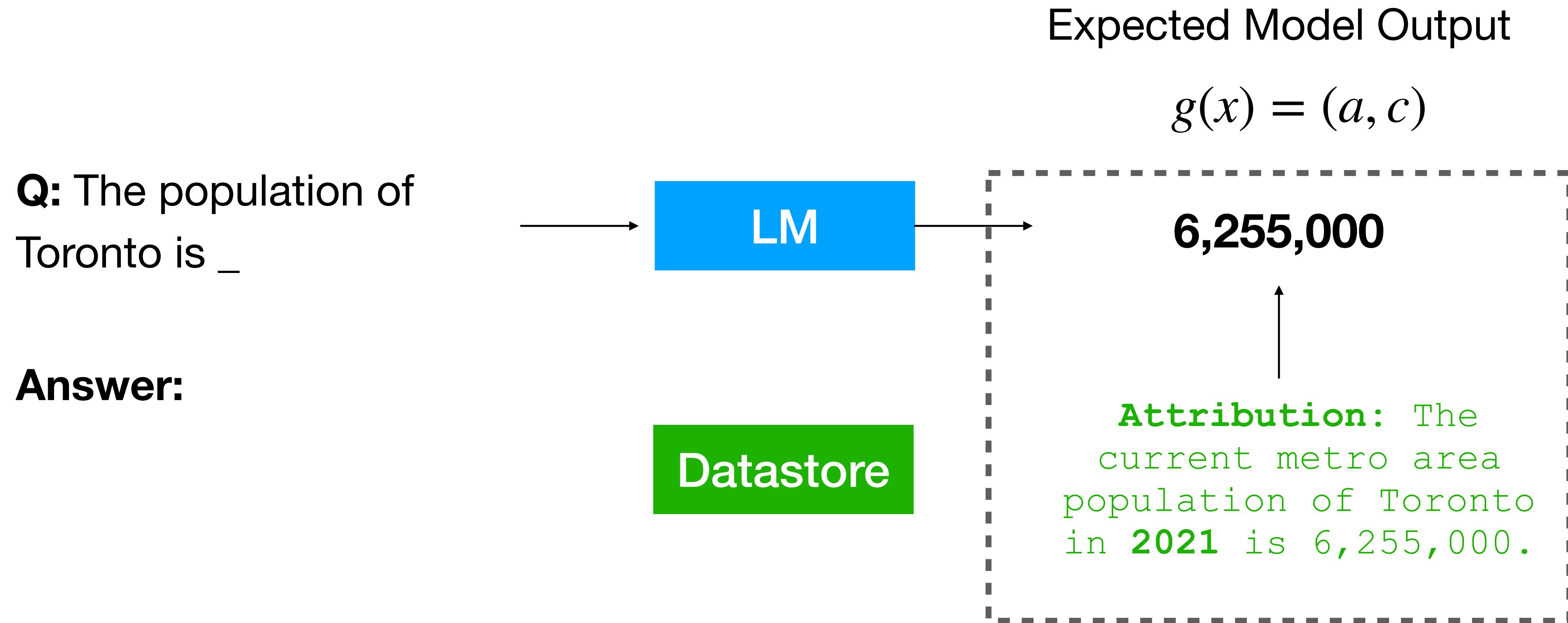
175B best-of-64

There are several reasons why not eating for long periods of time can make you lose your appetite. One is that not eating triggers the same response as starvation, adrenaline drives a process called lipolysis, which is the process your body undertakes when it wants to break down your fat reserves for energy, and also prevents you from burning through your muscle [1][3]. Another reason is that not eating means your body doesn't get the sugars it needs to operate properly, which causes a process called gluconeogenesis [2]. Also, leptin levels can rapidly decline in the fasted state [4], which may cause an increase in appetite.

1. Why Don't You Feel Hungry After Not Eating For a Long Time? – Neuro (getneuro.com) ↗
2. Why Don't You Feel Hungry After Not Eating For a Long Time? – Neuro (getneuro.com) ↗
3. Why Don't You Feel Hungry After Not Eating For a Long Time? – Neuro (getneuro.com) ↗
4. Fasting may change the body's hunger response - here's what to do about it - Diabetes (www.diabetes.co.uk) ↗

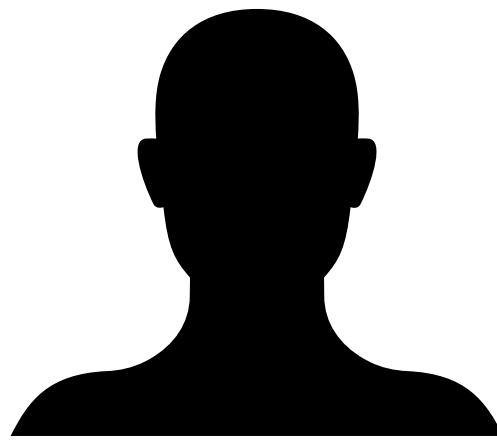
Nakano et al. 2021. “WebGPT: Browser-assisted question-answering with human feedback”

Attributions: AttributedQA (Bohnet et al., 2022)

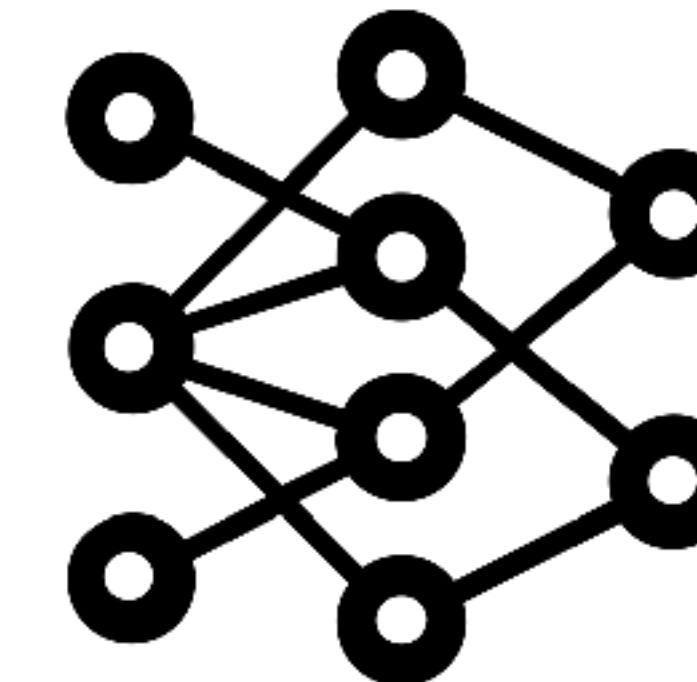


Attributions: AttributedQA (Bohnet et al., 2022)

Human Evaluation (AIS)



Automatic Evaluation (AutoAIS)



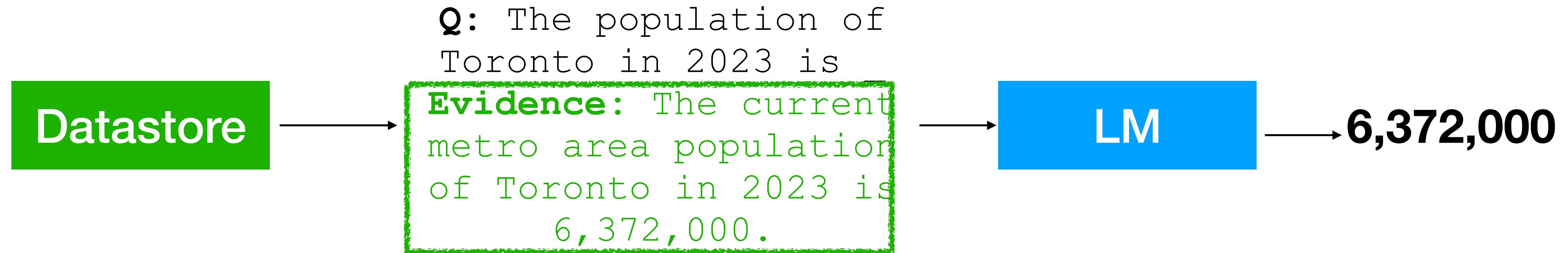
NLI model

1. Are all (a,c) interpretable?
2. Is any information in a supported by c?

$$E^A[g] = \frac{1}{n} \sum_{i=1}^n \text{AutoAIS}(x_i, g(x_i))$$

AttributedQA (Bohnet et al., 2022)

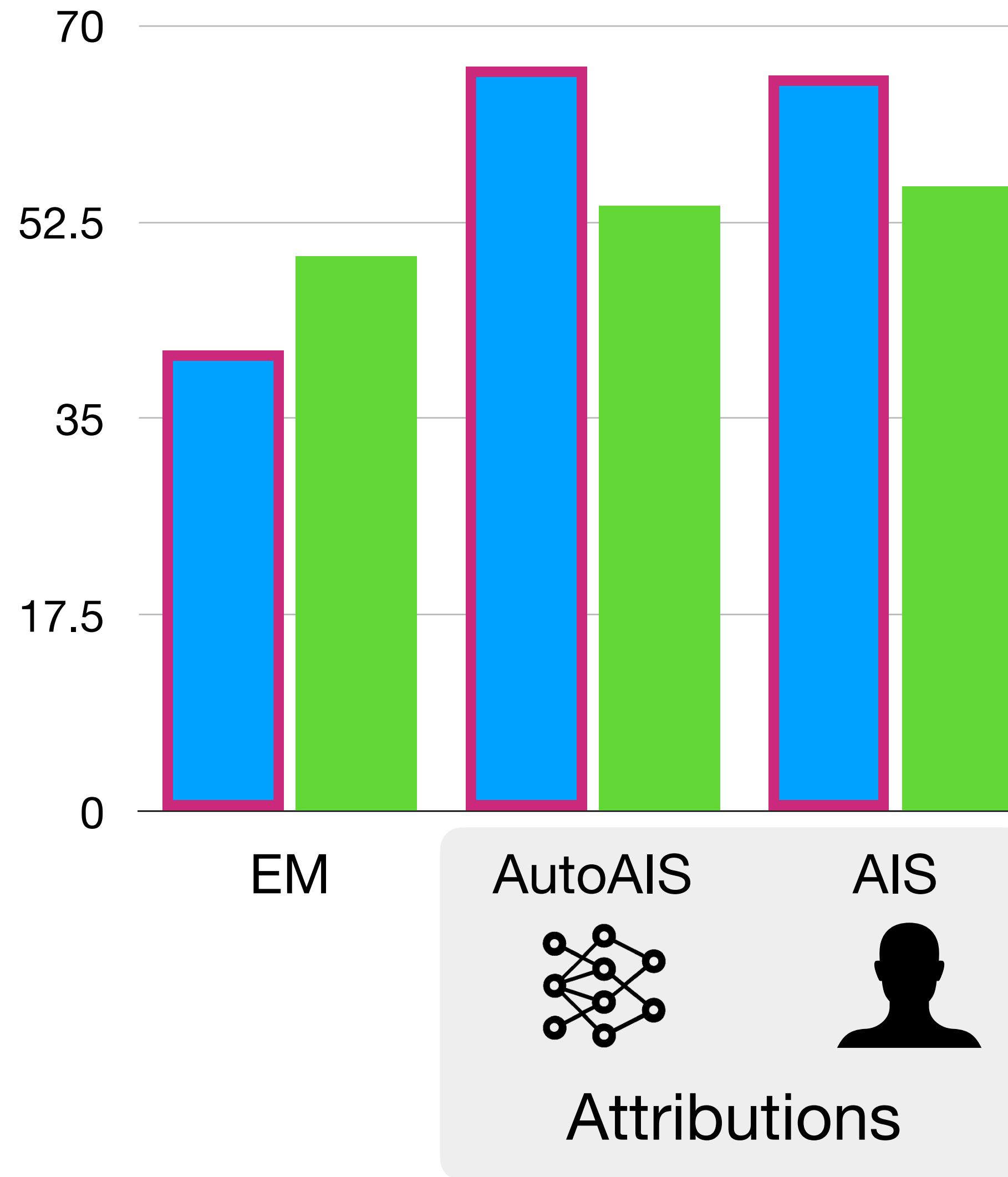
Retrieval-based LM



Post-hoc retrieval



AttributedQA (Bohnet et al., 2022)



Retrieval in context yields higher AIS than post-hoc retrieval

■ Retrieval-based LM ■ Post-hoc retrieval

When to use a retrieval-based LM

Long-tail

knowledge
update

Verifiability

Parameter-
efficiency

When to use a retrieval-based LM

Long-tail

knowledge
update

Verifiability

Parameter-
efficiency

Out of domain adaptations

(Shi et al., 2022; Zheng et al., 2021)

Khandelwal, et al. 2020. “Nearest Neighbor Zero-shot Inference”

Shi et al. 2022. “Nearest Neighbor Zero-shot Inference”

When to use a retrieval-based LM

Long-tail

knowledge
update

Verifiability

Parameter-
efficiency

Out of domain adaptations

(Shi et al., 2022; Zheng et al., 2021)

Privacy

(Huang et al., 2023)

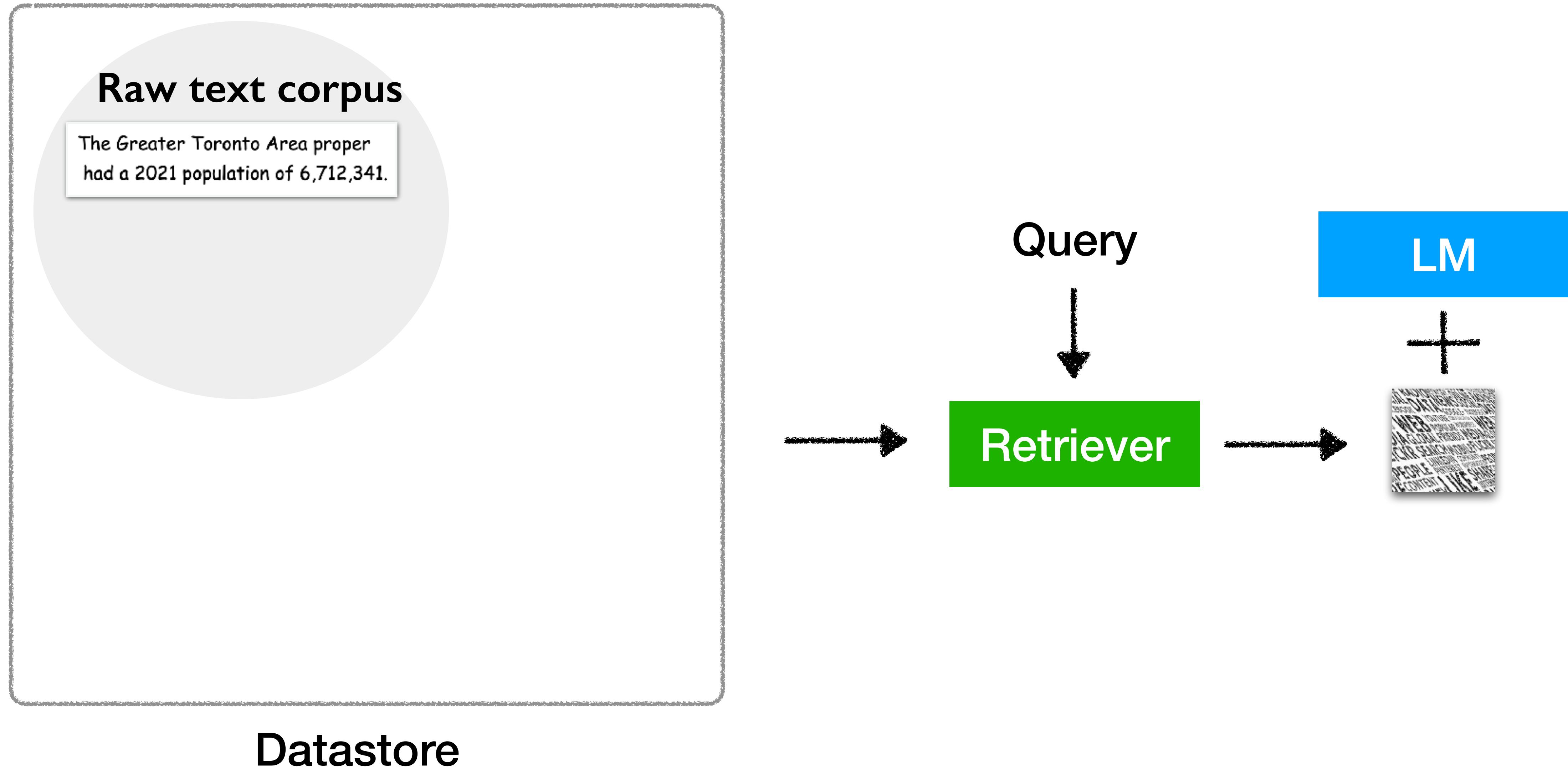
Khandelwal, et al. 2020. “Nearest Neighbor Zero-shot Inference”

