Section 3:
Retrieval-based LM: Architecture

Categorization of retrieval-based LMs

Categorization of retrieval-based LMs

What to retrieve?

Query

Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?

Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?
Tokens?

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

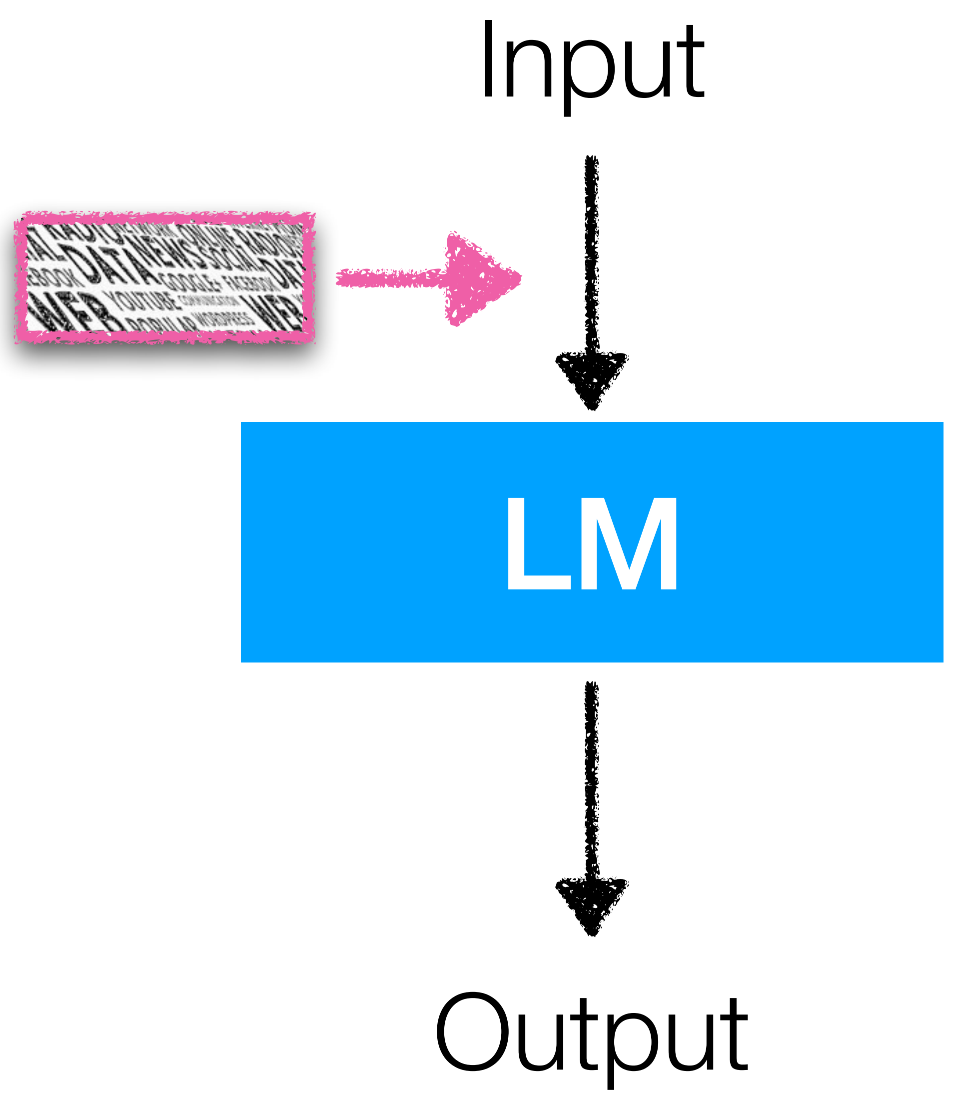
Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



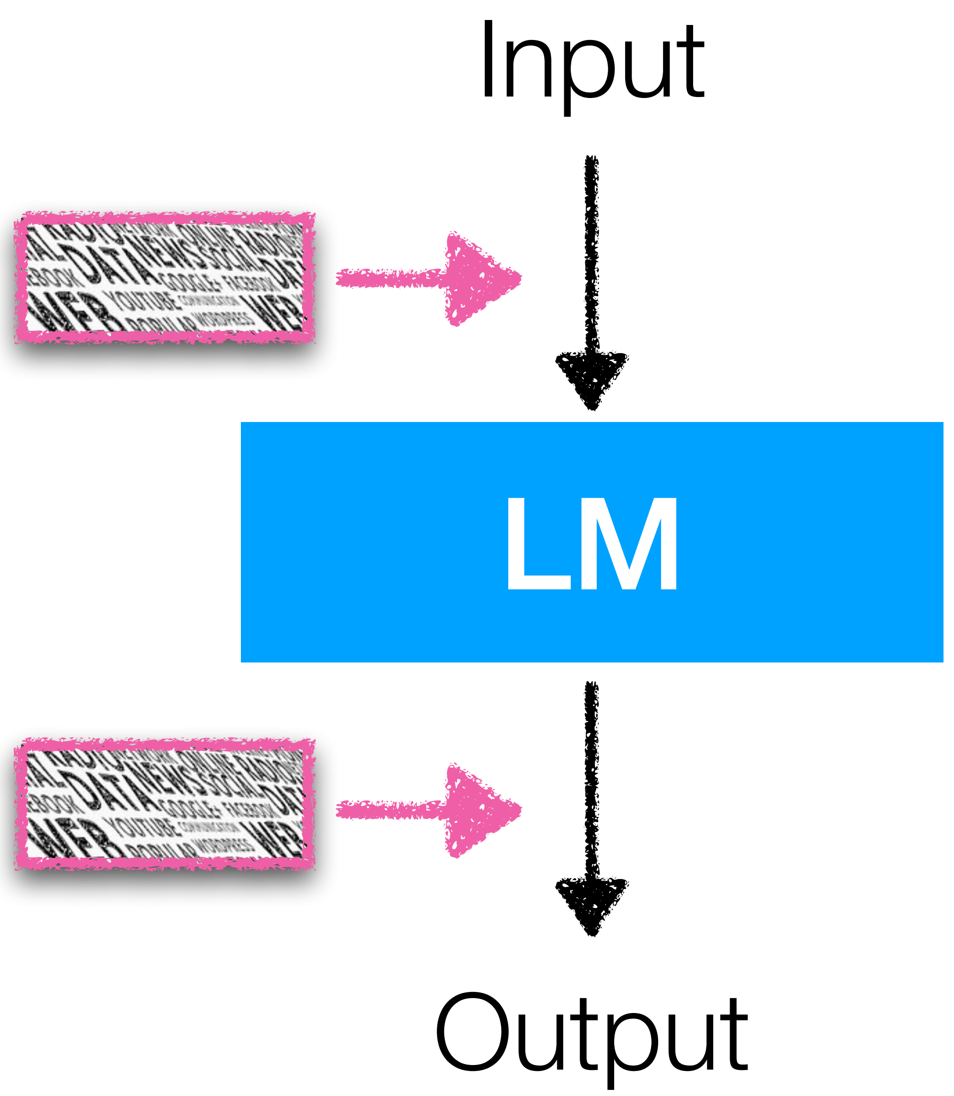
Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



Categorization of retrieval-based LMs

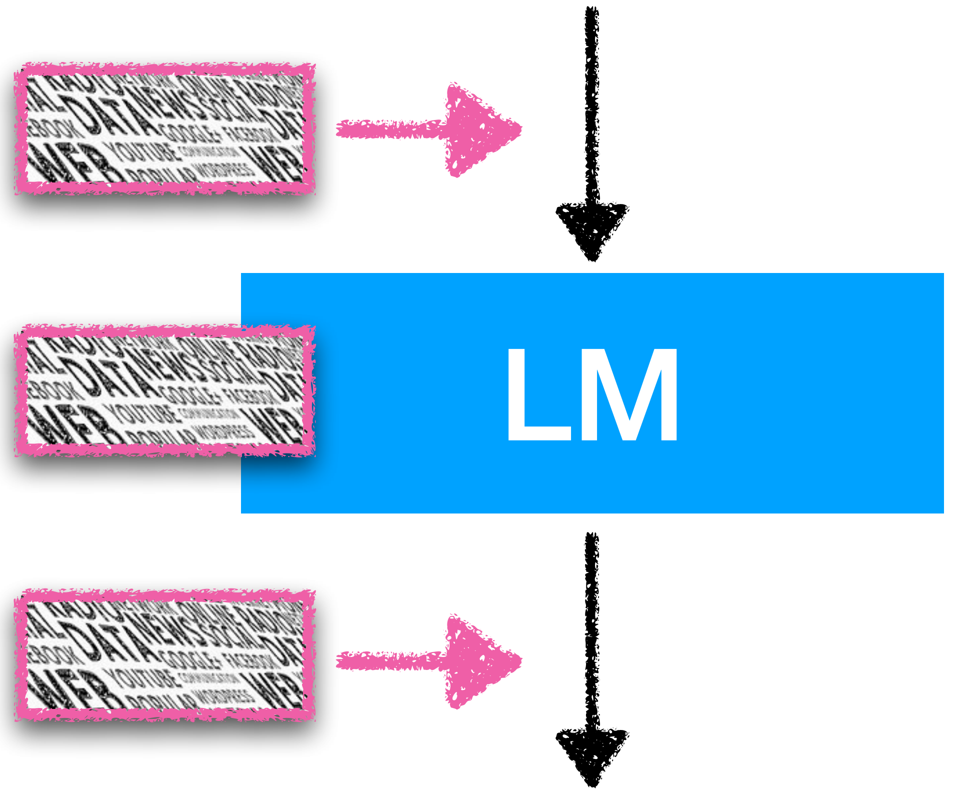
What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?

Input



Output

Categorization of retrieval-based LMs

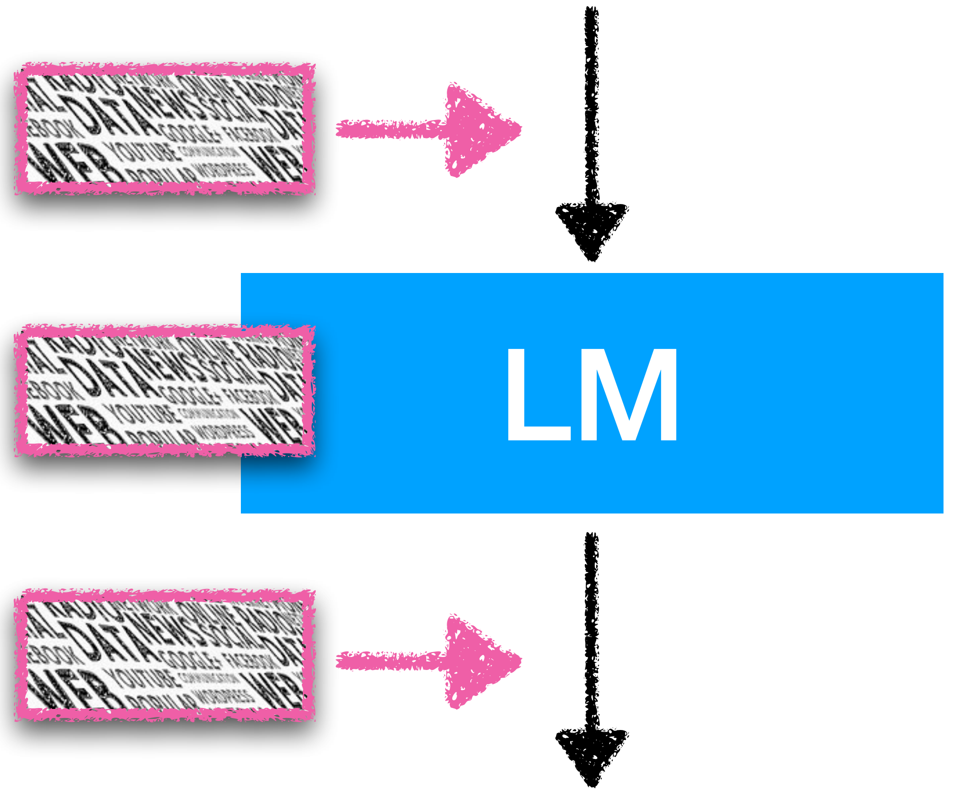
What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?

Input



Output

When to retrieve?

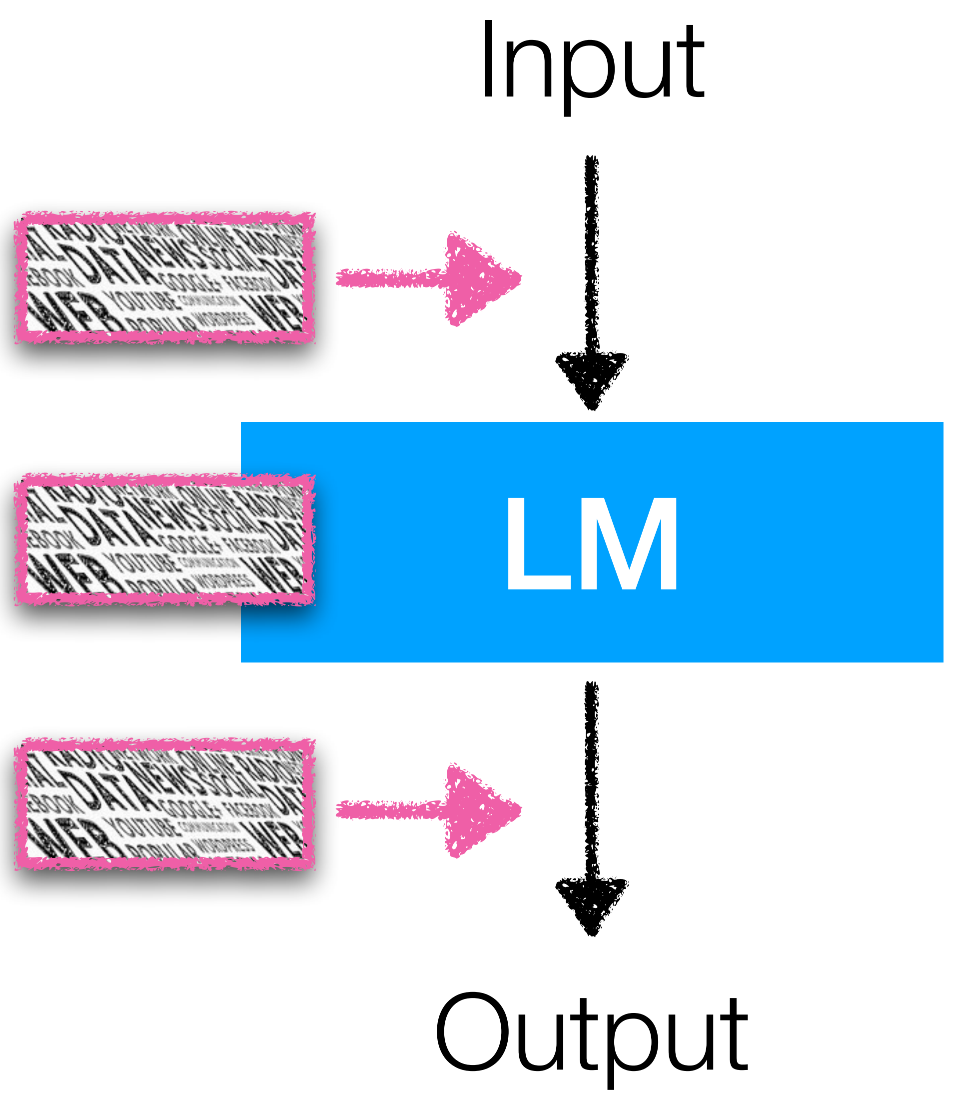
Categorization of retrieval-based LMs

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.

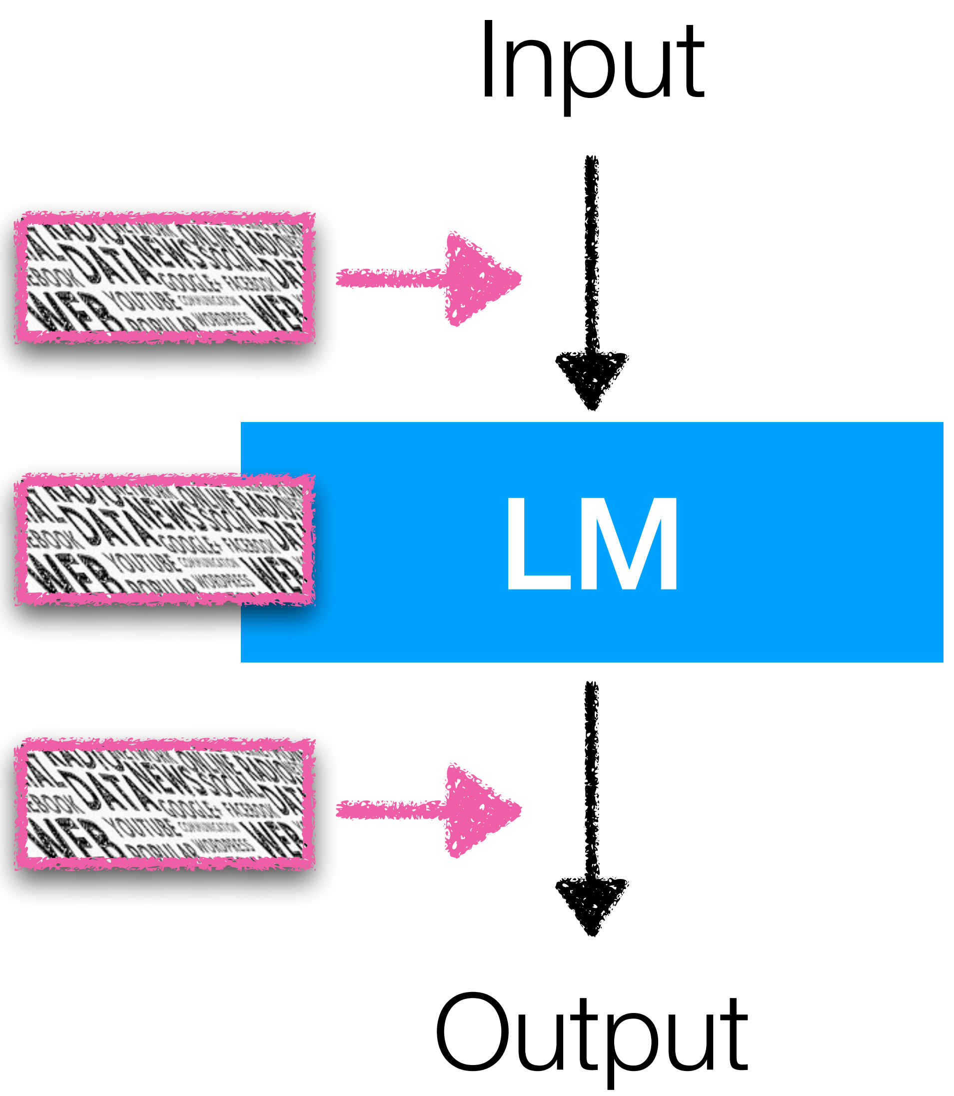
Categorization of retrieval-based LMs

What to retrieve?

Query

Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



When to retrieve?

w/ retrieval

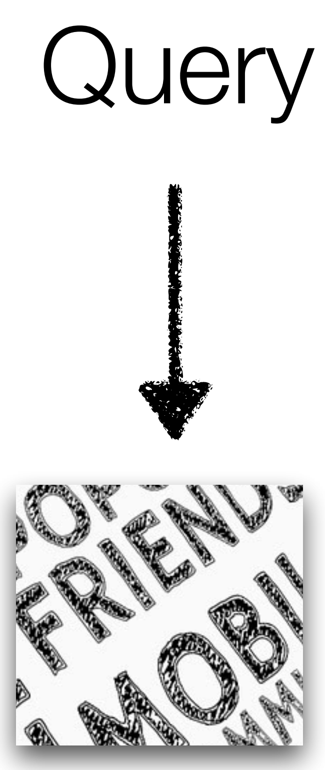
The capital city of Ontario is Toronto.

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

The capital city of Ontario is Toronto.

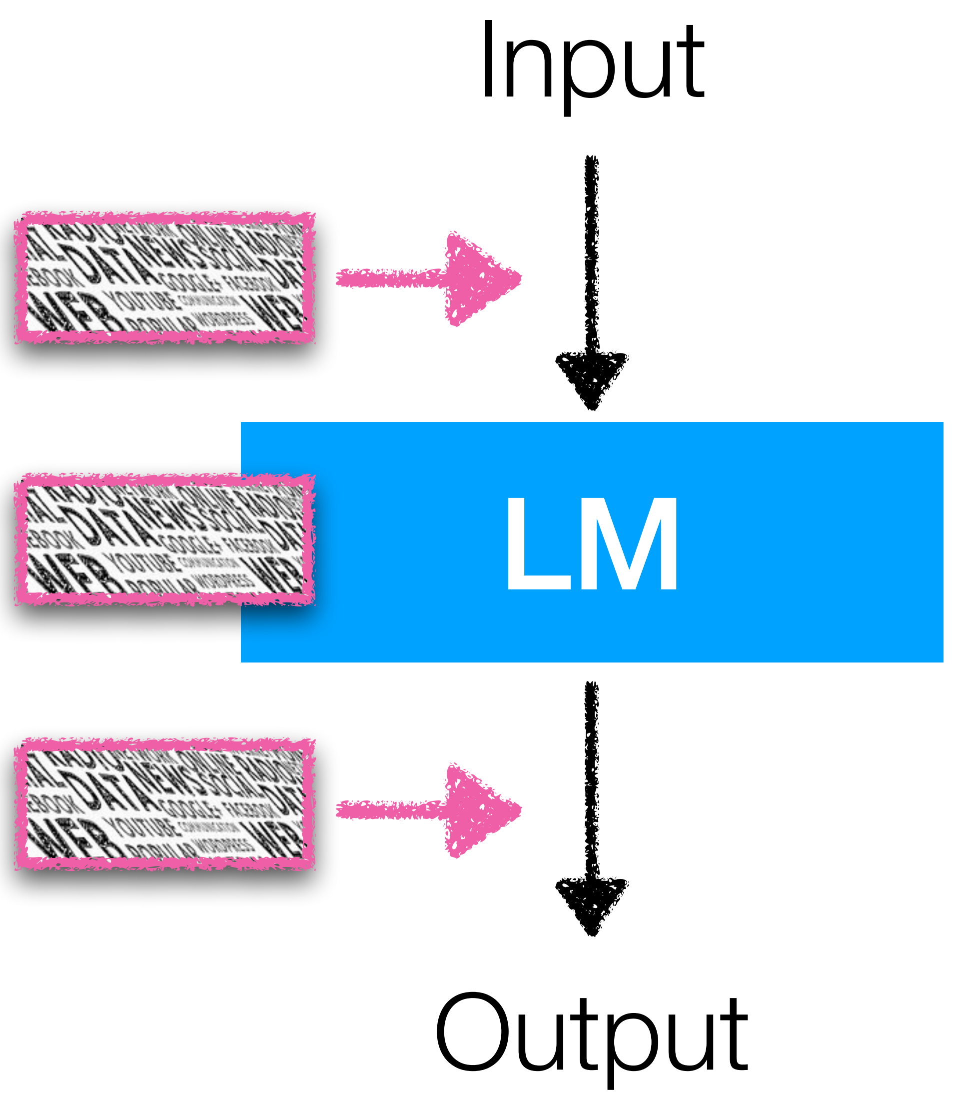
Categorization of retrieval-based LMs

What to retrieve?



Text chunks (passages)?
Tokens?
Something else?

How to use retrieval?



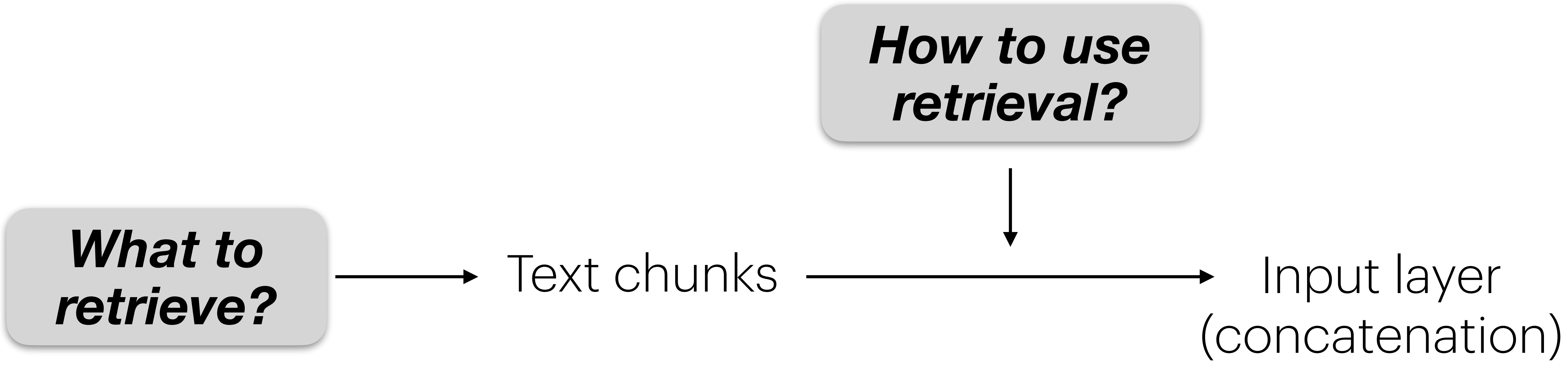
When to retrieve?

Roadmap

Roadmap

What to retrieve? → Text chunks

Roadmap


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

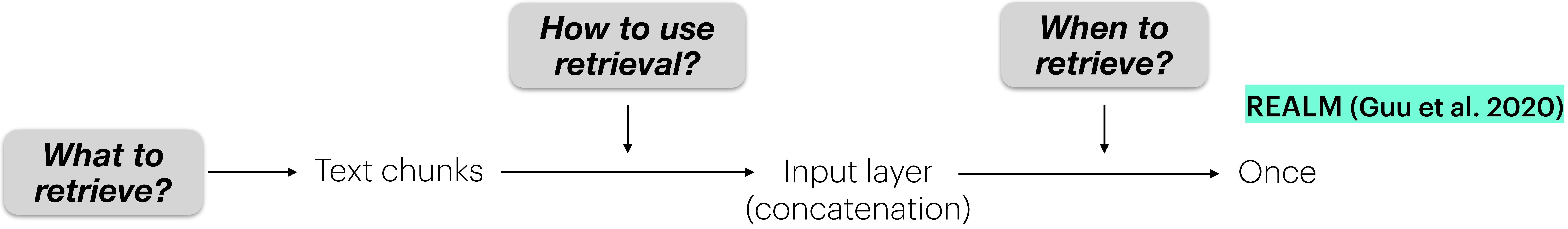
Roadmap

How to use retrieval?

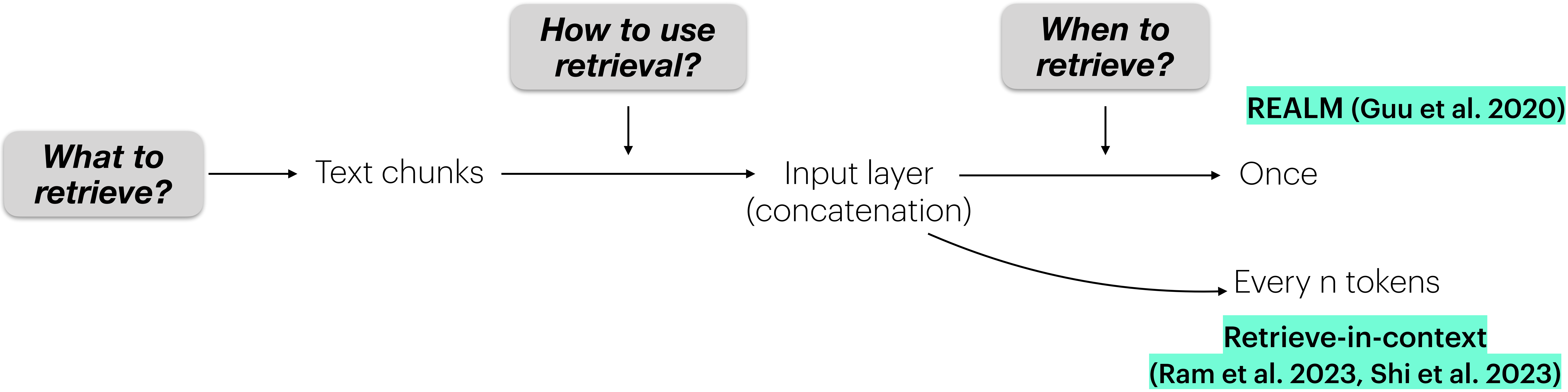
When to retrieve?

What to retrieve? → Text chunks → Input layer (concatenation) → Once

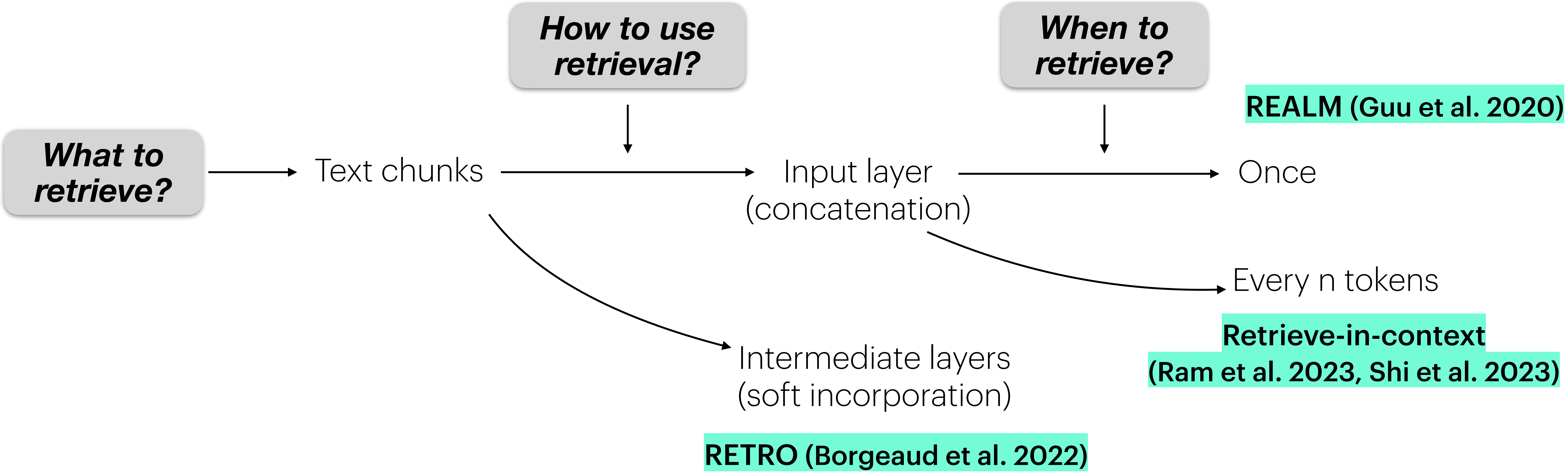
Roadmap


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

Roadmap

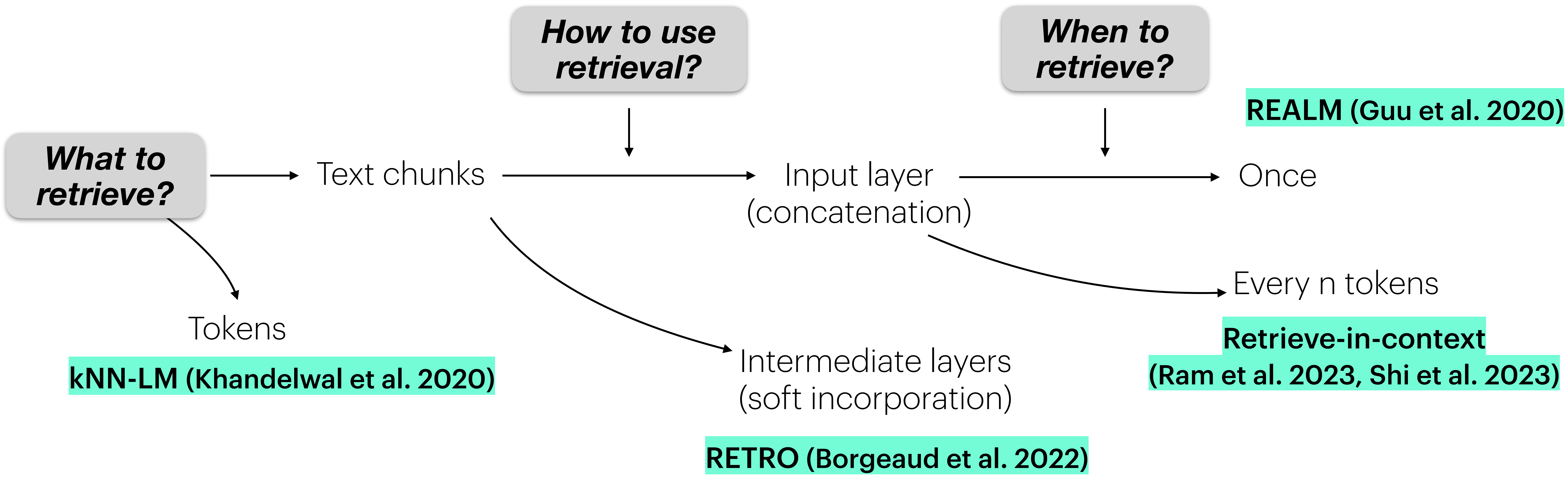

How to use retrieval?
When to retrieve?
REALM (Guu et al. 2020)
What to retrieve?
Text chunks
Input layer (concatenation)
Once
Every n tokens
Retrieve-in-context
(Ram et al. 2023, Shi et al. 2023)

Roadmap



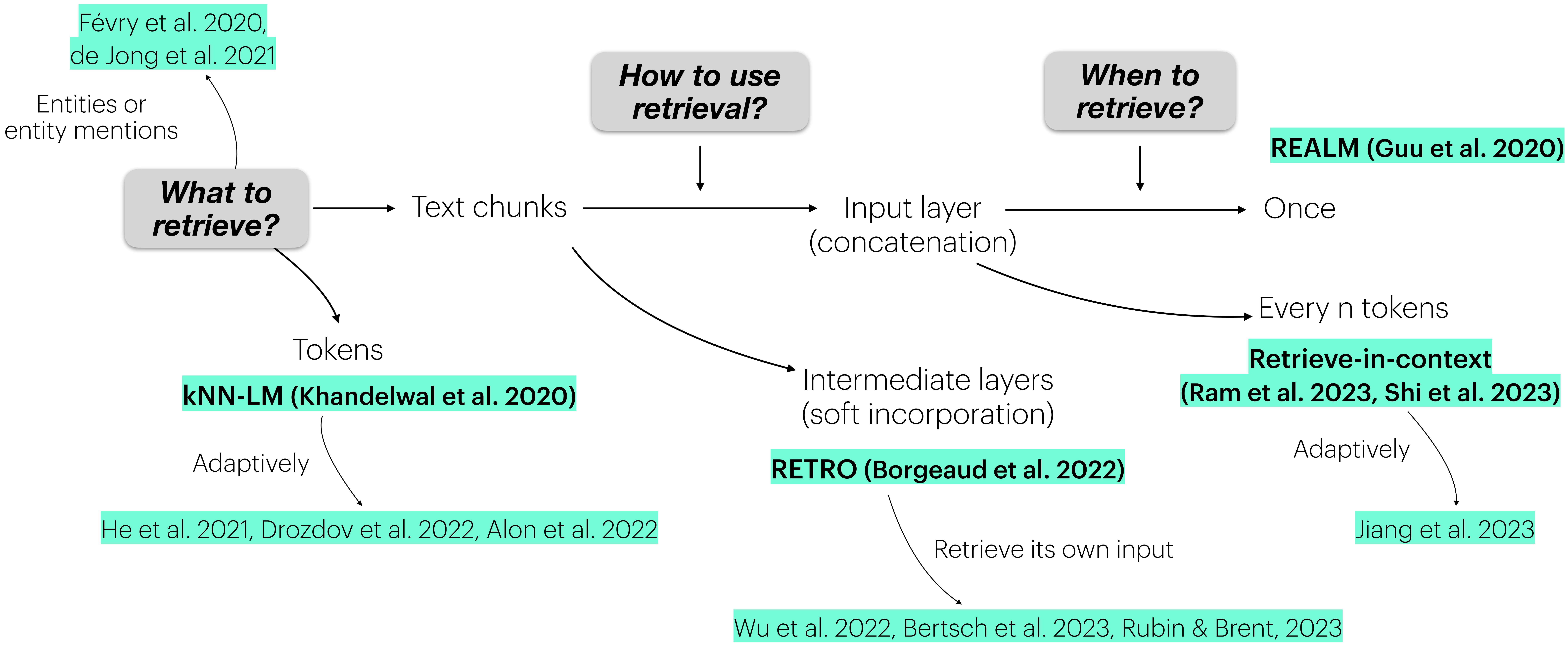
What to retrieve?
Text chunks
How to use retrieval?
Input layer (concatenation)
Intermediate layers (soft incorporation)
RETRO (Borgeaud et al. 2022)
When to retrieve?
REALM (Guu et al. 2020)
Once
Every n tokens
Retrieve-in-context (Ram et al. 2023, Shi et al. 2023)

Roadmap



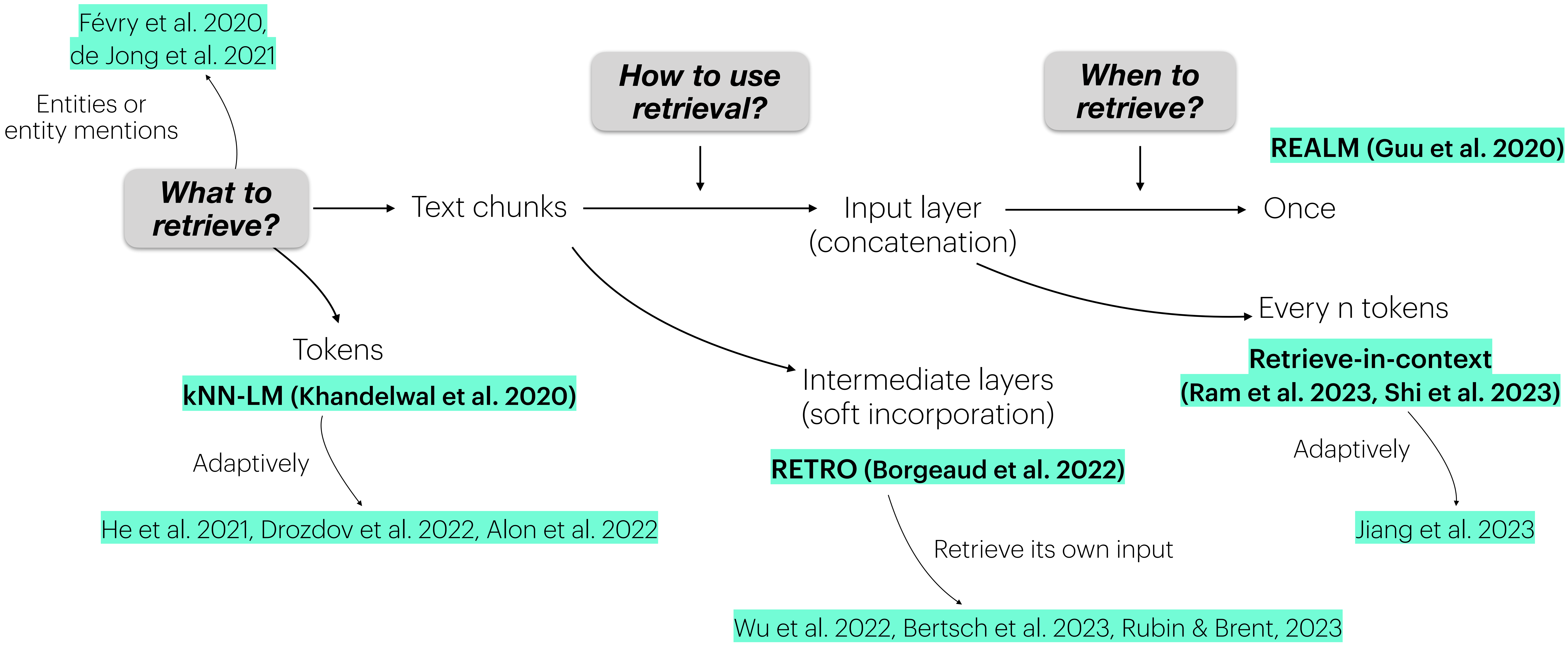
How to use retrieval?
When to retrieve?
REALM (Guu et al. 2020)
What to retrieve?
Text chunks
Input layer (concatenation)
Once
Every n tokens
Retrieve-in-context (Ram et al. 2023, Shi et al. 2023)
Tokens
kNN-LM (Khandelwal et al. 2020)
Intermediate layers (soft incorporation)
RETRO (Borgeaud et al. 2022)

Roadmap



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

Roadmap



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

REALM (Guu et al 2020)

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

REALM (Guu et al 2020)

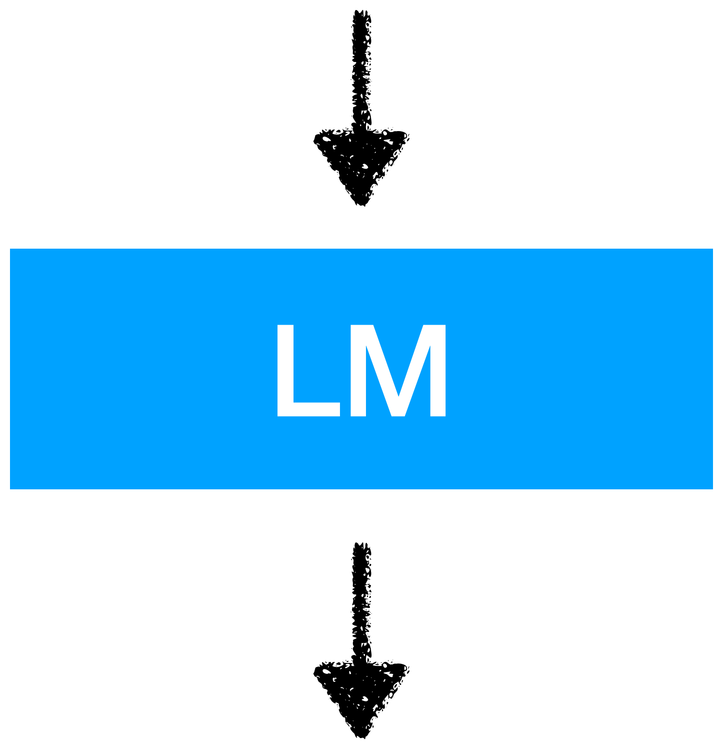
x = World Cup 2022 was the last with 32 teams before the increase to [MASK] in 2026.

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.

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

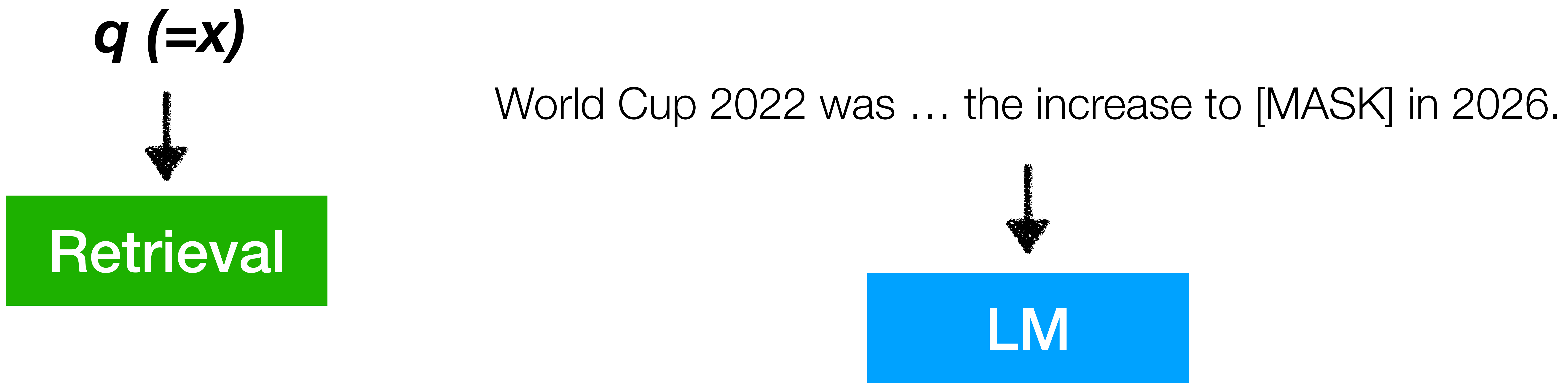


48

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

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"

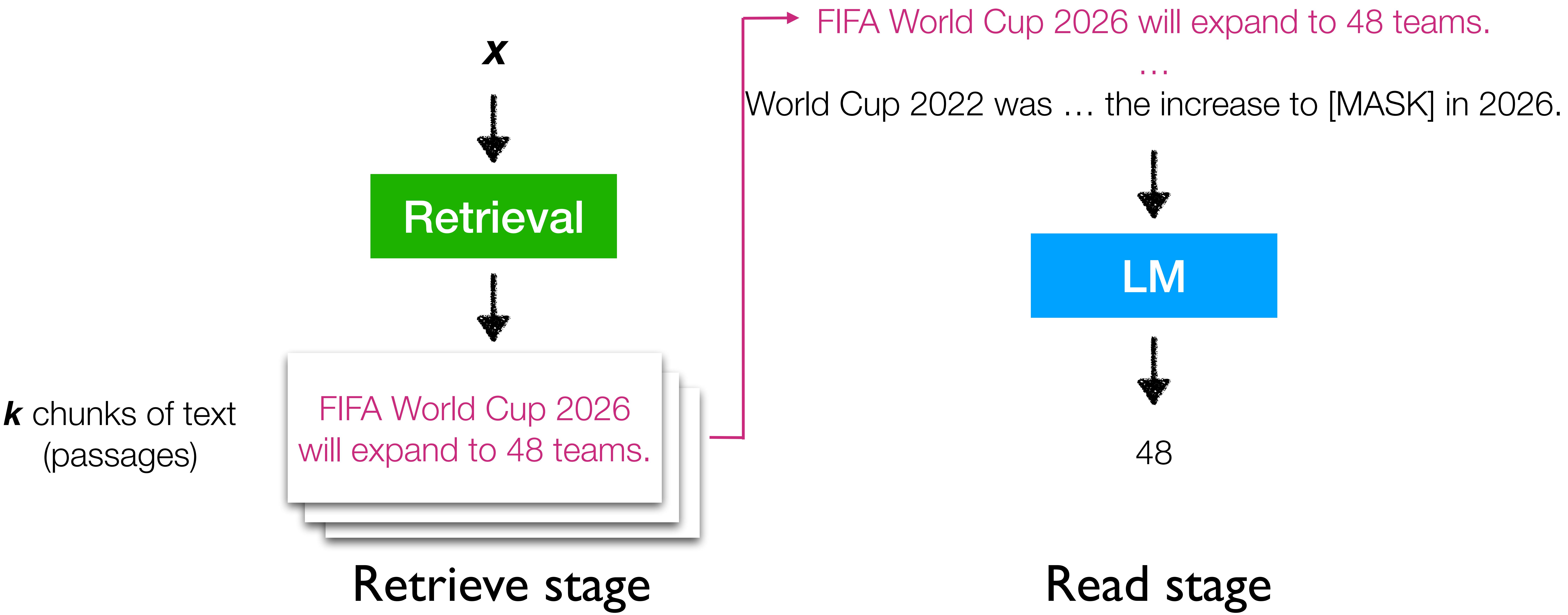
REALM (Guu et al 2020)

$\boldsymbol{x}$ = 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

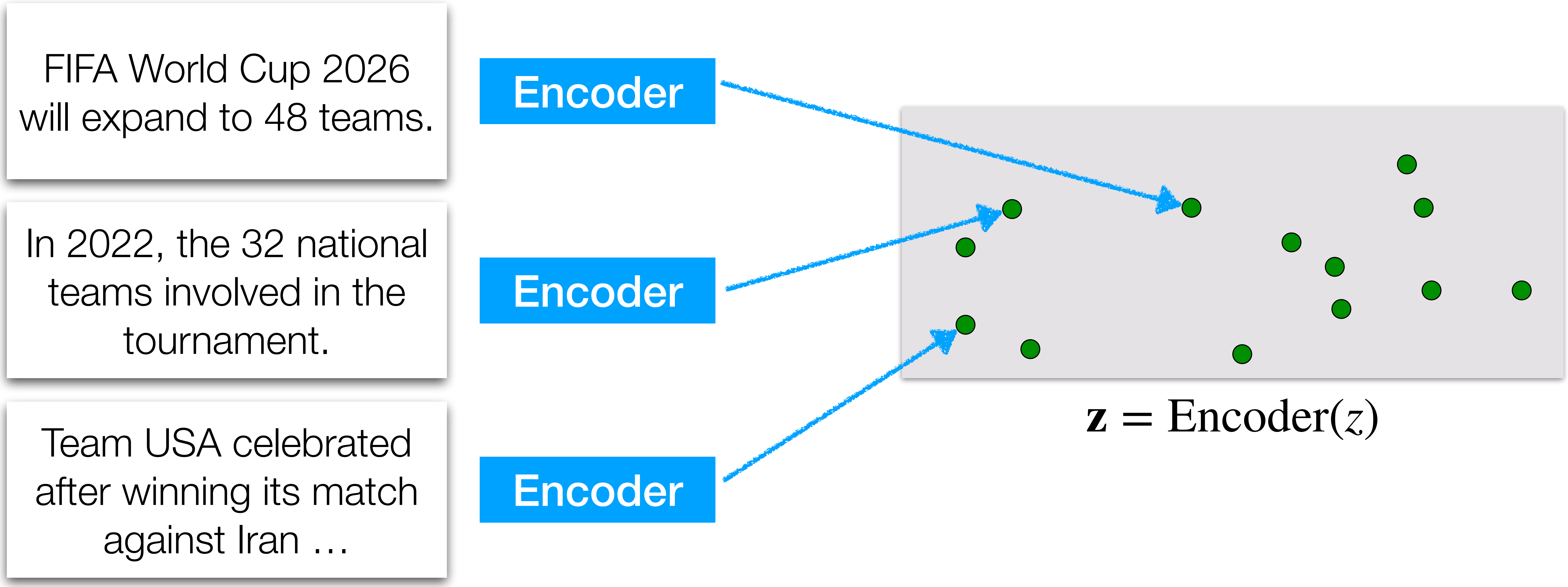
FIFA World Cup 2026 will expand to 48 teams.

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



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

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.

