ACL 2023 Tutorial:

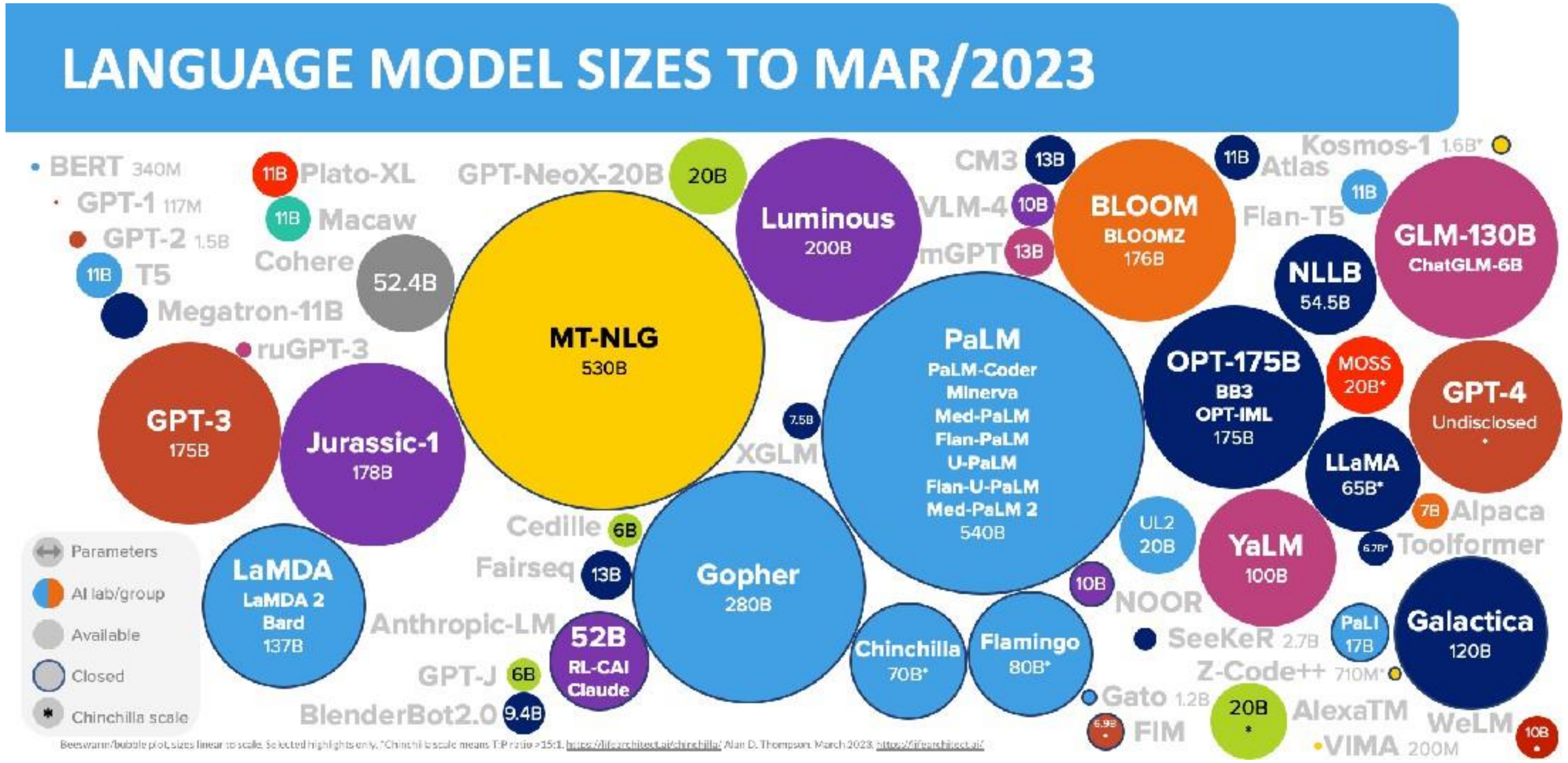
Retrieval-based Language Models and Applications

Akari Asai, Sewon Min, Zexuan Zhong, Danqi Chen

CNeRG Reading Group Presentation
(14th November, 2024)

Sayantan Adak
Kiran Purohit

The age of large language models (LLMs)



- Transformers-based, **fully parametric**

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

Retrieval for knowledge-intensive NLP tasks

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

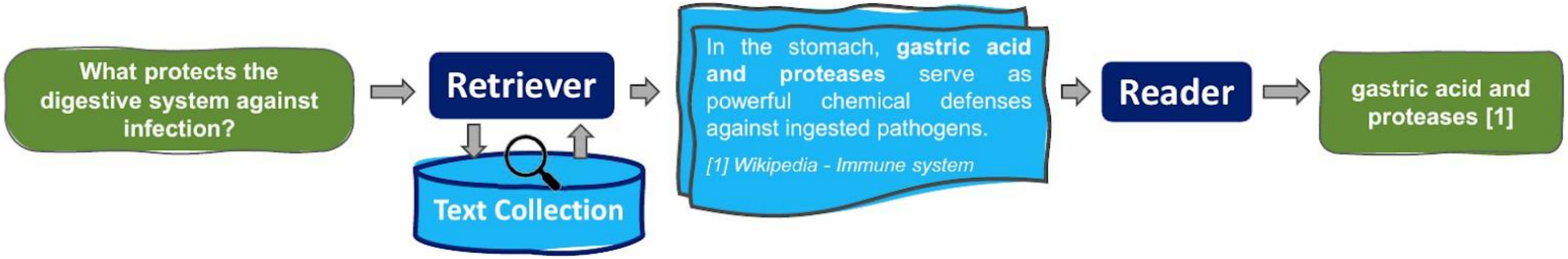


Image: http://ai.stanford.edu/blog/retrieval-based-NLP/

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

ChatGPT

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

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

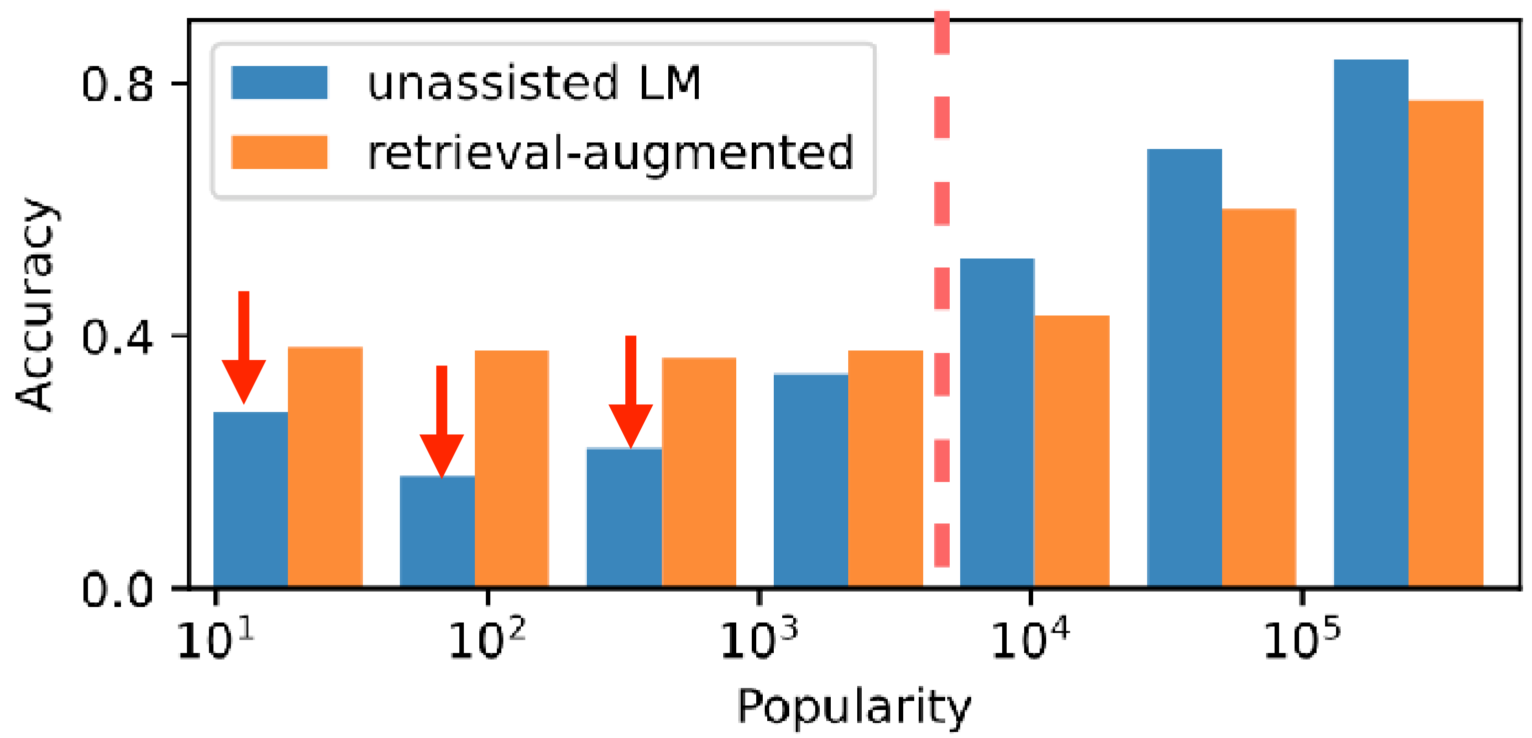
2. "**Deep Boltzmann Machines**" (with **R. Salakhutdinov**) - Published in **2009**, ..

…

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

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

What is Kathy Saltzman's occupation?



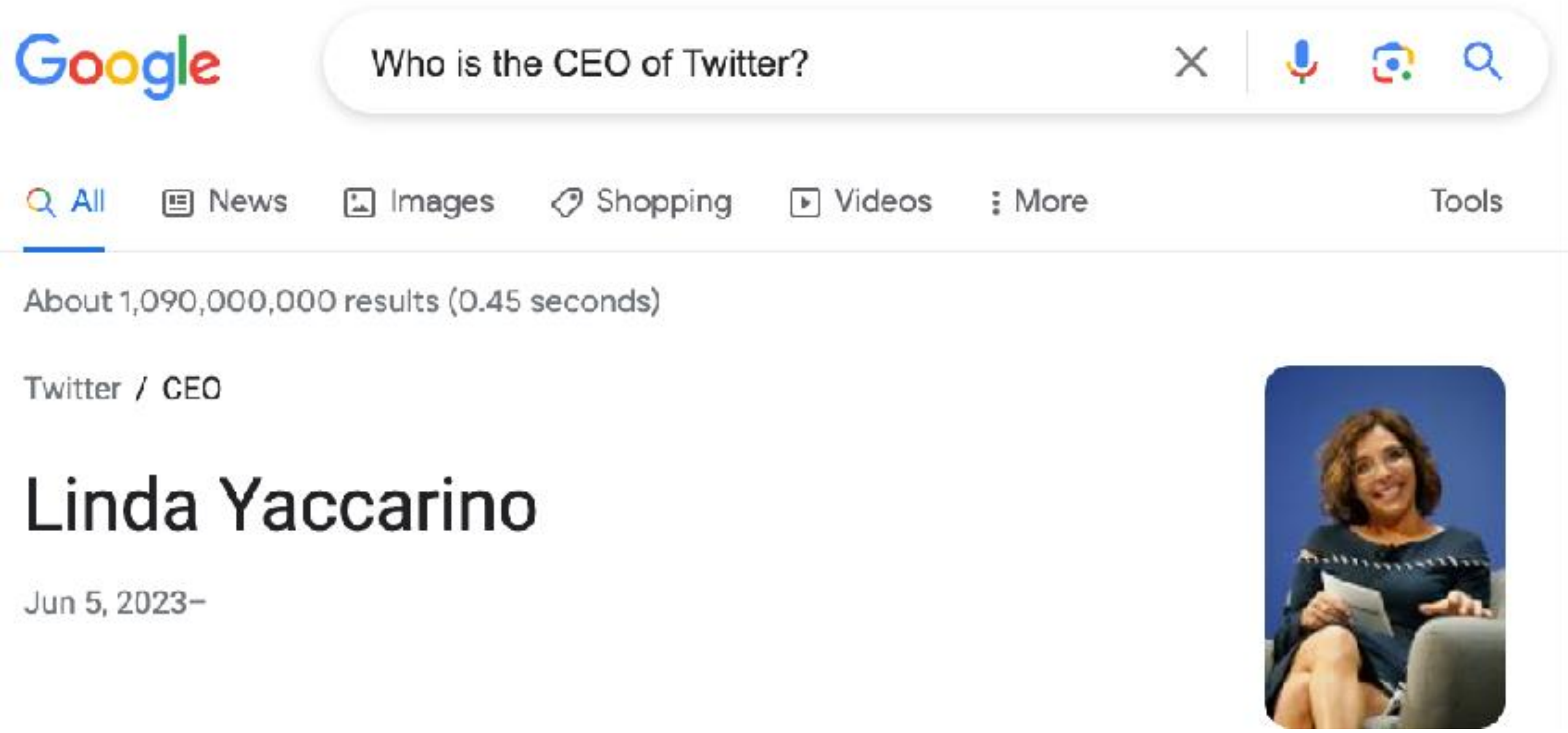
Why retrieval-based LMs?

LLMs’ knowledge is easily outdated and hard to update

Who is the CEO of Twitter?

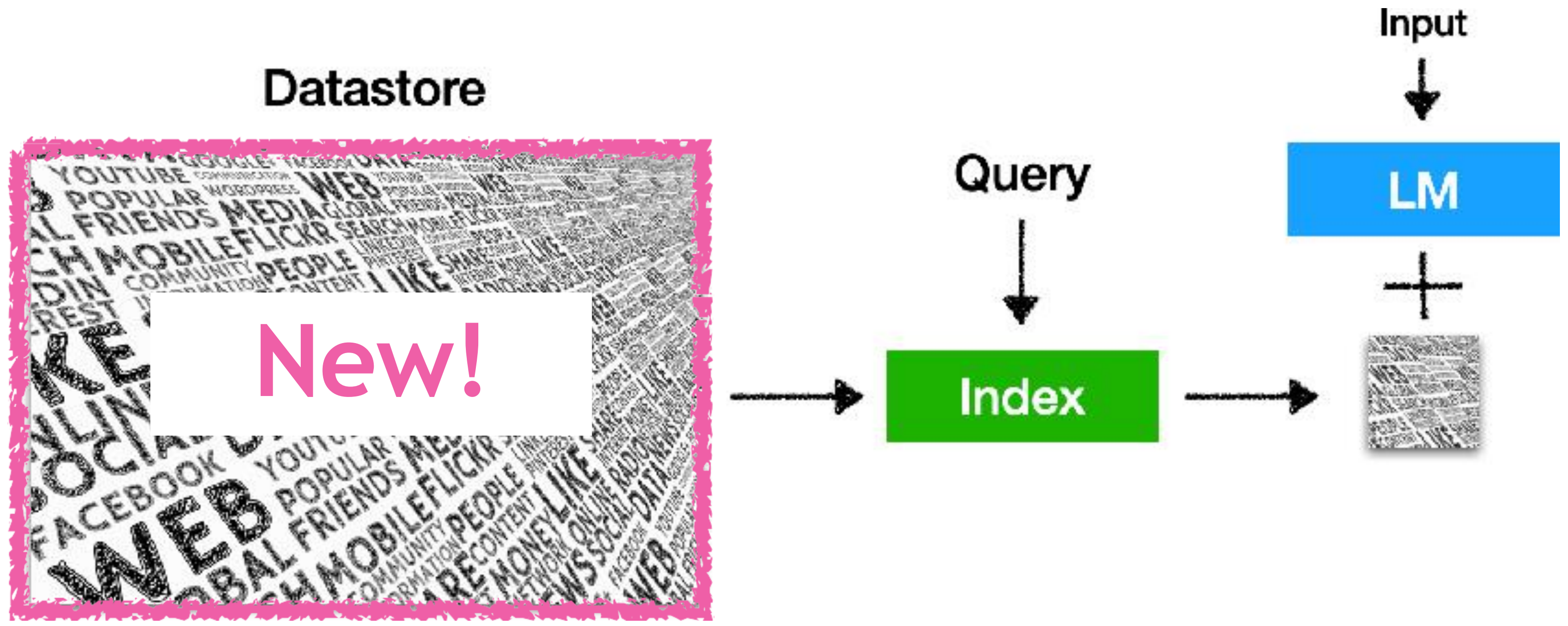
ChatGPT

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



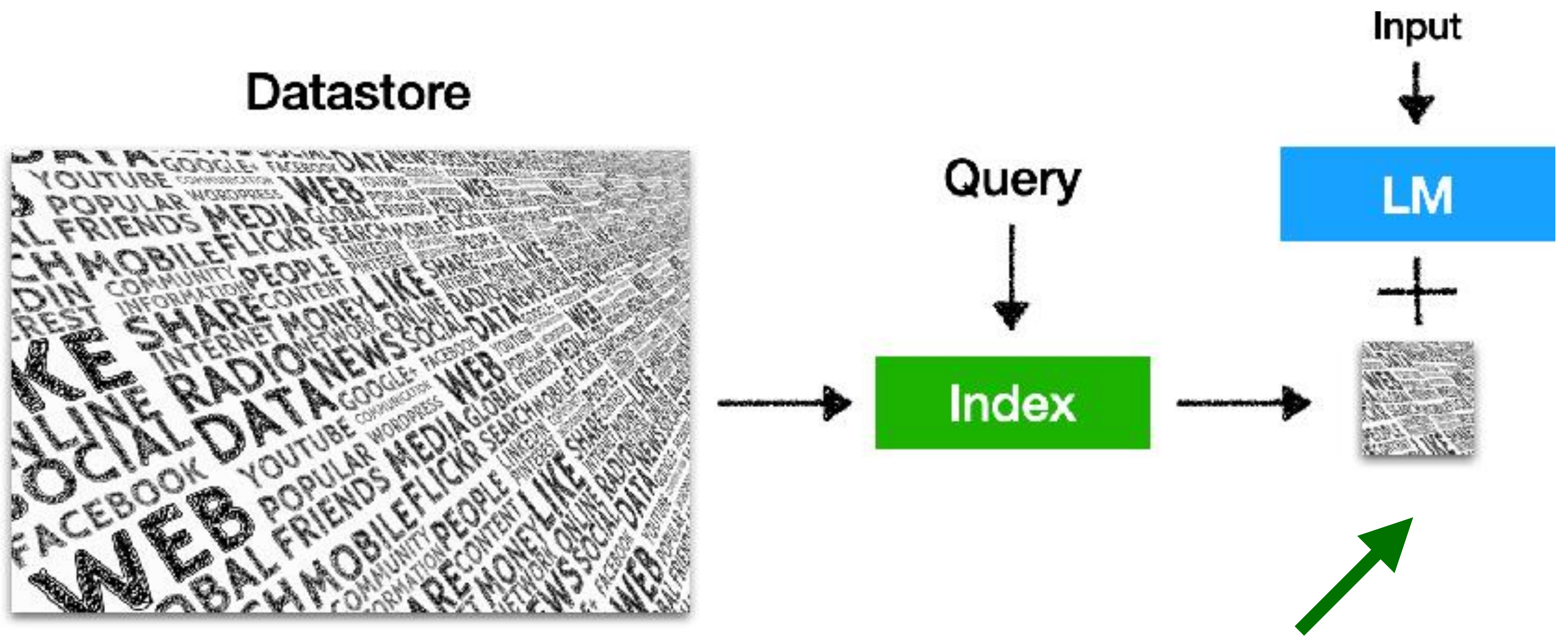
- Existing **knowledge editing** methods are still NOT scalable (active research!)

- The datastore can be easily **updated** and **expanded** - even without retraining!



Why retrieval-based LMs?

LLMs' output is challenging to interpret and verify



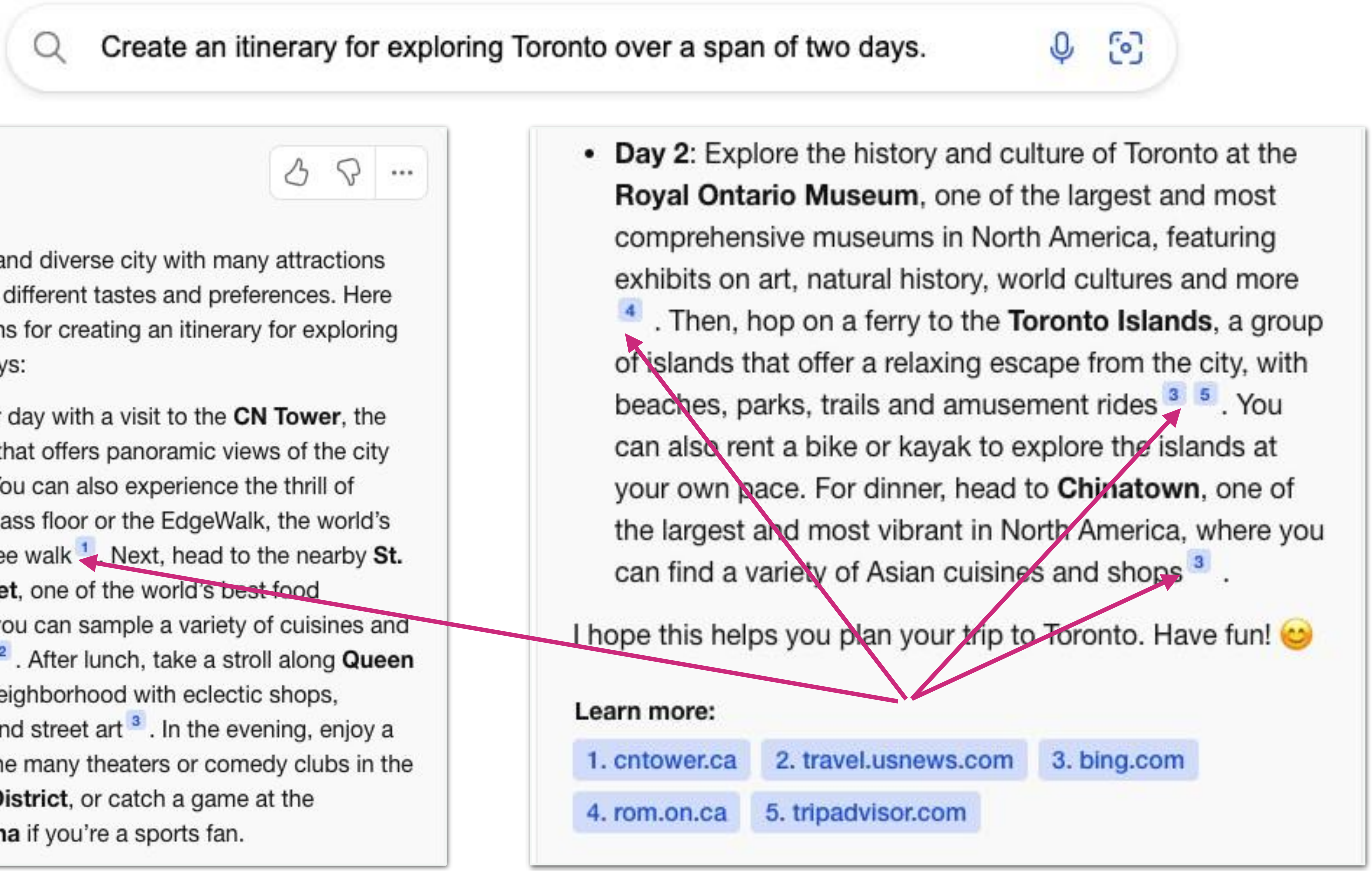
Can **trace knowledge source** from retrieval results - better **interpretability** & **control**

Generating text with citations

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

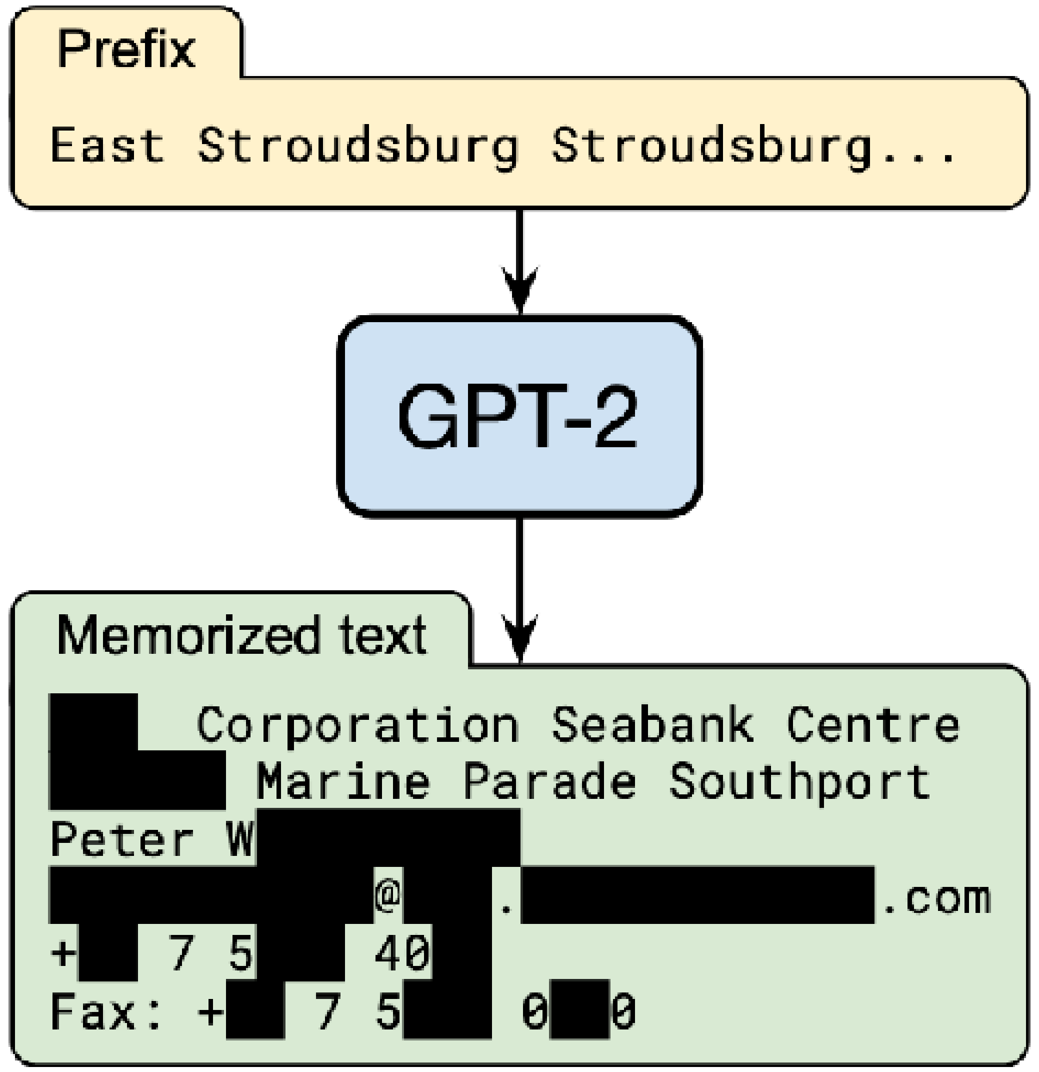
Why retrieval-based LMs?

LLMs' output is challenging to interpret and verify



Why retrieval-based LMs?

LLMs are shown to easily leak private training data



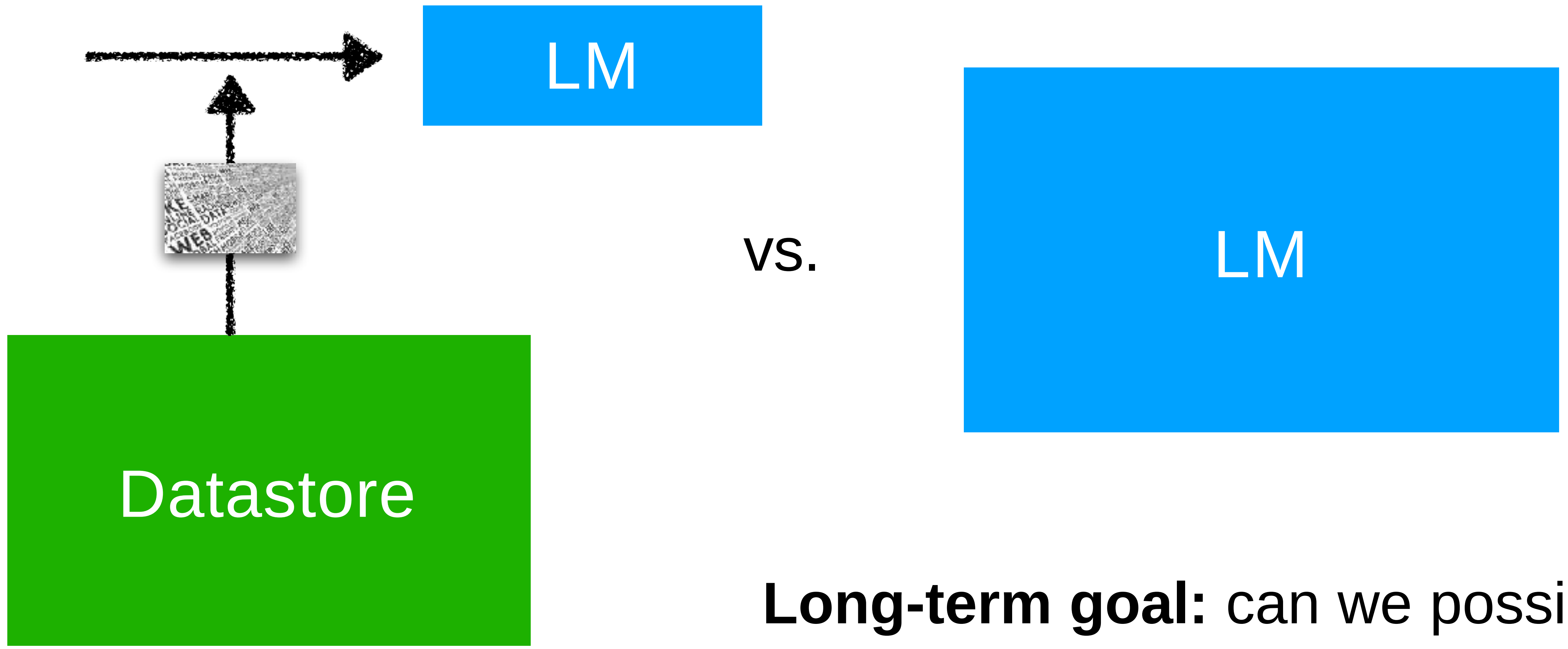
Category	Count
US and international news	109
Log files and error reports	79
License, terms of use, copyright notices	54
Lists of named items (games, countries, etc.)	54
Forum or Wiki entry	53
Valid URLs	50
Named individuals (non-news samples only)	46
Promotional content (products, subscriptions, etc.)	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

Carlini et al. 2021. “Extracting Training Data from Large Language Models”

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?

A Retrieval-based LM: Definition

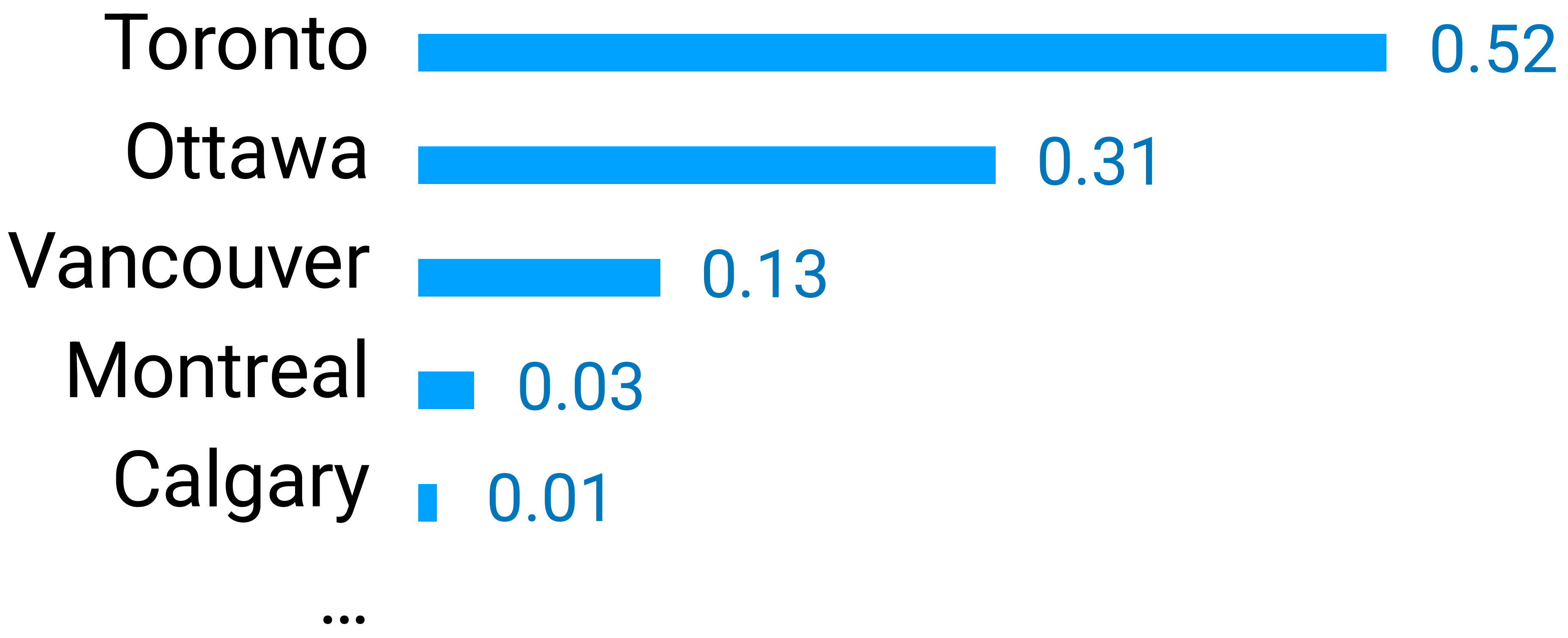
A language model (LM) that uses
an external datastore at test time

A Retrieval-based LM: Definition

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

A language model (LM)

$$P(x_n \mid x_1, x_2, \cdots, x_{n-1})$$



Language model (Transformers)

The capital city of Ontario is

x_1 x_2 x_{n-1}

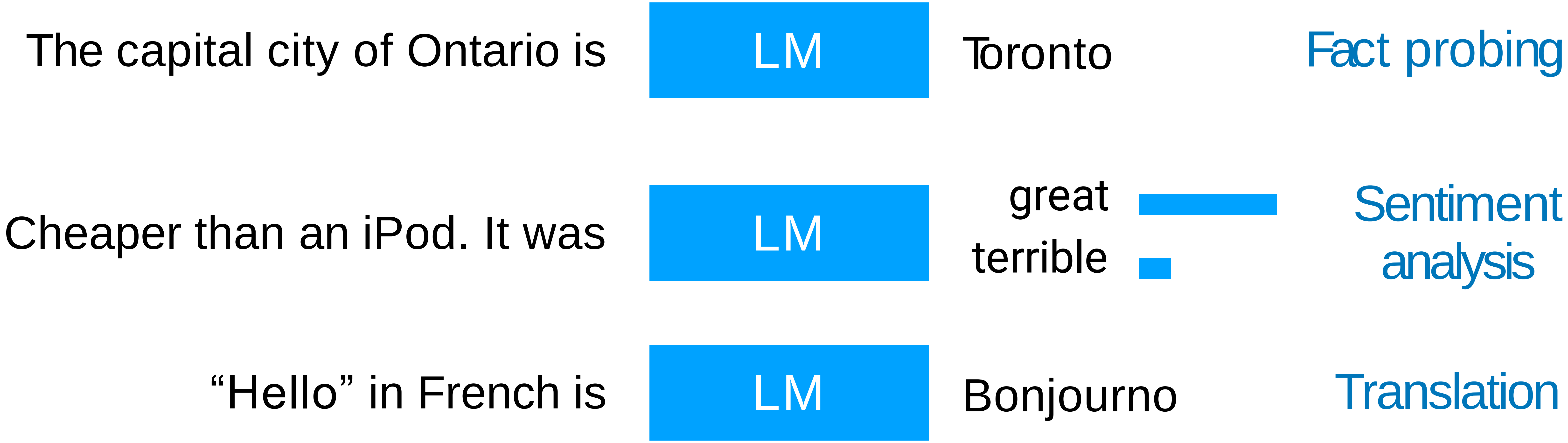
A language model (LM): Prompting

The capital city of Ontario is | LM | Toronto

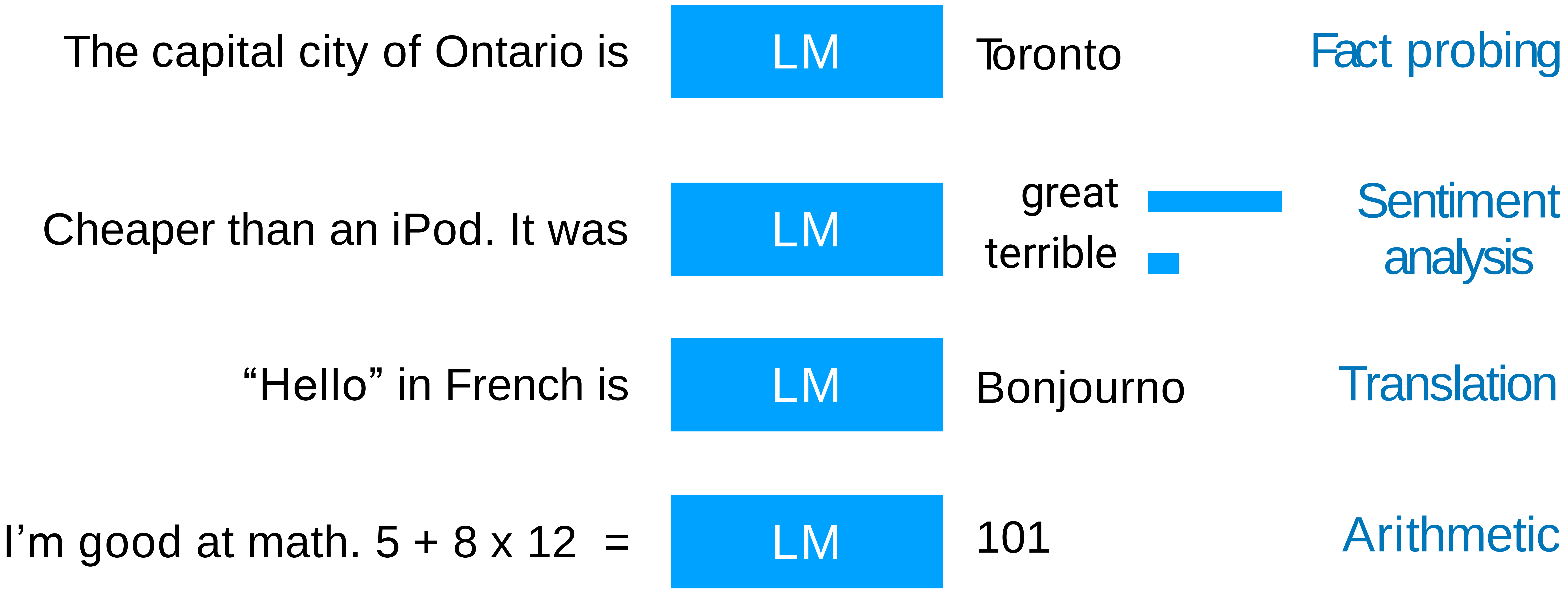
Fact probing

A language model (LM): Prompting

A language model (LM): Prompting



A language model (LM): Prompting



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: Datastore

Input

Query

LM

Index

Datastore

Raw text corpus

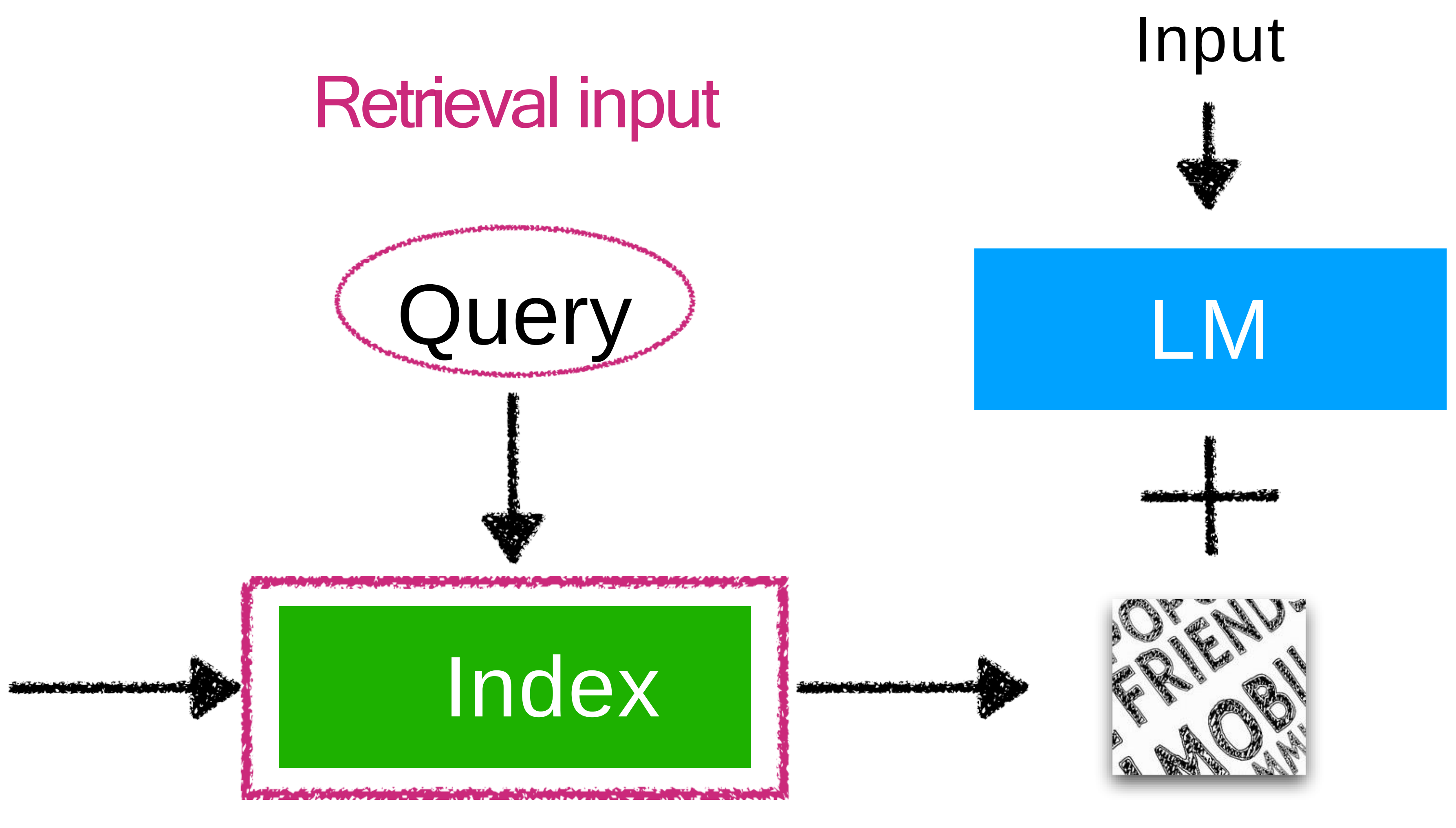
At least billions~ trillions of tokens

Not labeled datasets

Not structured data (knowledge bases)