Shi et al. 2022. “Nearest Neighbor Zero-shot Inference”

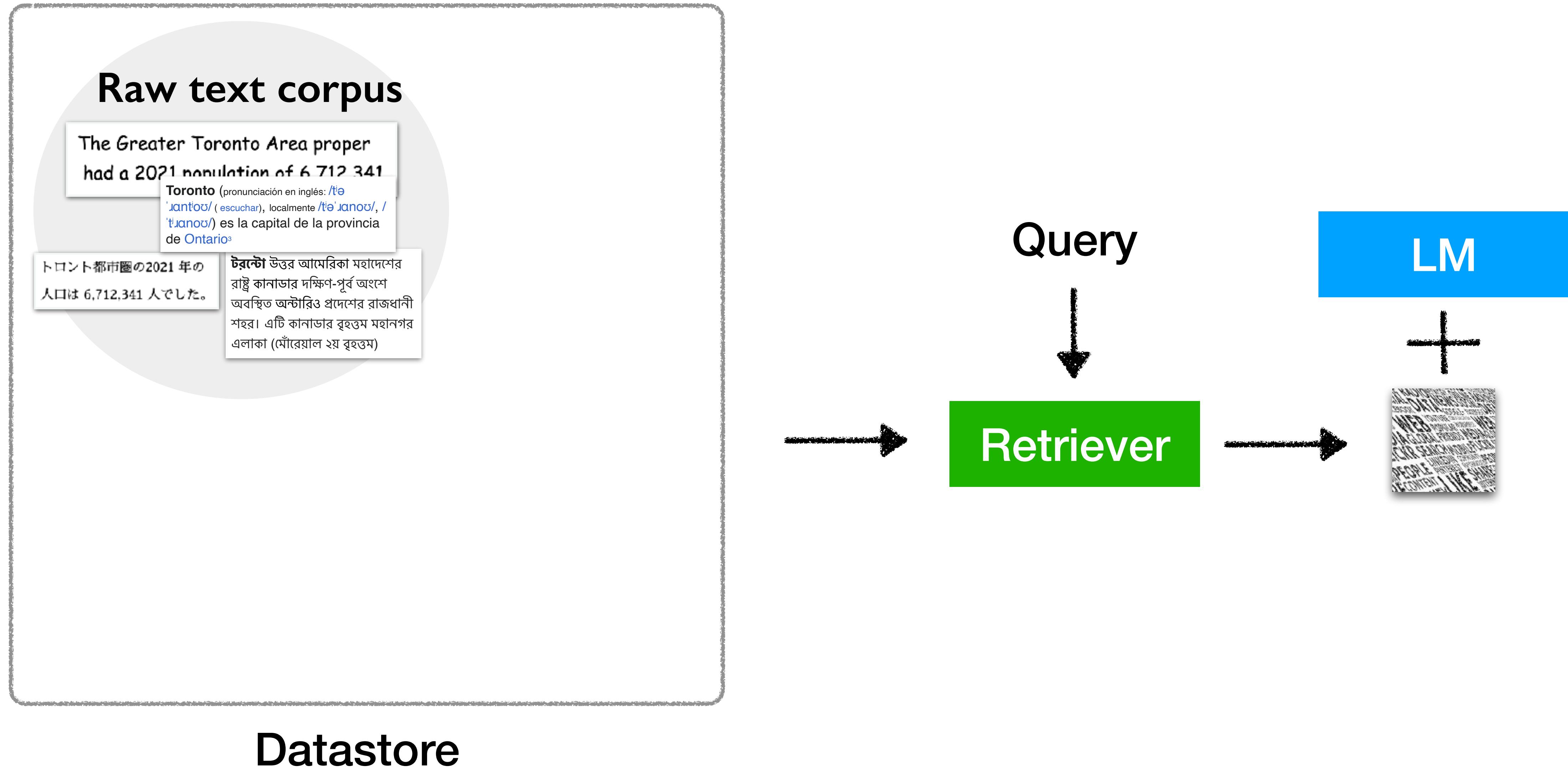
Huang et al. 2023. “Privacy Implications of Retrieval-Based Language Models”

Section 6: Multilingual & Multimodal

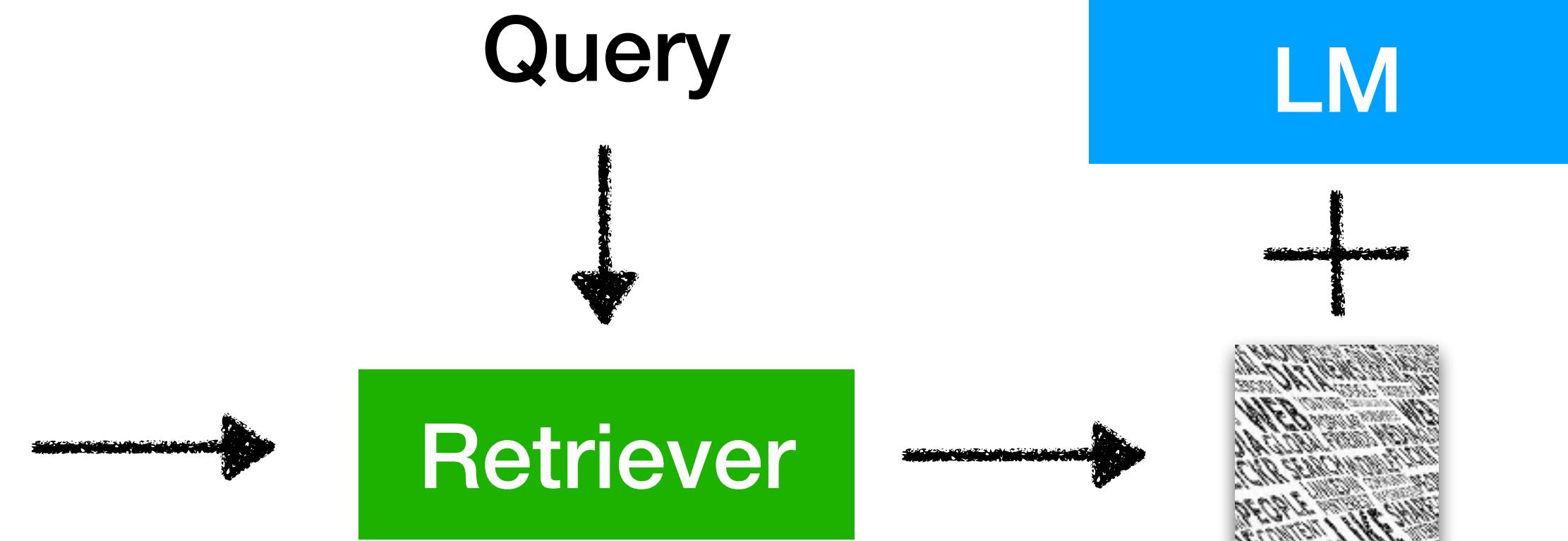
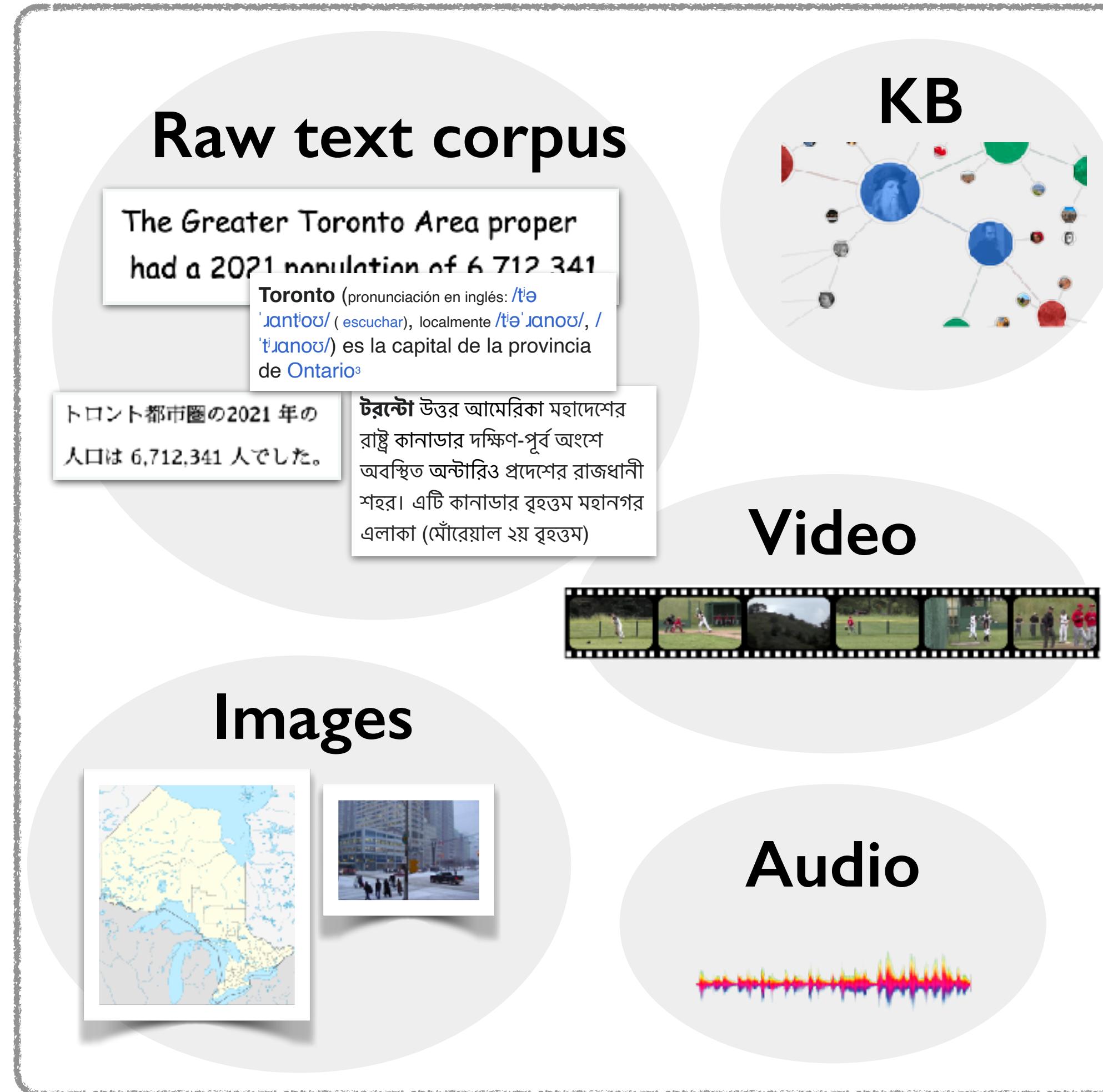
Retrieval-based LM for diverse knowledge sources