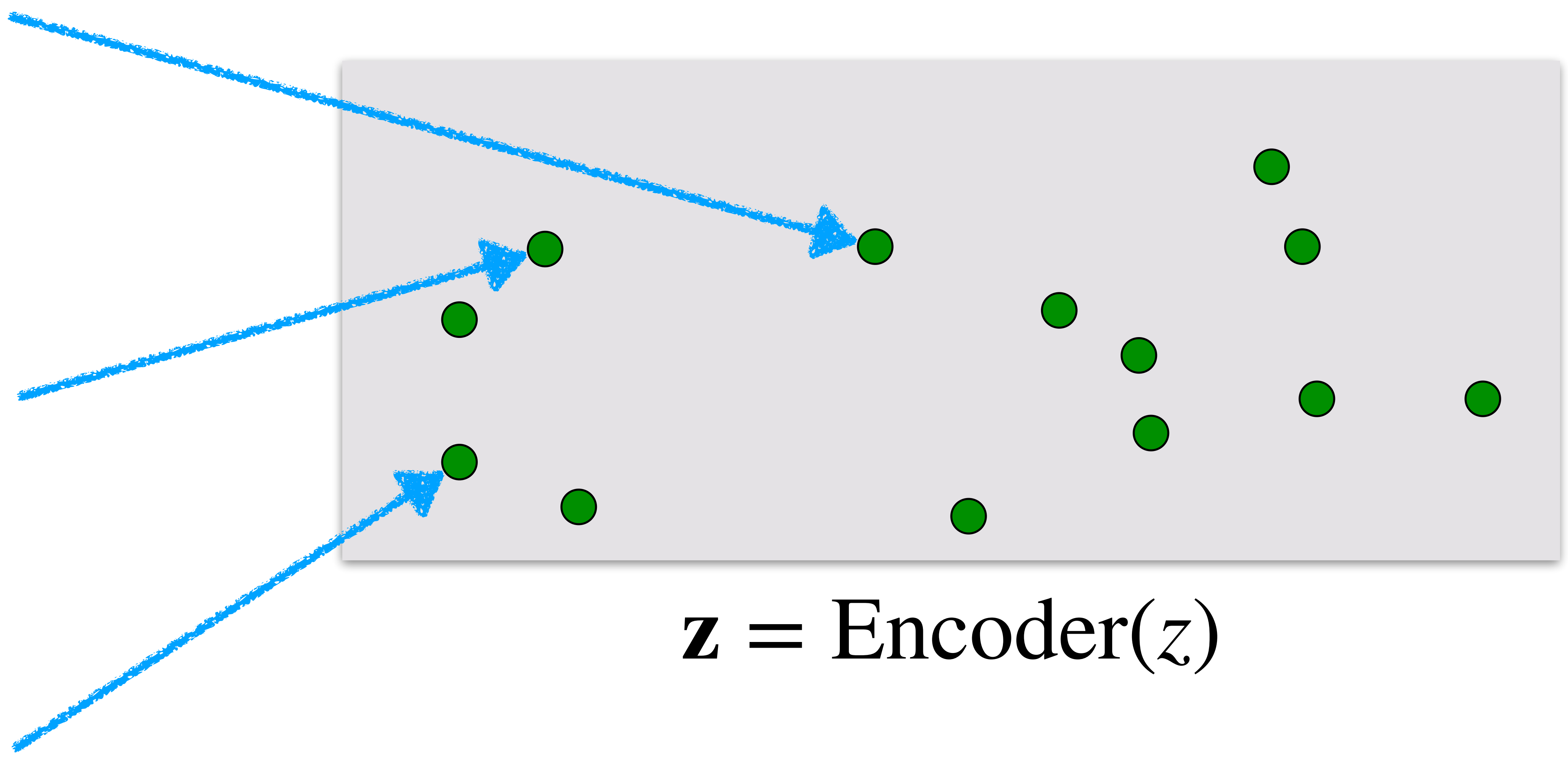
Encoder

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

Encoder

Team USA celebrated after winning its match against Iran …

Encoder



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

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.

Encoder

FIFA World Cup 2026 will expand to 48 teams.

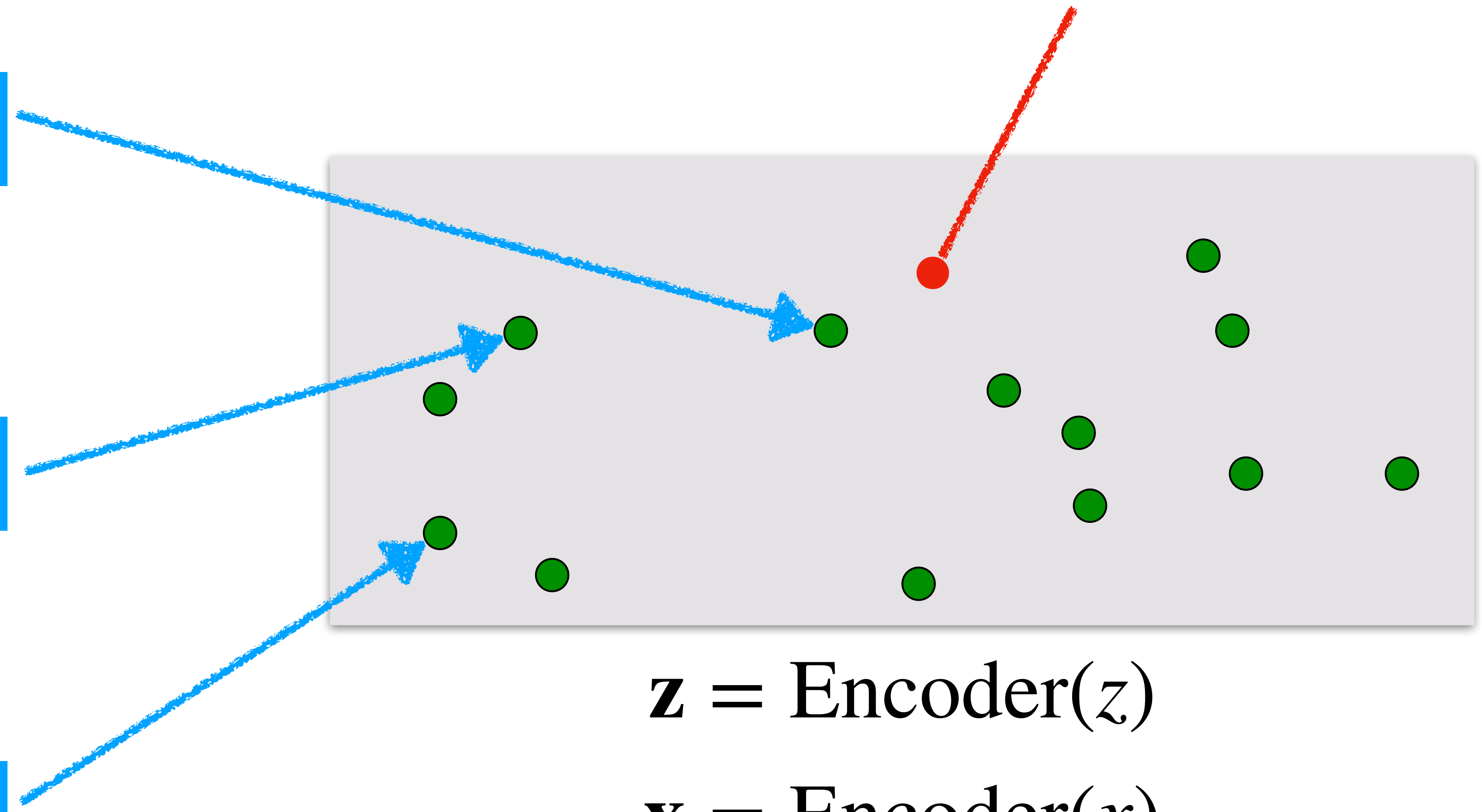
Encoder

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.

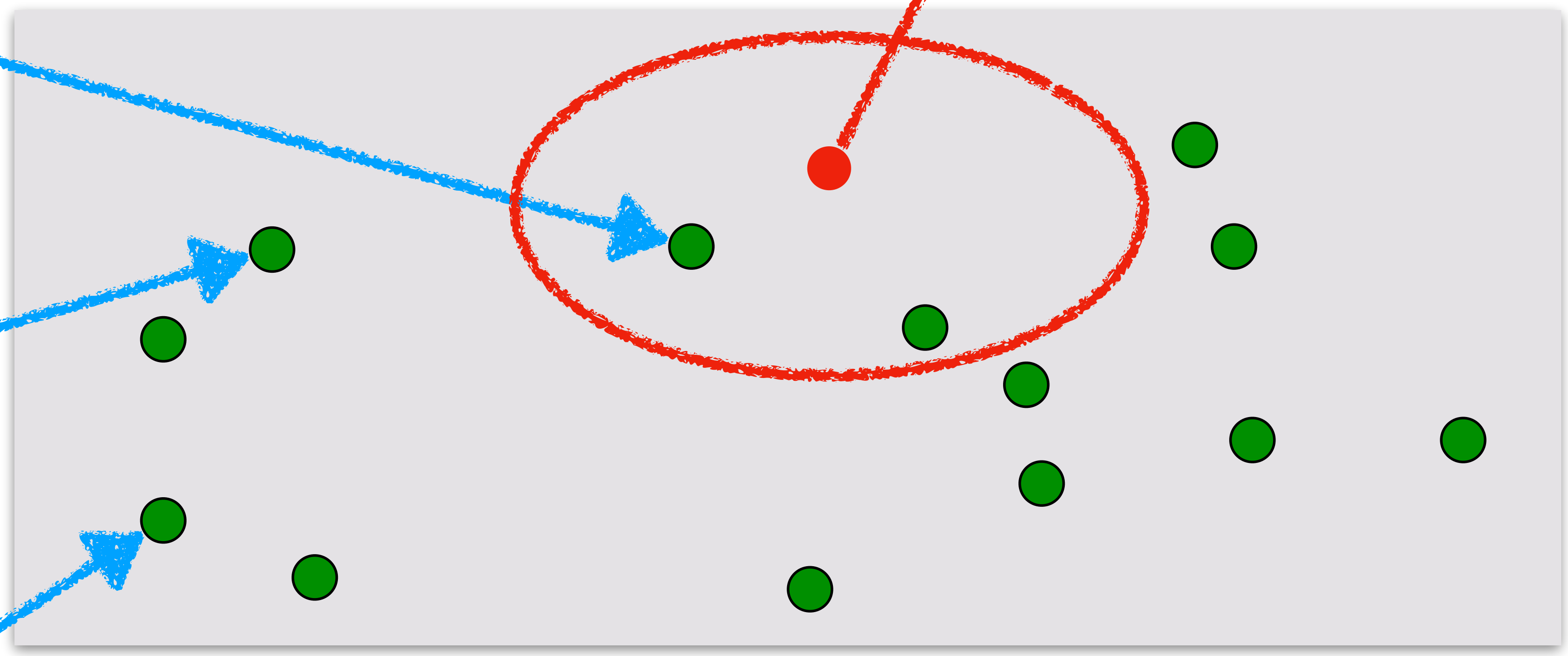
Encoder

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: (I) 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.

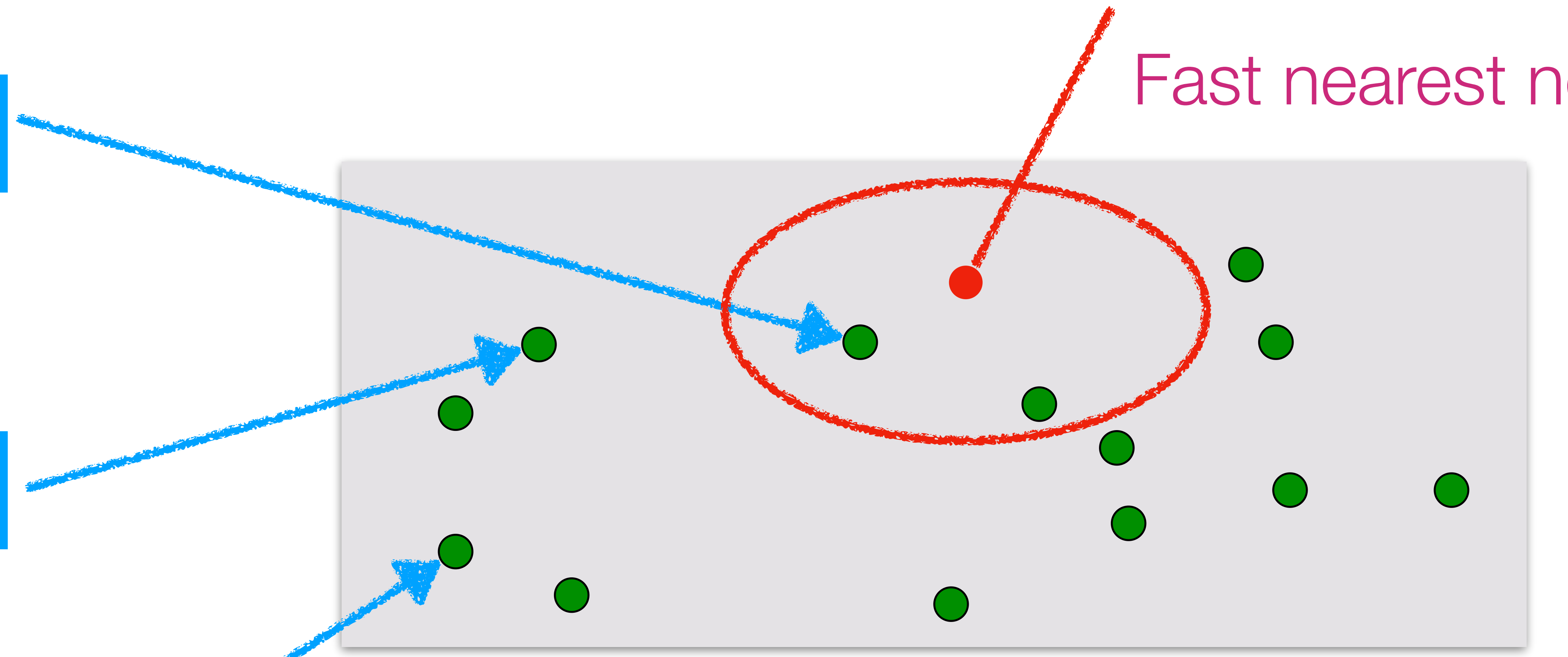
In 2022, the 32 national teams involved in the tournament.

Team USA celebrated after winning its match against Iran …

Encoder

Encoder

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

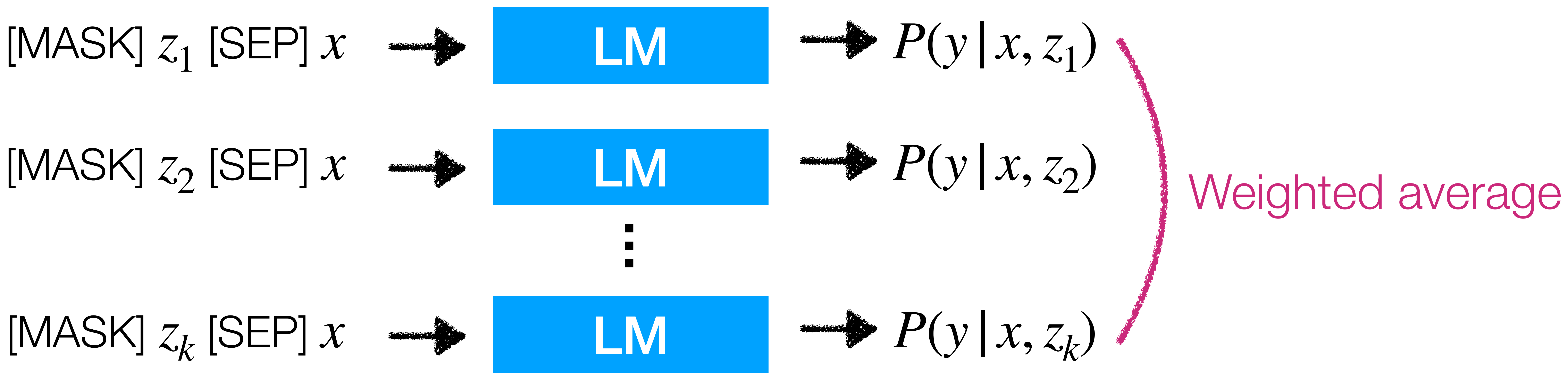
[MASK] z_1 [SEP] x → LM → $P(y \mid x, z_1)$

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

⋮

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

REALM: (2) Read stage



REALM: (2) Read stage

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

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

⋮

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

Weighted average

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

REALM: (2) Read stage

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

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

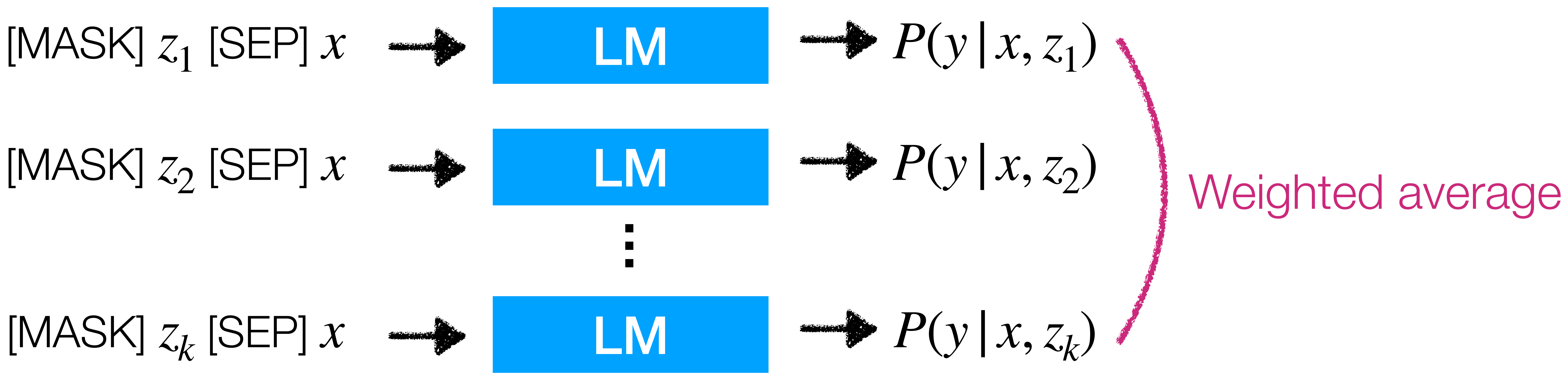
⋮

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

Weighted average

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

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

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

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

⋮

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

Weighted average

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

Need to approximate
→ Consider top *k* chunks only

REALM: (2) Read stage

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

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

⋮

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

Weighted average

0 if not one of top k

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

Need to approximate
→ Consider top k chunks only

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)

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

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

$\boldsymbol{x}$ = 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

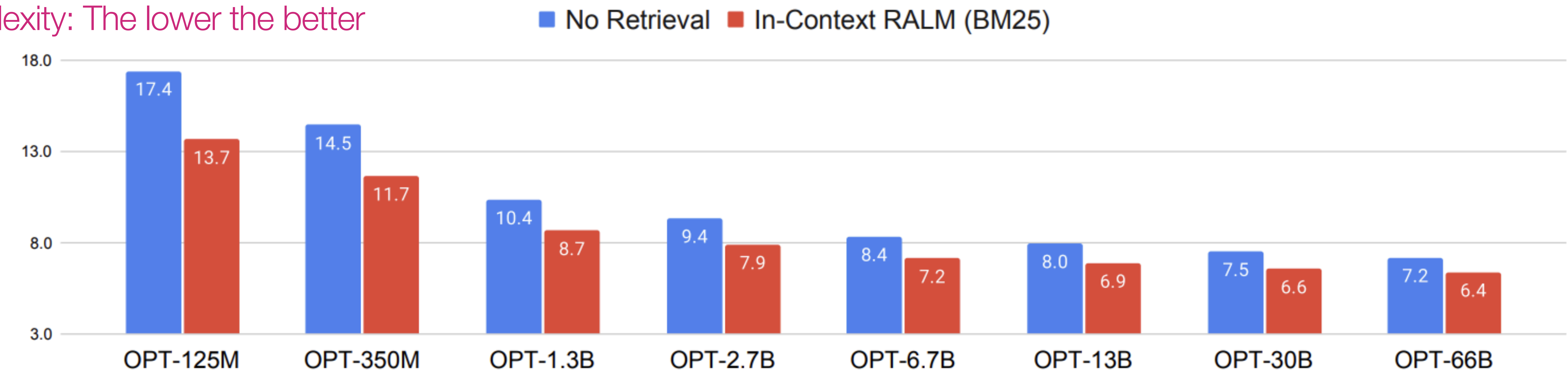
LM

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

Perplexity: The lower the better



Varying sizes of LMs

Retrieval helps over all sizes of LMs

Graphs from Ram et al. 2023

Retrieval-in-context LM

Is $\boldsymbol{q}$=$\boldsymbol{x}$ 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 $\boldsymbol{q}=\boldsymbol{x}$ 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 $\boldsymbol{q}=\boldsymbol{x}$ 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 $\boldsymbol{q}$=$\boldsymbol{x}$ necessary?

$\boldsymbol{x}$ = 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

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

Retrieval

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.

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.

Retrieval-in-context LM

Is $\boldsymbol{q}=\boldsymbol{x}$ necessary?

$\boldsymbol{x}$ = 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

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

Retrieval

Retrieval

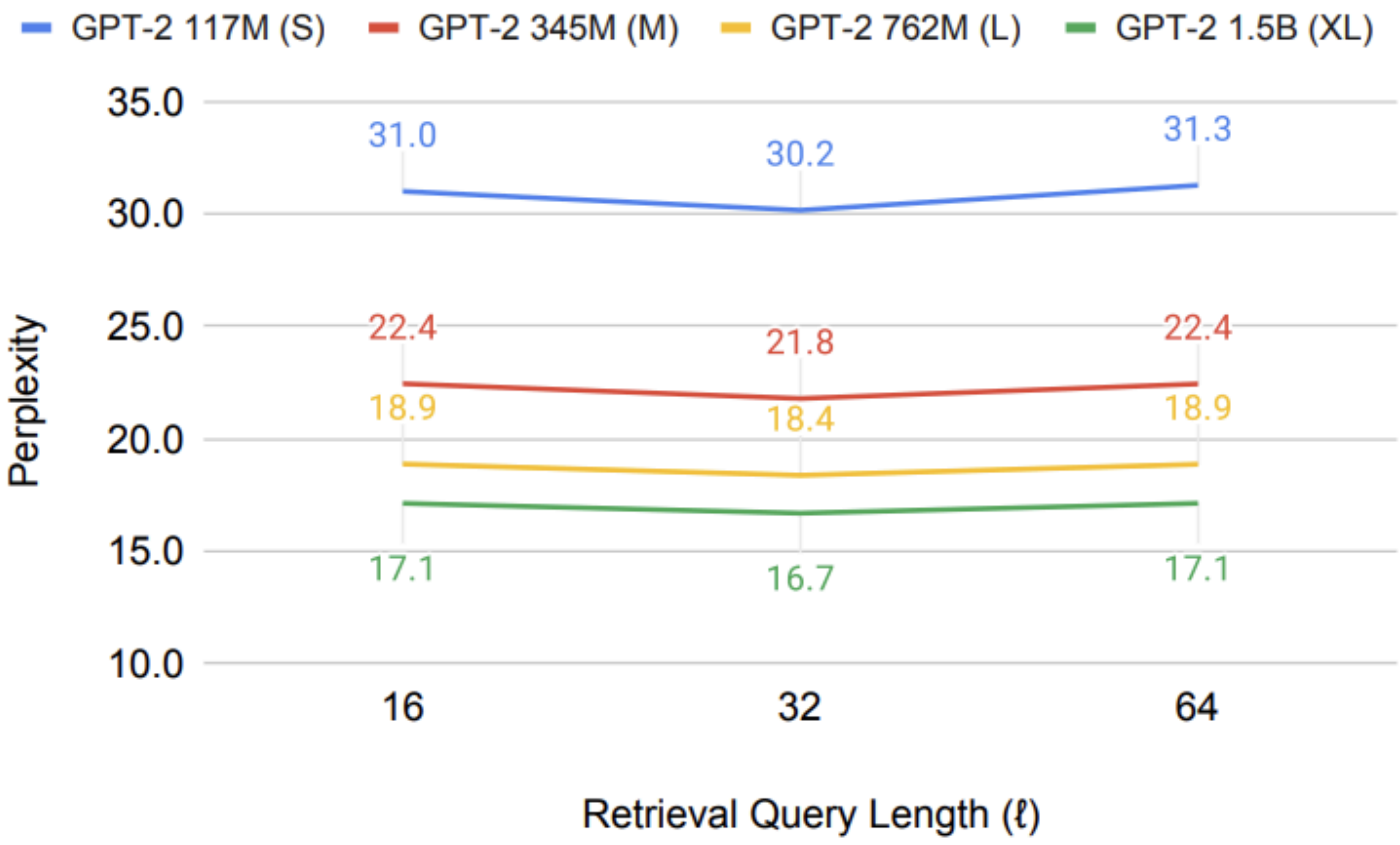
The U.S. national team defeated Iran 1-0.

Does not cover "tokens that will come next"

FIFA World Cup 2026 will expand to 48 teams.

more relevant to what will come next

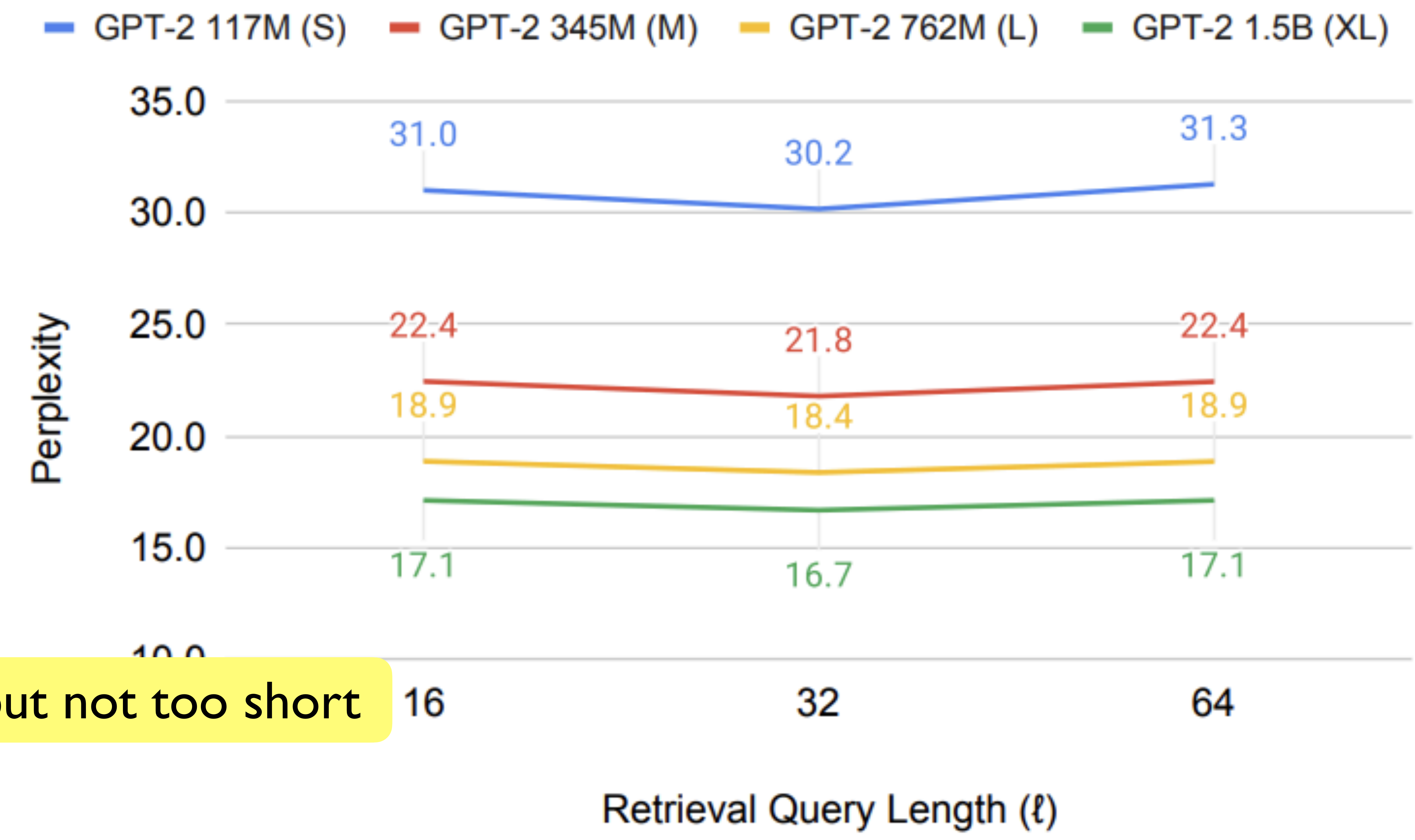
Retrieval-in-context LM



Shorter prefix (more recent tokens) as a query helps

Graphs from Ram et al. 2023

Retrieval-in-context LM



but not too short

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

↓

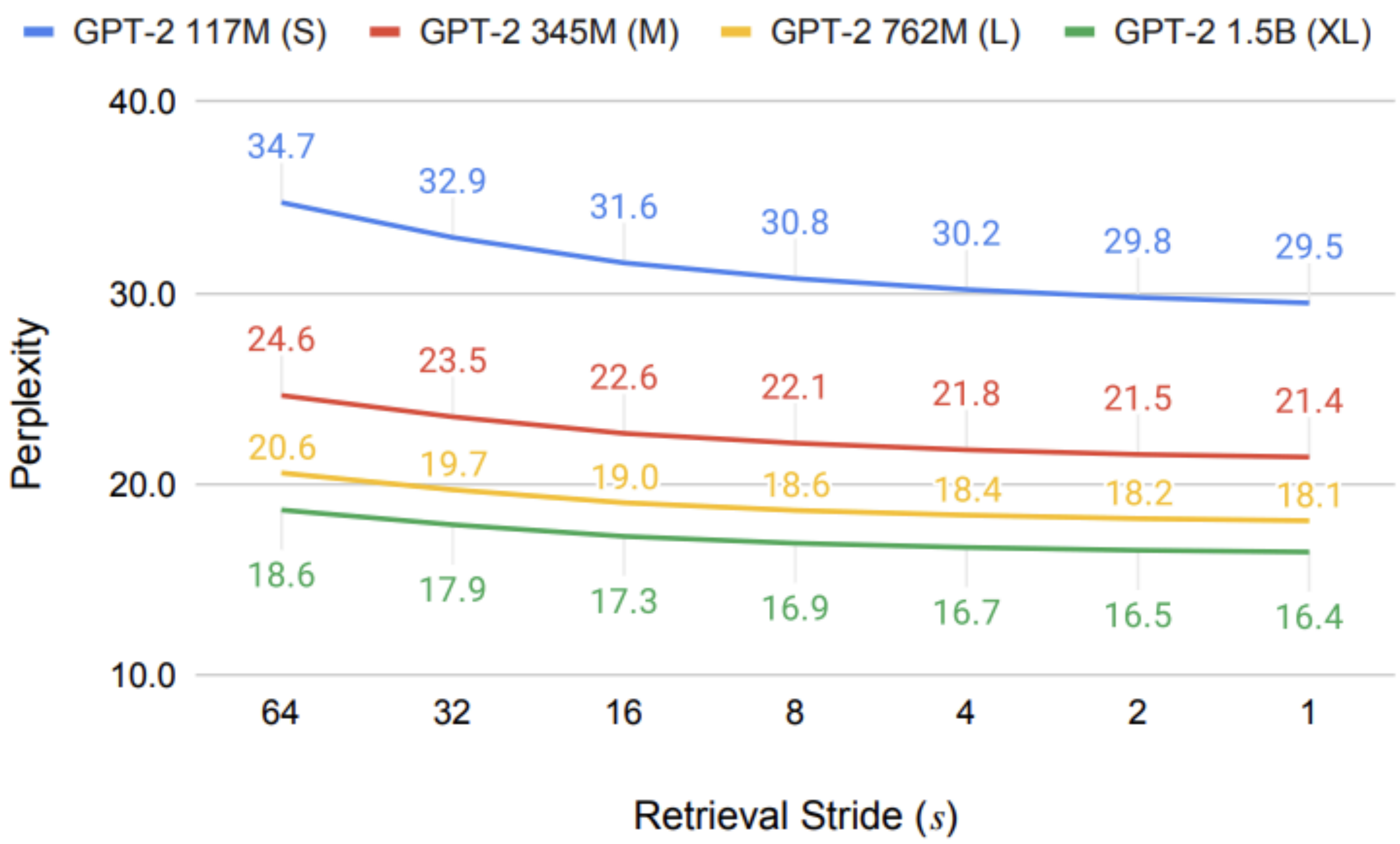
LM

↓

to 48 in the 2026 tournament.

Retrieval results from a new query explain them!

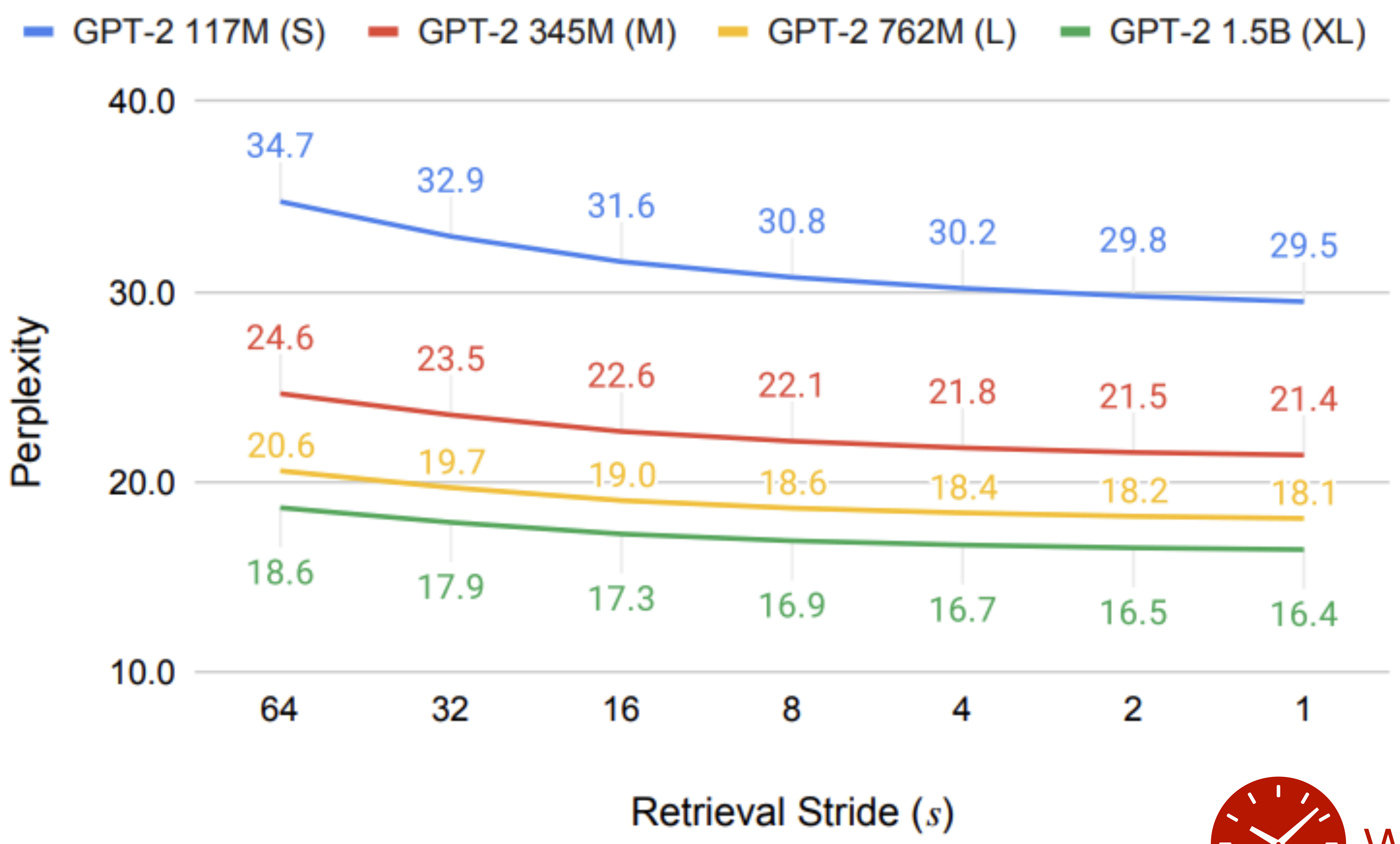
Retrieval-in-context LM



Retrieving more frequently helps

Graphs from Ram et al. 2023

Retrieval-in-context LM



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

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 <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)

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

RETRO (Borgeaud et al. 2021)

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

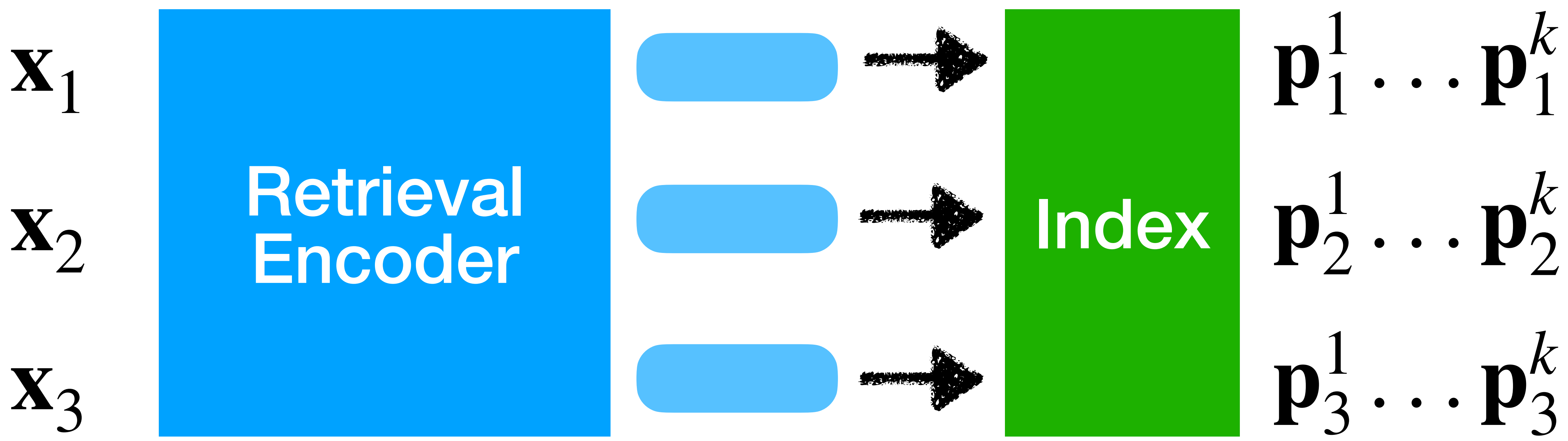
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$

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)

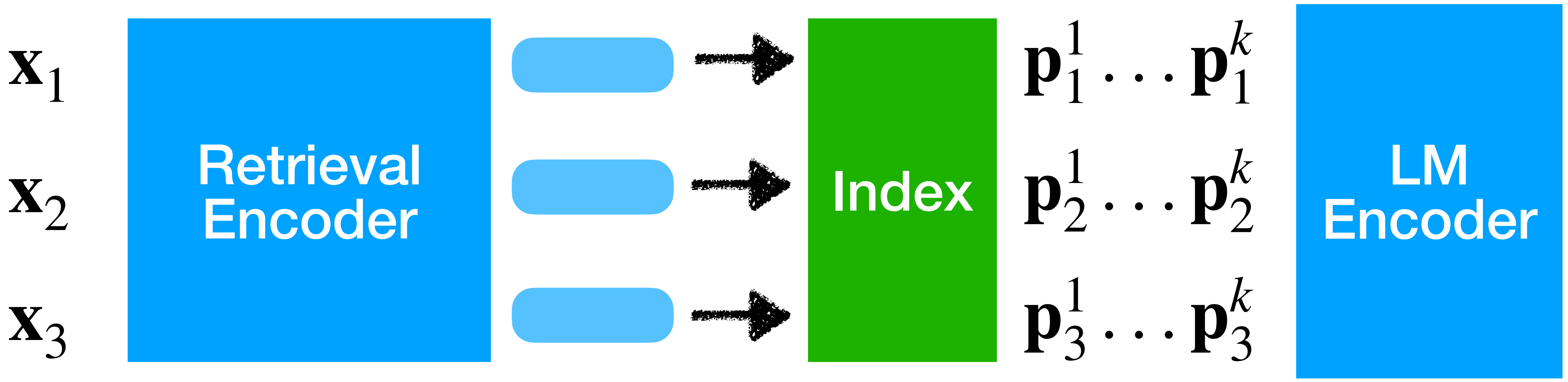


RETRO (Borgeaud et al. 2021)

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

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

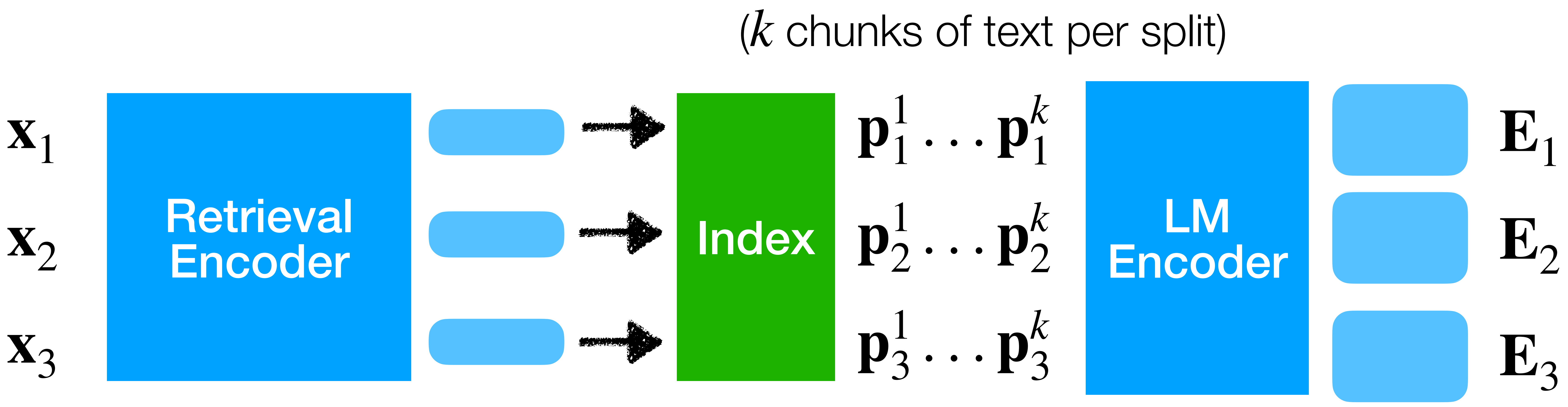
(k chunks of text per split)



RETRO (Borgeaud et al. 2021)

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

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

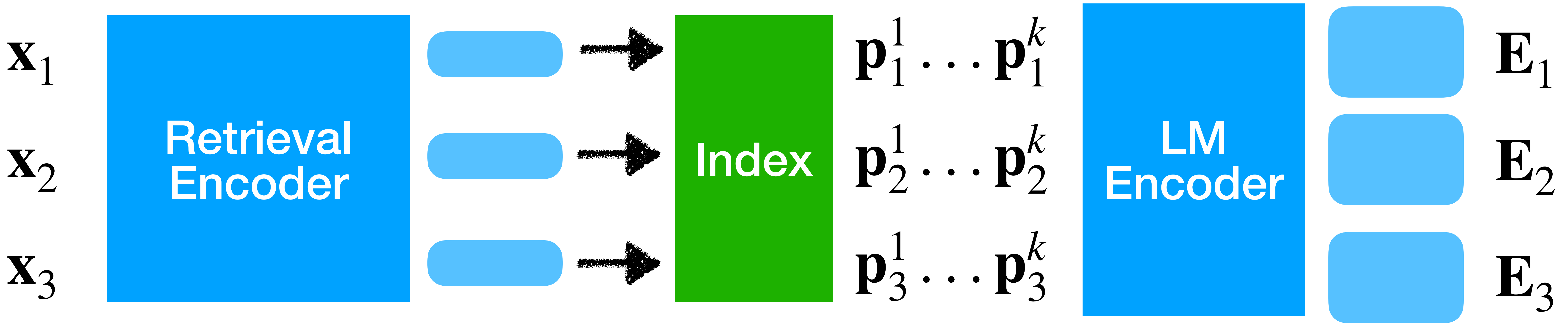


RETRO (Borgeaud et al. 2021)

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

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

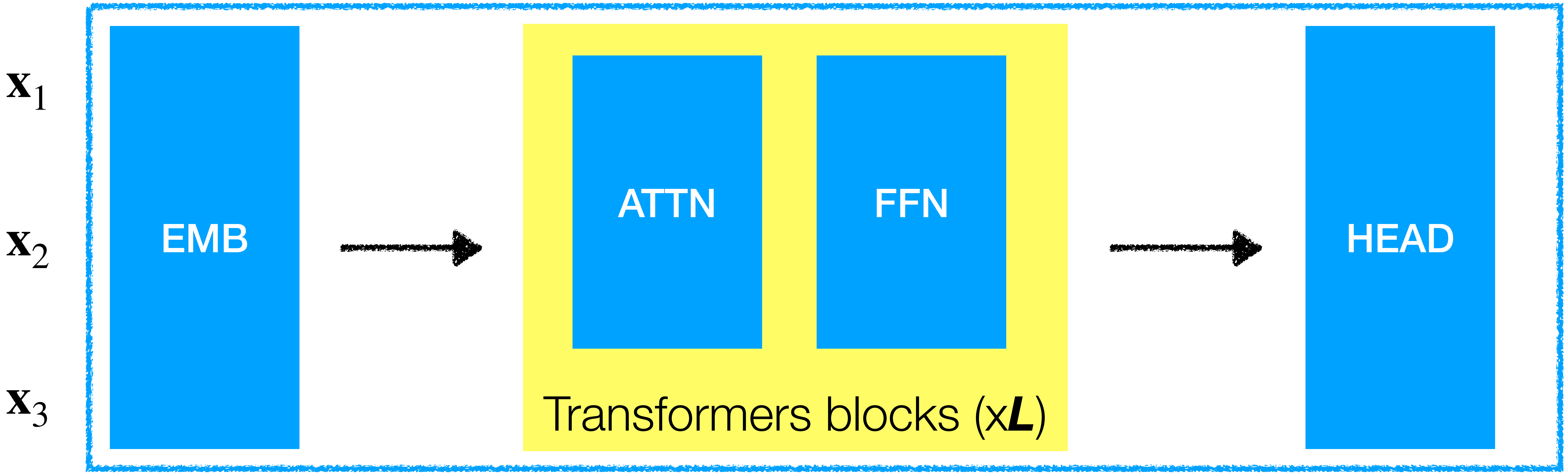
(k chunks of text per split)



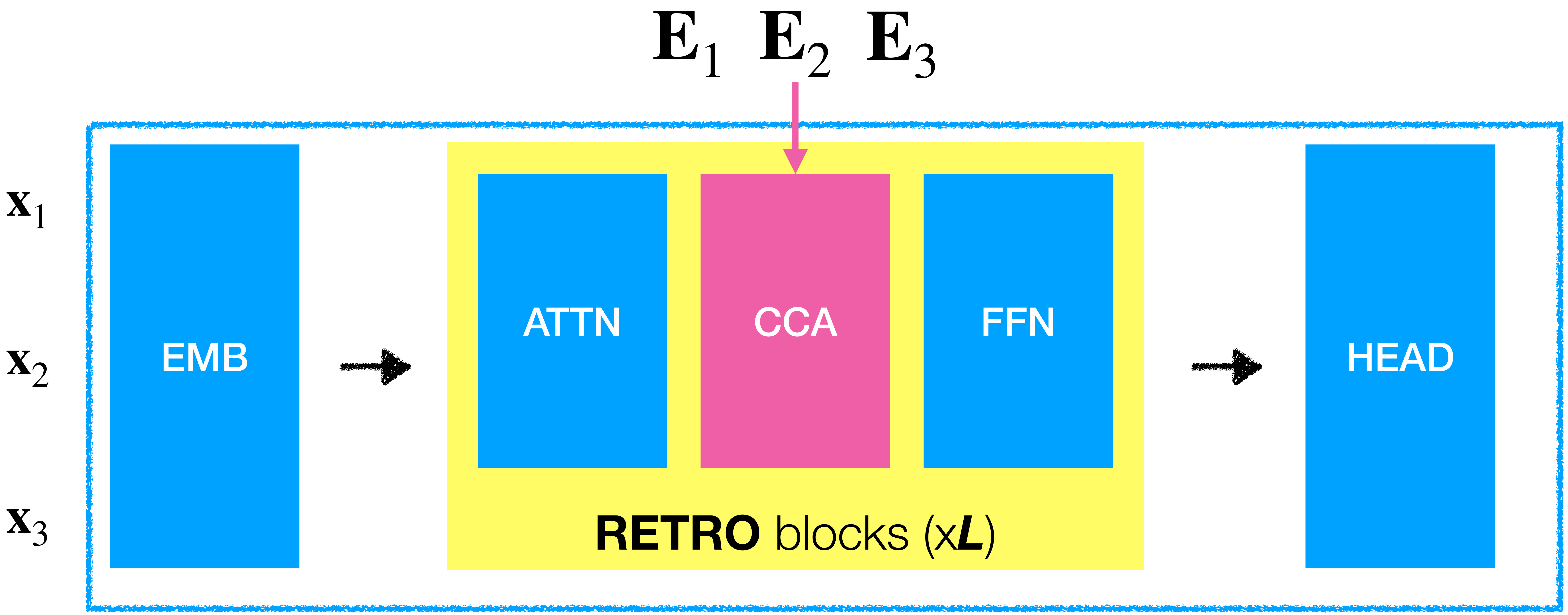
(A $r \times k \times d$ matrix)

(r = # tokens per text chunk)
(d = hidden dimension)
(k = # retrieved chunks per split)

