

Section 2: Definition & Preliminaries

A Retrieval-based LM: Definition

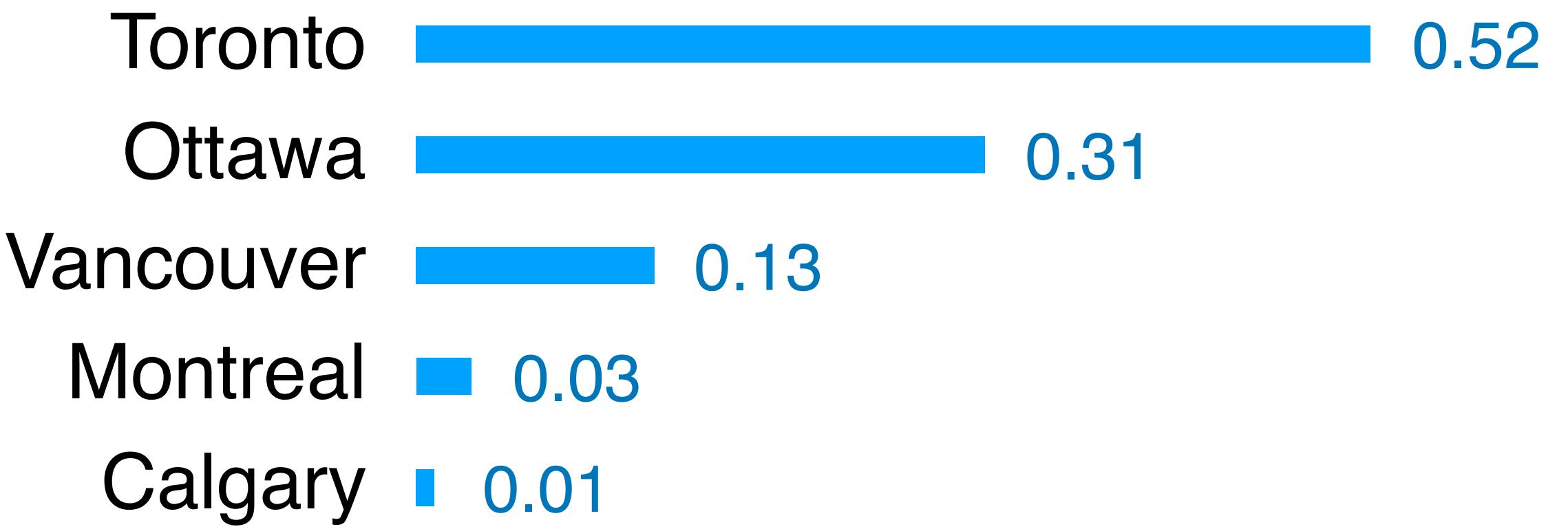
A language model (LM) that uses
an external datastore at test time

A Retrieval-based LM: Definition

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

A language model (LM)

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



Language model (Transformers)