Retrieval-based LM for diverse knowledge sources

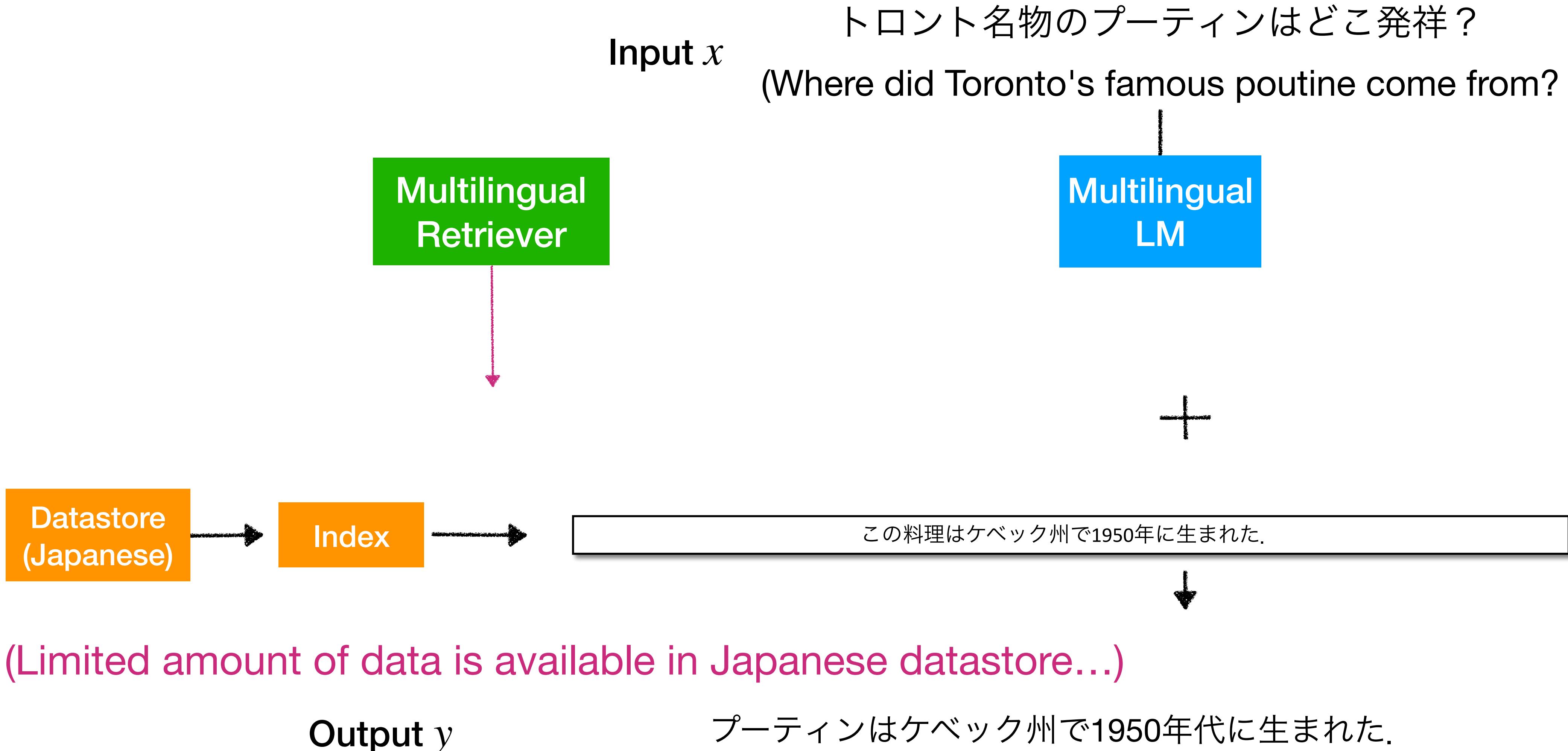


Retrieval-based LM for diverse knowledge sources

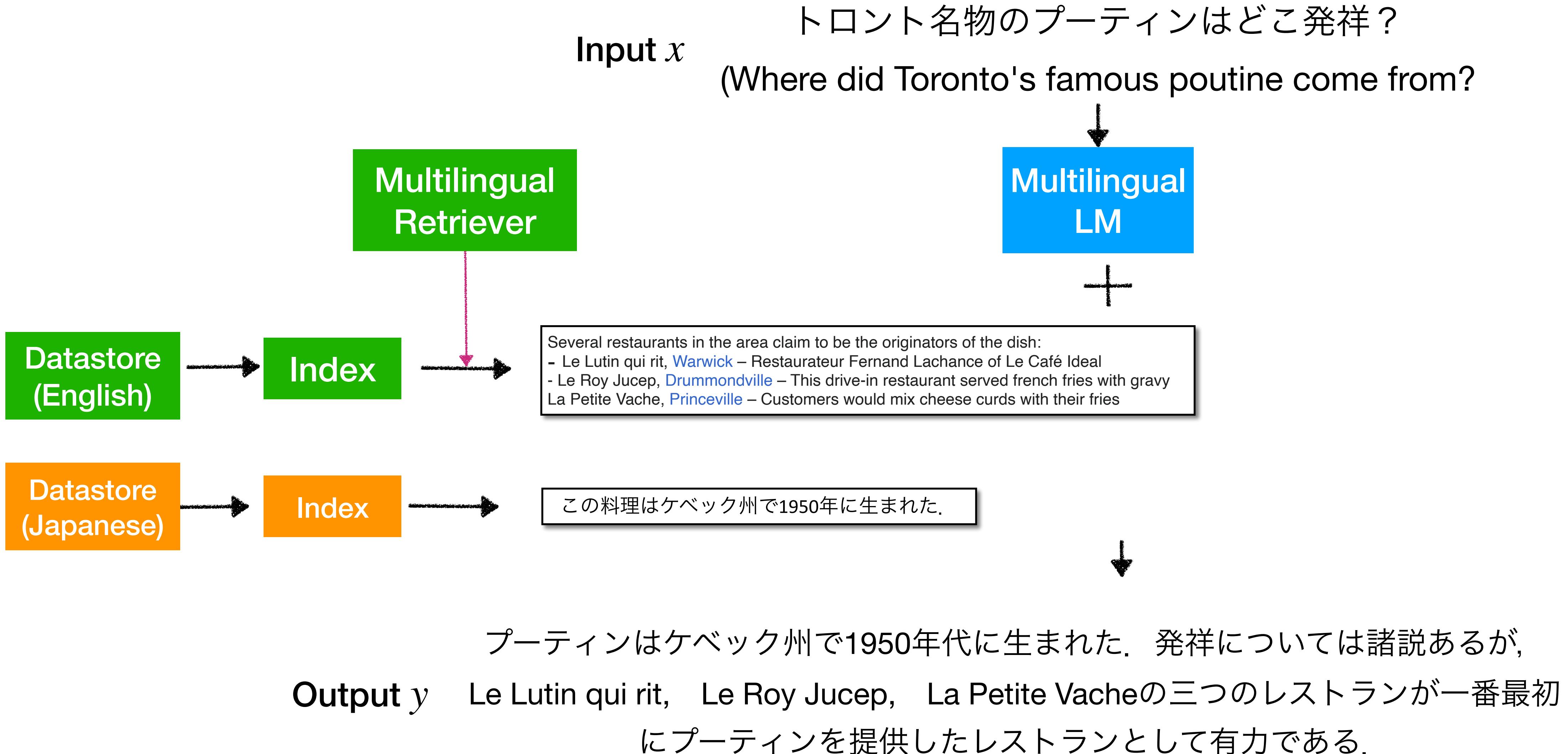


Datastore

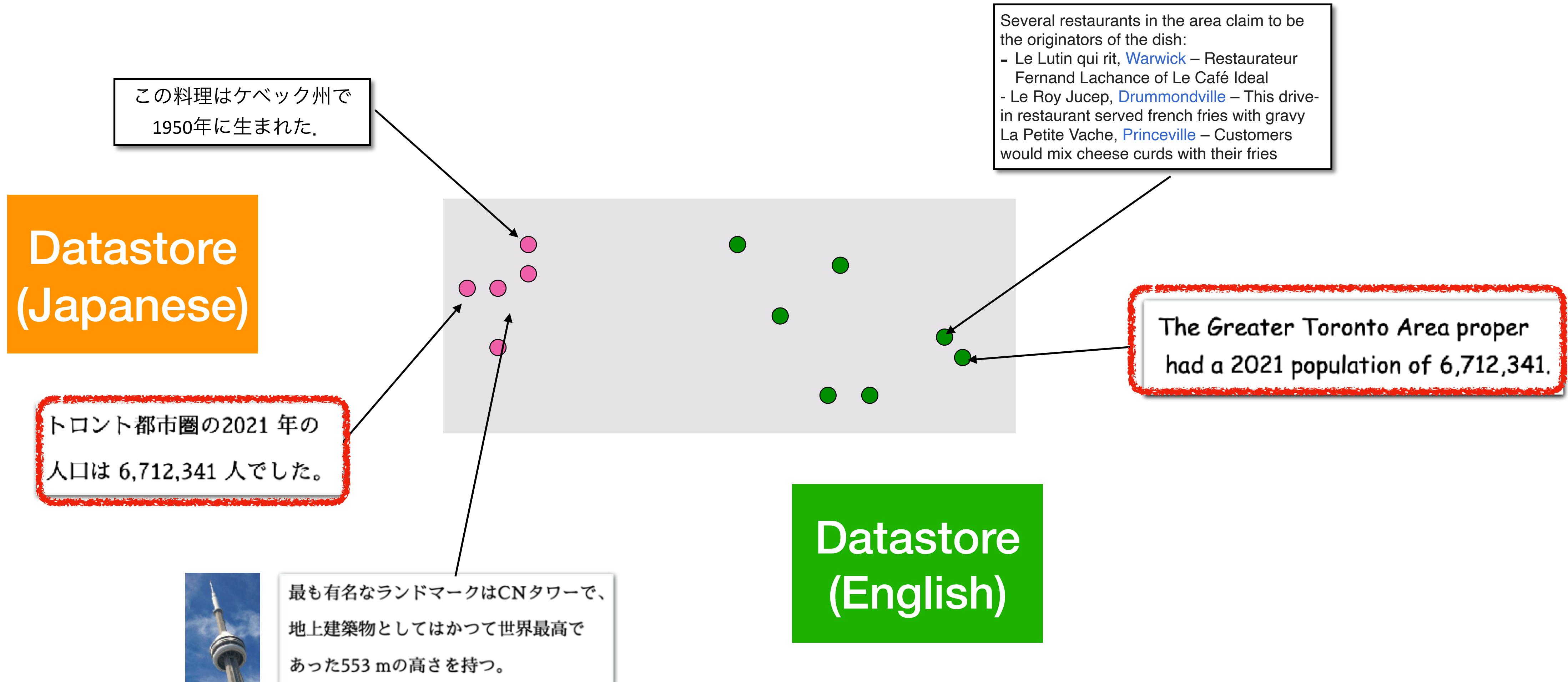
Multilingual Retrieval-based LM



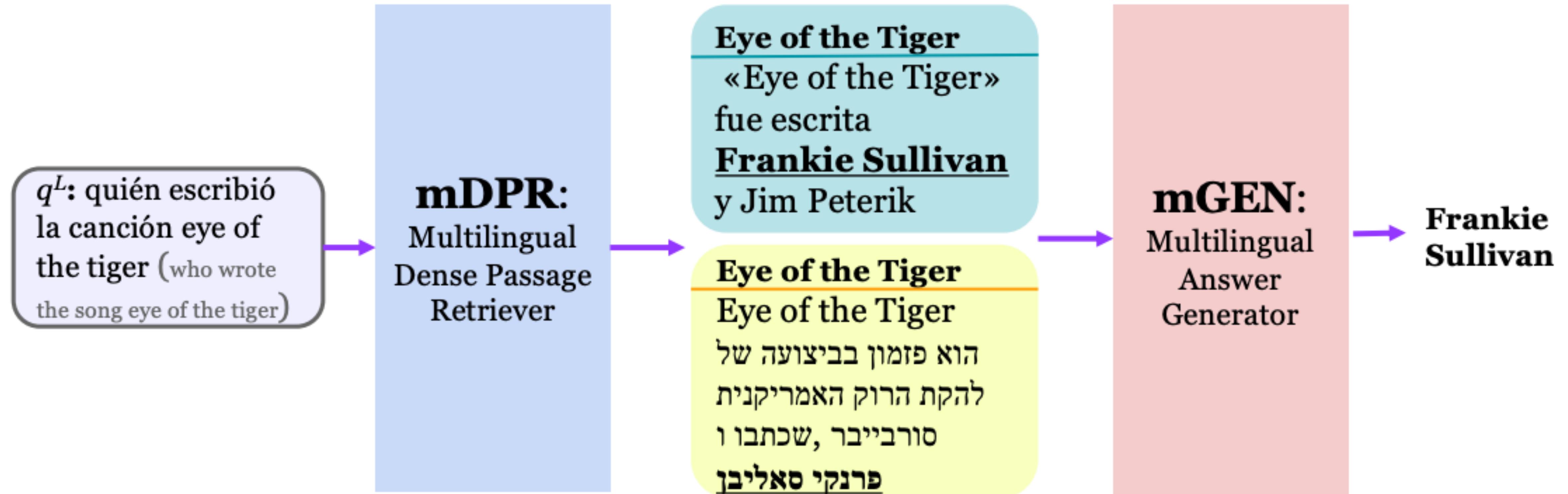
Multilingual Retrieval-based LM



Language biases in representation spaces



CORA (Asai et al., 2021)

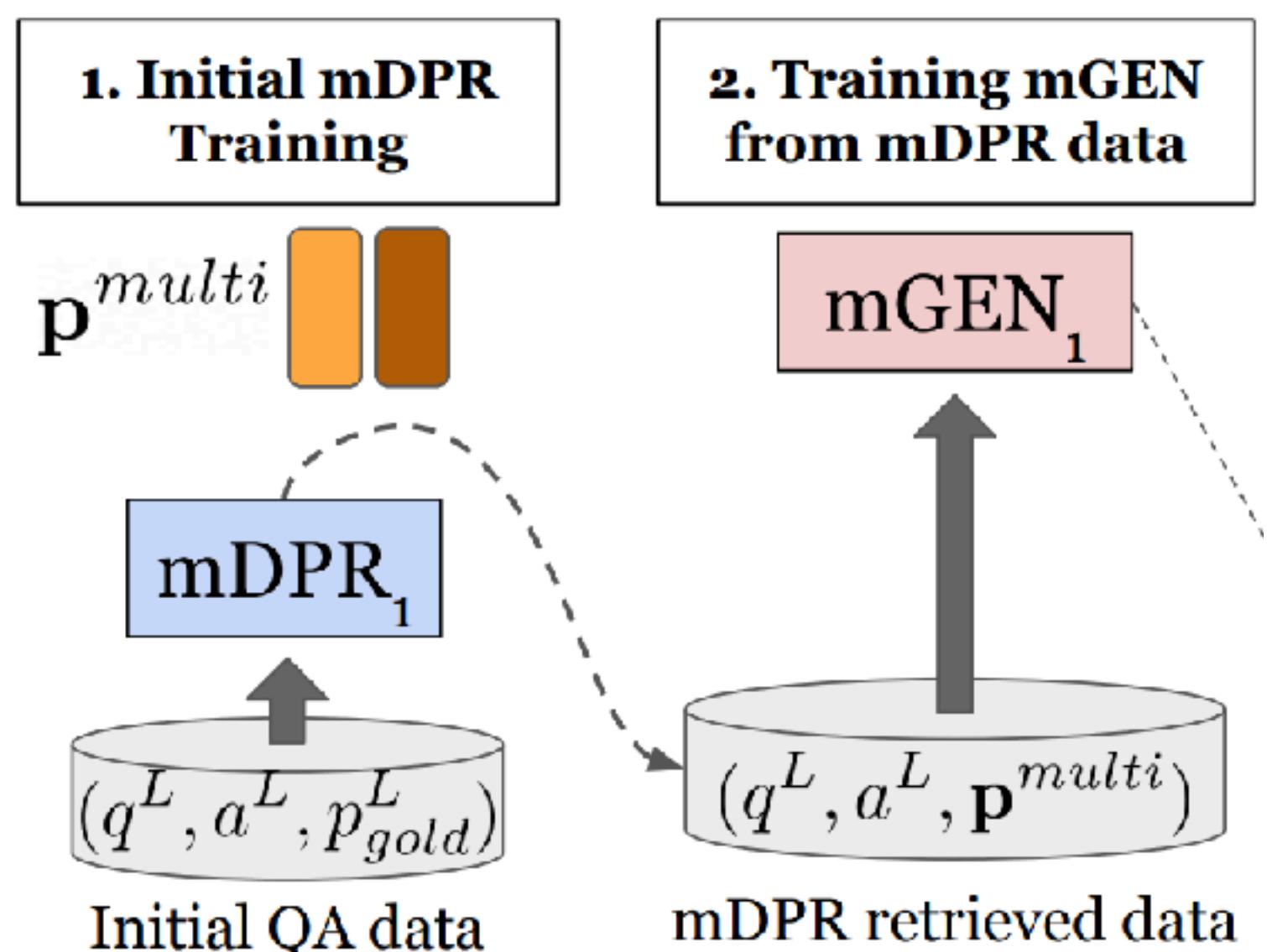


CORA: Iteratively training multilingual LM & retriever

Initial fine-tuning of retriever and LM
using task data

→ Training
- → Inference

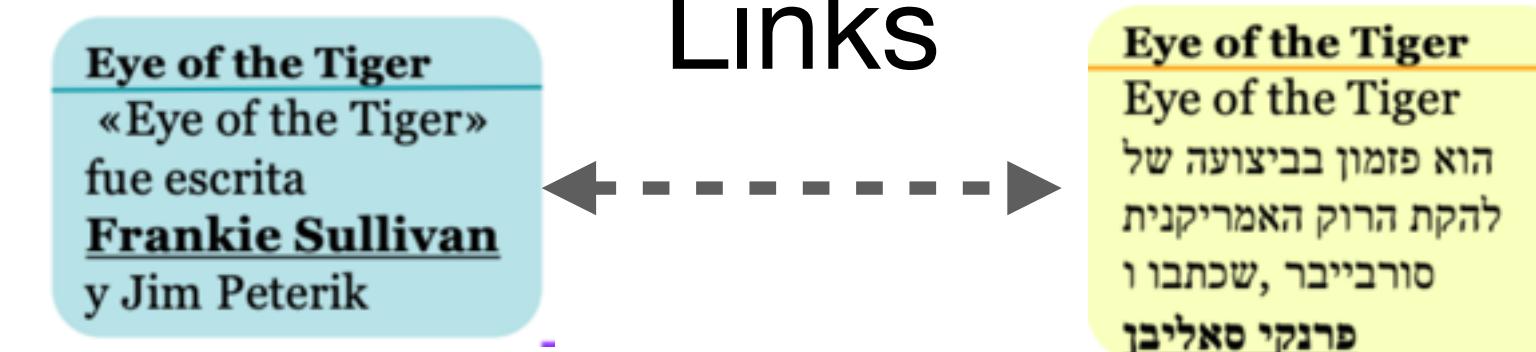
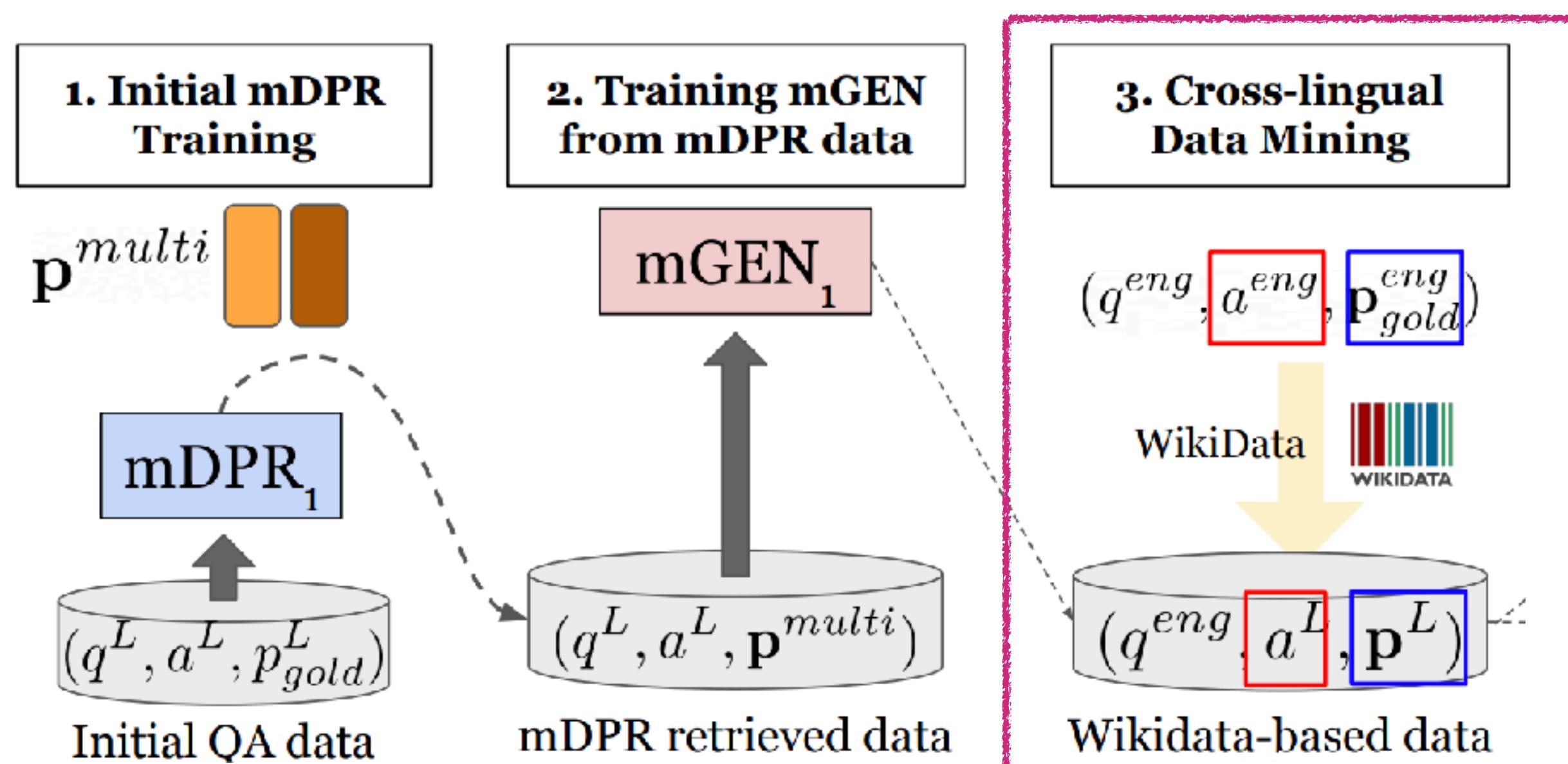
C^{multi}