Regular decoder

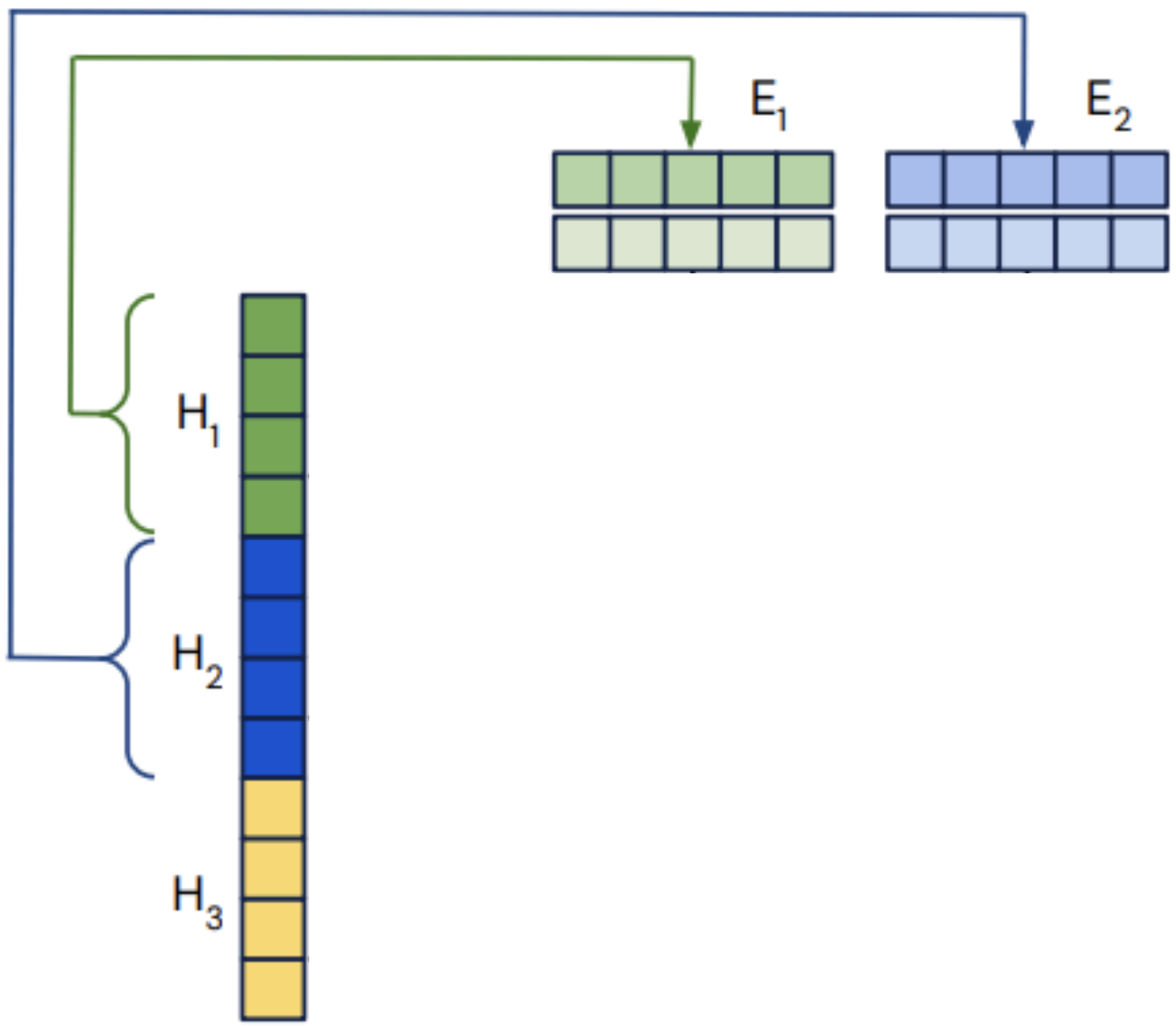


Decoder in RETRO



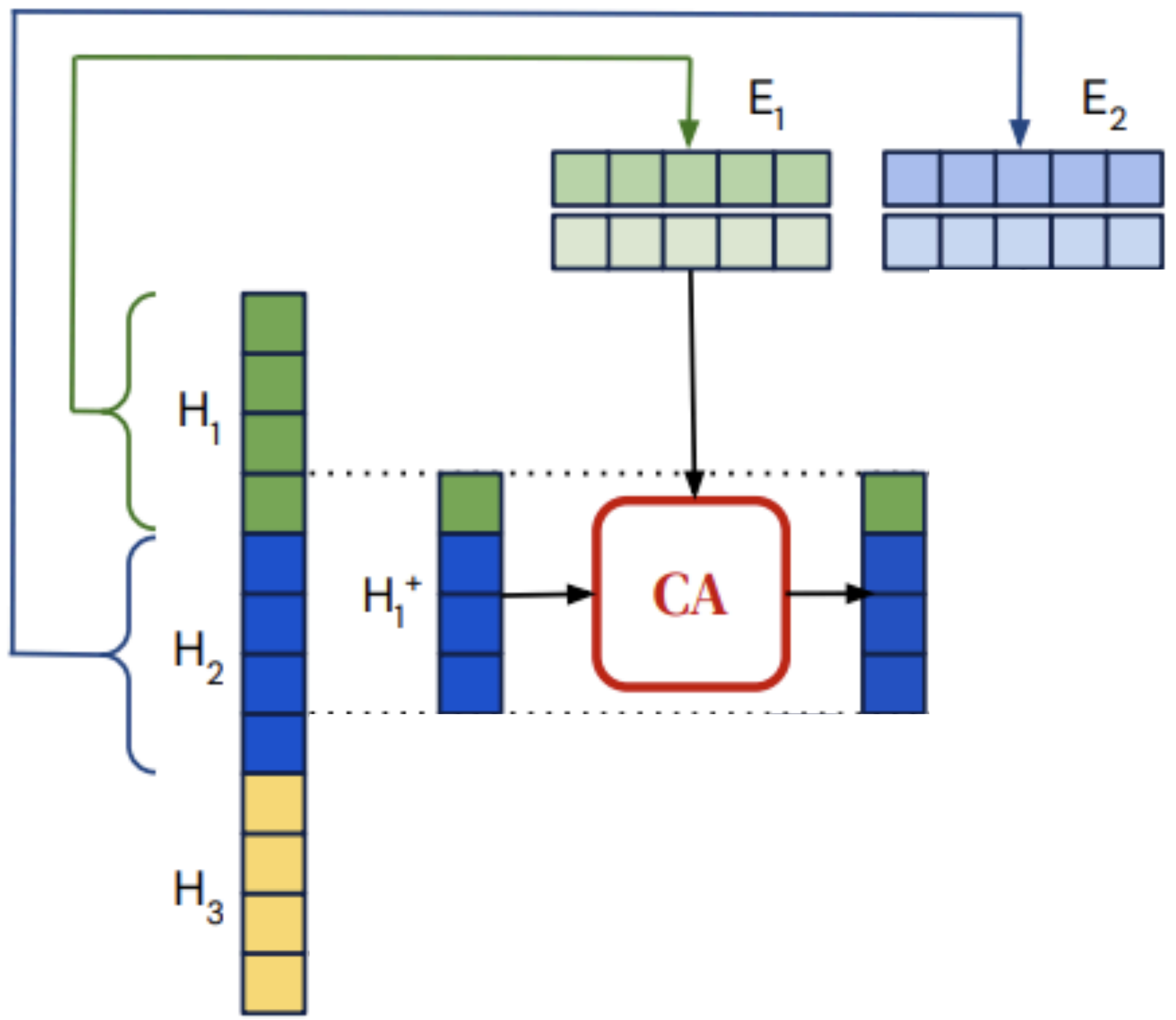
Chunked Cross Attention (CCA)

Chunked Cross Attention



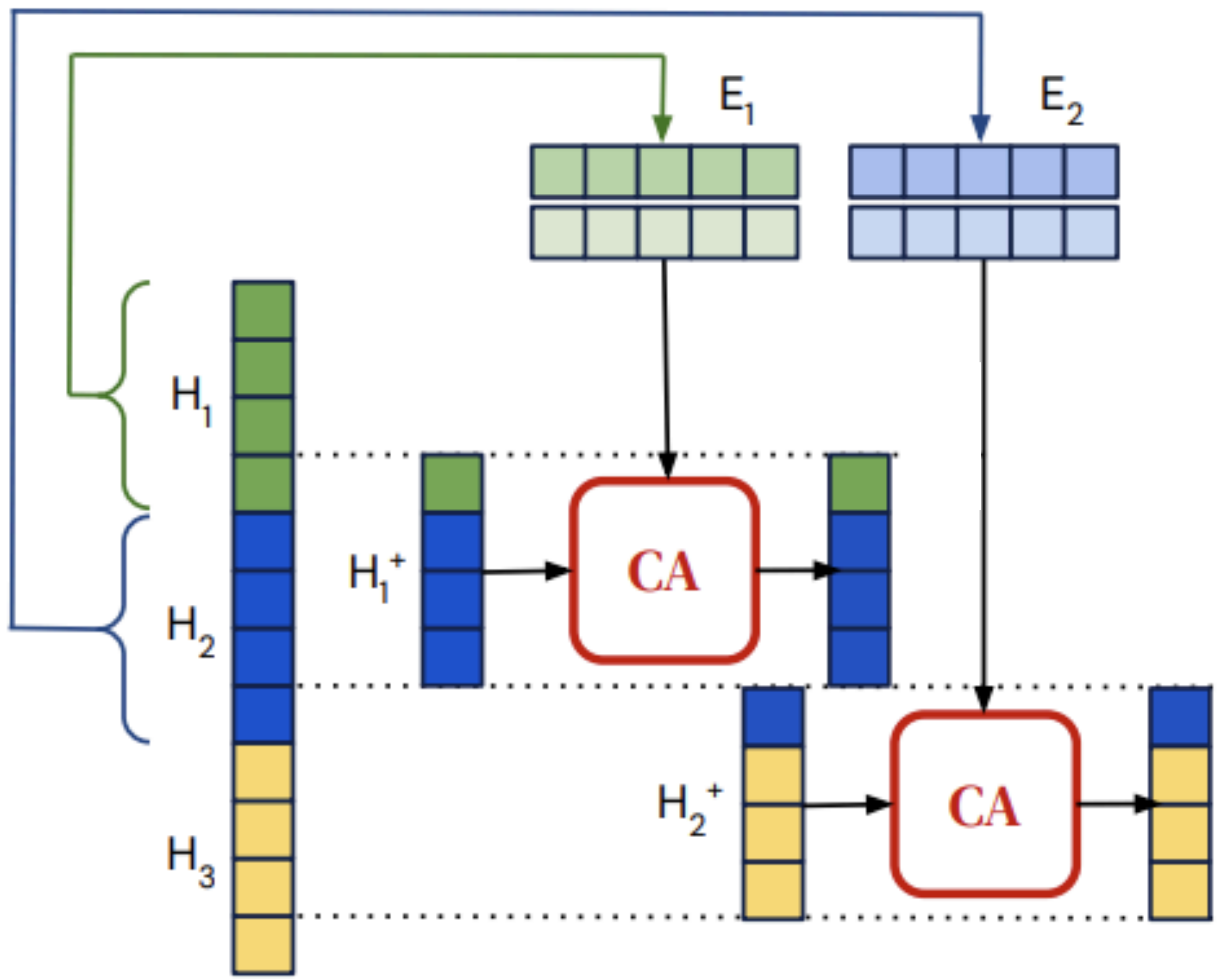
Outputs from the previous layer H

Chunked Cross Attention



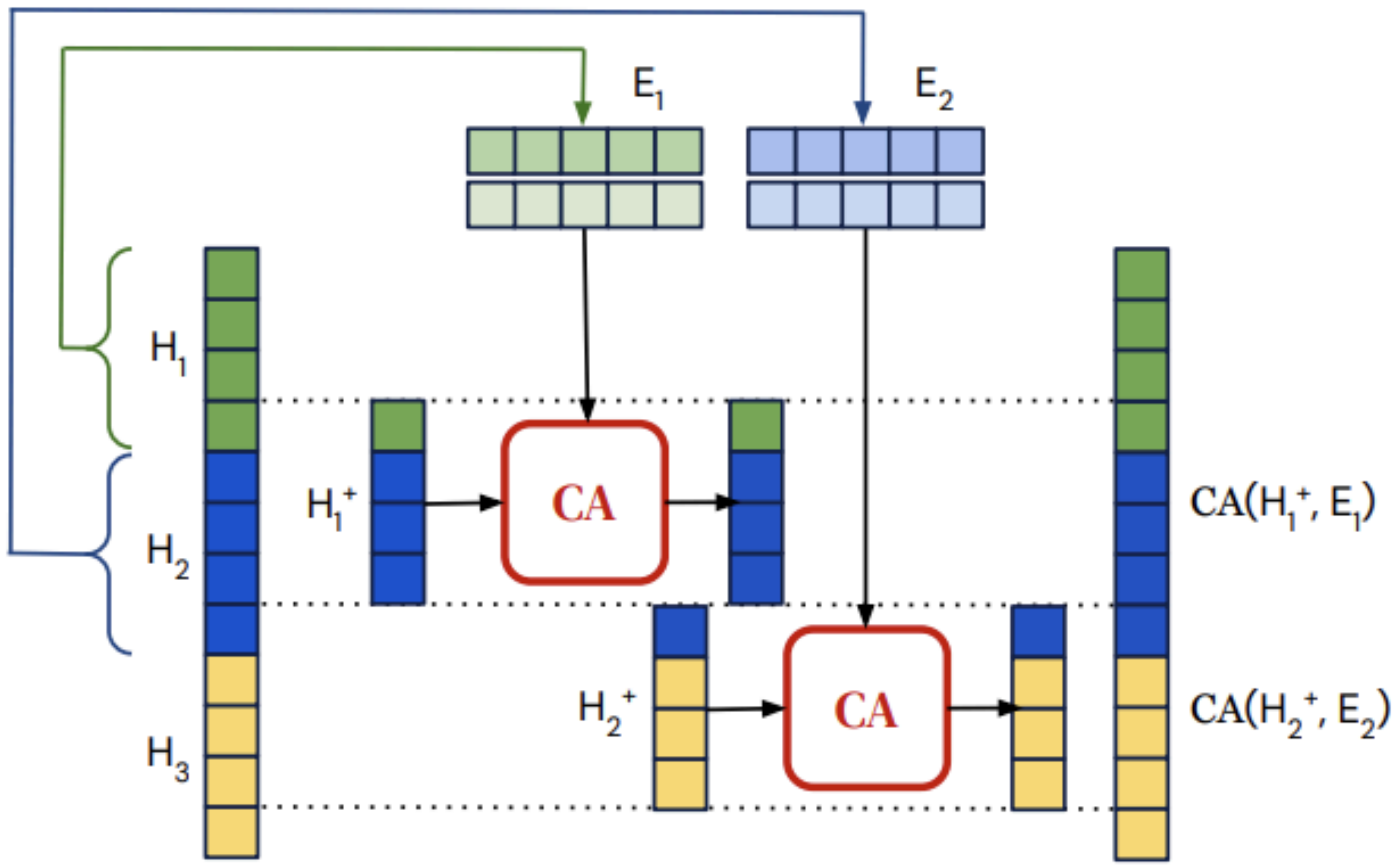
Outputs from the previous layer H

Chunked Cross Attention



Outputs from the previous layer H

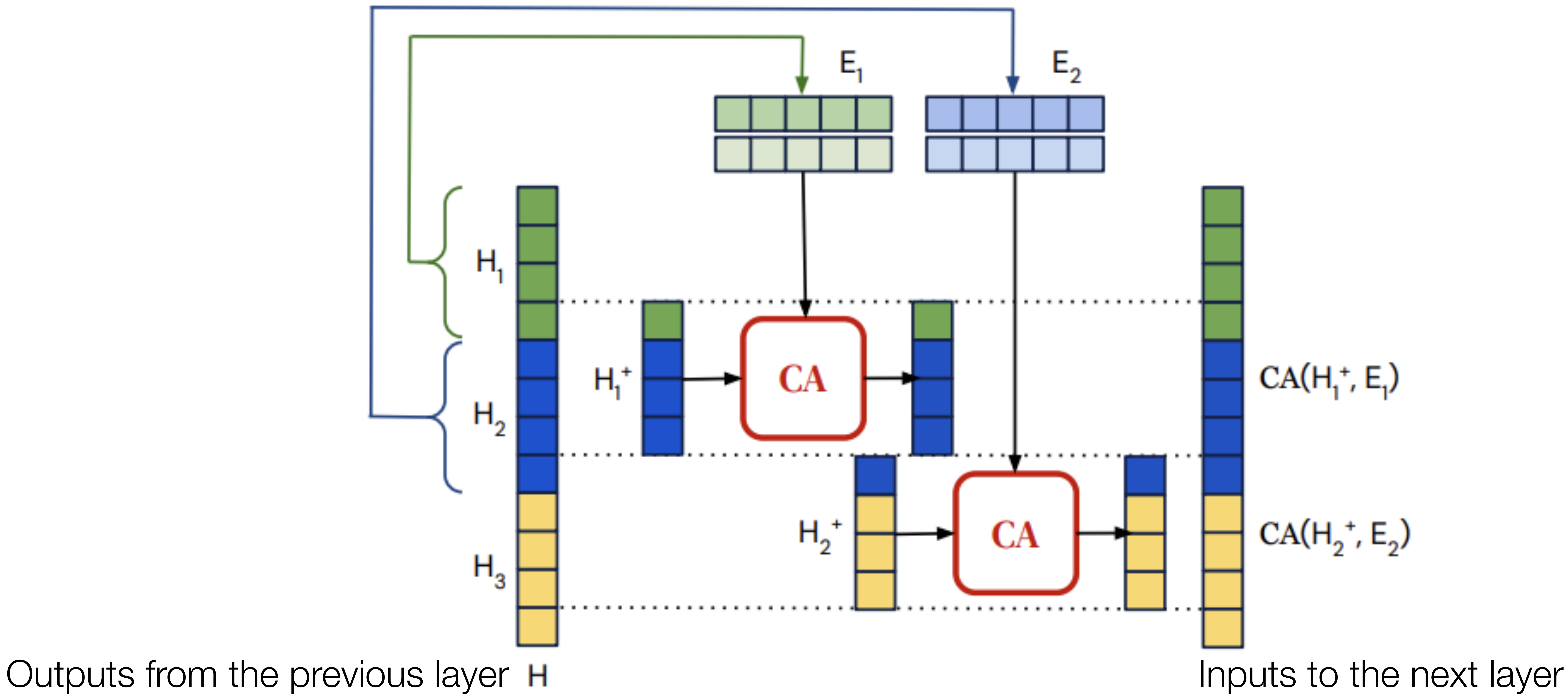
Chunked Cross Attention



Outputs from the previous layer H

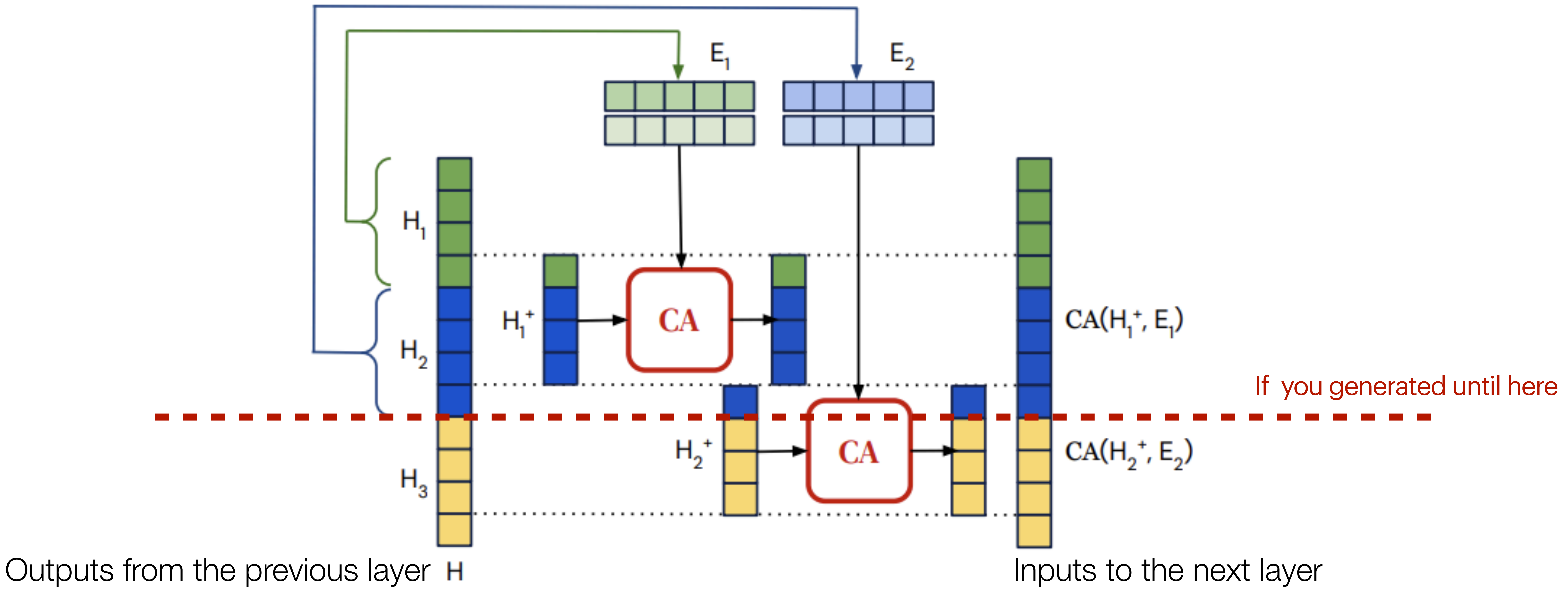
Inputs to the next layer

Chunked Cross Attention



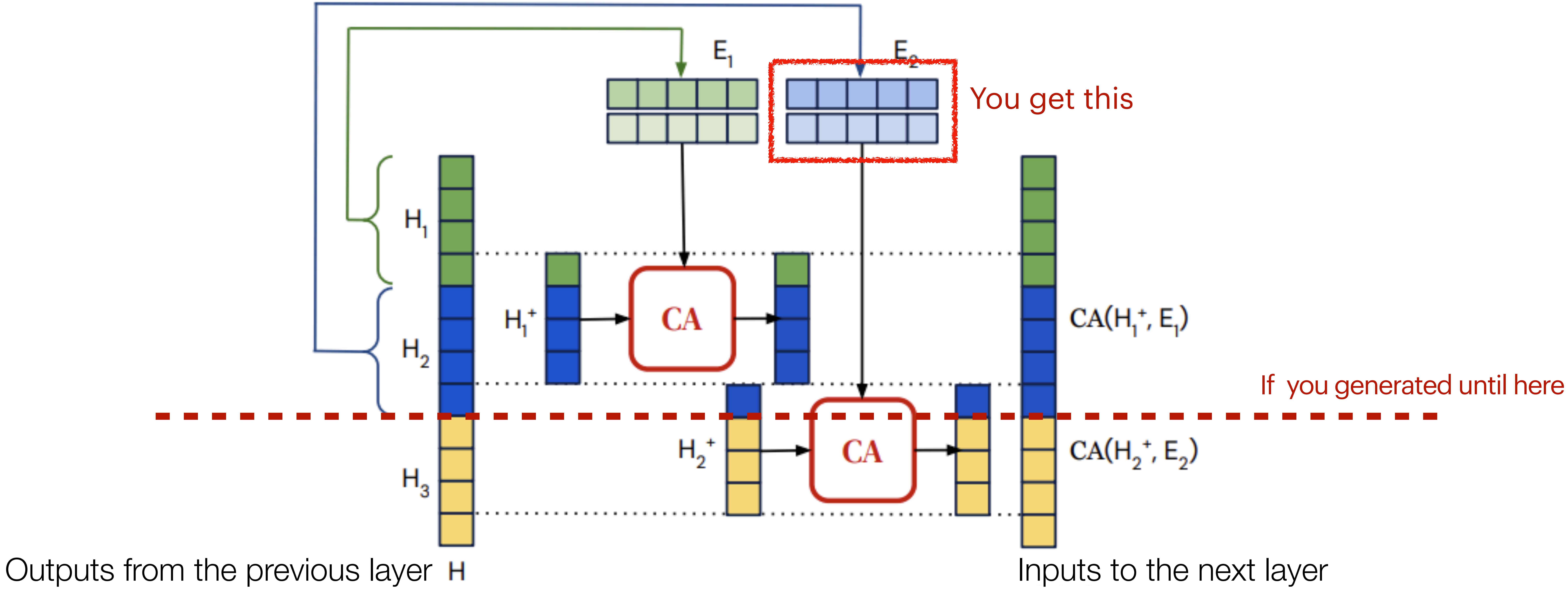
✔ Cross-attention can be computed ***in parallel***

Chunked Cross Attention



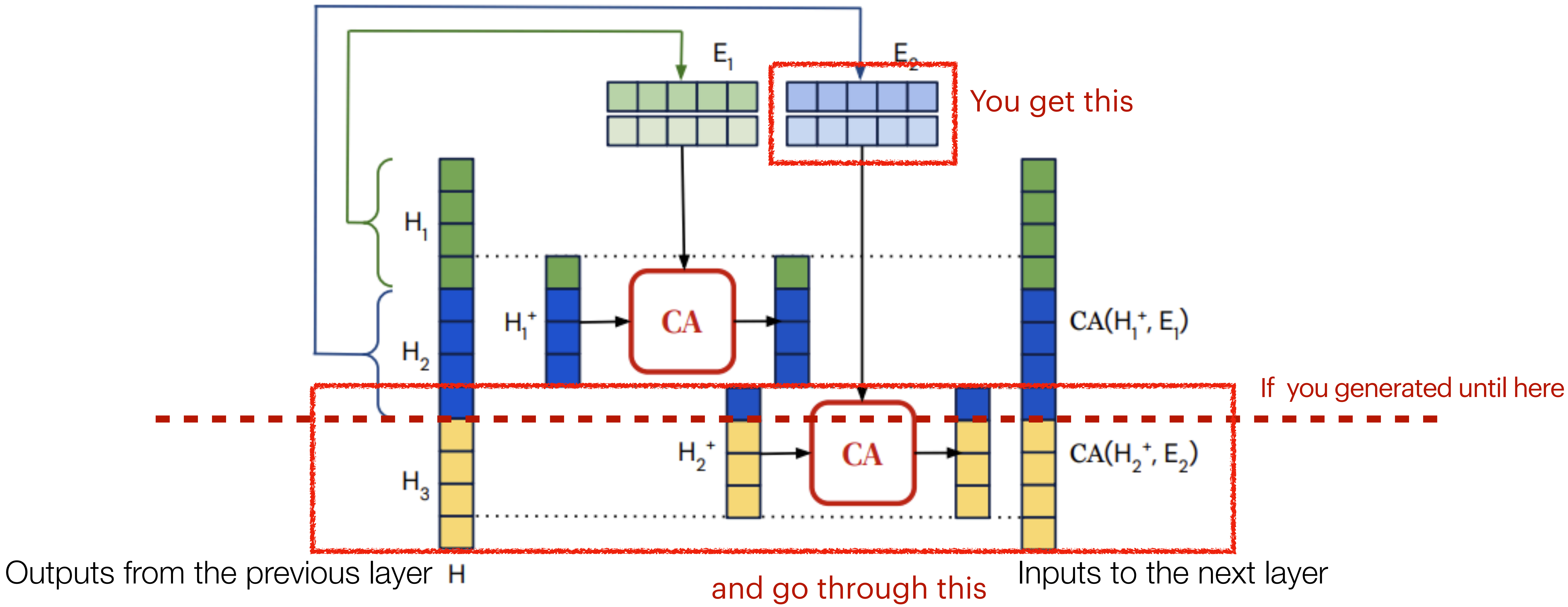
✔ Cross-attention can be computed ***in parallel***

Chunked Cross Attention



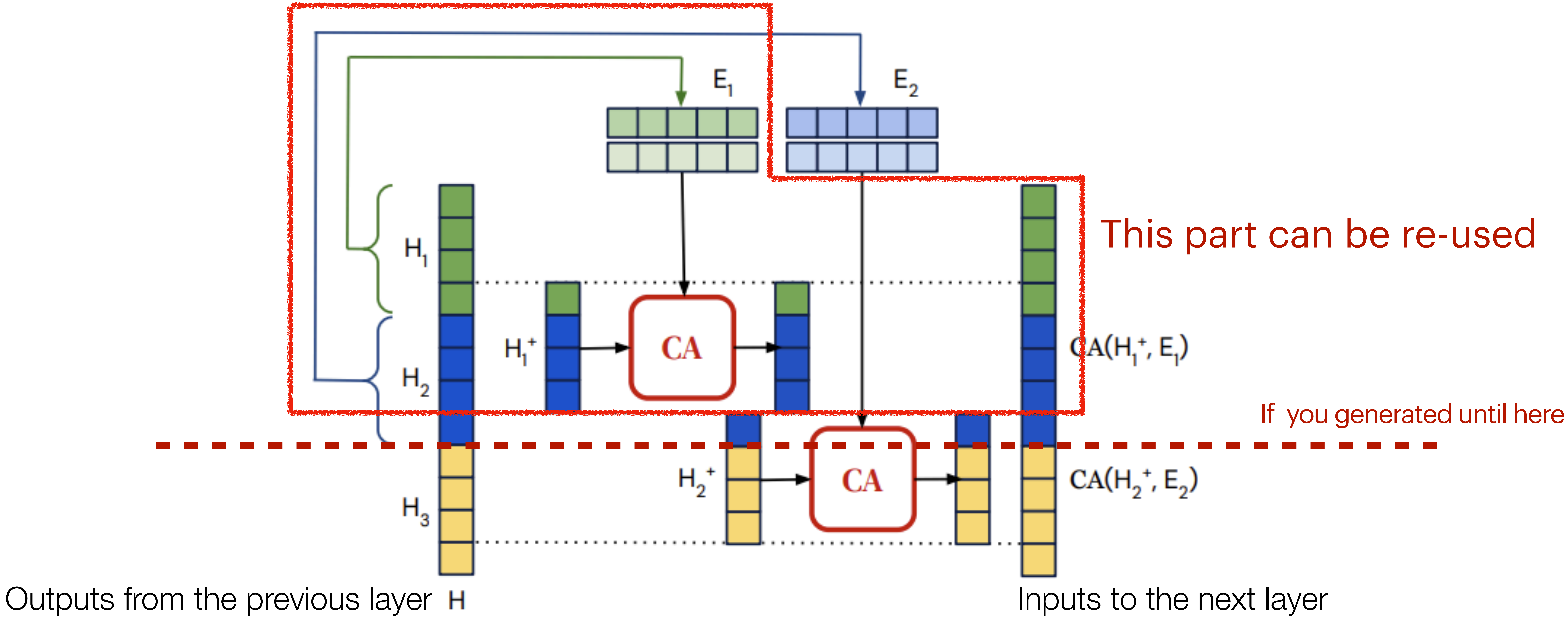
✔ Cross-attention can be computed ***in parallel***

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
*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

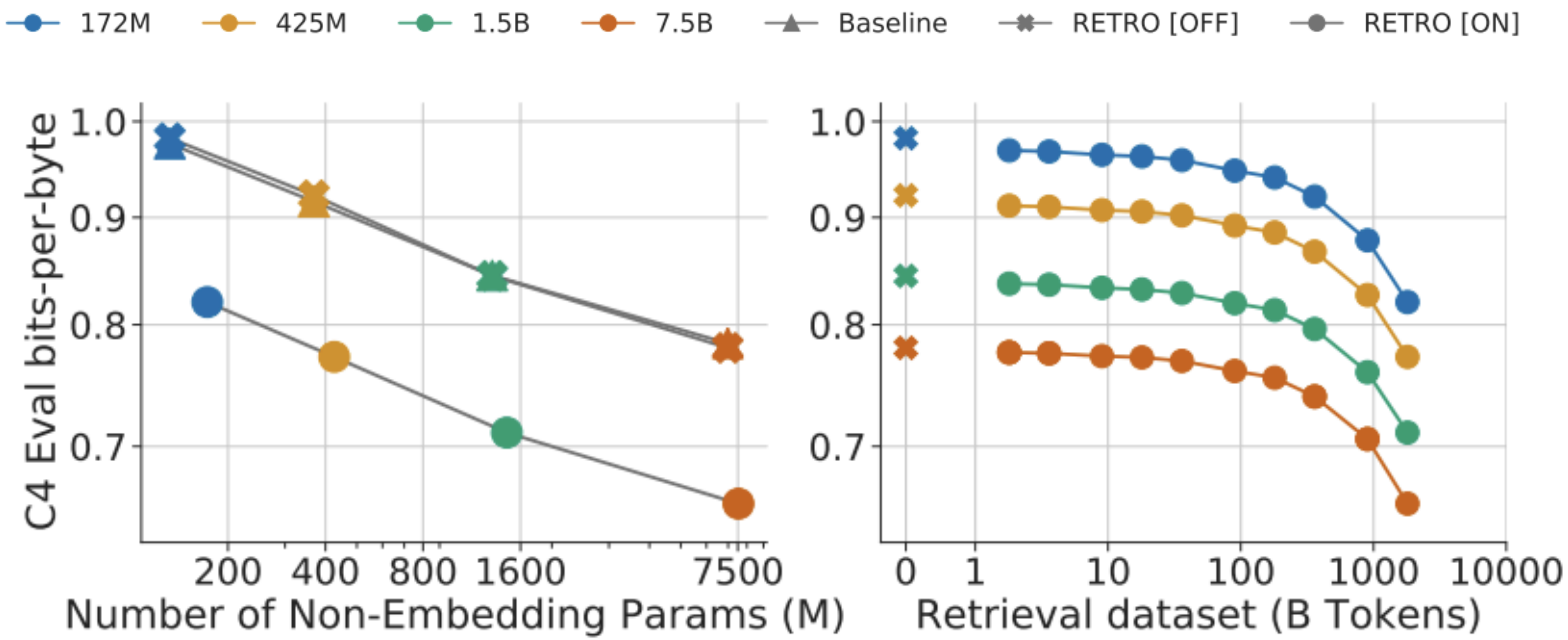
Results

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

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	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

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?

kNN-LM (Khandelwal et al. 2020)

Khandelwal et al. 2020. "Generalization through Memorization: Nearest Neighbor Language Models"

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.

Khandelwal et al. 2020. "Generalization through Memorization: Nearest Neighbor Language Models"

kNN-LM (Khandelwal et al. 2020)

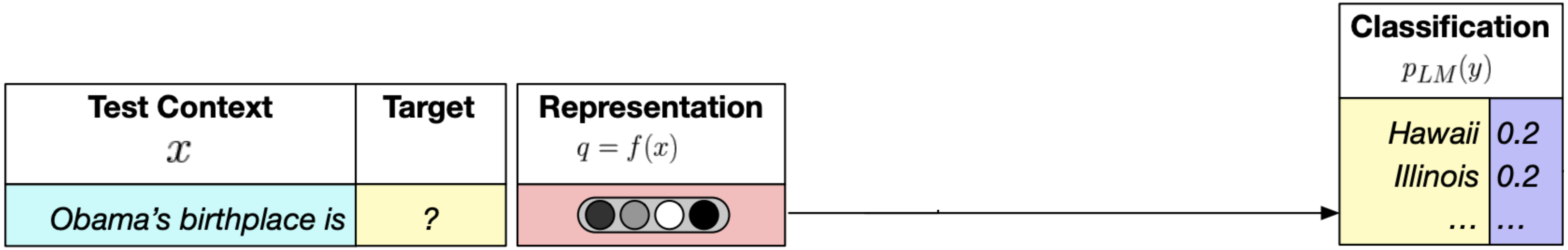
- ✓ A different way of using retrieval, where the LM outputs a nonparametric distribution over every token in the data.
- ✓ Can be seen as an incorporation in the “output” layer

Khandelwal et al. 2020. “Generalization through Memorization: Nearest Neighbor Language Models”

kNN-LM (Khandelwal et al. 2020)

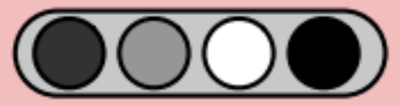
Test Context x	**Target**
Obama's birthplace is	?

kNN-LM (Khandelwal et al. 2020)



kNN-LM (Khandelwal et al. 2020)

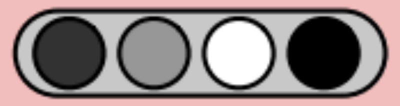
... Obama was senator for Illinois from 1997 to 2005, ... Barack is Married to Michelle and their first daughter, ... Obama was born in Hawaii, and graduated from Columbia University. ... Obama is a native of Hawaii, ...

Test Context x	**Target**	**Representation** $q = f(x)$
Obama's birthplace is	*?*	

kNN-LM (Khandelwal et al. 2020)

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

... 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, ...

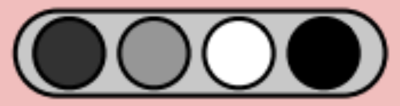
Test Context x	**Target**	**Representation** $q = f(x)$
Obama's birthplace is	?	

kNN-LM (Khandelwal et al. 2020)

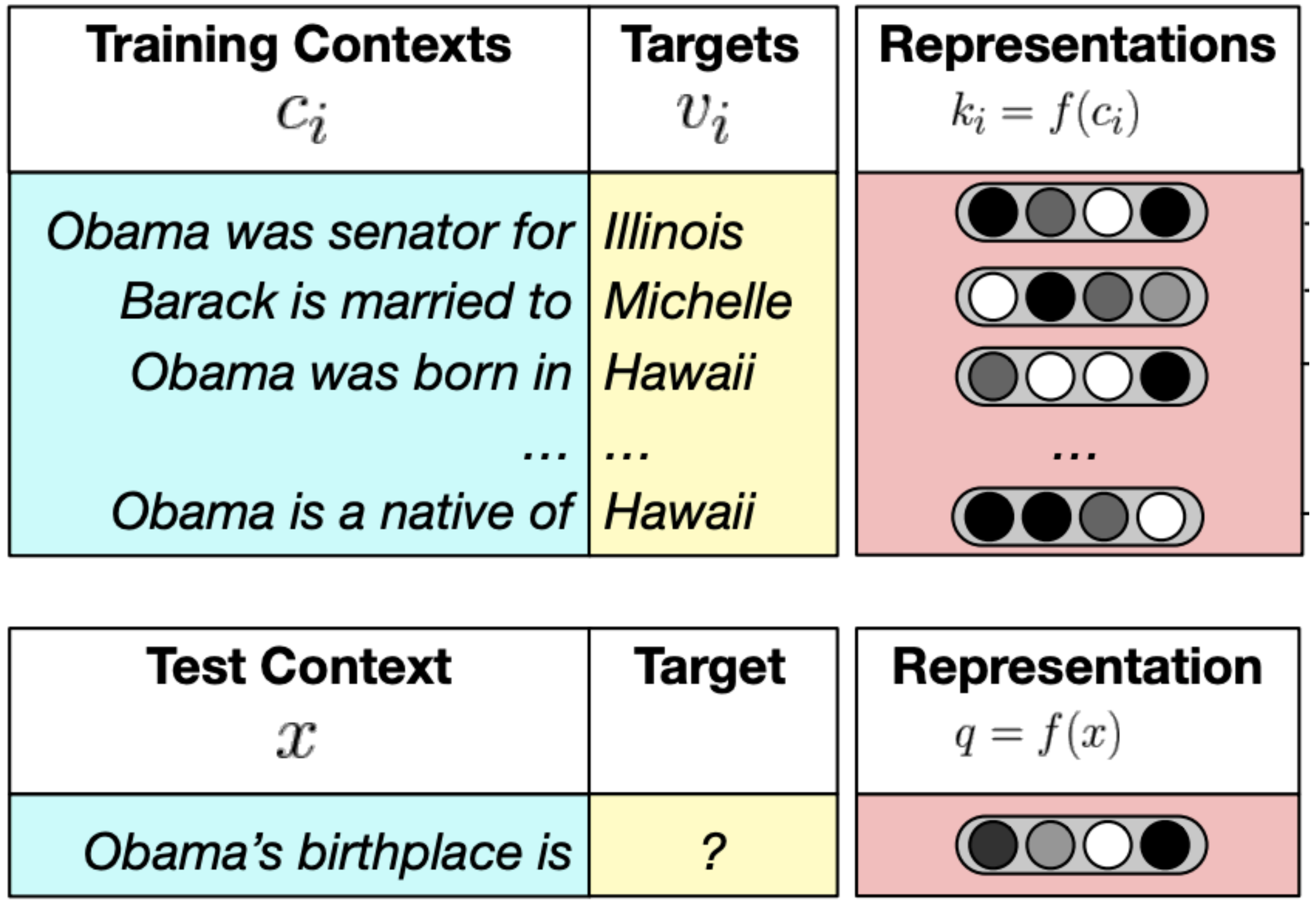
The size of the datastore = # of tokens in the corpus (>1B)

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

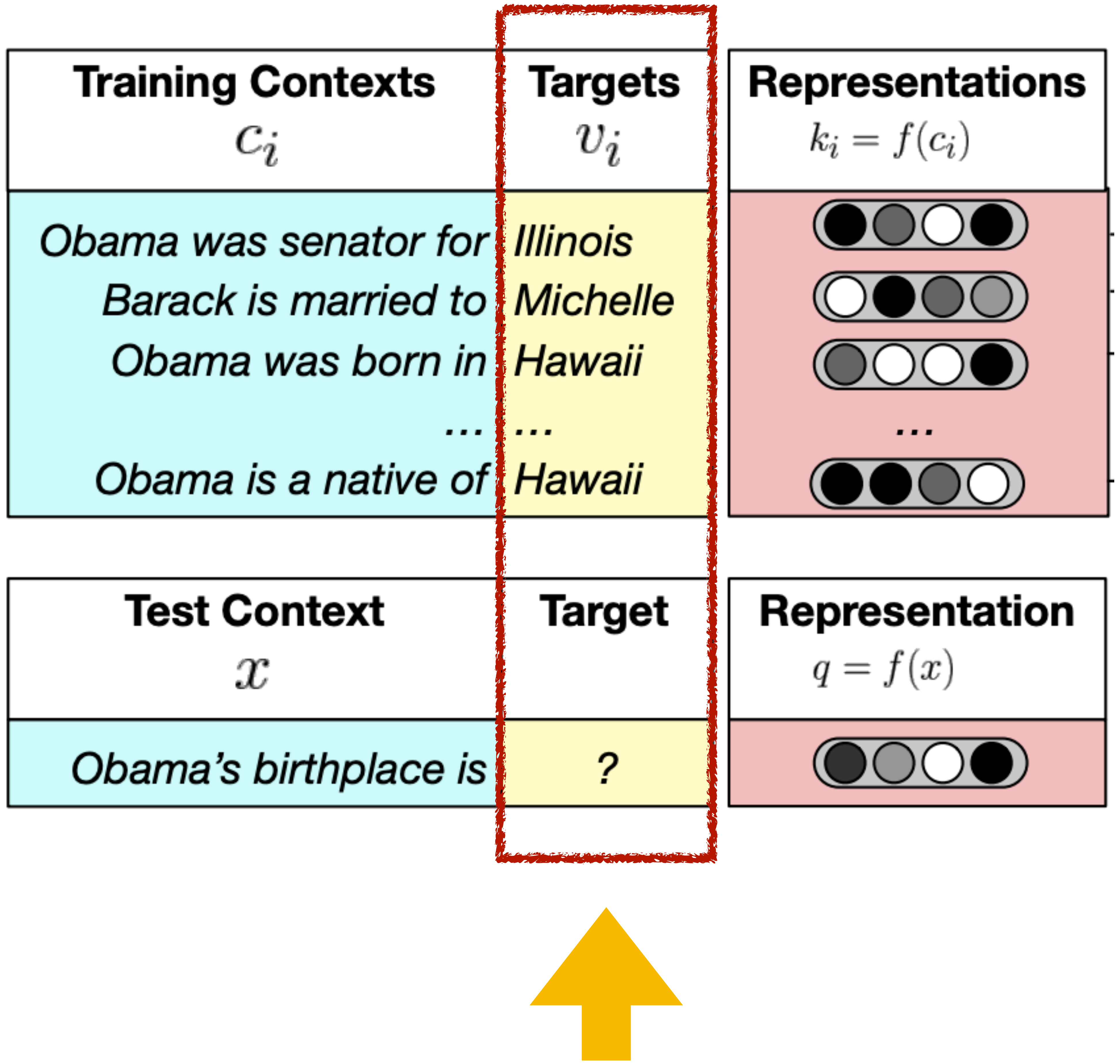
... Obama was senator for Illinois from 1997 to 2005, ... Barack is Married to Michelle and their first daughter, ... Obama was born in Hawaii, and graduated from Columbia University. ... Obama is a native of Hawaii, ...

Test Context x	**Target**	**Representation** $q = f(x)$
Obama's birthplace is	?	

kNN-LM (Khandelwal et al. 2020)

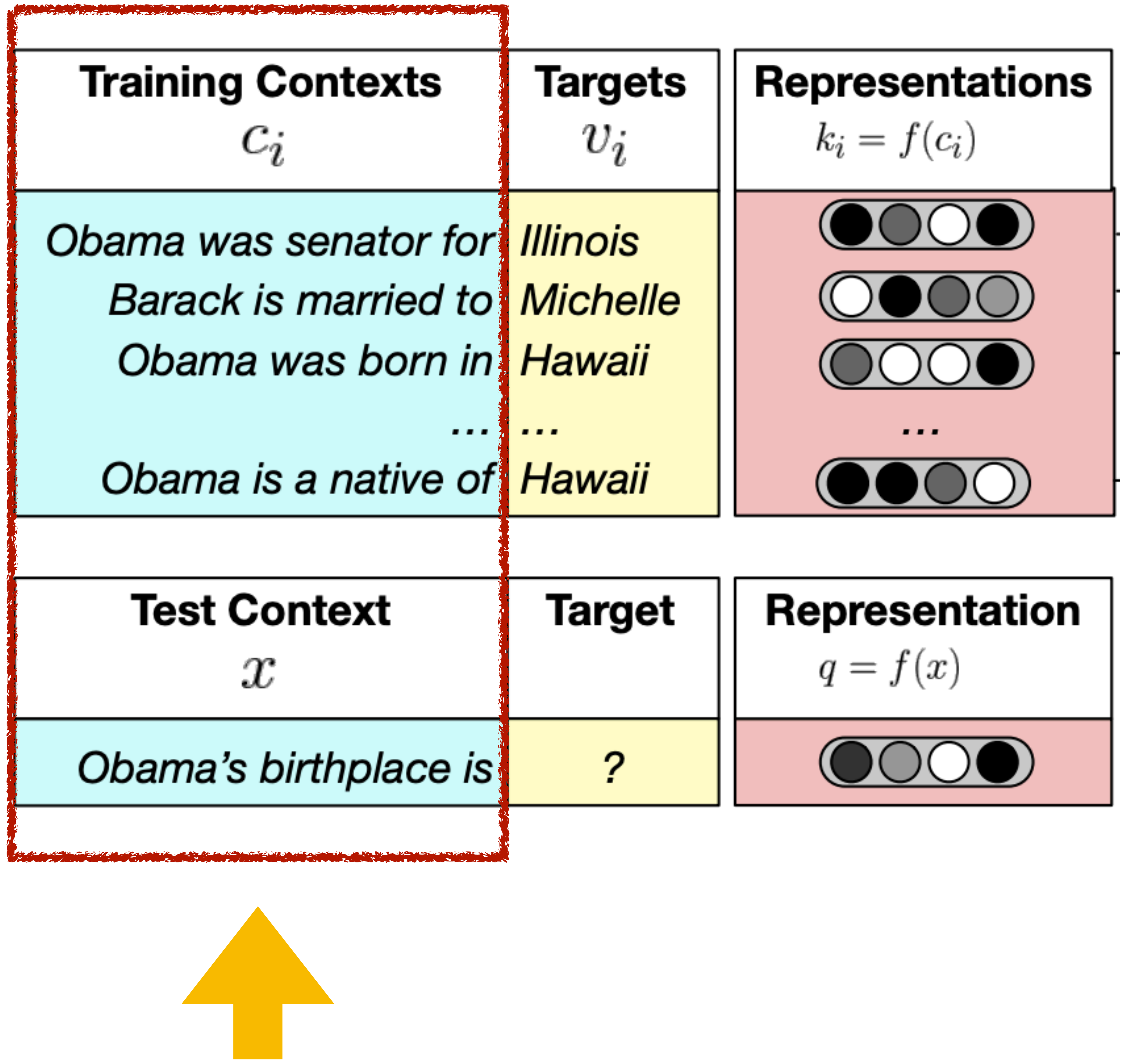


kNN-LM (Khandelwal et al. 2020)



Which tokens <u>in a datastore</u> are close to the next token?

kNN-LM (Khandelwal et al. 2020)

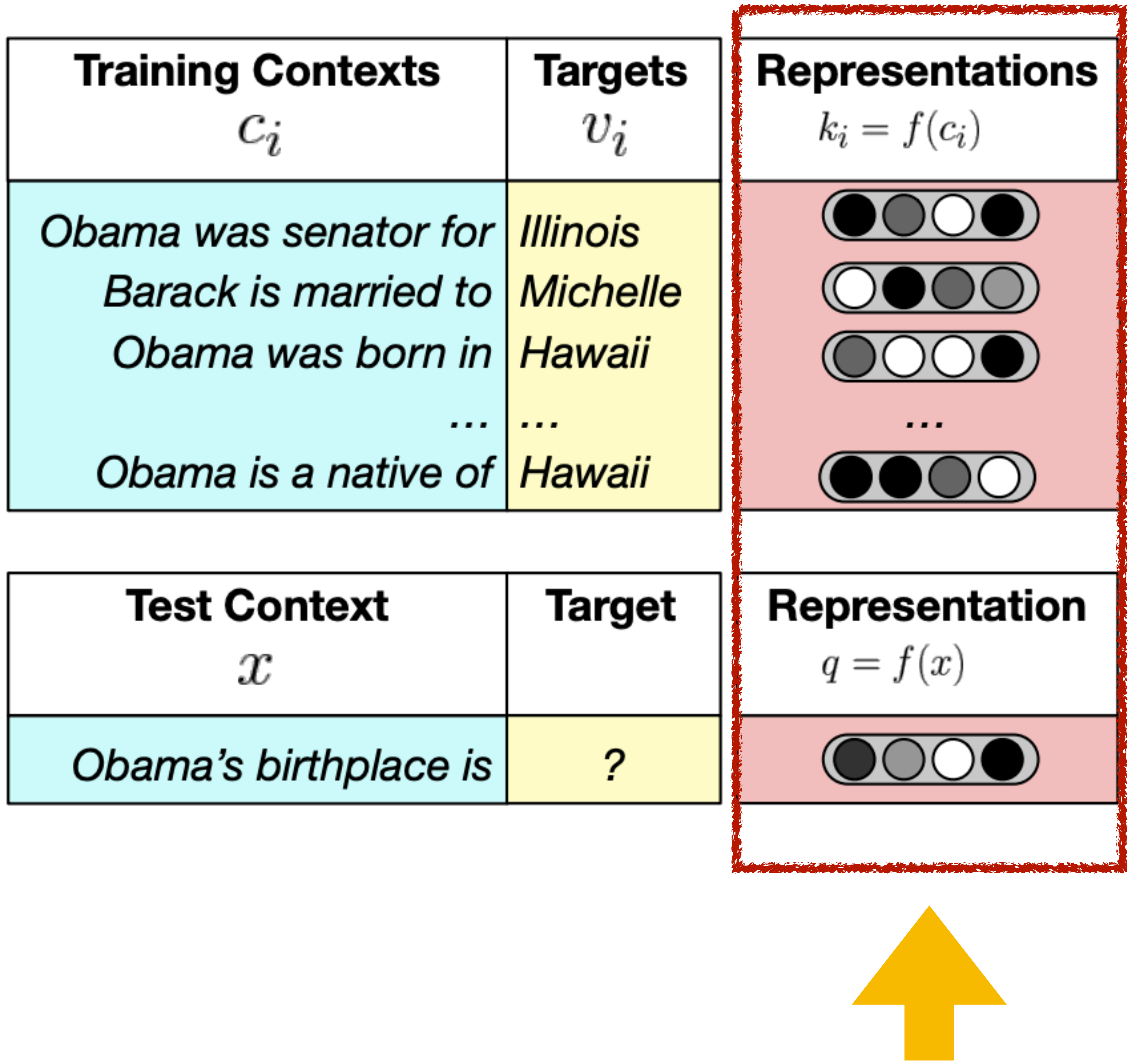


Which tokens <u>in a datastore</u> are close to the next token?

=

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

kNN-LM (Khandelwal et al. 2020)



Which tokens <u>in a datastore</u> are close to the next token?