The capital city of Ontario is

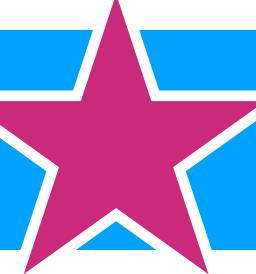
x_1

x_2

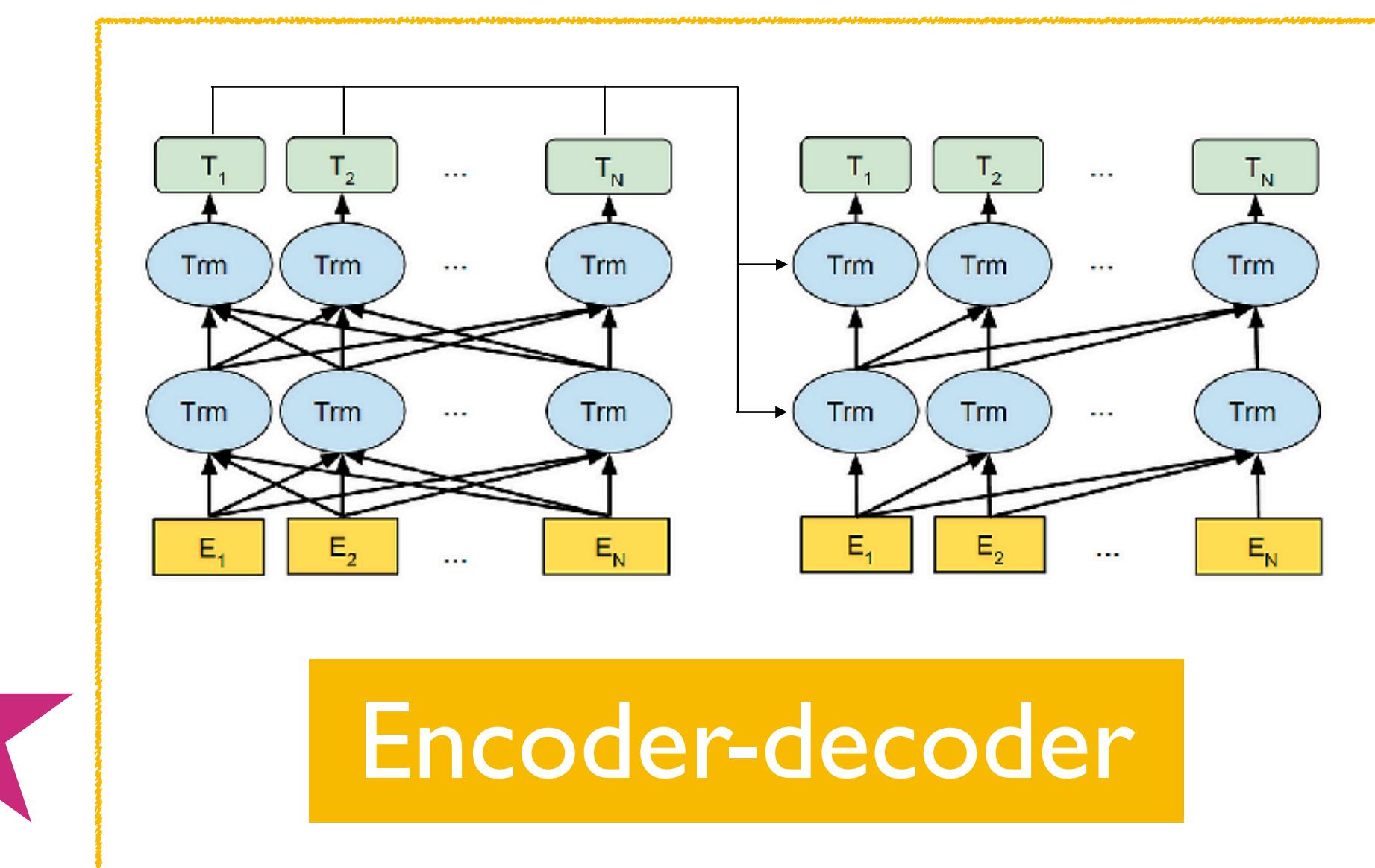