CORA: Iteratively training multilingual LM & retriever

Retrieve positive paragraphs in other languages using language links

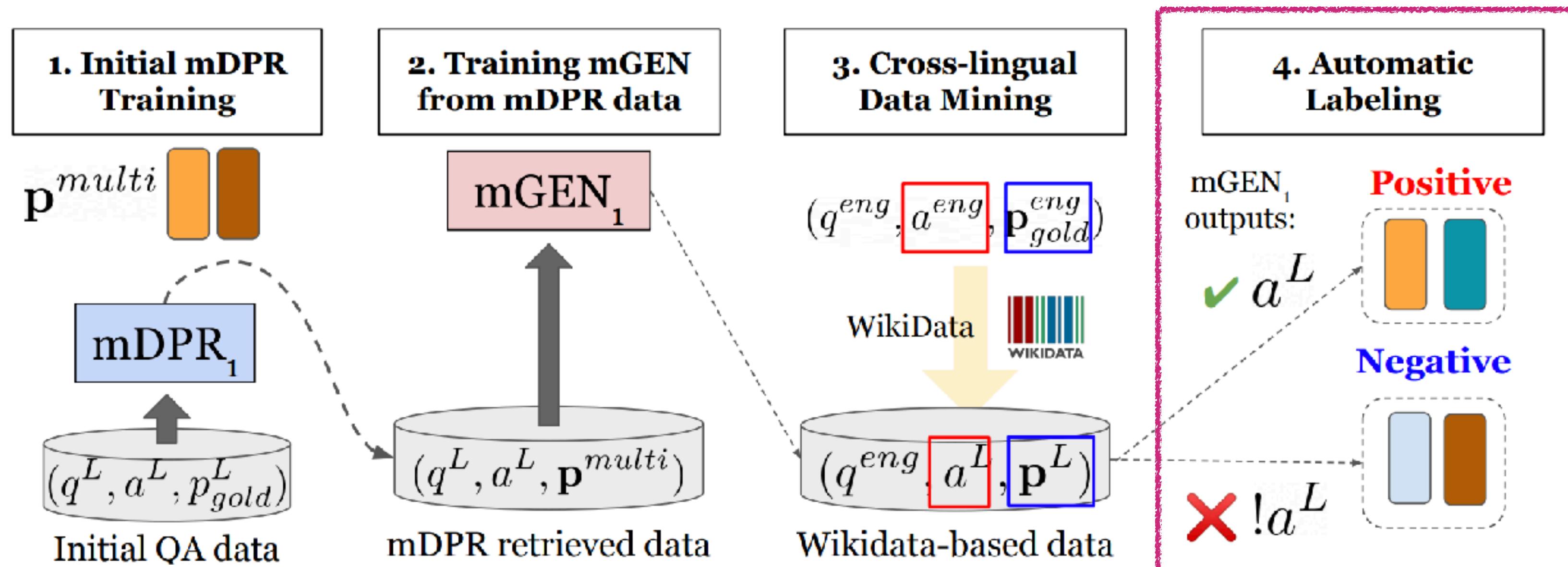
→ Training
-→ Inference



CORA: Iteratively training multilingual LM & retriever

Add new positive / negative paragraphs
based on whether LMs can answer correctly

→ Training
-→ Inference

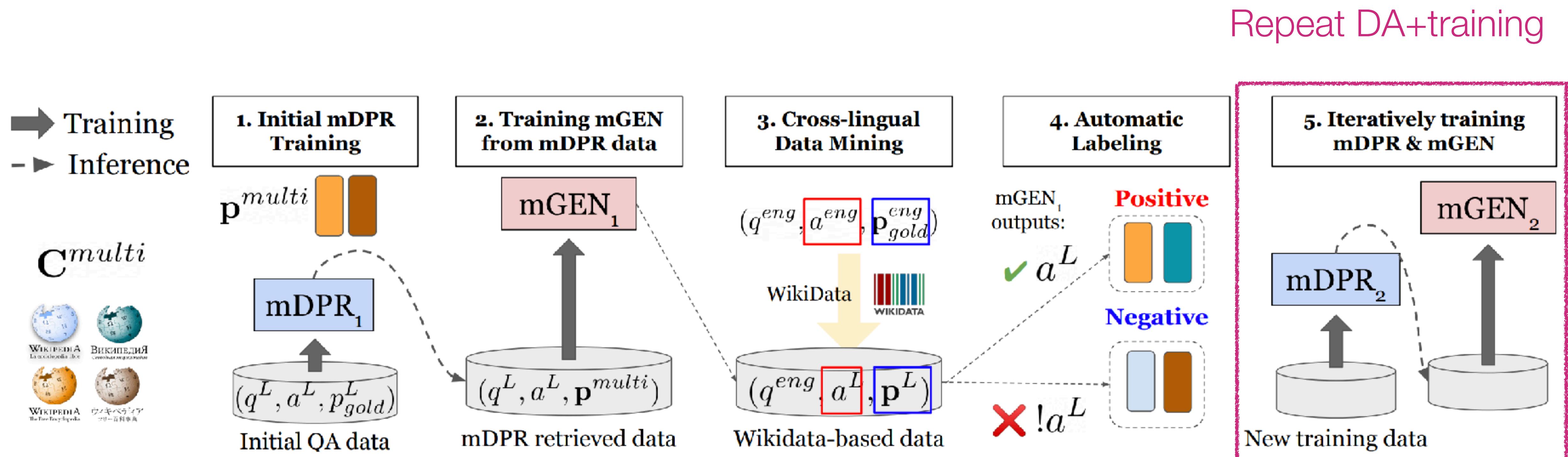


Links

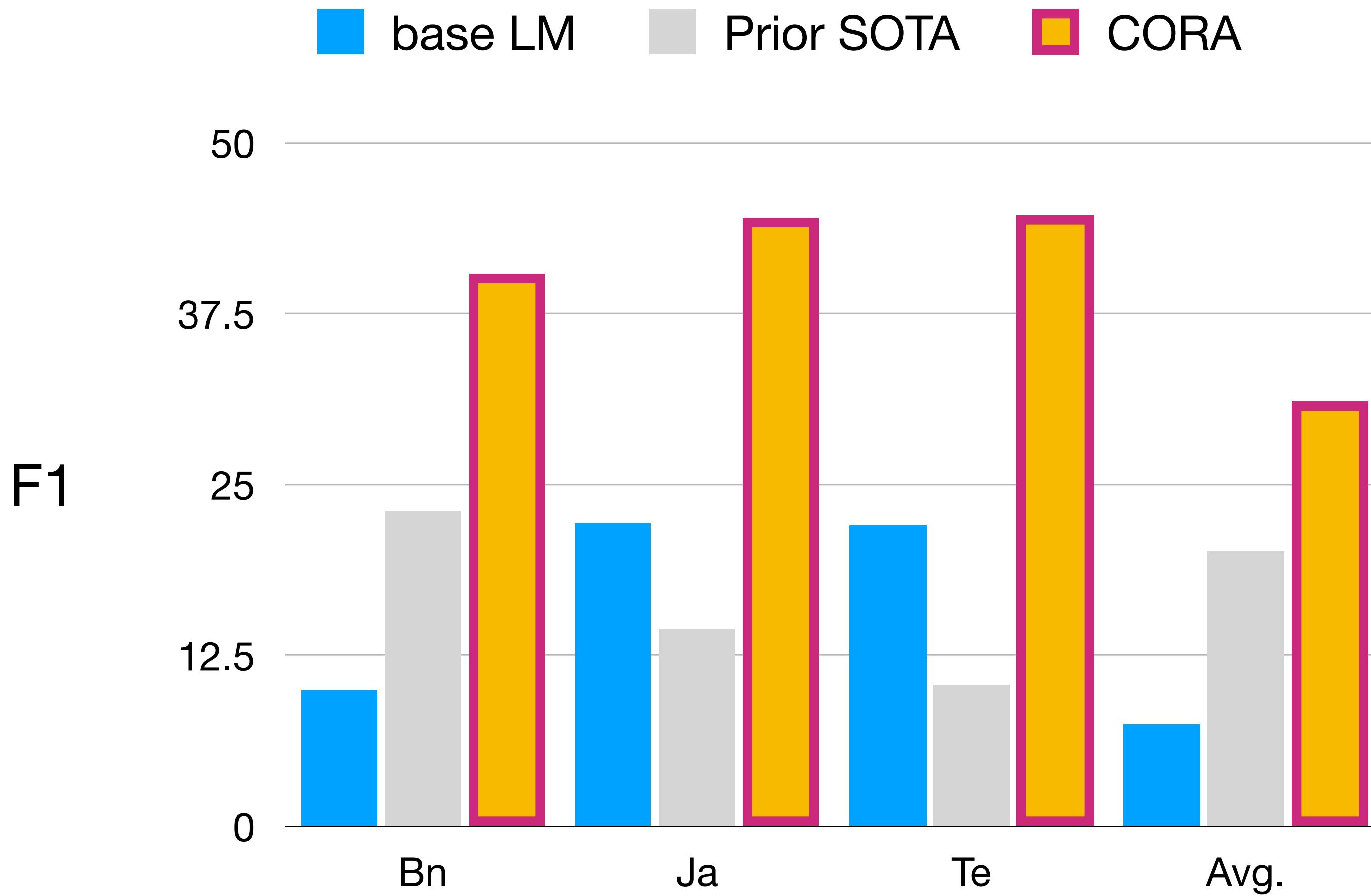
Eye of the Tiger
«Eye of the Tiger»
fue escrita
Frankie Sullivan
y Jim Peterik

Eye of the Tiger
Eye of the Tiger
הוא פומון בbijou של
להקת הרוק האמריקנית
סורבייכר, שכתבו ו
פרנקי סאליבן

CORA: Iteratively training multilingual LM & retriever

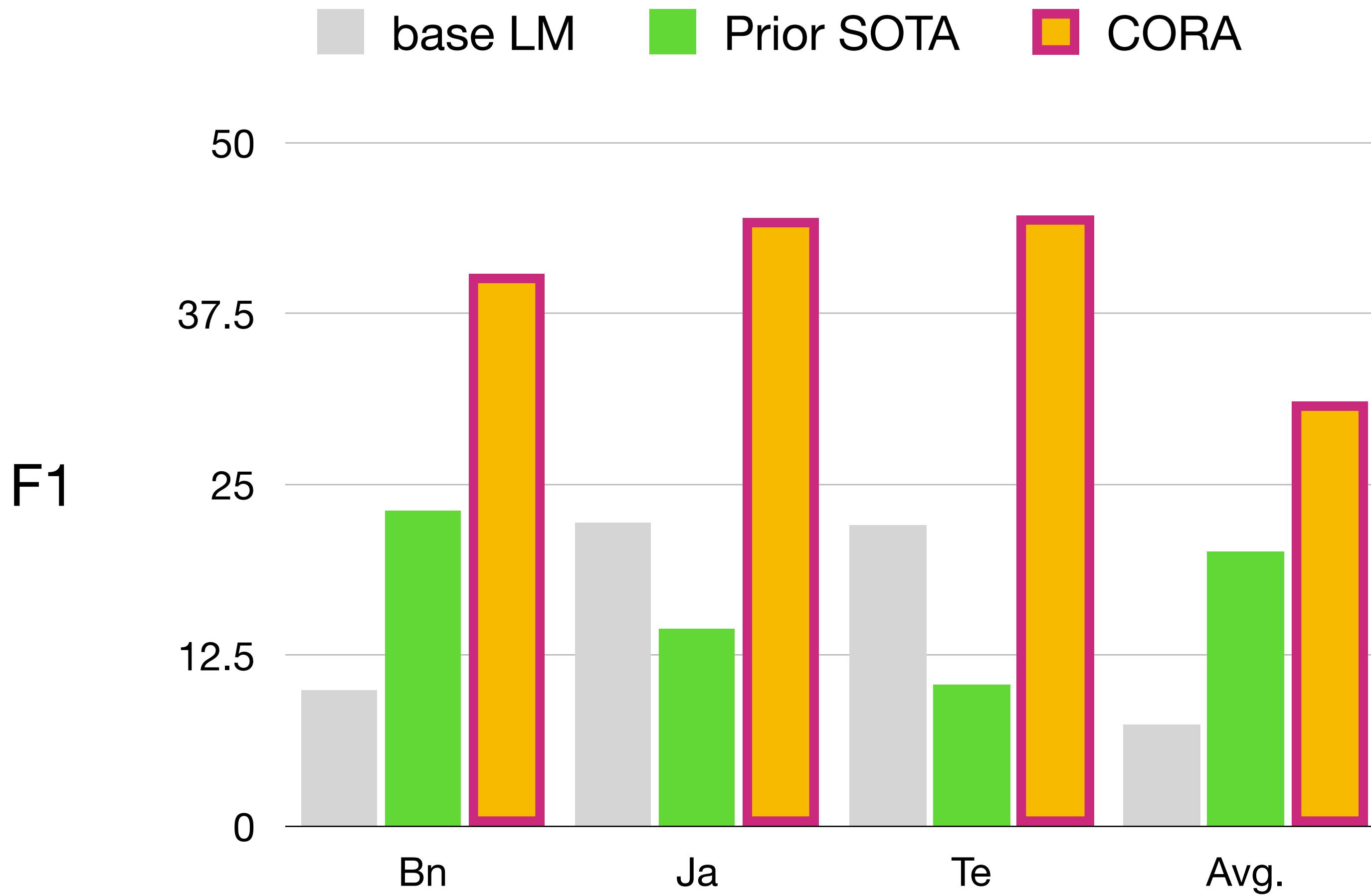


Results



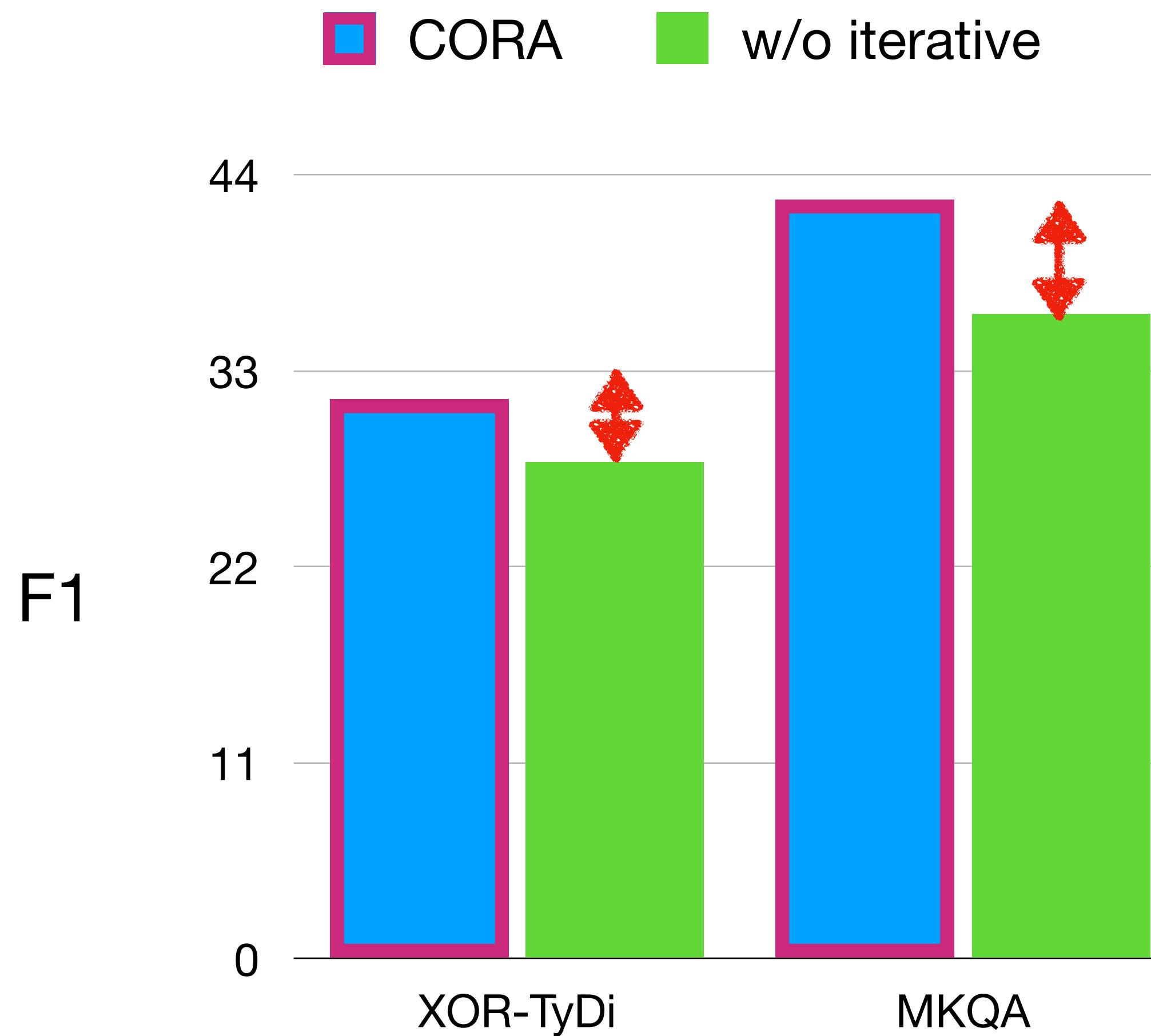
Large gains from fine-tuned
LM without retrieval