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

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

# of total docs

# of docs containing i

# of occurrences of i in j

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

$$\mathrm{sim}(i, j) = \mathrm{tf}_{i,j} \times \log \frac{N}{\mathrm{df}_i}$$

# of total docs

# of docs containing i

# of occurrences of i in *j*

An entire field of study on how to get (or learn) the similarity function better

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 $\underbrace{\text{argTop-}k_{d\in\mathcal{D}}\text{sim}(q,d)}_{k \text{ elements from a datastore}}$ 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

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

Index: given q, return $\underbrace{\text{argTop-}k_{d \in \mathcal{D}} \text{sim}(q, d)}_{k \text{ elements from a datastore}}$ through fast nearest neighbor search

Inference: Search

Input

LM

+

Query

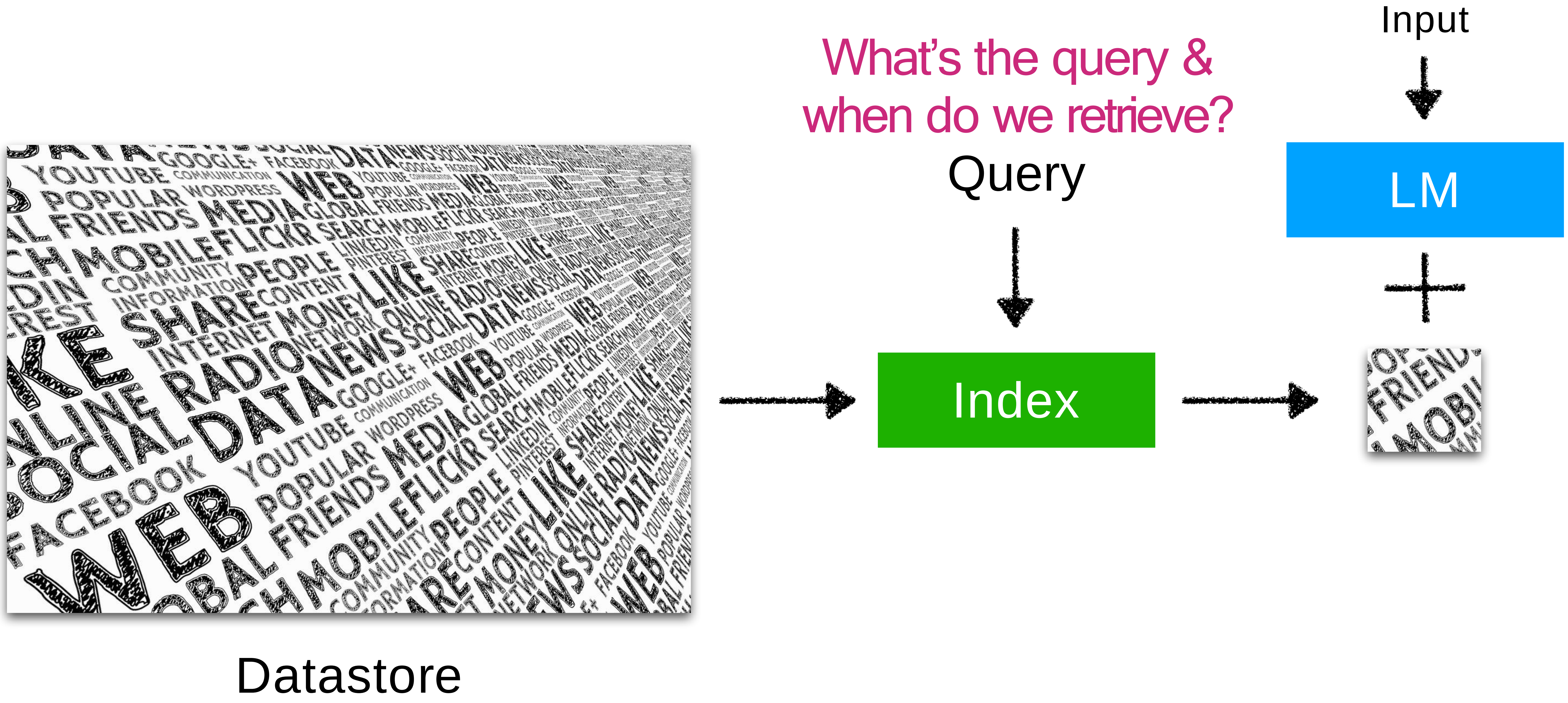
Index

fast nearest neighbor search

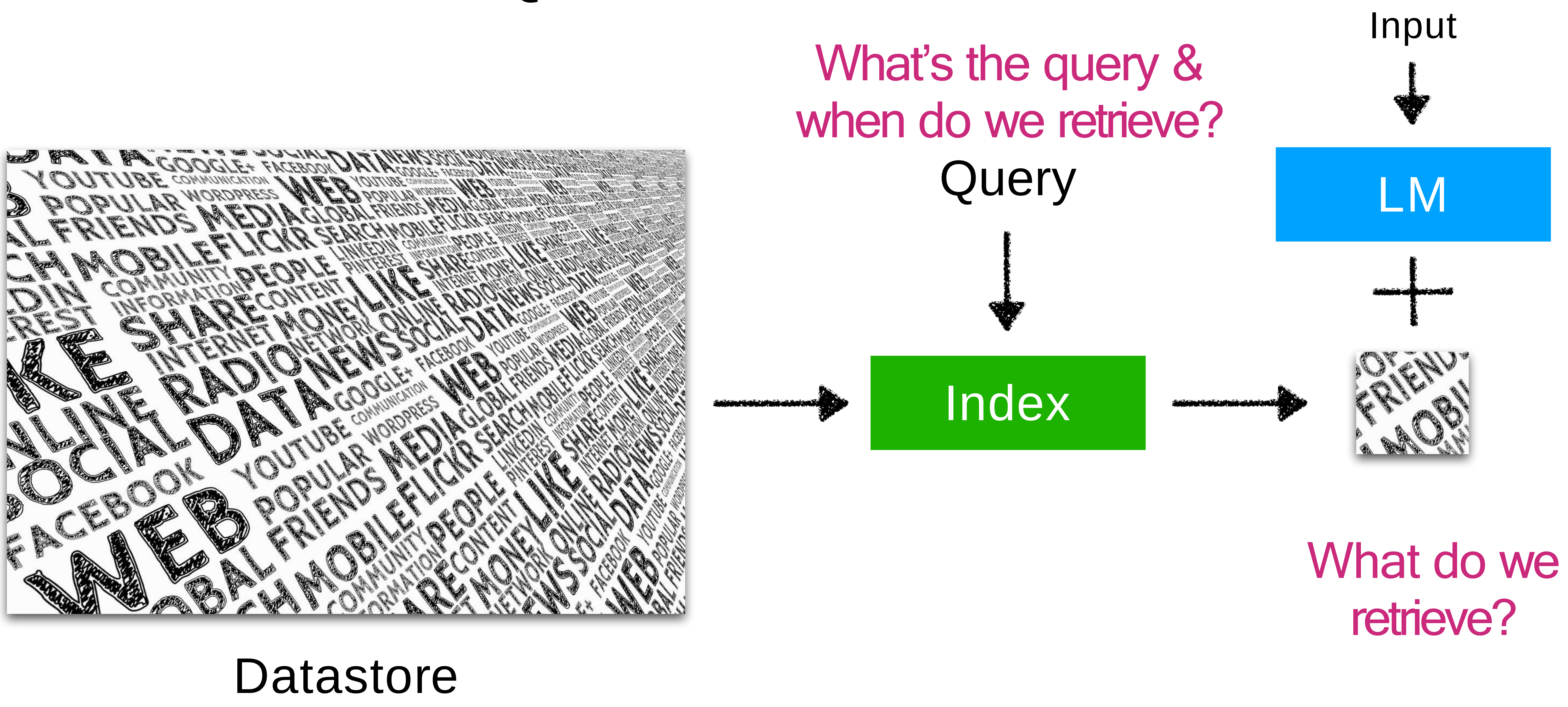
In this tutorial, we assume we can do it fast & accurate

Datastore

Questions to answer



Questions to answer



Questions to answer

Input

What's the query &
when do we retrieve?

Query

LM

Datastore

Index

How do we
use retrieval?

What do we
retrieve?

Retrieval-based LM: Architecture

Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?

Tokens?

Something else?

Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?

Input

LM

Output

Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?

Tokens?

Something else?

How to use retrieval?

Input

LM

Output

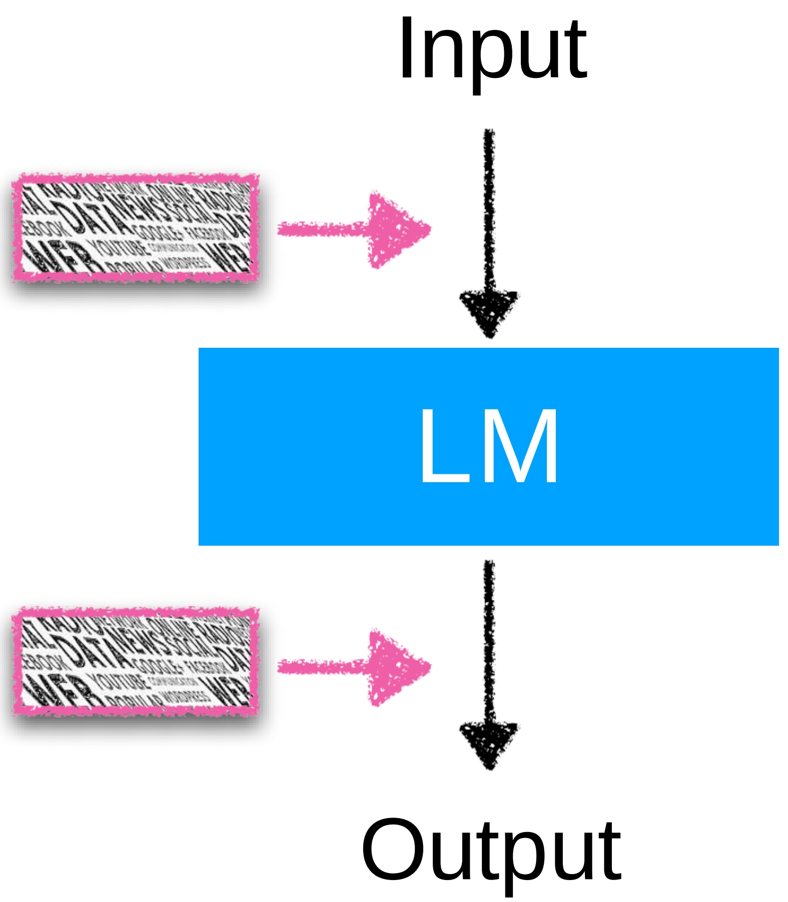
Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



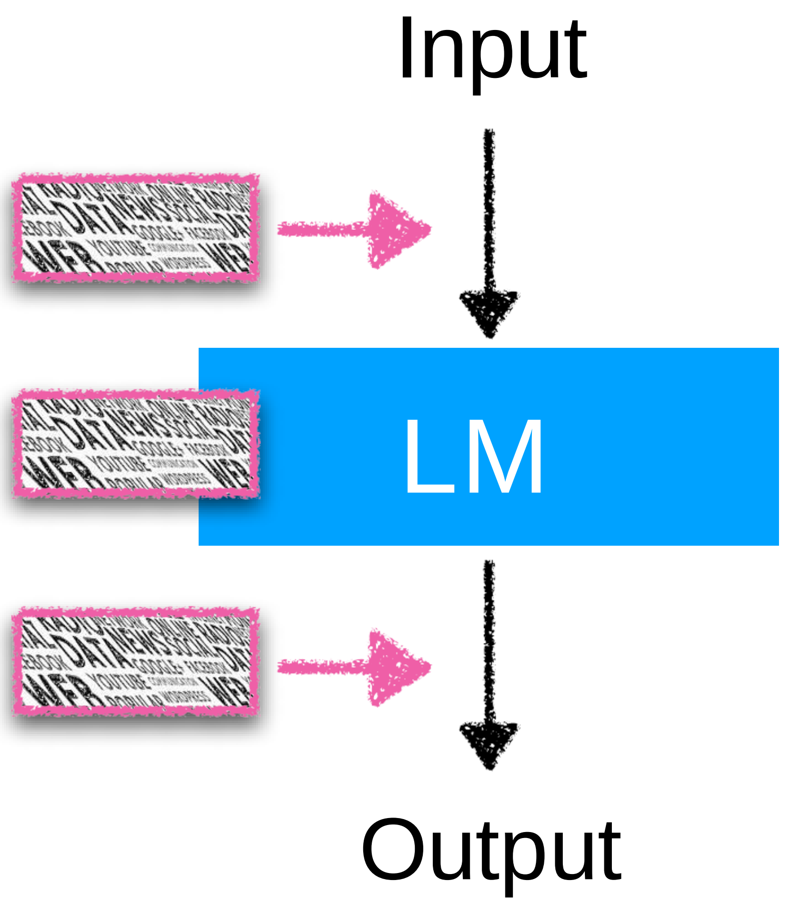
Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



Categorization of retrieval-based LMs

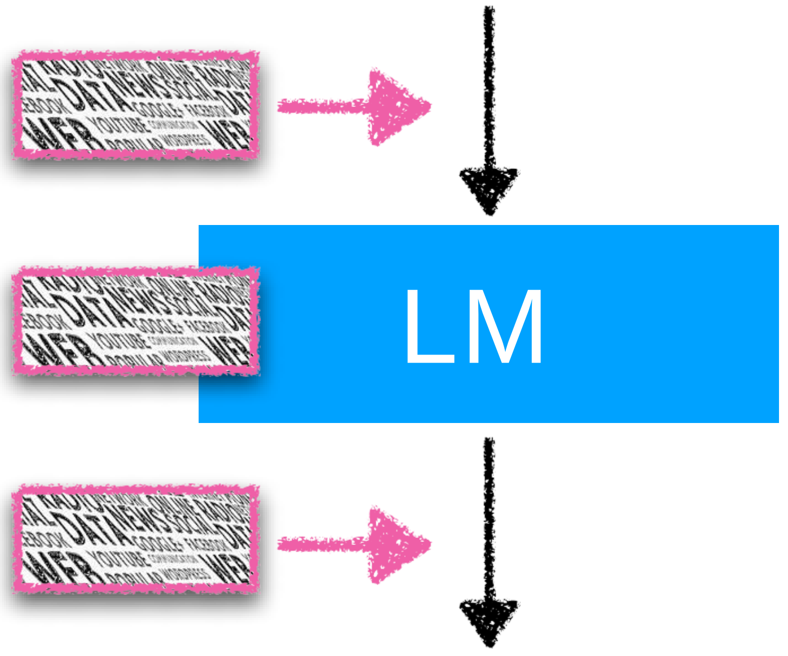
What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?

Input



Output

When to retrieve?

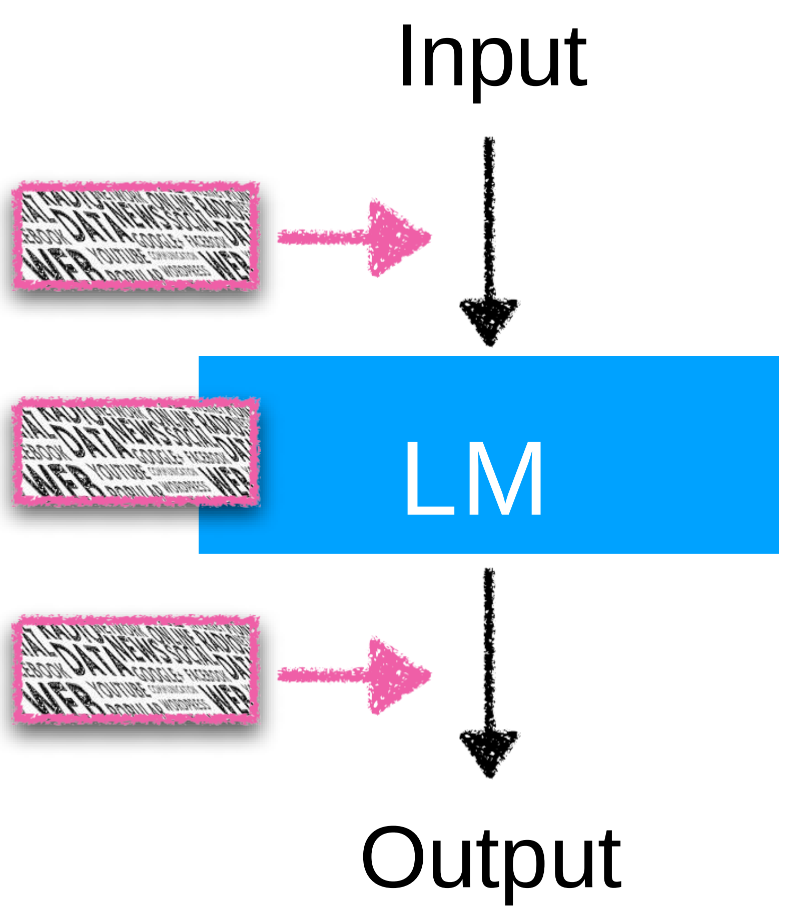
Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



When to retrieve?

w/ retrieval

The capital city of Ontario is Toronto.

Categorization of retrieval-based LMs

What to retrieve?

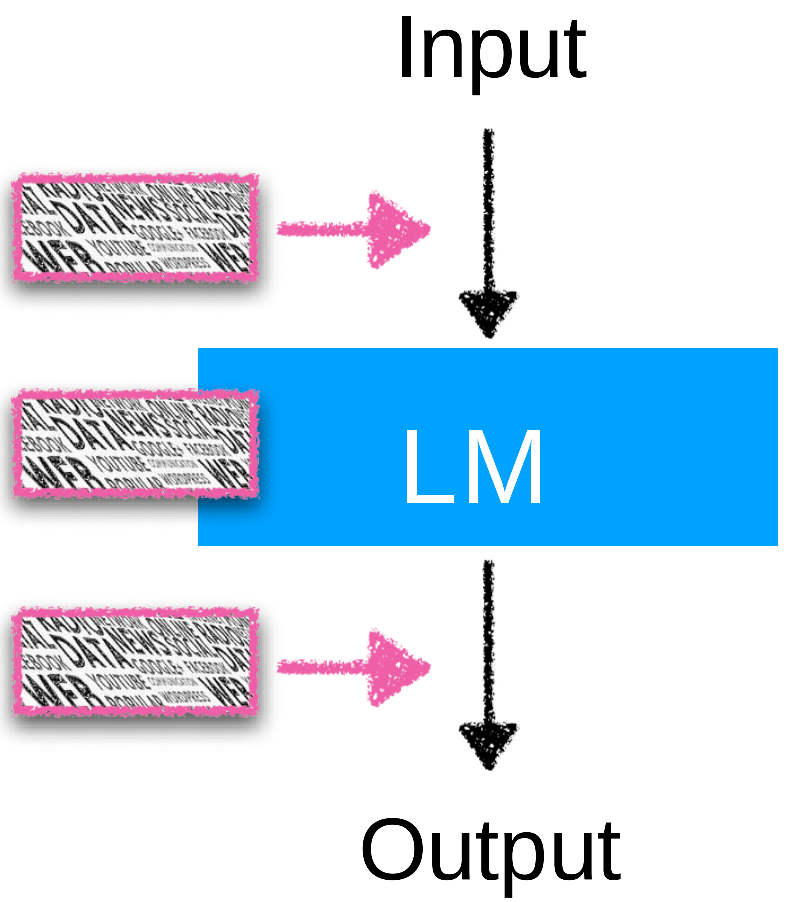
Query

Text chunks (passages)?

Tokens?

Something else?

How to use retrieval?



When to retrieve?

w/ retrieval

The capital city of Ontario is Toronto.

w/ retrieval w/ r w/r w/r w/ r w/r w/r

The capital city of Ontario is Toronto.

Categorization of retrieval-based LMs

What to retrieve?

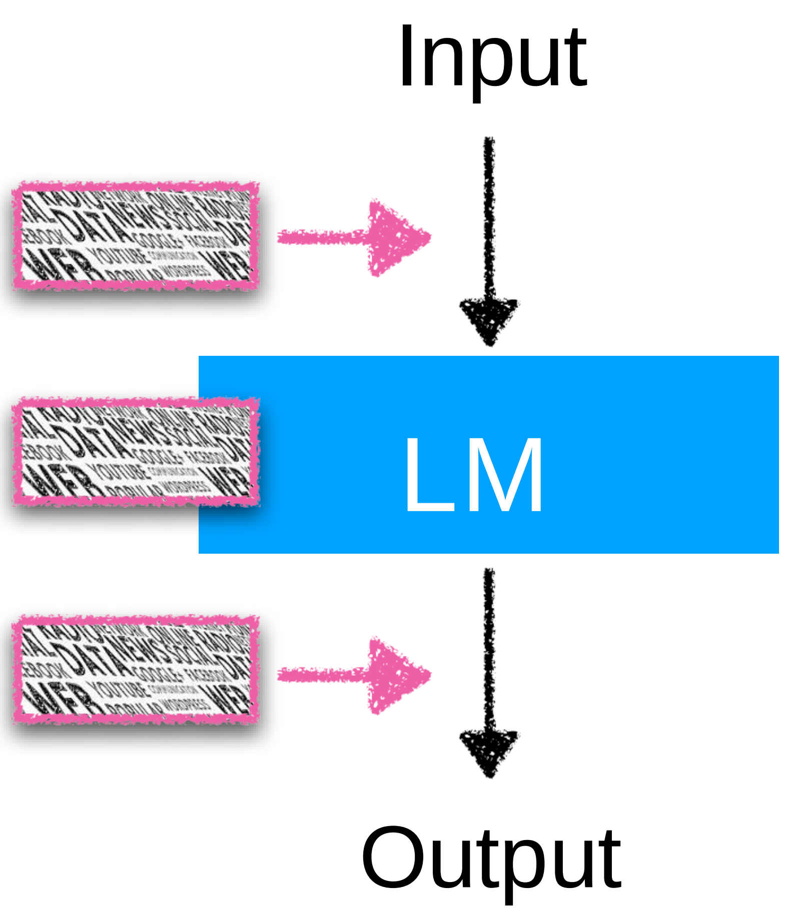
Query

Text chunks (passages)?

Tokens?

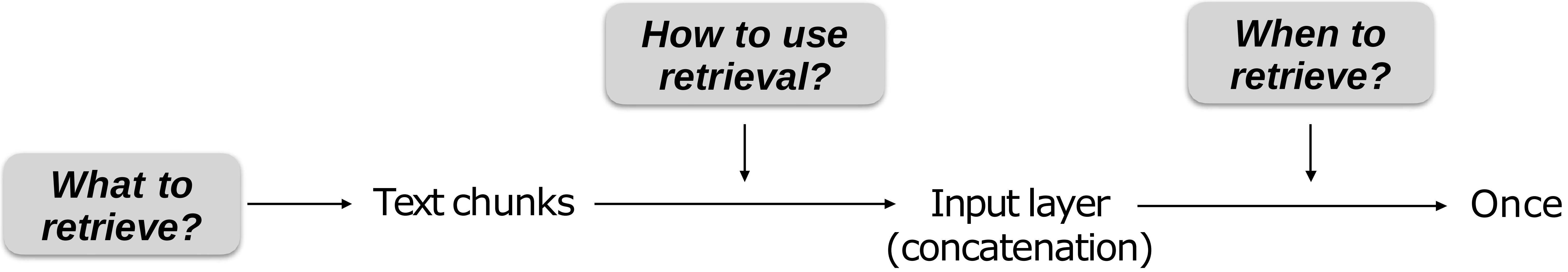
Something else?

How to use retrieval?



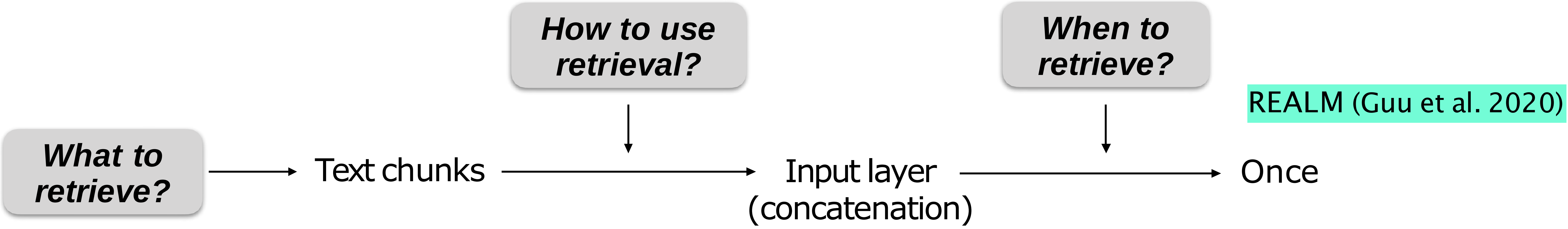
When to retrieve?

Roadmap

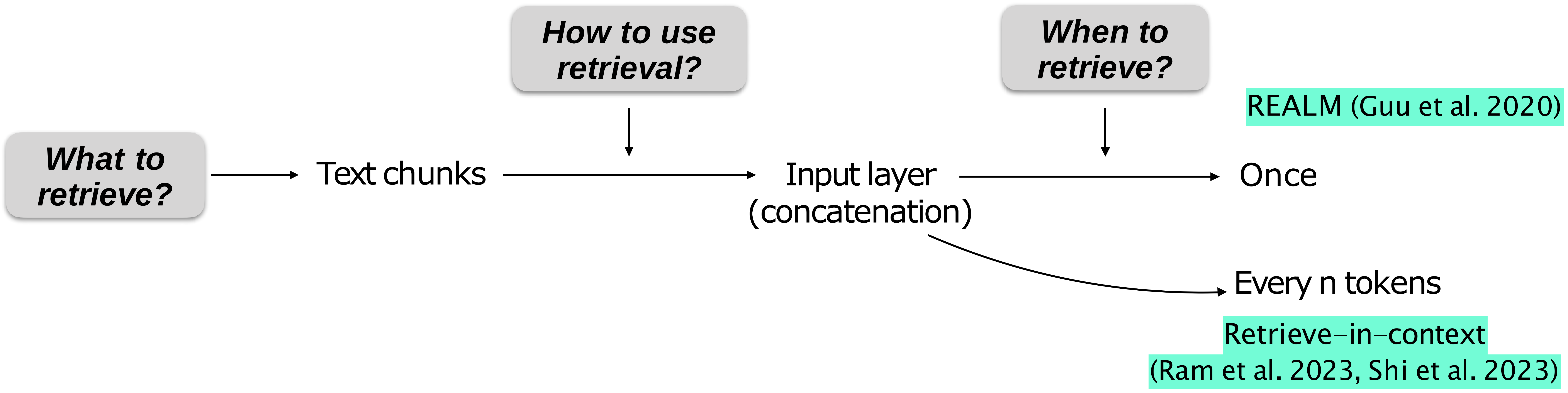


How to use retrieval?
When to retrieve?
What to retrieve?
Text chunks
Input layer (concatenation)
Once

Roadmap

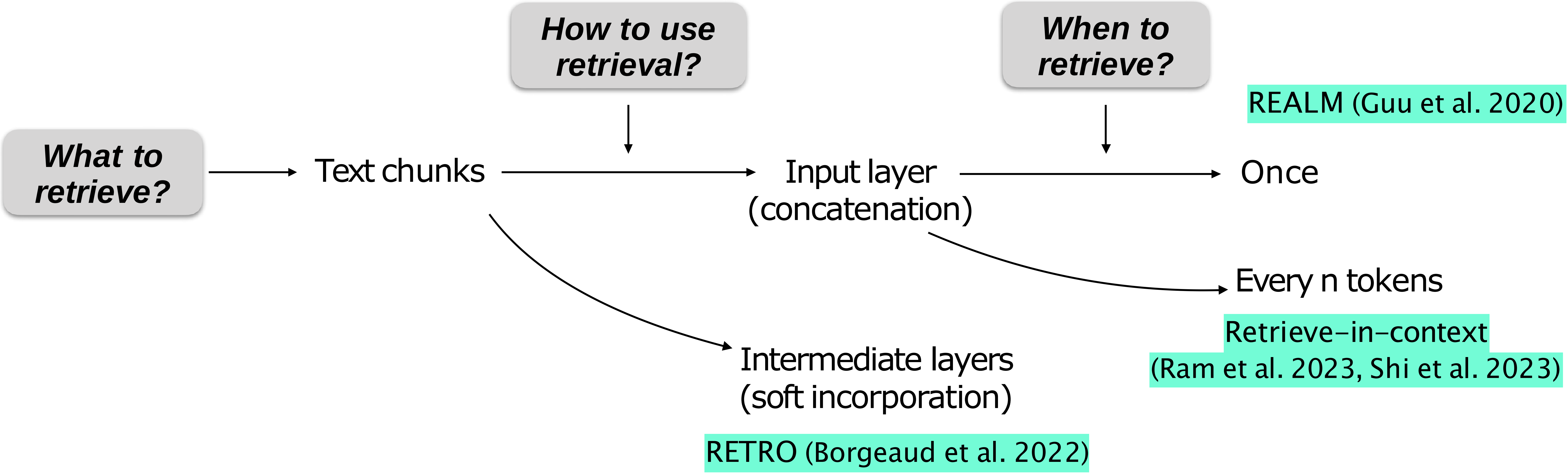

How to use retrieval?
When to retrieve?
REALM (Guu et al. 2020)
What to retrieve?
Text chunks
Input layer (concatenation)
Once

Roadmap



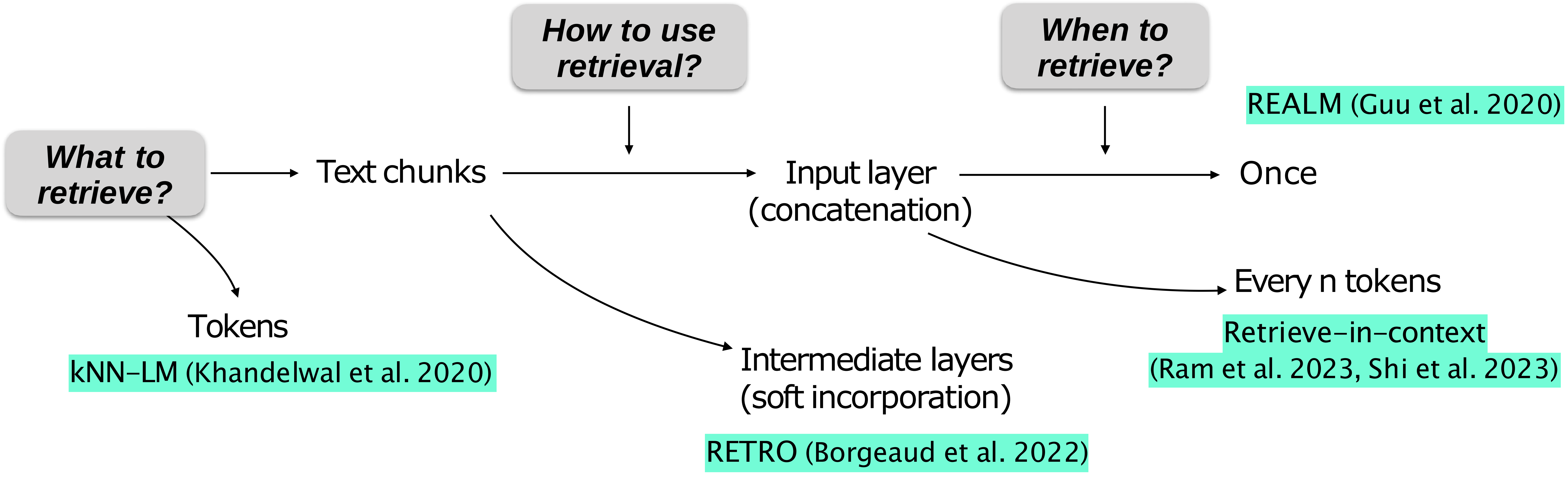
How to use retrieval?
When to retrieve?
REALM (Guu et al. 2020)
What to retrieve?
Text chunks
Input layer (concatenation)
Once
Every n tokens
Retrieve-in-context
(Ram et al. 2023, Shi et al. 2023)

Roadmap

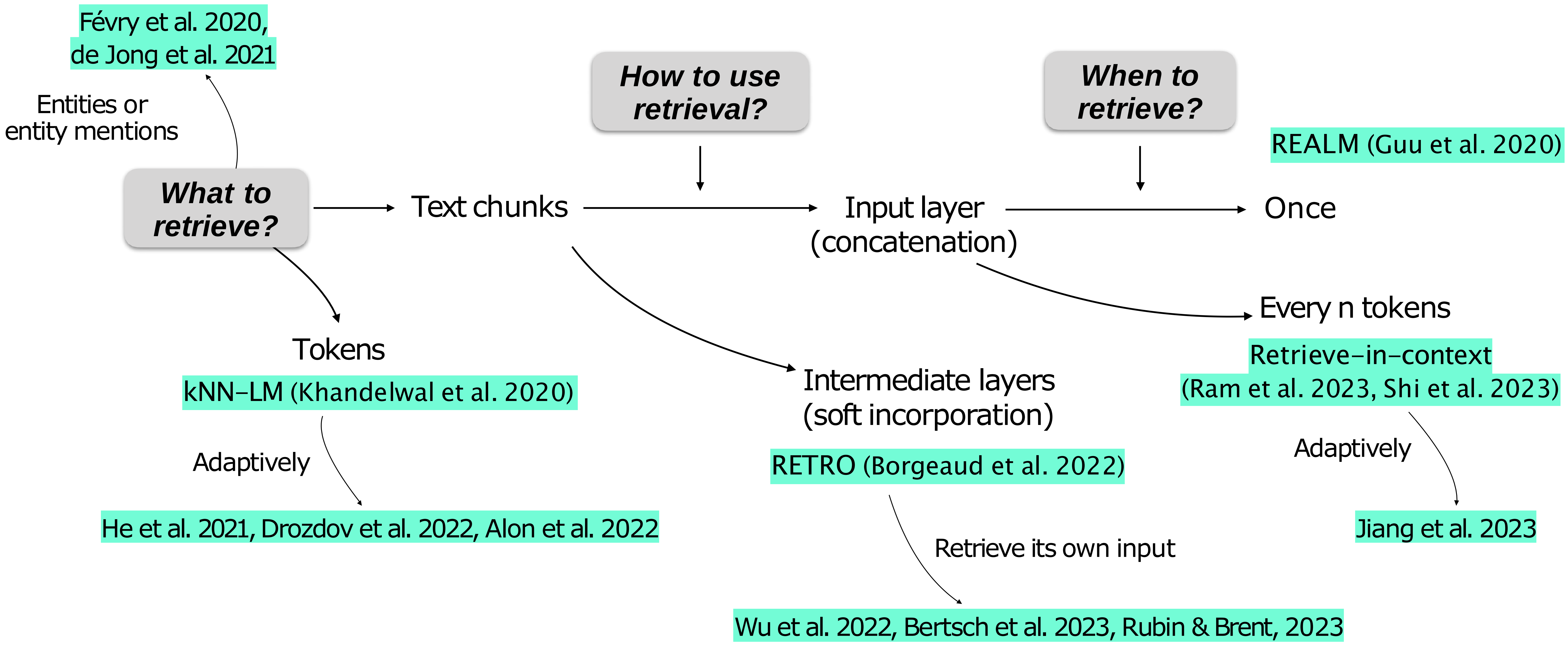


What to retrieve?
Text chunks
How to use retrieval?
Input layer (concatenation)
Intermediate layers (soft incorporation)
RETRO (Borgeaud et al. 2022)
When to retrieve?
REALM (Guu et al. 2020)
Once
Every n tokens
Retrieve-in-context (Ram et al. 2023, Shi et al. 2023)