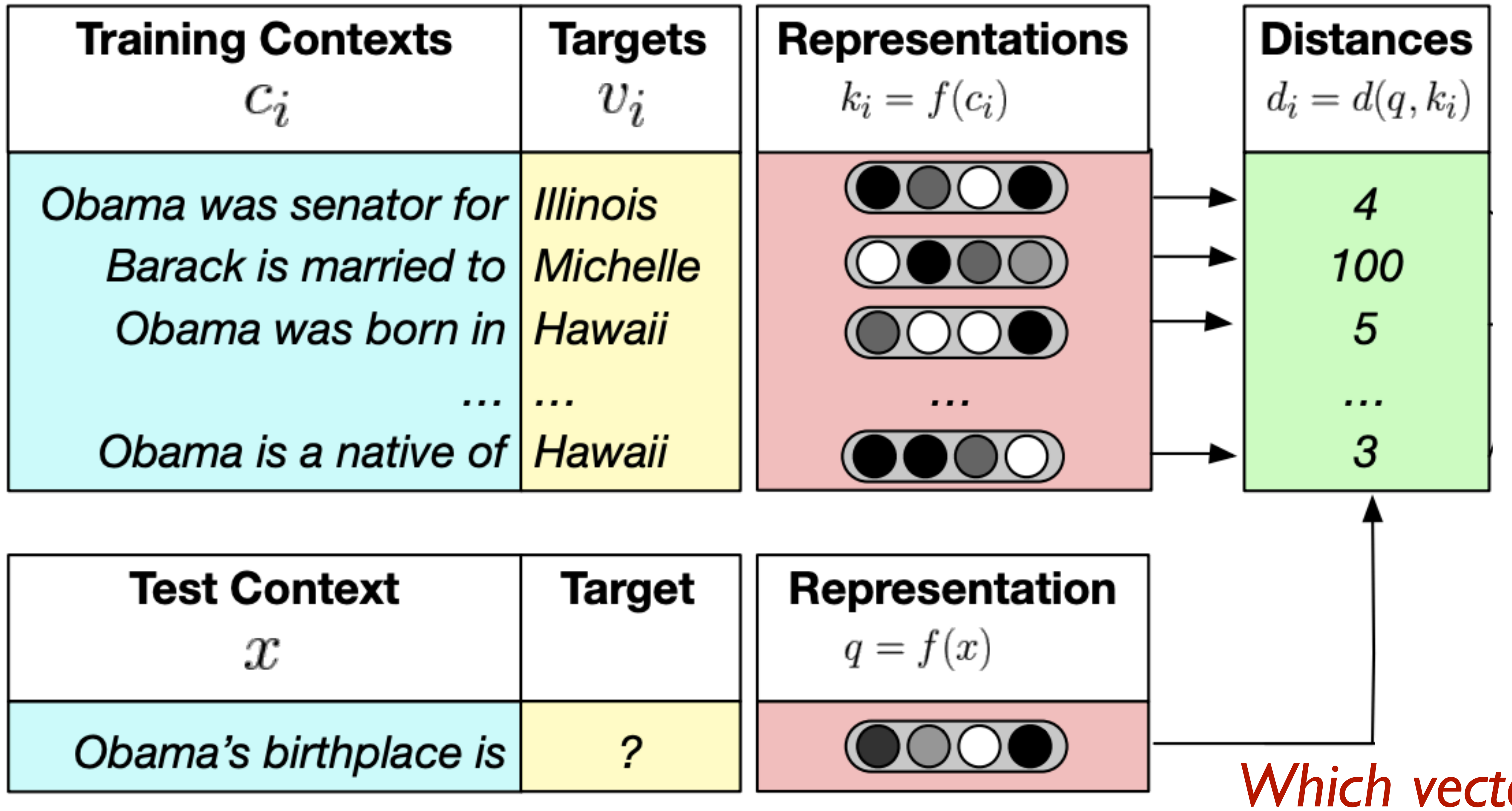
=

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

=

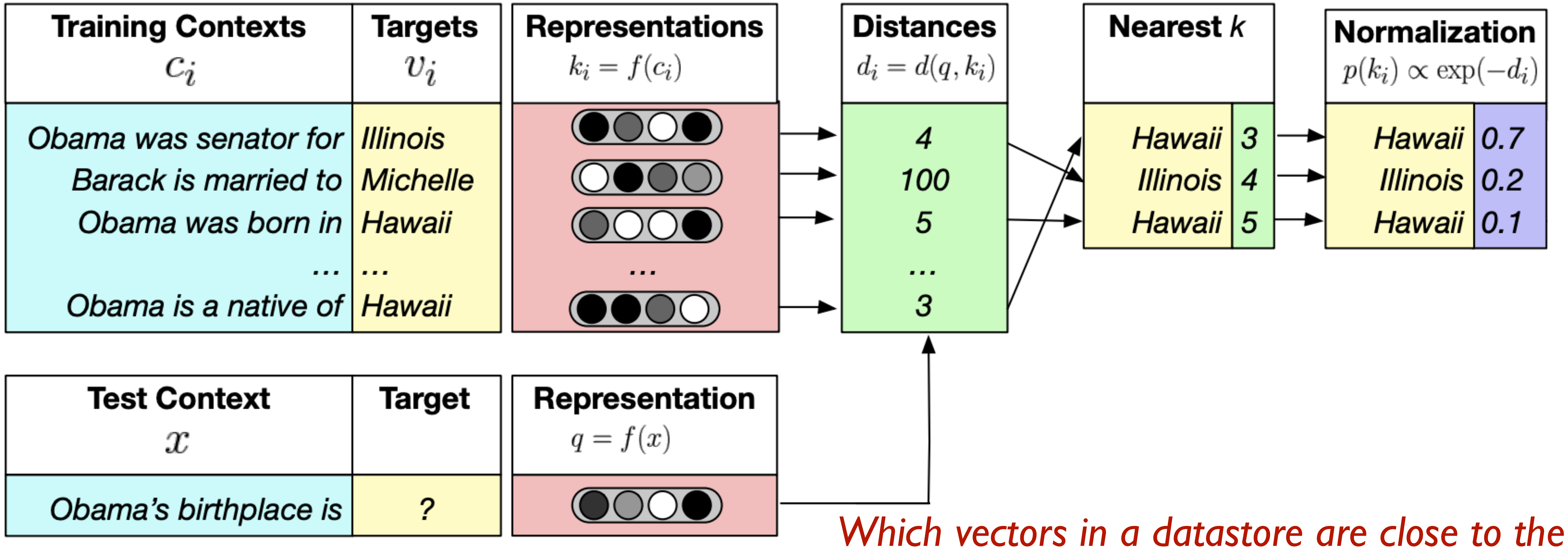
Which vectors in a datastore are close to the vector we have?

kNN-LM (Khandelwal et al. 2020)



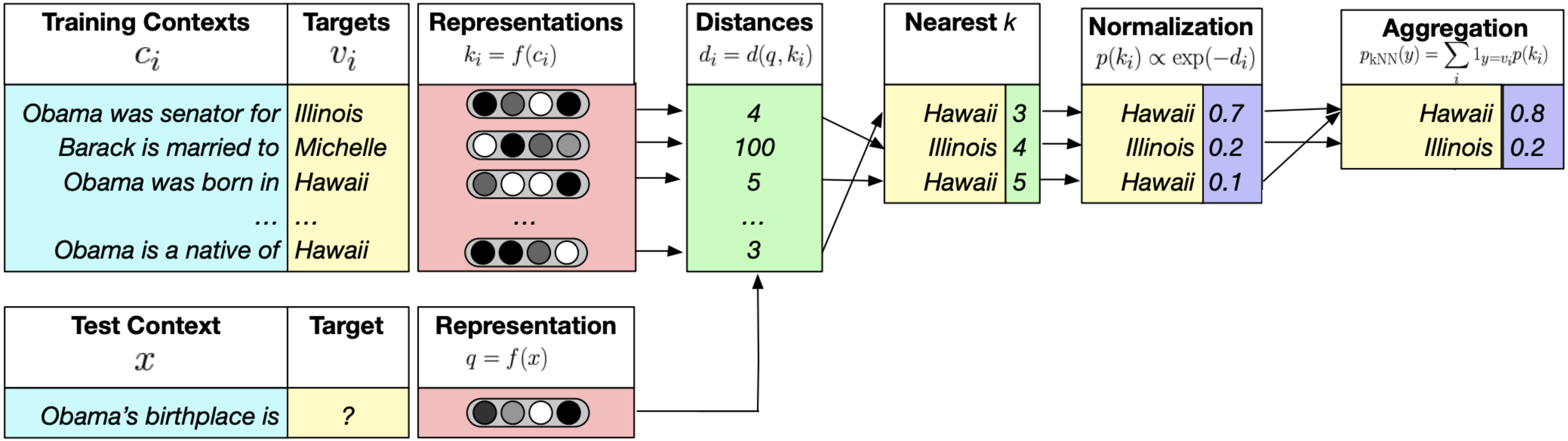
Which vectors in a datastore are close to the vector we have?

kNN-LM (Khandelwal et al. 2020)

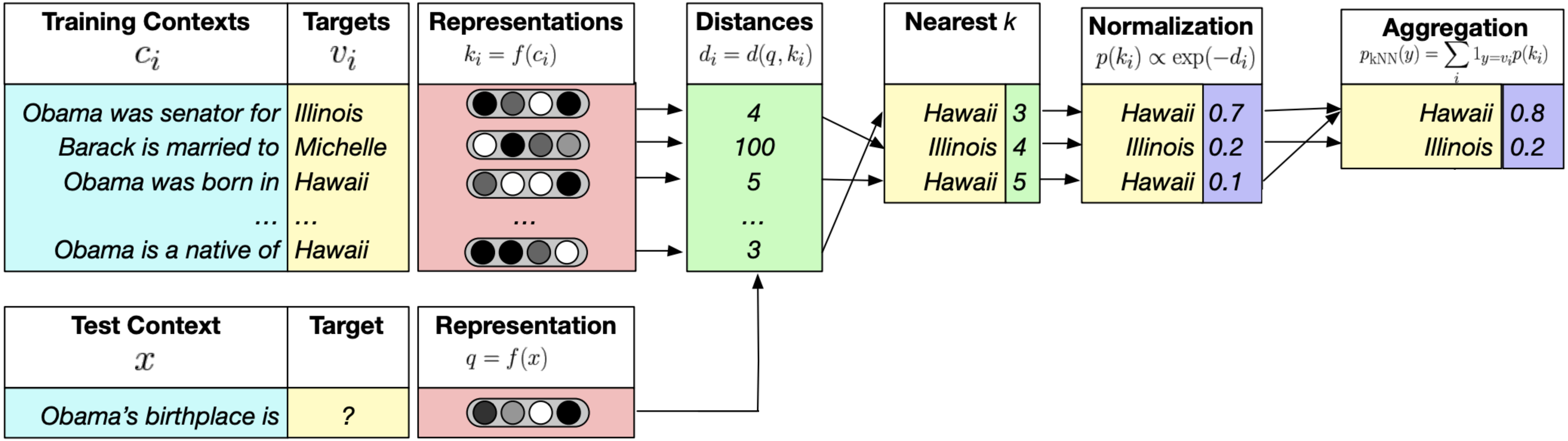


Which vectors in a datastore are close to the vector we have?

kNN-LM (Khandelwal et al. 2020)

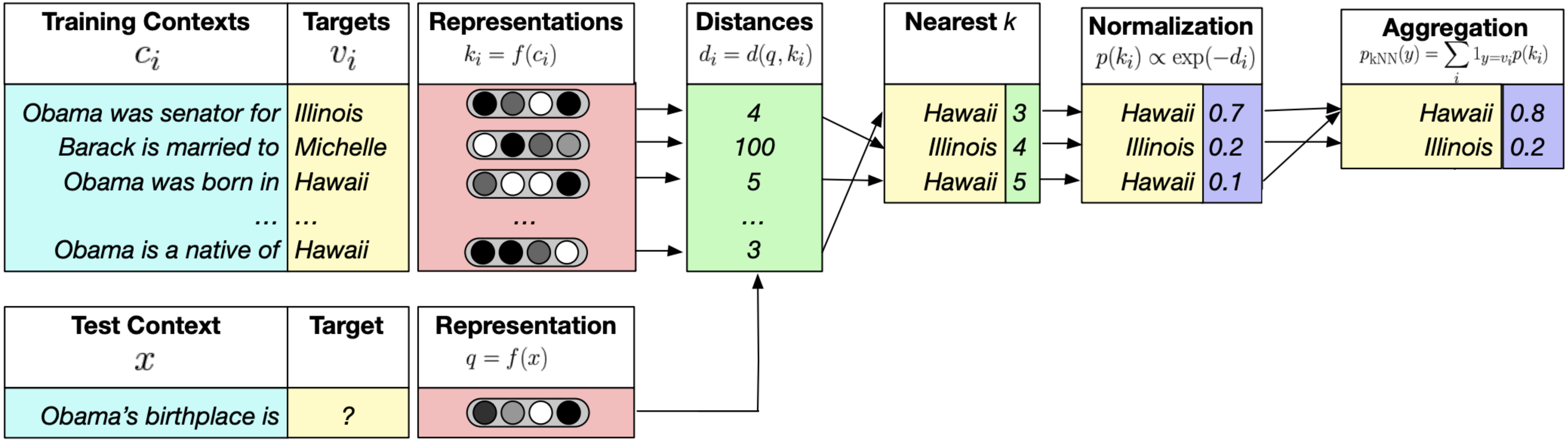


kNN-LM (Khandelwal et al. 2020)



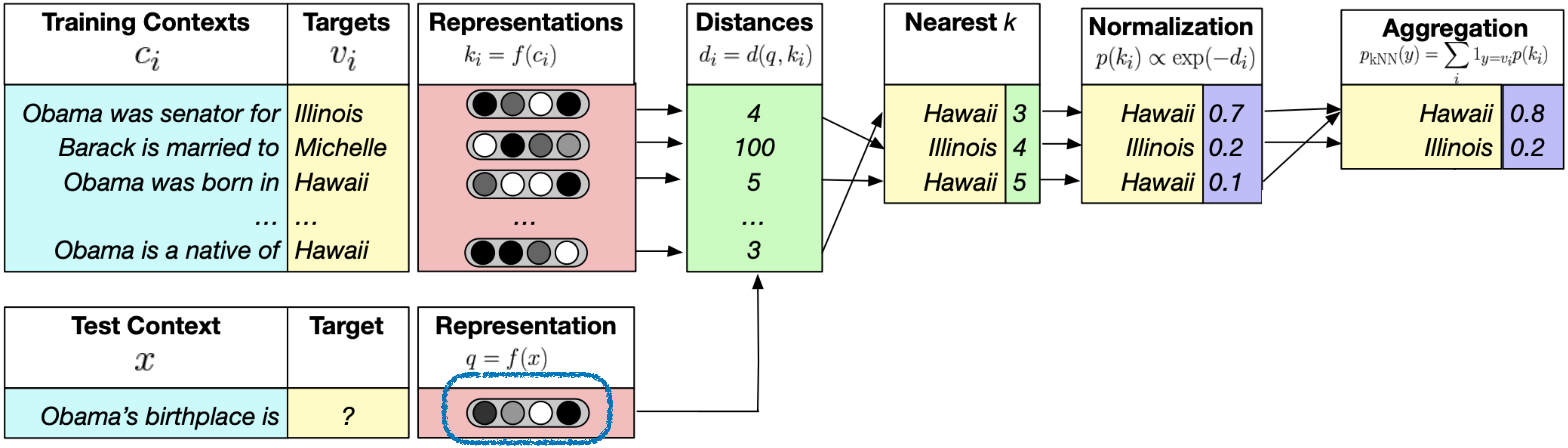
$$P_{k\mathrm{NN}}(y \mid x) \propto \sum_{(k,v)\in\mathcal{D}} \mathbb{1}[v = y]\,\mathrm{sim}(k, x)$$

kNN-LM (Khandelwal et al. 2020)



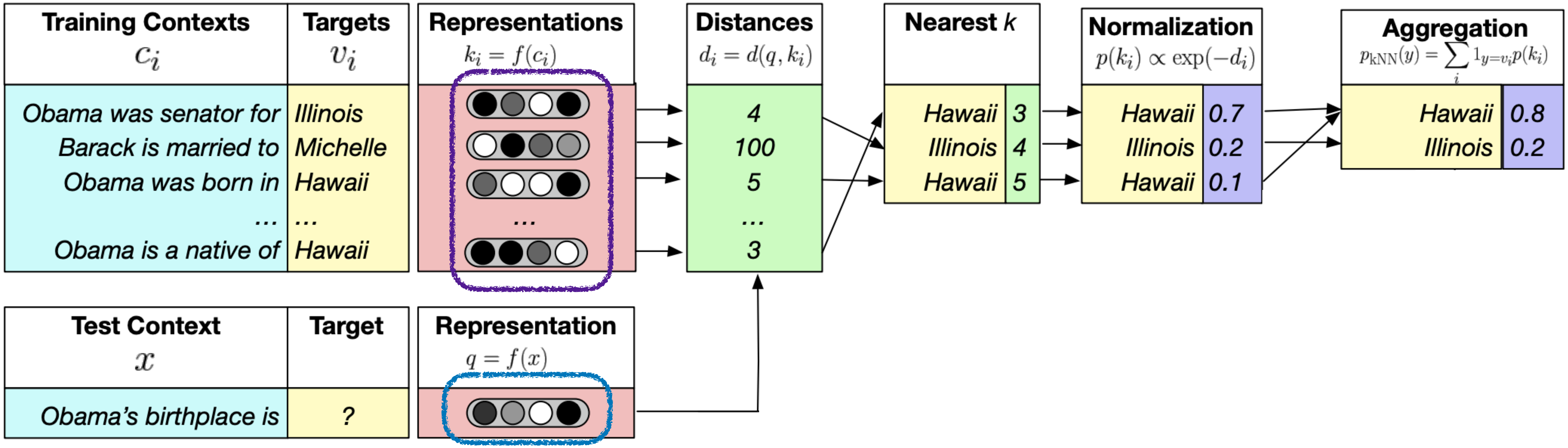
$$P_{kNN}(y \mid x) \propto \sum_{(k,v)\in\mathcal{D}} \mathbb{I}[v = y]\,\mathrm{sim}(k, x) \qquad \mathrm{sim}(k, x) = \exp\left(-d(\mathrm{Enc}(k), \mathrm{Enc}(x))\right)$$

kNN-LM (Khandelwal et al. 2020)



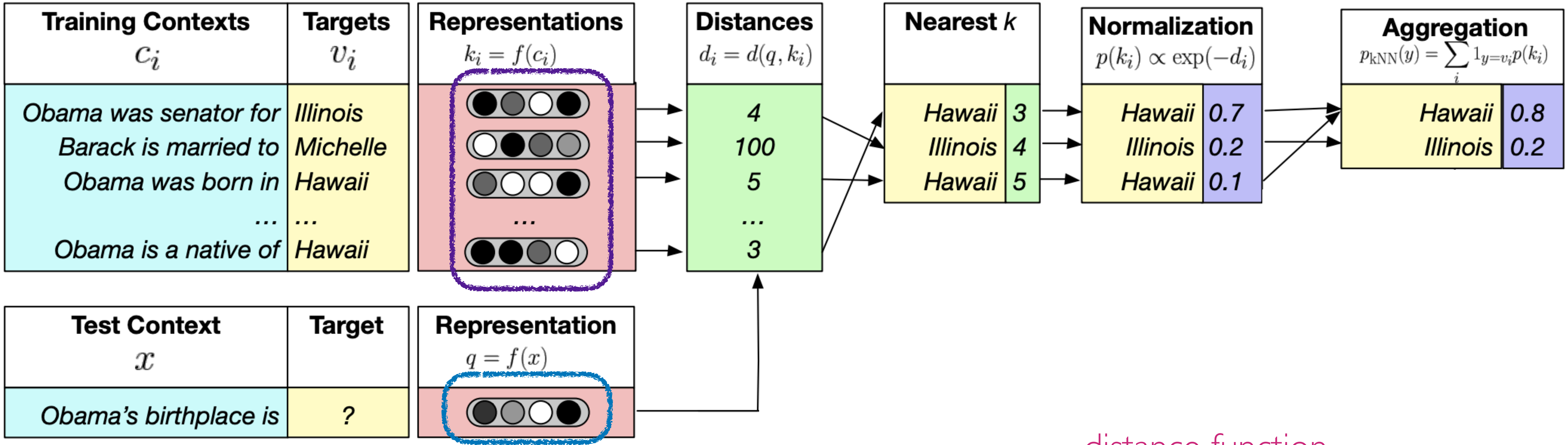
$$P_{k\mathrm{NN}}(y \mid x) \propto \sum_{(k,v)\in\mathcal{D}} \mathbb{1}[v = y]\,\mathrm{sim}(k, x) \qquad \mathrm{sim}(k, x) = \exp\left(-d(\mathrm{Enc}(k), \underline{\mathrm{Enc}(x)})\right)$$

kNN-LM (Khandelwal et al. 2020)



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

kNN-LM (Khandelwal et al. 2020)

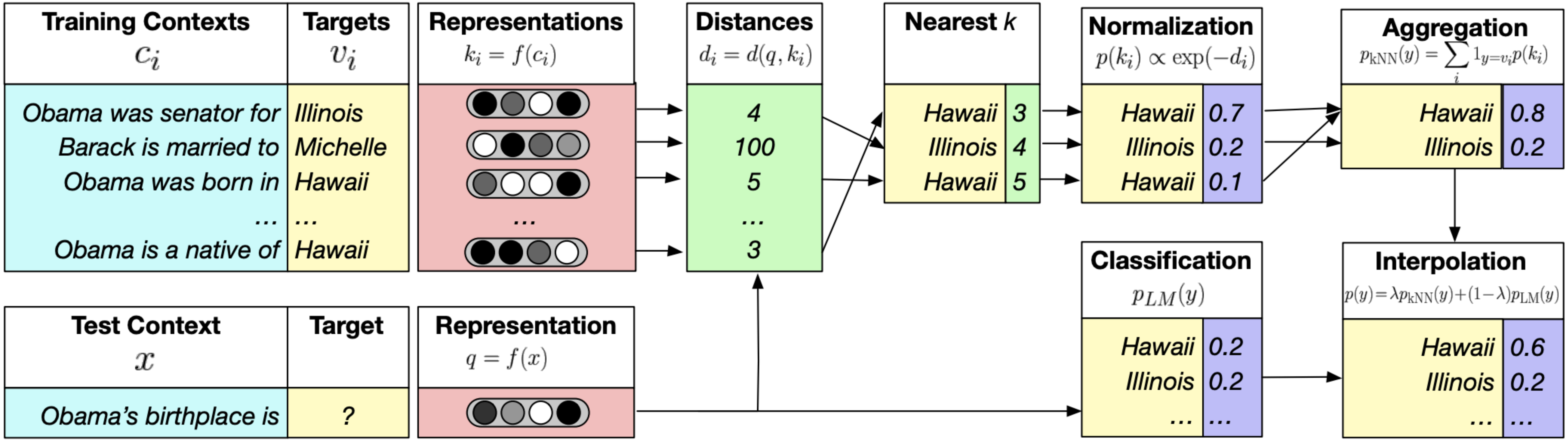


$$P_{kNN}(y \mid x) \propto \sum_{(k,v)\in\mathcal{D}} \mathbb{1}[v = y]\mathrm{sim}(k, x)$$

distance function

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

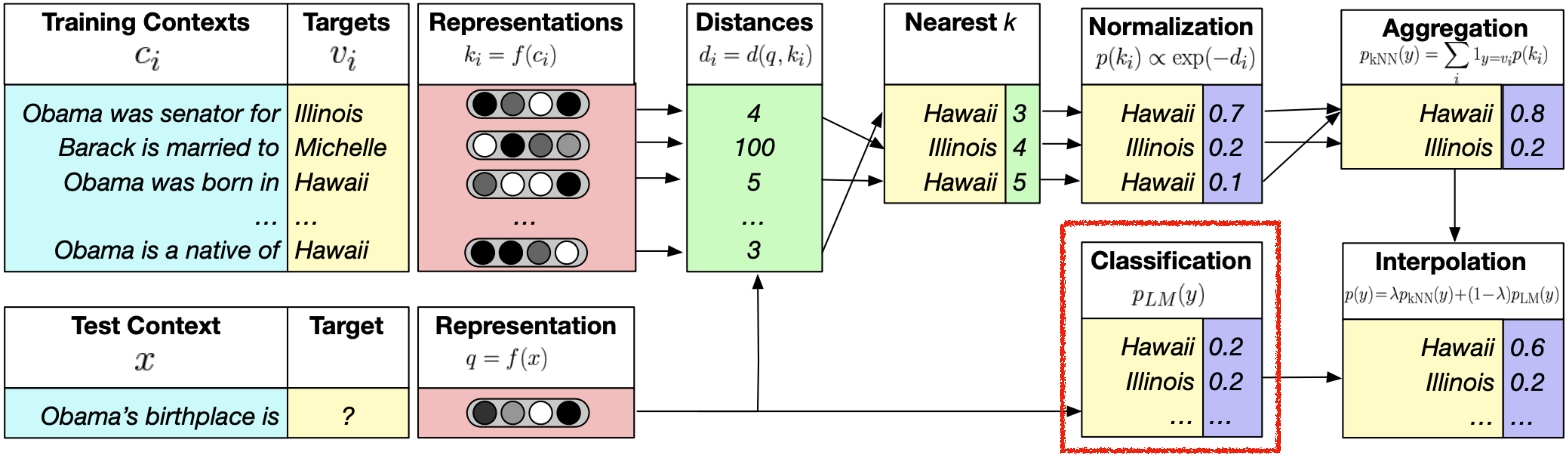
kNN-LM (Khandelwal et al. 2020)



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

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

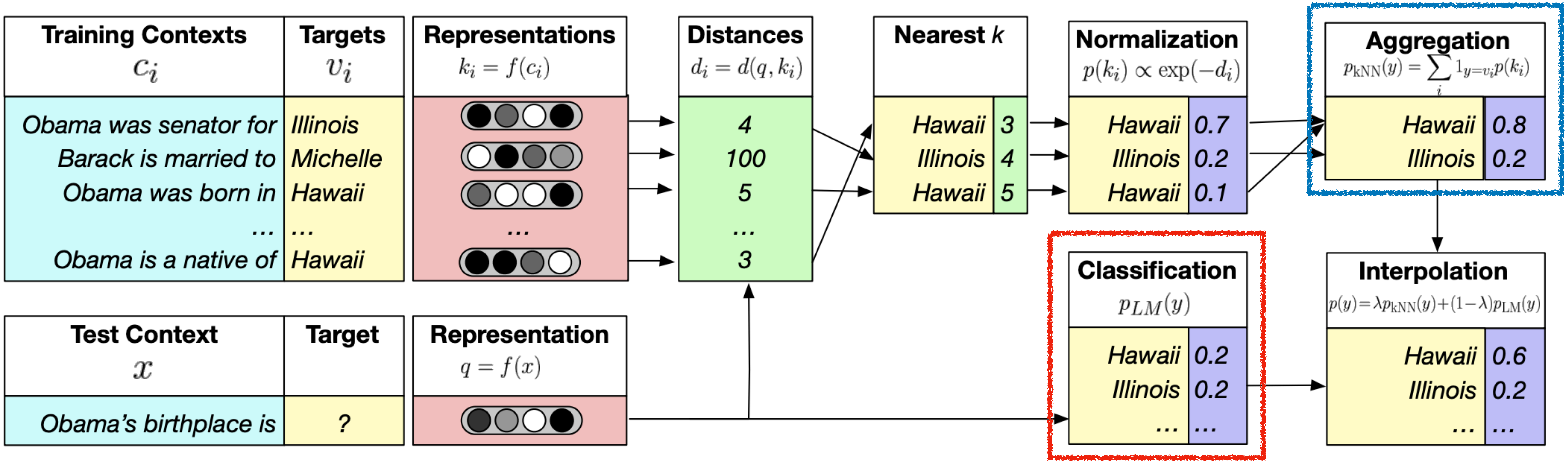
kNN-LM (Khandelwal et al. 2020)



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

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

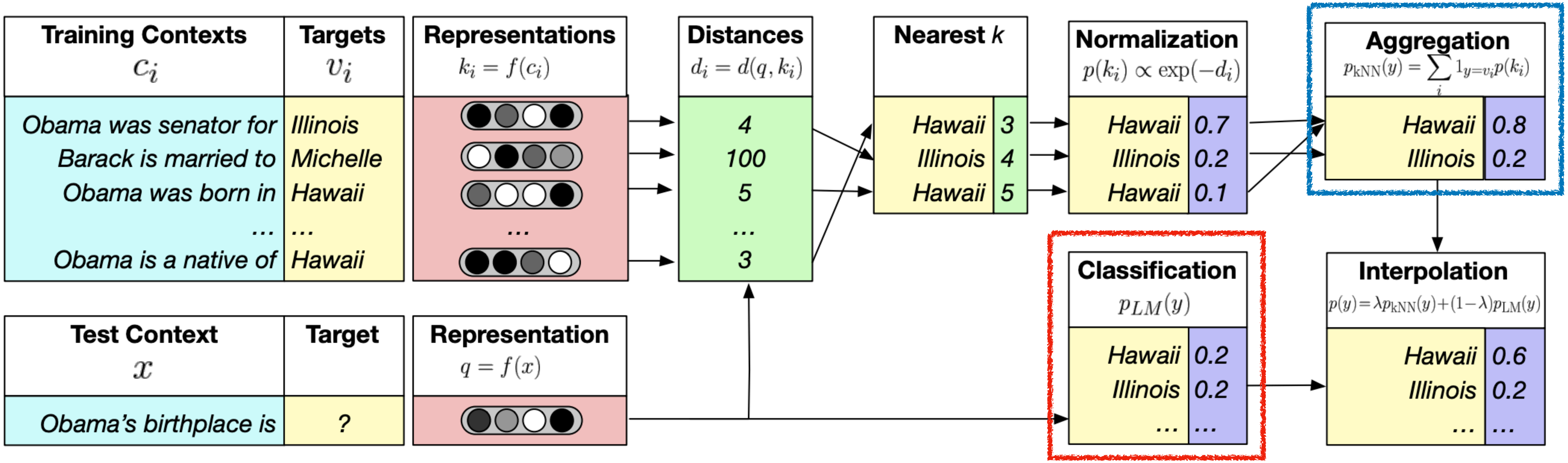
kNN-LM (Khandelwal et al. 2020)



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

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

kNN-LM (Khandelwal et al. 2020)



λ: hyperparameter

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

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

kNN-LM - why?

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

kNN-LM - why?

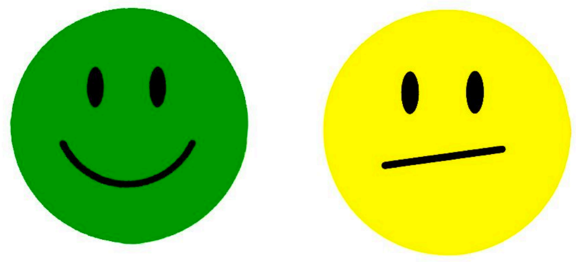
Training contexts	Targets
10/10, would buy this	***cheap***
Item delivered broken. Very	***cheap***
To check the version of PyTorch, you can use	*torch*
You are permitted to bring a	*torch*
A group of infections … one of the	*torch*

kNN-LM - why?

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

kNN-LM - why?

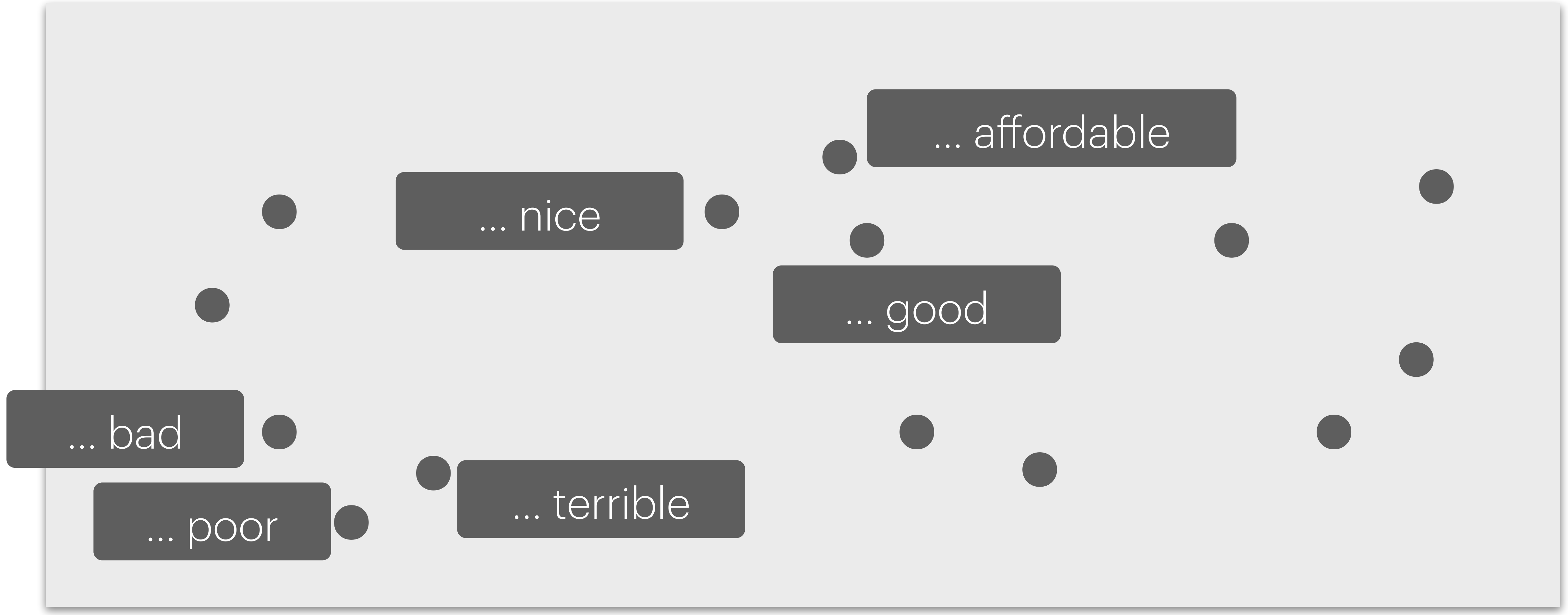
Training contexts	Targets
10/10, would buy this	***cheap***
Item delivered broken. Very	***cheap***
To check the version of PyTorch, you can use	*torch*
You are permitted to bring a	*torch*
A group of infections … one of the	*torch*



kNN-LM - why?

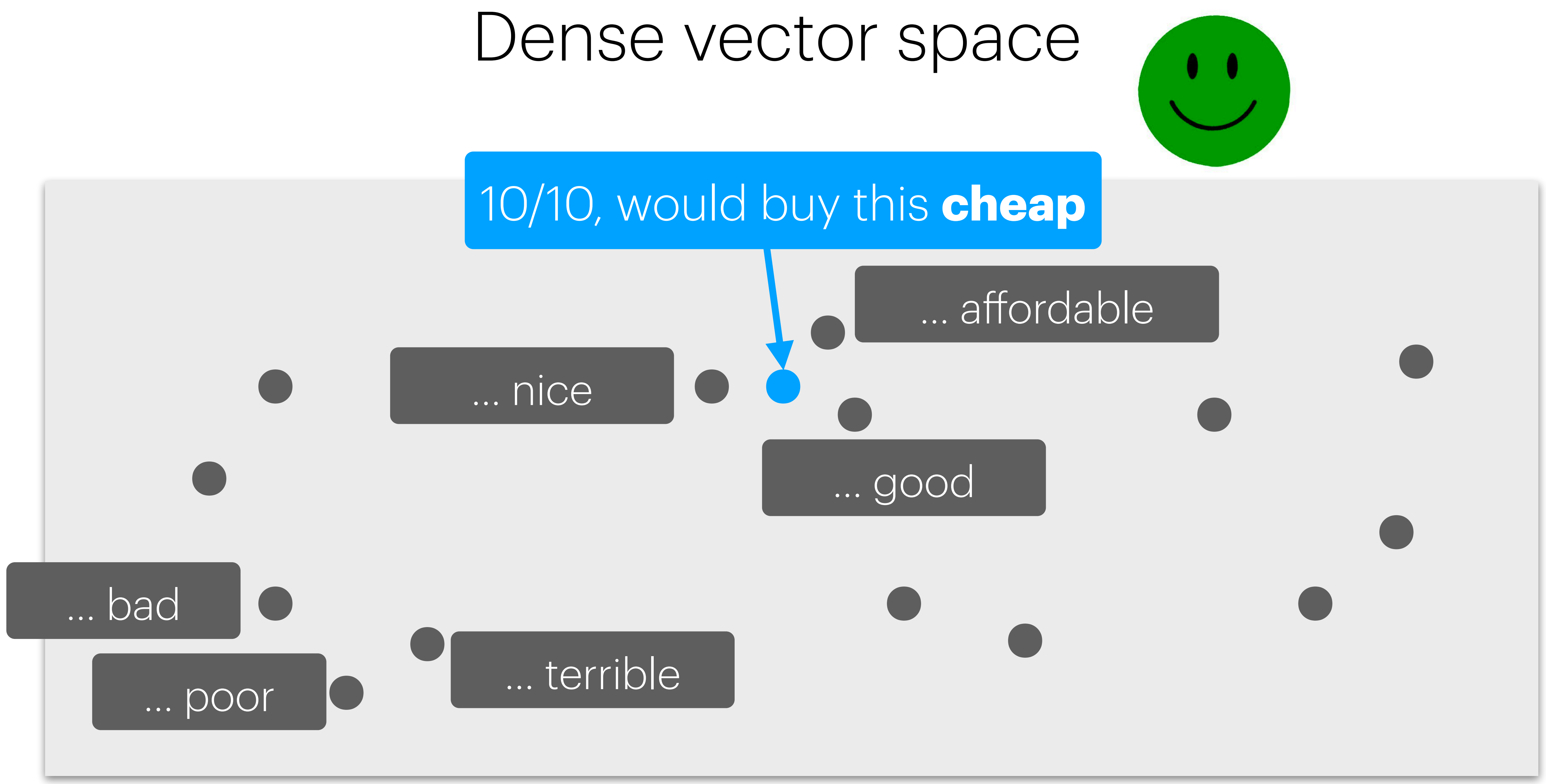
Training contexts	Targets
10/10, would buy this	***cheap***
Item delivered broken. Very	***cheap***
To check the version of PyTorch, you can use	*torch*
You are permitted to bring a	*torch*
A group of infections … one of the	*torch*

Dense vector space



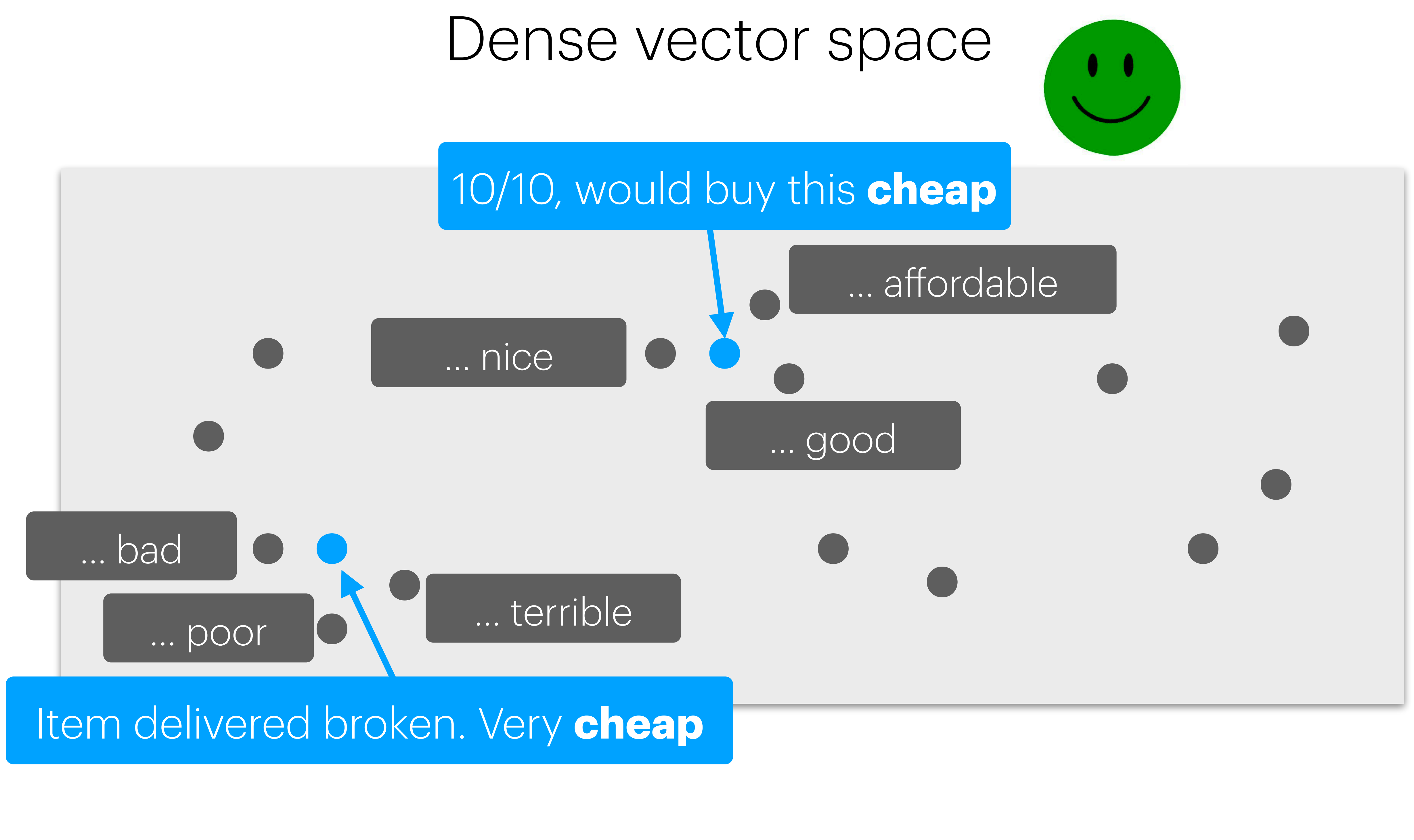
kNN-LM - why?

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



kNN-LM - why?

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



kNN-LM - why?

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

kNN-LM - why?

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

kNN-LM - why?

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

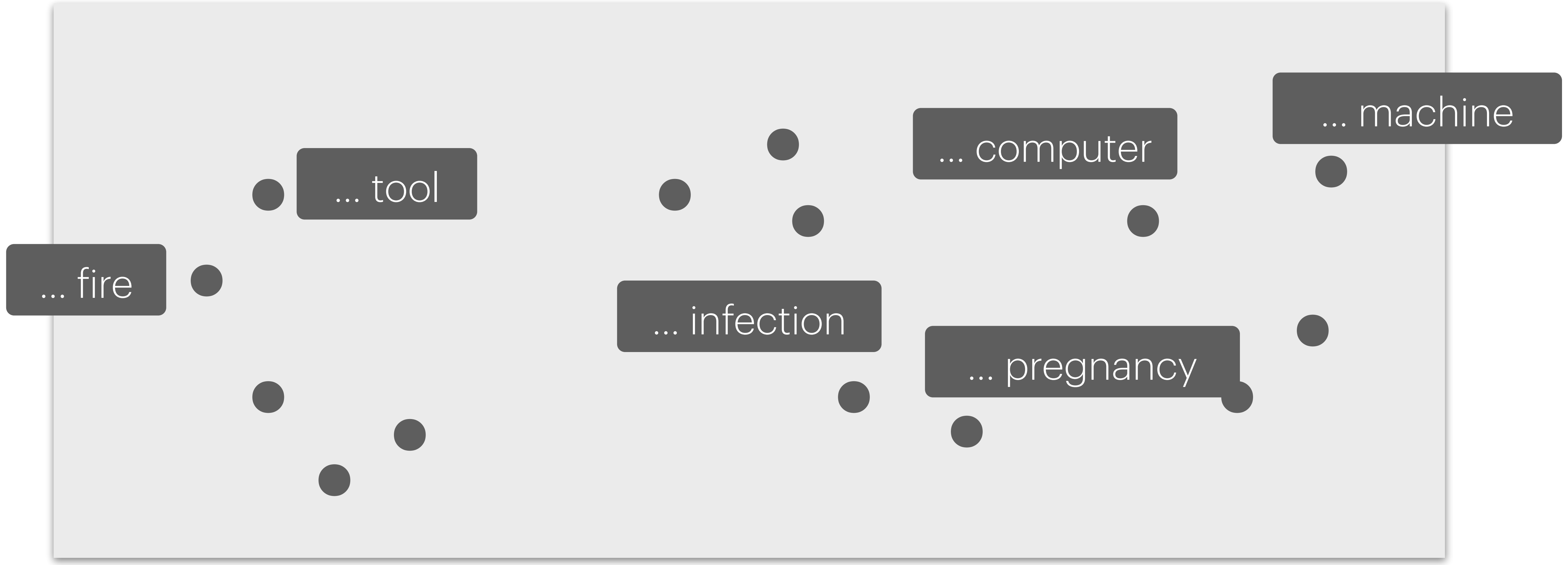
kNN-LM - why?

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

kNN-LM - why?

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

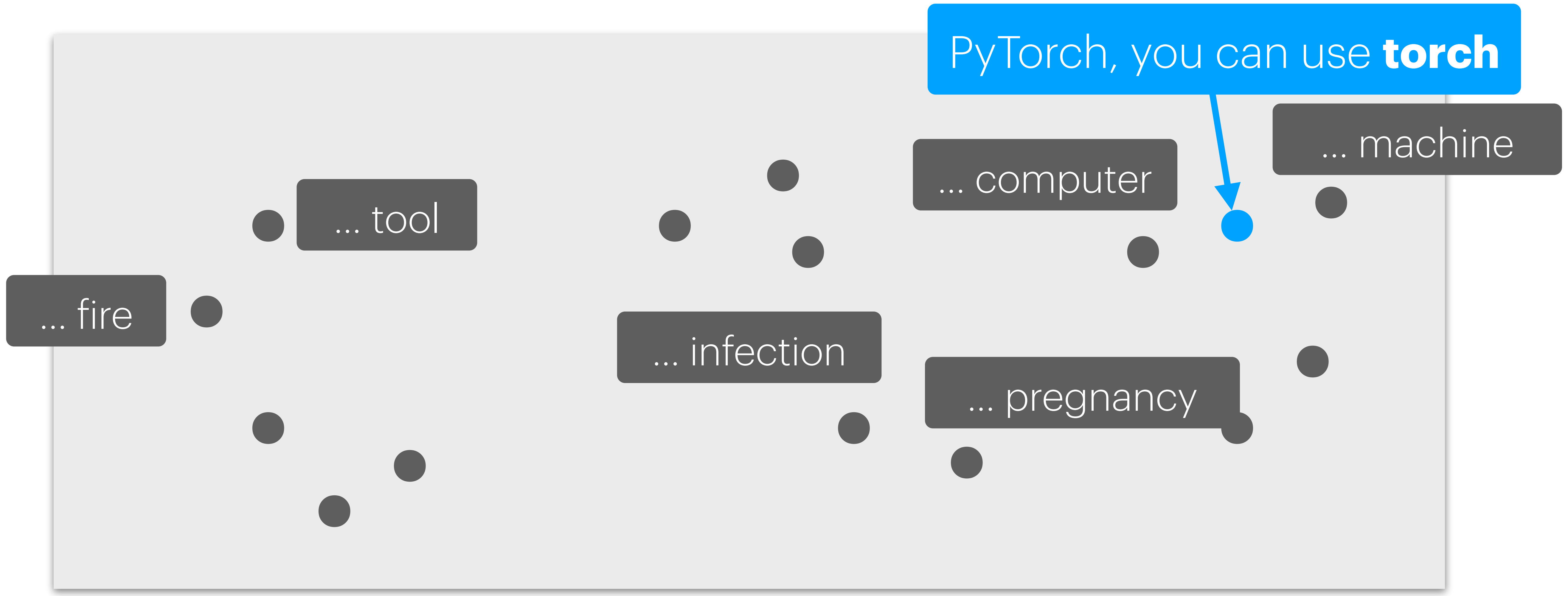
Dense vector space



kNN-LM - why?

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

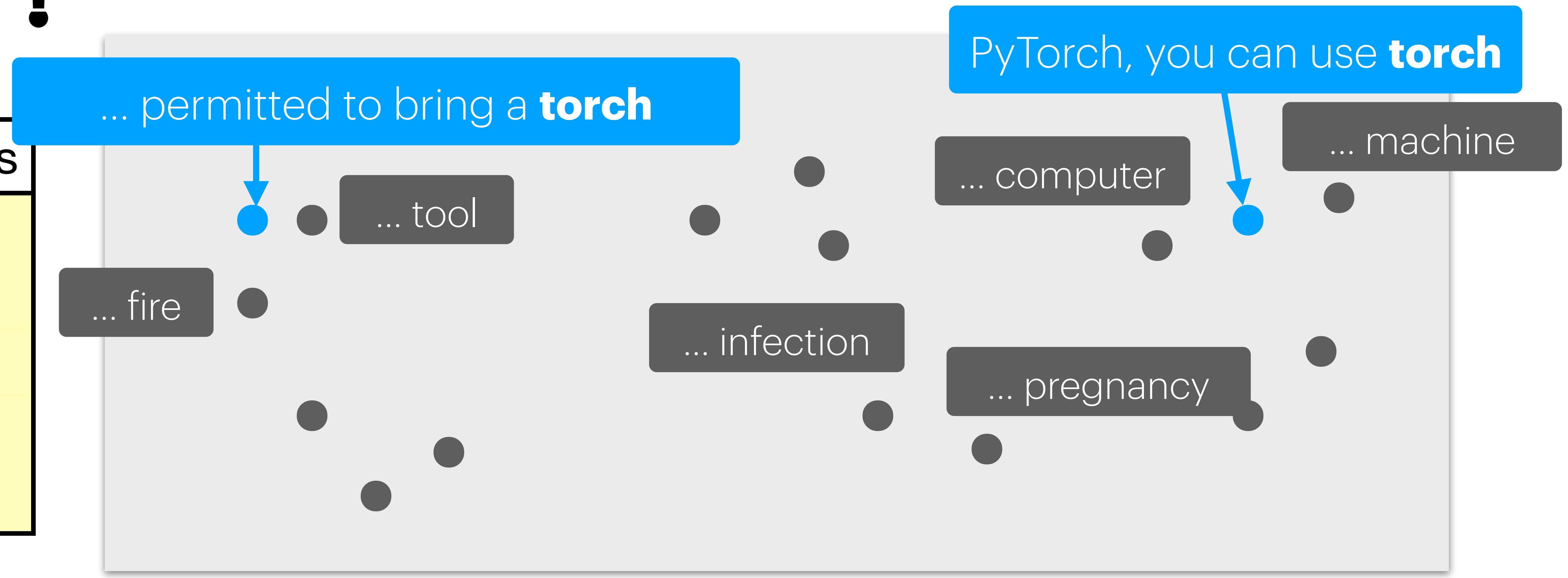
Dense vector space



kNN-LM - why?