Results



Significantly outperforms
prior SOTA

Ablations: Effects of iterative retrieval



Iterative training of retriever and LM
gives large performance improvements

Multilingual retrieval-based LMs for diverse tasks

Question Answering

- * CL-ReLKT (Limkonchotiwat et al., 2022): knowledge transfer for better cross-lingual retrieval training
- * Gen-TyDi QA (Muller et al., 2023): generate full sentence answers for cross-lingual QA.
- * AfriQA (Ogundepo et al., 2023):: Cross-lingual Open-Retrieval Question Answering for African Languages: the first open-domain QA datasets for African languages

Fact verification

- * CONCRETE (Hung et al., 2022): Improving cross-lingual fact-checking with cross-lingual retrieval

Dialogue

- * Cross-lingual Knowledge-grounded Dialogue (Kim et al 2021): a Korean knowledge-grounded dialogue system that learns to generate Korean response given English & Korean knowledge

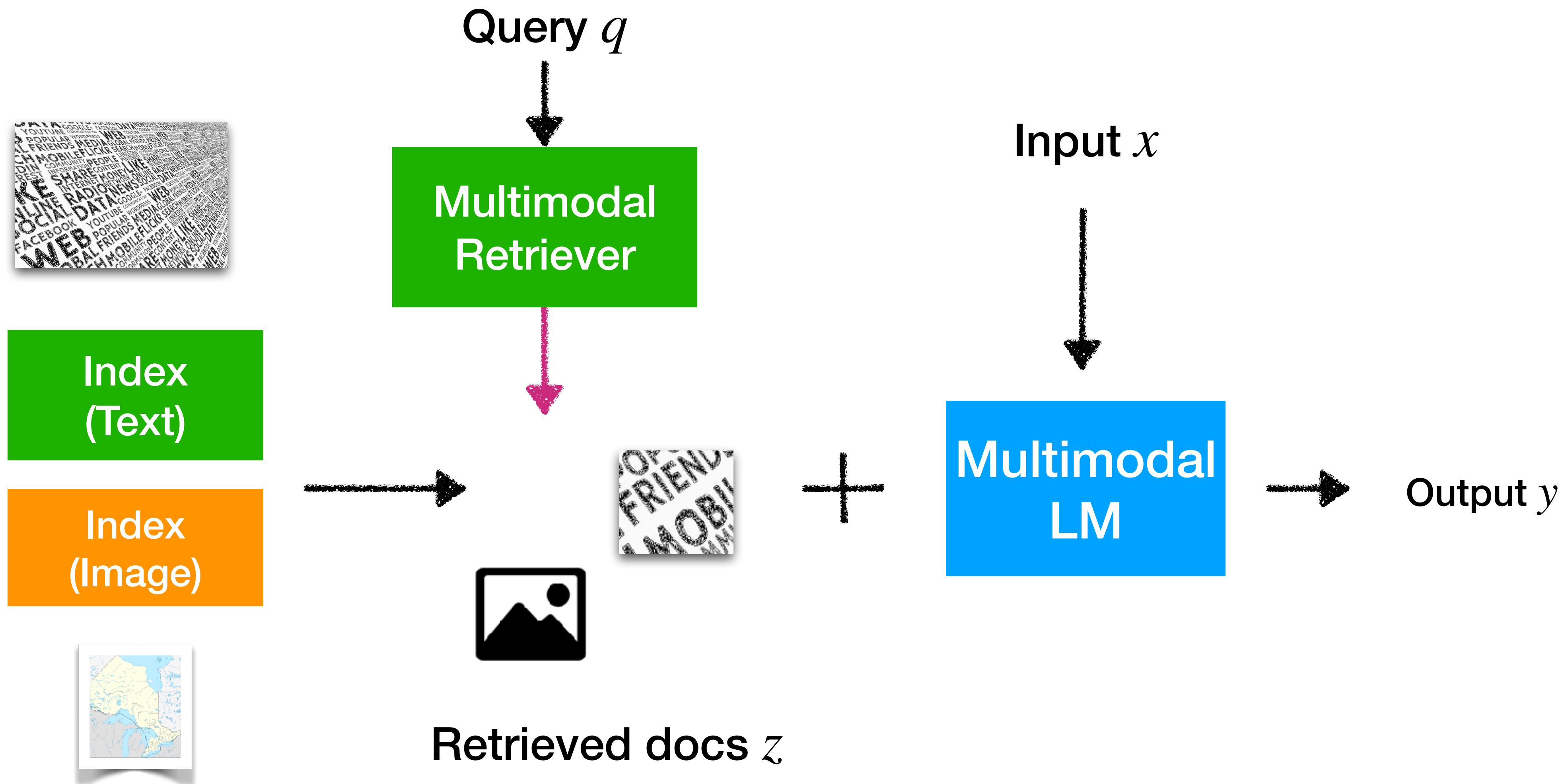
Event extraction

- * R-GQA (Du and Ji, 2022): retrieve similar QA pairs for event argument extraction.

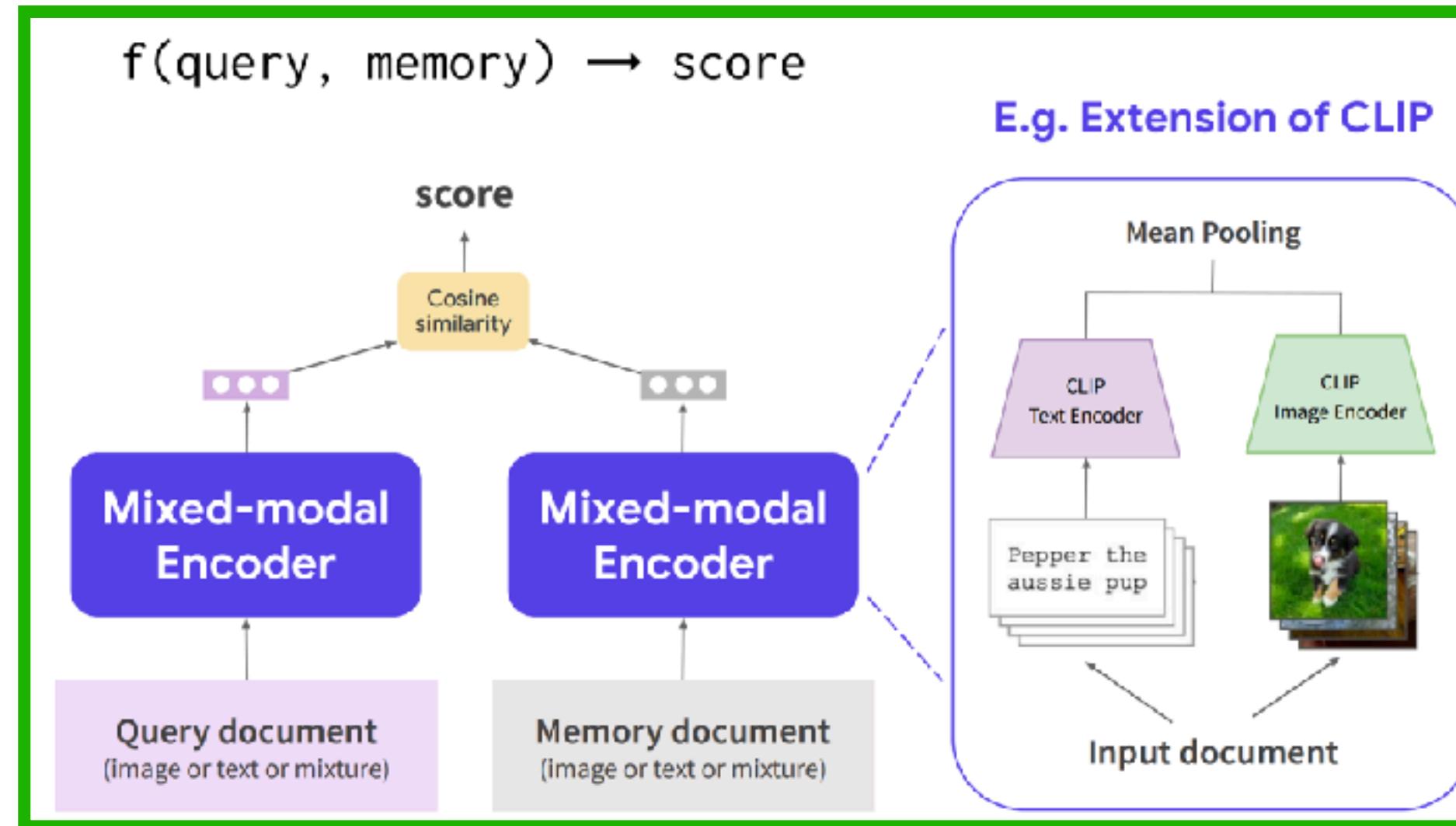
Key-phrase generations

- * Retrieval-augmented Multilingual Key phrase Generation (Gao et al 2022): Using iterative training to improve retrieval & LM for key phrase generations

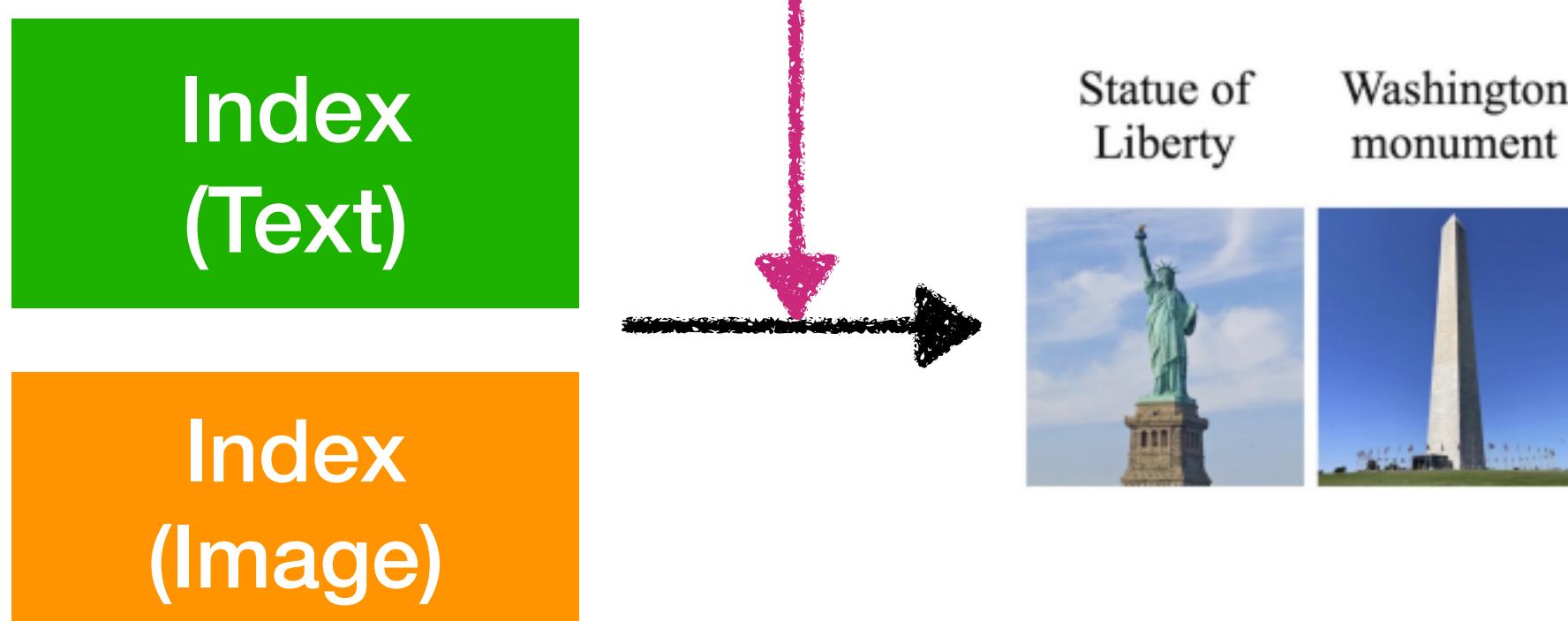
Multi-modal retrieval-based LMs



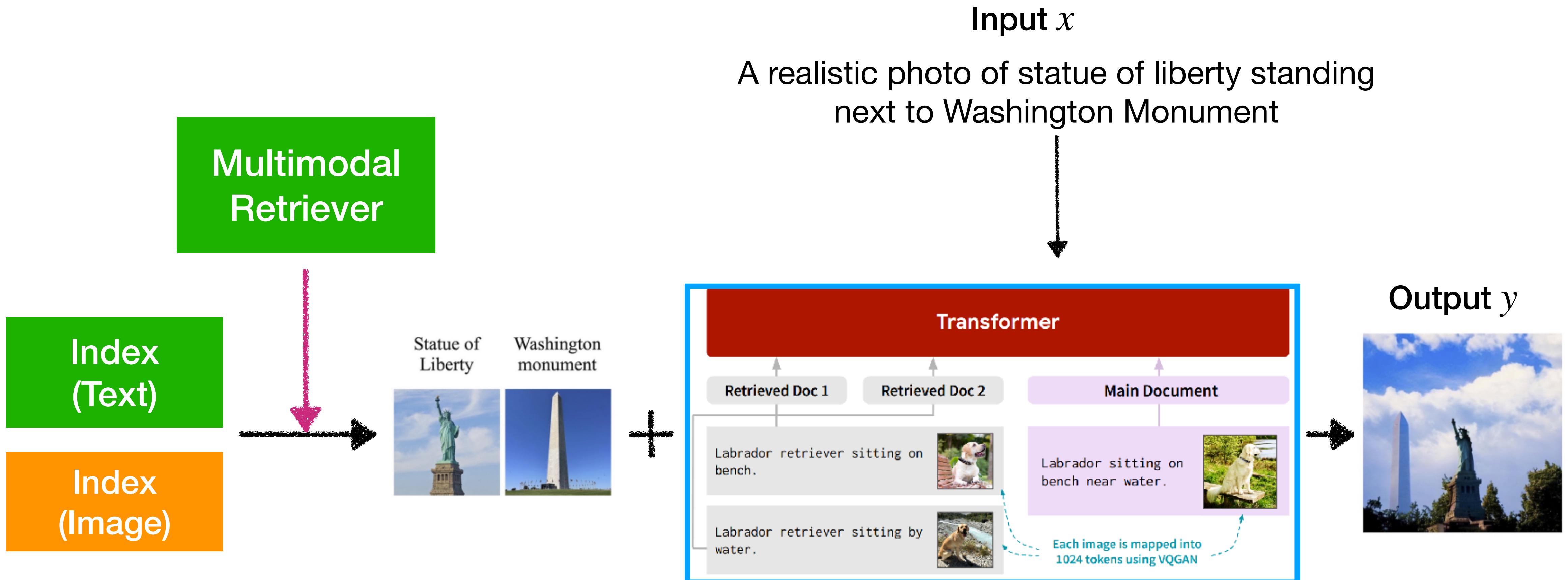
RA-CM3 (Yasunaga et al., 2023)