Roadmap


How to use retrieval?
When to retrieve?
REALM (Guu et al. 2020)
What to retrieve?
Text chunks
Input layer (concatenation)
Once
Every n tokens
Retrieve-in-context
(Ram et al. 2023, Shi et al. 2023)
Tokens
kNN-LM (Khandelwal et al. 2020)
Intermediate layers (soft incorporation)
RETRO (Borgeaud et al. 2022)

Roadmap



Févry et al. 2020,
de Jong et al. 2021
Entities or
entity mentions
How to use
retrieval?
When to
retrieve?
REALM (Guu et al. 2020)
What to
retrieve?
Text chunks
Input layer
(concatenation)
Once
Every n tokens
Tokens
Retrieve-in-context
(Ram et al. 2023, Shi et al. 2023)
kNN-LM (Khandelwal et al. 2020)
Intermediate layers
(soft incorporation)
Adaptively
Adaptively
RETRO (Borgeaud et al. 2022)
He et al. 2021, Drozdov et al. 2022, Alon et al. 2022
Jiang et al. 2023
Retrieve its own input
Wu et al. 2022, Bertsch et al. 2023, Rubin & Brent, 2023

REALM (Guu et al 2020)

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

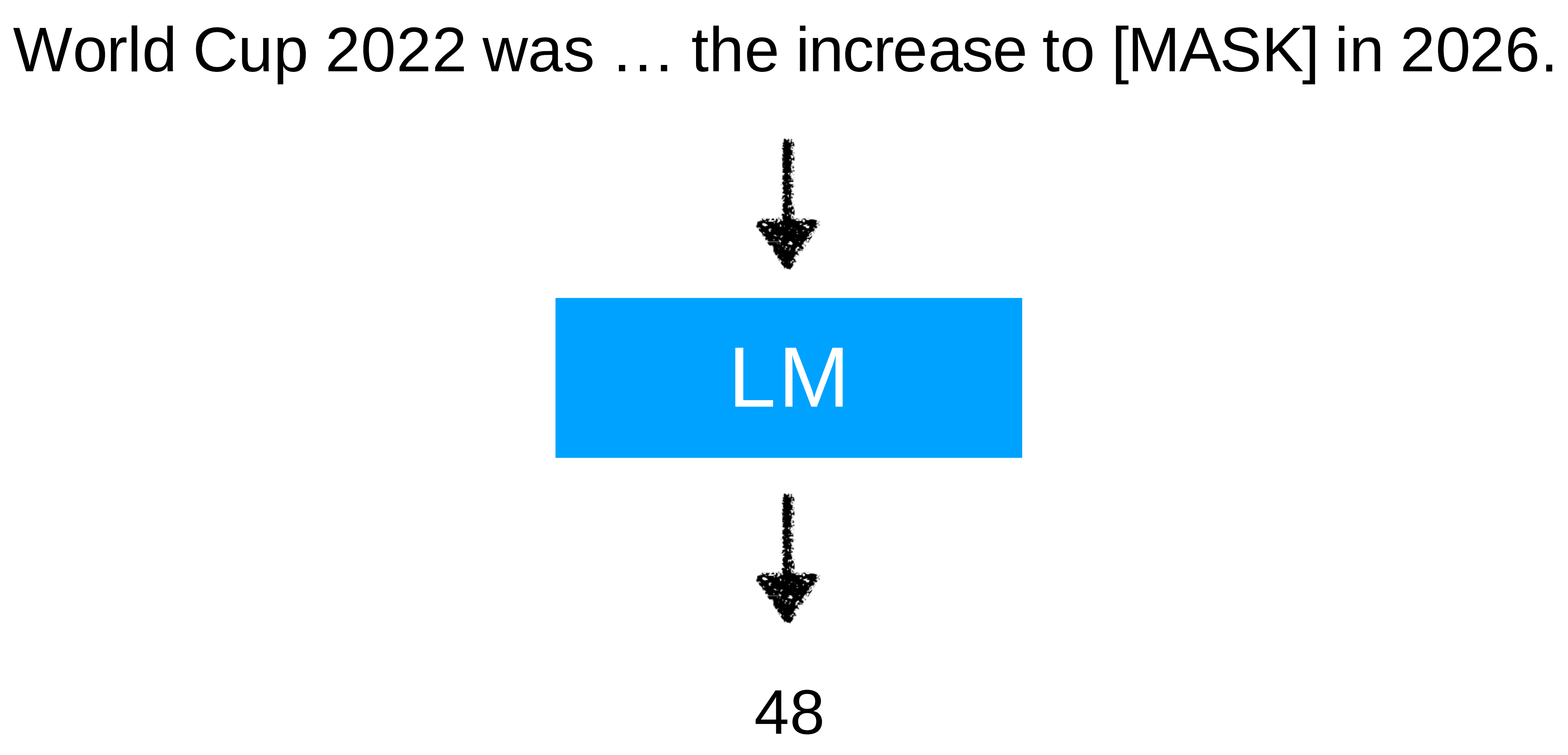
REALM (Guu et al 2020)

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

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

REALM (Guu et al 2020)

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



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

REALM (Guu et al 2020)

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

***q* (=*x*)**

Retrieval

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

LM

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

REALM (Guu et al 2020)

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

***q* (=*x*)**

Retrieval

k chunks of text (passages)

FIFA World Cup 2026 will expand to 48 teams.

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

LM

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

REALM (Guu et al 2020)

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

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

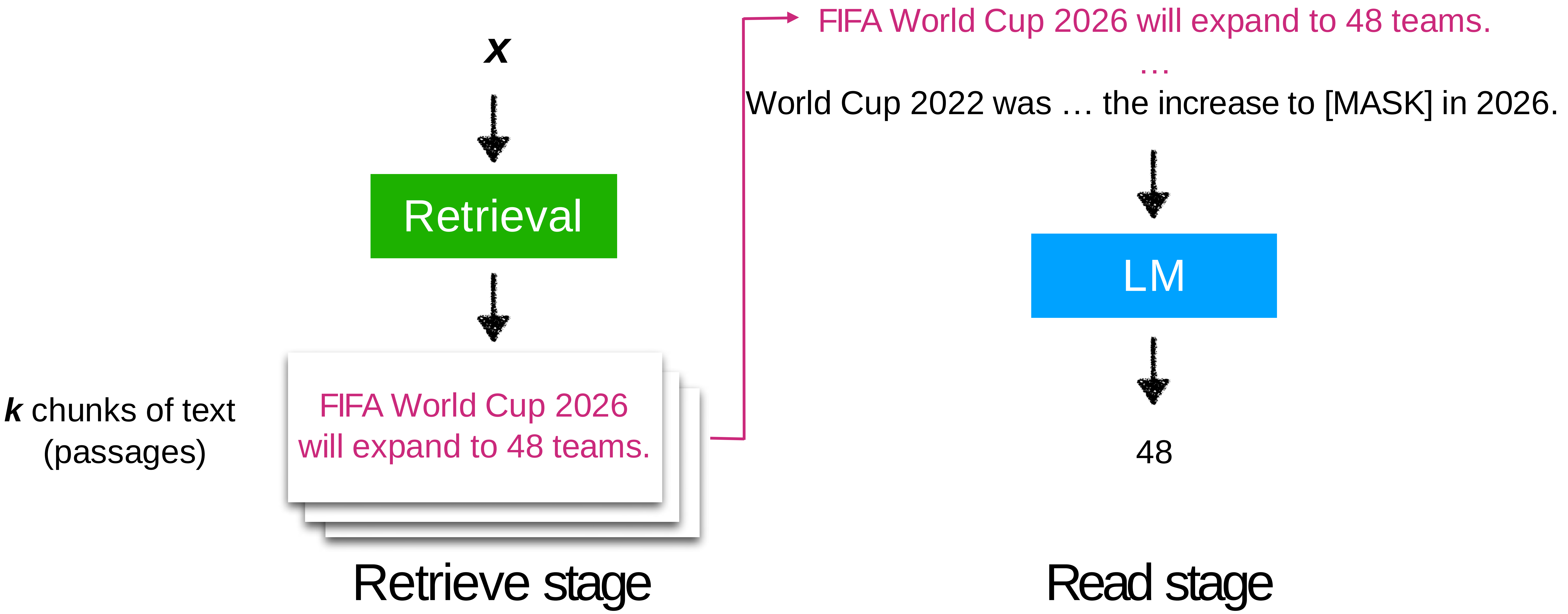
REALM (Guu et al 2020)

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

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

REALM (Guu et al 2020)

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



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

REALM: (1) Retrieve stage

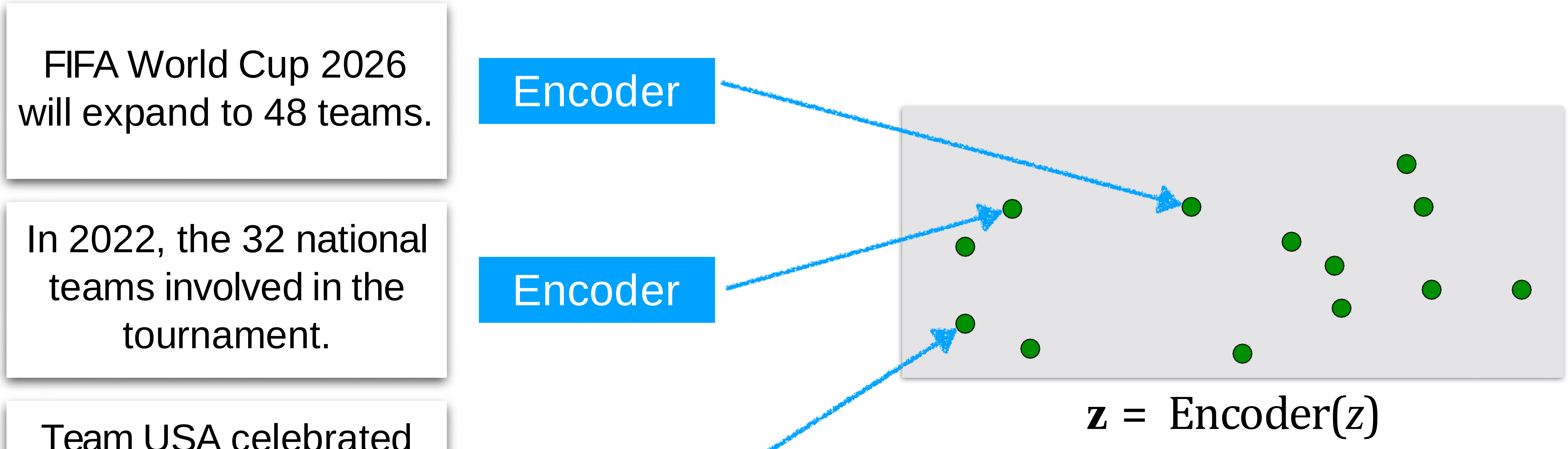
FIFA World Cup 2026 will expand to 48 teams.

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: (1) Retrieve stage



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

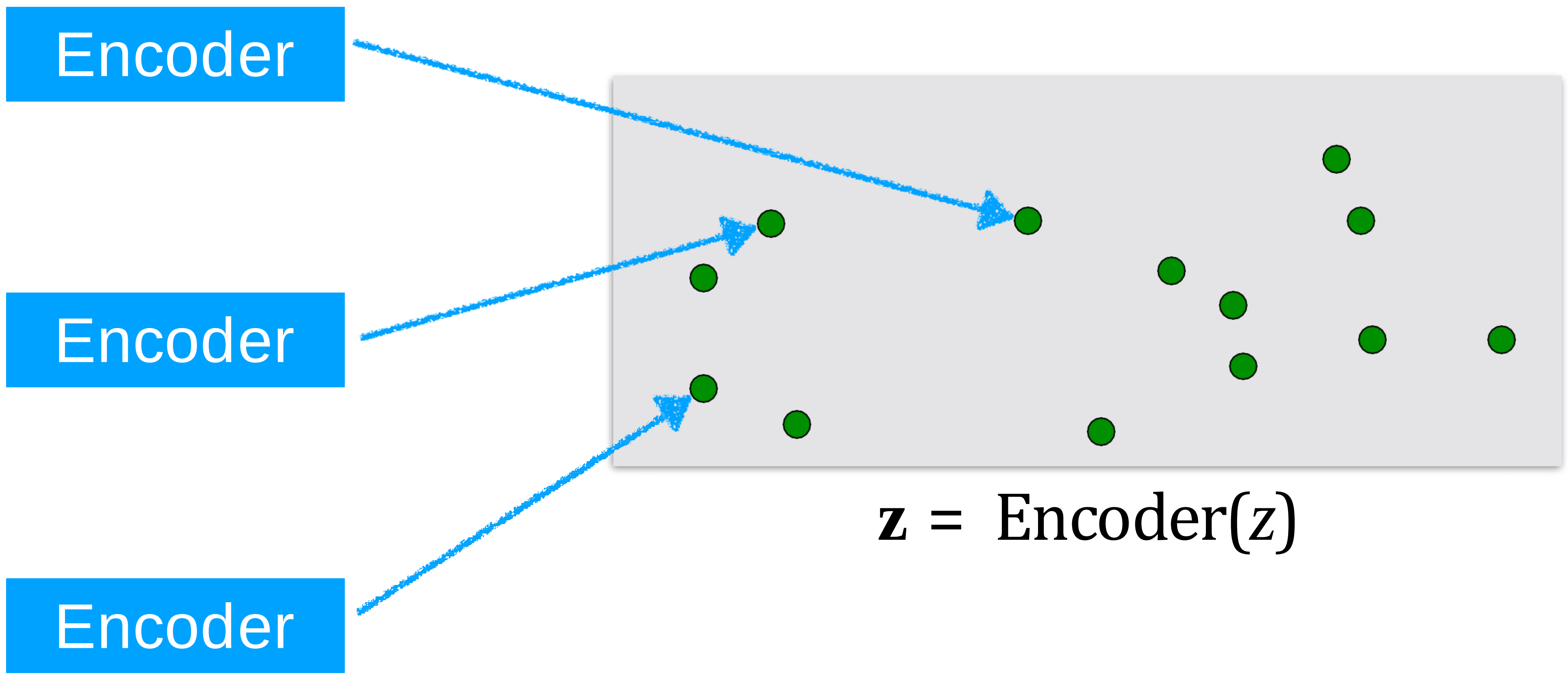
REALM: (1) Retrieve stage

$\boldsymbol{x}$ = 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 …



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

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

REALM: (1) Retrieve stage

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

Encoder

FIFA World Cup 2026 will expand to 48 teams.

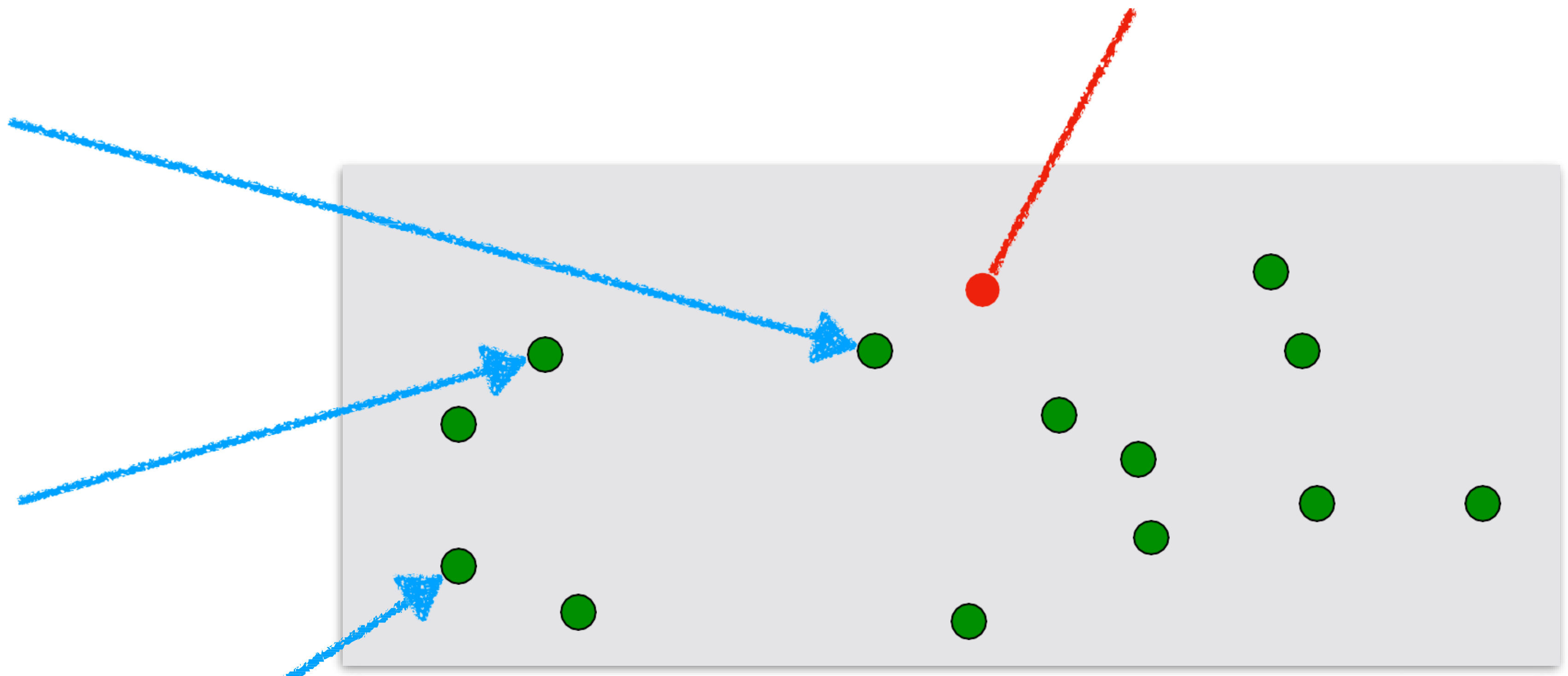
Encoder

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

Encoder

Team USA celebrated after winning its match against Iran …

Encoder



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

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

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

REALM: (1) Retrieve stage

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

Encoder

Fast nearest neighbor search

FIFA World Cup 2026 will expand to 48 teams.

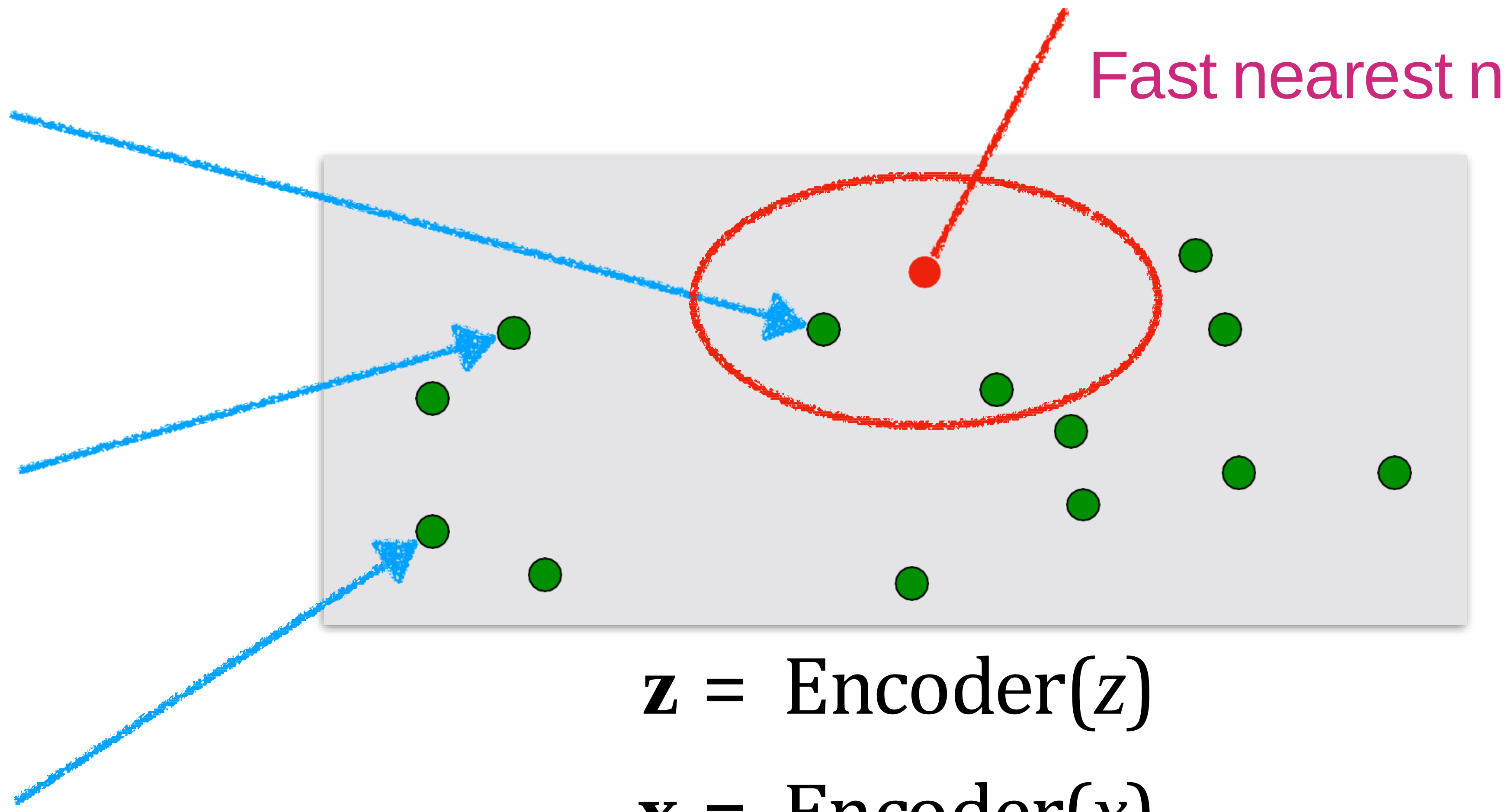
Encoder

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

Encoder

Team USA celebrated after winning its match against Iran …

Encoder



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

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

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

REALM: (1) Retrieve stage

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

Encoder

Fast nearest neighbor search

FIFA World Cup 2026 will expand to 48 teams.

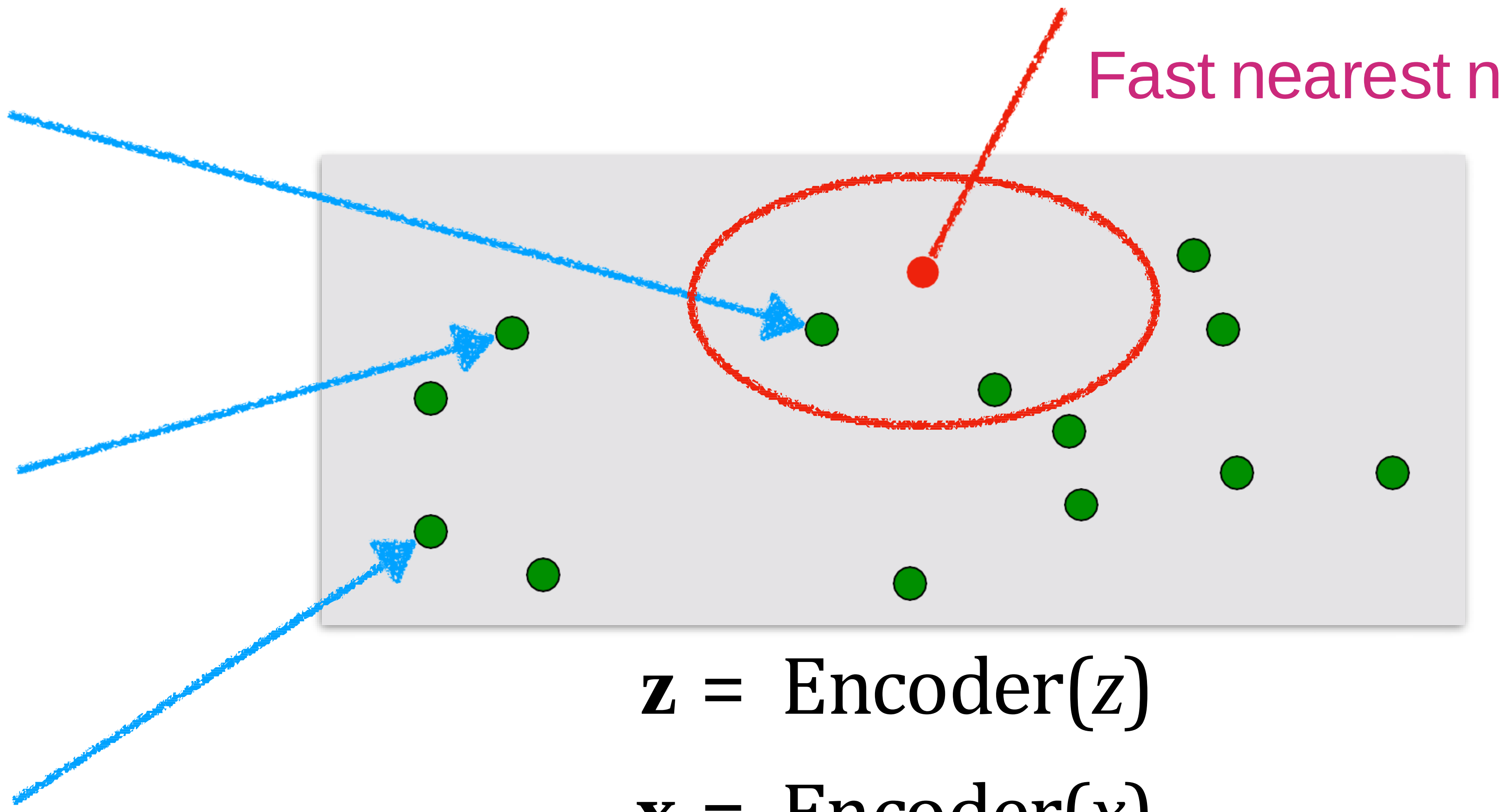
Encoder

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

Encoder

Team USA celebrated after winning its match against Iran …

Encoder



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

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

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

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

k retrieved chunks

REALM: (2) Read stage

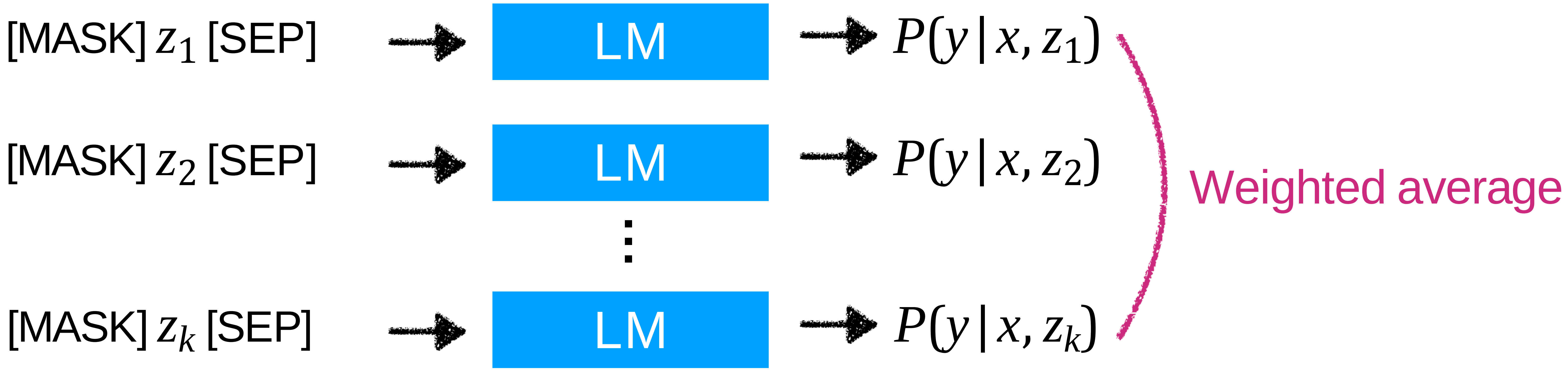
[MASK] z_1 [SEP] $\rightarrow$ LM $\rightarrow P(y \mid x, z_1)$

[MASK] z_2 [SEP] $\rightarrow$ LM $\rightarrow P(y \mid x, z_2)$

$\vdots$

[MASK] z_k [SEP] $\rightarrow$ LM $\rightarrow P(y \mid x, z_k)$

REALM: (2) Read stage



REALM: (2) Read stage

[MASK] z_1 [SEP] → LM → $P(y|x, z_1)$

[MASK] z_2 [SEP] → LM → $P(y|x, z_2)$

⋮

[MASK] z_k [SEP] → LM → $P(y|x, z_k)$

Weighted average

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

REALM (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

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

↓

FIFA World Cup 2026 will expand to 48 teams.

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

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

↓

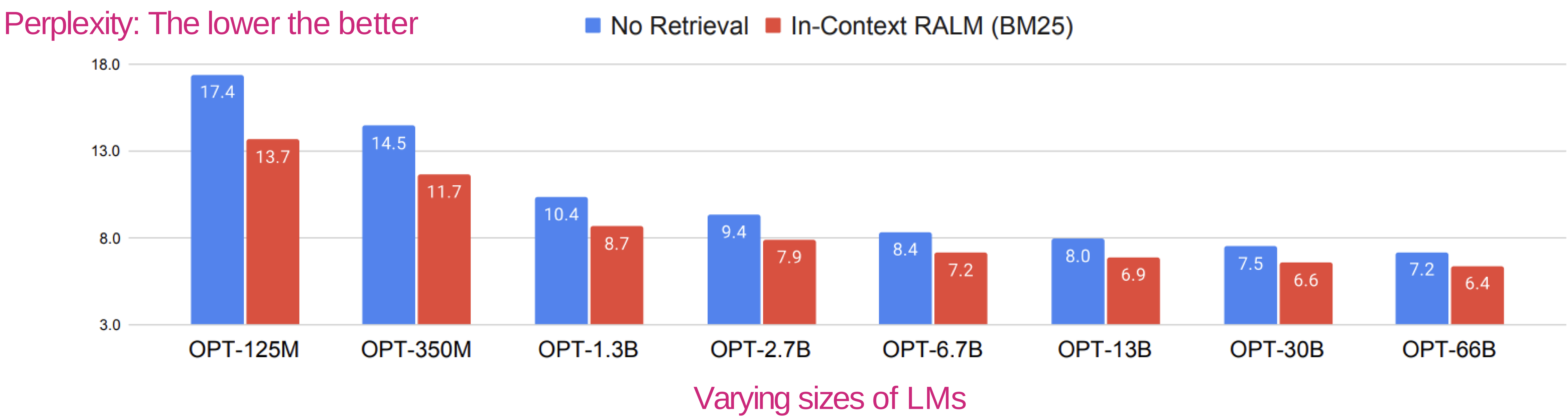
LM

↓

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



Retrieval helps over all sizes of LMs

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

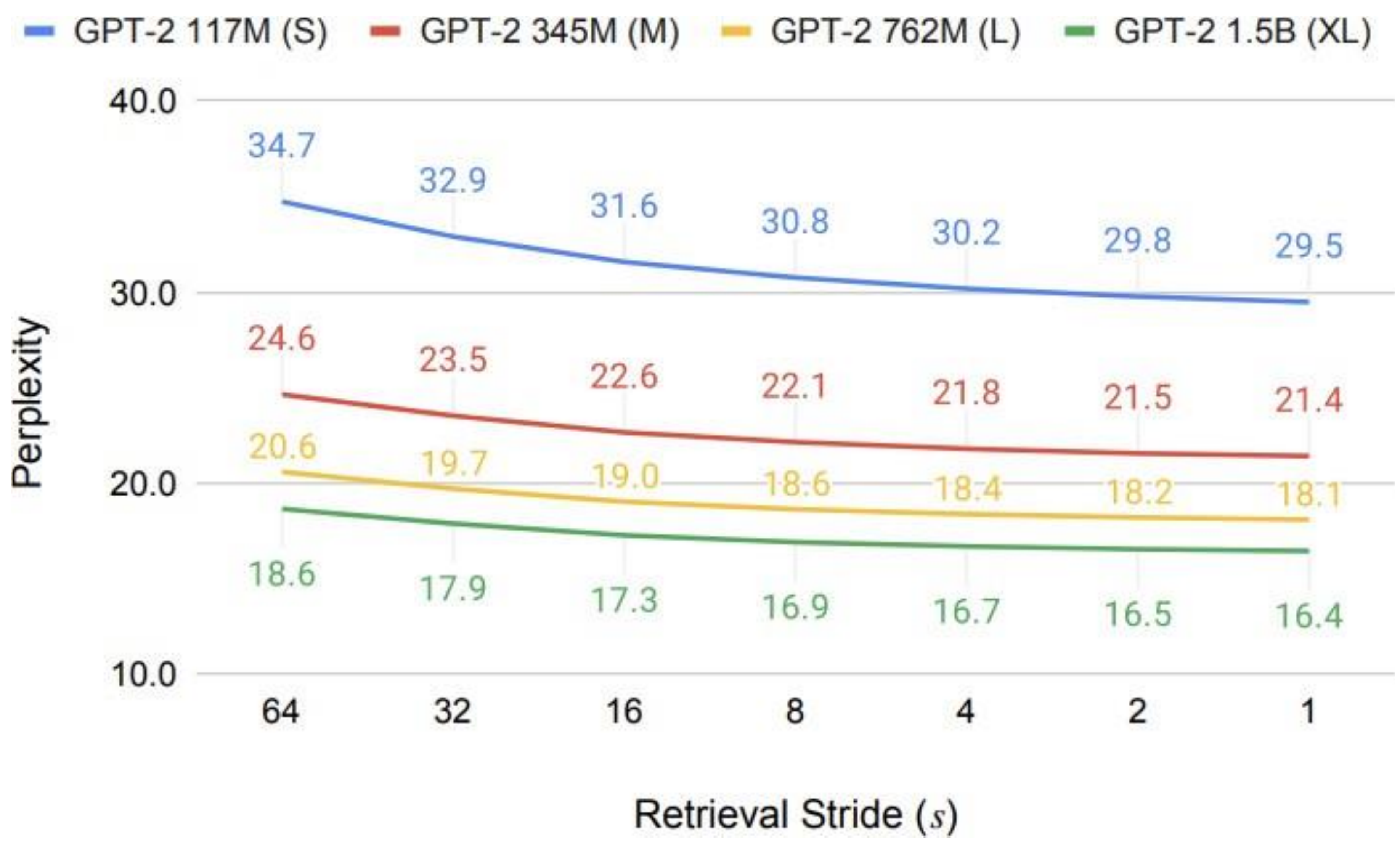
FIFA World Cup 2026 will expand to 48 teams. World Cup 2022 was the last with 32 teams, before the increase

LM

to 48 in the 2026 tournament.

Retrieval results from a new query explain them!