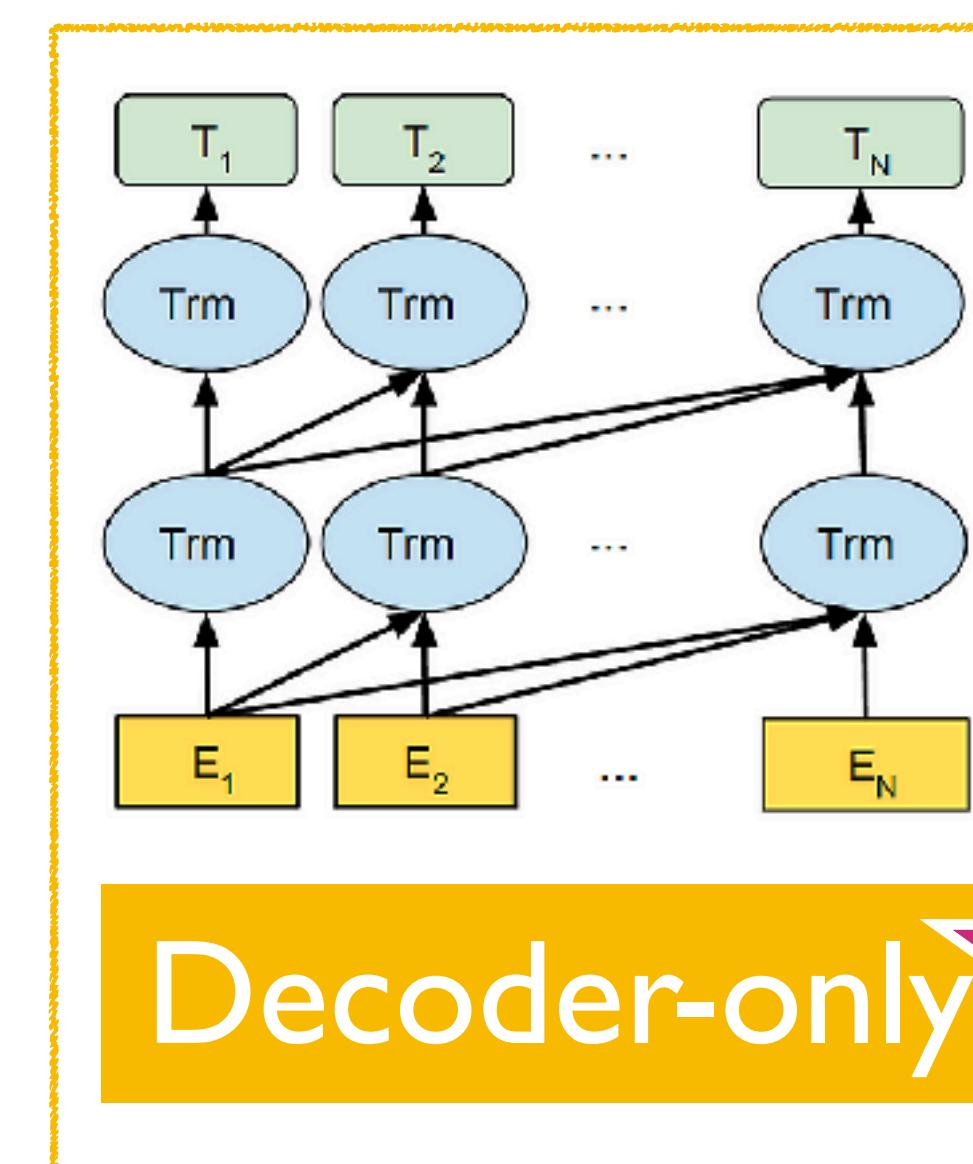
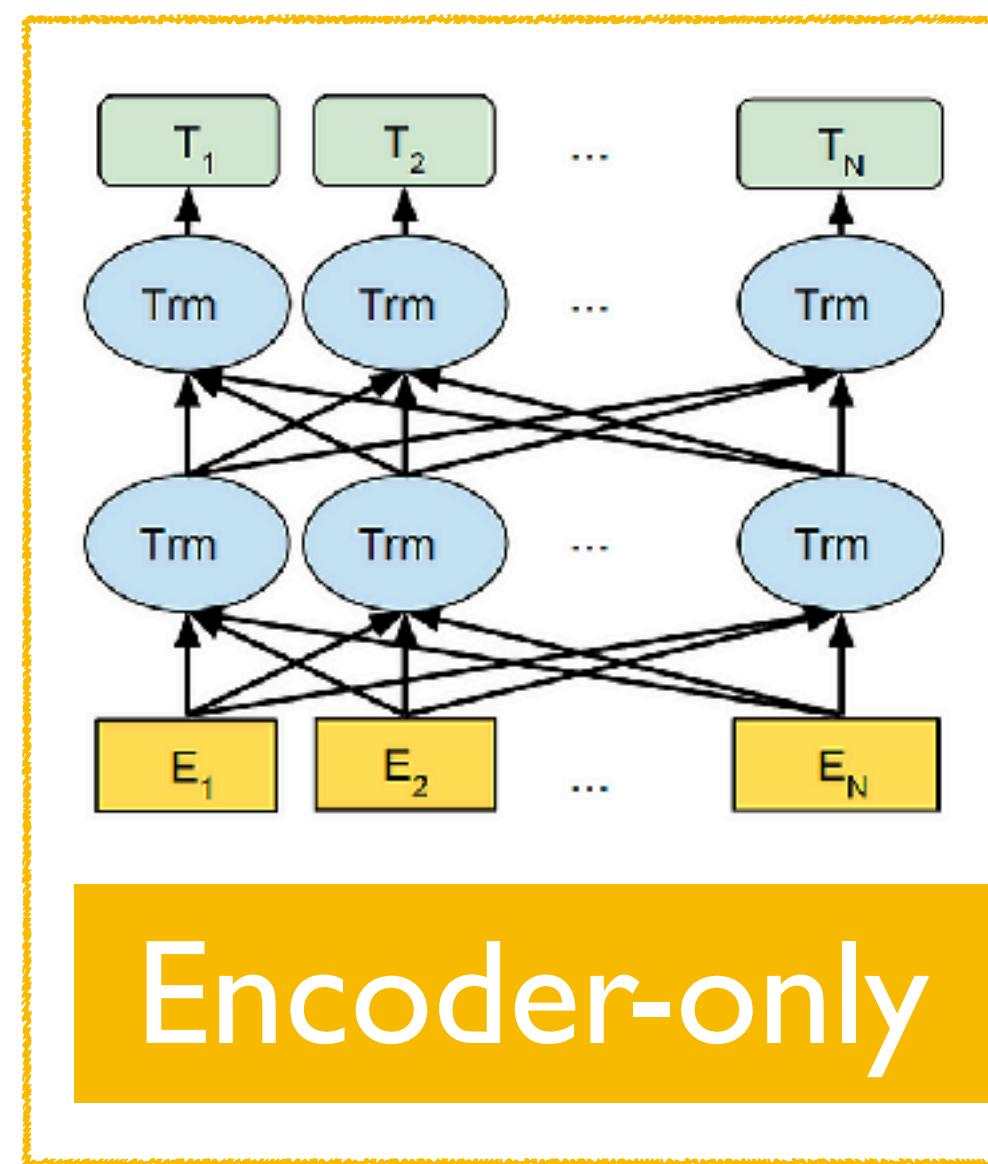
...