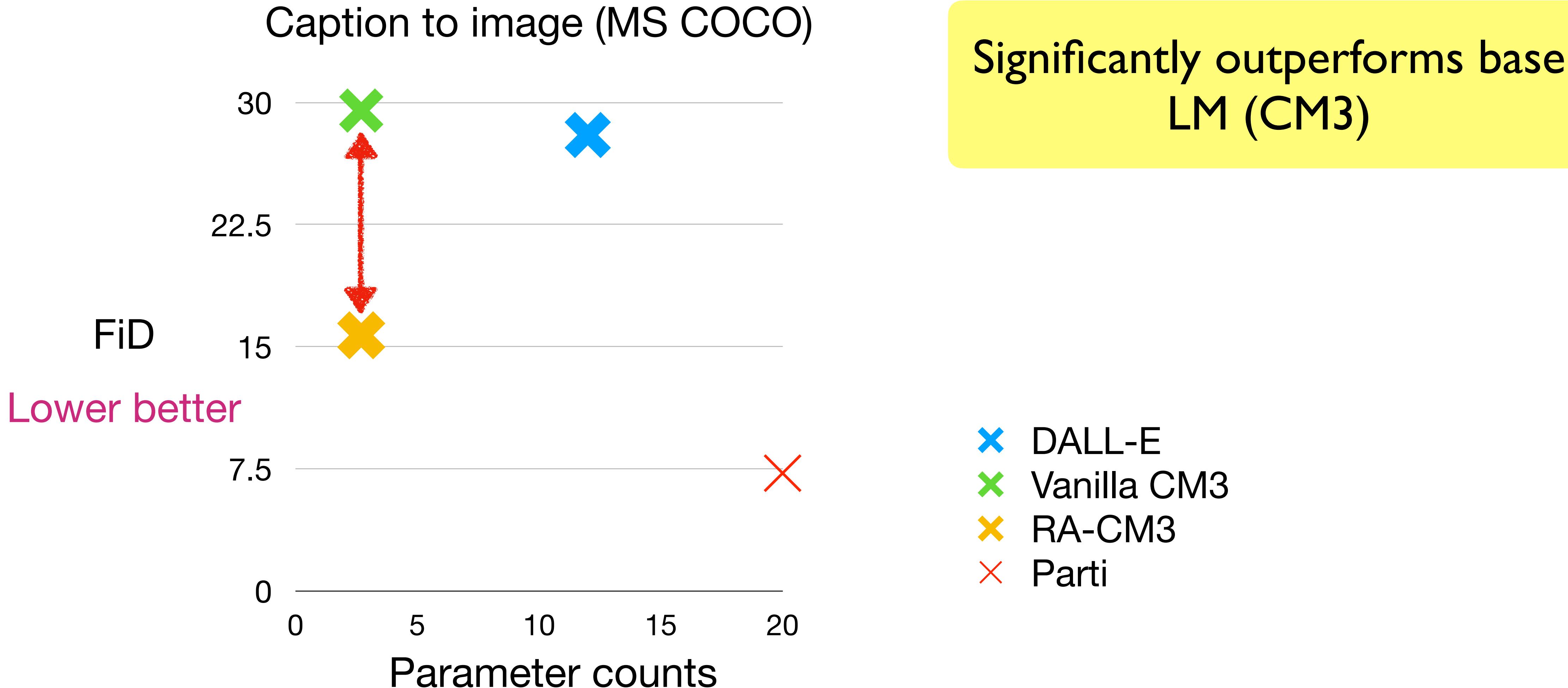
A realistic photo of statue of liberty standing next to Washington Monument



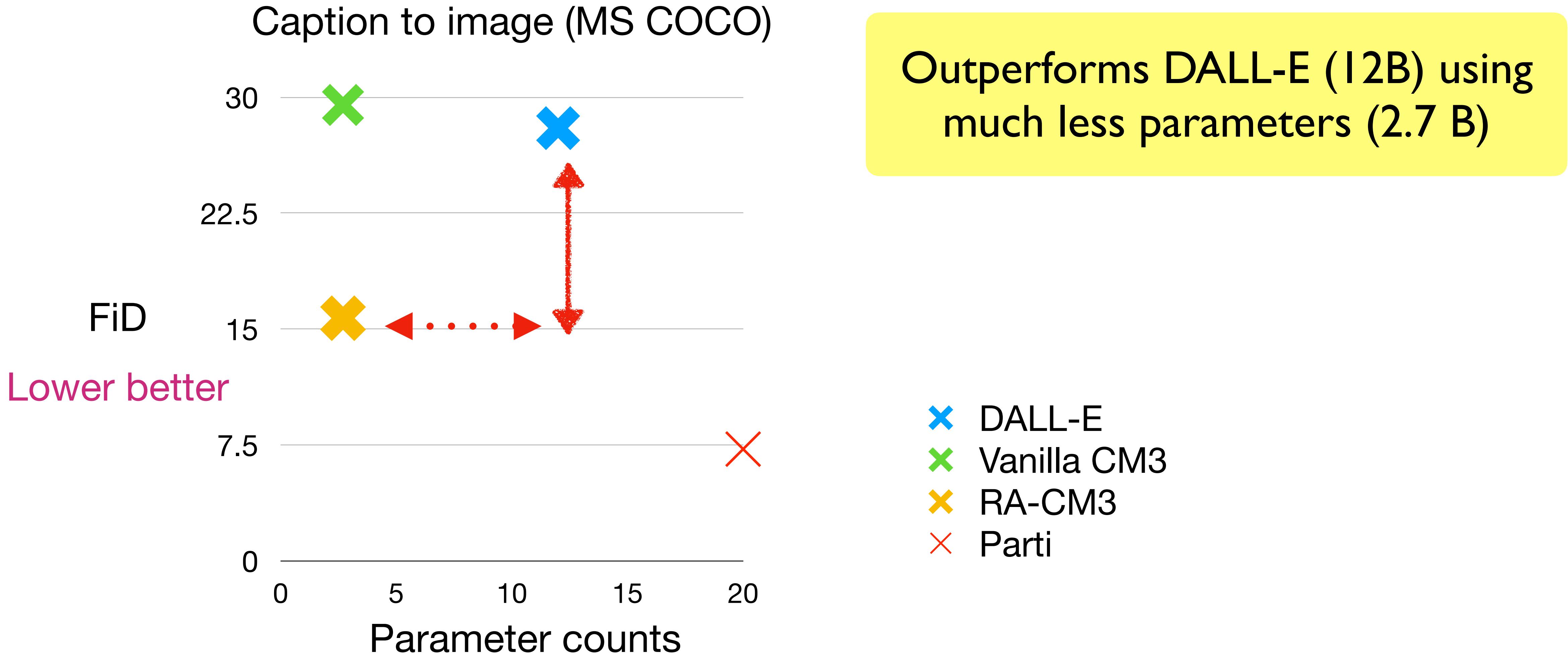
RA-CM3 (Yasunaga et al., 2023)



Results



Results

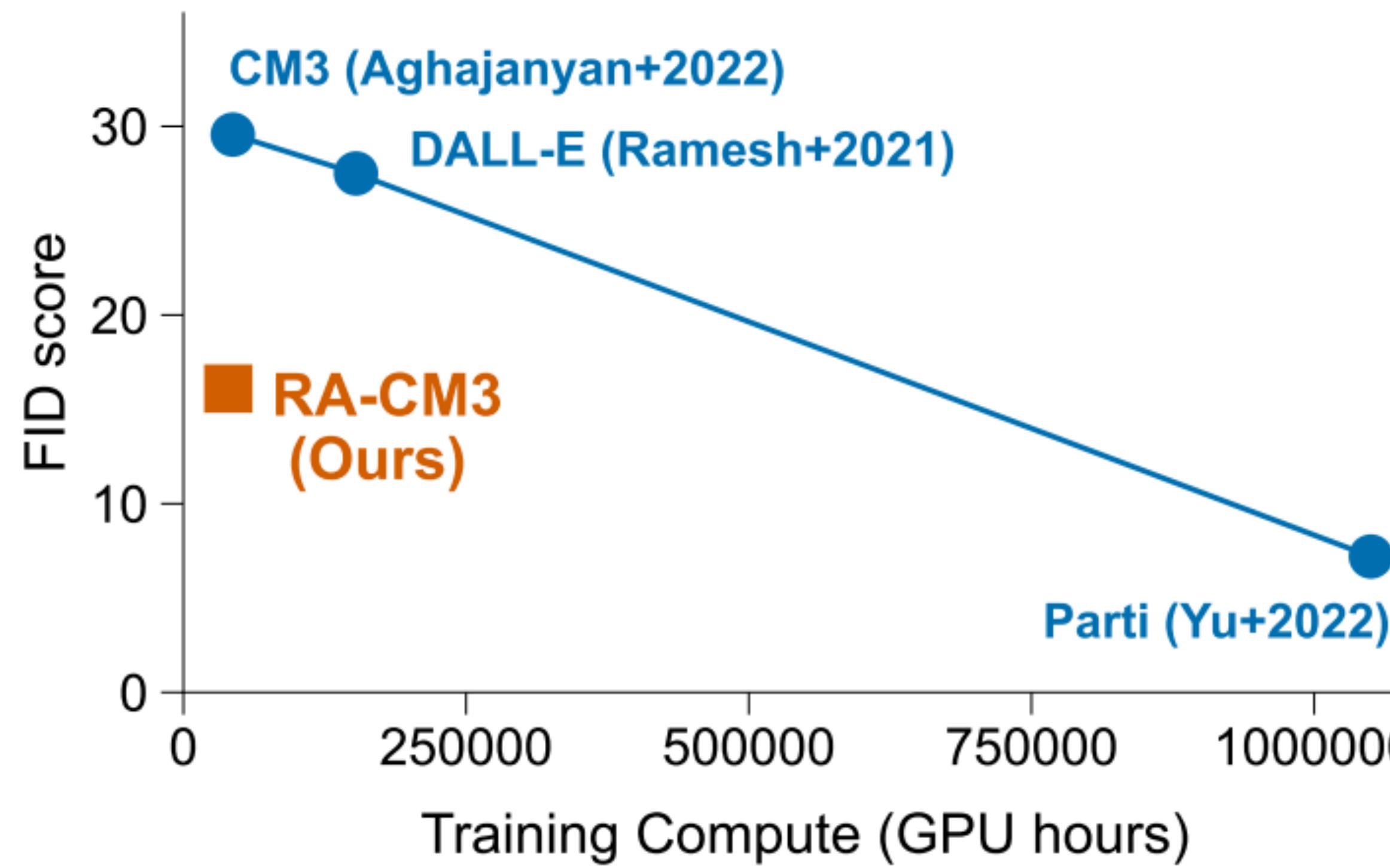


Yasunaga et al. 2023. "Retrieval-Augmented Multimodal Language Modeling"

Results

Caption to image (MS COCO)

FID score (↓) vs Training Compute



Achieves significantly better
training efficiency

More applications beyond text

Multi-modal Retrieval-augmented Pre-training

- * RAVEAL (Hu et al 2023): Pre-training visual-language model using knowledge memory

Multi-modal Question Answering

- * MuRAG (Chen et al., 2022)

Multi-modal Classification

- * ALIGN (Gur et al., 2021)

Multimodal using image and text have been actively studied

More applications beyond text

Multi-modal Retrieval-augmented Pre-training

- * RAVEAL (Hu et al 2023): Pretraining visual-language model using knowledge memory

Multi-modal Question Answering

- * MuRAG (Chen et al., 2022)

Multi-modal Classification

- * ALIGN (Gur et al., 2021)

Retrieval-augmented training for molecules

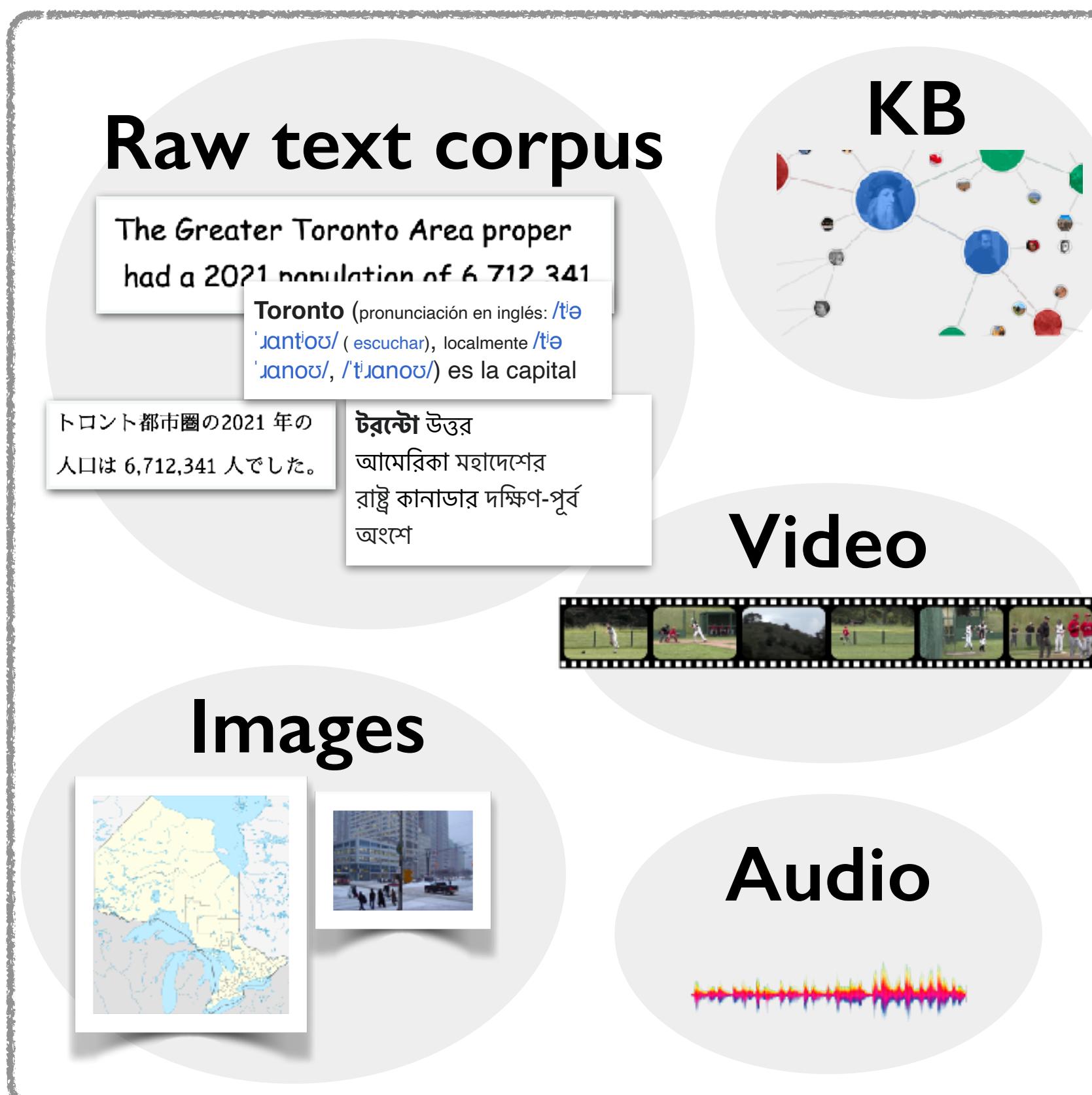
- * Retrieval-based Molecule Generation (Wang et al., 2023)

Retrieval-augmented 3D motion generations

- * ReMoDiffus (Zhang et al., 2023)

New extensions for new input / output modality!