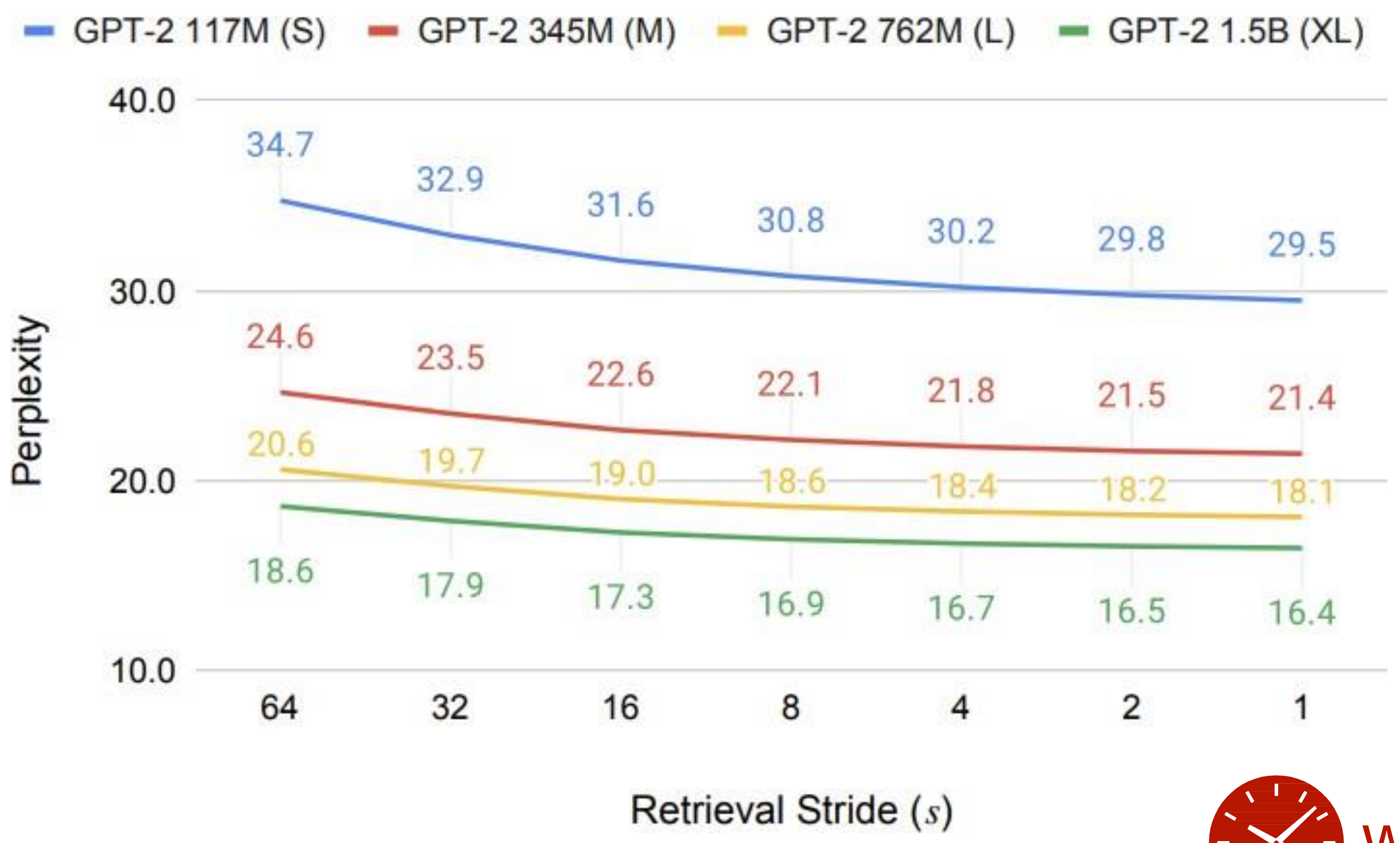
Retrieval-in-context LM



Retrieving more frequently helps

Graphs from Ram et al. 2023

Retrieval-in-context LM



with cost in inference time

Retrieving more frequently helps

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

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 <u>many</u> text chunks, <u>frequently</u>

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*

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*

- ✓ Scale the datastore (1.8T tokens)

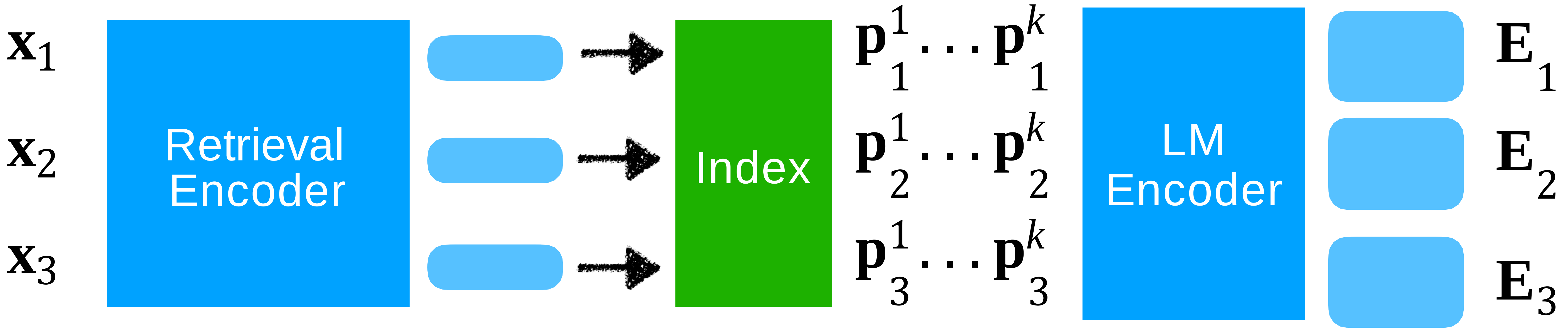
Borgeaud et al. 2021. “Improving language models by retrieving from trillions of tokens”

RETRO (Borgeaud et al. 2021)

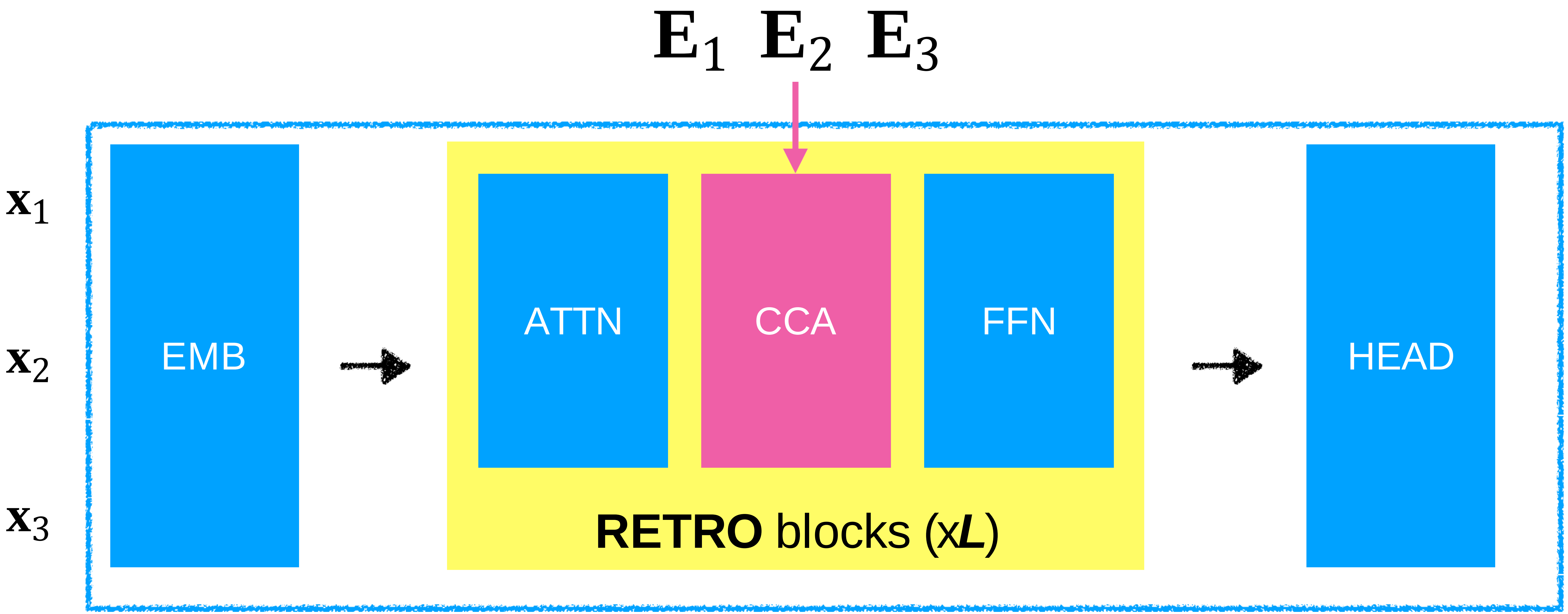
$\boldsymbol{x}$ = World Cup 2022 was the last with 32 teams, before the increase to

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

(k chunks of text per split)



Decoder in RETRO



Chunked Cross Attention (CCA)

Results

Perplexity: The lower the better

Model	Retrieval Set	#Database tokens	#Database keys	Valid	Test
Adaptive Inputs (Baevski and Auli, 2019)	-	-	-	17.96	18.65
SPALM (Yogatama et al., 2021)	Wikipedia	3B	3B	17.20	17.60
*k*NN-LM (Khandelwal et al., 2020)	Wikipedia	3B	3B	16.06	16.12
Megatron (Shoeybi et al., 2019)	-	-	-	-	10.81
Baseline transformer (ours)	-	-	-	21.53	22.96
*k*NN-LM (ours)	Wikipedia	4B	4B	18.52	19.54
RETRO	Wikipedia	4B	0.06B	18.46	18.97
RETRO	C4	174B	2.9B	12.87	10.23
RETRO	MassiveText (1%)	18B	0.8B	18.92	20.33
RETRO	MassiveText (10%)	179B	4B	13.54	14.95
RETRO	MassiveText (100%)	1792B	28B	**3.21**	**3.92**

Significant improvements by retrieving from 1.8 trillion tokens

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	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?

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

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?

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

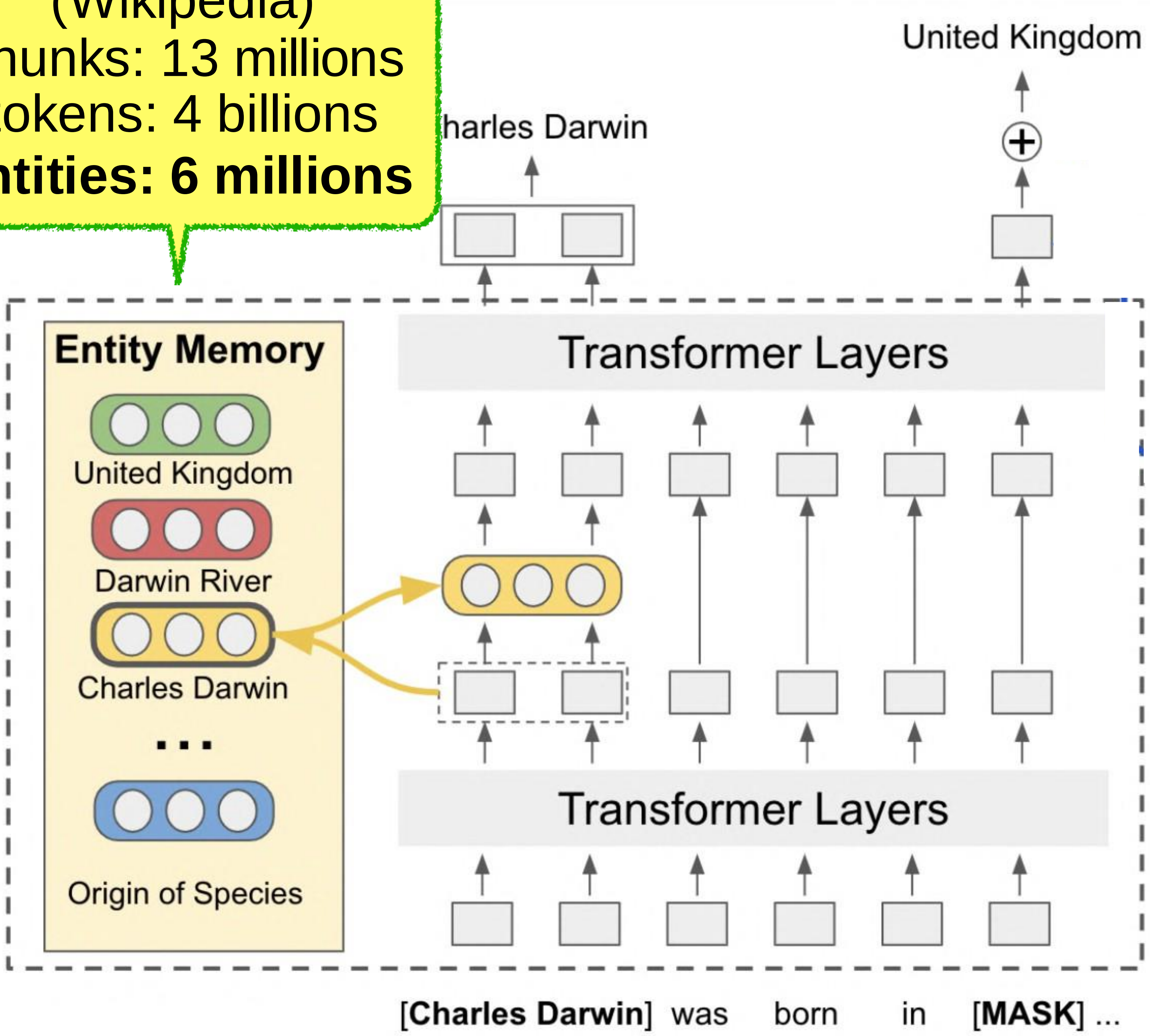
	What do retrieve?	How to use retrieval?	When to retrieve?
REALM (Guu et al 2020)	Text chunks	Input layer	Once
Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)	Text chunks	Input layer	Every n tokens
RETRO (Borgeaud et al. 2021)	Text chunks	Intermediate layers	Every n tokens
kNN-LM (Khandelwal et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

(Wikipedia)
chunks: 13 millions
tokens: 4 billions
entities: 6 millions

Need text with entity detected



Need entity linker

Fevry et al. 2020. "Entities as Experts: Sparse Memory Access with Entity Supervision"

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

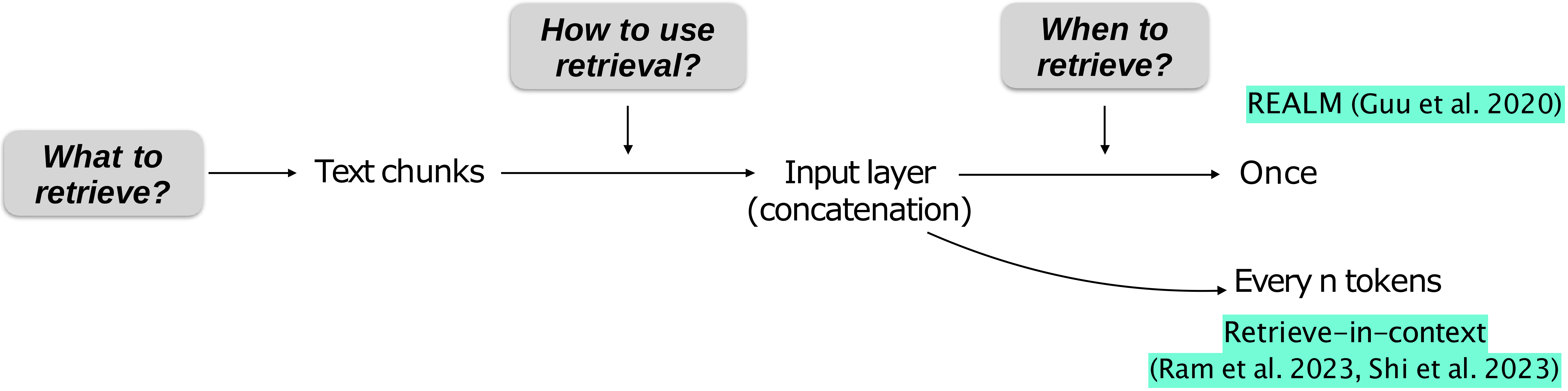
Summary

	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

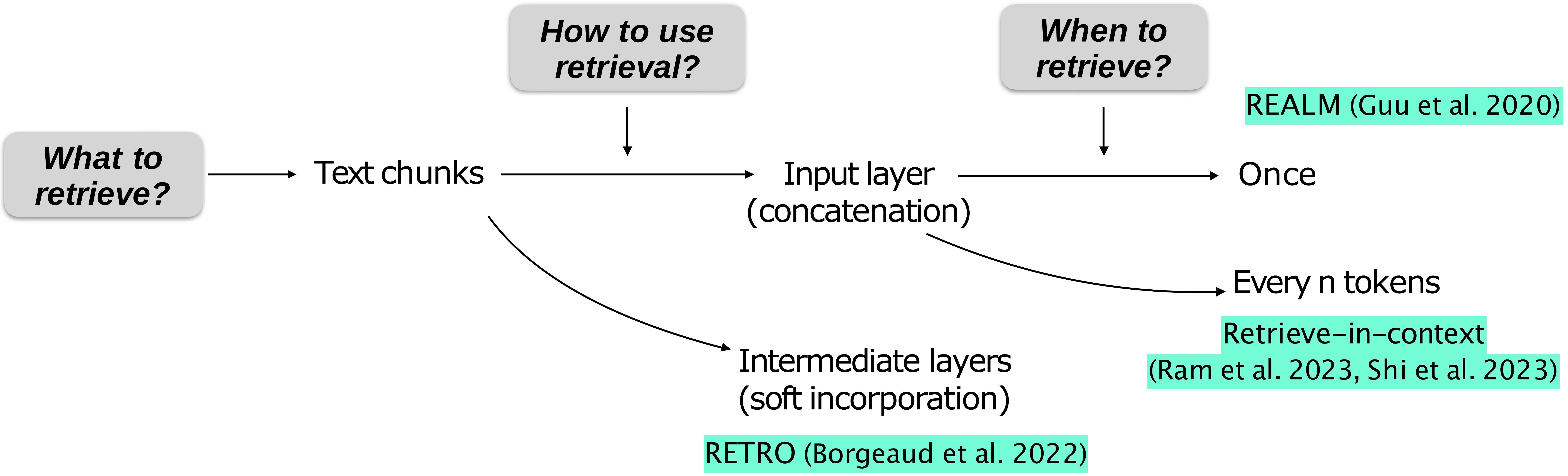
Additional entity detection required

Wrapping up

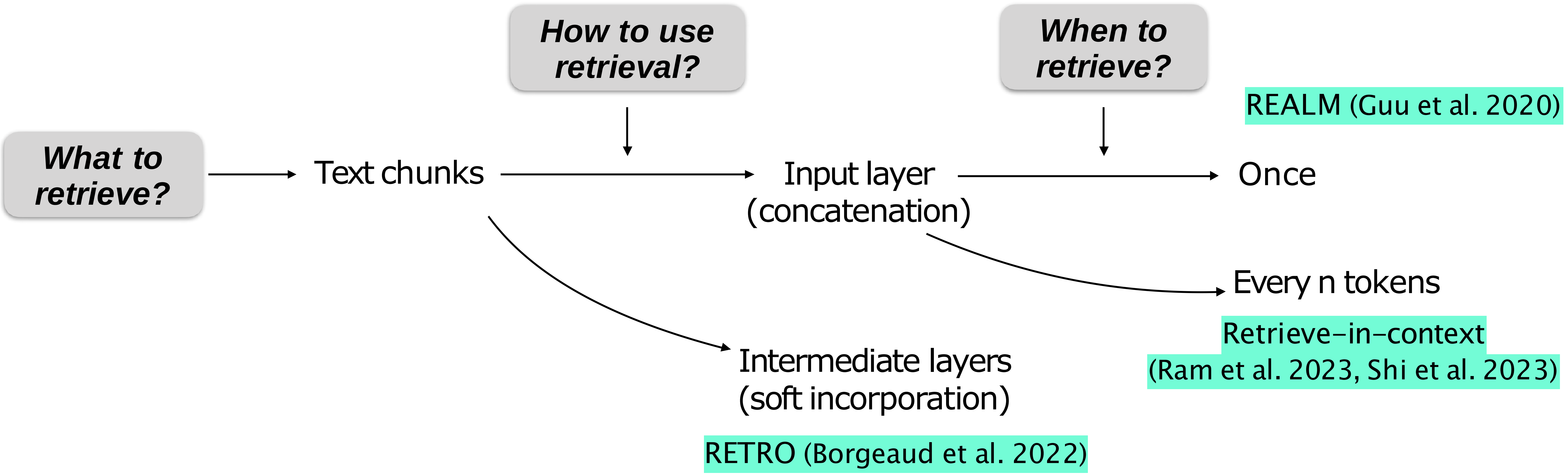


More frequent retrieval =better in performance, but slower

Wrapping up


What to retrieve?
Text chunks
How to use retrieval?
Input layer (concatenation)
Intermediate layers (soft incorporation)
RETRO (Borgeaud et al. 2022)
When to retrieve?
REALM (Guu et al. 2020)
Once
Every n tokens
Retrieve-in-context (Ram et al. 2023, Shi et al. 2023)

Wrapping up



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

Retrieval-based LMs: Training

Challenges of updating retrieval models

Datastore

$\mathbf{x}_1$

$\mathbf{x}_2$

$\mathbf{x}_3$

$\ldots$

Encoder → Index

We may encode a lot of (>100M) text chunks using the encoder!

Challenges of updating retrieval models

Datastore

$\mathbf{x}_1$

$\mathbf{x}_2$

$\mathbf{x}_3$

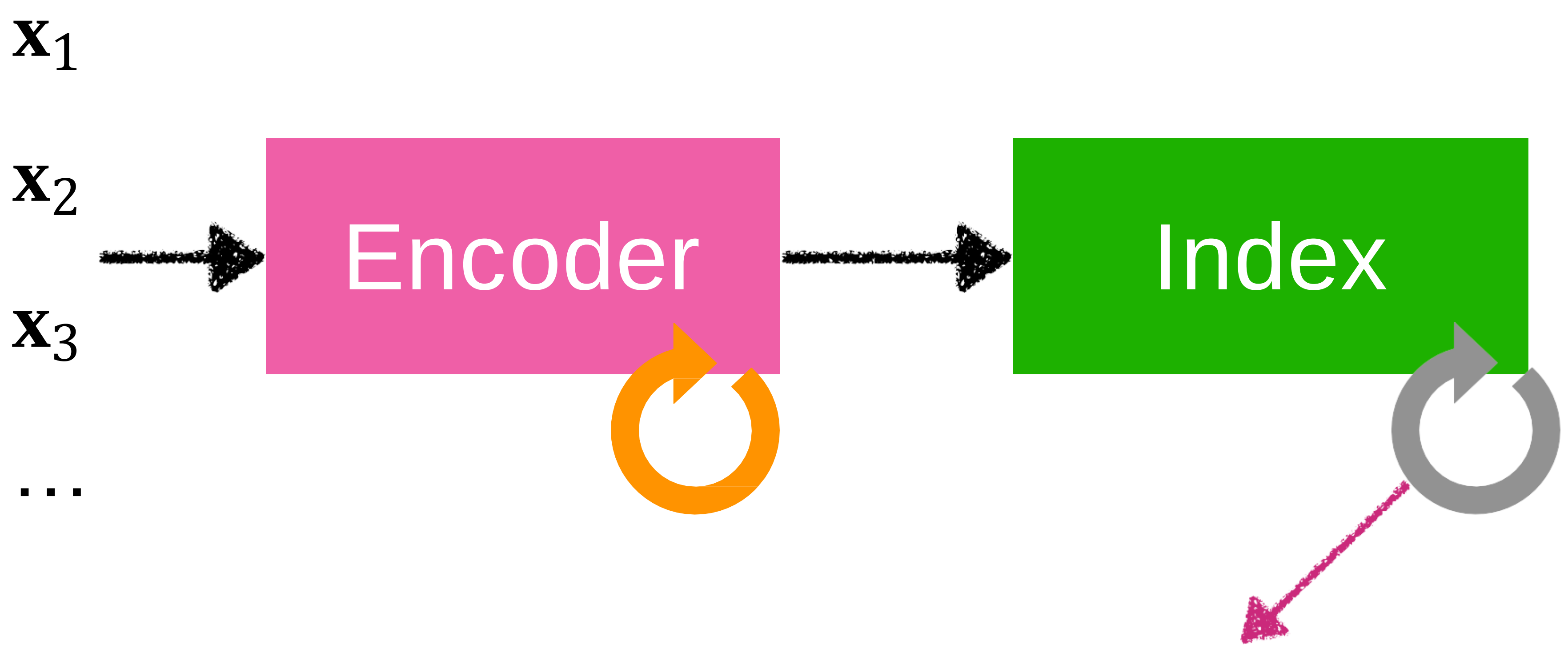
$\ldots$

Encoder → Index

During training, we will update the encoder

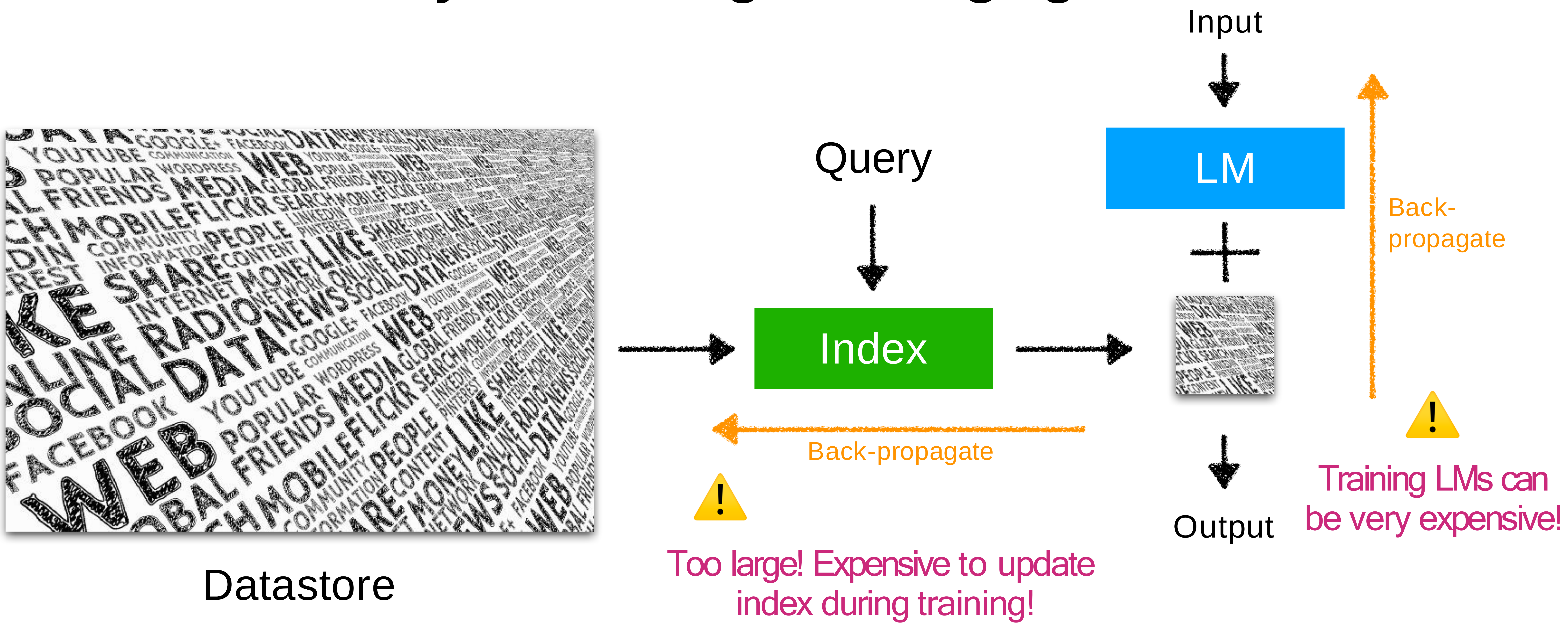
Challenges of updating retrieval models

Datastore



Re-indexing will be very expensive!

Why is training challenging?



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

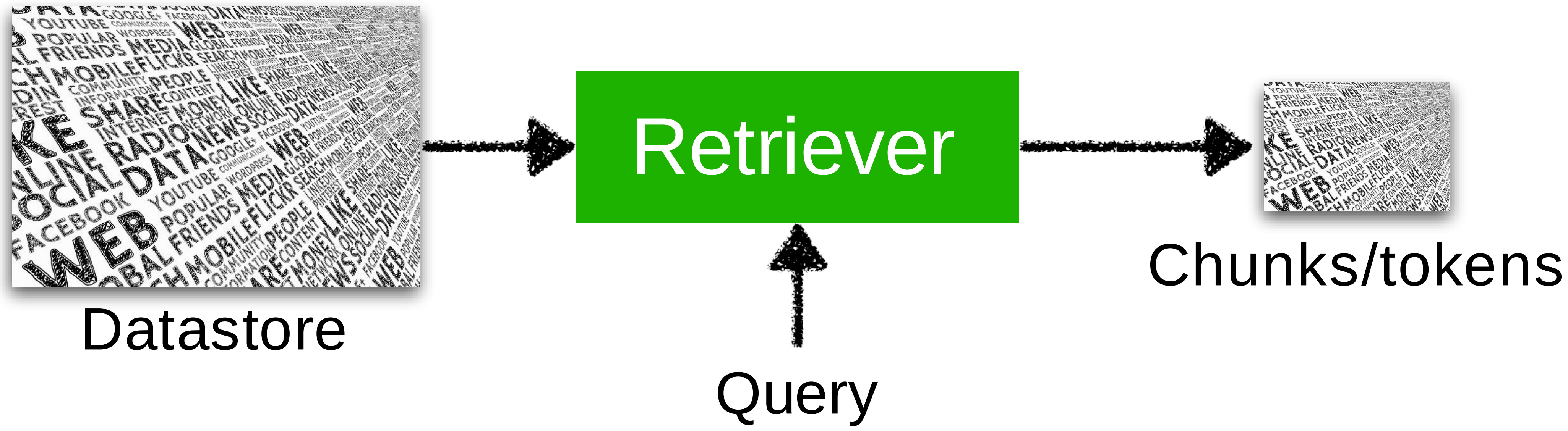
- **Independent training**
- Sequential training
- Joint training w/ asynchronous index update
- Joint training w/ in-batch approximation

Independent training

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

- Training language models

- Training retrieval models

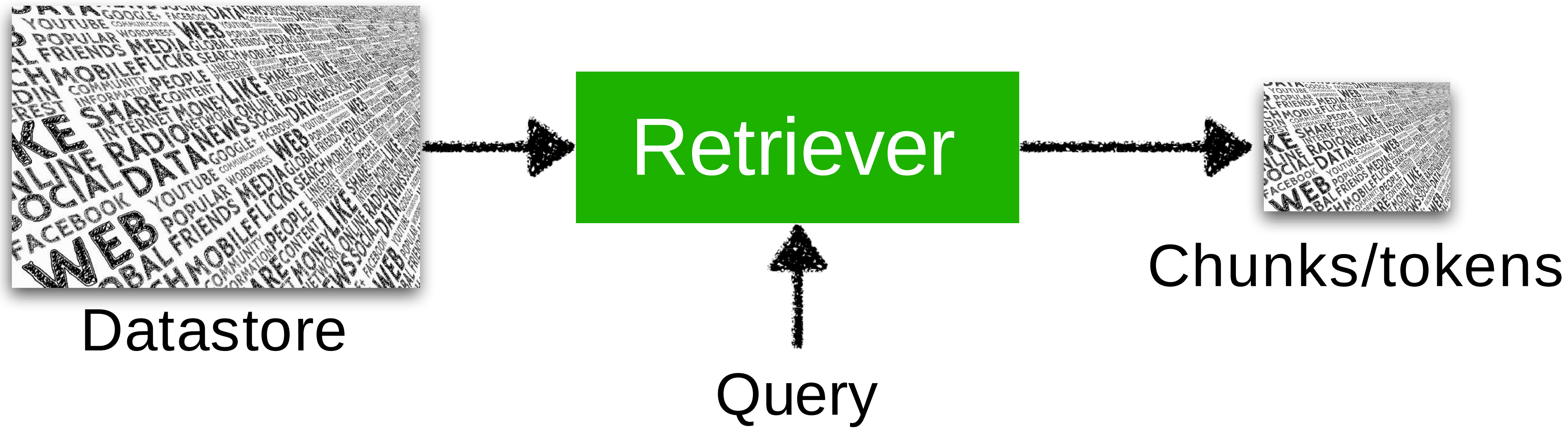


Independent training

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

- Training language models

- Training retrieval models



Training language models

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

Training language models

Input → LM → Output

← Back-propagate

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

GPT

PaLM

LLaMA

GPT-J

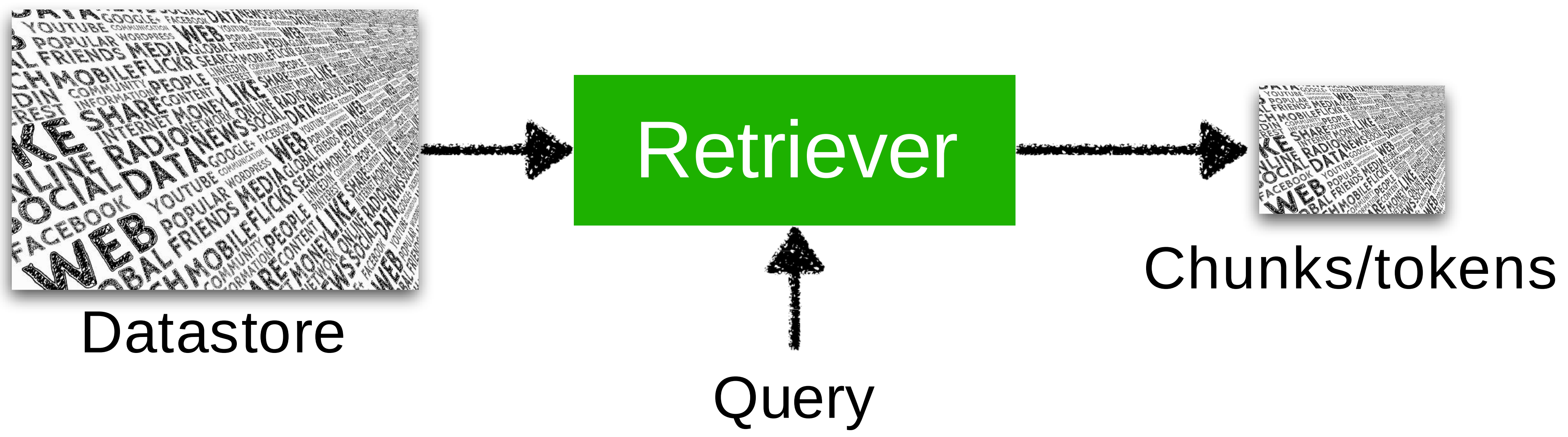
……

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, …]

Lexical overlap

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

[0, 1.2, 0.4, 0, 0.8, 0, …]

Text chunks

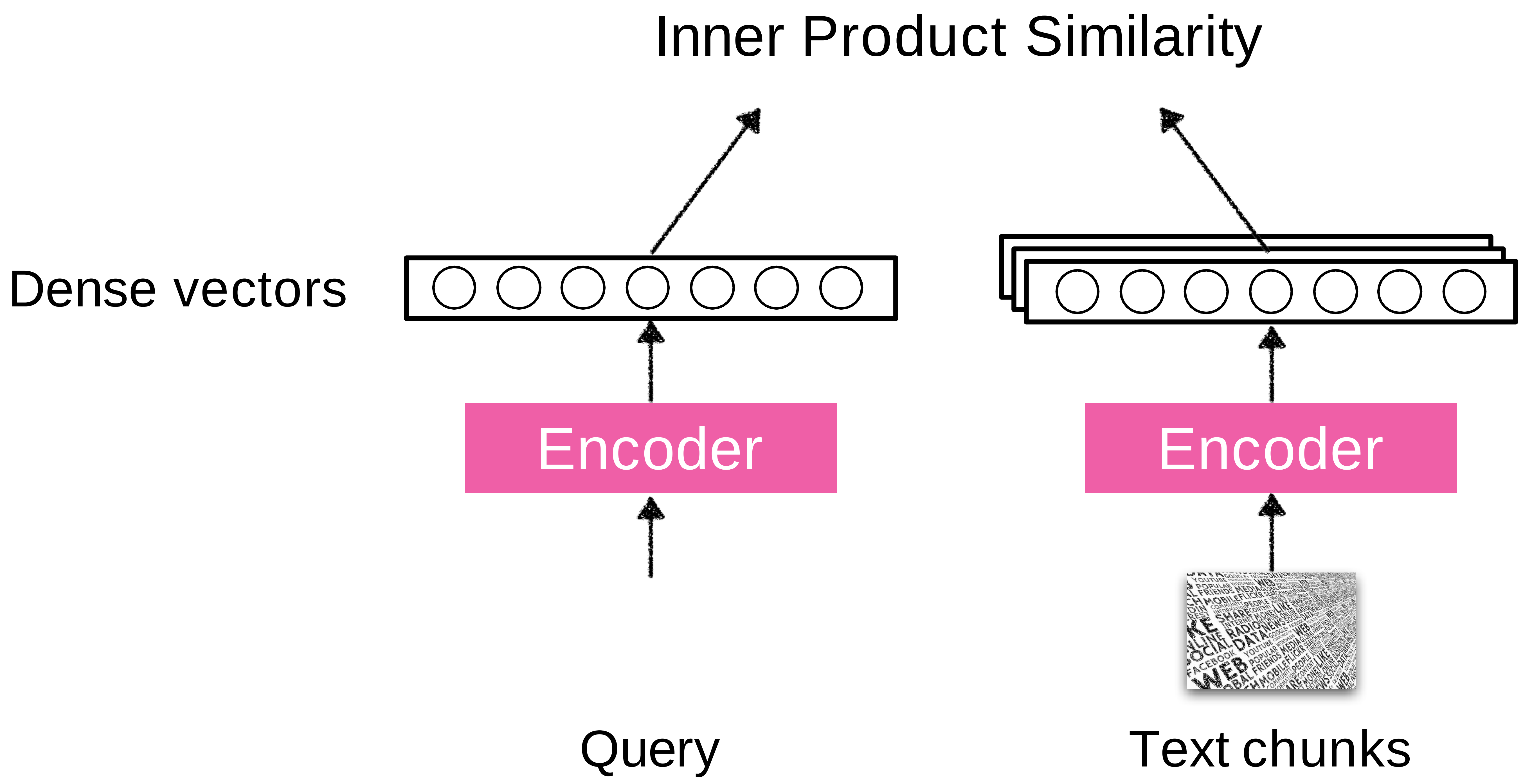
Sparse vectors

No training needed!