x_{n-1}

A language model (LM): Categories

Toronto
Autoregressive LM 
The capital city of Ontario is _____

capital Ontario
Masked LM
The _____ city of _____ is Toronto



A language model (LM): Prompting

The capital city of Ontario is

LM

Toronto

Fact probing

Cheaper than an iPod. It was

LM

great
terrible

Sentiment analysis

“Hello” in French is

LM

Bonjourno

Translation

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

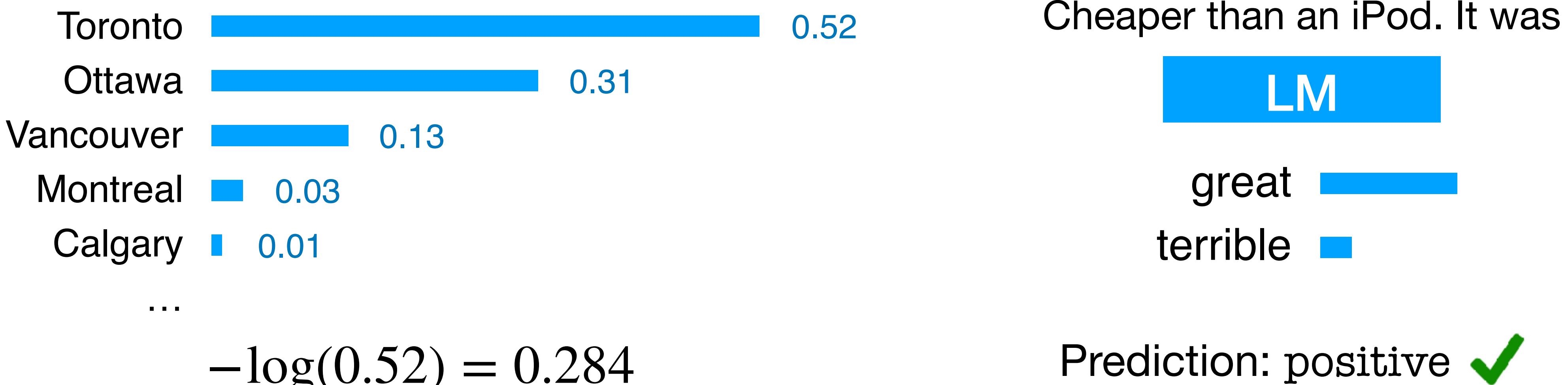
LM

101

Arithmetic

A language model (LM)