Wrapping up



Extension to multilingual

Cross-lingual retrieval and generation to overcome **datastore scarcity** in many world languages

Extension to multimodal

Key effectivenesses (Section 5; long-tail, efficiency) apply to diverse modality

Section 7

Challenges & Opportunities

Tutorial summary

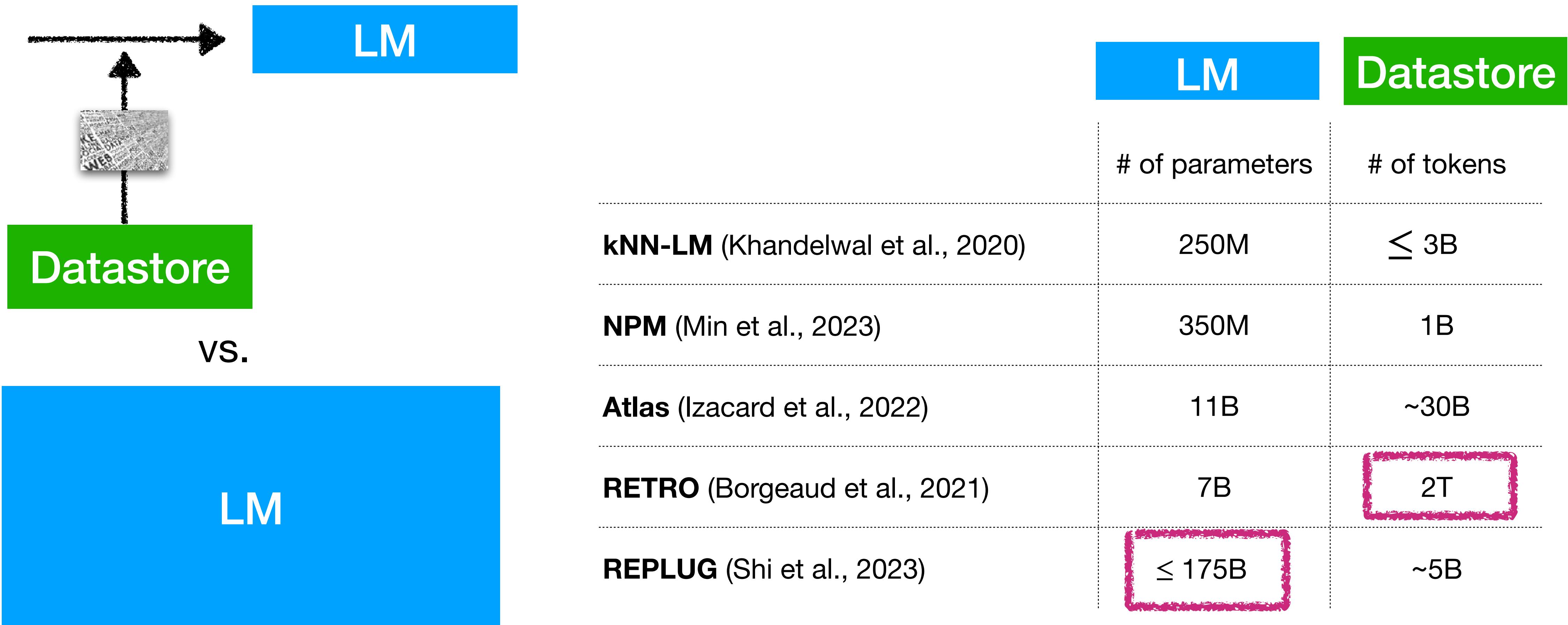
- **Definition & preliminaries**
- **Retrieval-based LMs: Architecture**
 - What to retrieve: tokens, text blocks, entity mentions
 - How to use retrieval: input layer, intermediate layers, output layers
 - When to retrieve: once, every n tokens, every token
- **Retrieval-based LMs: Training**
 - Independent training
 - Sequential training
 - Joint training with asynchronous index update
 - Joint training with in-batch approximation

Tutorial summary (cont'd)

- **Applications**
 - What tasks: knowledge-intensive NLP, code generation, classification etc
 - How to adapt: retrieval-based prompting, fine-tuning, reinforcement learning
 - When to use: long-tail, knowledge update, verifiability, parameter-efficiency, privacy, out of domain adaptations
- **Multilingual & multimodal extensions**
 - Multilingual: retrieve and generate cross-lingually to overcome scarcity of datastore
 - Multimodal: multimodal extensions of retrieval-based LMs for new modality

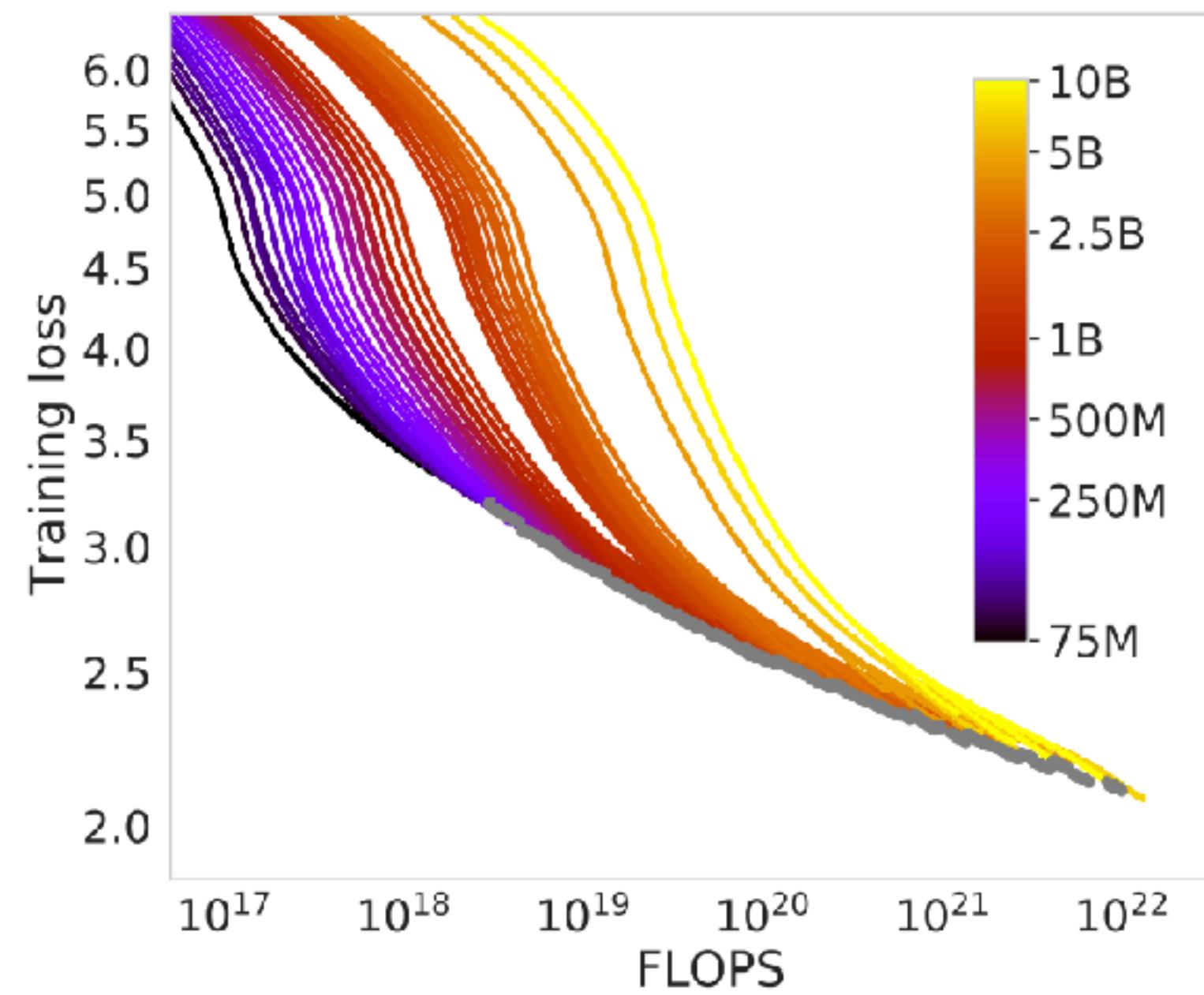
Challenge: Scaling retrieval-based LMs

A small LM + a large datastore \approx a large parametric LM?

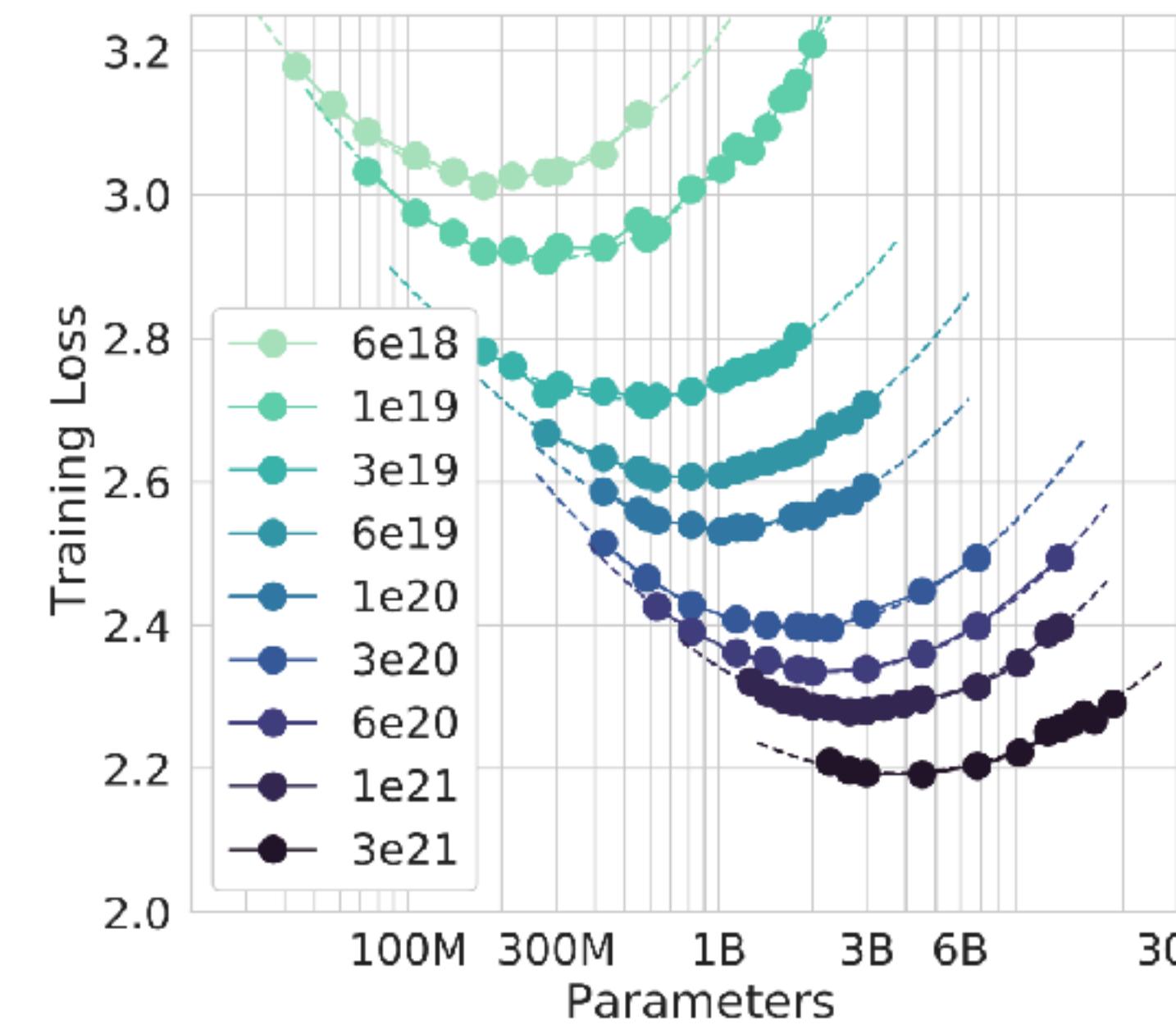


Challenge: Scaling retrieval-based LMs

Scaling law?



Scaling law for parametric LMs (Kalpan et al., 2020; Hoffman et al., 2022)



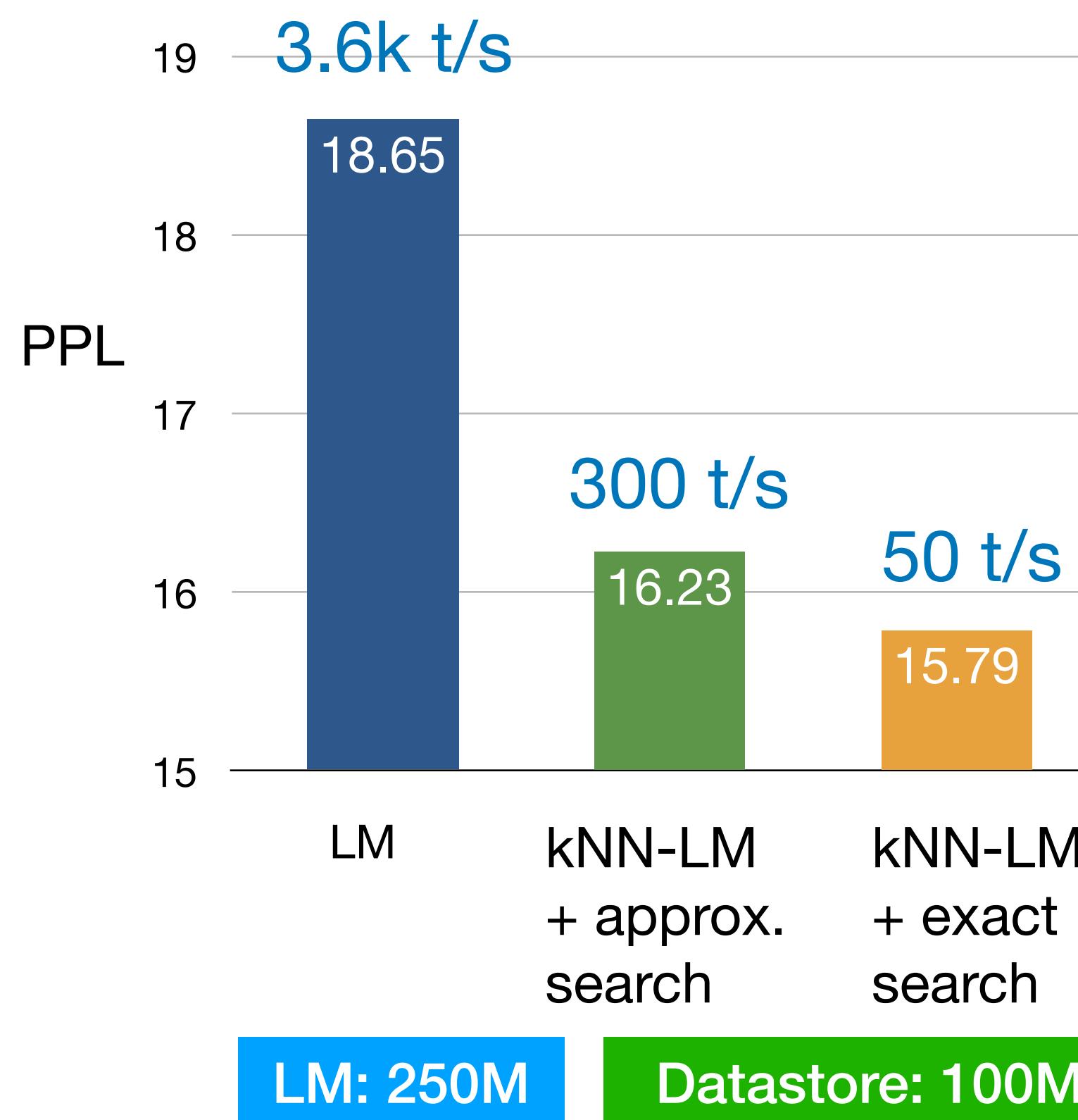
Loss as a function of:

- Training tokens
- Model sizes
- Training FLOPs
- + Datastore sizes?

Dependent on architecture and training methods!

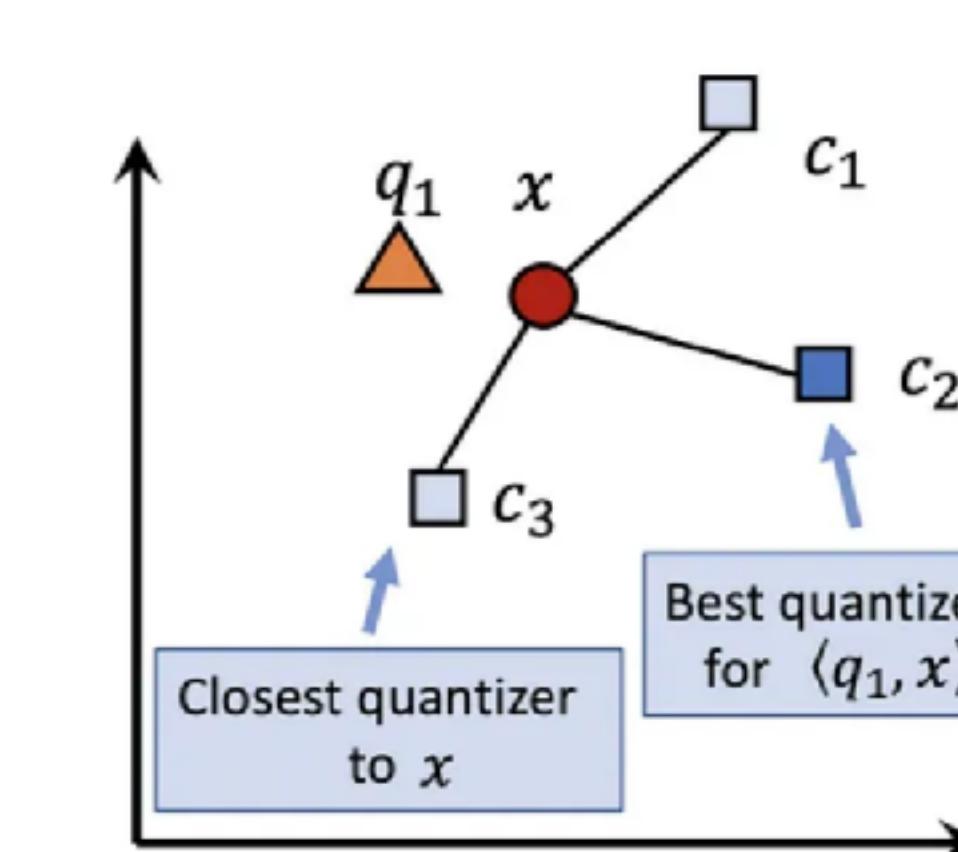
Challenge: Scaling retrieval-based LMs

Measured on NVIDIA RTX 3090 GPU
(Zhong et al., 2022) with a FAISS indexer
(Johnson et al., 2021) with 32 CPUs



Efficiency of similarity search

- >12 times slower even with **approximate nearest neighbor search implemented on GPUs**
- Efficient similarity search on GPUs/TPUs becomes the bottleneck of scaling retrieval-based LMs



Active research area!