Ramos, 2003. "Using TF-IDF to Determine Word Relevance in Document Queries"

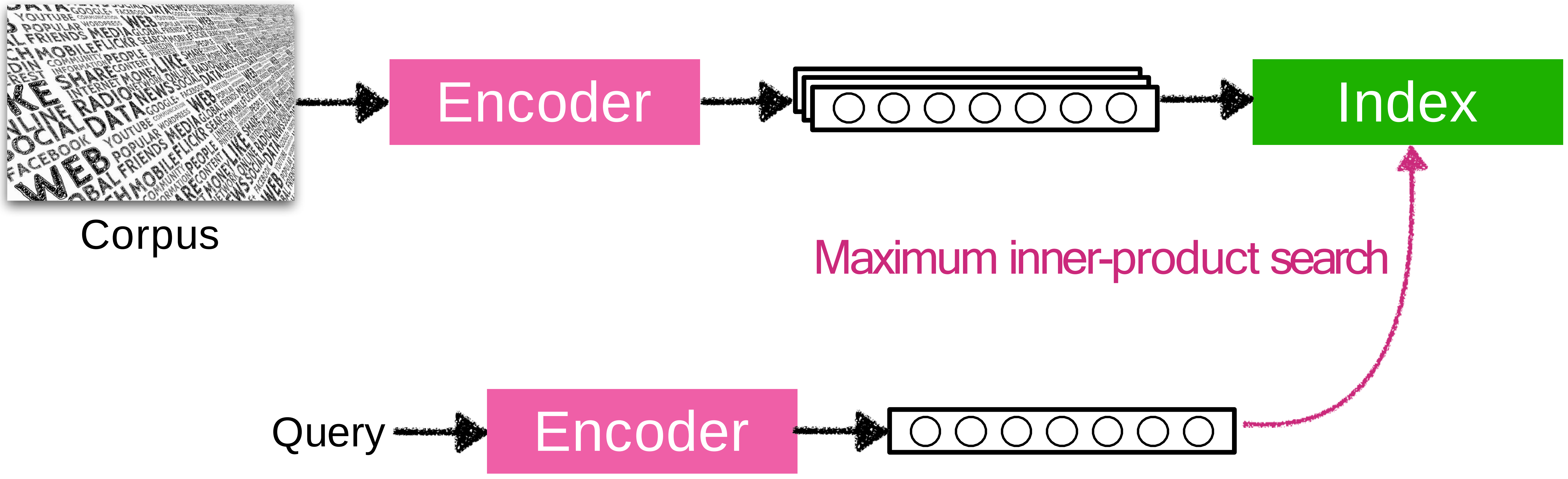
Robertson and Zaragoza, 2009. "The Probabilistic Relevance Framework: BM25 and Beyond"

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

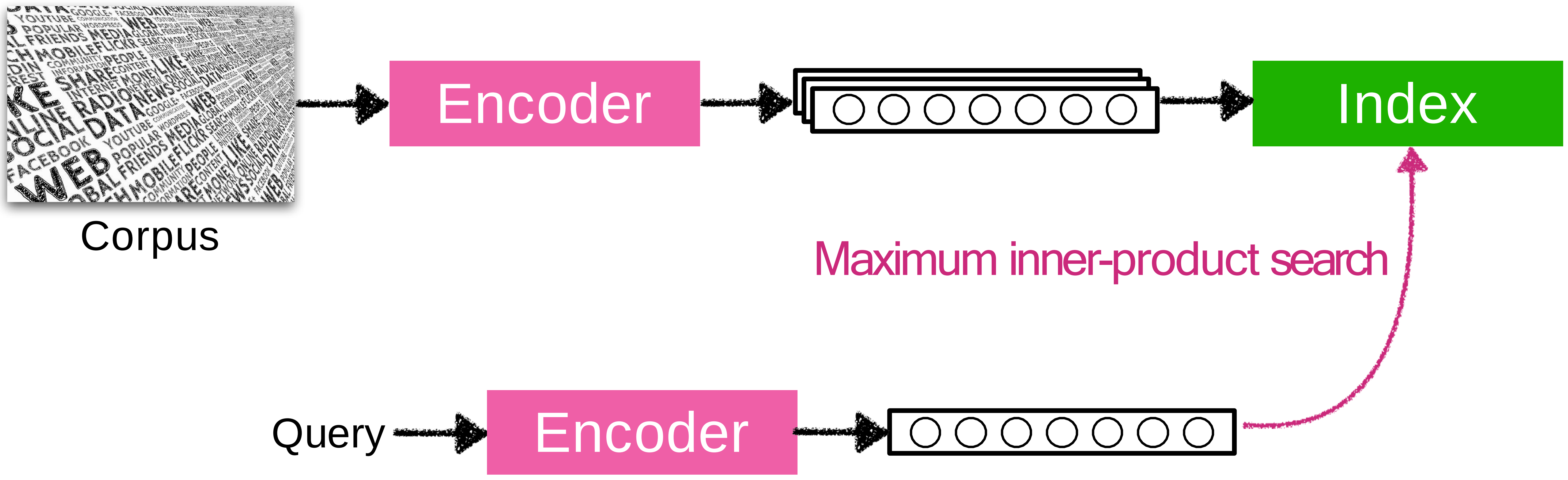


Karpukhin et al., 2020. “Dense Passage Retrieval for Open-Domain Question Answering”

Dense retrievers: Inference

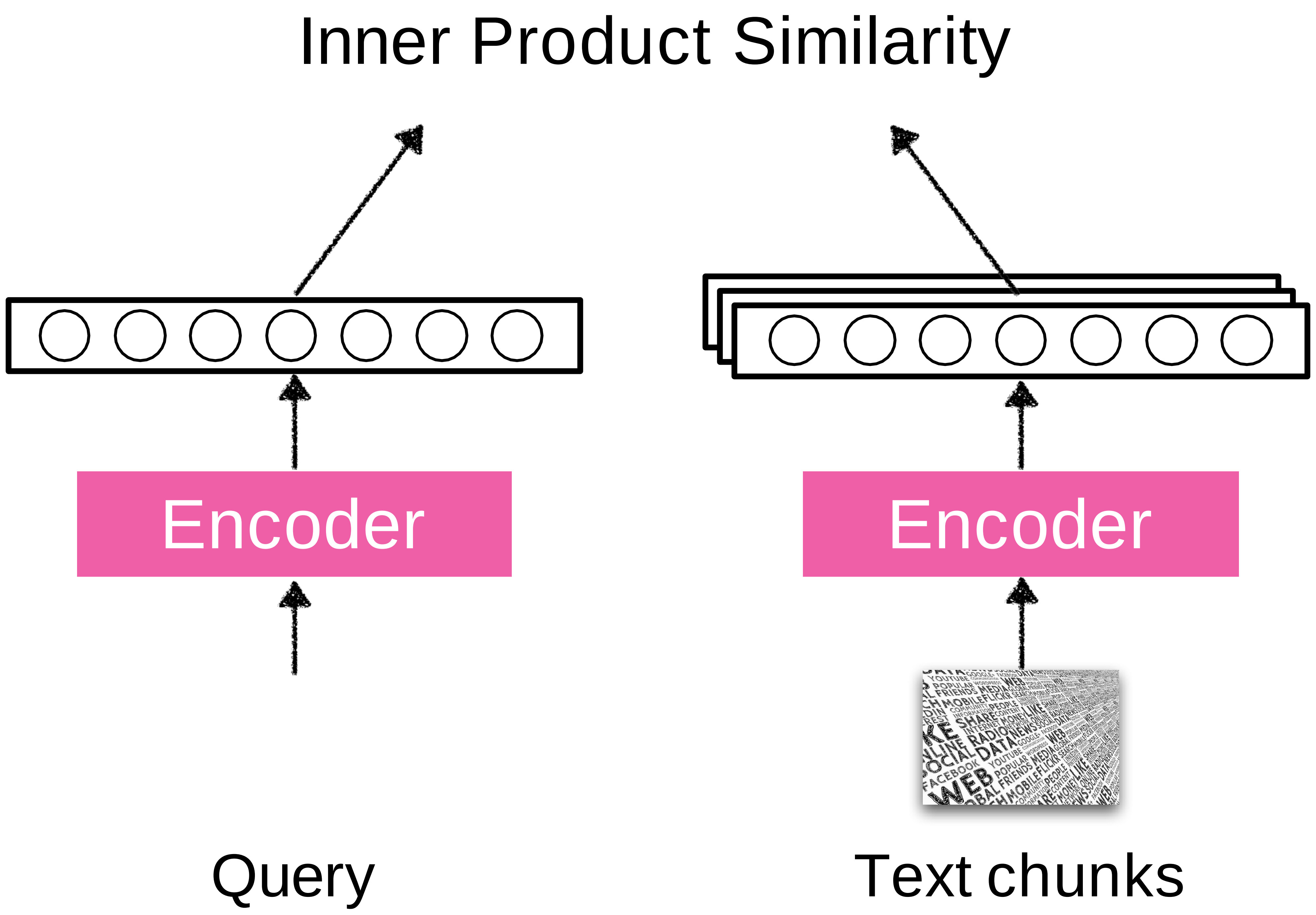


Dense retrievers: Inference

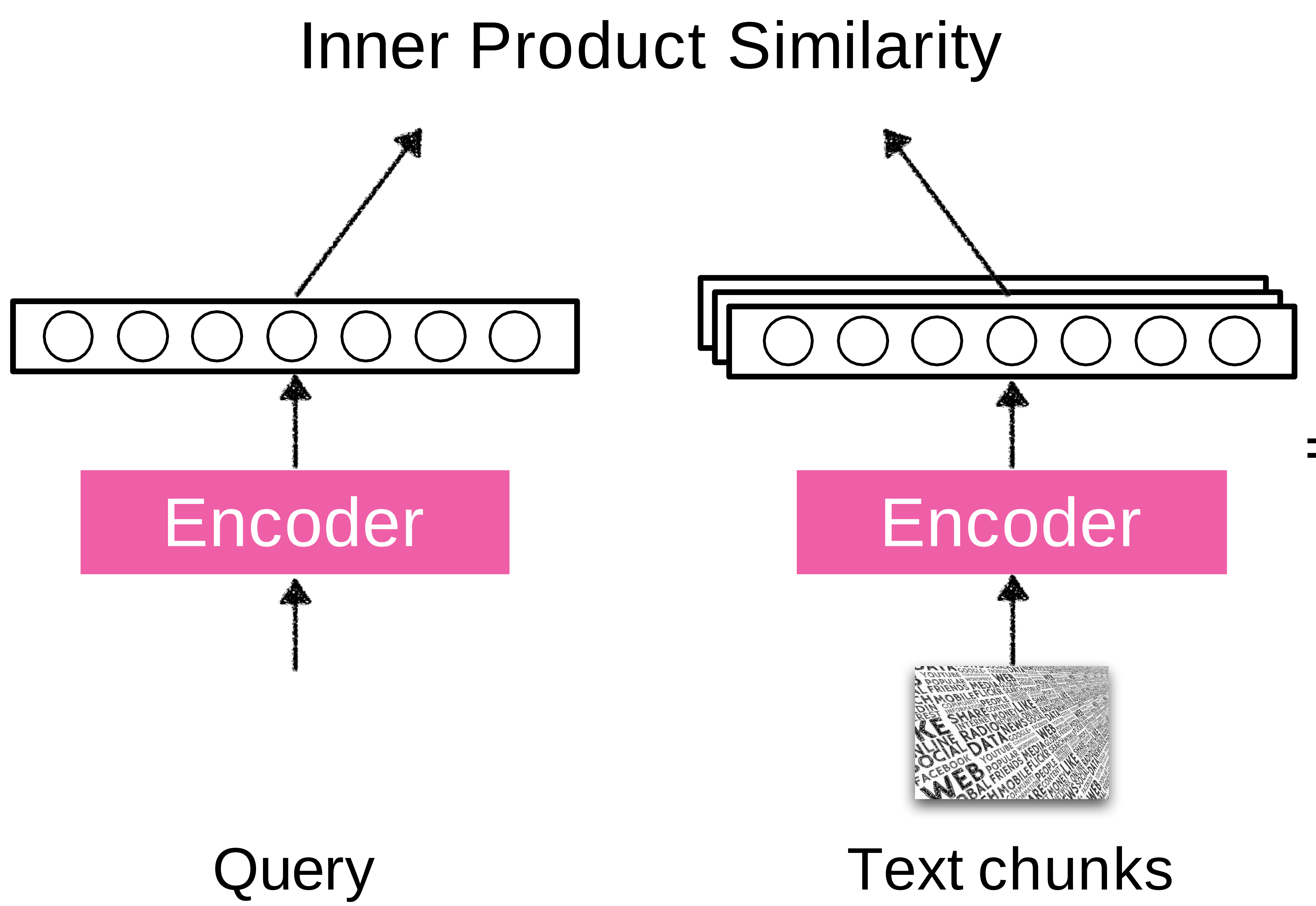


How to train dense retrieval models?

Training dense retrieval models: DPR



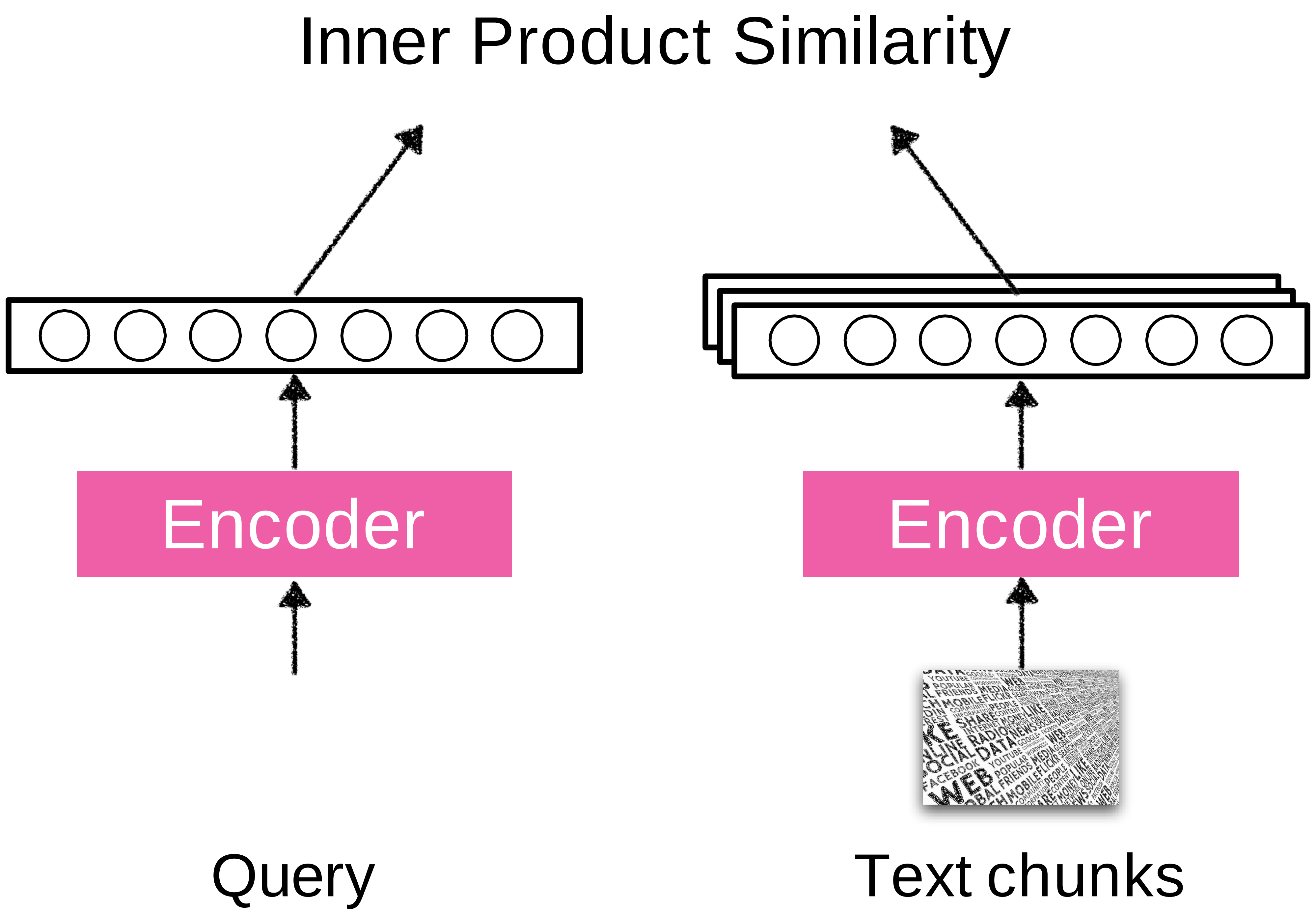
Training dense retrieval models: DPR



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

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

Training dense retrieval models: DPR

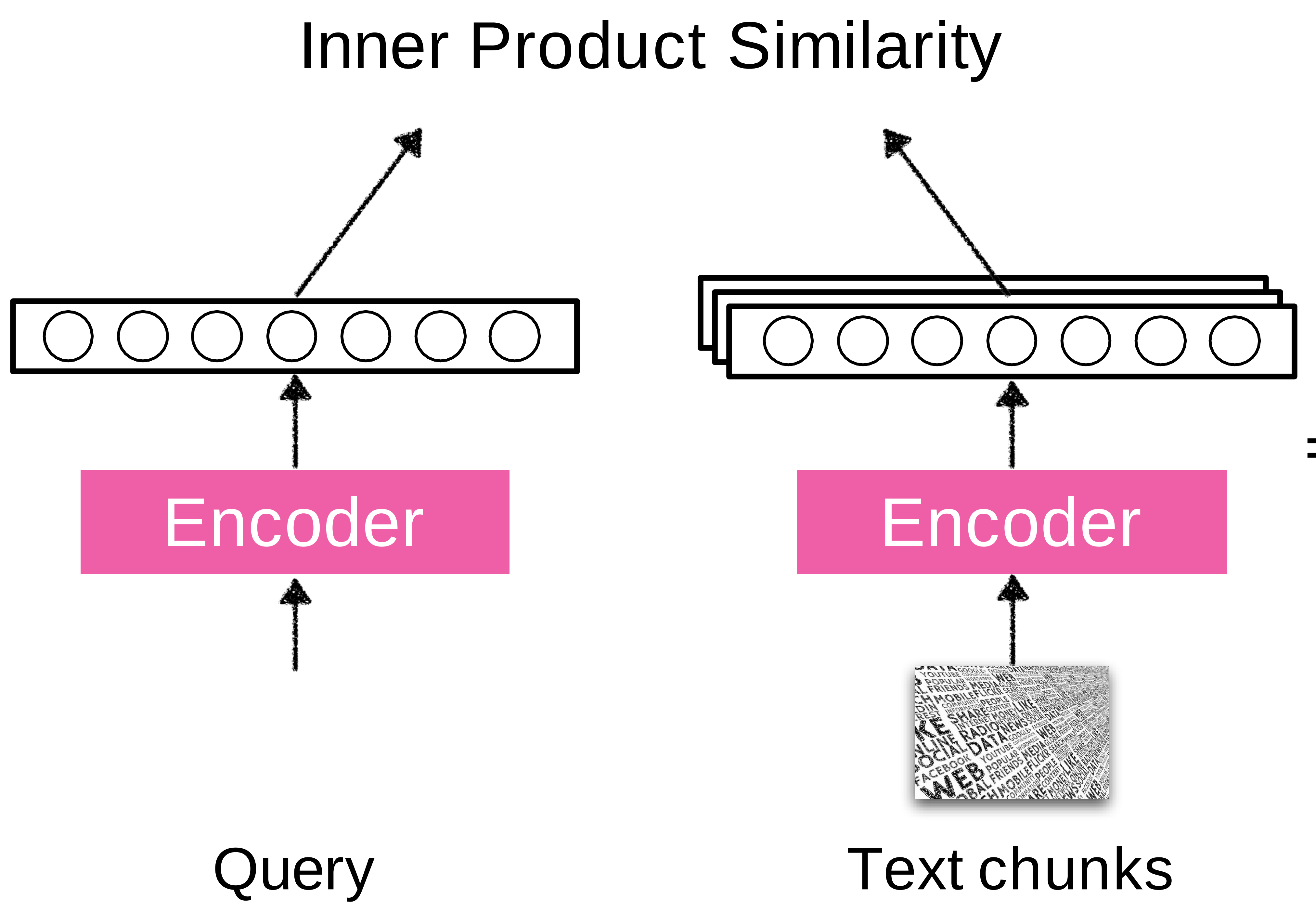


$$L(q, p^+, p_1^-, p_2^-, \ldots, p_n^-)$$

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

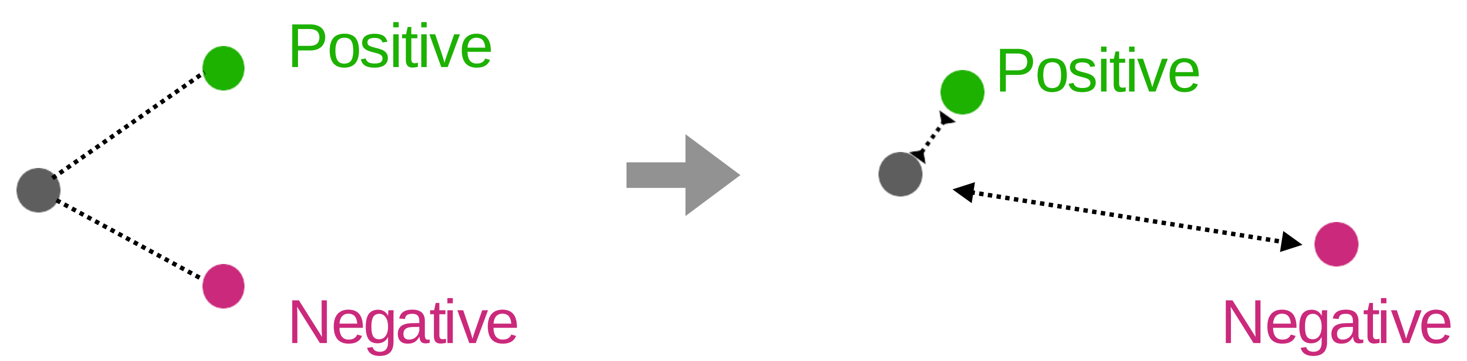
Contrastive learning

Training dense retrieval models: DPR

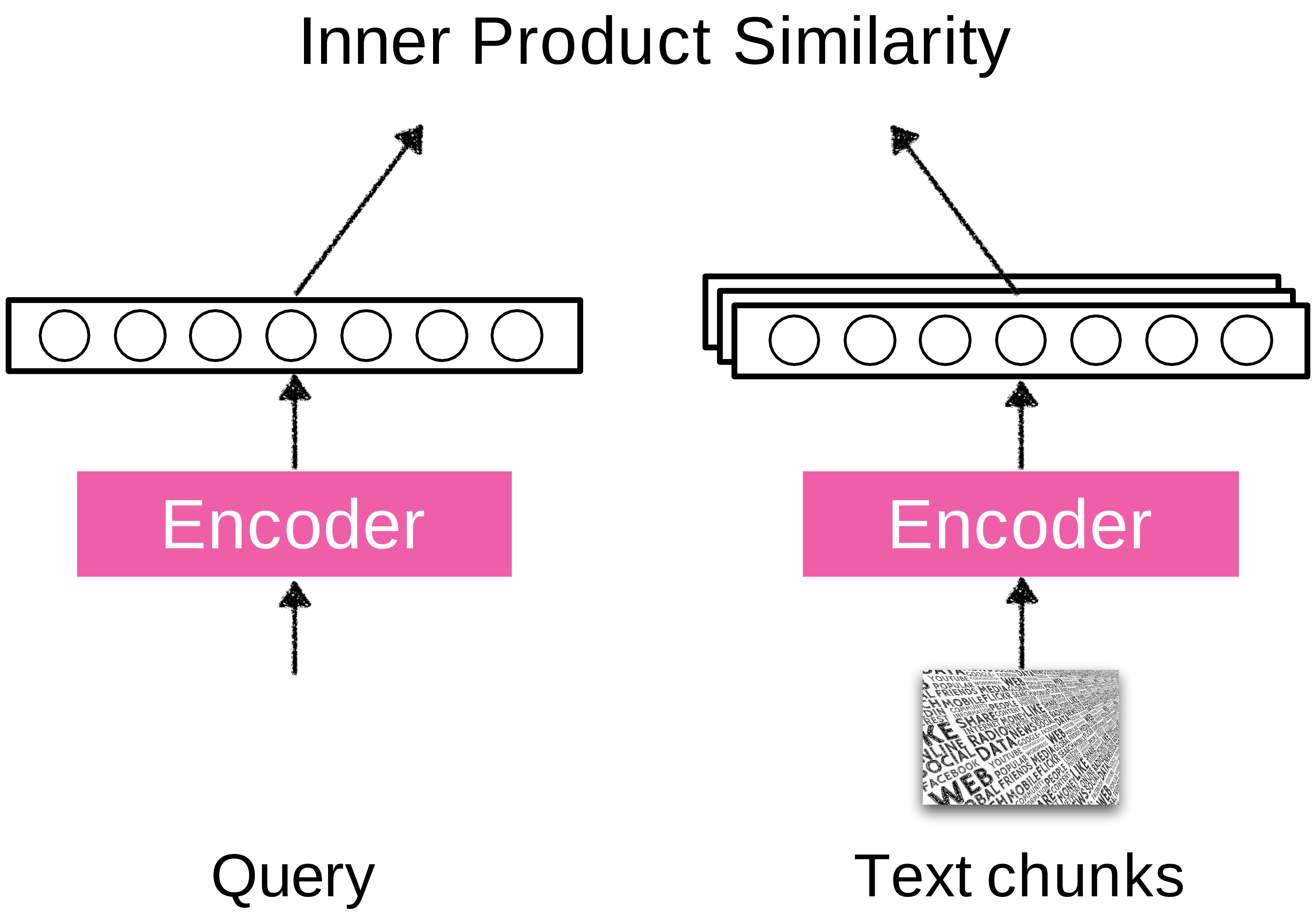


$$L(q, p^+, p_1^-, p_2^-, \ldots, p_n^-) = -\log \frac{\exp(\mathrm{sim}(q, p^+))}{\exp(\mathrm{sim}(q, p^+)) + \sum_{j=1}^{n} \exp(\mathrm{sim}(q, p_j^-))}$$

Contrastive learning



Training dense retrieval models: DPR



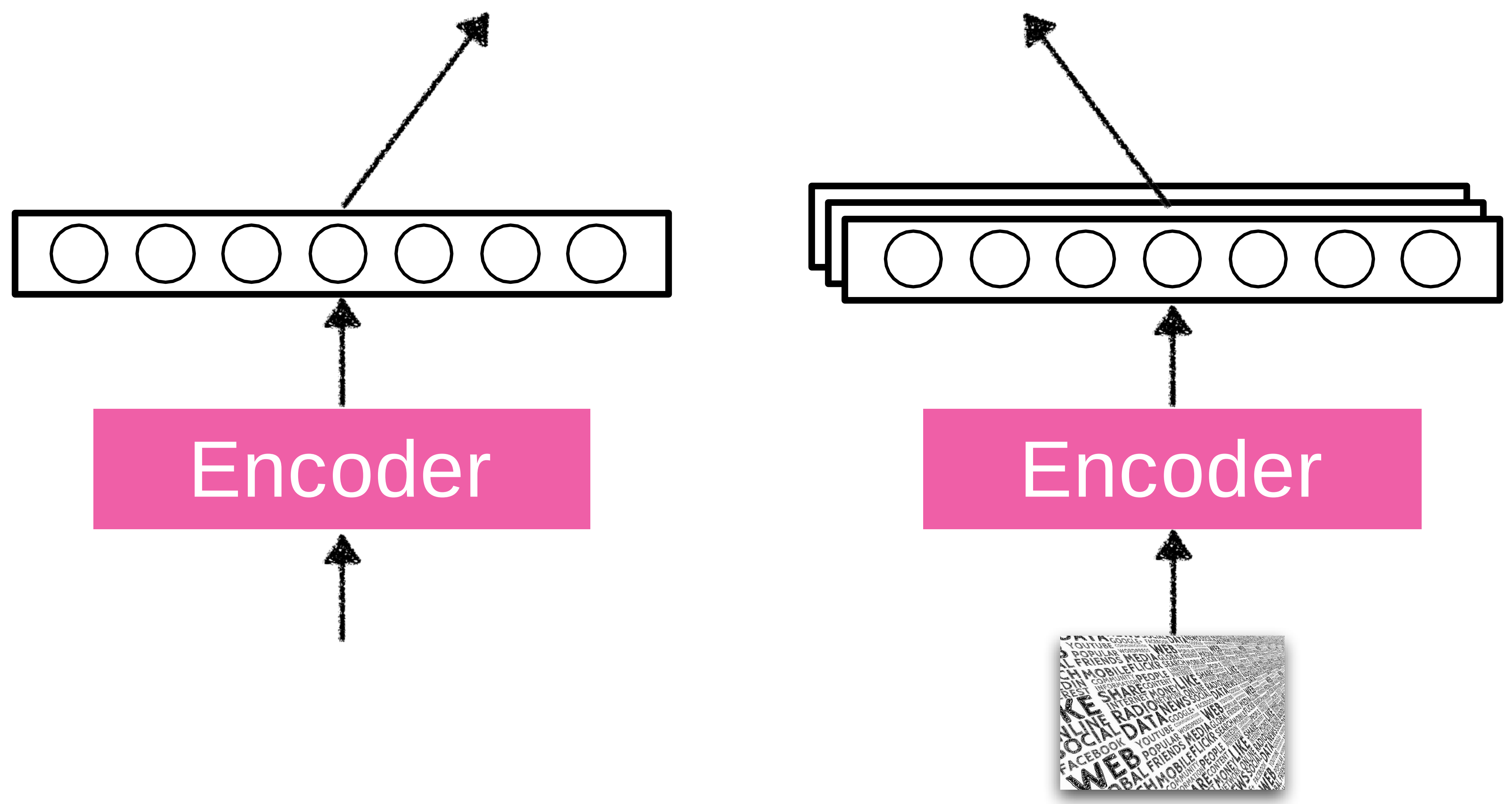
$$L(q, p^+, p_1^-, p_2^-, \ldots, p_n^-)$$

Positive passage

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

Training dense retrieval models: DPR

Inner Product Similarity



Query

Text chunks

Negative passages

Too expensive to consider all negatives!

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

Positive passage

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

Training with “in-batch” negatives

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

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

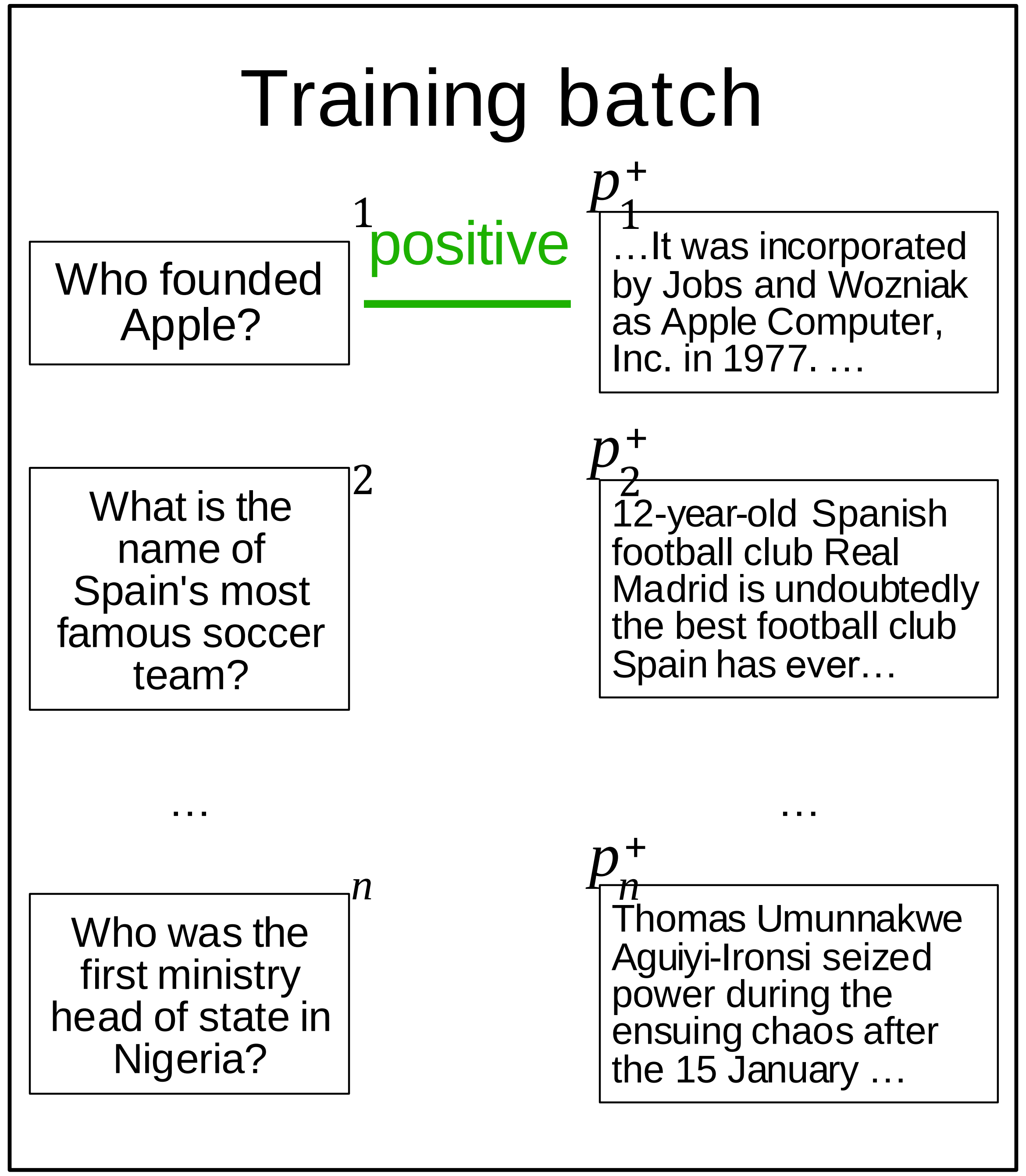
Training batch

1	p_1^+
Who founded Apple?	…It was incorporated by Jobs and Wozniak as Apple Computer, Inc. in 1977. …
2	p_2^+
What is the name of Spain's most famous soccer team?	12-year-old Spanish football club Real Madrid is undoubtedly the best football club Spain has ever…
…	…
n	p_n^+
Who was the first ministry head of state in Nigeria?	Thomas Umunnakwe Aguiyi-Ironsi seized power during the ensuing chaos after the 15 January …

Training with “in-batch” negatives

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

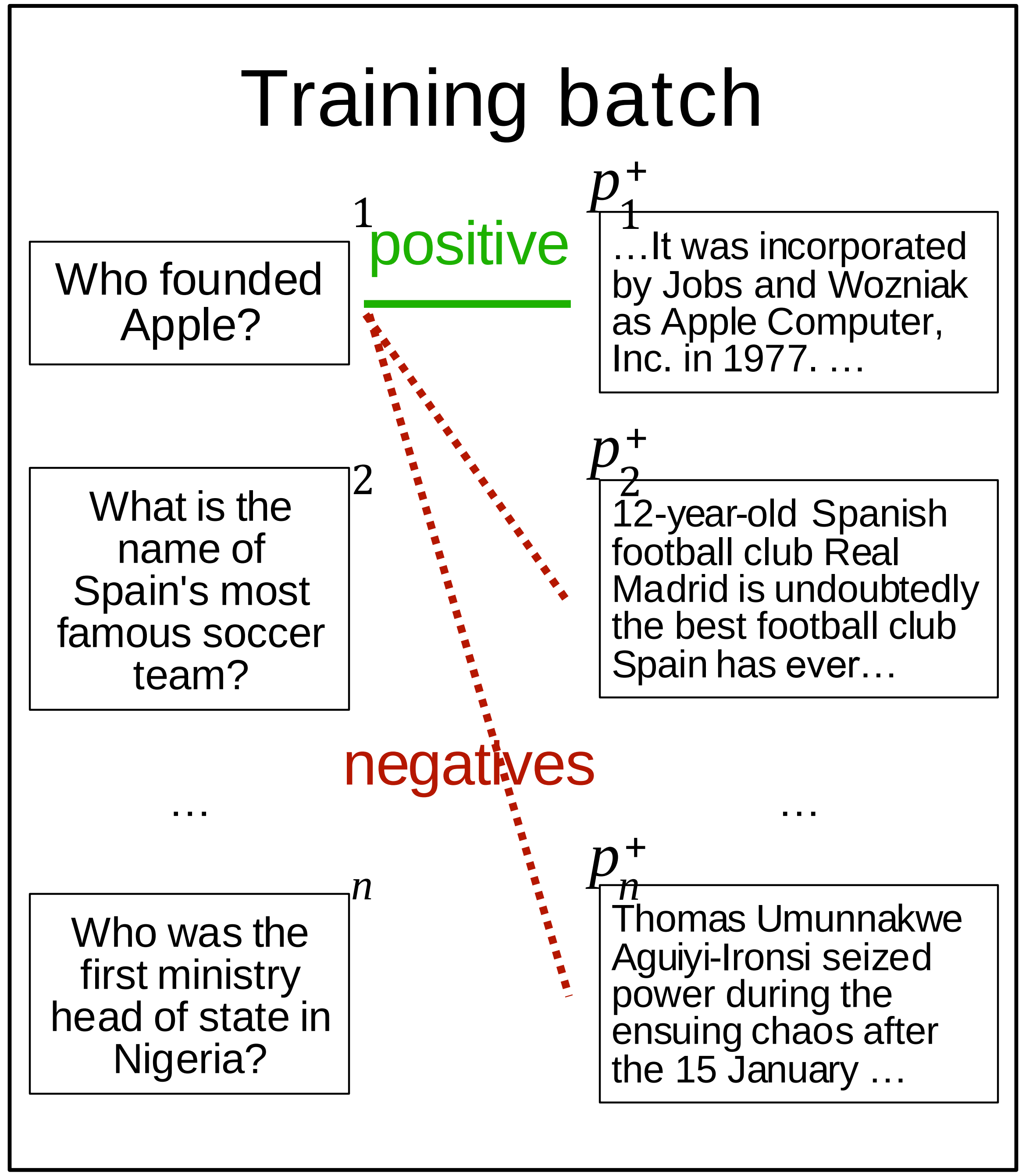
$$= -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^{n} \exp(\text{sim}(q, p_j^-))}$$



Training with “in-batch” negatives

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

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



Back-propagation to all in-batch negatives!

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

↓

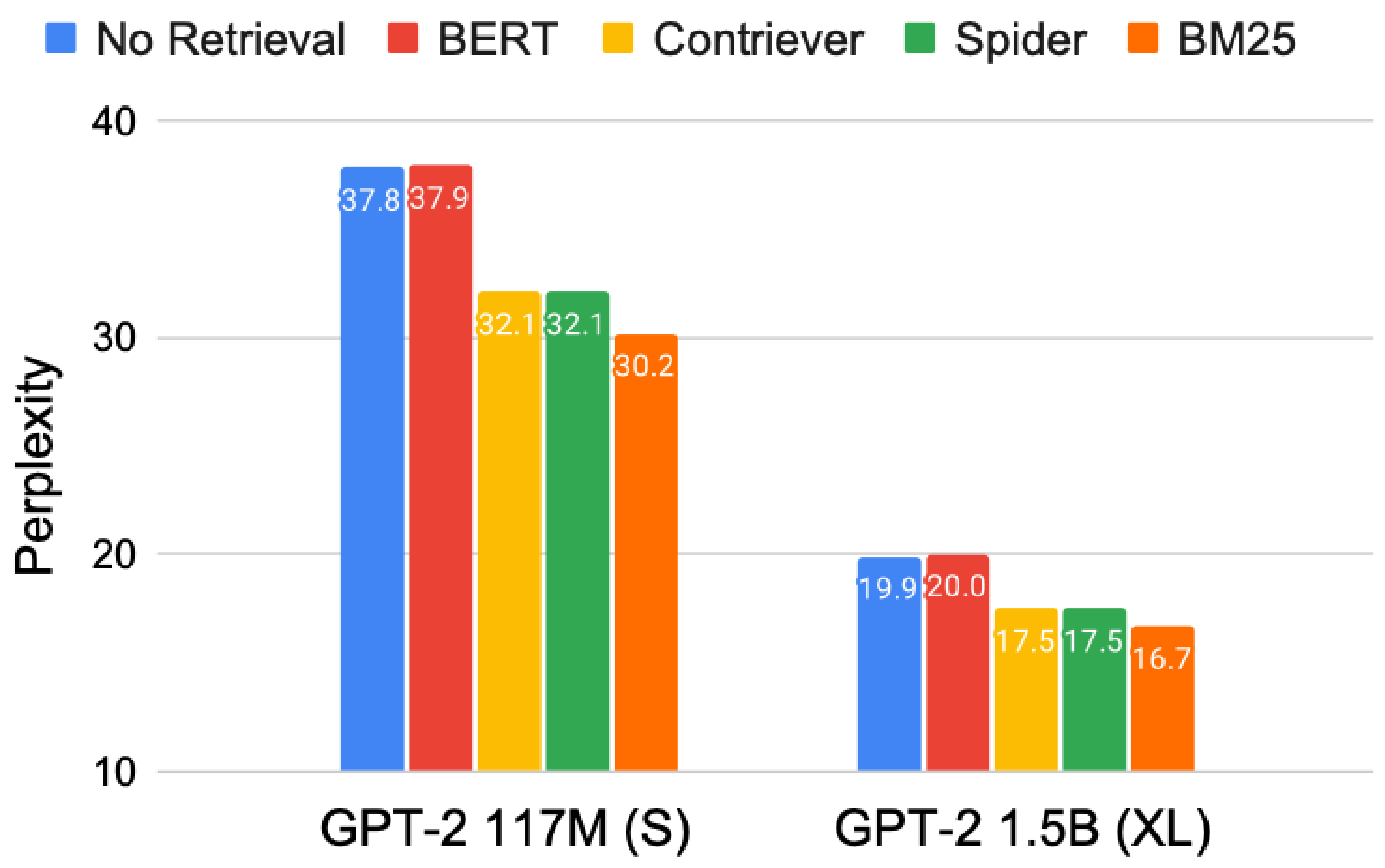
LM | GPT, OPT, LLaMA, …

↓

48 in the 2026 tournament.