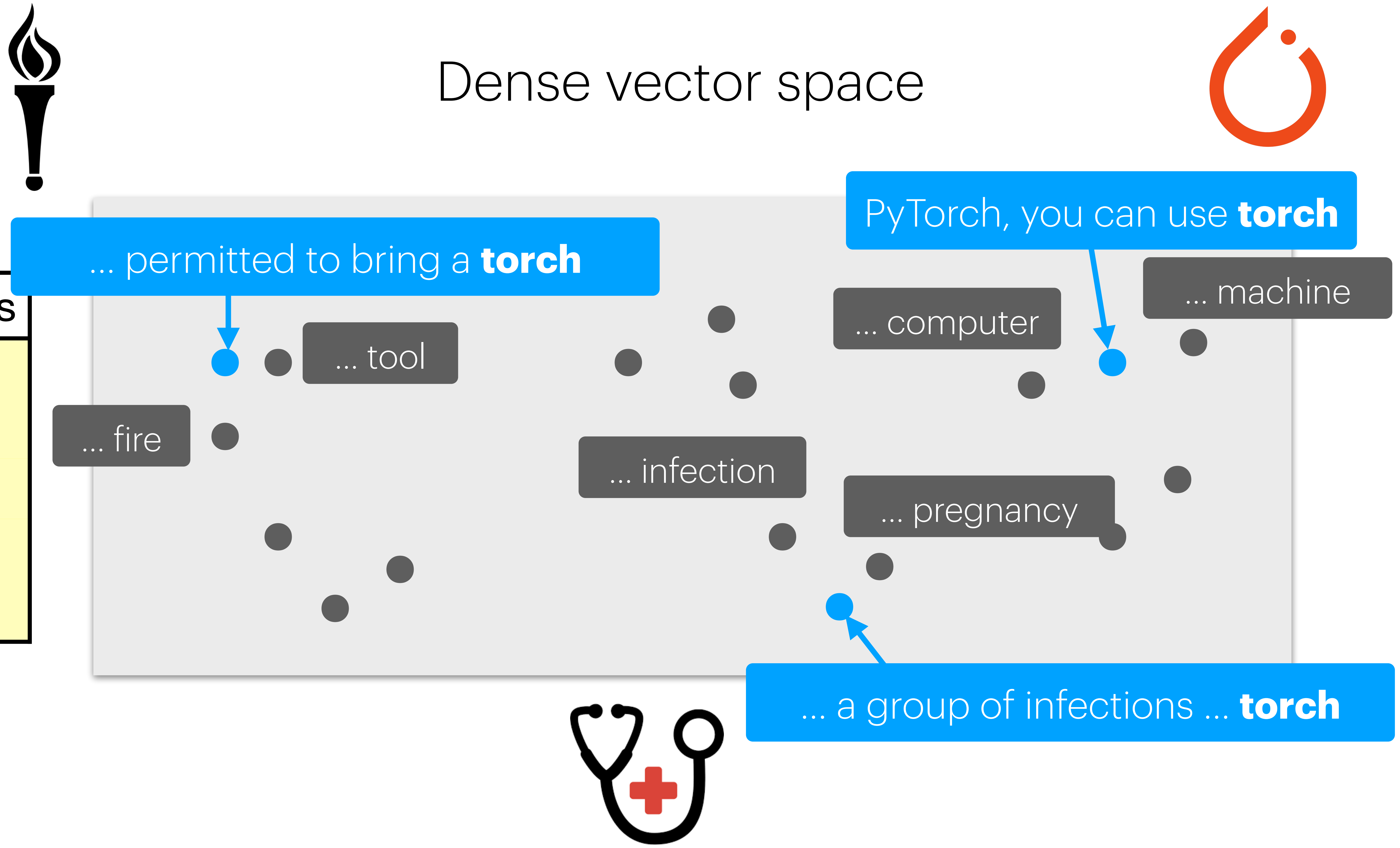
Training contexts	Targets
10/10, would buy this	*cheap*
Item delivered broken. Very	*cheap*
To check the version of PyTorch, you can use	*torch*
You are permitted to bring a	*torch*
A group of infections … one of the	*torch*

Dense vector space



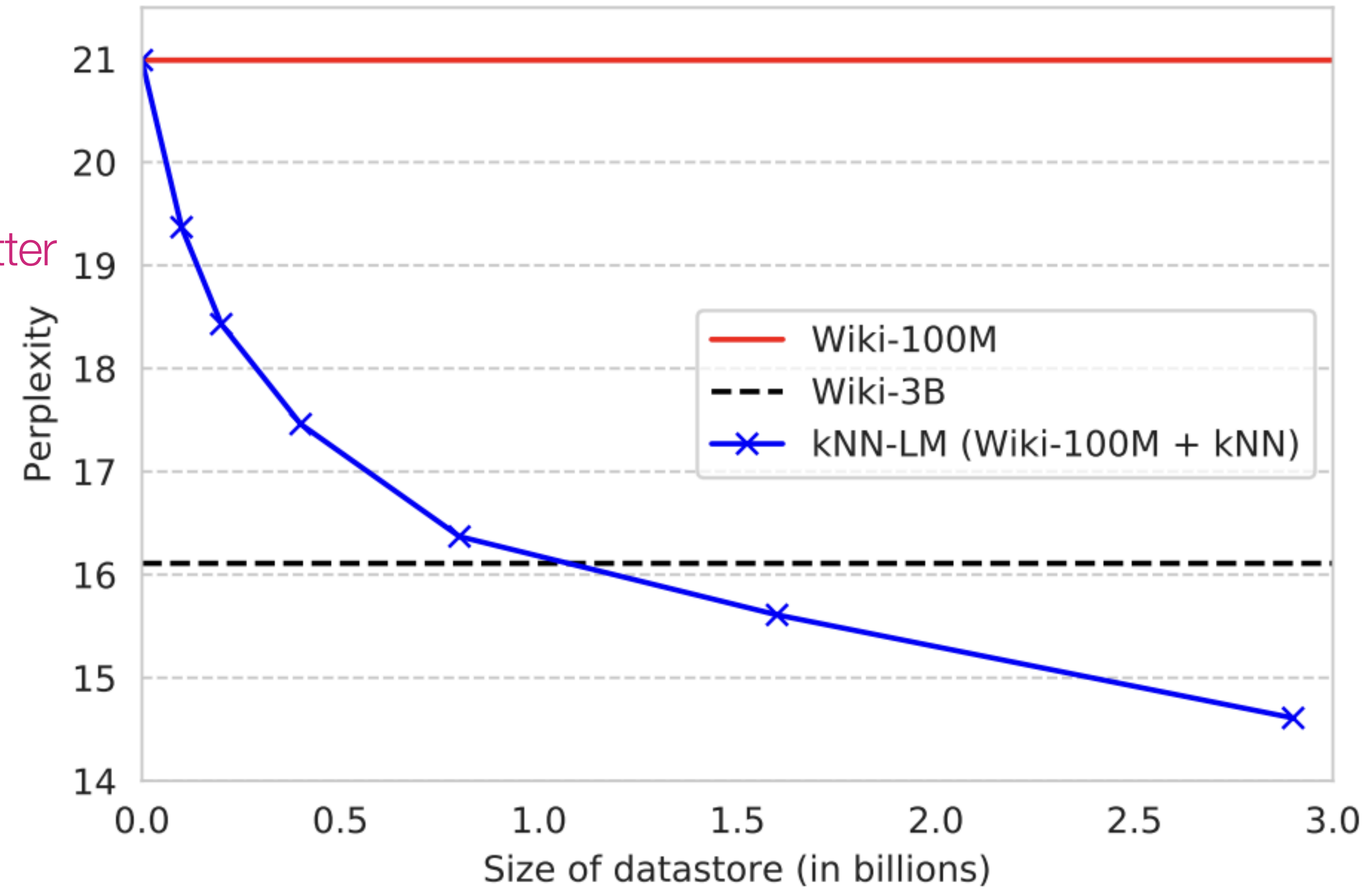
kNN-LM - why?

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

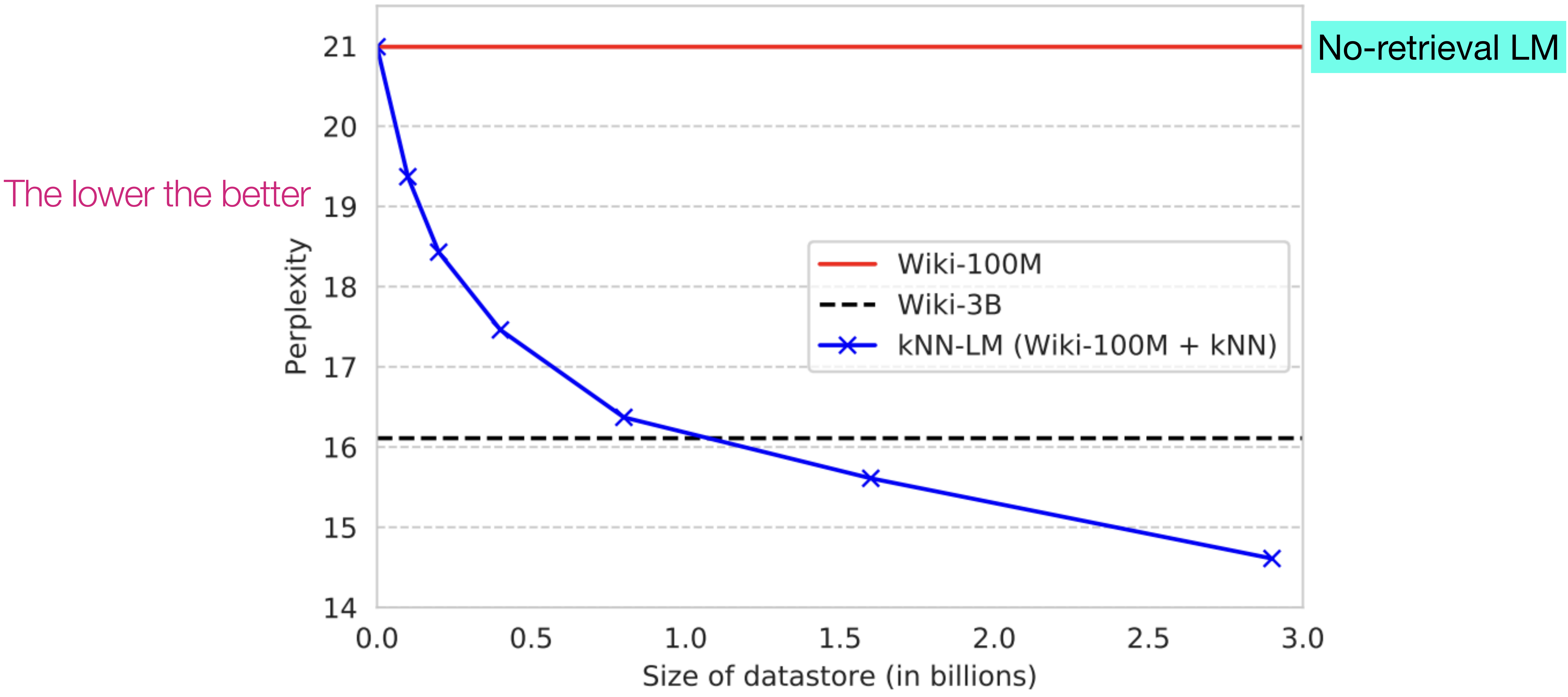


kNN-LM - results

The lower the better

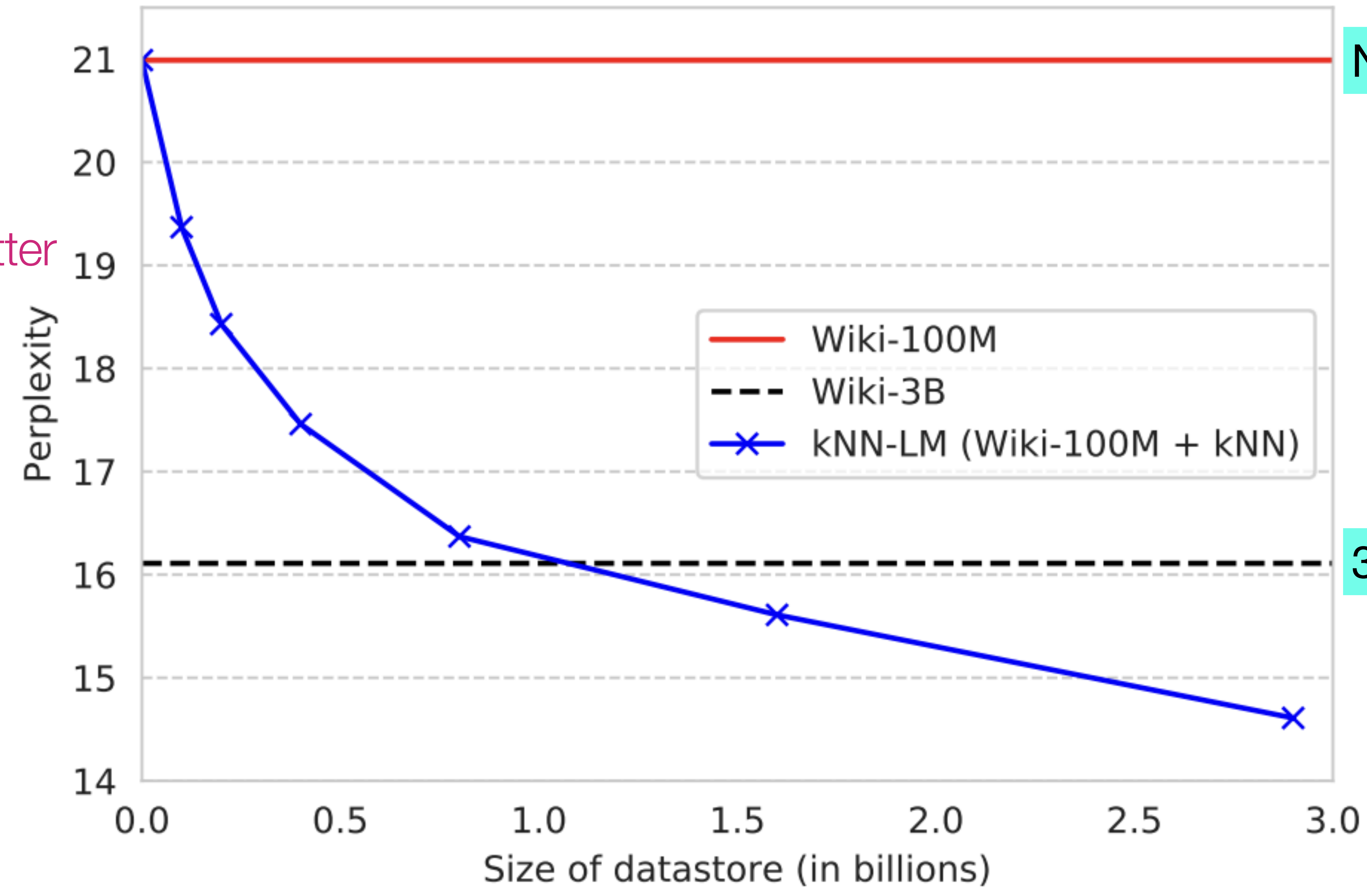


kNN-LM - results



kNN-LM - results

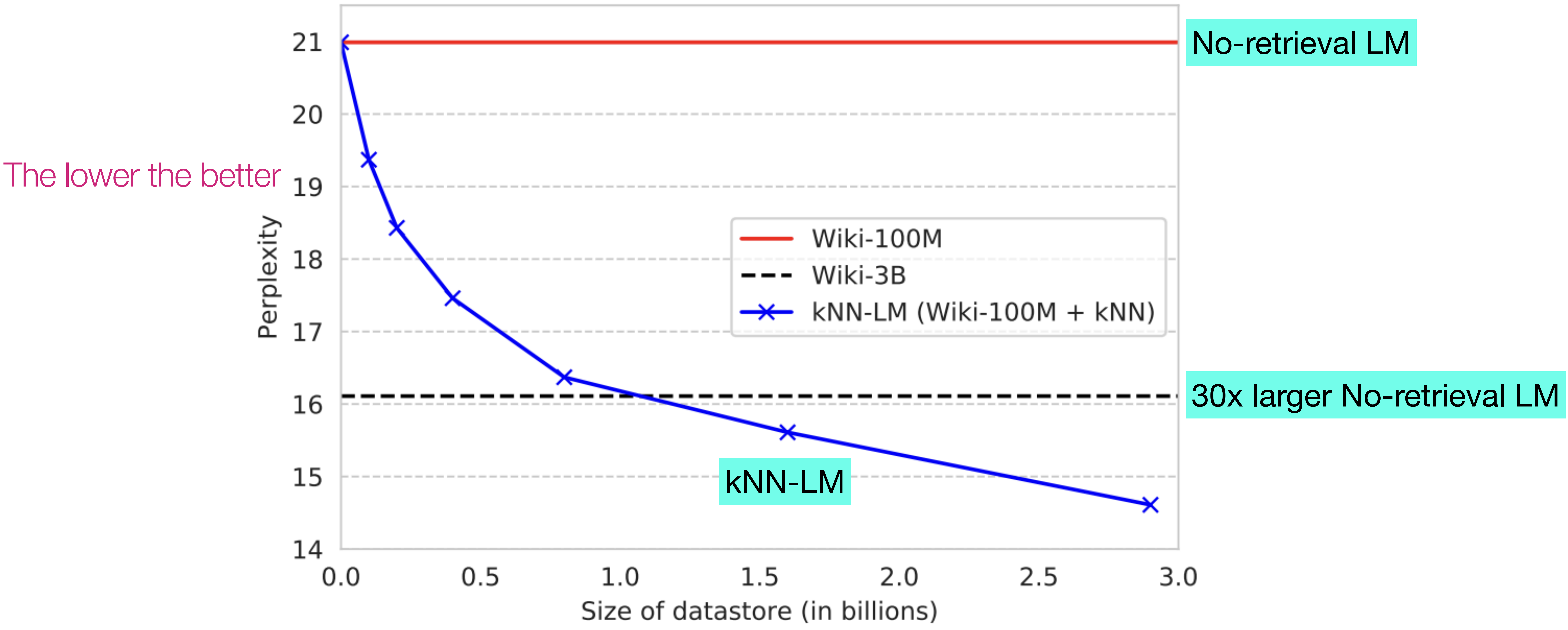
The lower the better



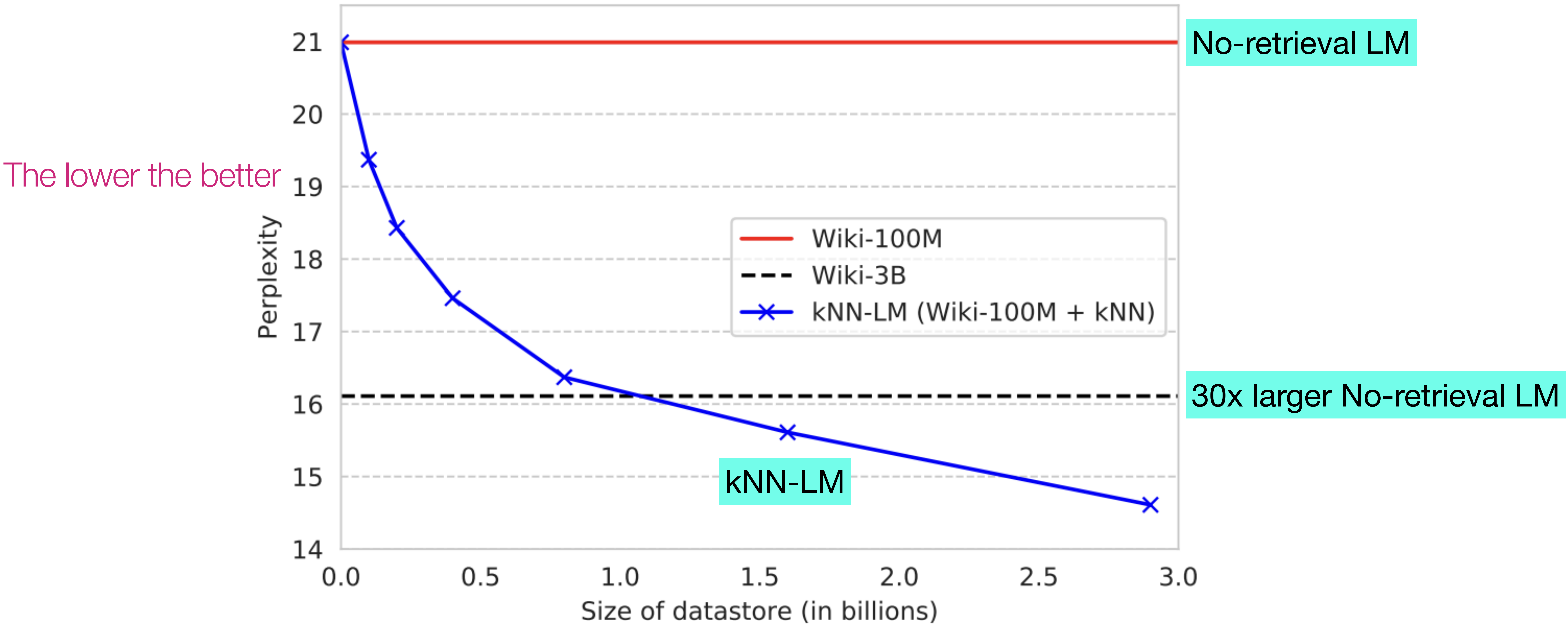
No-retrieval LM

30x larger No-retrieval LM

kNN-LM - results

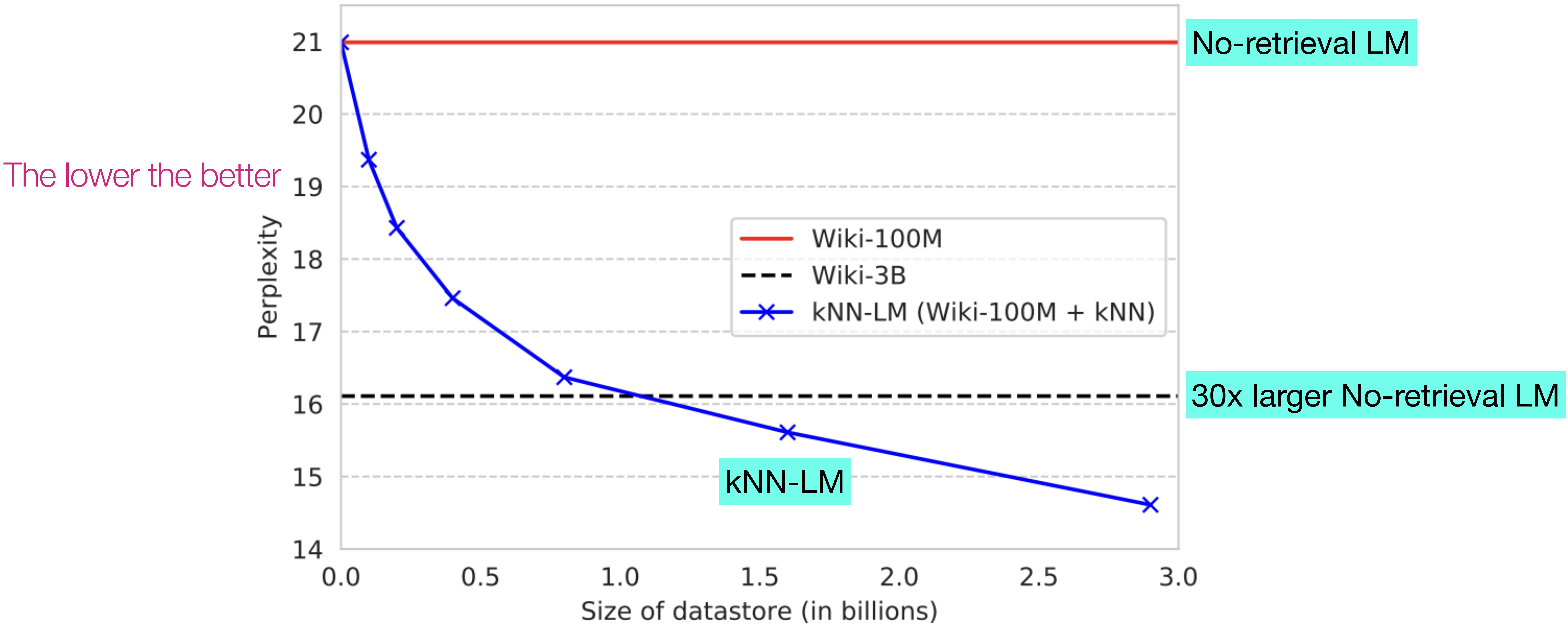


kNN-LM - results



Outperforms no-retrieval LM

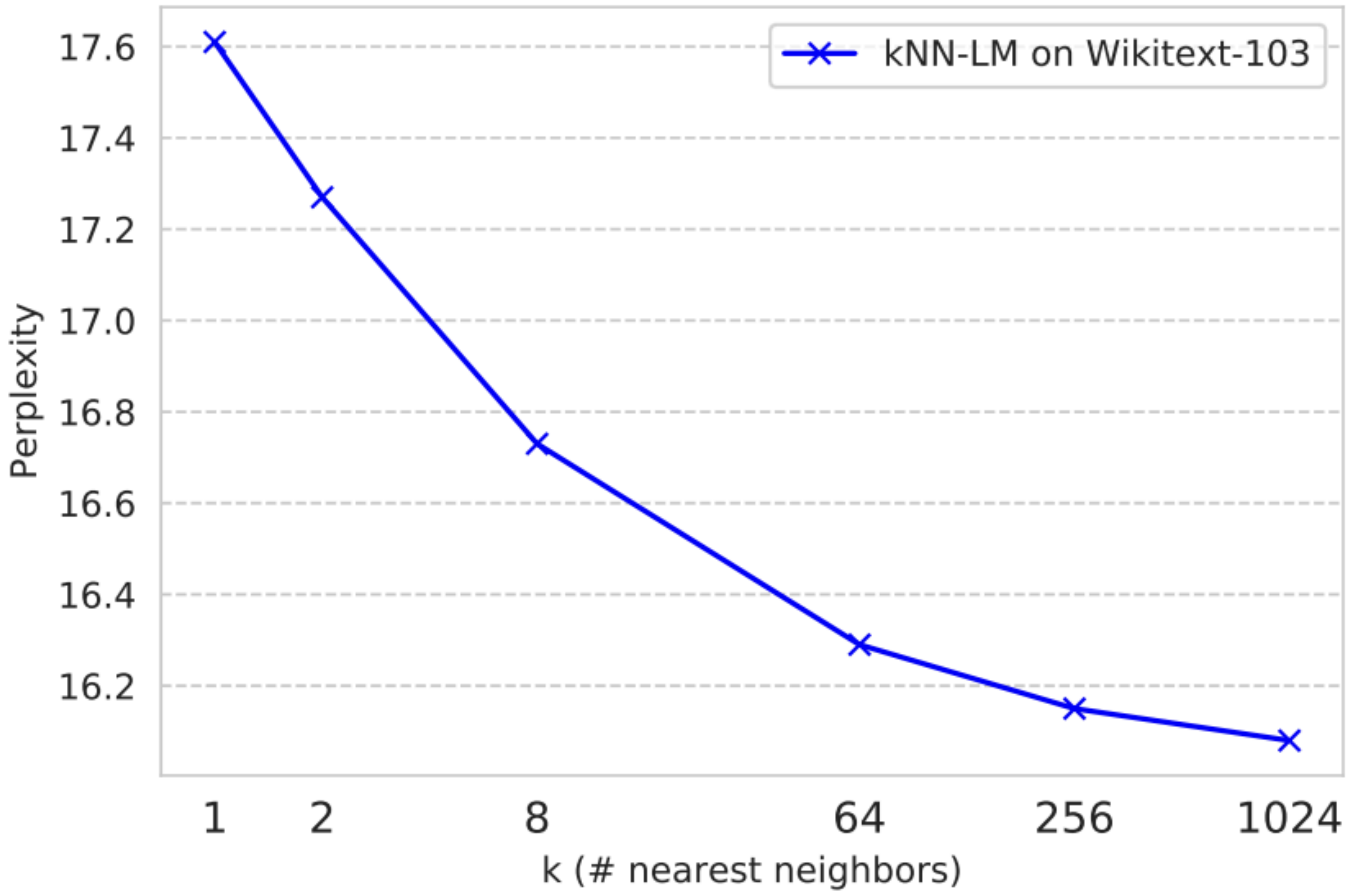
kNN-LM - results



Outperforms no-retrieval LM

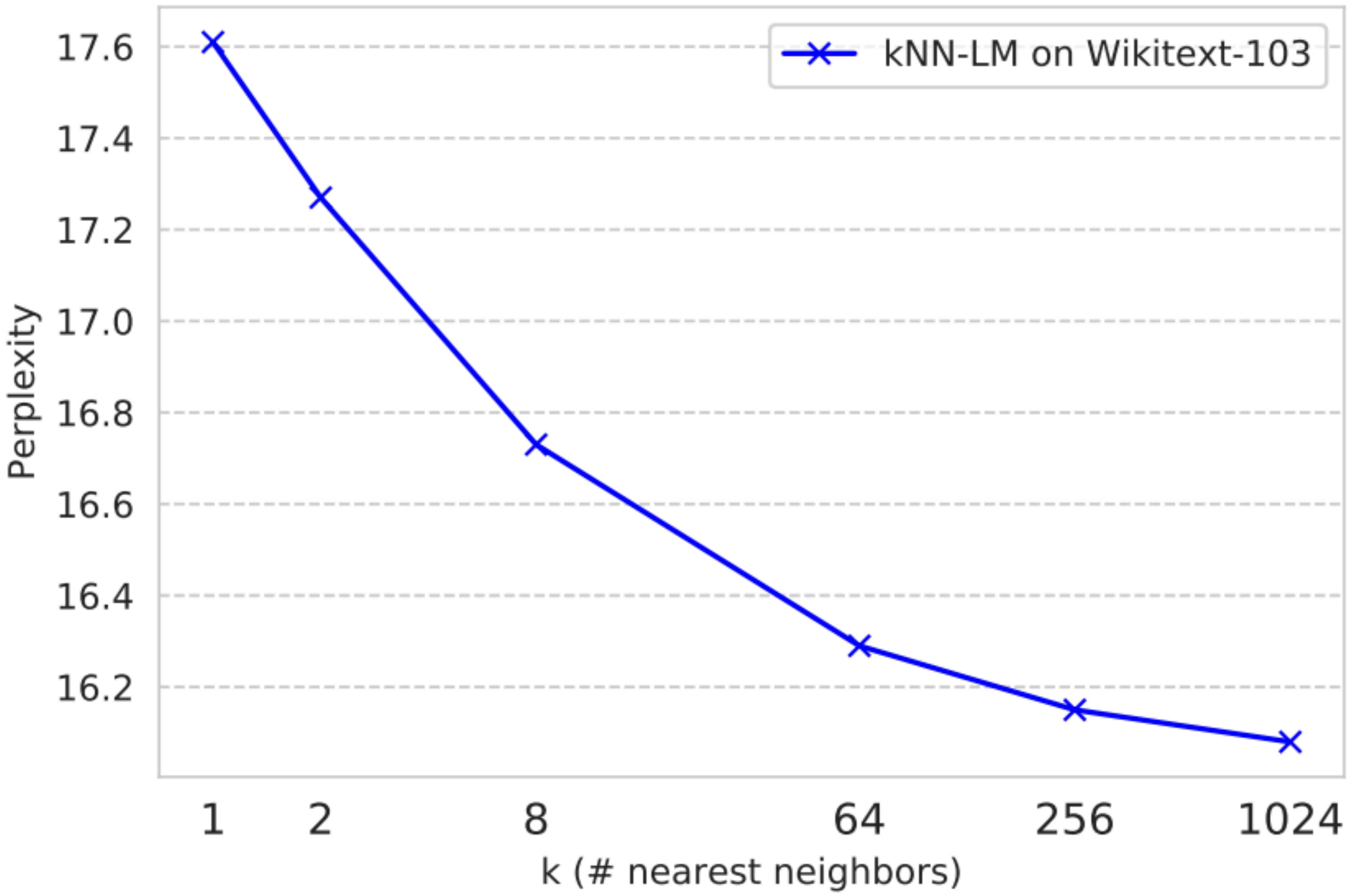
Better with bigger datastore

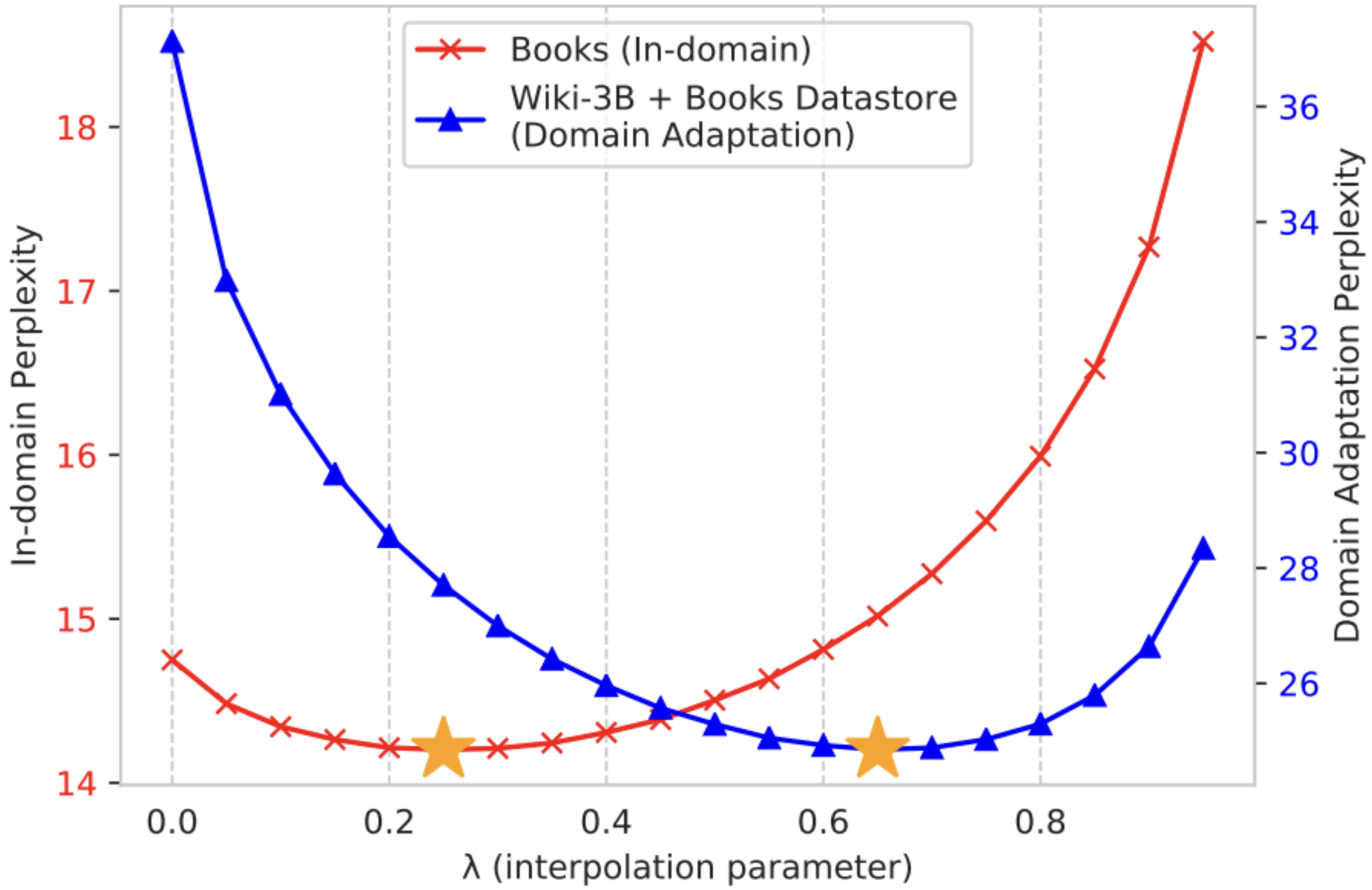
kNN-LM - results



Better with bigger ***k***

kNN-LM - results



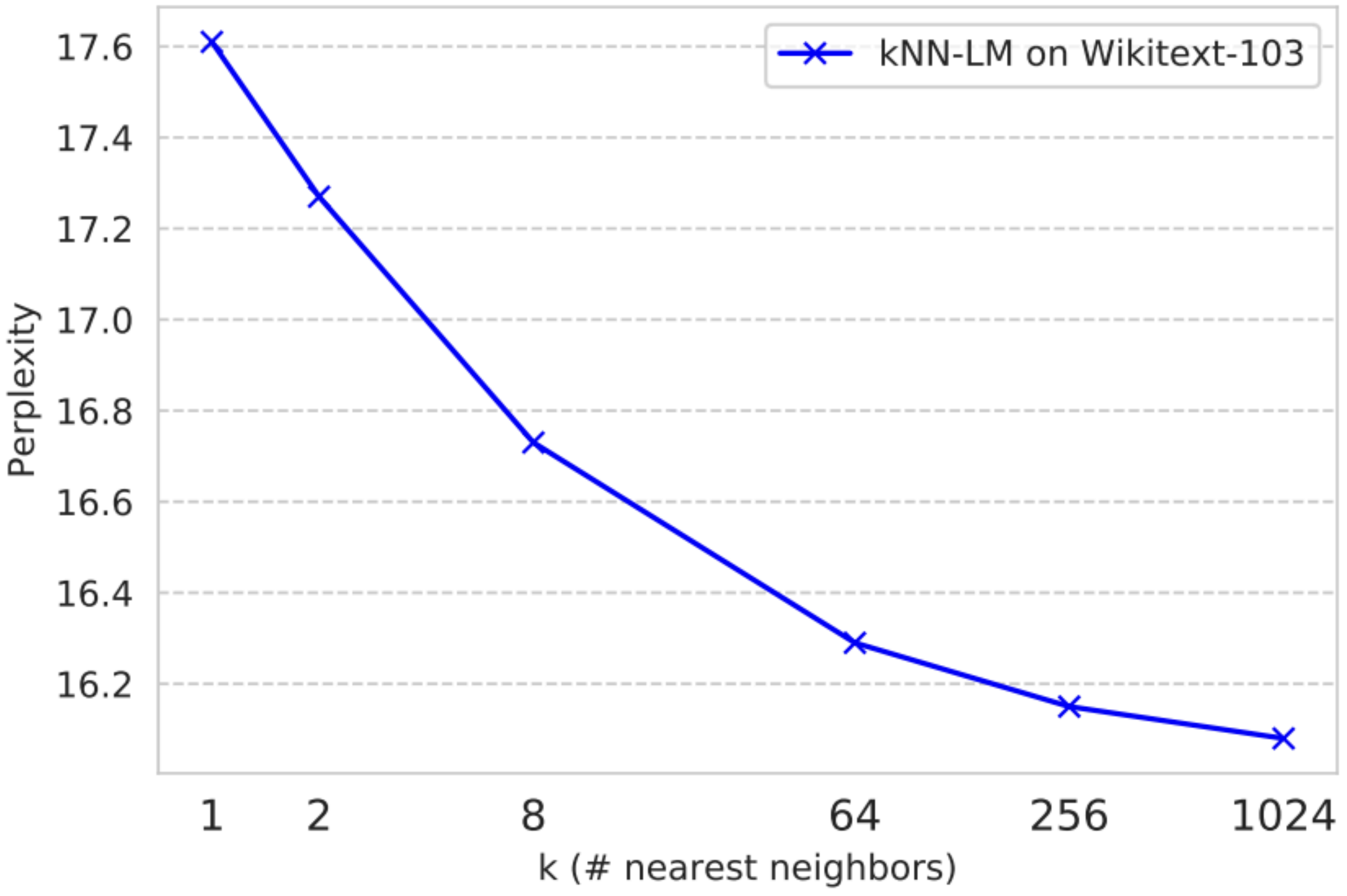


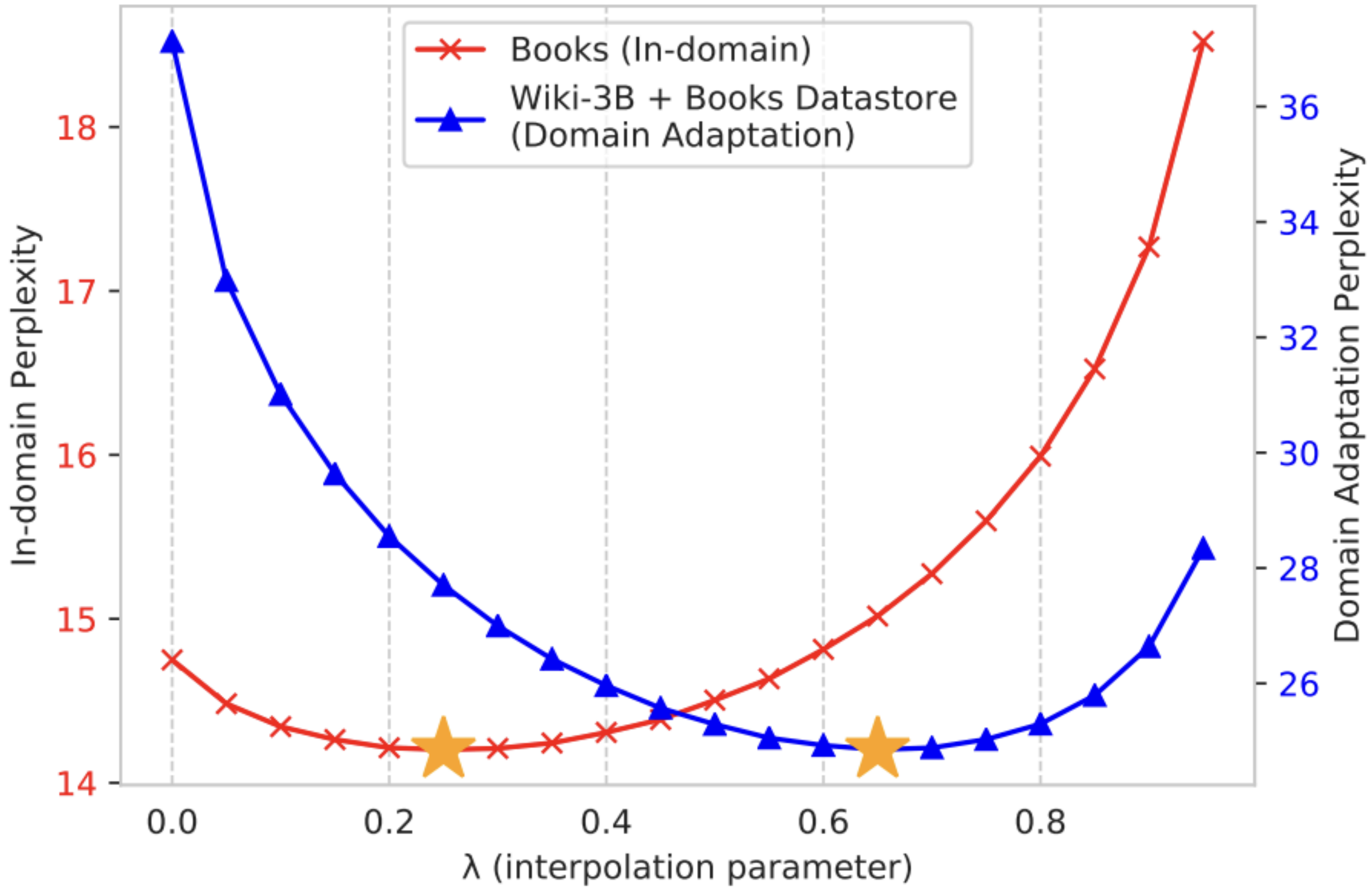
Better with bigger ***k***

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

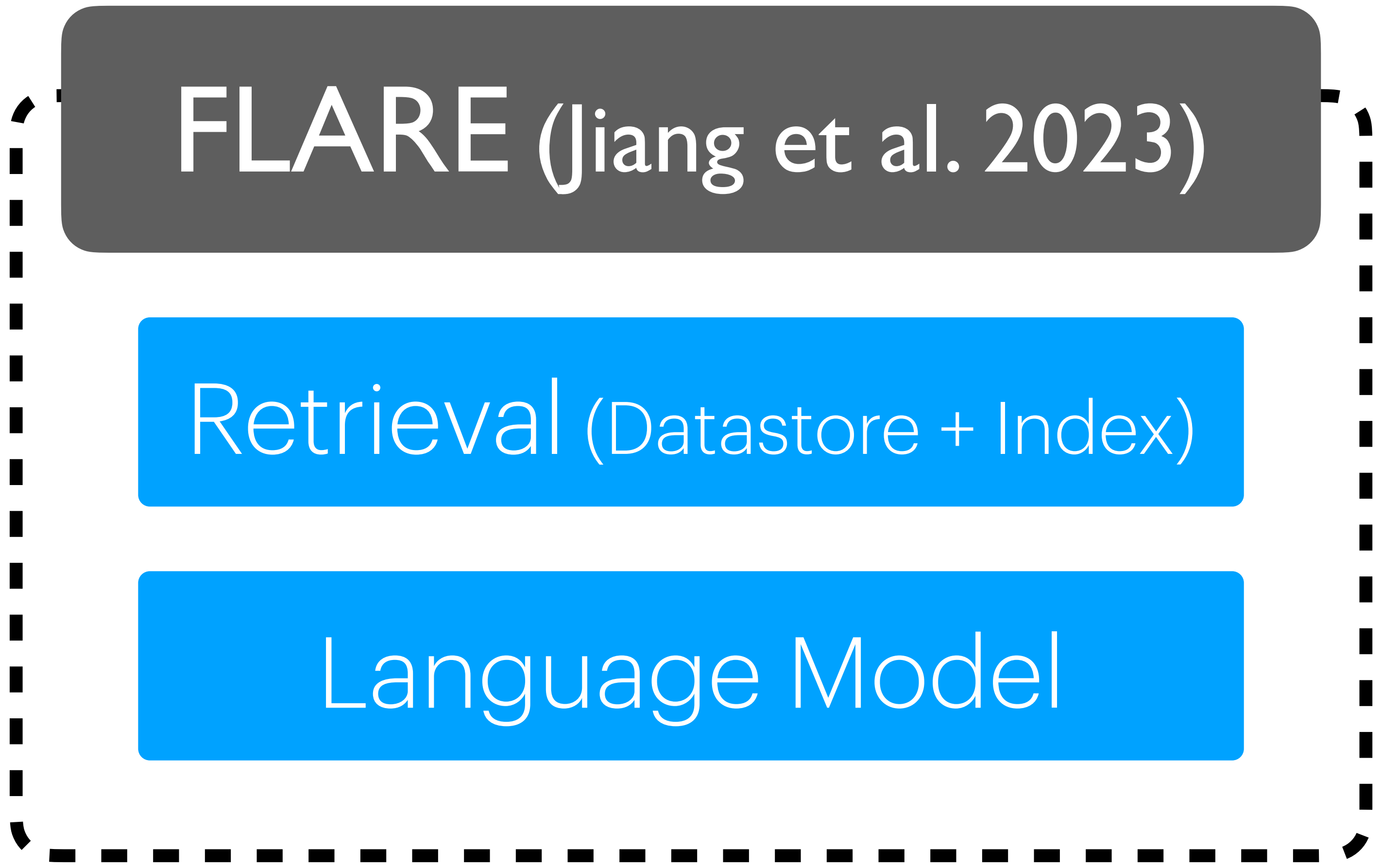
Adaptive retrieval of text chunks
(following retrieve-in-context)

Adaptive retrieval of tokens
(following kNN-LM)

Adaptive retrieval of *chunks*

- *Judge necessity*

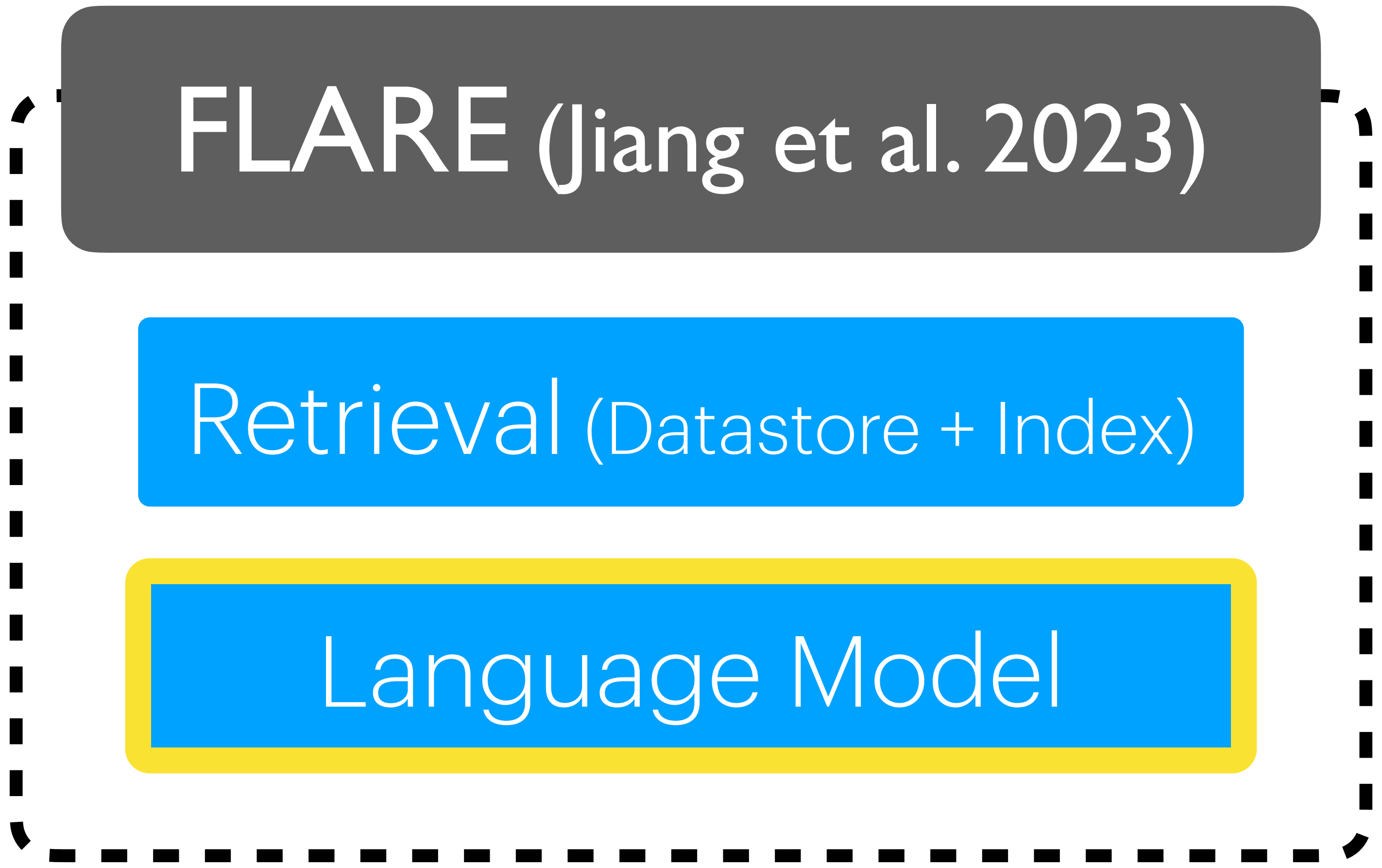
Input: Generate a summary about Joe Biden.



Jiang et al. “Active Retrieval Augmented Generation”

Adaptive retrieval of *chunks*

- *Judge necessity*



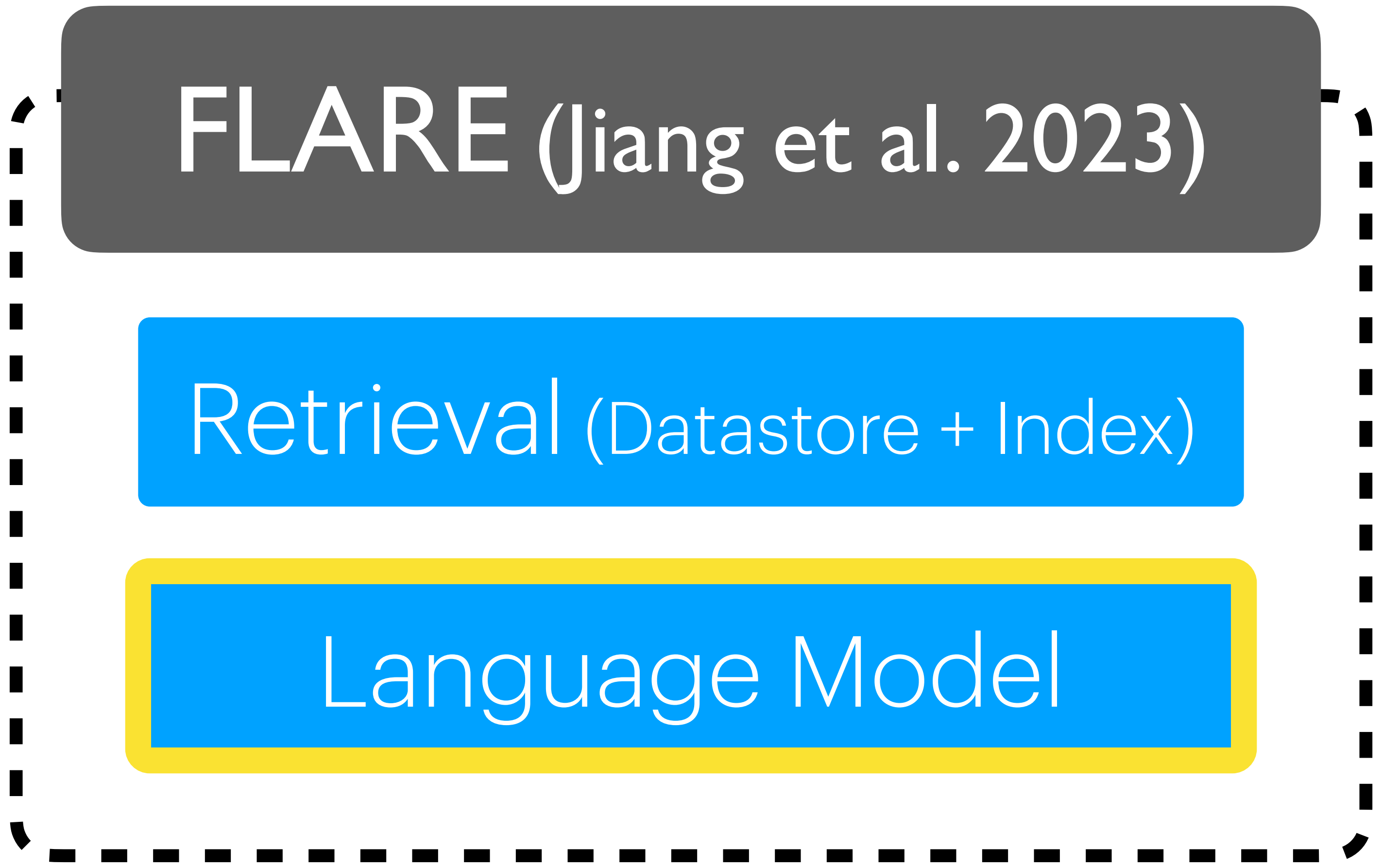
Input: Generate a summary about Joe Biden.