Challenge: Scaling retrieval-based LMs

Efficiency of similarity search

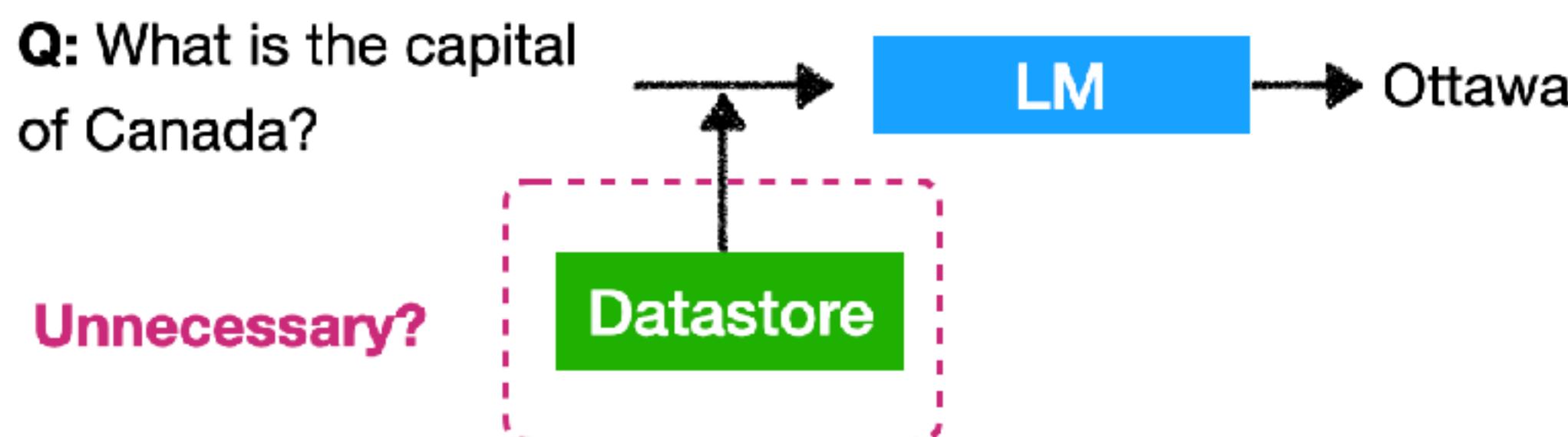
Efficient nearest neighbor search:

- Better loss for vector quantization (Guo et al., 2020)
- Optimization over different accelerators (Chern et al., 2022)
- Adaptive representations (Rege et al., 2023)

Space efficiency?

It also requires a large disk space for storage if the datastore becomes too large - storing **vectors or raw text?**

Middle ground: adaptively retrieve only when necessary (He et al., 2021; Mallen et al., 2023)



Challenge: Retrieval-based LMs for applications

Open-ended text generation?

DA

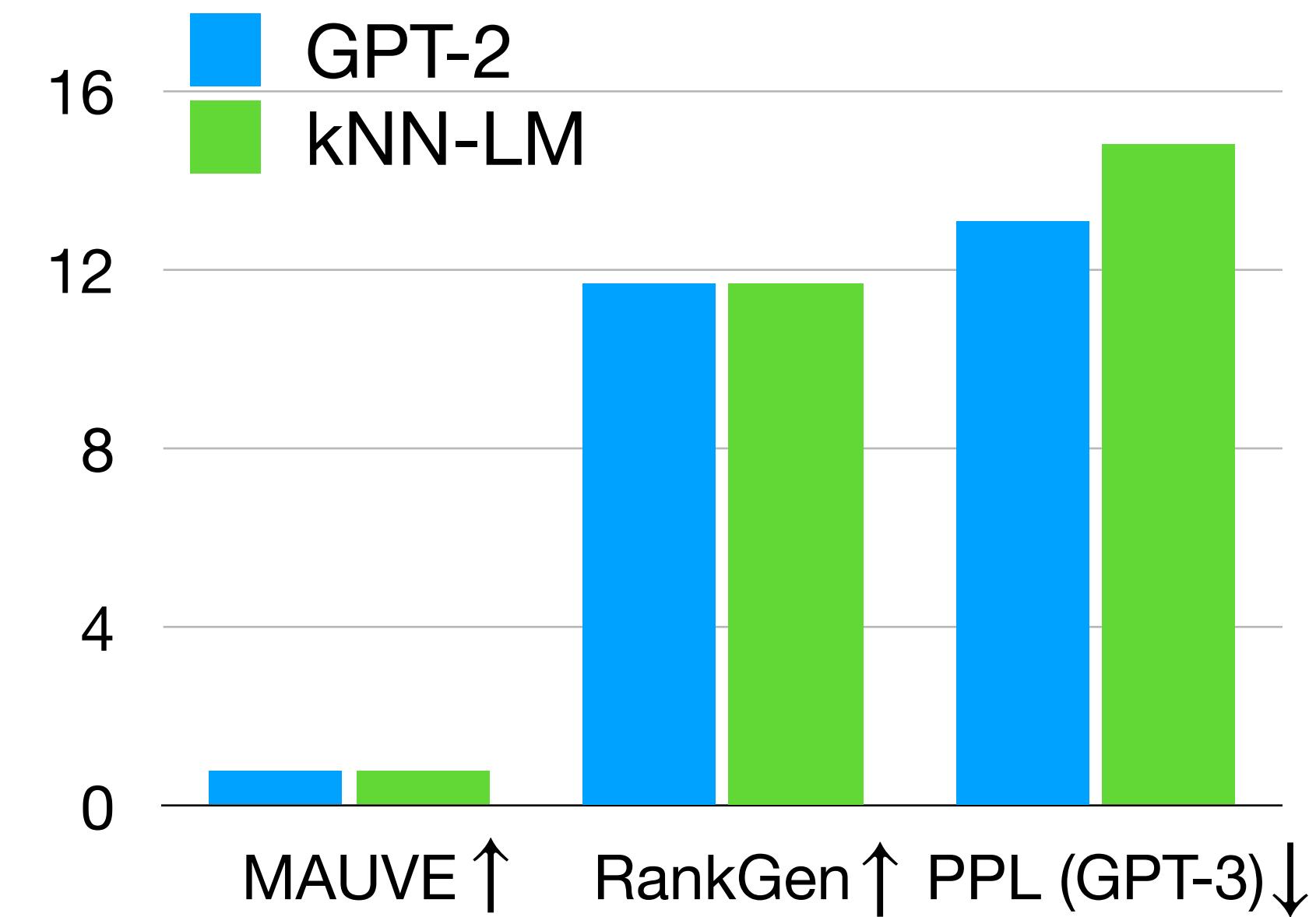
Hey I am going to ACL in Toronto next week. Can you tell me how I should explore the conference?



Certainly! Attending a conference like ACL (Association for Computational Linguistics) can be a fantastic opportunity to learn, network, and engage with the research community. Here are some tips to help you make the most of your time at the conference:

1. Review the schedule: Take some time to familiarize yourself with the conference program in advance. Note the sessions, workshops, tutorials, and keynotes that interest you the most. Identify any must-attend sessions and plan your schedule accordingly.
2. Create a personalized agenda: Once you have a clear understanding of the sessions you want to attend, create a personalized agenda. Note down the session names, locations, and times so you can easily navigate the conference venue and stay organized.
3. Engage in networking: Conferences provide an excellent opportunity to connect with fellow researchers and professionals in your field. Be proactive and introduce yourself to others during coffee breaks, poster sessions, or social events. Don't be afraid to strike up conversations, exchange ideas, and ask questions. Networking can lead to valuable collaborations and partnerships.
4. Attend keynote speeches: Keynote speeches are typically delivered by renowned experts in

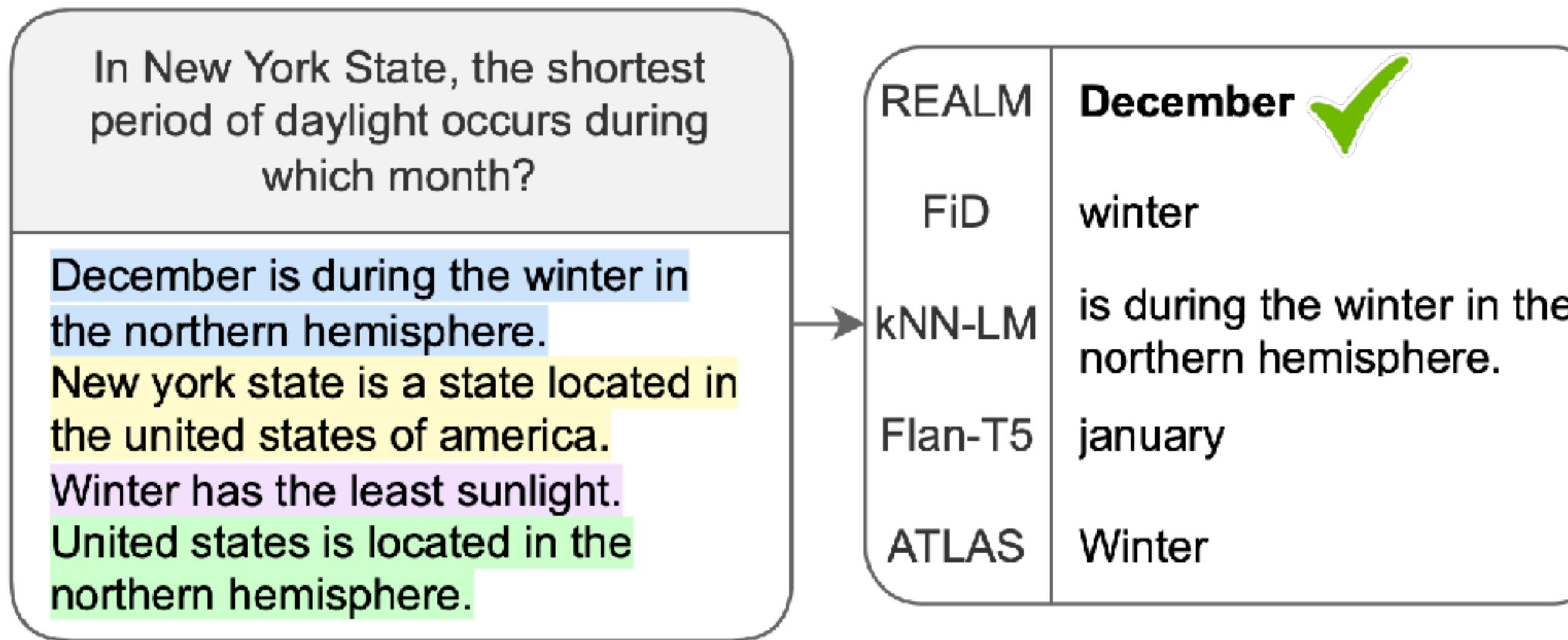
Exposure bias? Better decoding algorithms?



Wang et al. 2023. “kNN-LM Does Not Improve Open-ended Text Generation”

Challenge: Retrieval-based LMs for applications

Complex reasoning tasks?



Retrieval-based LMs struggle with **multi-step entailments** or **logical reasoning**:

- Retrieving based on similarity of queries is insufficient 😢
- LMs also struggle with synthesizing information from multiple statements 😢

Challenge: Retrieval-based LMs for applications

Complex reasoning tasks?

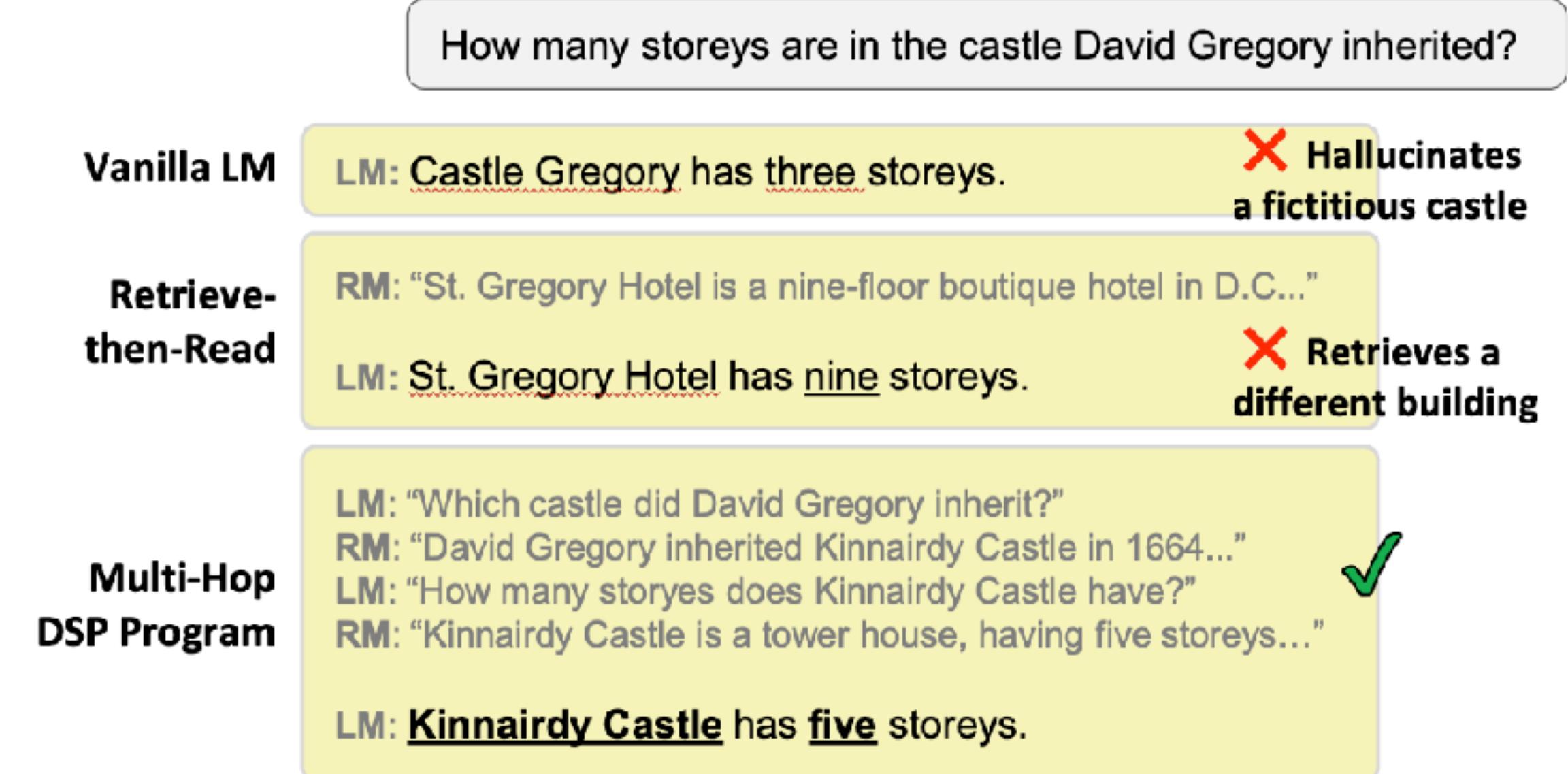
Potential solutions?

- **Iterative retrieval & query reformulation**

- FLARE (Jiang et al., 2023), Self-Ask (Press et al., 2022), ReAct (Yao et al., 2023)

- Decompose the task into a **multi-hop program** for complex reasoning

- DSP: Demonstrate-Search-Predict



LM: Language Model, RM: Retrieval Model (both frozen)

Open questions

- What is the best **architecture & training method** for retrieval-based LMs in practice?
- We still don't know yet how to best **scale up** these models - Scaling law?
- We may need to explore alternative **decoding or adaptation methods** in downstream tasks (e.g., open-ended text generation, complex reasoning)!



Q & A

Thank you for joining us today!

All the materials are at
<https://acl2023-retrieval-lm.github.io/>