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

Retrieval-in-context in LM



Better retrieval model

Better base LMs

→ Better **retrieval-based LMs**

Each component can be improved separately

Independent training

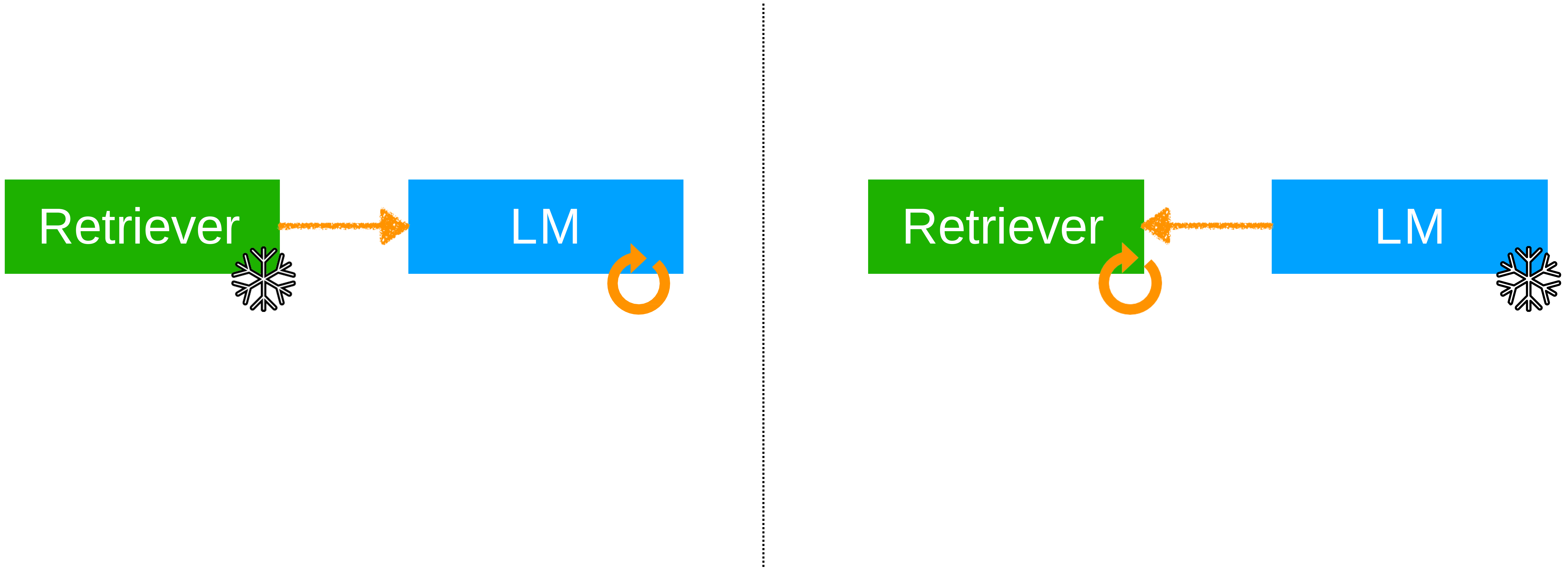
- Work with off-the-shelf models (no extra training required)
- Each part can be improved independently
- LMs are not trained to leverage retrieval
- Retrieval models are not optimized for LM tasks/domains

Training methods for retrieval-based LMs

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

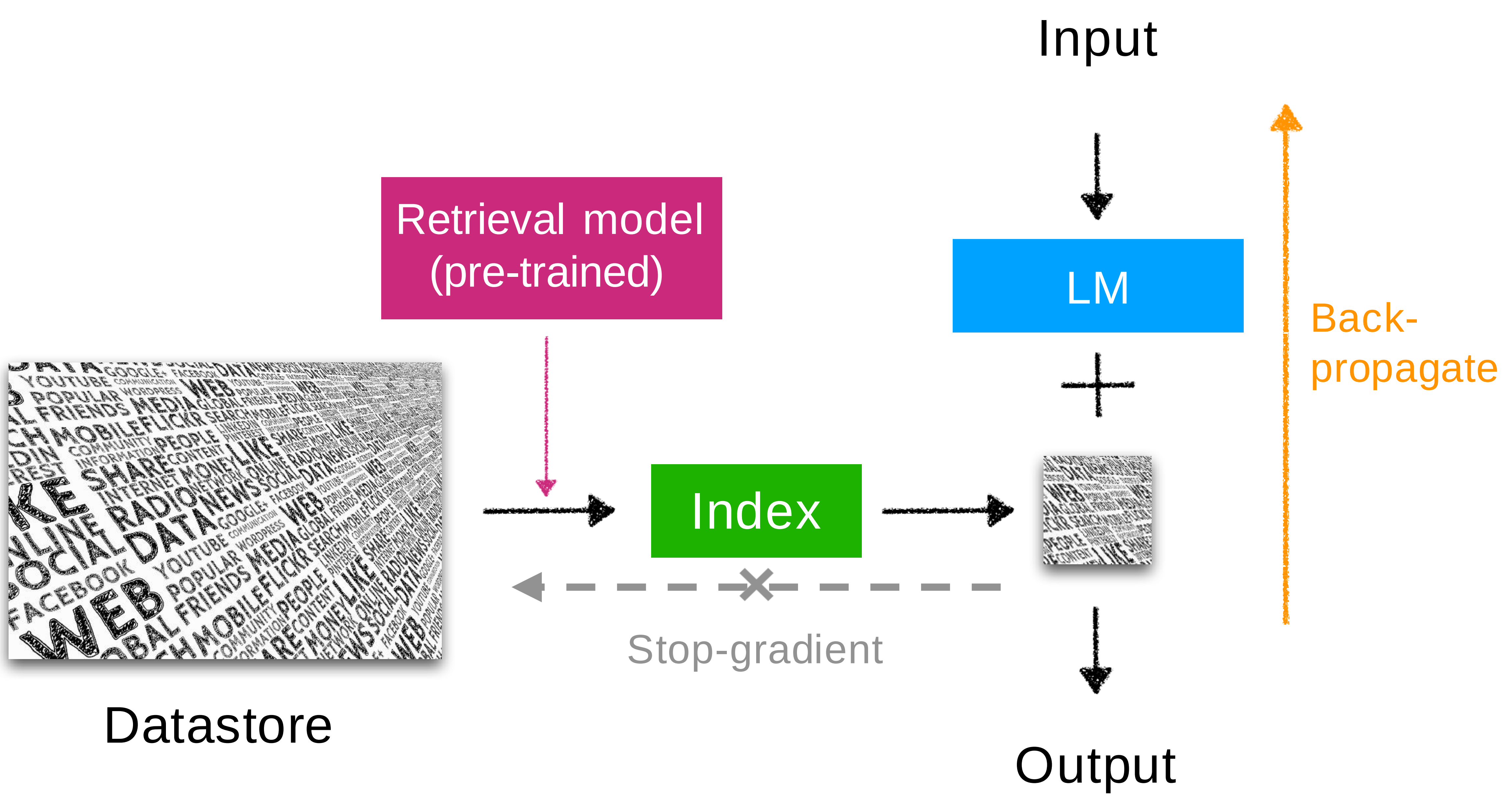
Sequential training

- 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

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



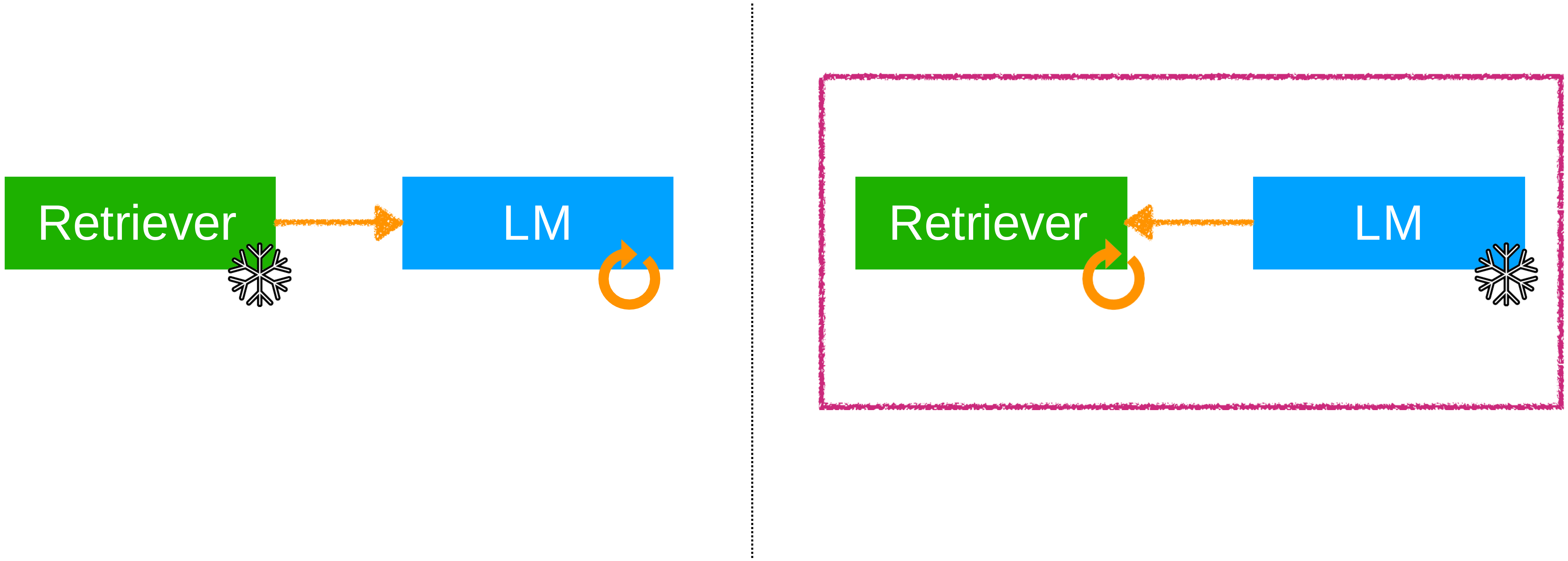
RETRO (Borgeaud et al. 2021)

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

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

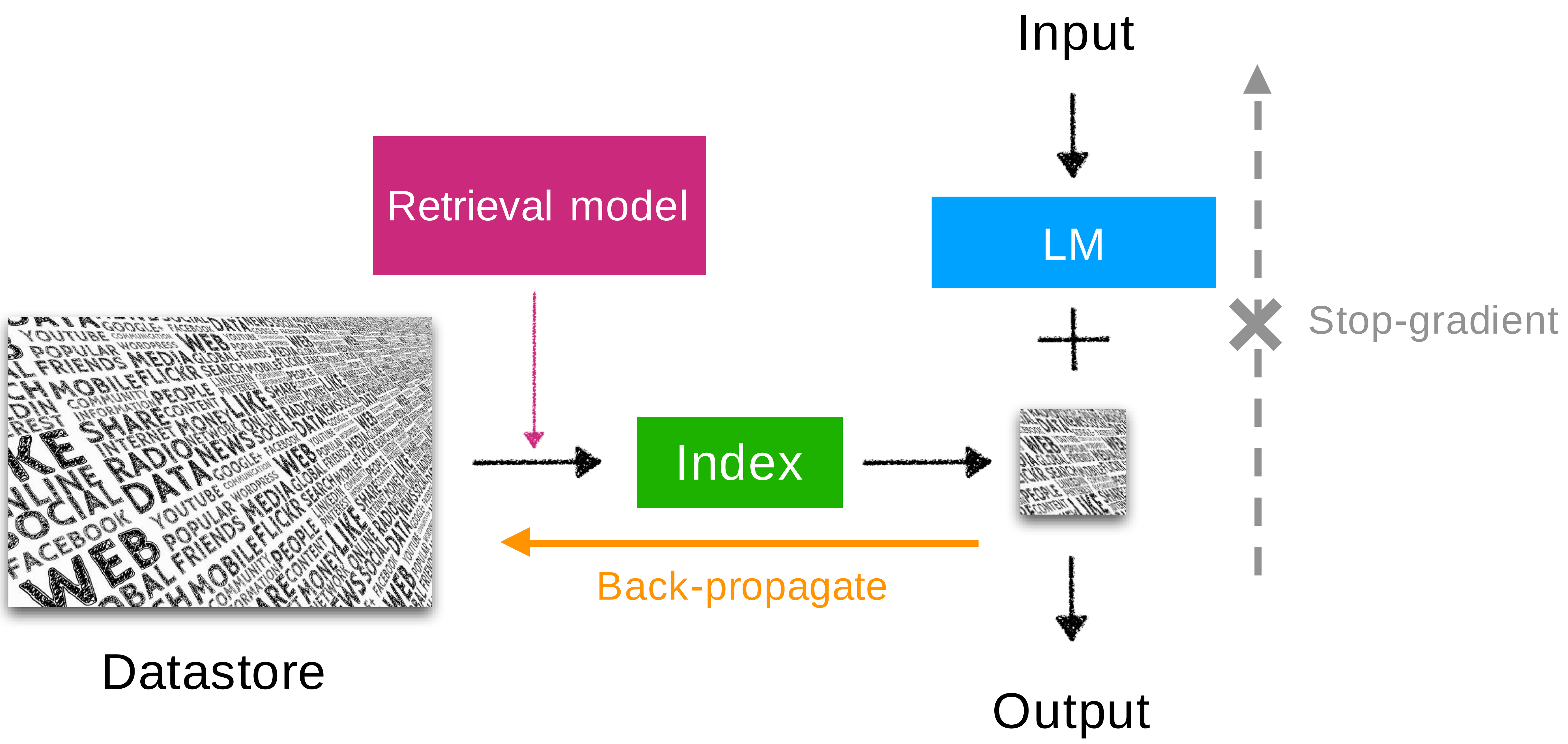
Sequential training

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

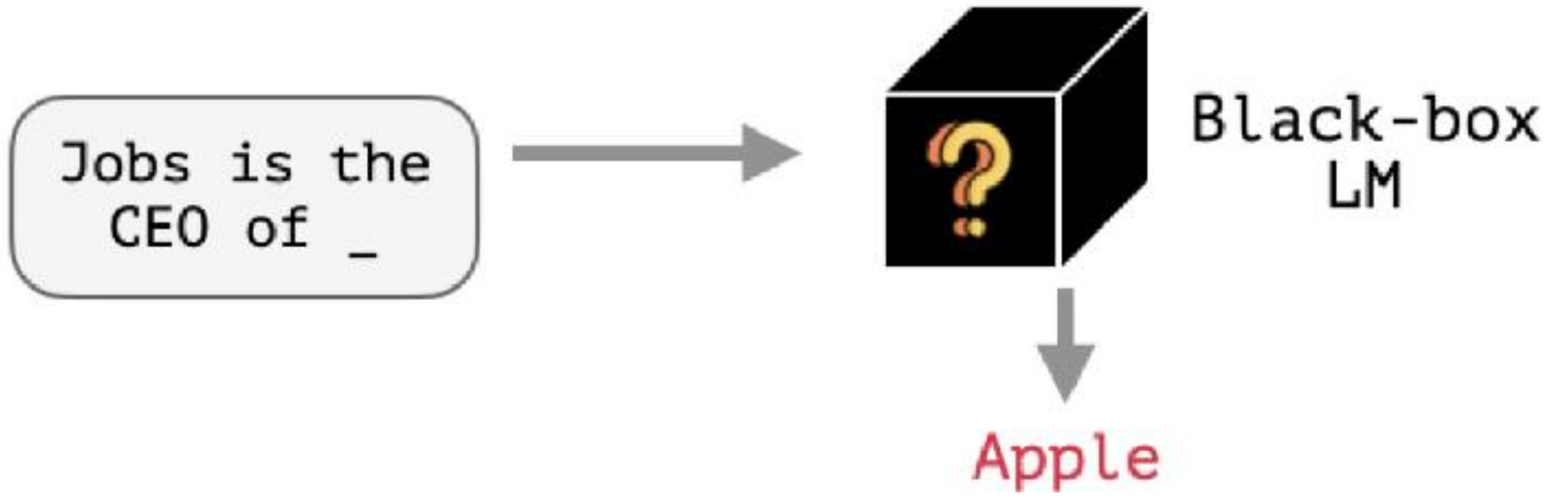


Sequential training

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



REPLUG (Shi et al. 2023)



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

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!

Training methods for retrieval-based LMs

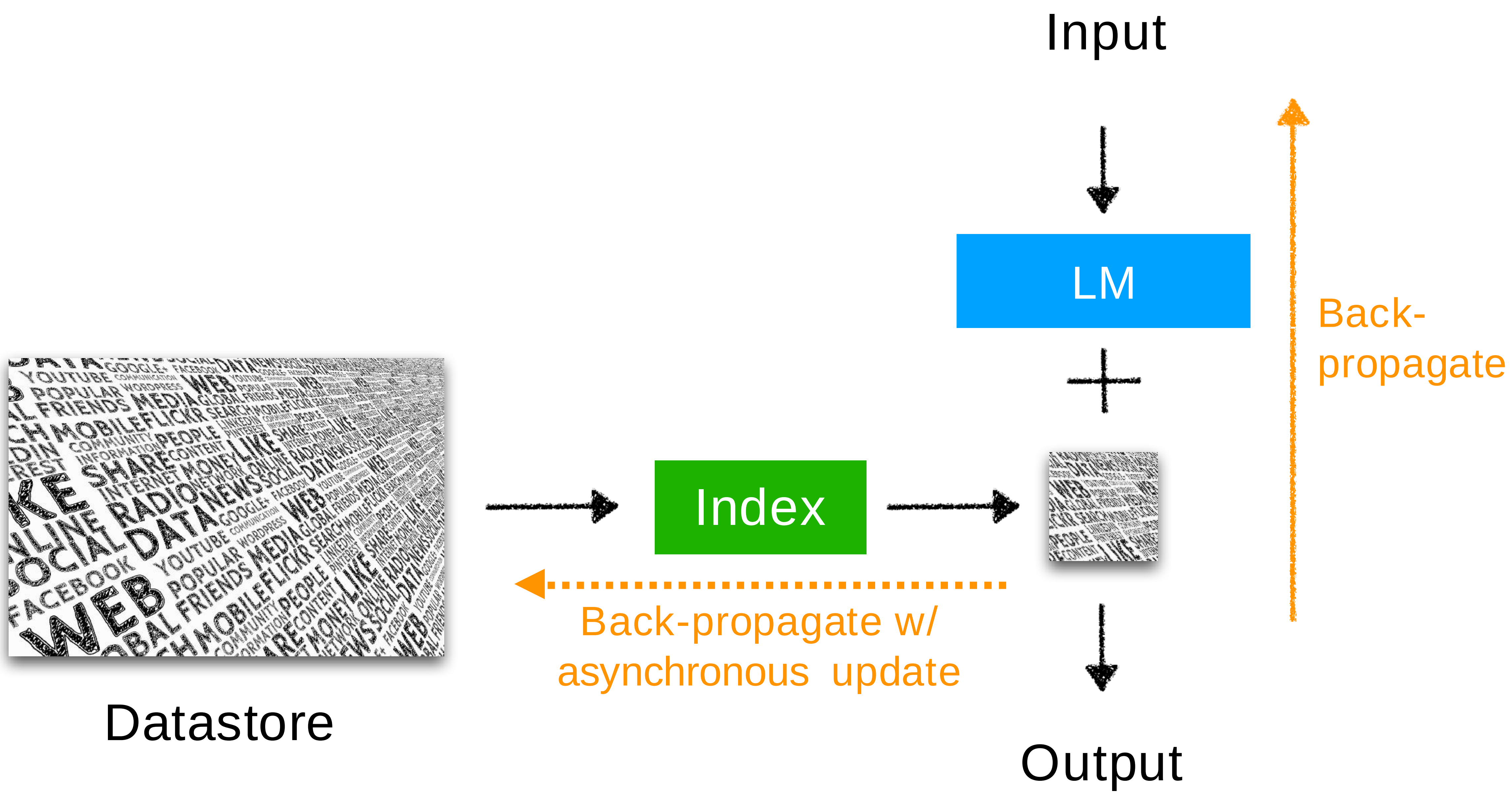
- Independent training
- Sequential training
- Joint training w/ asynchronous index update
- Joint training w/ in-batch approximation

Training methods for retrieval-based LMs

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

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

Datastore

$\mathbf{x}_1$

$\mathbf{x}_2$

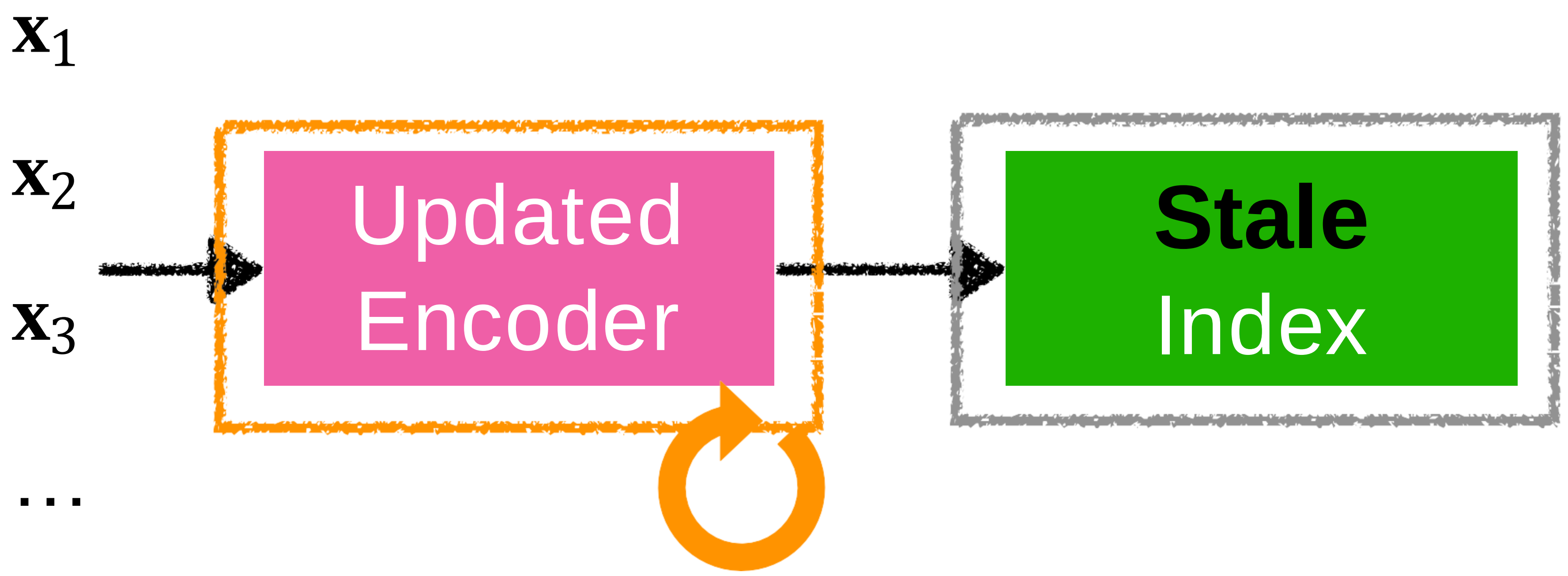
$\mathbf{x}_3$

$\dots$

$\rightarrow$ Encoder $\rightarrow$ Index

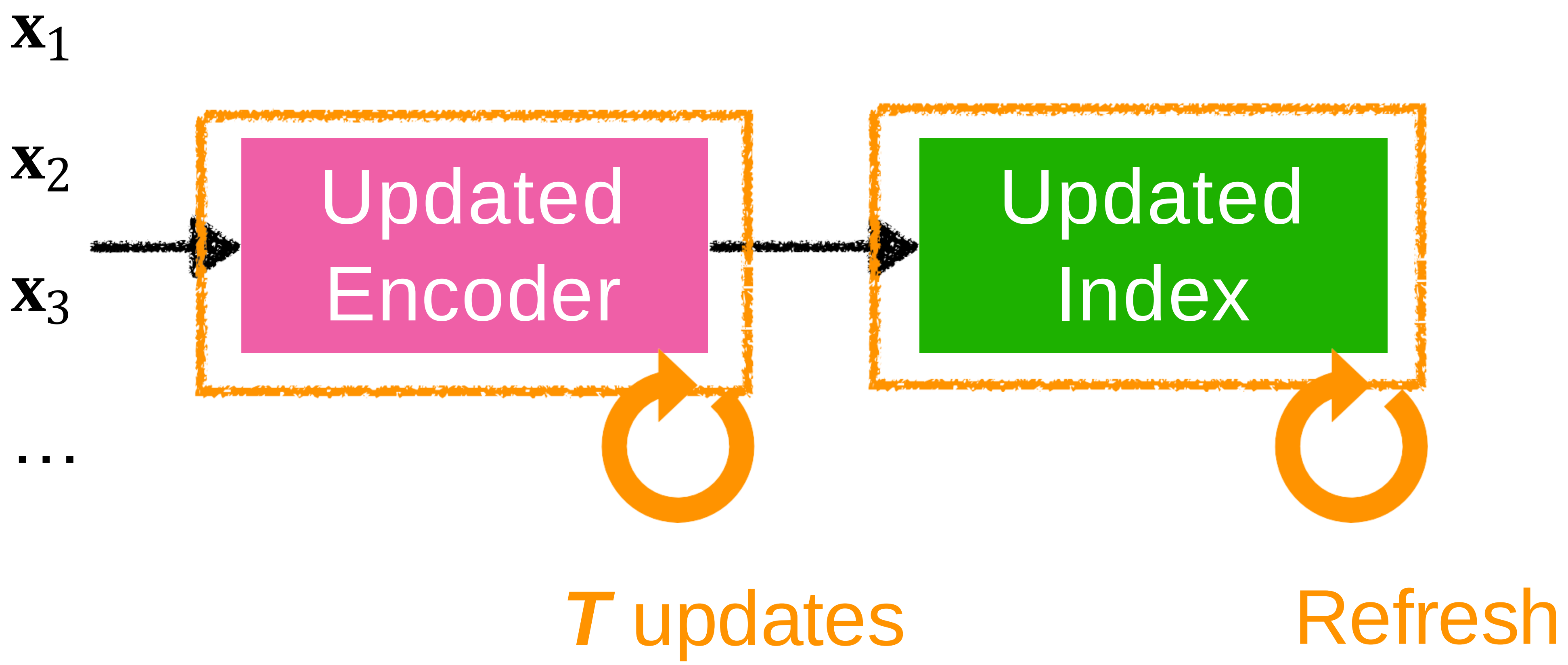
Asynchronous index update

Datastore



Asynchronous index update

Datastore



REALM (Guu et al. 2020)

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

***q* (=*x*)**

Index

The pyramidion on top allows for less material higher up the pyramid.

The pyramidion on top … the pyramid.

…

The [MASK] at the top of the pyramid.

LM

pyramid

$P(z \mid x)$

$P(y \mid x, z)$

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

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

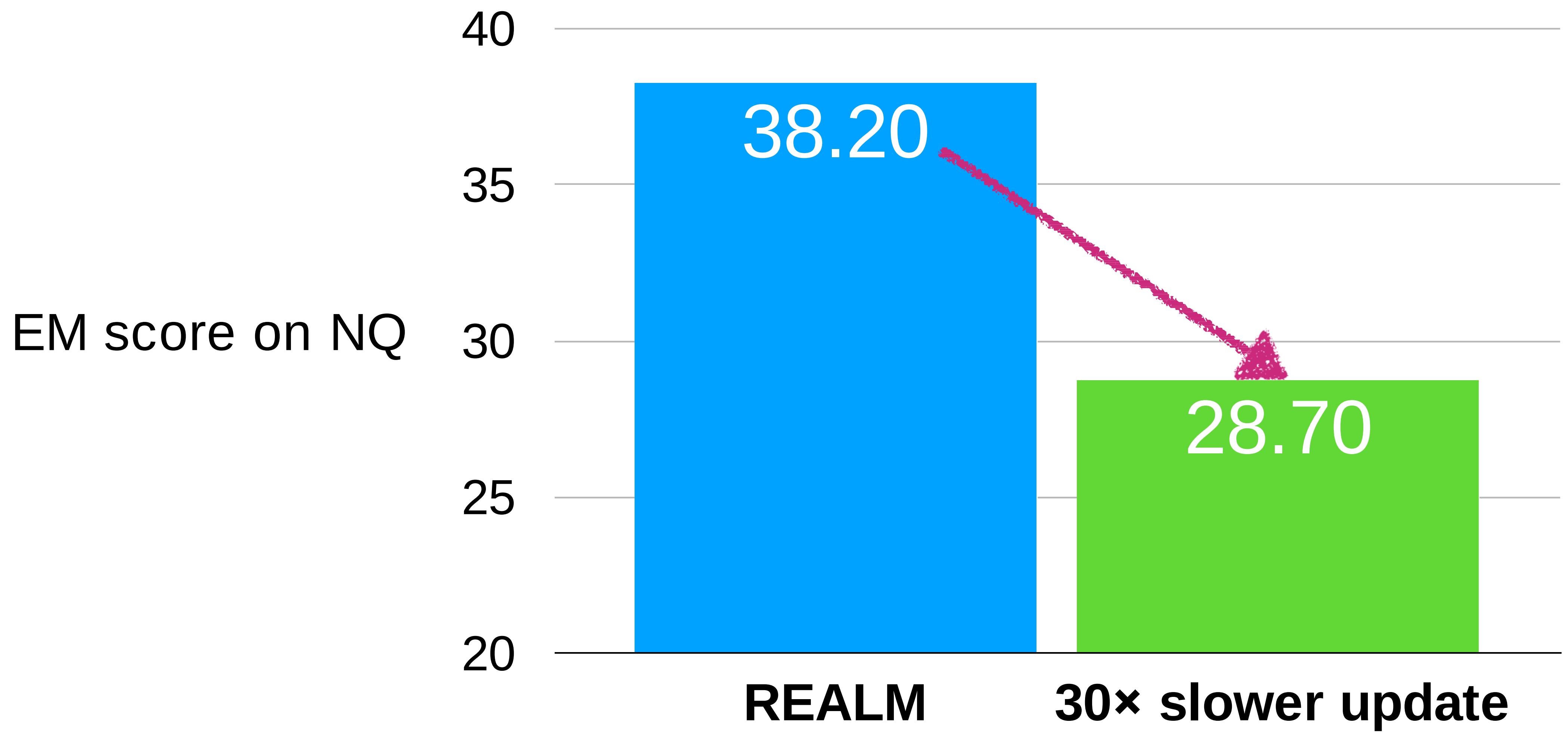
REALM: updating the index every 500 training steps

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

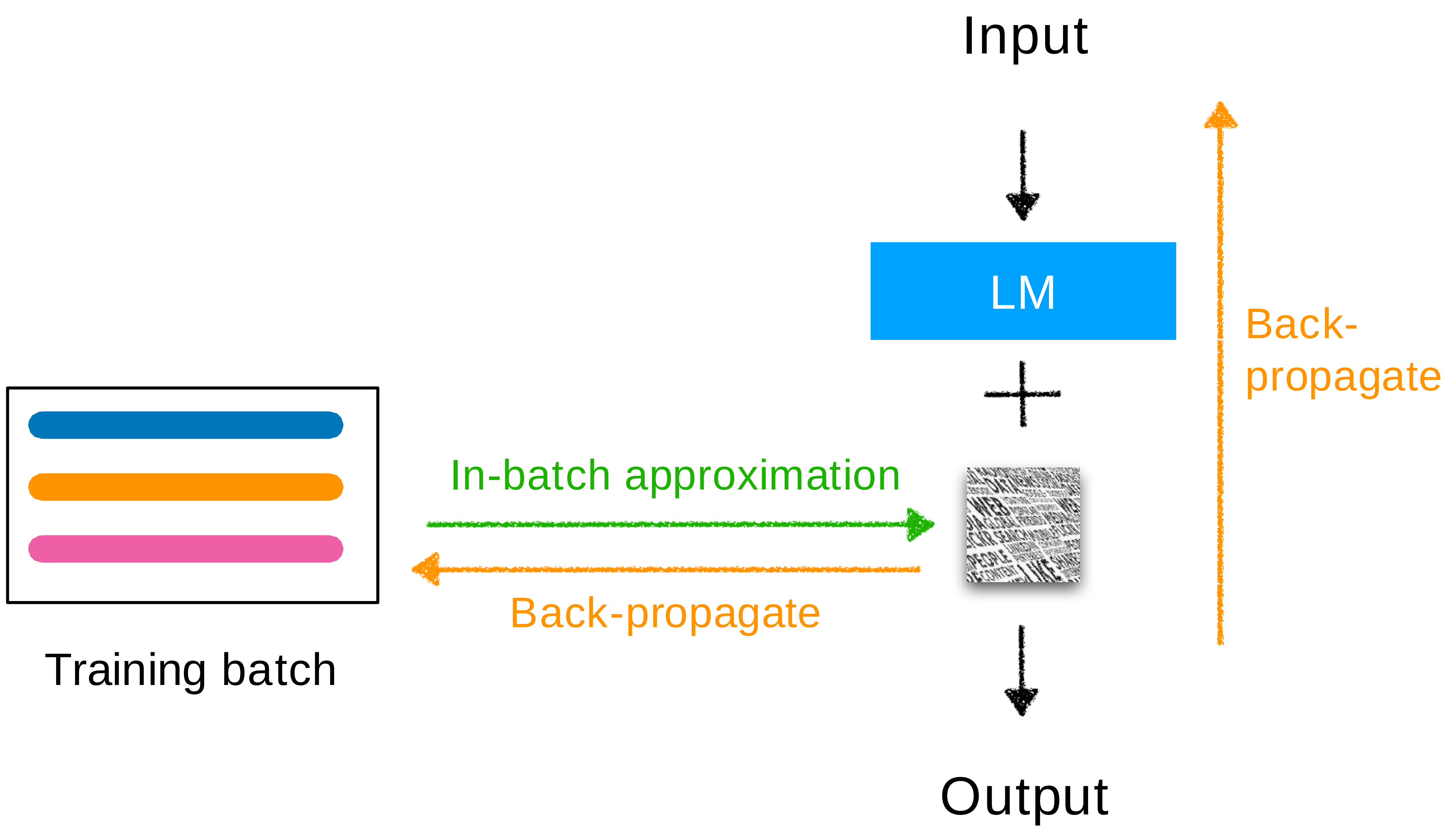


Training methods for retrieval-based LMs

- 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

Full corpus

Encoder

Full index

In-batch approximation

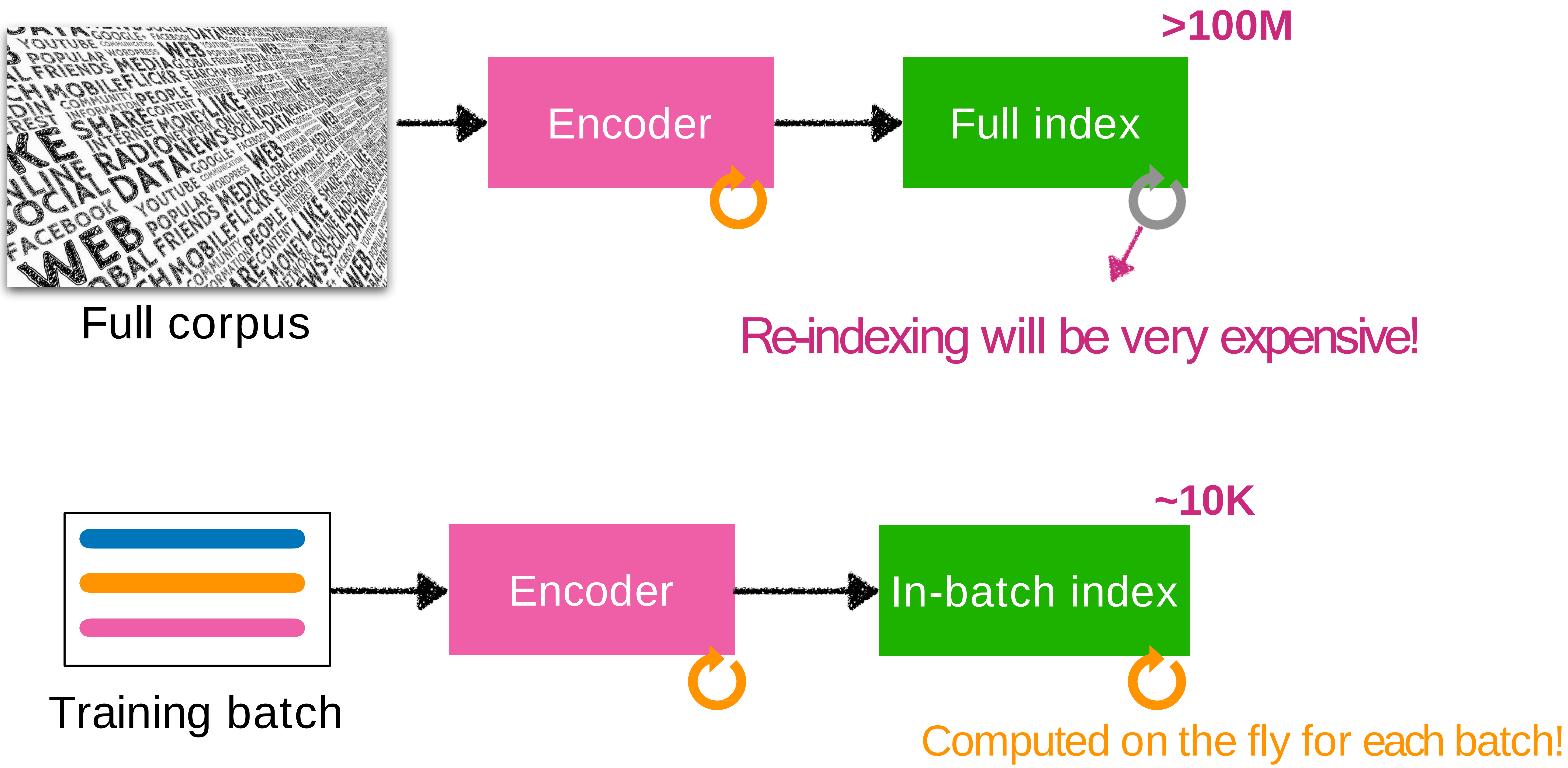
Full corpus

Encoder → Full index

>100M

Re-indexing will be very expensive!