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

Jiang et al. "Active Retrieval Augmented Generation"

Adaptive retrieval of *chunks*

- *Judge necessity*



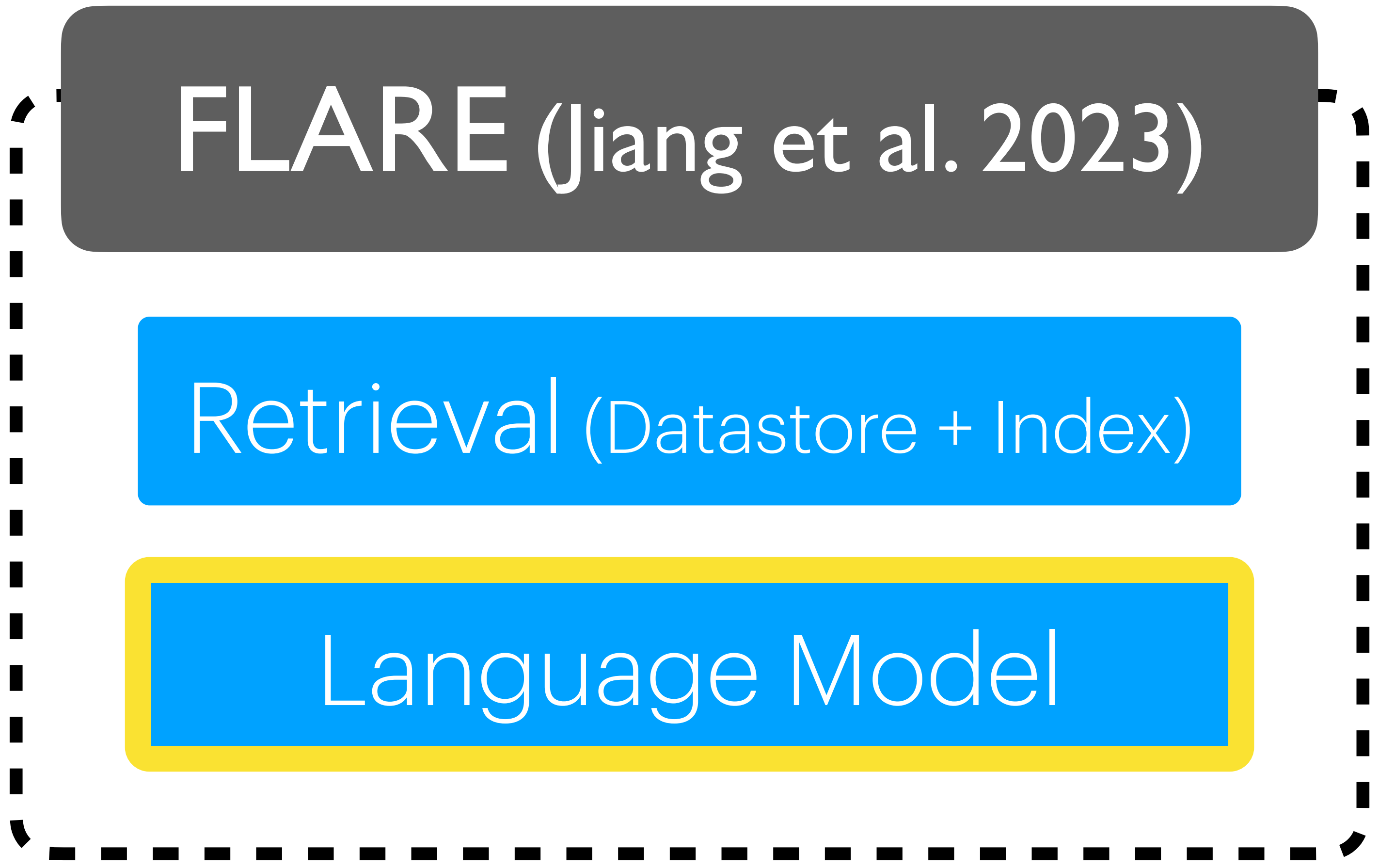
Input: Generate a summary about Joe Biden.

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

Jiang et al. "Active Retrieval Augmented Generation"

Adaptive retrieval of *chunks*

- *Judge necessity*



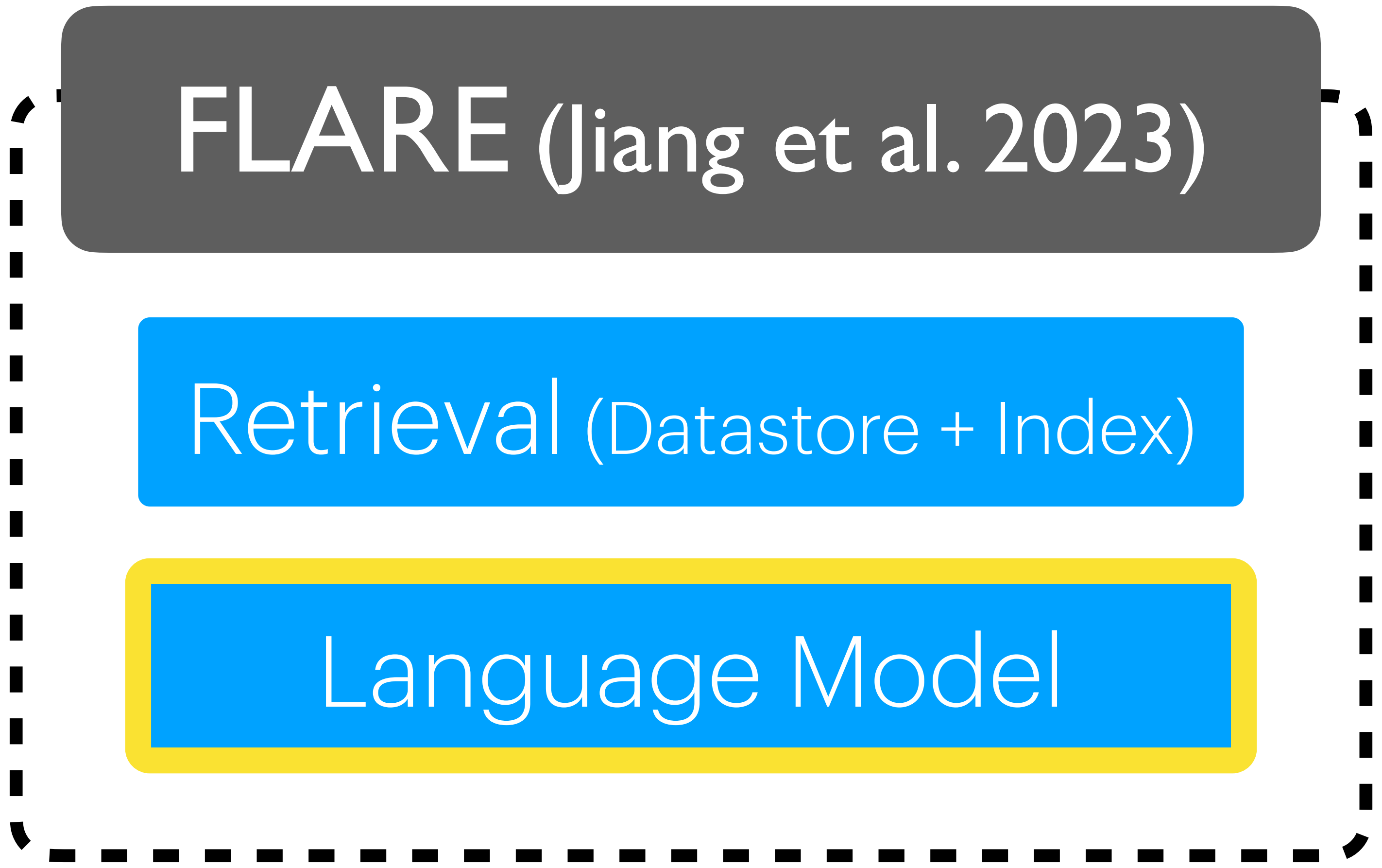
Input: Generate a summary about Joe Biden.

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

Jiang et al. "Active Retrieval Augmented Generation"

Adaptive retrieval of *chunks*

- *Judge necessity*



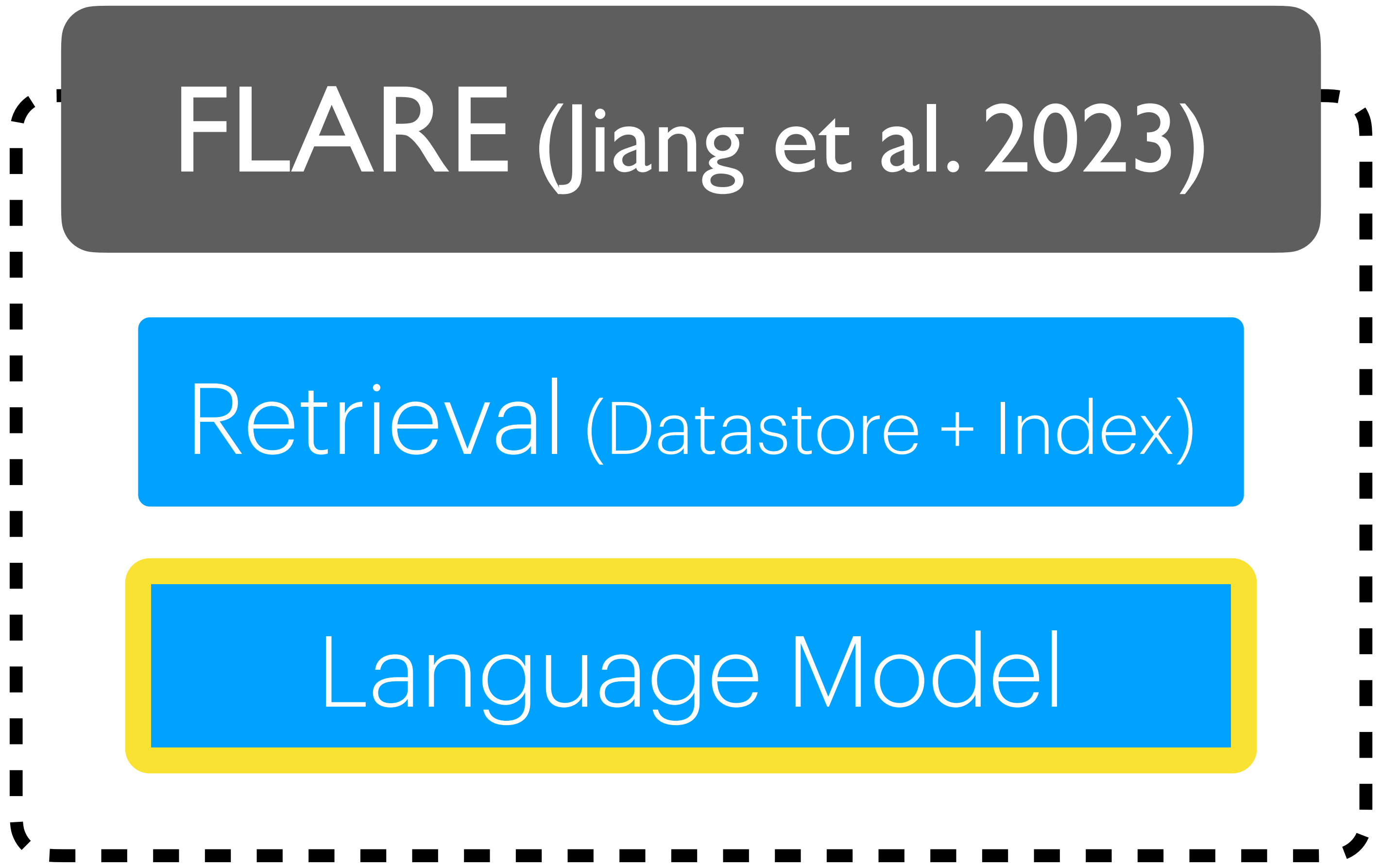
Input: Generate a summary about Joe Biden.

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

Jiang et al. “Active Retrieval Augmented Generation”

Adaptive retrieval of *chunks*

- *Judge necessity*



Input: Generate a summary about Joe Biden.

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

Jiang et al. “Active Retrieval Augmented Generation”

Adaptive retrieval of *chunks*

- *Judge necessity*

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

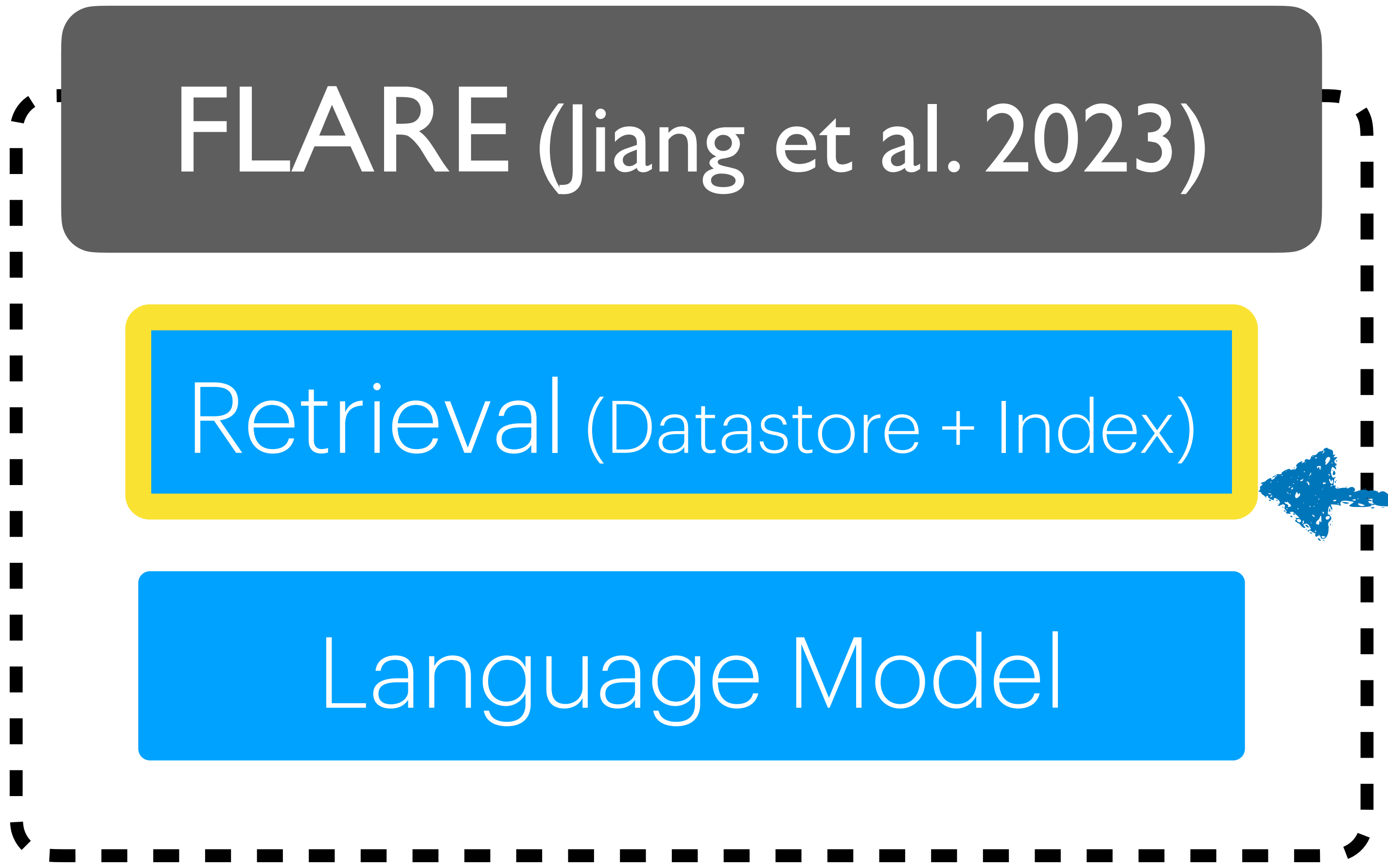
Input: Generate a summary about Joe Biden.

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

Jiang et al. "Active Retrieval Augmented Generation"

Adaptive retrieval of *chunks*

- *Judge necessity*



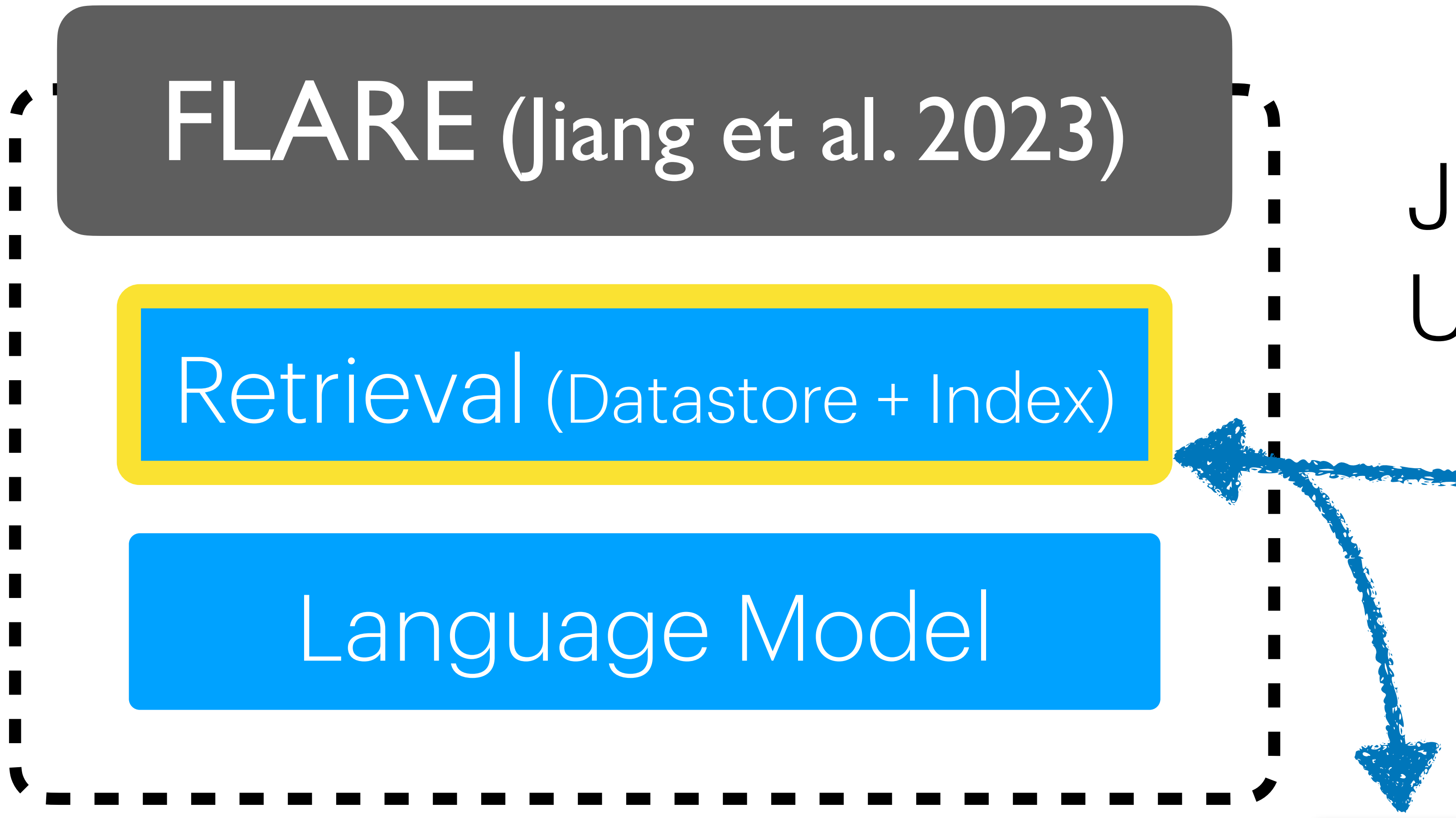
Input: Generate a summary about Joe Biden.

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

Jiang et al. “Active Retrieval Augmented Generation”

Adaptive retrieval of *chunks*

- *Judge necessity*



Input: Generate a summary about Joe Biden.

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

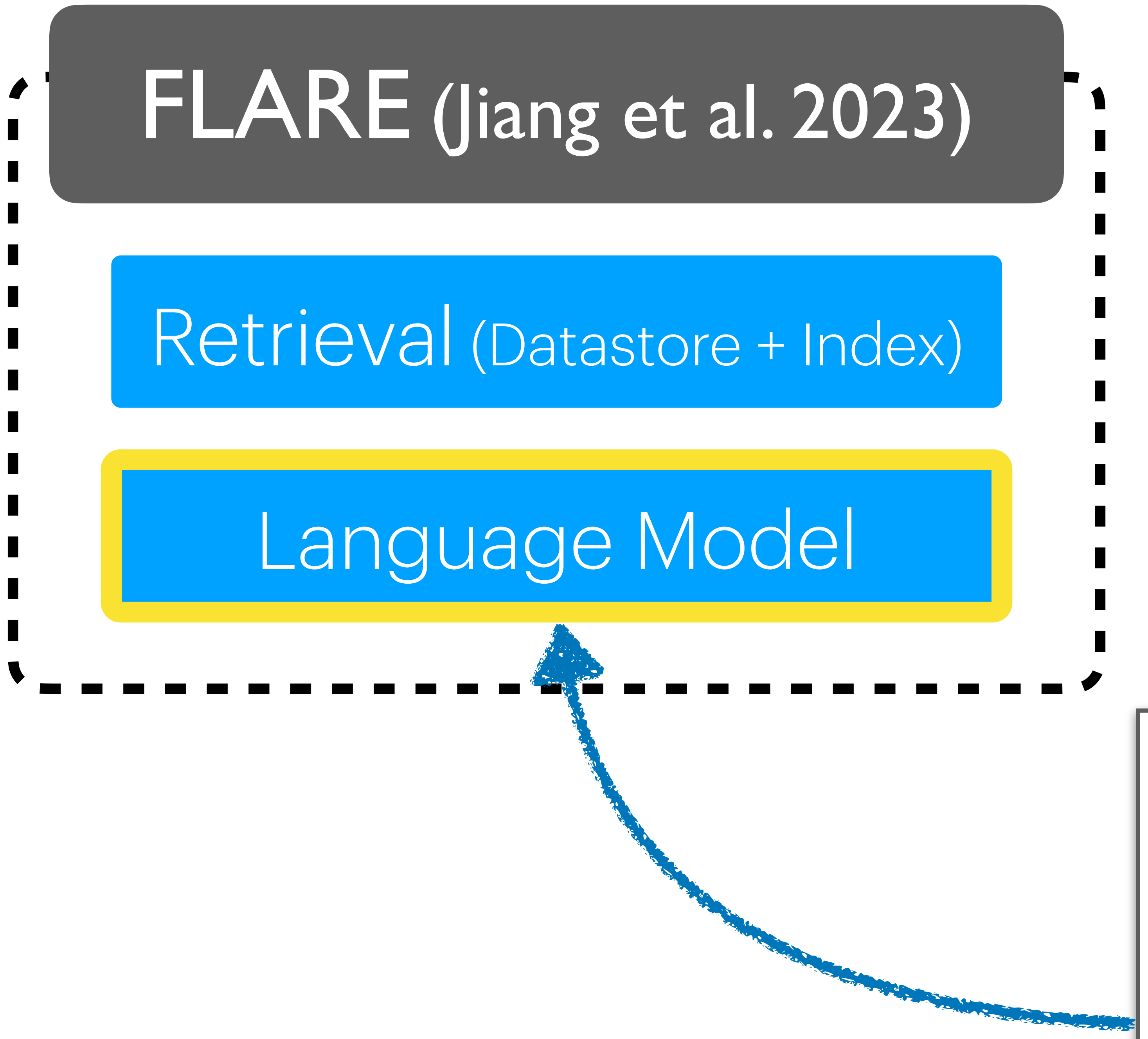
Joe Biden
At the University of Delaware in Newark, Biden ... earned a Bachelor of Arts degree in 1965 with a double major in history and political science.

Jiang et al. "Active Retrieval Augmented Generation"

Adaptive retrieval of *chunks*

- *Judge necessity*

Input: Generate a summary about Joe Biden.



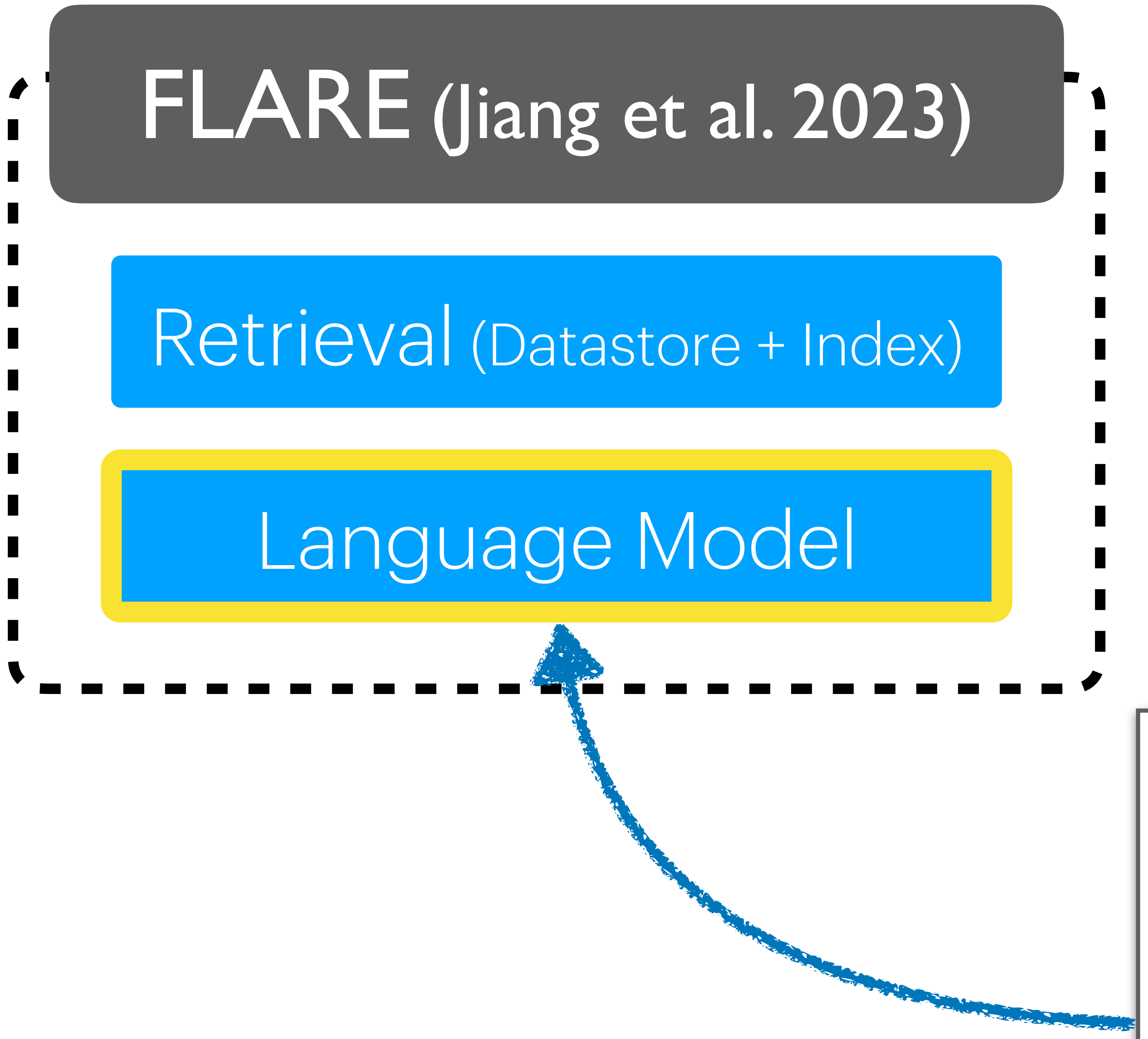
Joe Biden (born November 20, 1942) is the 46th president of the United States. ~~Joe Biden attended **[mask]**, where he earned **[mask]**.~~

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.

Jiang et al. "Active Retrieval Augmented Generation"

Adaptive retrieval of *chunks*

- *Judge necessity*



Input: Generate a summary about Joe Biden.

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

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.

Jiang et al. "Active Retrieval Augmented Generation"

Adaptive retrieval of *tokens*

- Judge necessity

Adaptive retrieval of *tokens*

- *Judge necessity*

Joe Biden graduated from the University of Delaware .

Adaptive retrieval of *tokens*

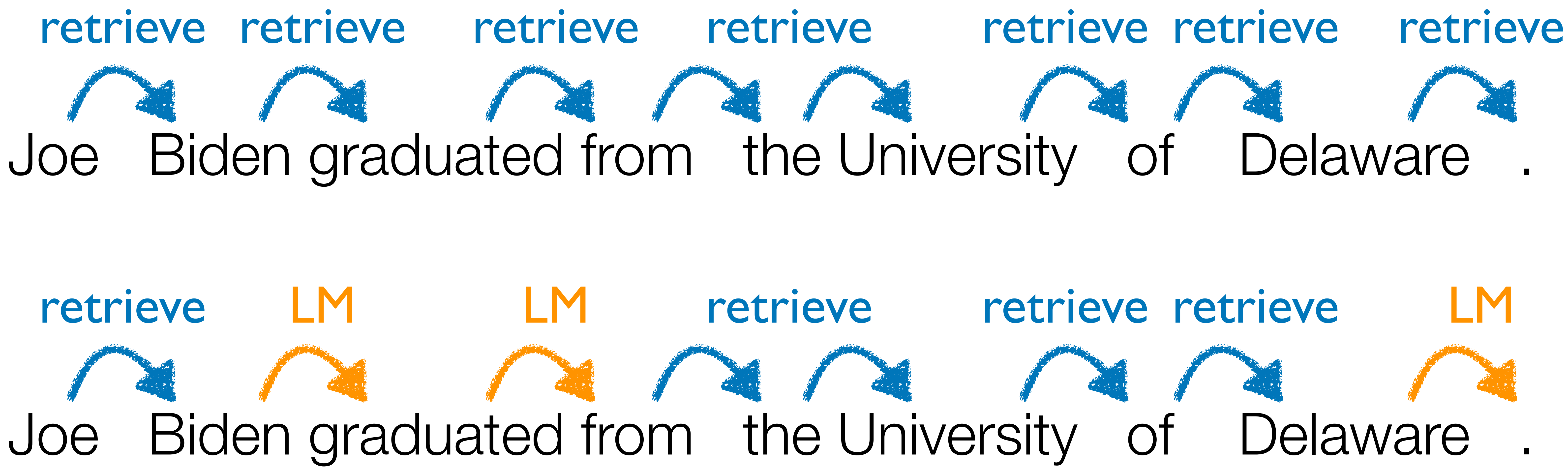
- Judge necessity

retrieve retrieve retrieve retrieve retrieve retrieve retrieve

Joe Biden graduated from the University of Delaware .

Adaptive retrieval of *tokens*

- *Judge necessity*



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

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 .

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

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

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 .

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

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

$\rightarrow \ \lambda = 0 \quad \textit{if} \ \lambda < \gamma$

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

Adaptive retrieval of *tokens*

- Use local info

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

Adaptive retrieval of *tokens*

- Use local info

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

Joe Biden graduated from

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

Adaptive retrieval of *tokens*

- Use local info

Training contexts	Targets
At	*the*
At the	*University*
At the Universty	*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

Alon et al. 2022. “Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval”

Adaptive retrieval of *tokens*

- Use local info

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

retrieve retrieve

Joe Biden graduated from the University

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

Adaptive retrieval of *tokens*

- Use local info

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

retrieve retrieve retrieve

Joe Biden graduated from the University of

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

Adaptive retrieval of *tokens*

- Use local info

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

retrieve retrieve retrieve retrieve

Joe Biden graduated from the University of Delaware.

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

Adaptive retrieval of *tokens*

- Use local info

Training contexts	Targets
At	*the*
At the	*University*
At the Universty	*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

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

Adaptive retrieval of *tokens*

- *Use local info*

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

pointer

retrieve pointer

Joe Biden graduated from the University

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

Adaptive retrieval of *tokens*

- Use local info

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

pointer

retrieve pointer pointer

Joe Biden graduated from the University of

Alon et al. 2022. “Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval”

Adaptive retrieval of *tokens*

- Use local info

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

pointer

retrieve pointer pointer pointer

Joe Biden graduated from the University of Delaware.

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

Adaptive retrieval of *tokens*

- Use local info

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

pointer

retrieve pointer pointer pointer

Joe Biden graduated from the University of Delaware.

Retrieve once, and save other searches!

Alon et al. 2022. "Neuro-Symbolic Language Modeling with Automaton-augmented Retrieval"

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

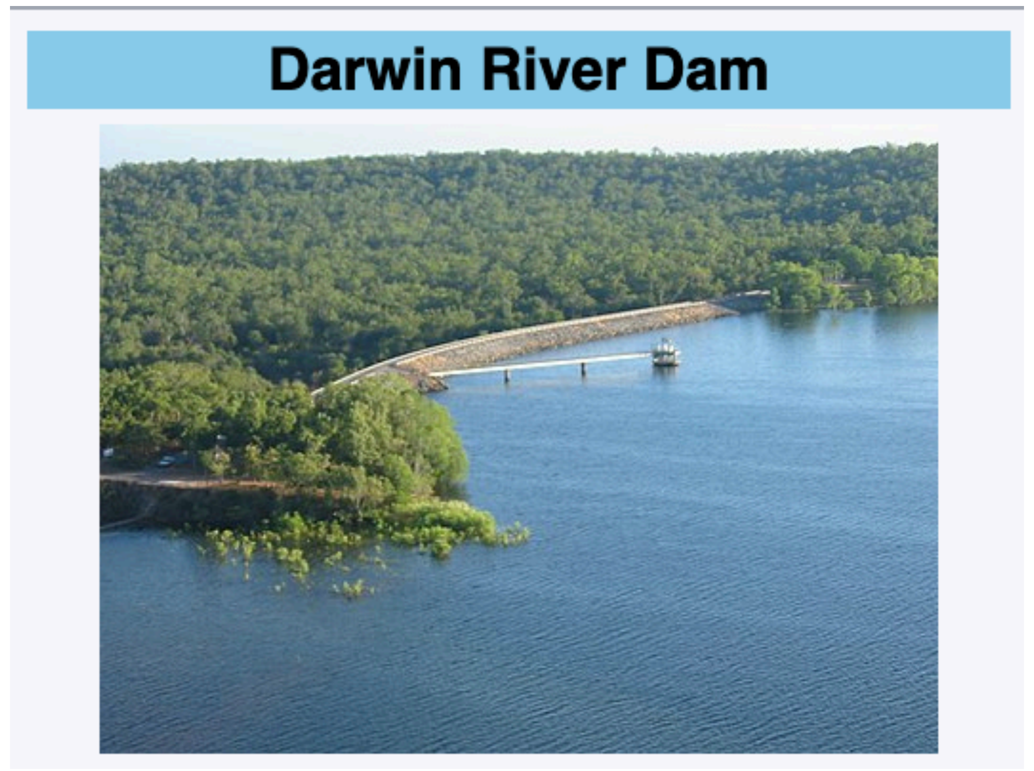
	What do retrieve?	How to use retrieval?	When to retrieve?
REALM (Guu et al 2020)	Text chunks	Input layer	Once
Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)	Text chunks	Input layer	Every n tokens
RETRO (Borgeaud et al. 2021)	Text chunks	Intermediate layers	Every n tokens
kNN-LM (Khandelwal et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

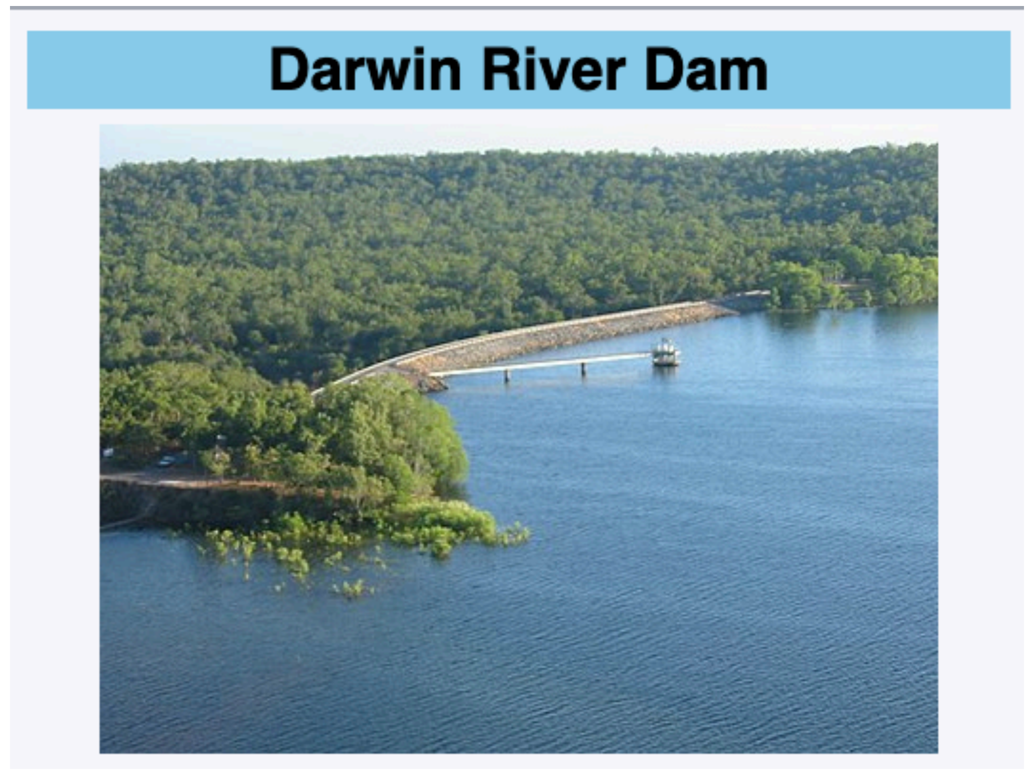
Entities as Experts (Fevry et al. 2020)

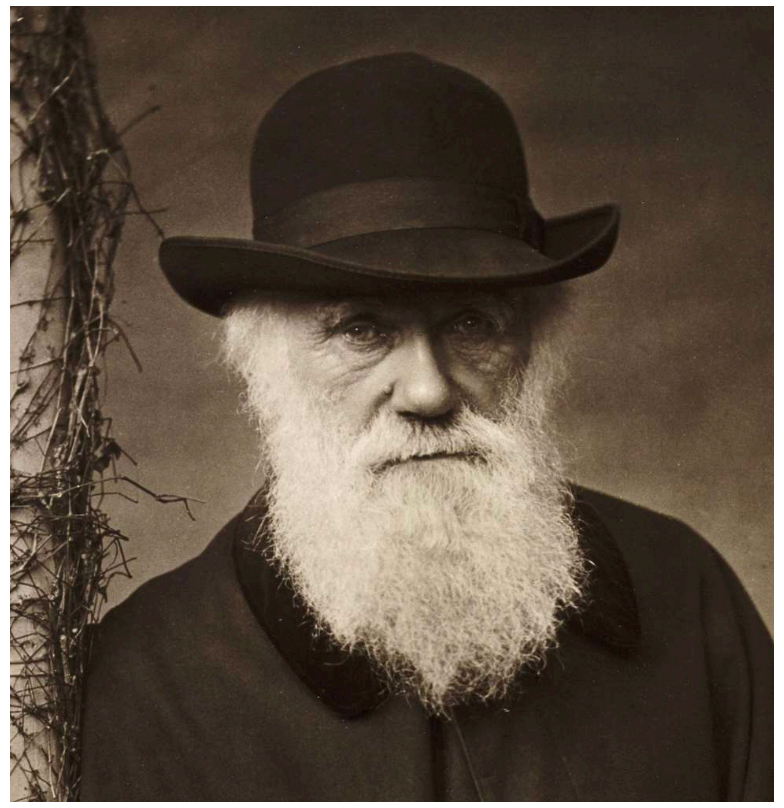
Entities as Experts (Fevry et al. 2020)

Entities as Experts (Fevry et al. 2020)

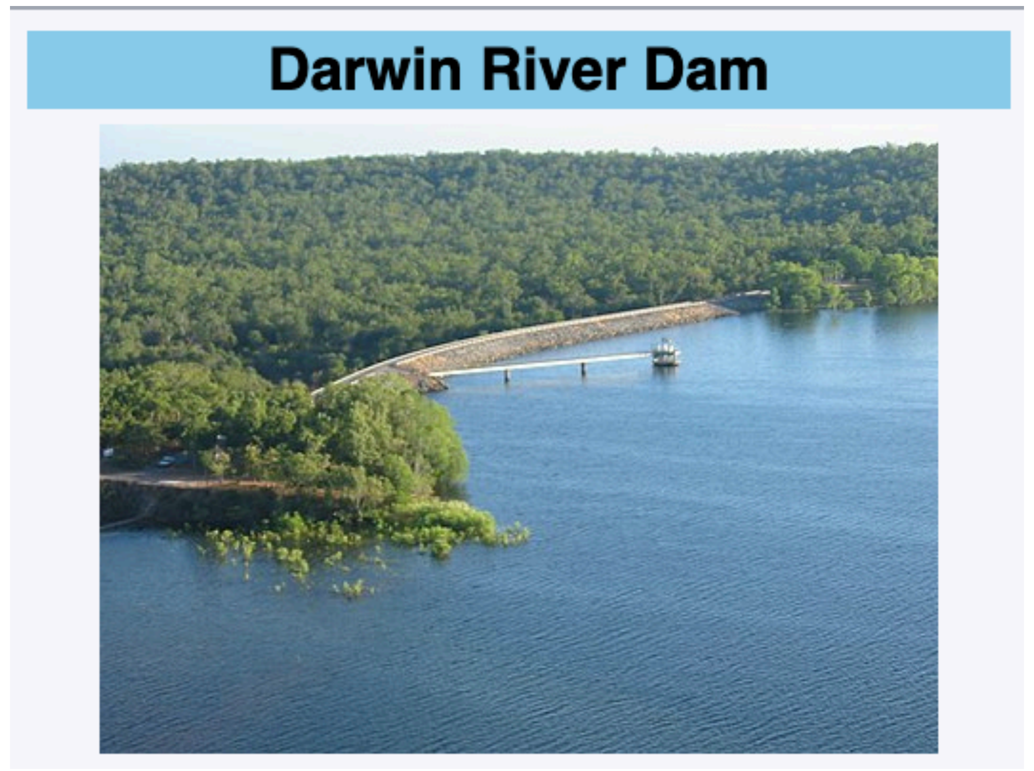


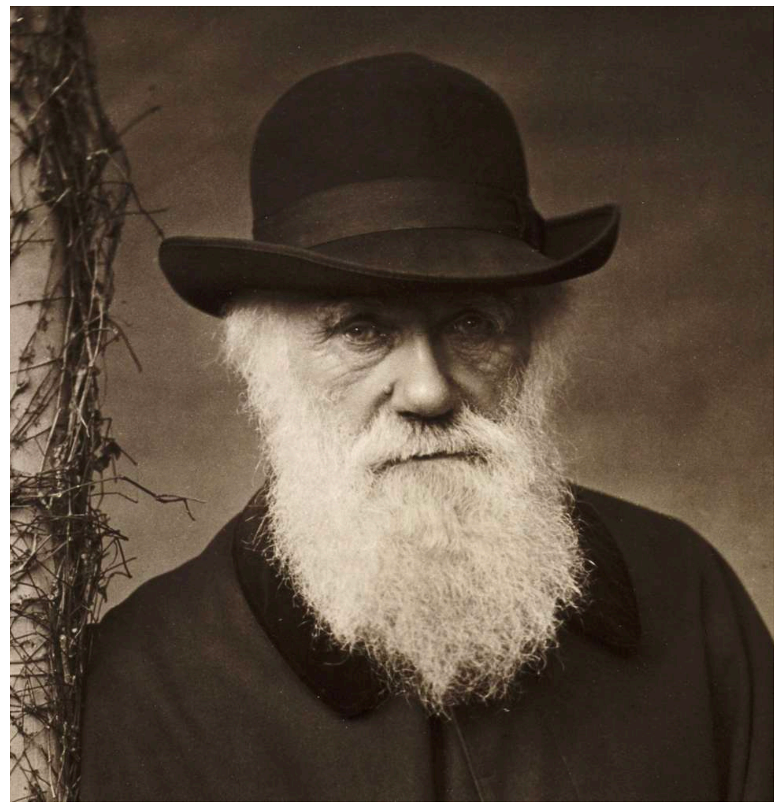
Entities as Experts (Fevry et al. 2020)

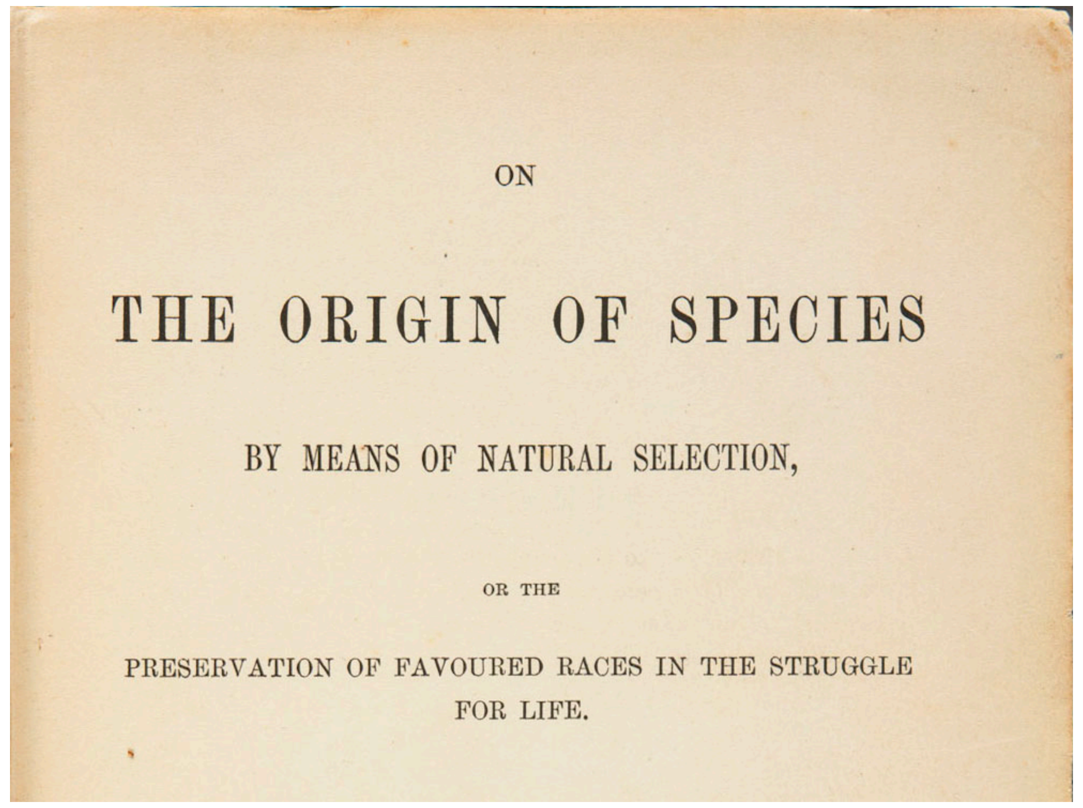




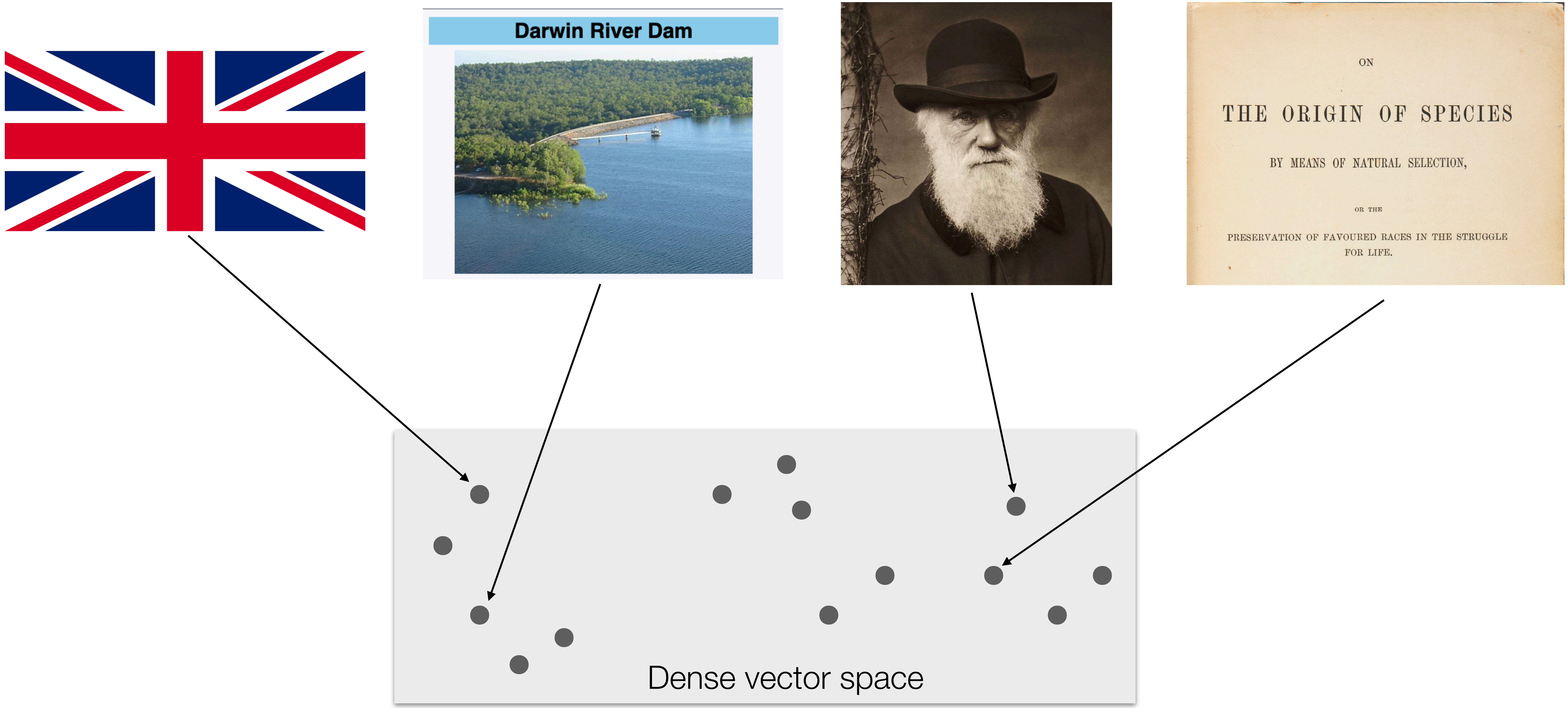
Entities as Experts (Fevry et al. 2020)