In-batch approximation



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

Applications

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

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

…

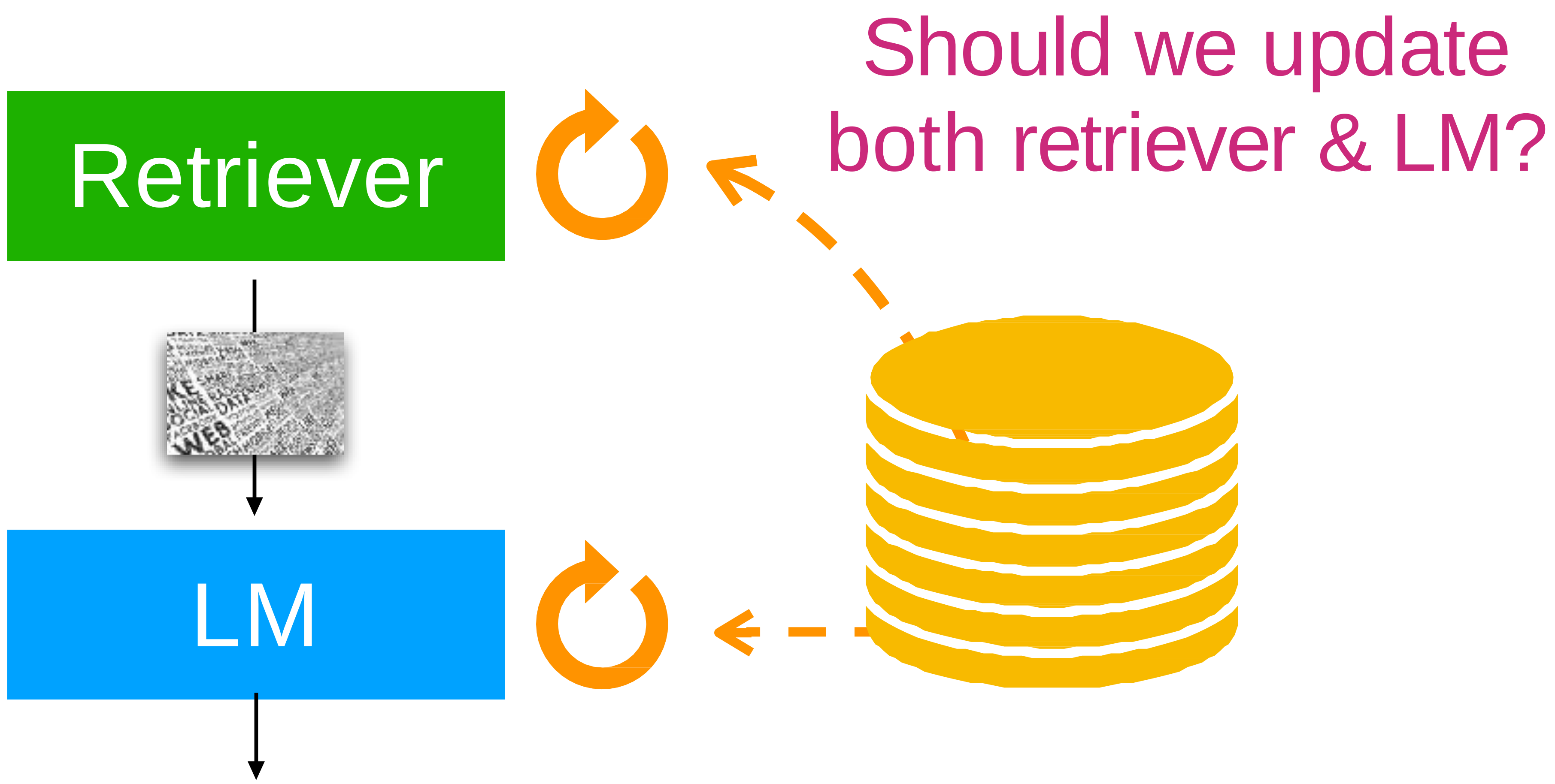
How to **adapt?**

- Fine-tuning
- Reinforcement learning
- Prompting

How to adapt a retrieval-based LM for a task

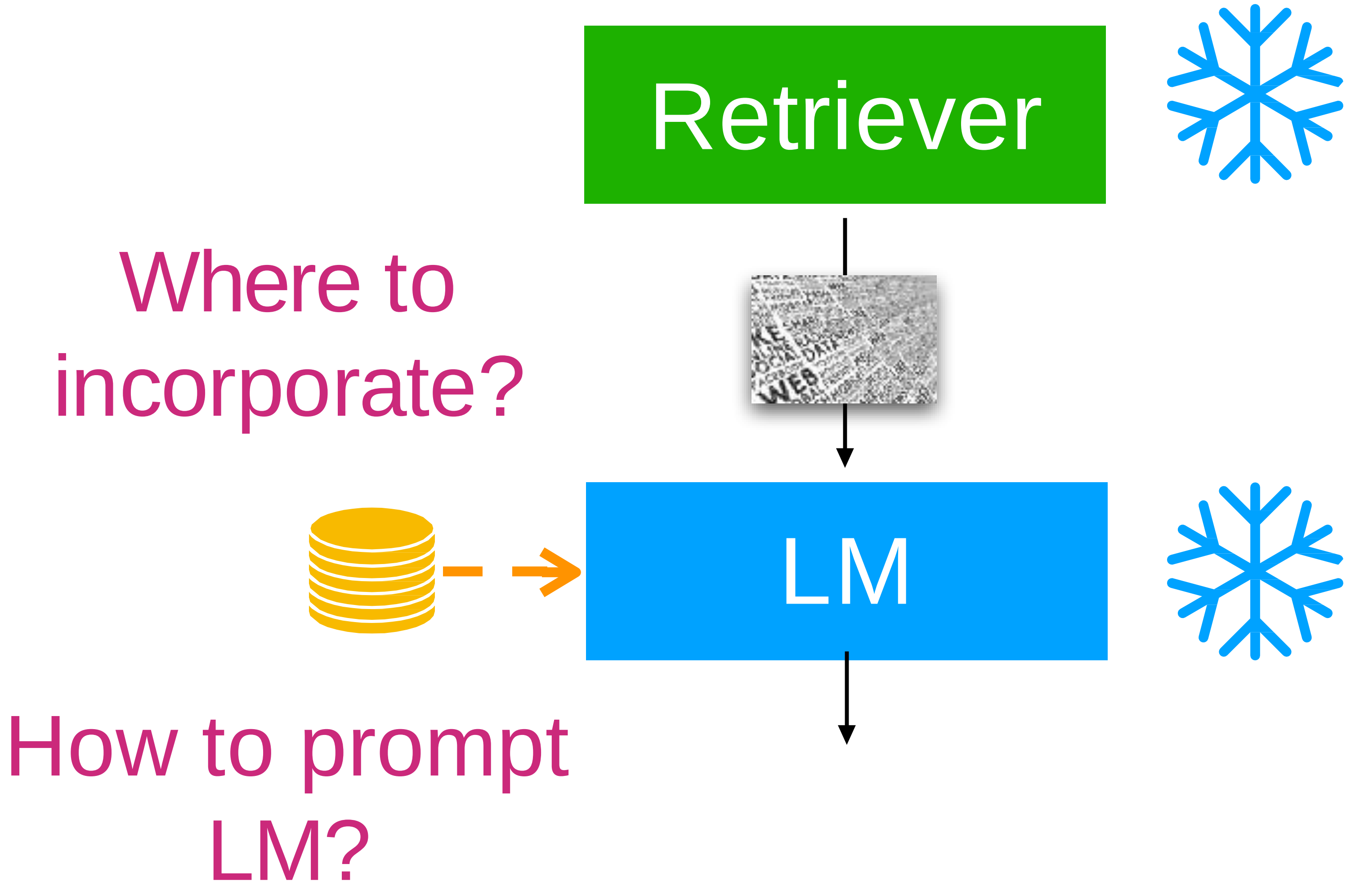
Fine-tuning (+RL)

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



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?

LM

Yes it is.

Effectiveness of retrieval-based LMs

Long-tail

knowledge update

Verifiability

Parameter-efficiency

Q: Where is Toronto Zoo located? → LM → <u>1361A</u> Old Finch Avenue, in Scarborough, Ontario

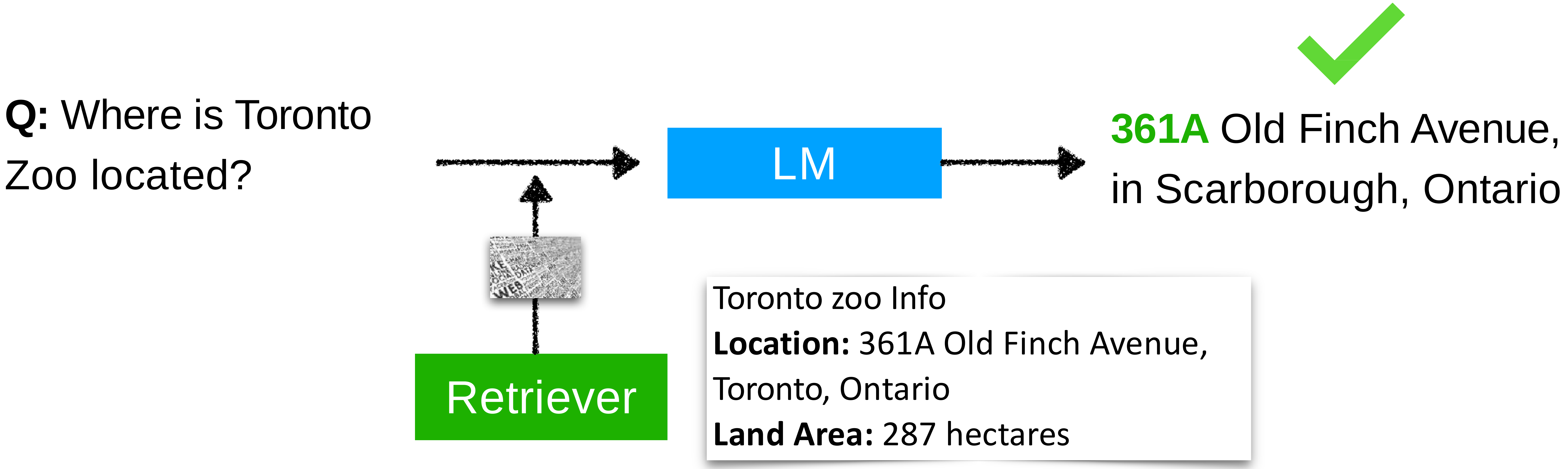
Effectiveness of retrieval-based LMs

Long-tail

knowledge update

Verifiability

Parameter-efficiency



Effectiveness of retrieval-based LMs

Long-tail

Knowledge update

Verifiability

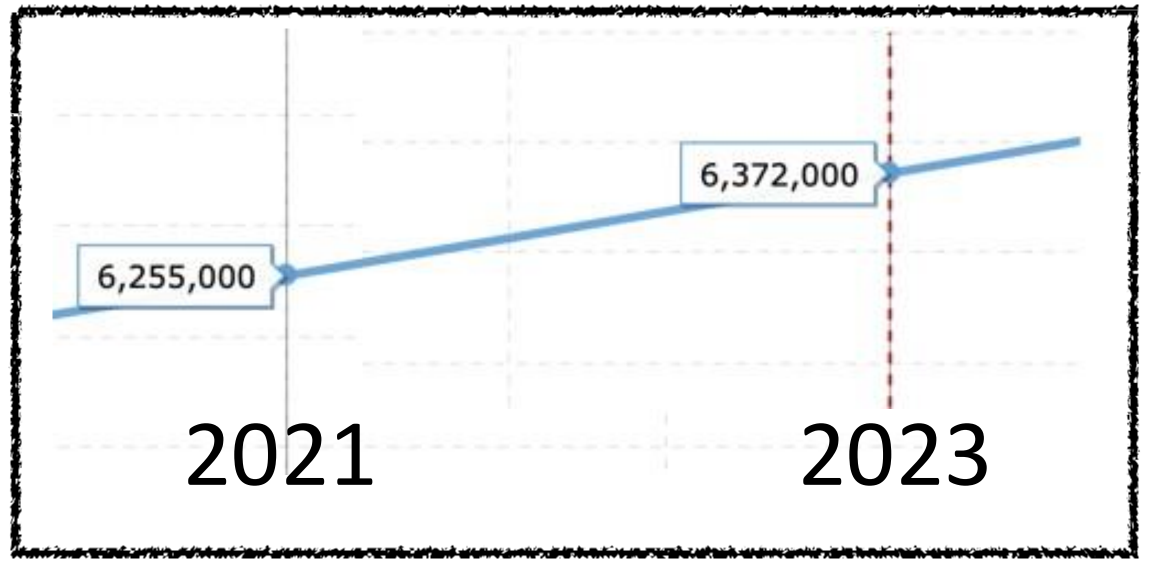
Parameter-efficiency

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

LM

Trained on the **2021** corpus

A: 6,255,000



Effectiveness of retrieval-based LMs

Long-tail

Knowledge update

Verifiability

Parameter-efficiency

Trained on the **2021** corpus

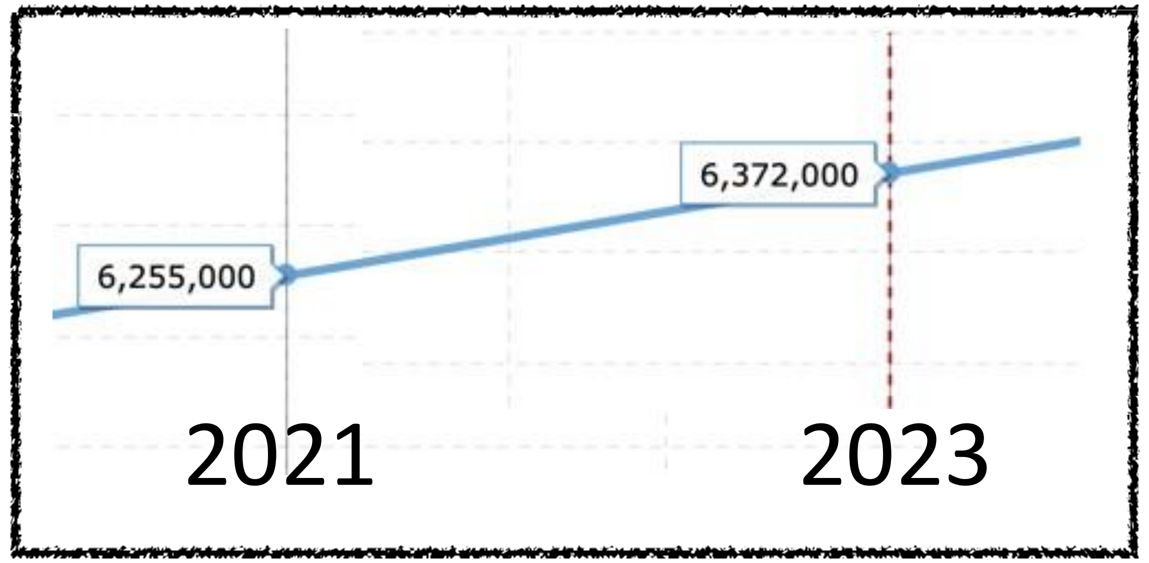
Q: What is the population of Toronto Metropolitan area in 2023?

LM

A: 6,372,000

Collected in 2023

Retriever



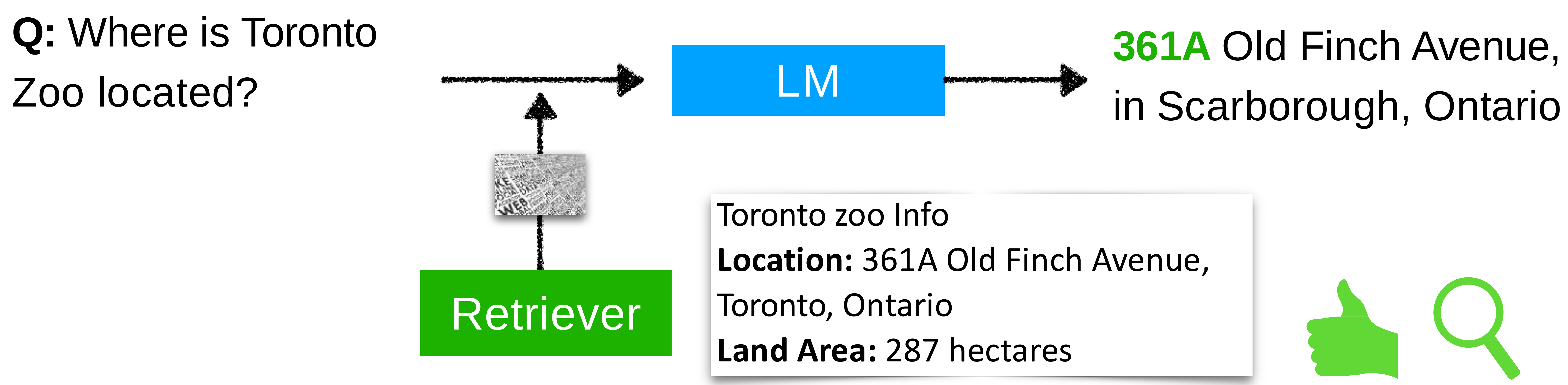
Effectiveness of retrieval-based LMs

Long-tail

knowledge update

Verifiability

Parameter-efficiency



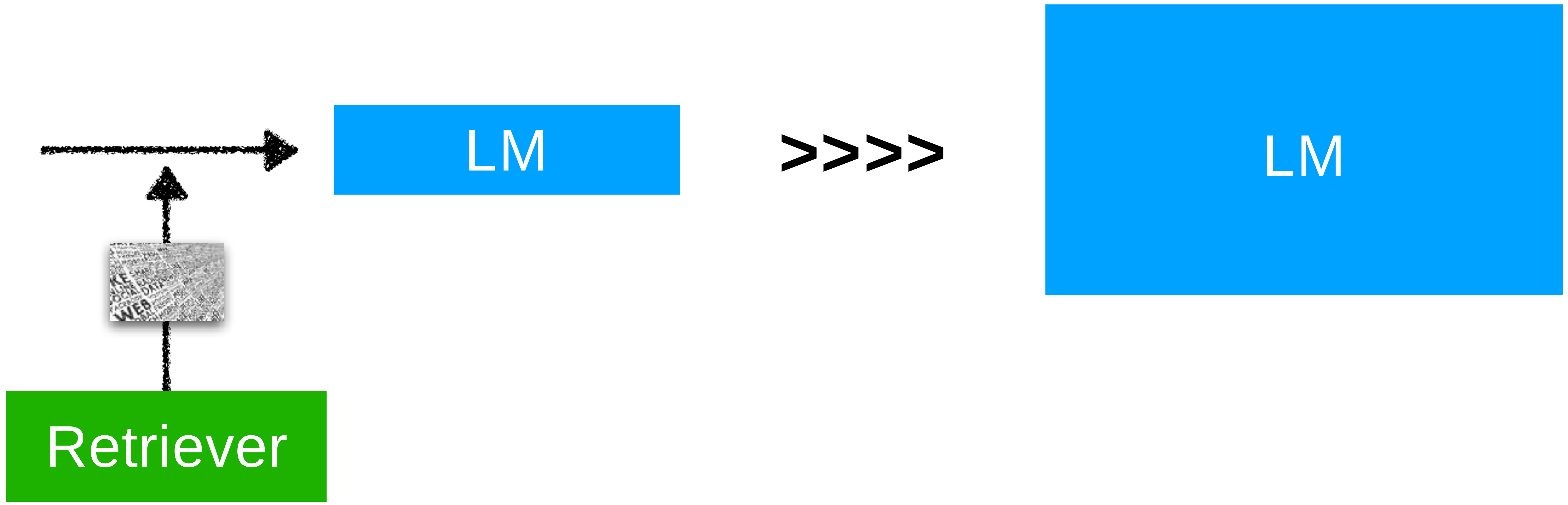
Effectiveness of retrieval-based LMs

Long-tail

knowledge update

Verifiability

Parameter-efficiency



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?**

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

Prompting

k-shot instances (k=0, 32 … etc)

Doesn't require LM training on tasks!

Q: who Is the US president
A: Joe Biden
##
Q: What is the capital of US?
A: Washington DC.
##

Training instances (demonstrations)

LM

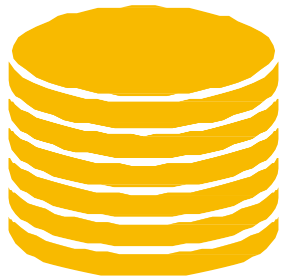
Toronto

Q: what is the Ontario capital?
A:

Test instances

Retrieval-based prompting

k-shot instances (k=0, 32 … etc)



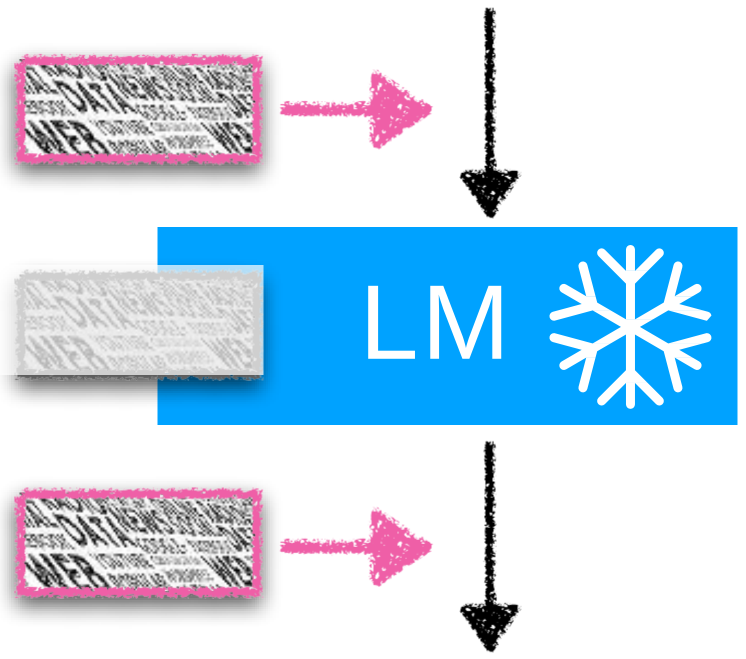
Q: who Is the US president
A: Joe Biden
##
Q: What is the capital of US?
A: Washington DC.
##
Q: what is the Ontario capital?
A:

Retriever

LM

Toronto

Design choice of retrieval-based Prompting

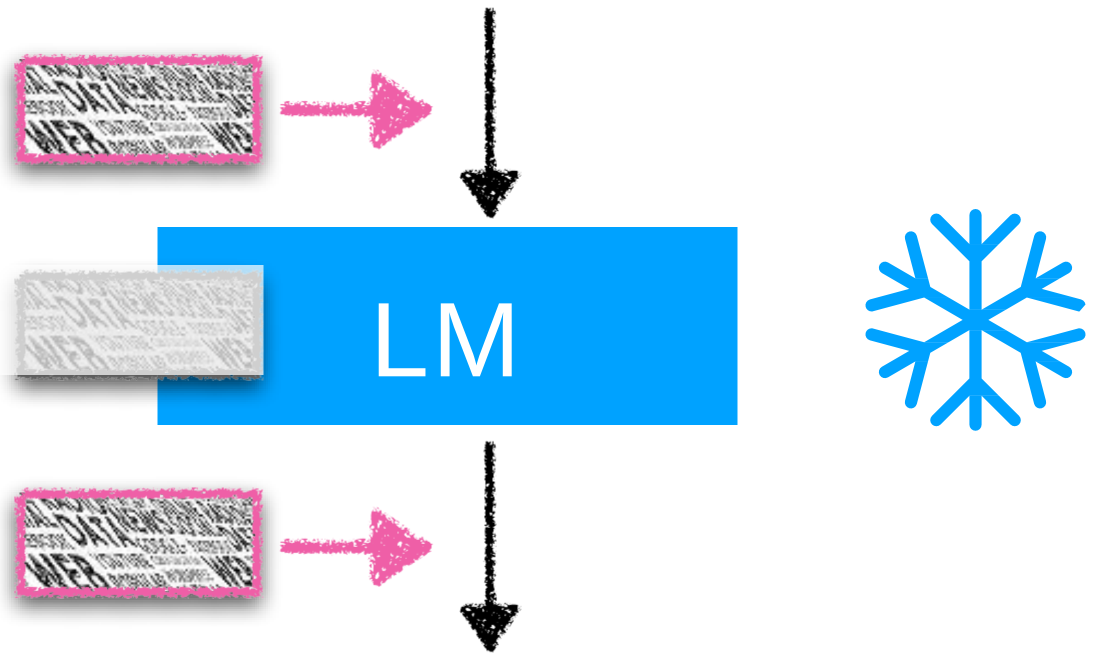


Input space:
Incorporate retrieved context in input space

Intermediate layers:
N/A

Output space:
Interpolate token probability distributions in output space

Retrieval-based Prompting

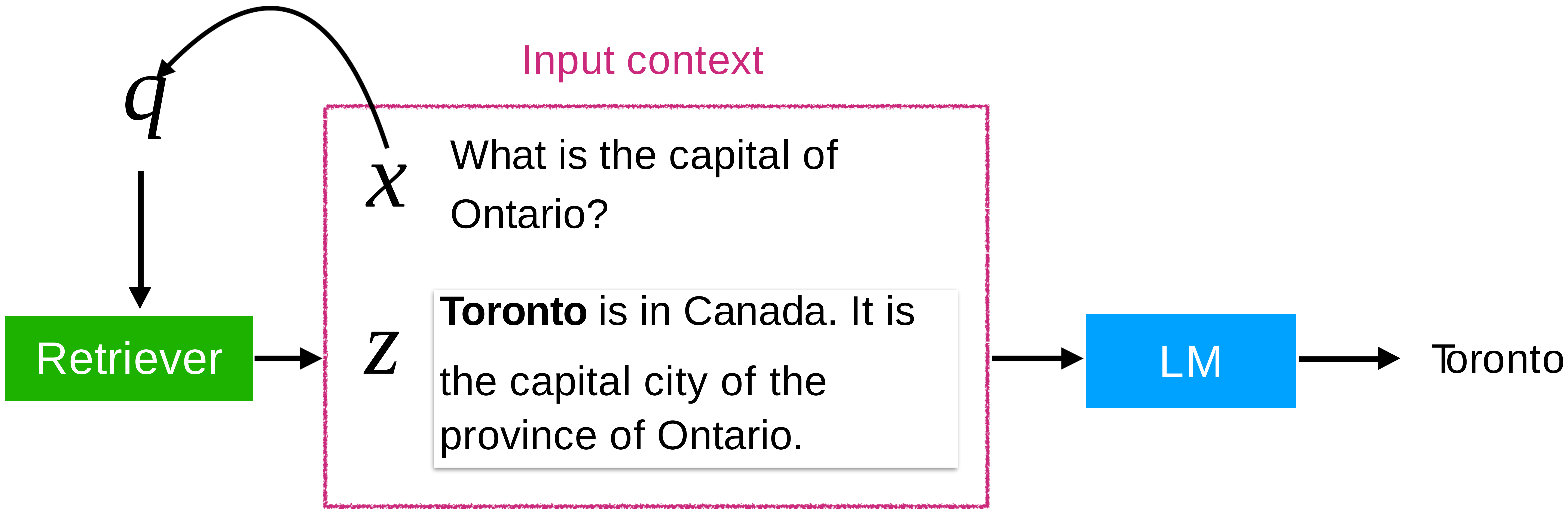


Input space:
Incorporate retrieved context in input space

Intermediate layers:
N/A

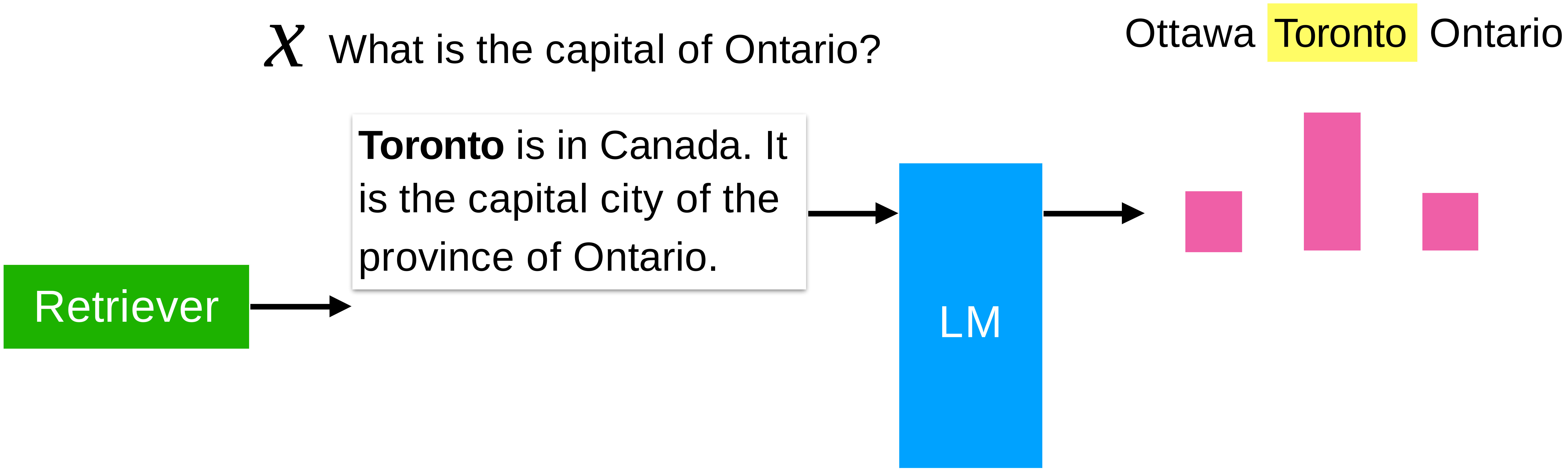
Output space:
Interpolate token probability distributions in output space

Retrieval-in-context

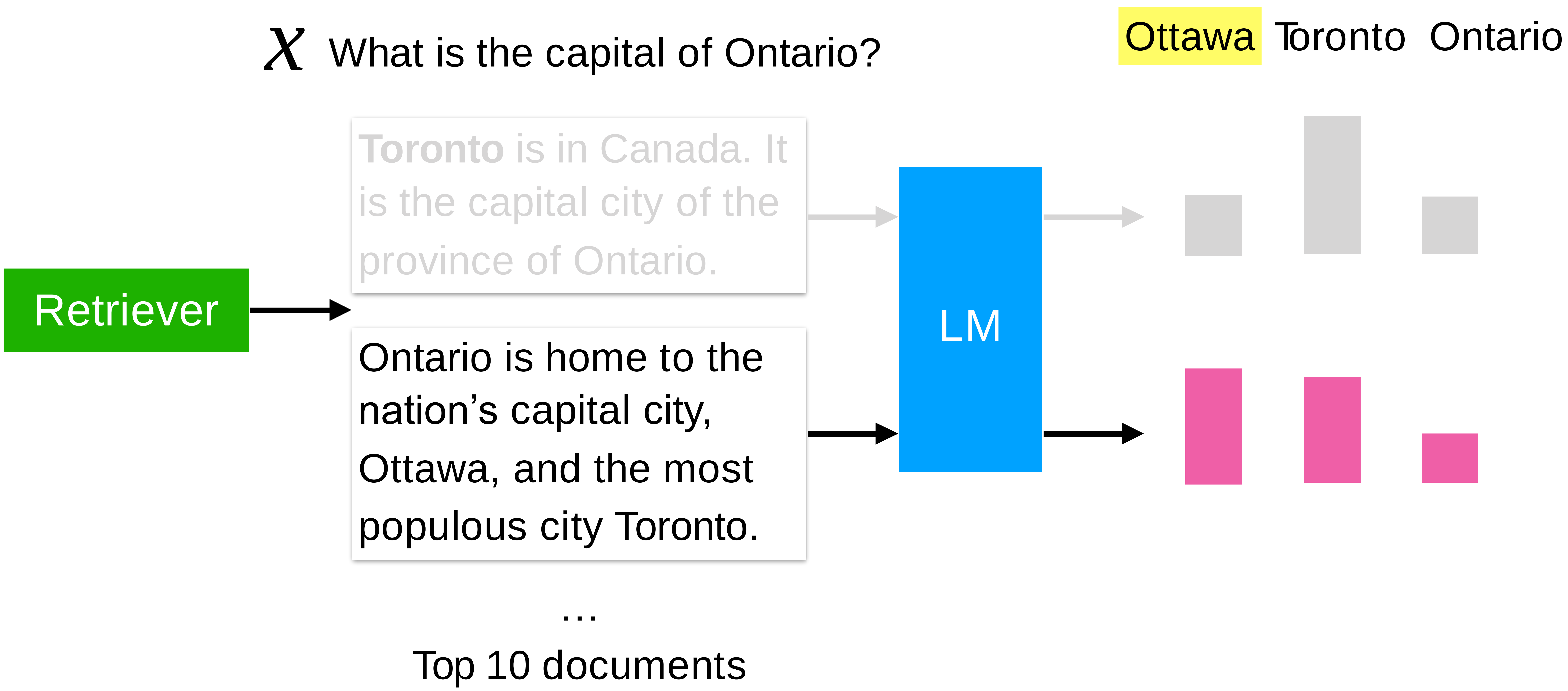


(Shi et al., 2023; Ram et al., 2022; Mallen et al., 2022; Yu et al., 2022; Press et al., 2022; *inter alia*)

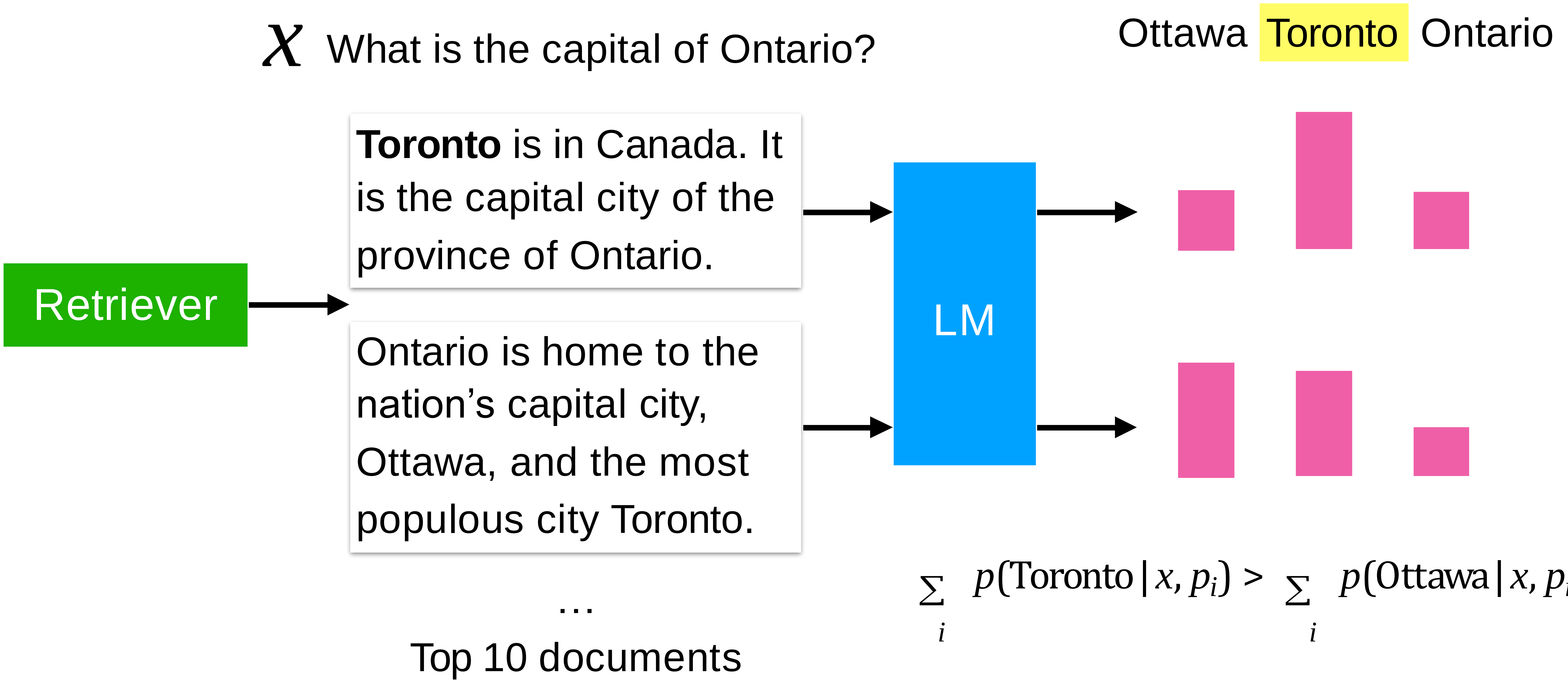
REPLUG (Shi et al., 2023; Section 3&4)



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

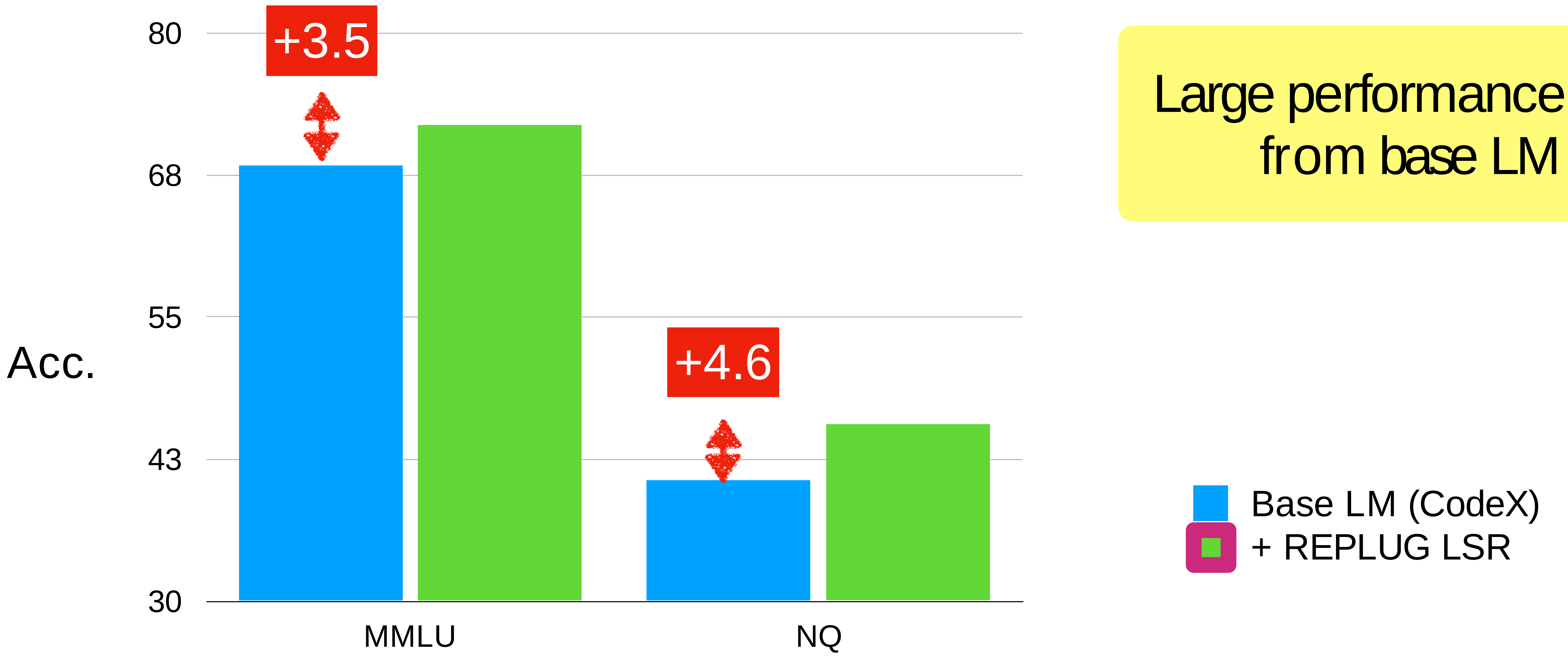


REPLUG (Shi et al., 2023; Section 3&4)