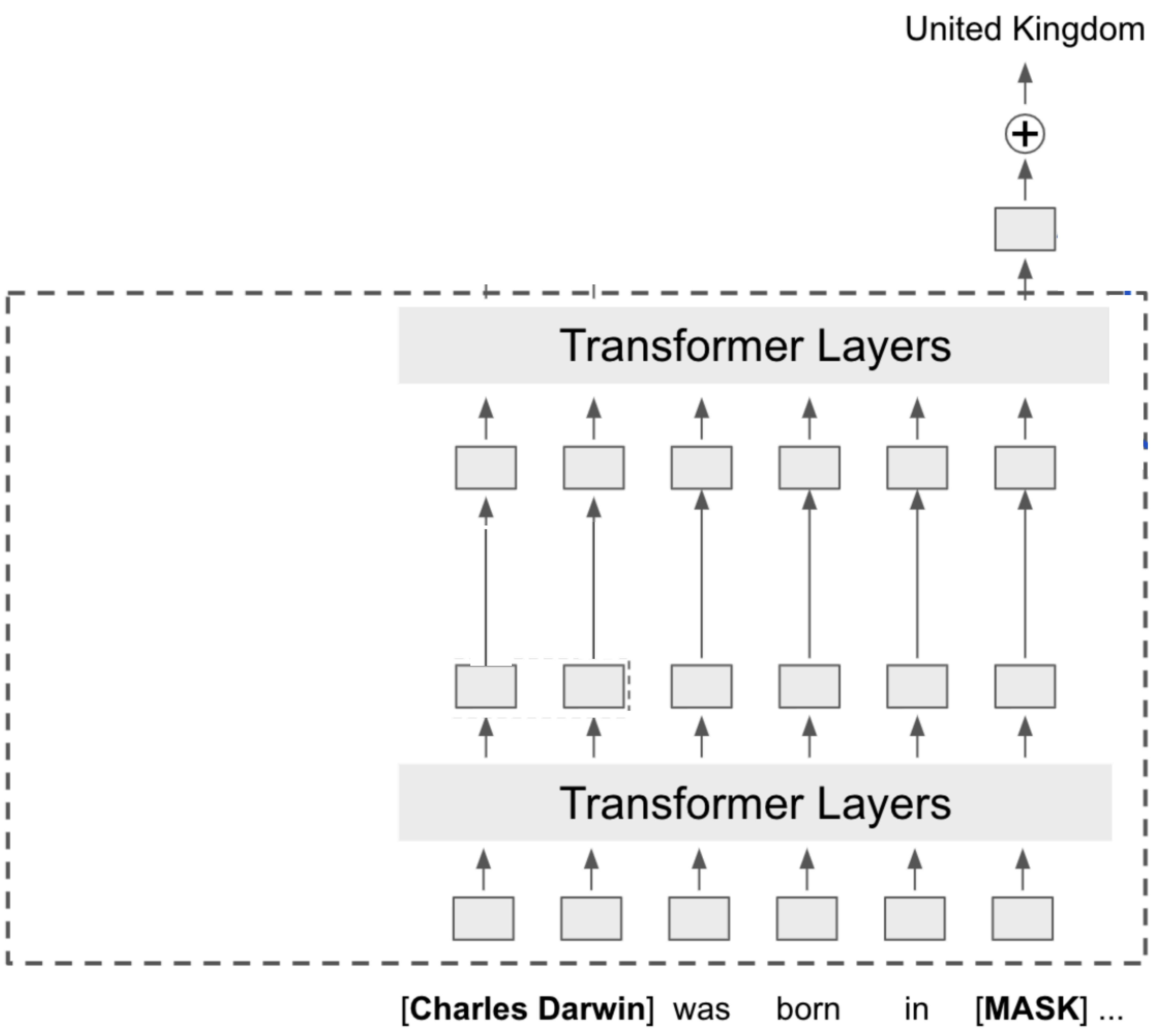




Entities as Experts (Fevry et al. 2020)

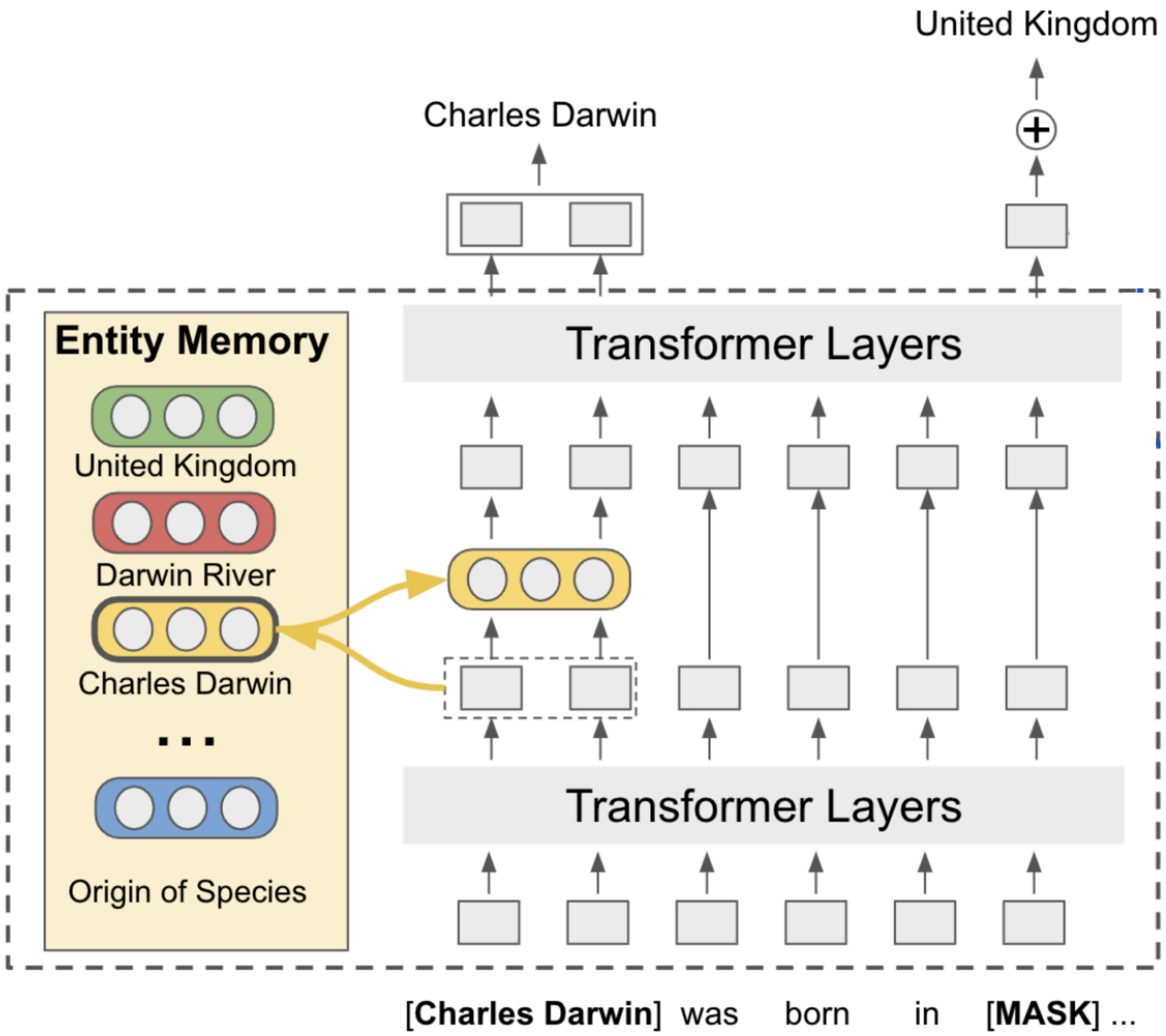


Entities as Experts (Fevry et al. 2020)



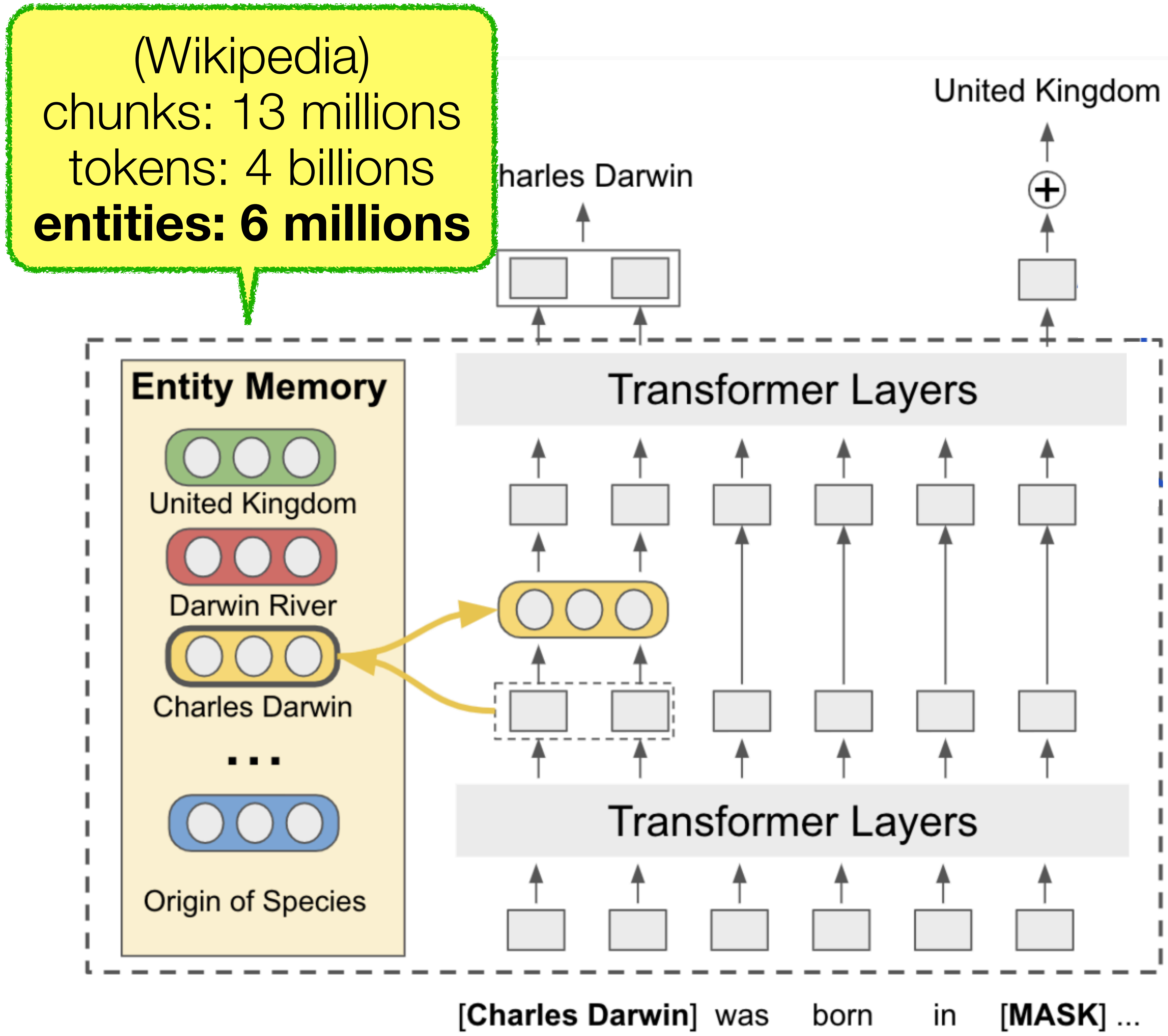
Fevry et al. 2020. "Entities as Experts: Sparse Memory Access with Entity Supervision"

Entities as Experts (Fevry et al. 2020)



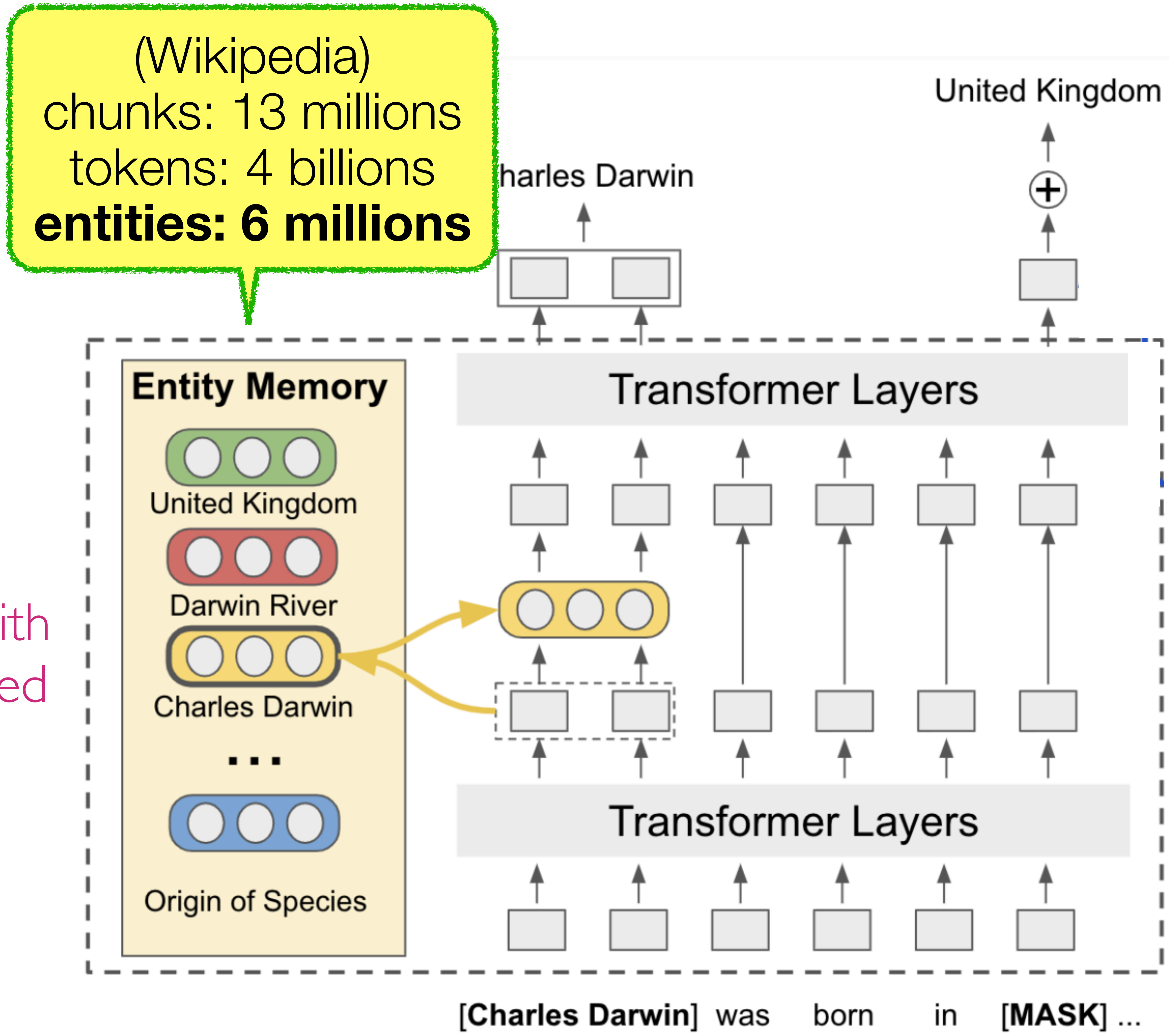
Fevry et al. 2020. “Entities as Experts: Sparse Memory Access with Entity Supervision”

Entities as Experts (Fevry et al. 2020)



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

Entities as Experts (Fevry et al. 2020)



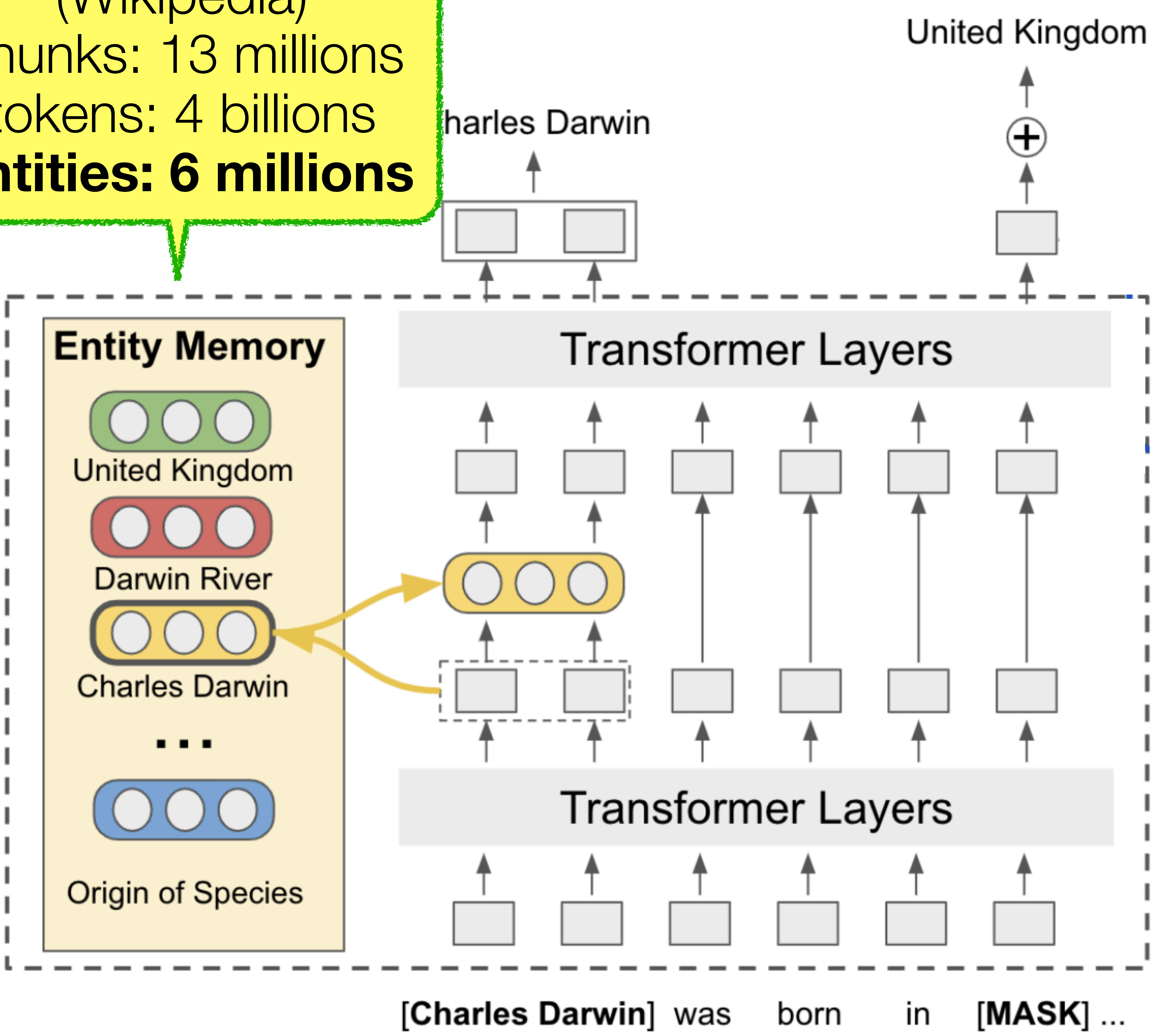
Need text with entity detected

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

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"

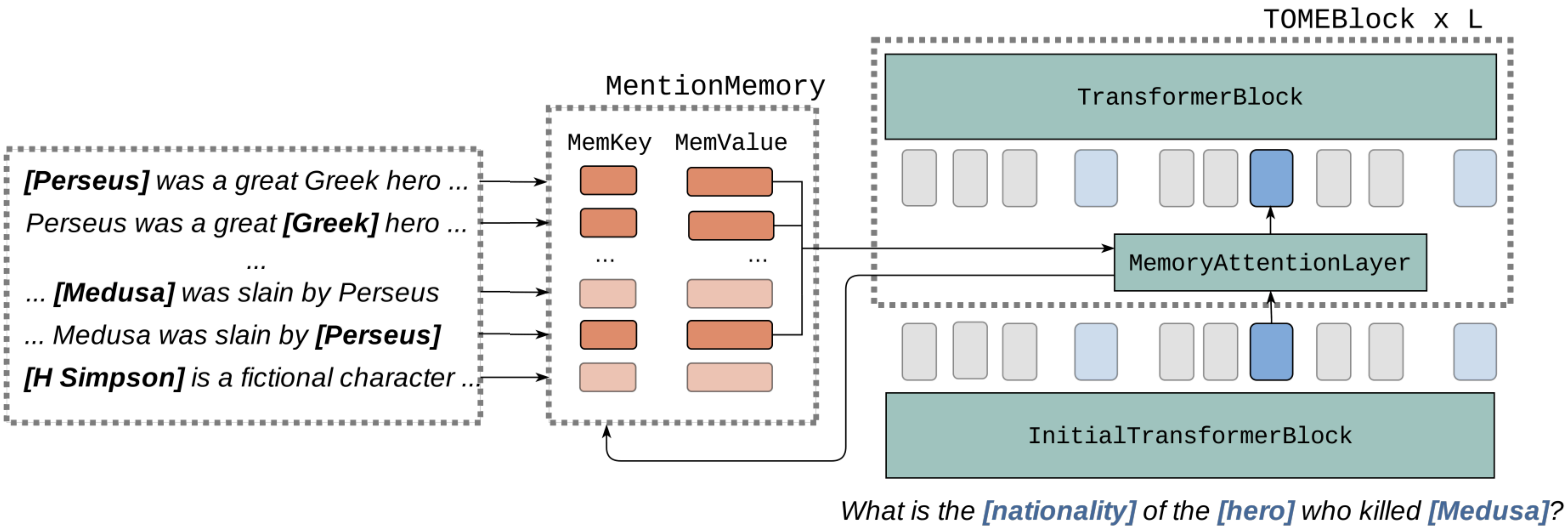
Mention Memory (de Jong et al. 2022)

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)

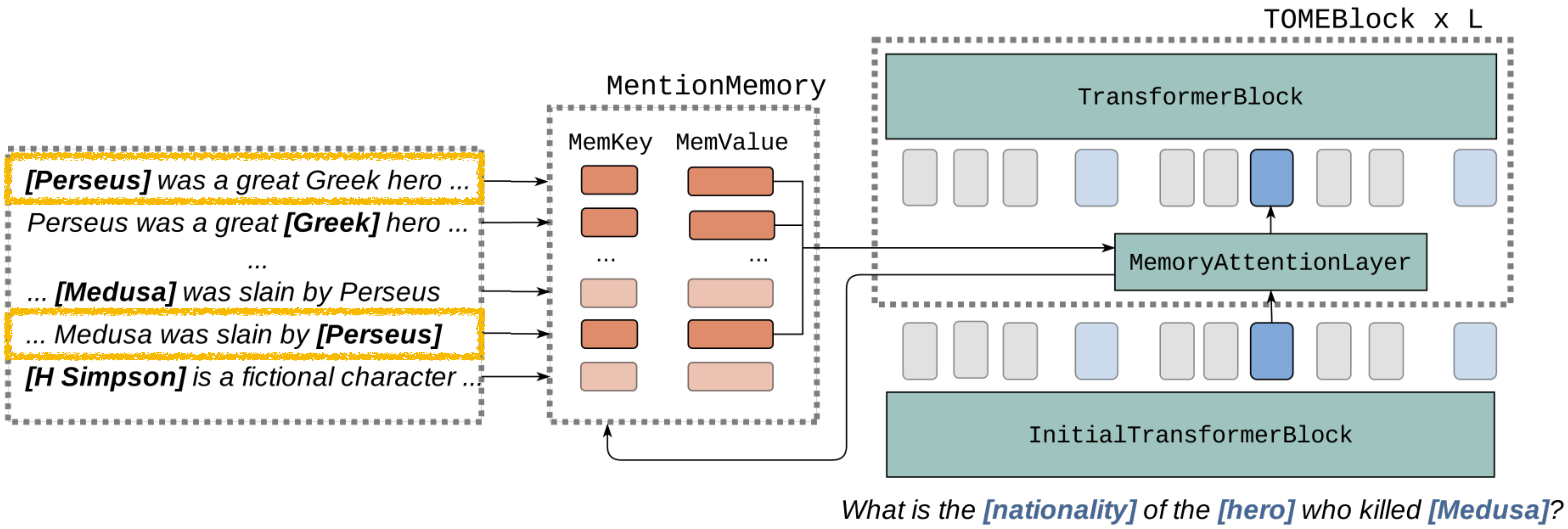
One vector per entity → One vector per entity *mention*



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

Mention Memory (de Jong et al. 2022)

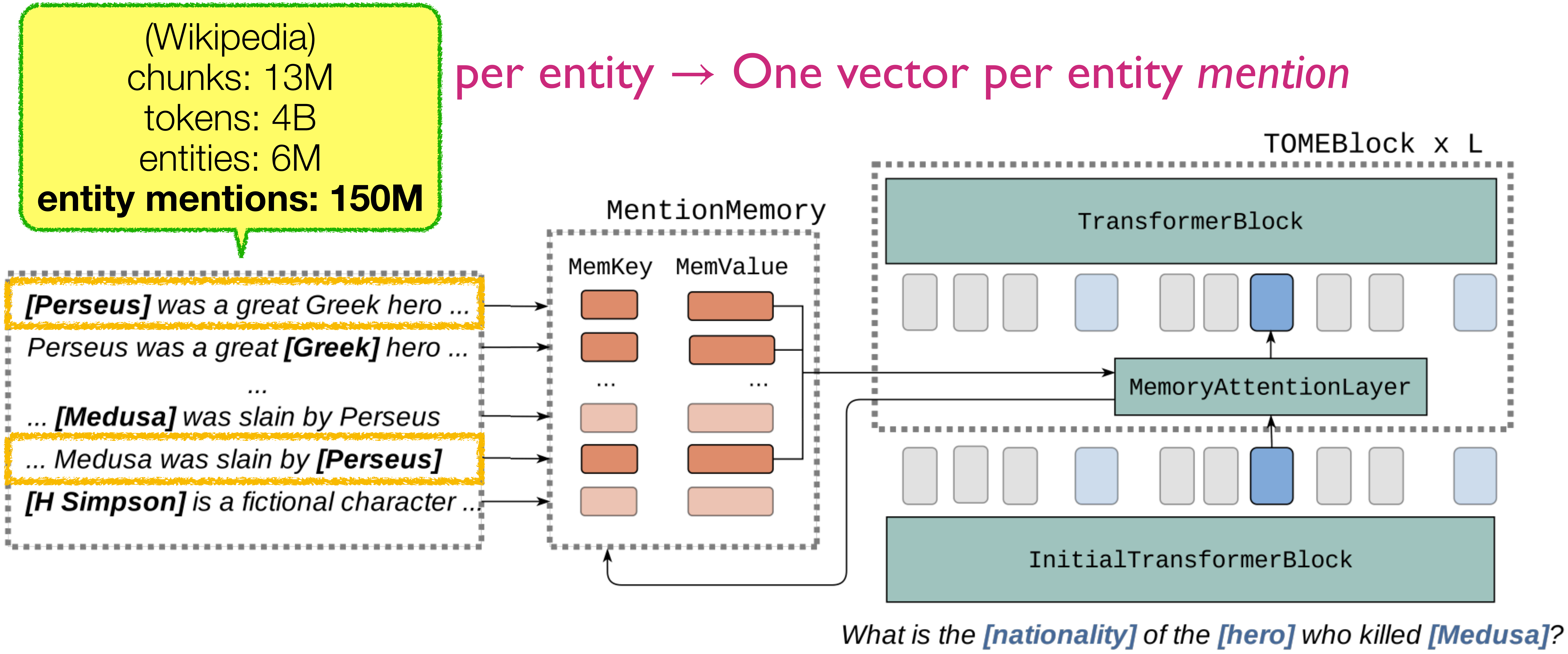
One vector per entity → One vector per entity *mention*



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

Mention Memory (de Jong et al. 2022)

per entity → One vector per entity *mention*



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

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 <u>the external text</u>

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

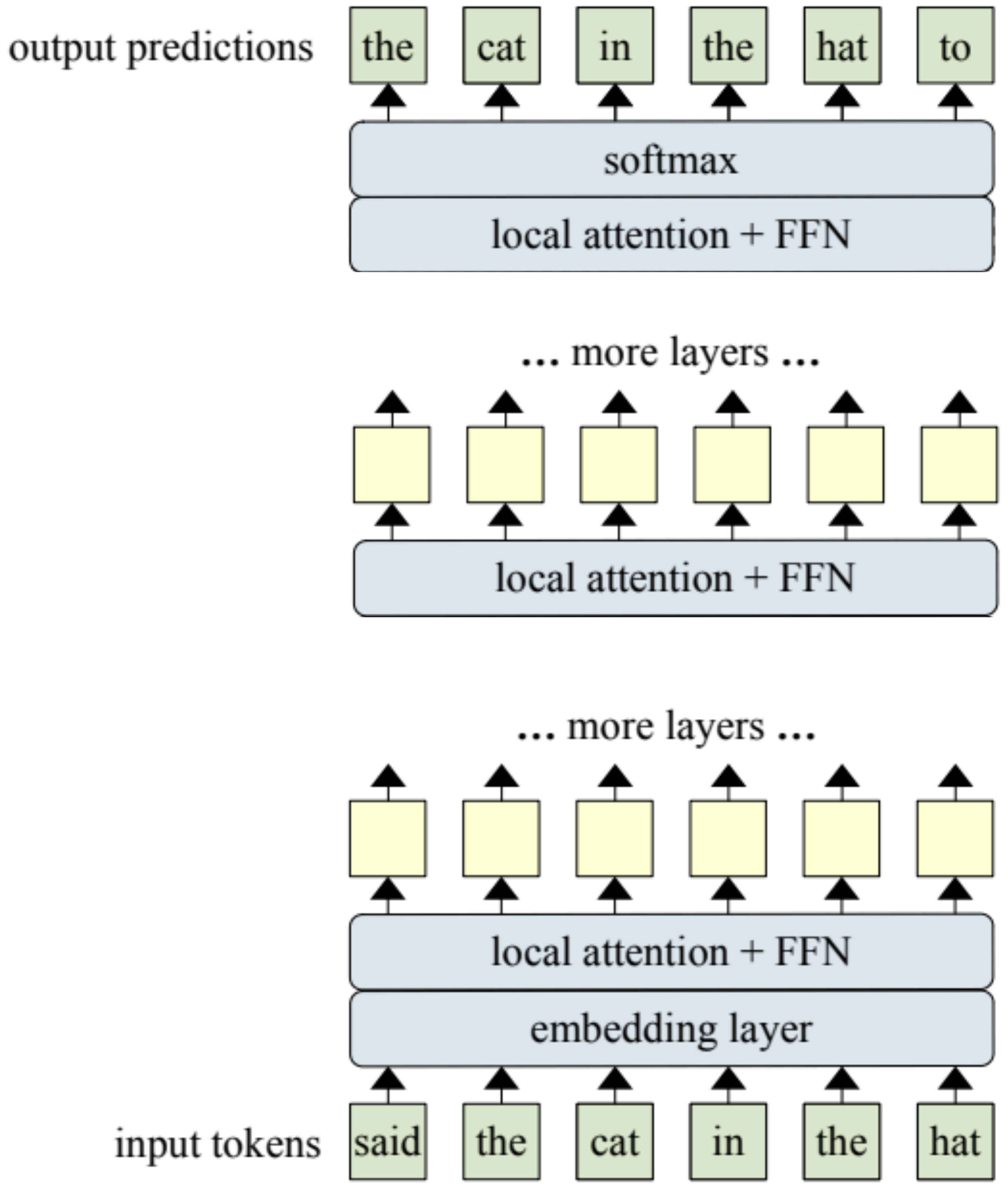
All models retrieve from <u>the external text</u>

What else can we do with these models?

Retrieval for long-range LM

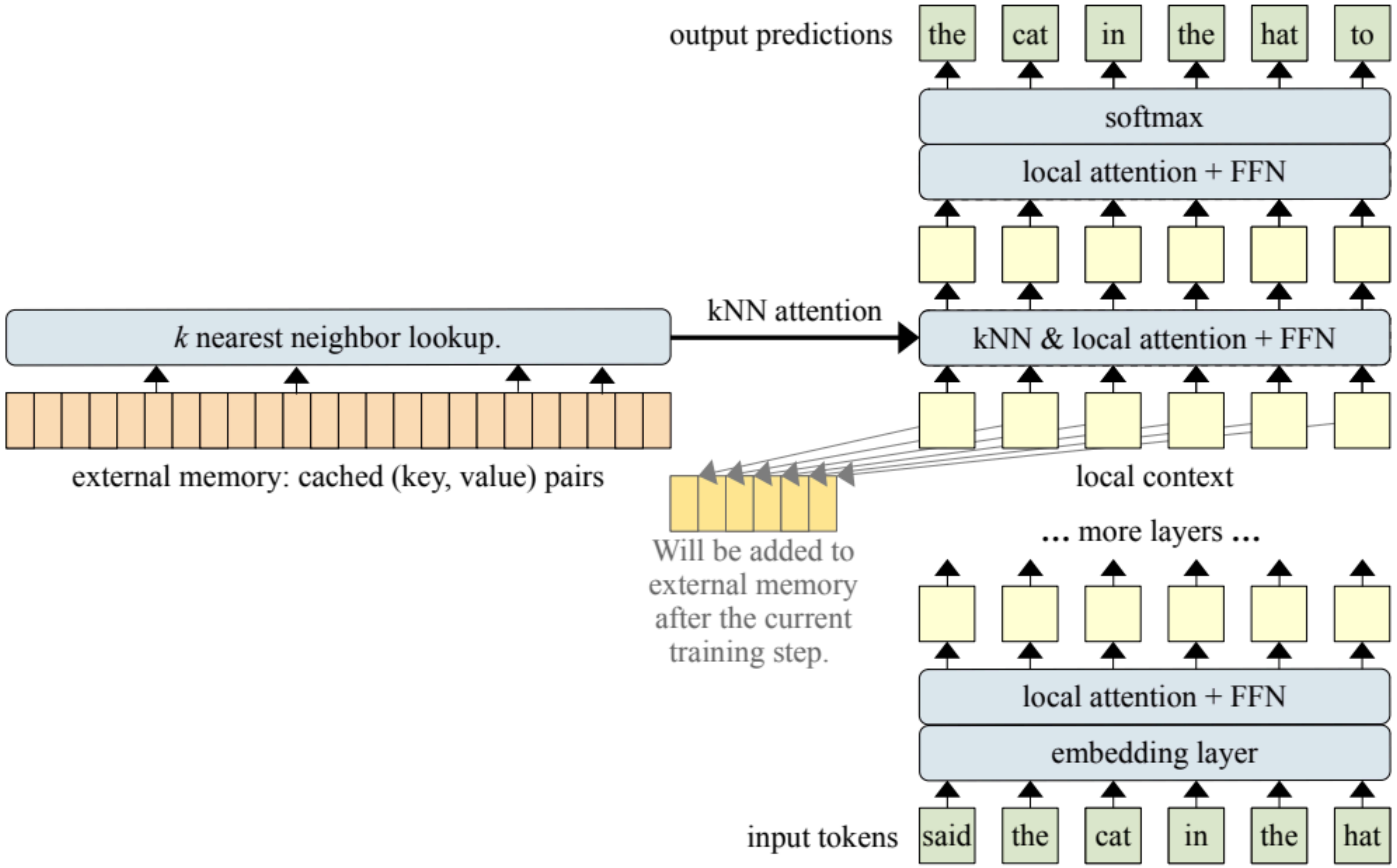
Wu et al. 2022. Memorizing Transformers **(Figure source)**
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval

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

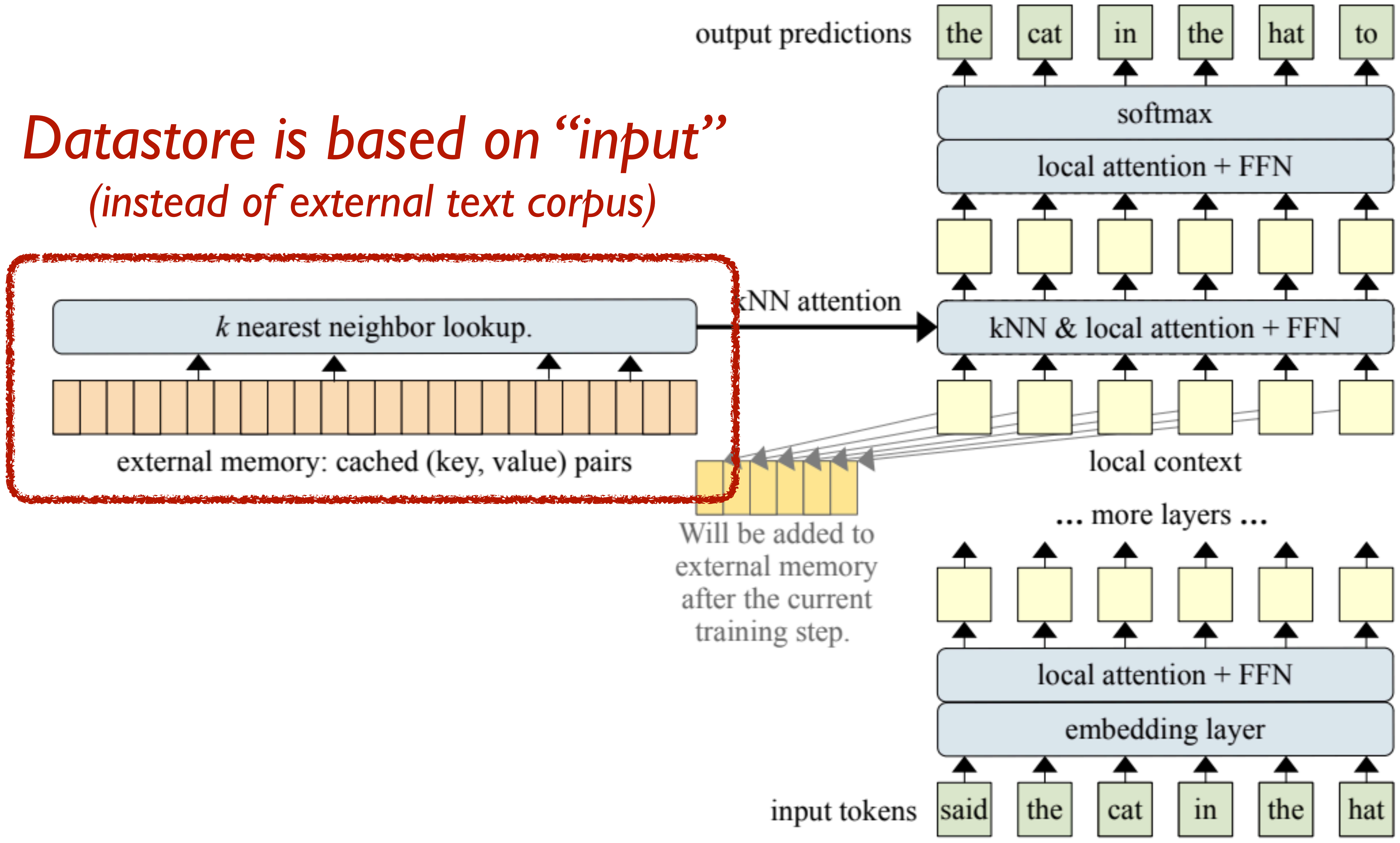
Retrieval for long-range LM



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

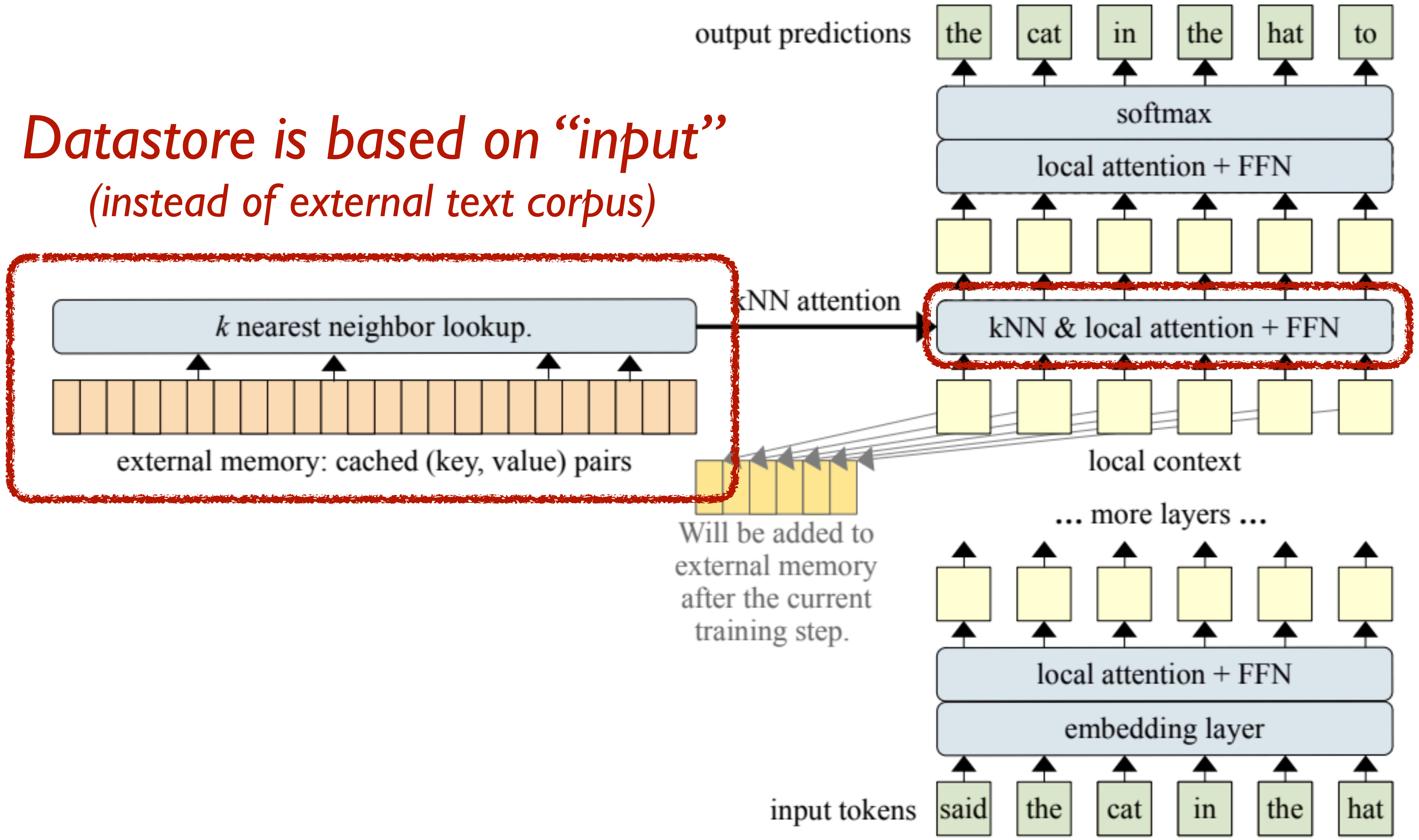
Datastore is based on "input"
(instead of external text corpus)



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

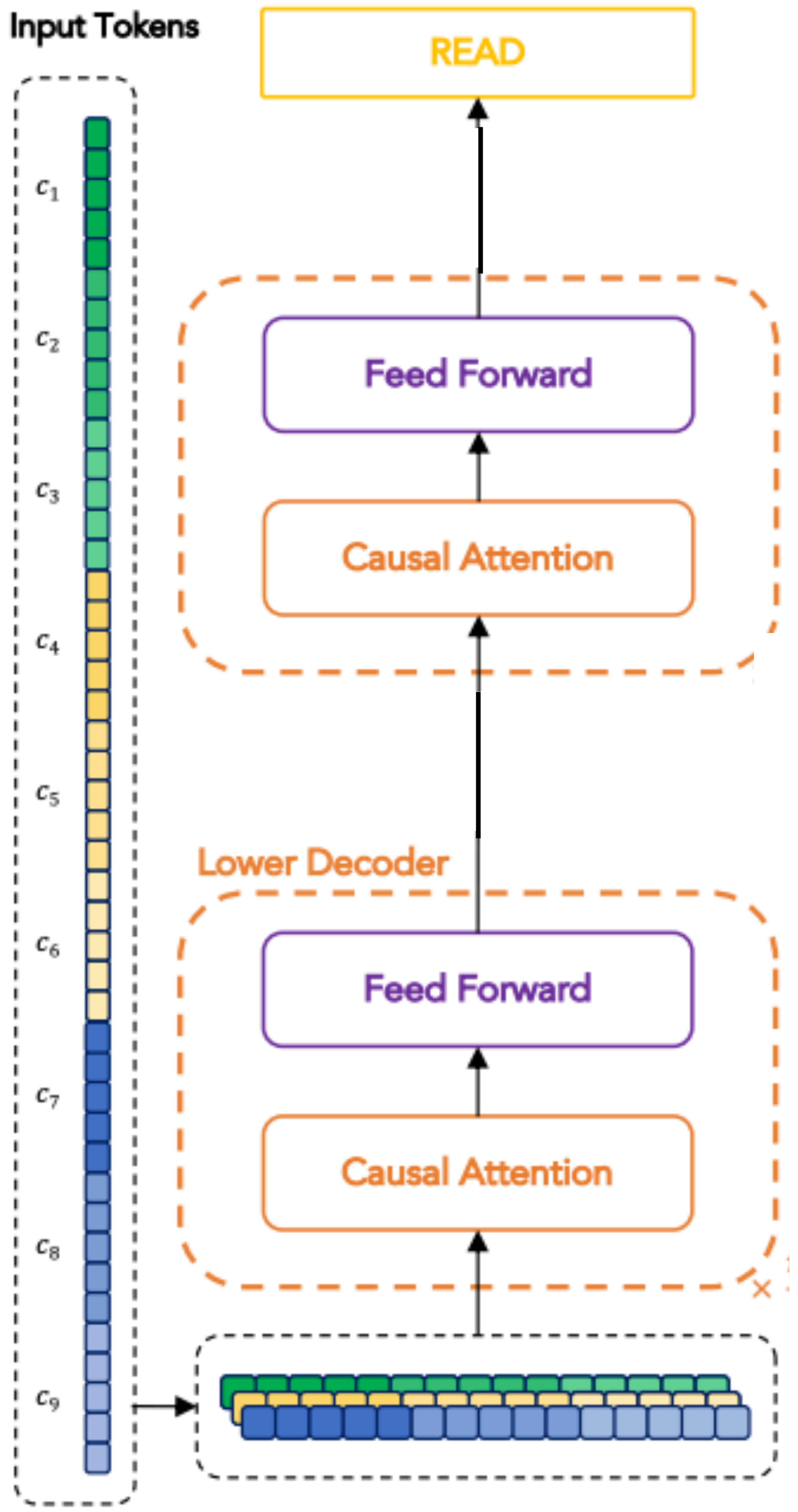
Datastore is based on “input”
(instead of external text corpus)



kNN search incorporated in the attention layer

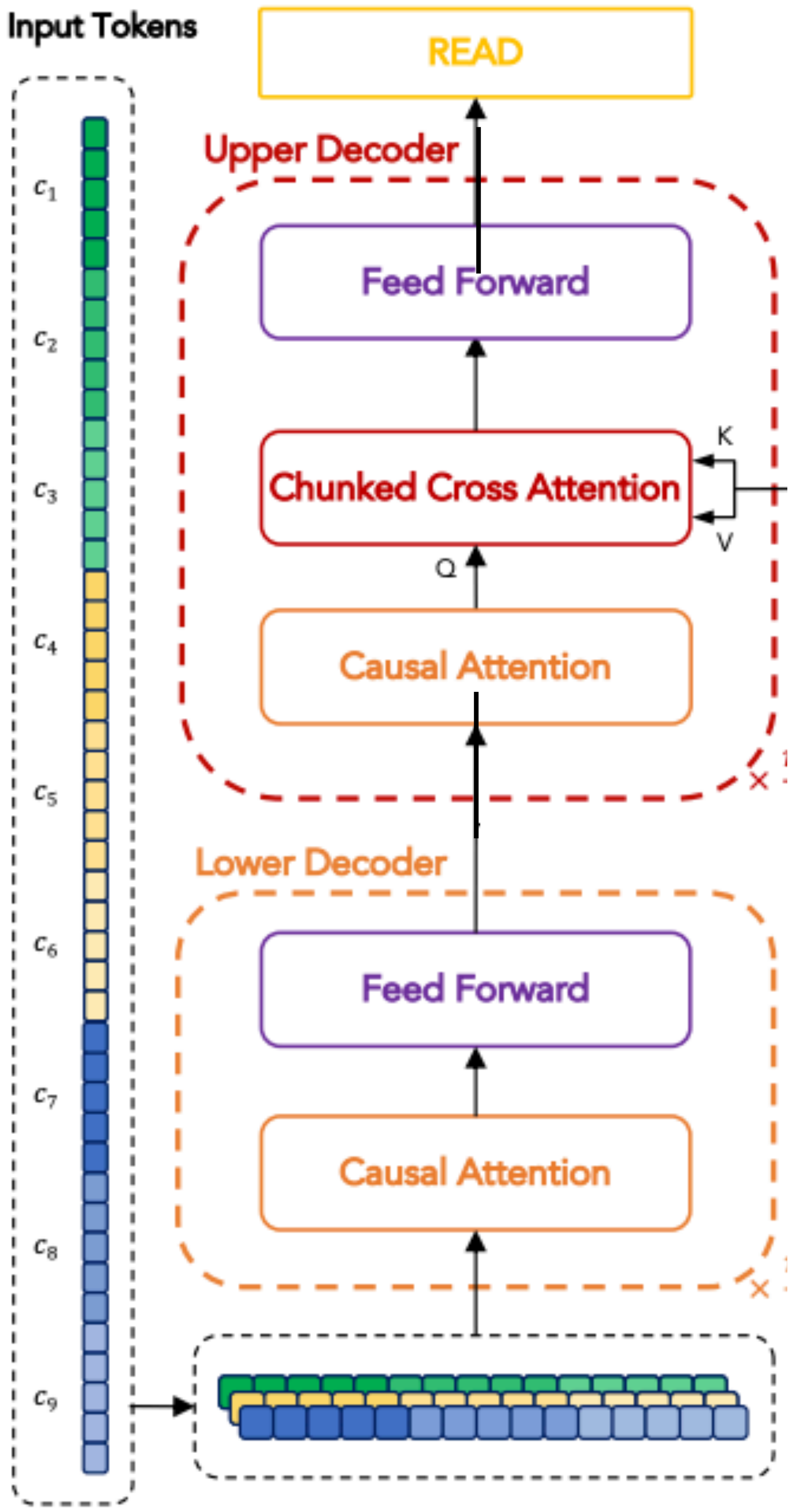
Wu et al. 2022. Memorizing Transformers **(Figure source)**
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval

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)**

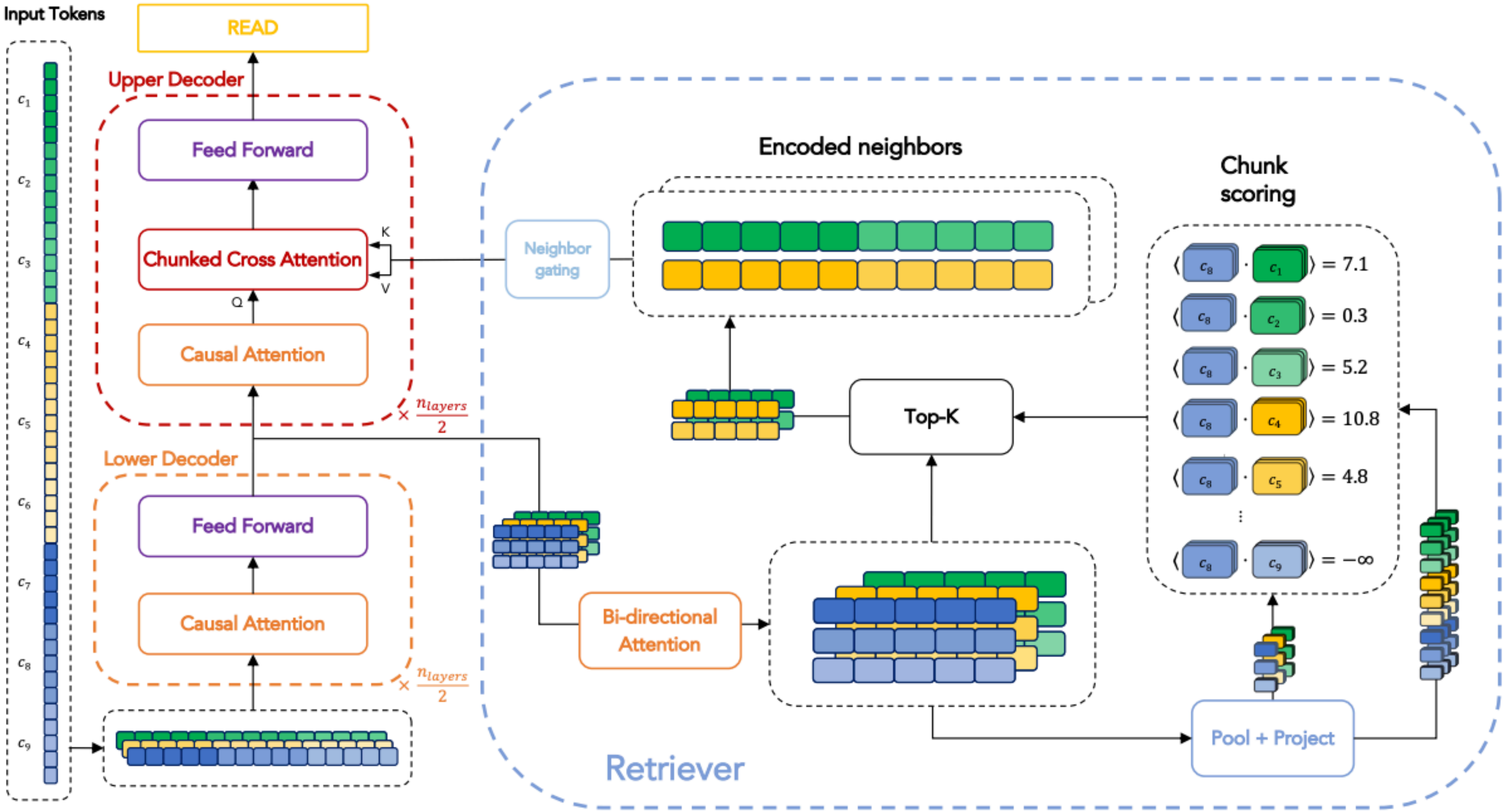
Retrieval for long-range LM



Chunked Cross Attention

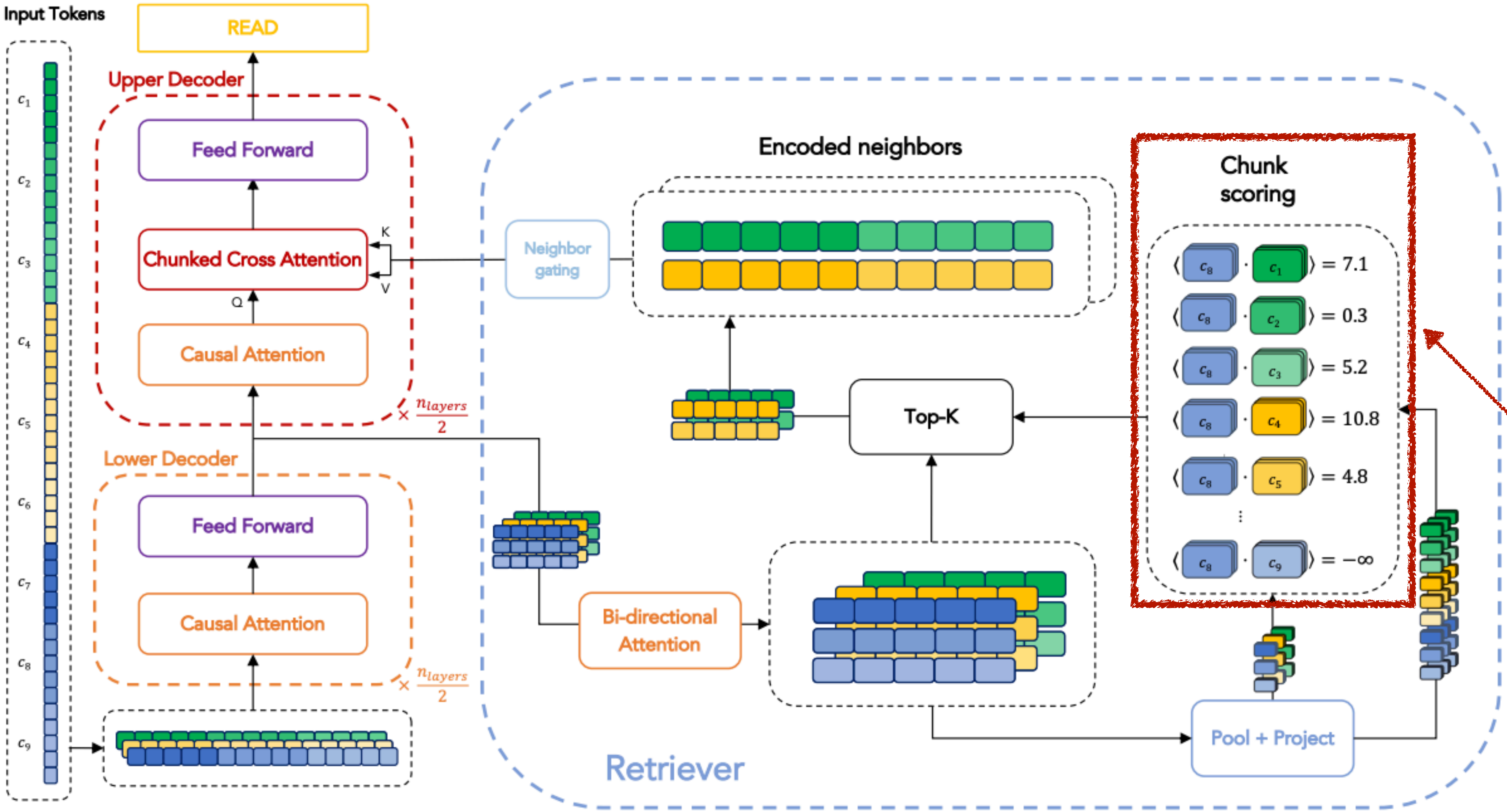
Wu et al. 2022. Memorizing Transformers
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval **(Figure source)**

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)**

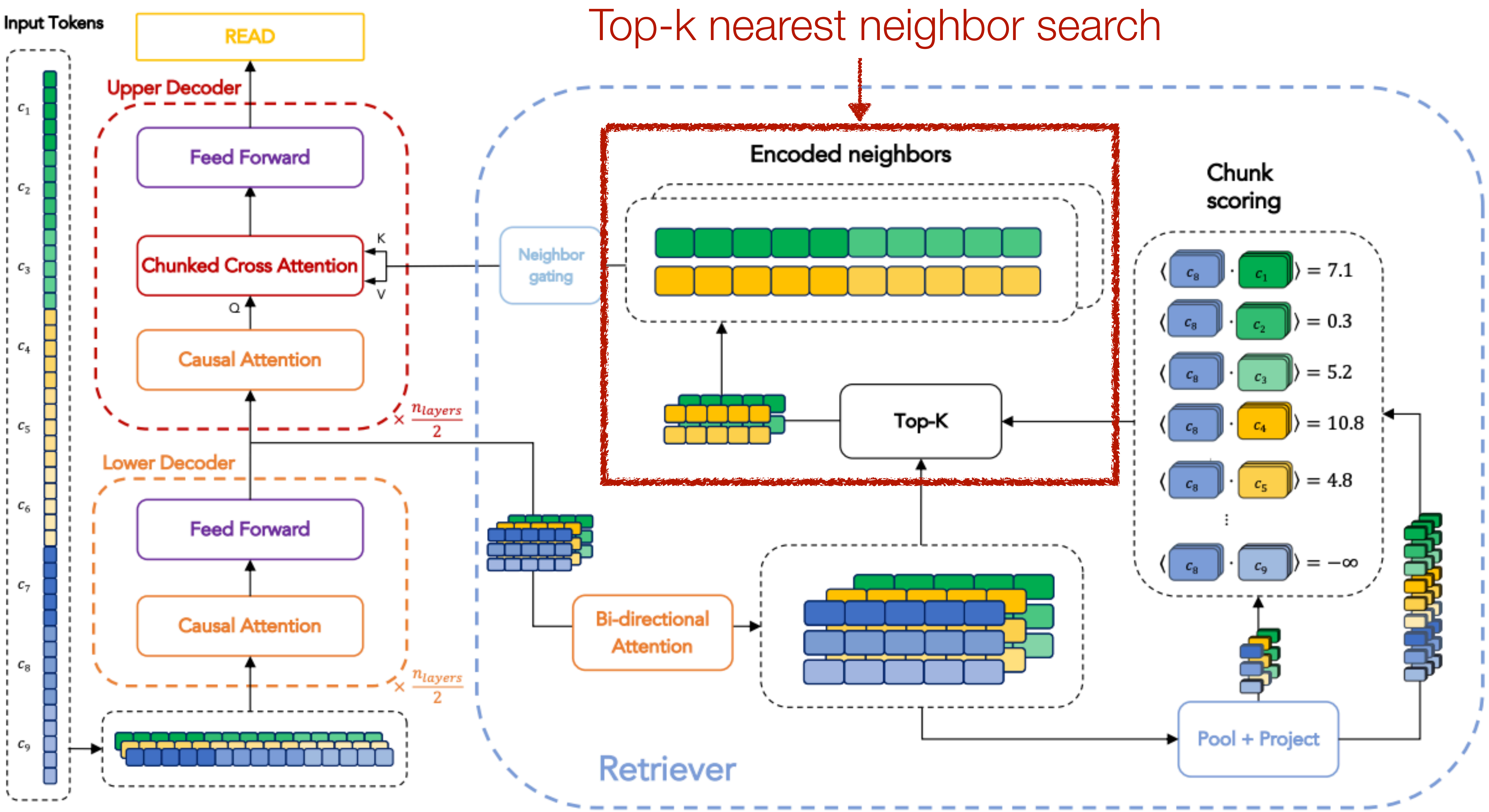
Retrieval for long-range LM



Every chunk is assigned a similarity score

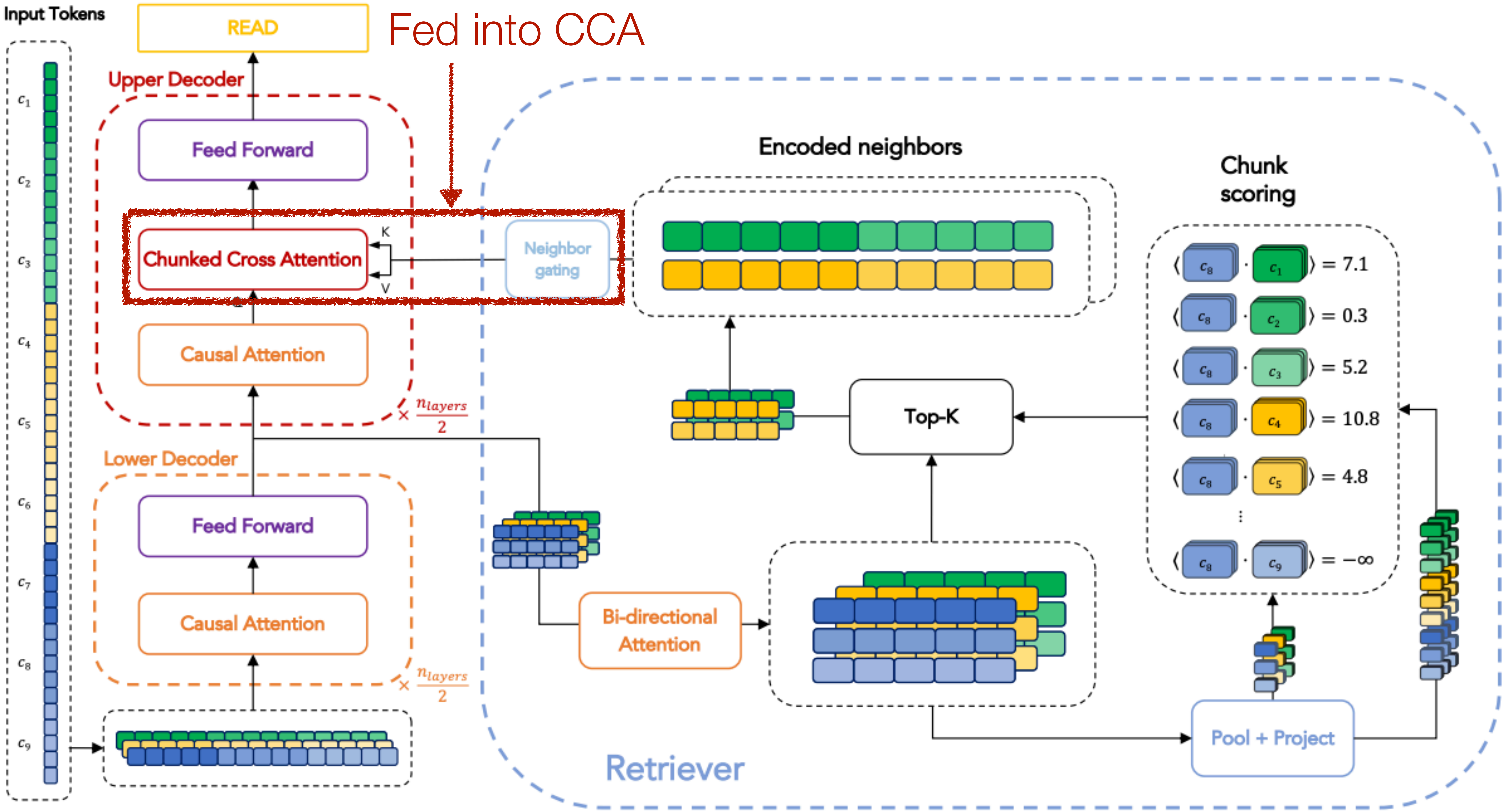
Wu et al. 2022. Memorizing Transformers
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval **(Figure source)**

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

How to use retrieval?

When to retrieve?

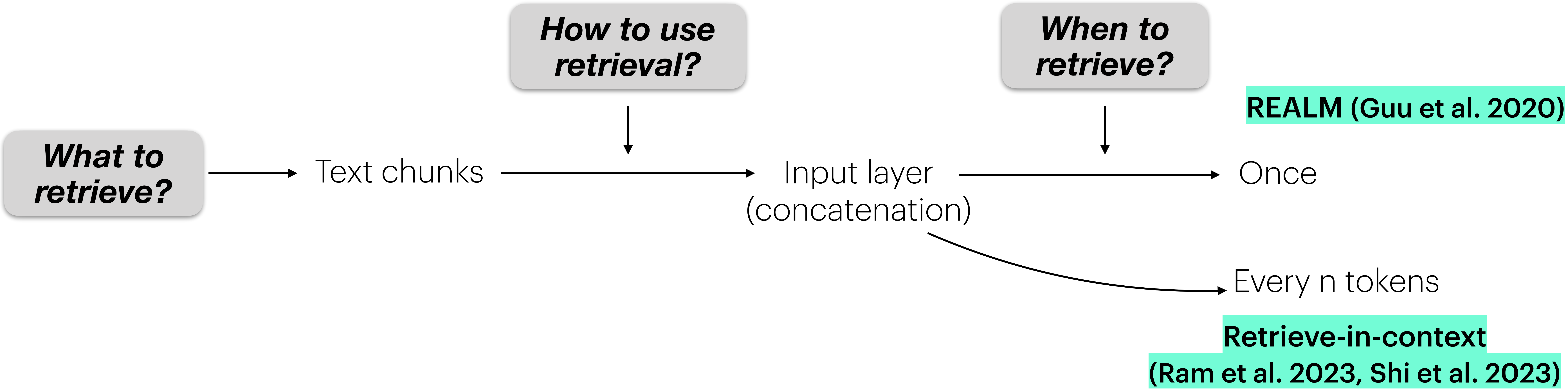
REALM (Guu et al. 2020)

What to retrieve? → Text chunks → Input layer (concatenation) → Once

Every n tokens

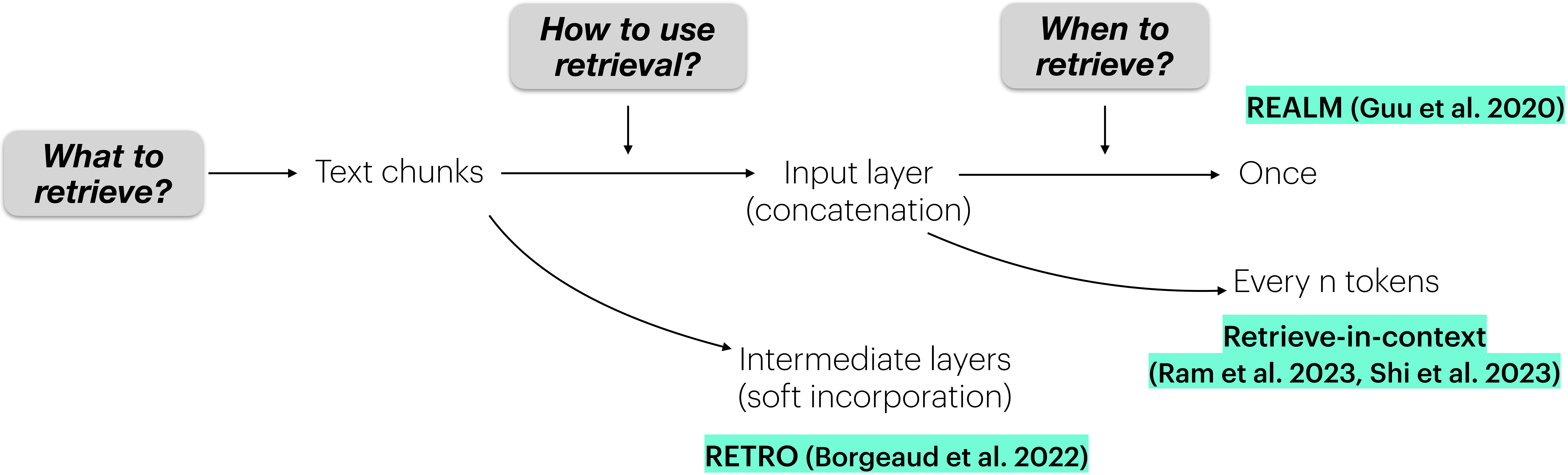
Retrieve-in-context (Ram et al. 2023, Shi et al. 2023)

Wrapping up



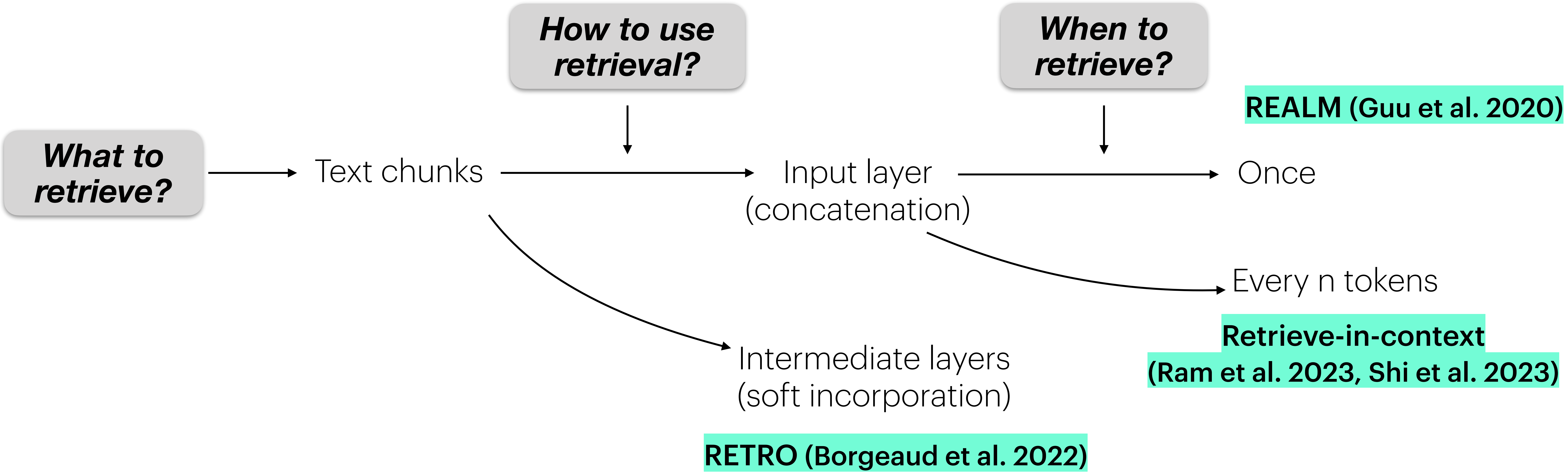
More frequent retrieval = better in performance, but slower

Wrapping up



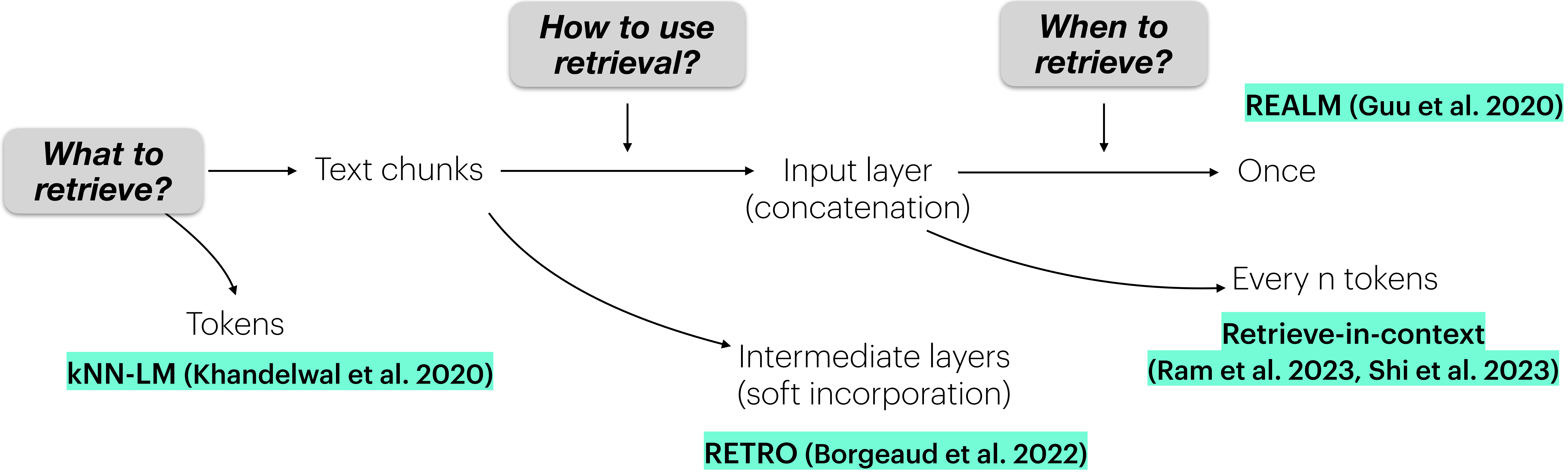
How to use retrieval?
When to retrieve?
REALM (Guu et al. 2020)
What to retrieve?
Text chunks
Input layer (concatenation)
Once
Every n tokens
Retrieve-in-context
(Ram et al. 2023, Shi et al. 2023)
Intermediate layers (soft incorporation)
RETRO (Borgeaud et al. 2022)

Wrapping up

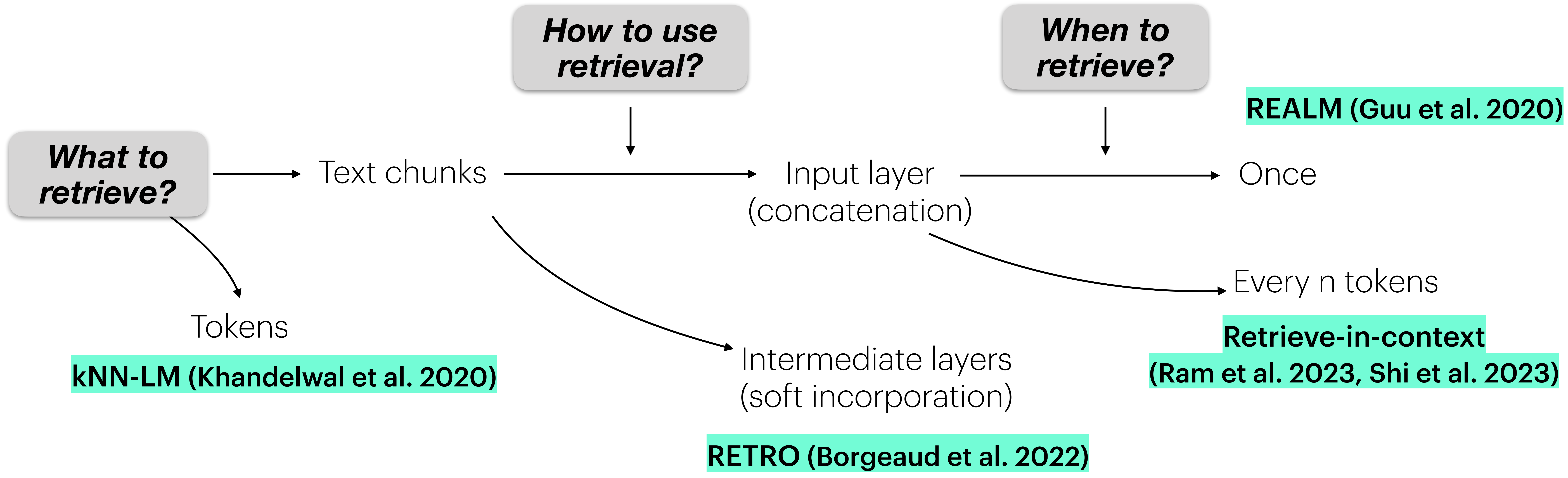


- Input layer: Simple but can be slower
- Intermediate layers: More complex (need training) but can be designed to be more efficient

Wrapping up


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)

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

What to retrieve?

Text chunks

Tokens

kNN-LM (Khandelwal et al. 2020)

Adaptively

He et al. 2021, Alon et al. 2022

How to use retrieval?

Input layer (concatenation)

Intermediate layers (soft incorporation)

RETRO (Borgeaud et al. 2022)

When to retrieve?

Once

REALM (Guu et al. 2020)

Every n tokens

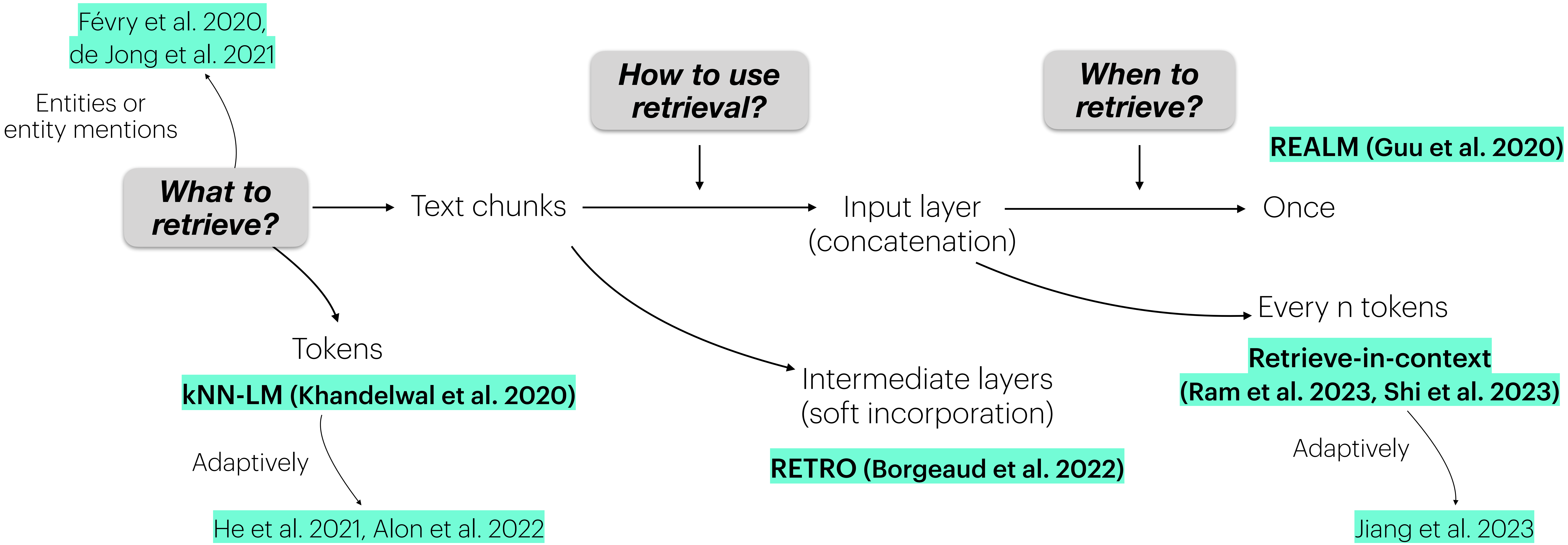
Retrieve-in-context (Ram et al. 2023, Shi et al. 2023)

Adaptively

Jiang et al. 2023

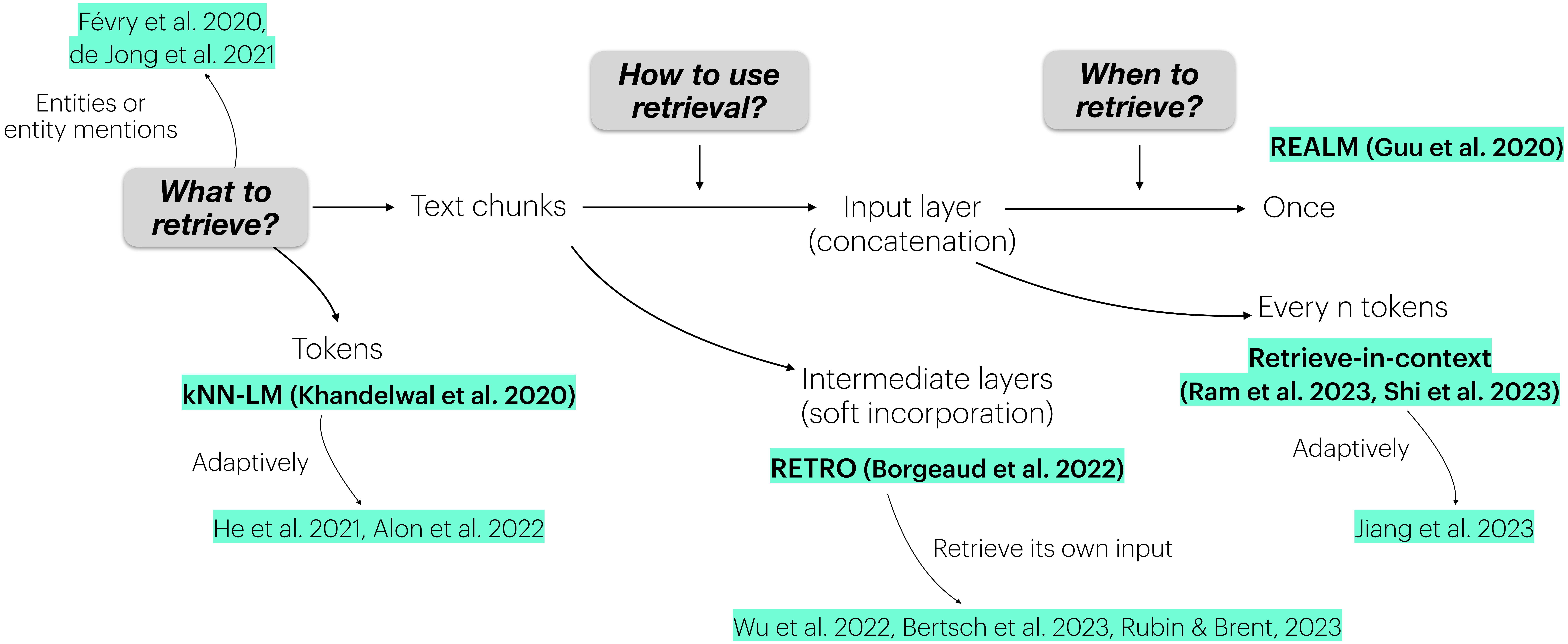
Adaptive retrieval can improve efficiency

Wrapping up



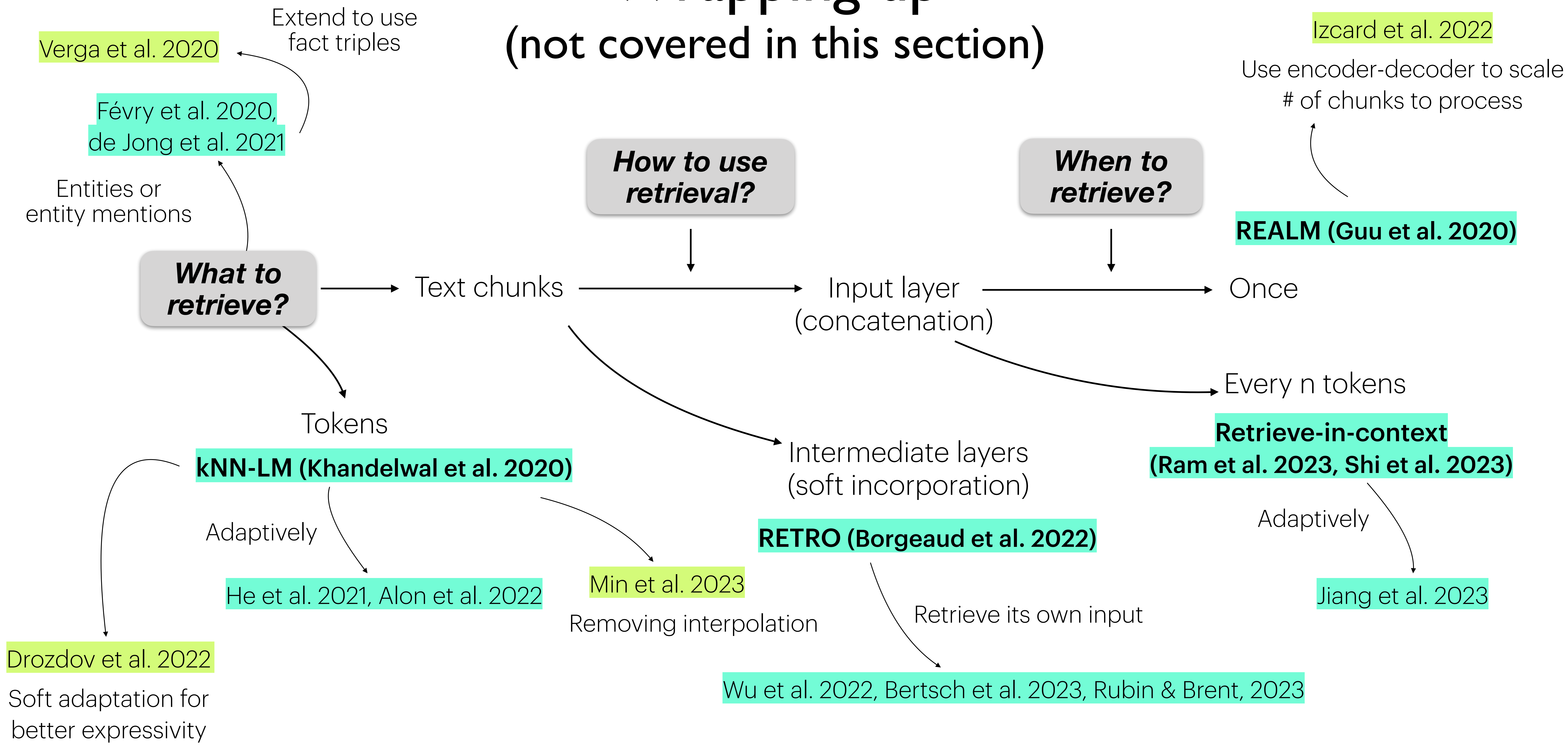
Entities or entity mentions instead of every token or chunk

Wrapping up

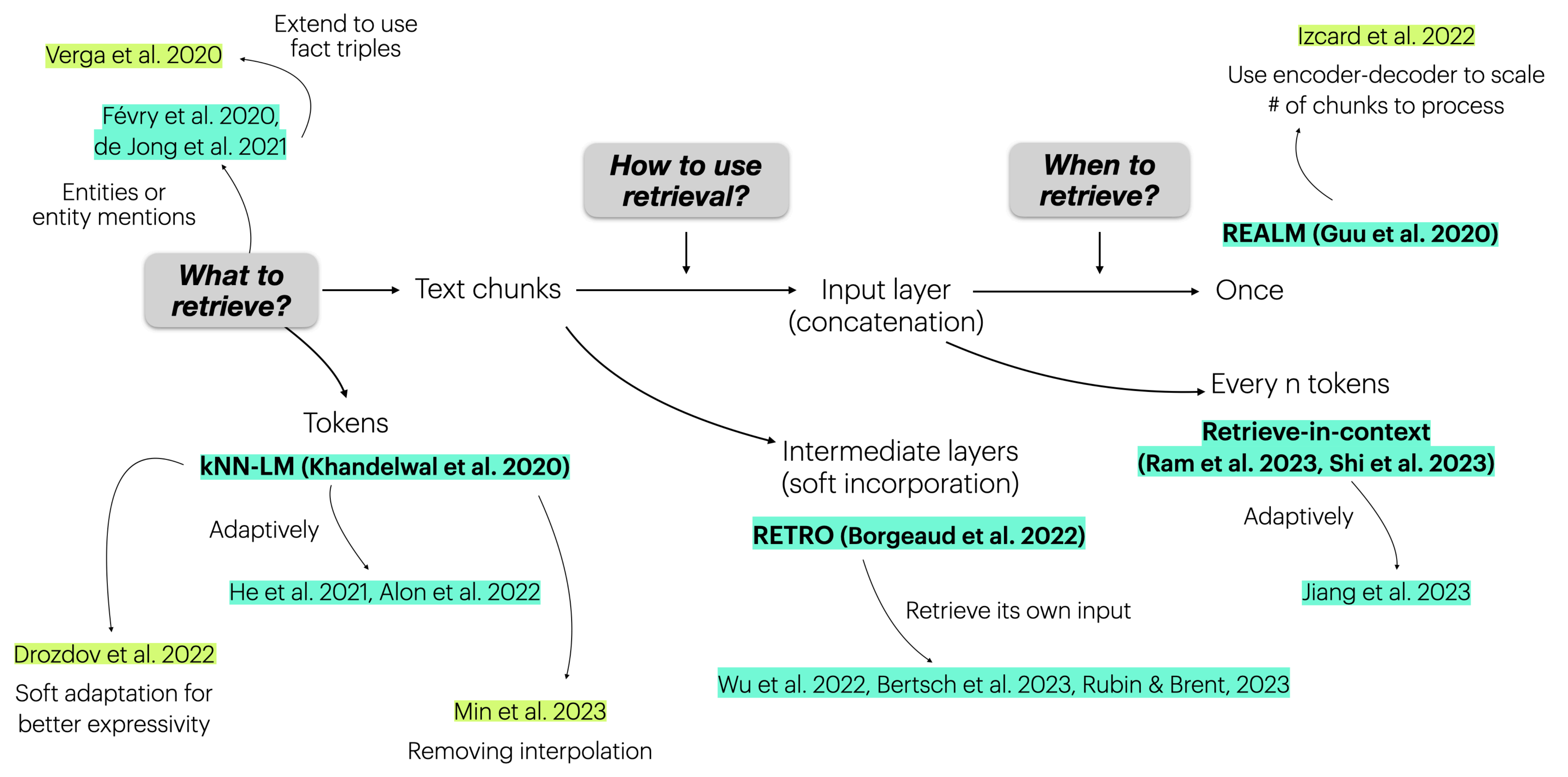


We can use a similar approach for long-sequence modeling

Wrapping up
(not covered in this section)

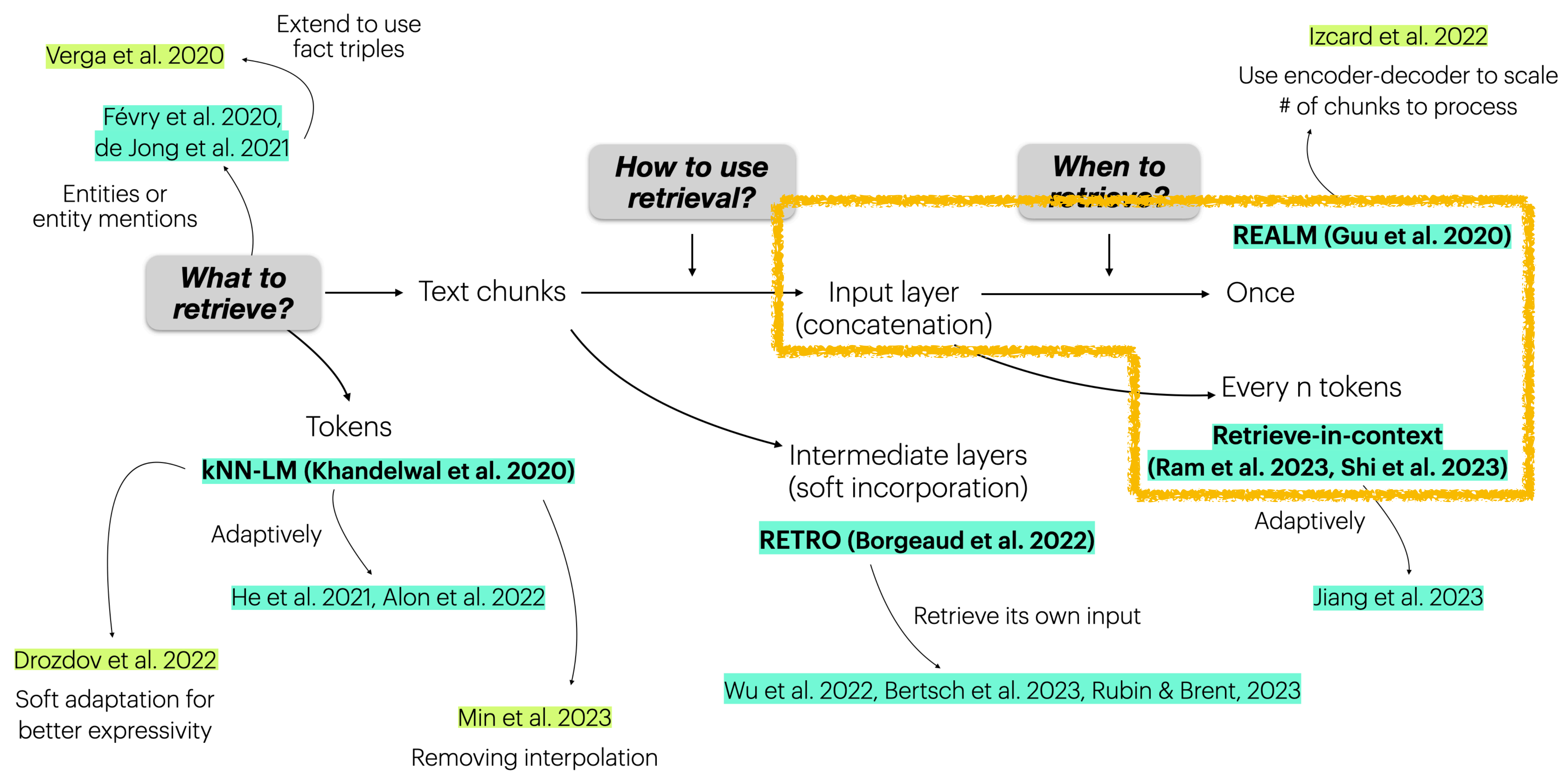


Wrapping up



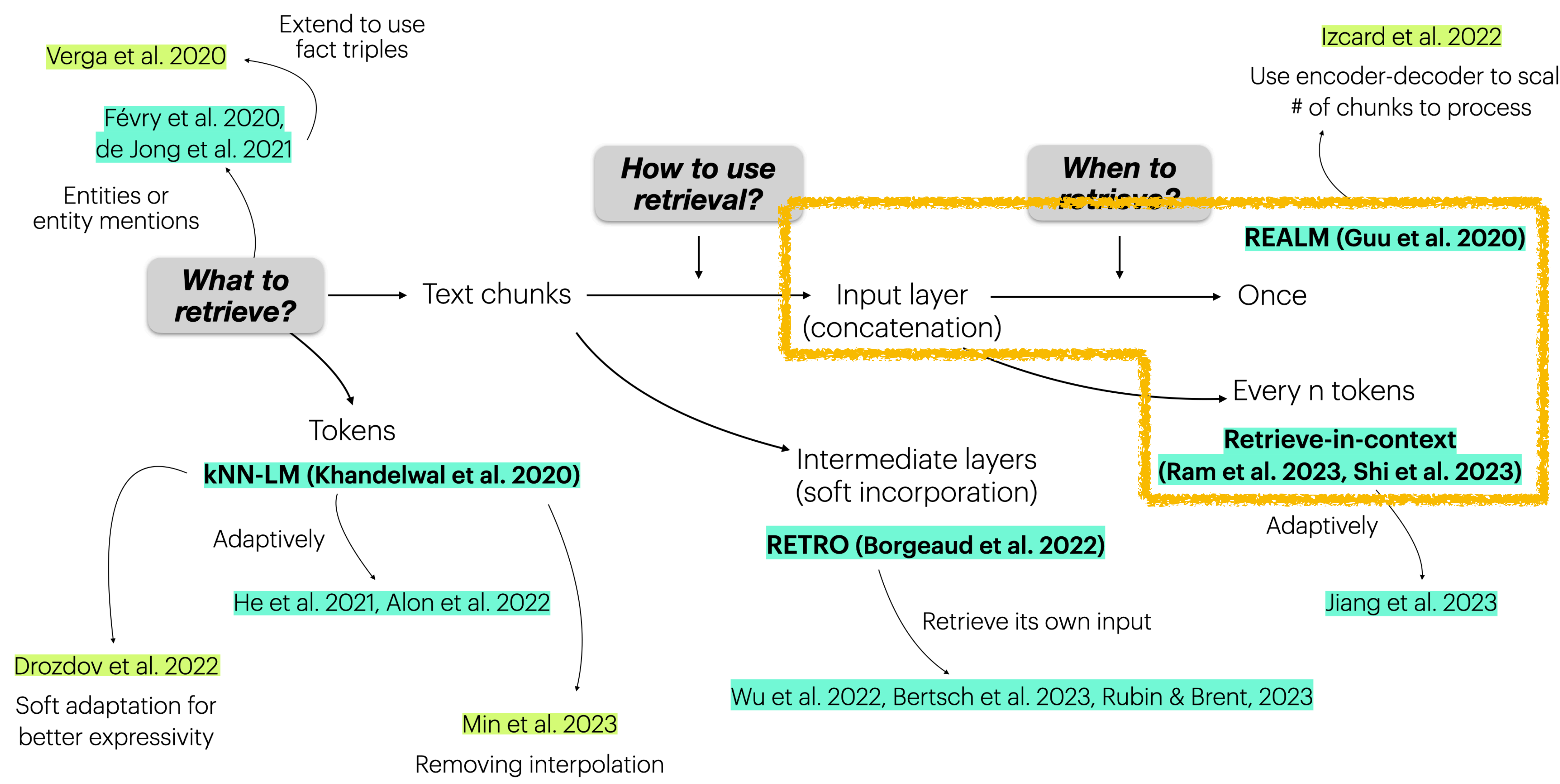
Perplexity WebGPT YOU Chat GPT Extension

Wrapping up



Perplexity WebGPT YOU Chat GPT Extension

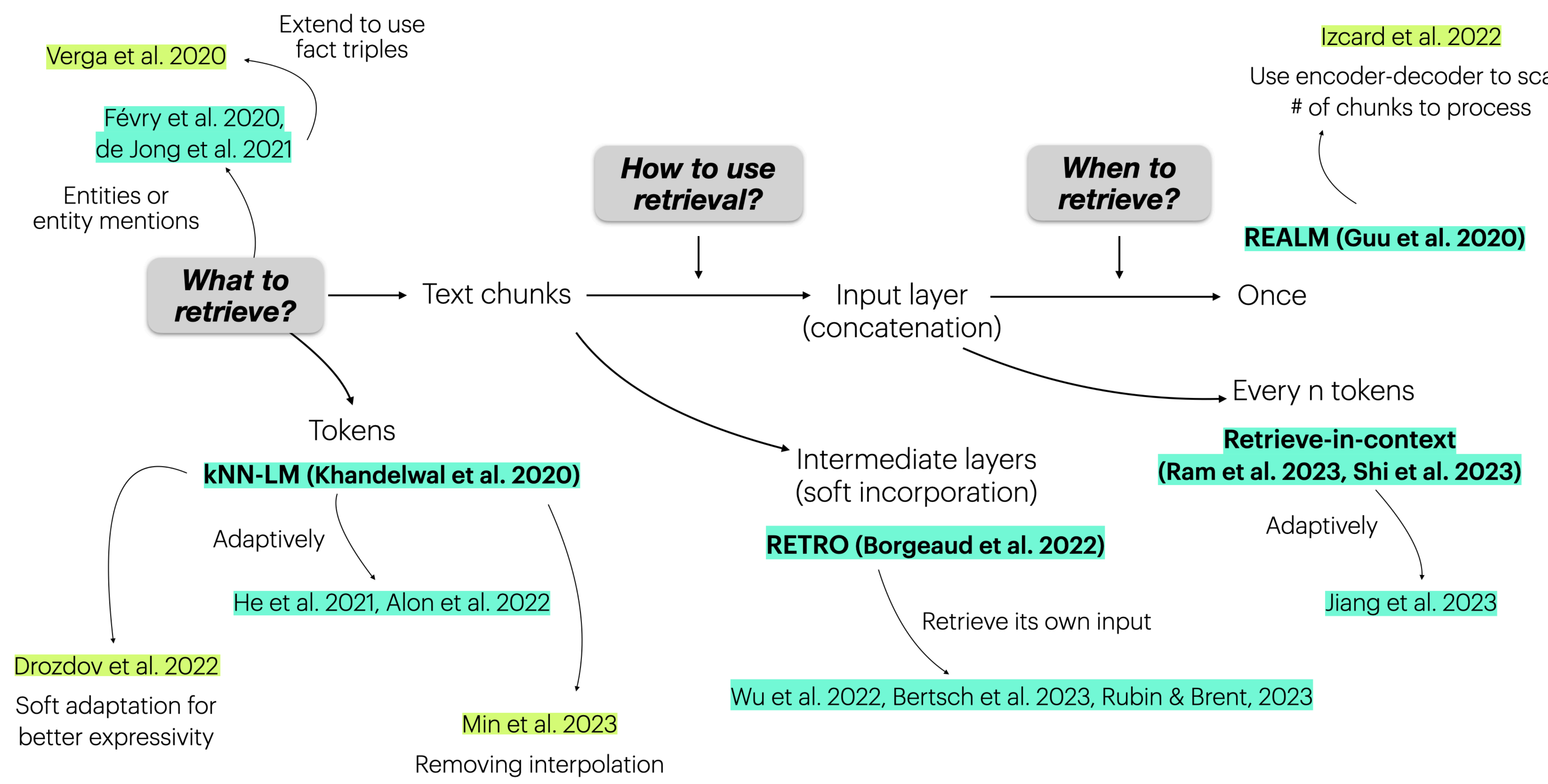
Wrapping up



Perplexity WebGPT YOU 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**!