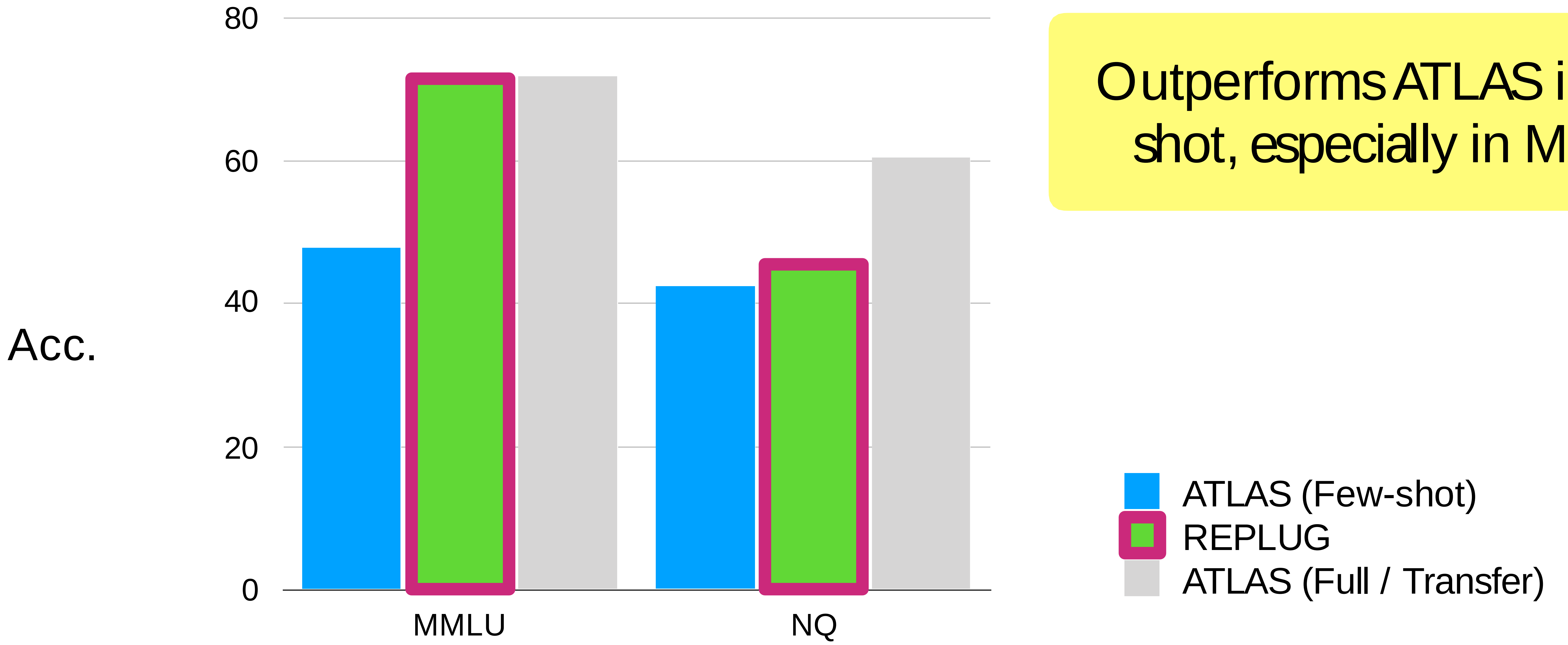
$$\sum_i p(\text{Toronto} \mid x, p_i) > \sum_i p(\text{Ottawa} \mid x, p_i)$$

REPLUG: Results on QA & MMLU



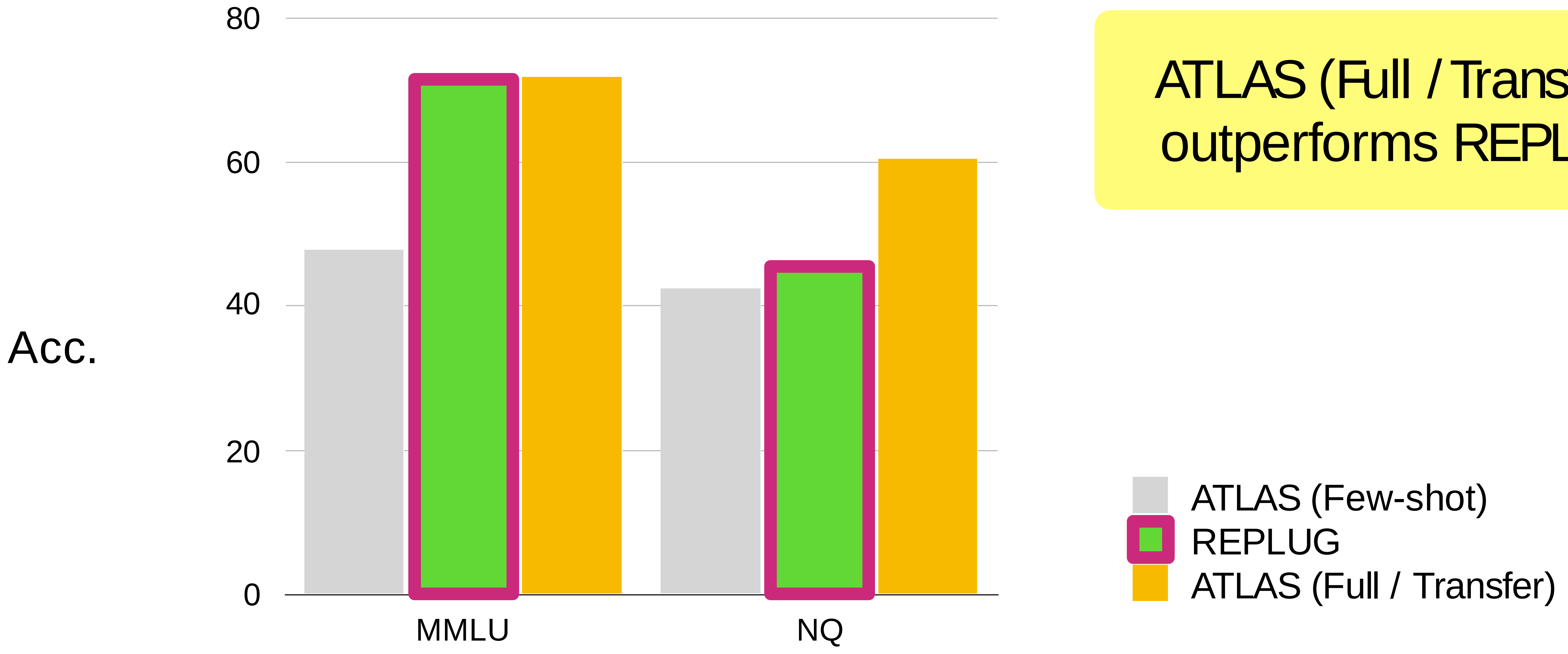
Large performance gain from base LM

REPLUG: Comparison with ATLAS



Outperforms ATLAS in few-shot, especially in MMLU

REPLUG: Comparison with ATLAS



ATLAS (Full / Transfer) outperforms REPLUG

Summary of downstream adaptations

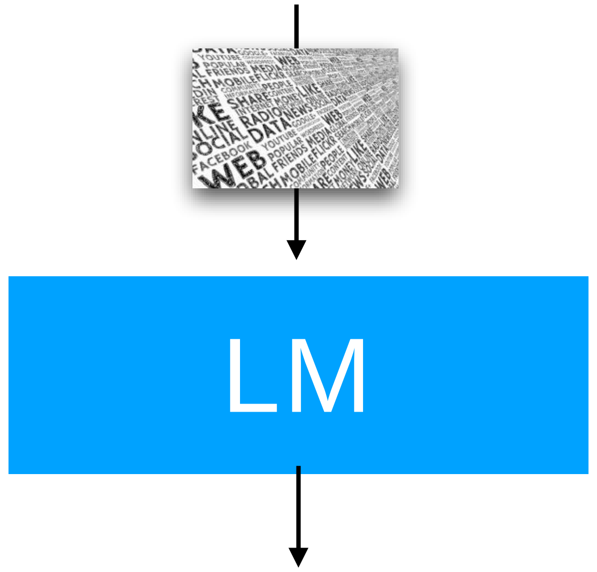
	Target task	Adaptation method	Datastore
ATLAS (Izacard et al., 2022)	Knowledge-intensive	Fine-tuning (Retriever & LM)	Wikipedia | CC
GopherCite (Menick et al., 2022)	Open-domain QA, Long-form QA	Fine-tuning + RL (LM)	Google Search Results
kNN-prompt (Shi et al., 2022)	Classification	Prompting (output)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

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 completive with more data

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

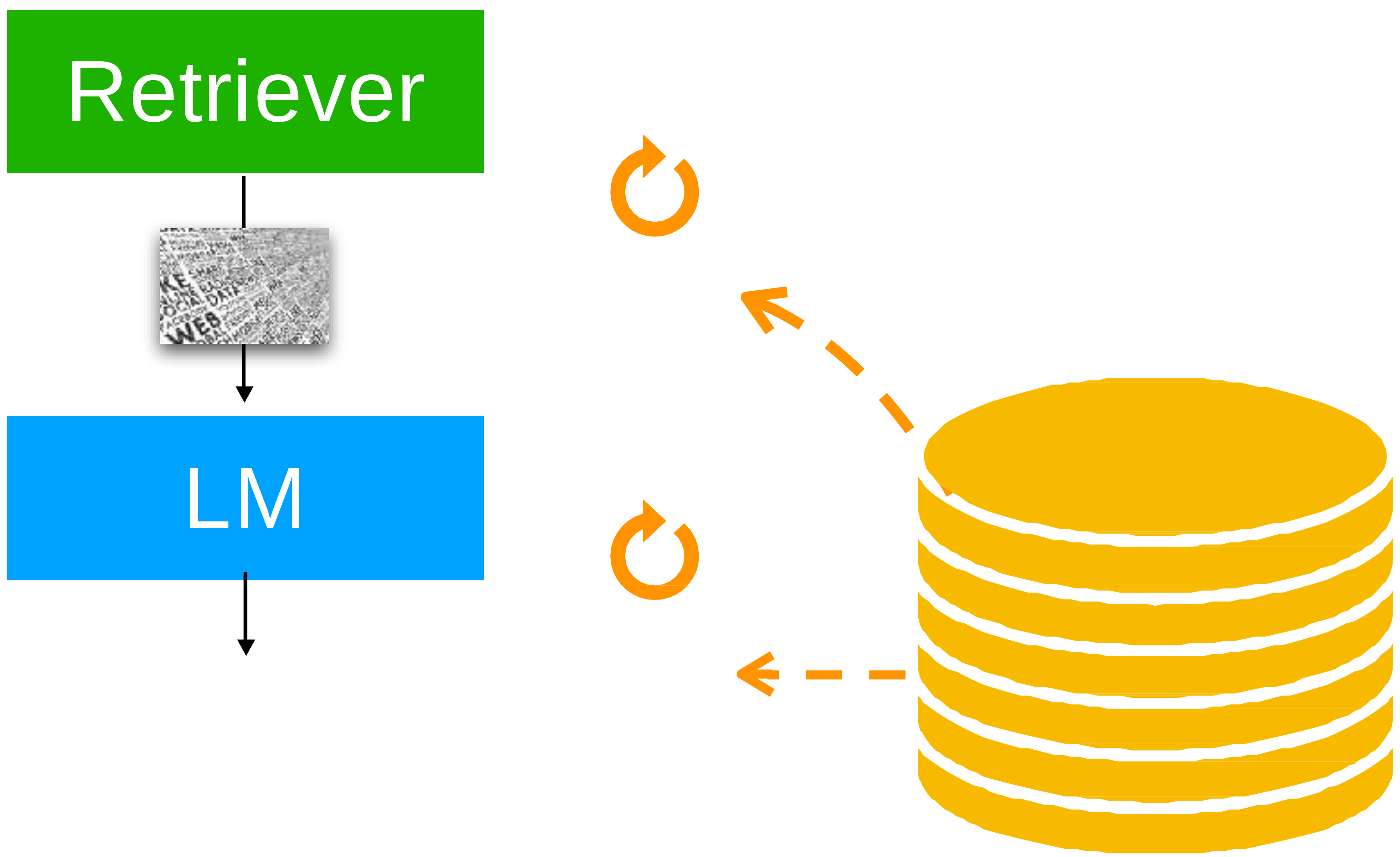
What is **data store?**

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

Adapting retrieval-based LMs for tasks

Fine-tuning

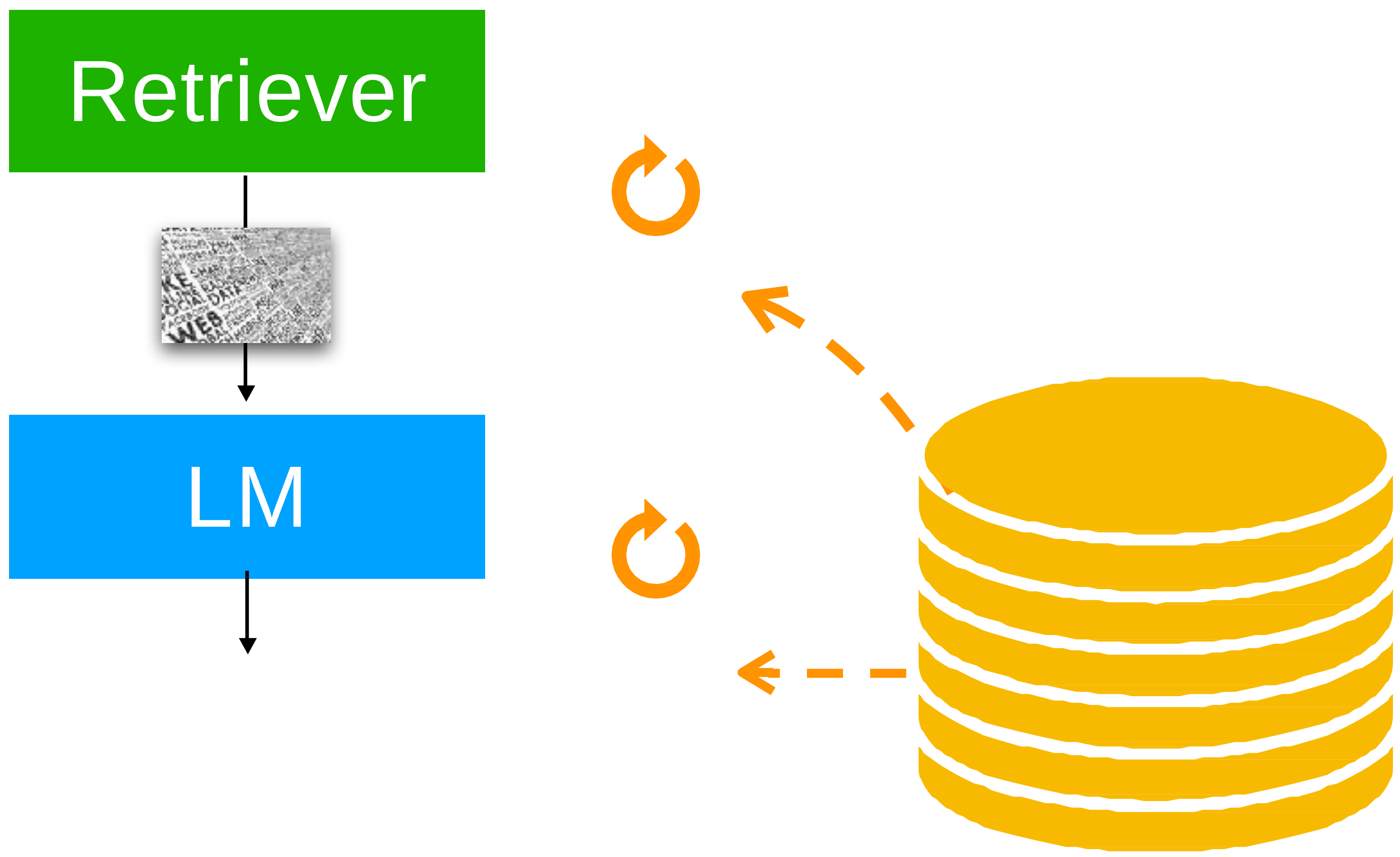
Training LM and / or retriever
on task-data & data store



Adapting retrieval-based LMs for tasks

Fine-tuning

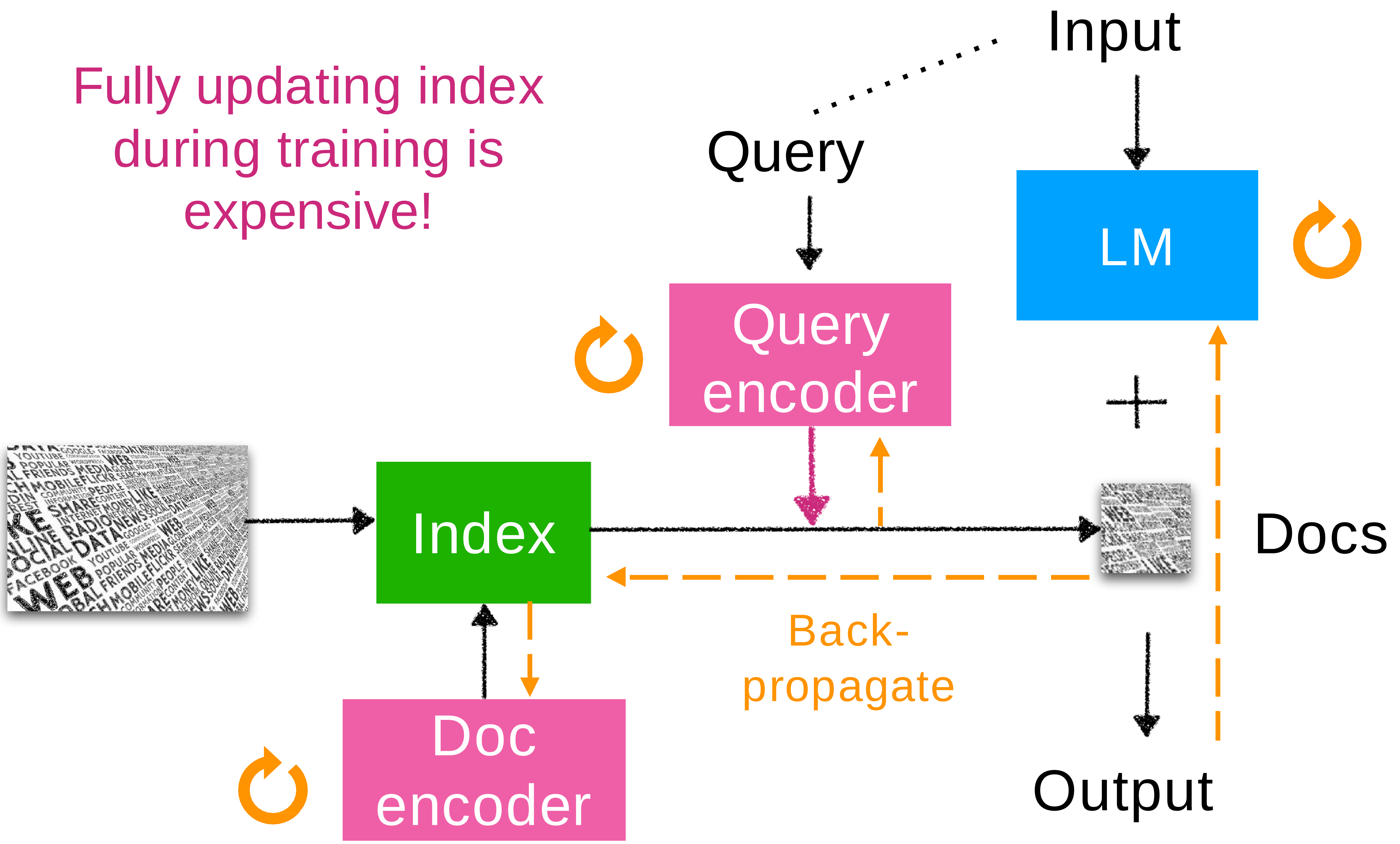
Training LM and / or retriever
on task-data & data store

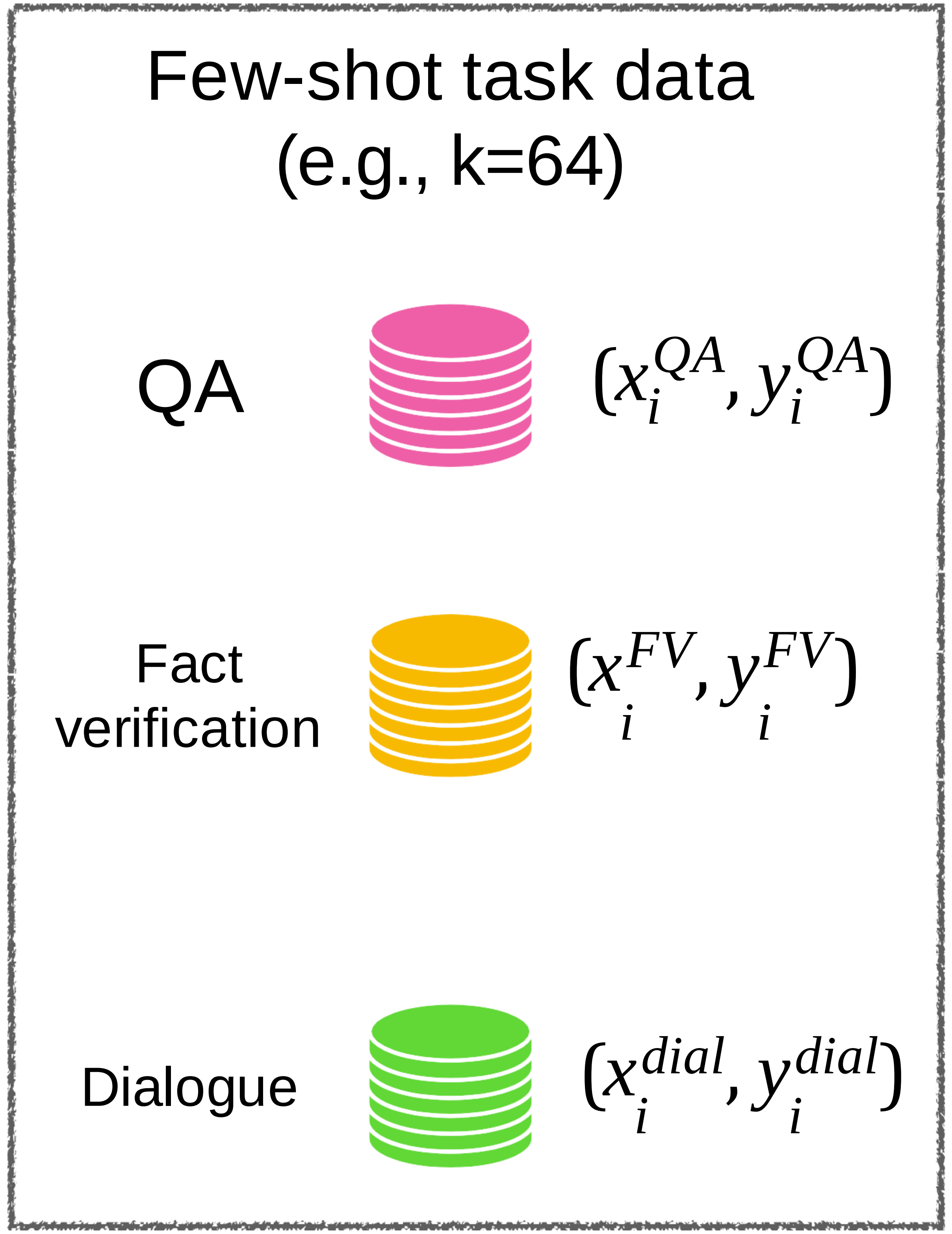


Costs of retrieval-based LM training (Section 4)

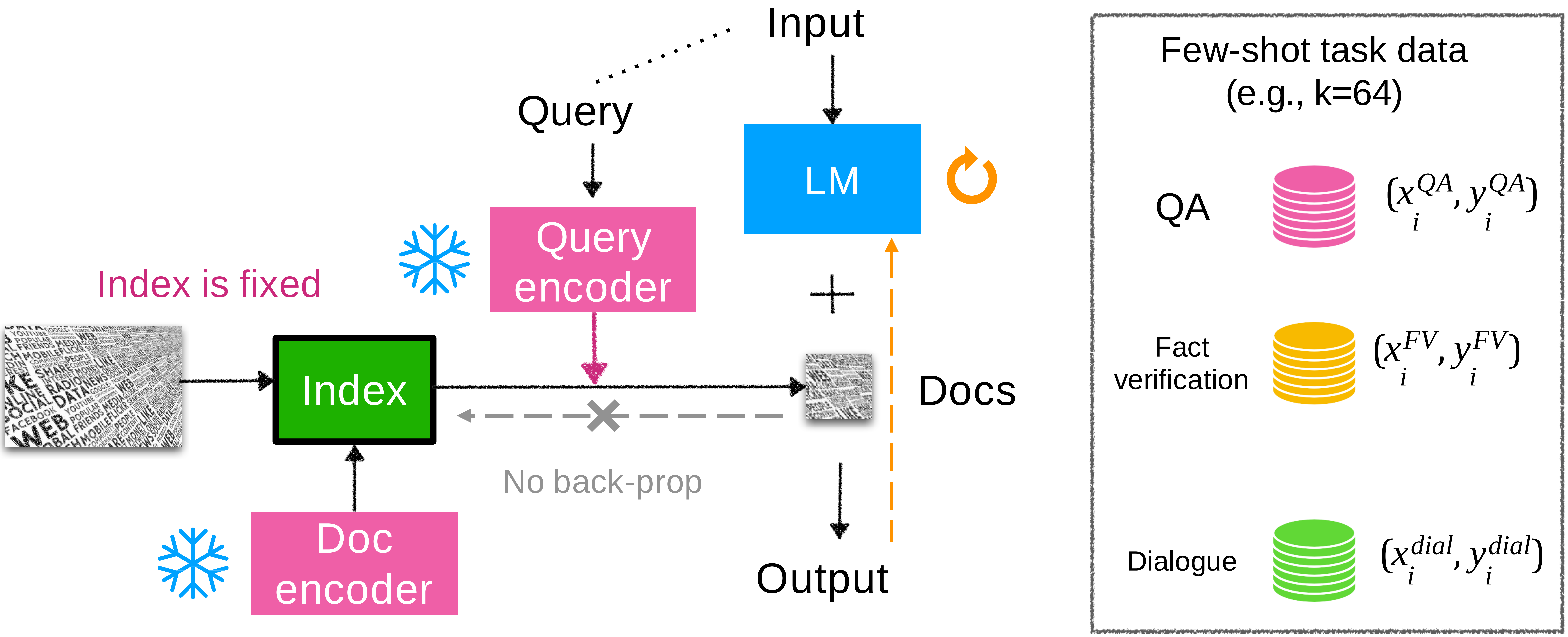
Independent training (DPR)
Asynchronous updates (REALM)
…

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

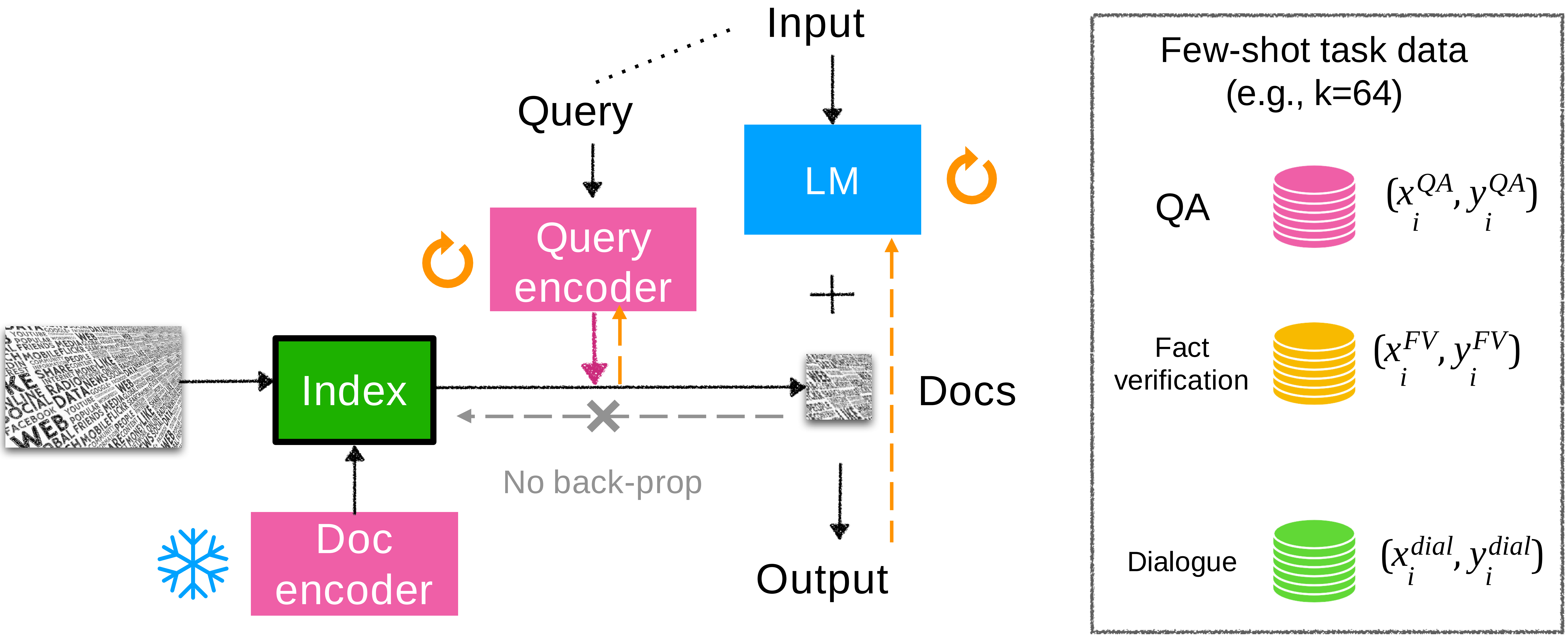




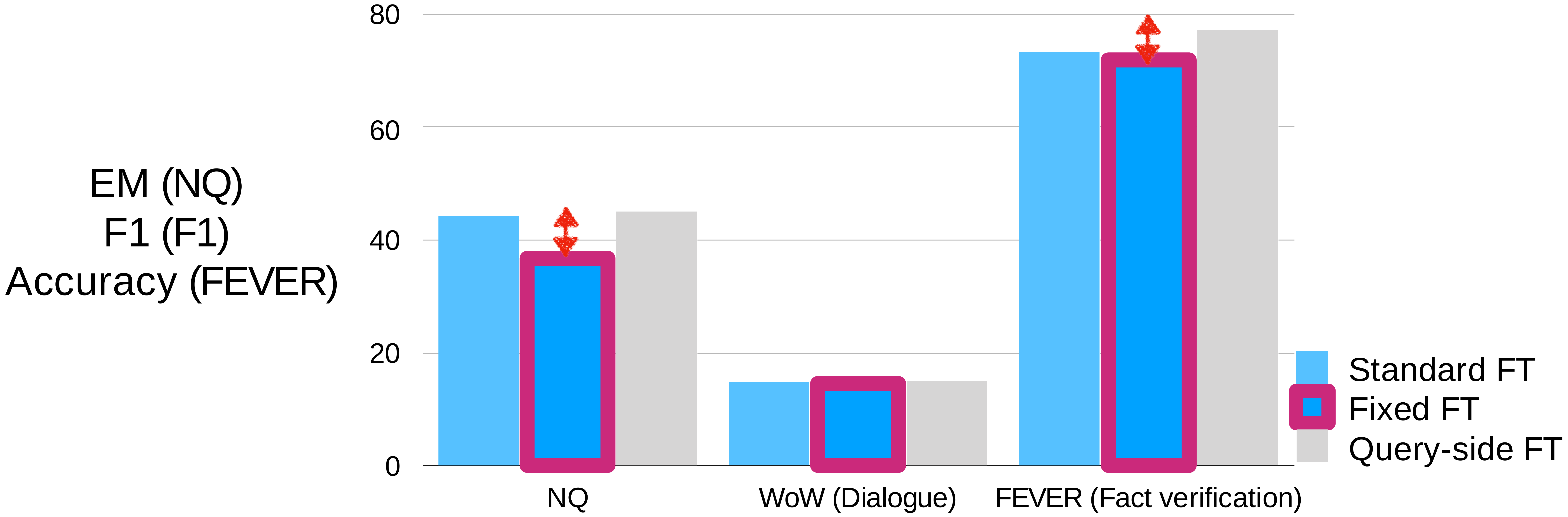
ATLAS: Fixed retrieval with fine-tuned LM



ATLAS: Query-side fine-tuning

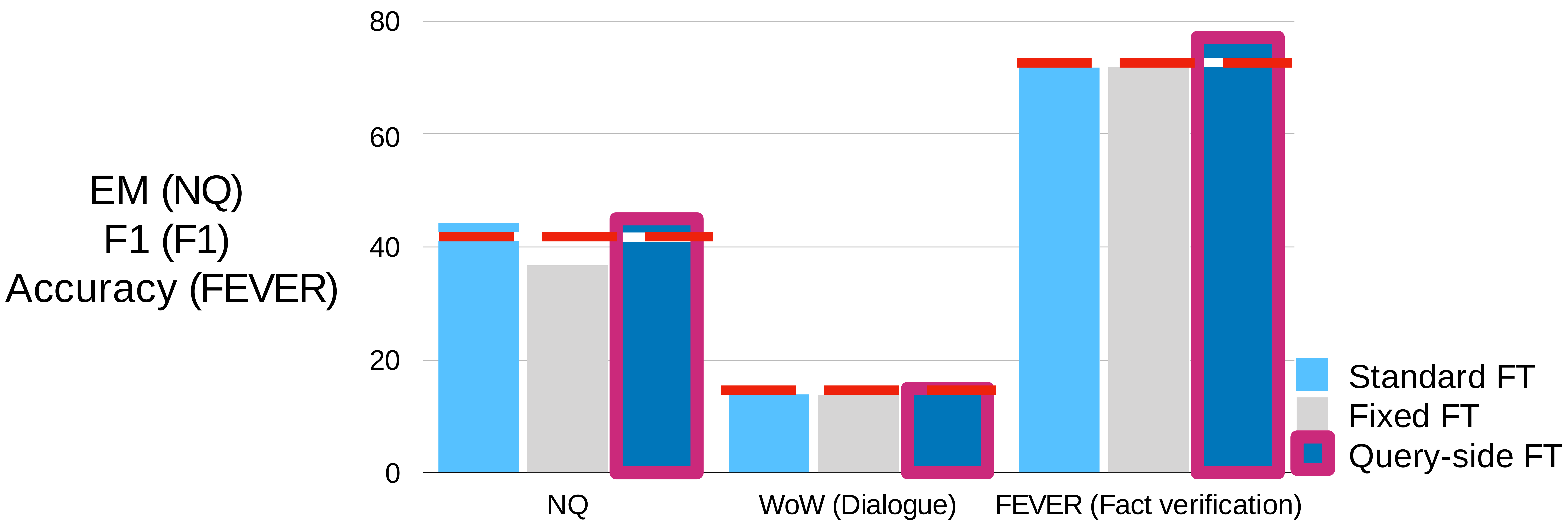


Ablations of efficient retrieval training



Fixed FT shows large performance drop on QA.

Ablations of efficient retrieval training



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

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

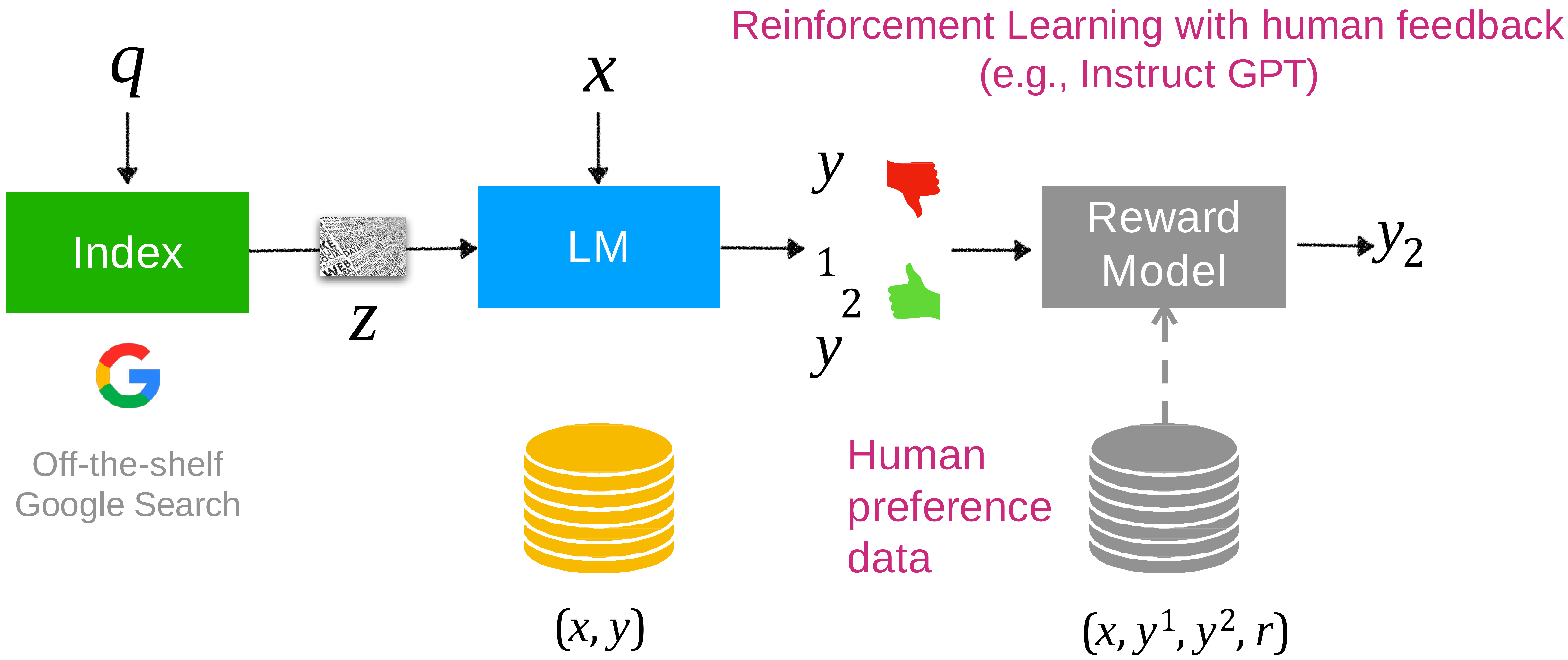
How to **adapt?**

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

What is **data store?**

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

GopherCite: RLHF for answering with verified quotes



References

Architecture:
REALM: Retrieval-Augmented Language Model Pre-Training (Guu et al., 2020)
In-Context Retrieval-Augmented Language Models (Ram et al., 2023)
REPLUG: Retrieval-Augmented Black-Box Language Models (Shi et al., 2023)
Improving language models by retrieving from trillions of tokens (Borgeaud et al., 2022)
Generalization through Memorization: Nearest Neighbor Language Models (Khandelwal et al., 2020)

Training:
Dense Passage Retrieval for Open-Domain Question Answering (Karpukhin et al., 2020)
Improving language models by retrieving from trillions of tokens (Borgeaud et al., 2022 ;also in Section 3)
Atlas: Few-shot Learning with Retrieval Augmented Language Models (Izacard et al., 2022)
Training Language Models with Memory Augmentation (Zhong et al., 2022)

Application:
Atlas: Few-shot Learning with Retrieval Augmented Language Models (Izacard et al., 2022; also in Section 4)
Teaching language models to support answers with verified quotes (Menick et al., 2022)
REPLUG: Retrieval-Augmented Black-Box Language Models (Shi et al., 2023; also in Section 3)

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

Thank you