Often evaluated with



Perplexity

Downstream accuracy

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

(More in Section 5)

A Retrieval-based LM: Definition

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

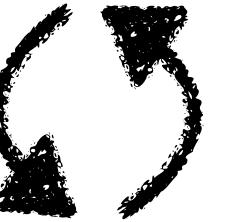
A Retrieval-based LM: Definition

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

Typical Los



The capital city of Ontario is **Toronto**



LM

Training time

The capital city of Ontario is _____



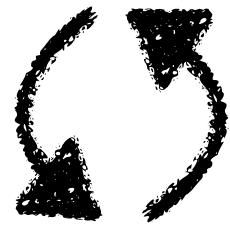
LM

Test time

Retrieval-based LMs



The capital city of Ontario is **Toronto**



LM

Training time



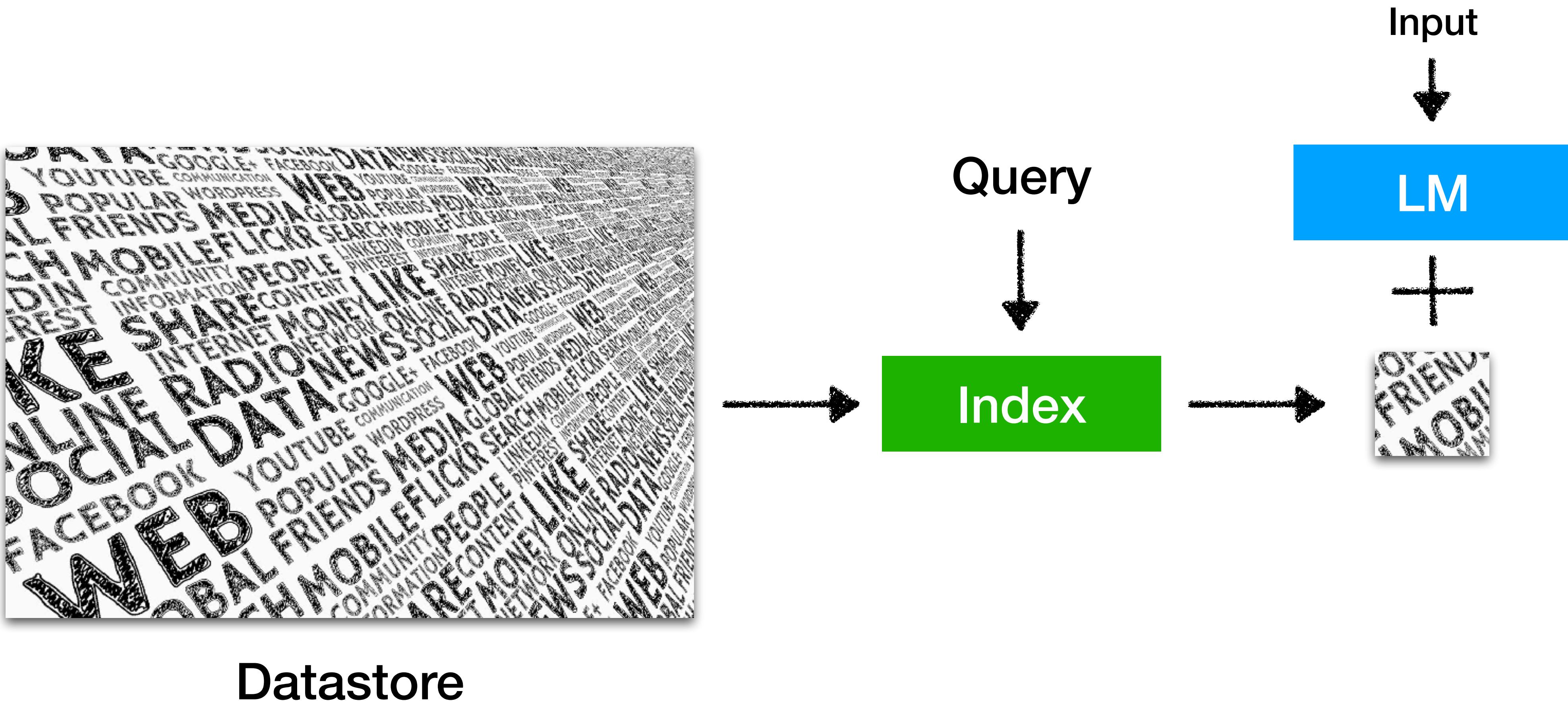
The capital city of Ontario is _____



LM

Test time

Inference



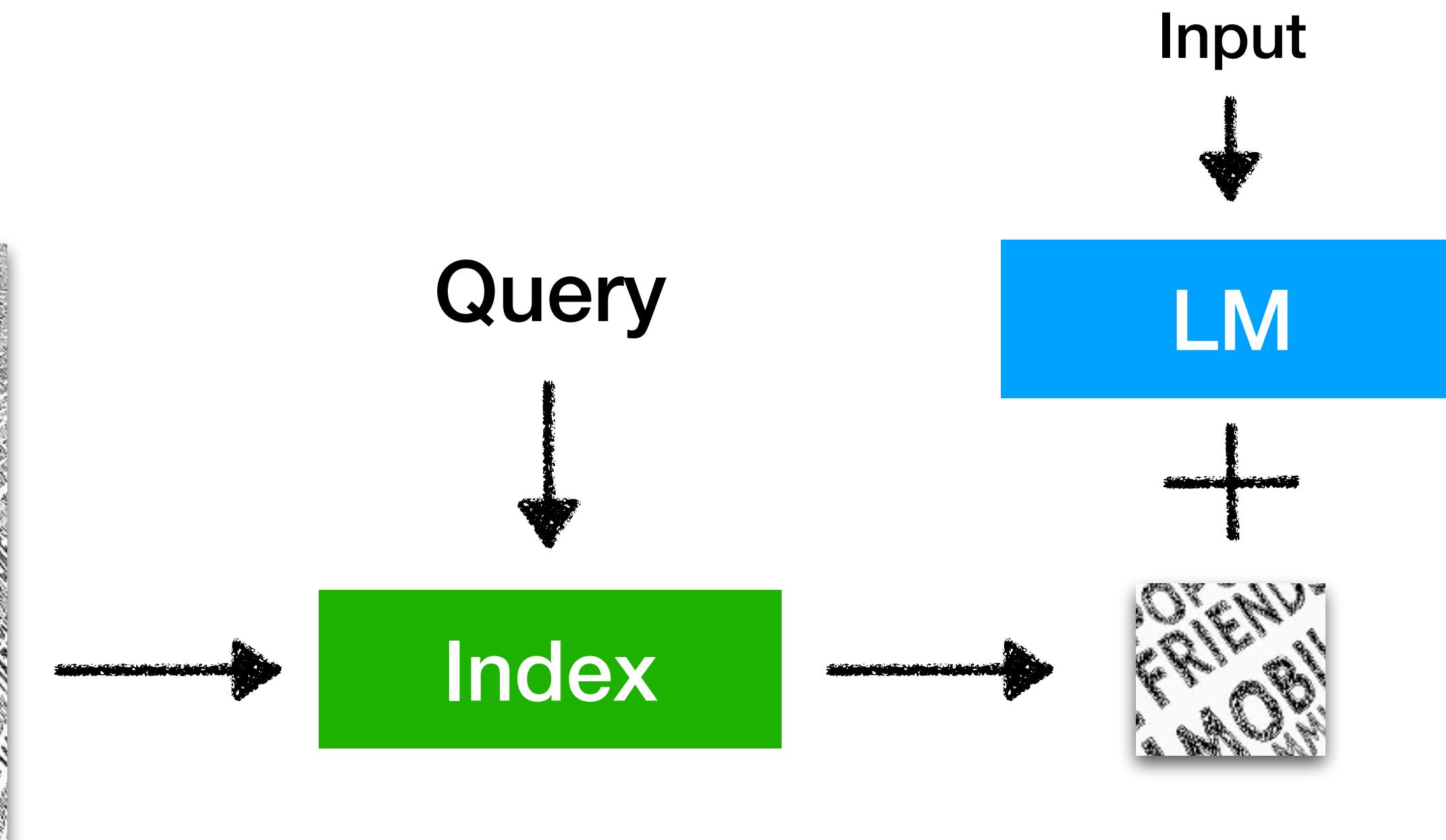
Inference: Datastore



Datastore
Raw text corpus

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

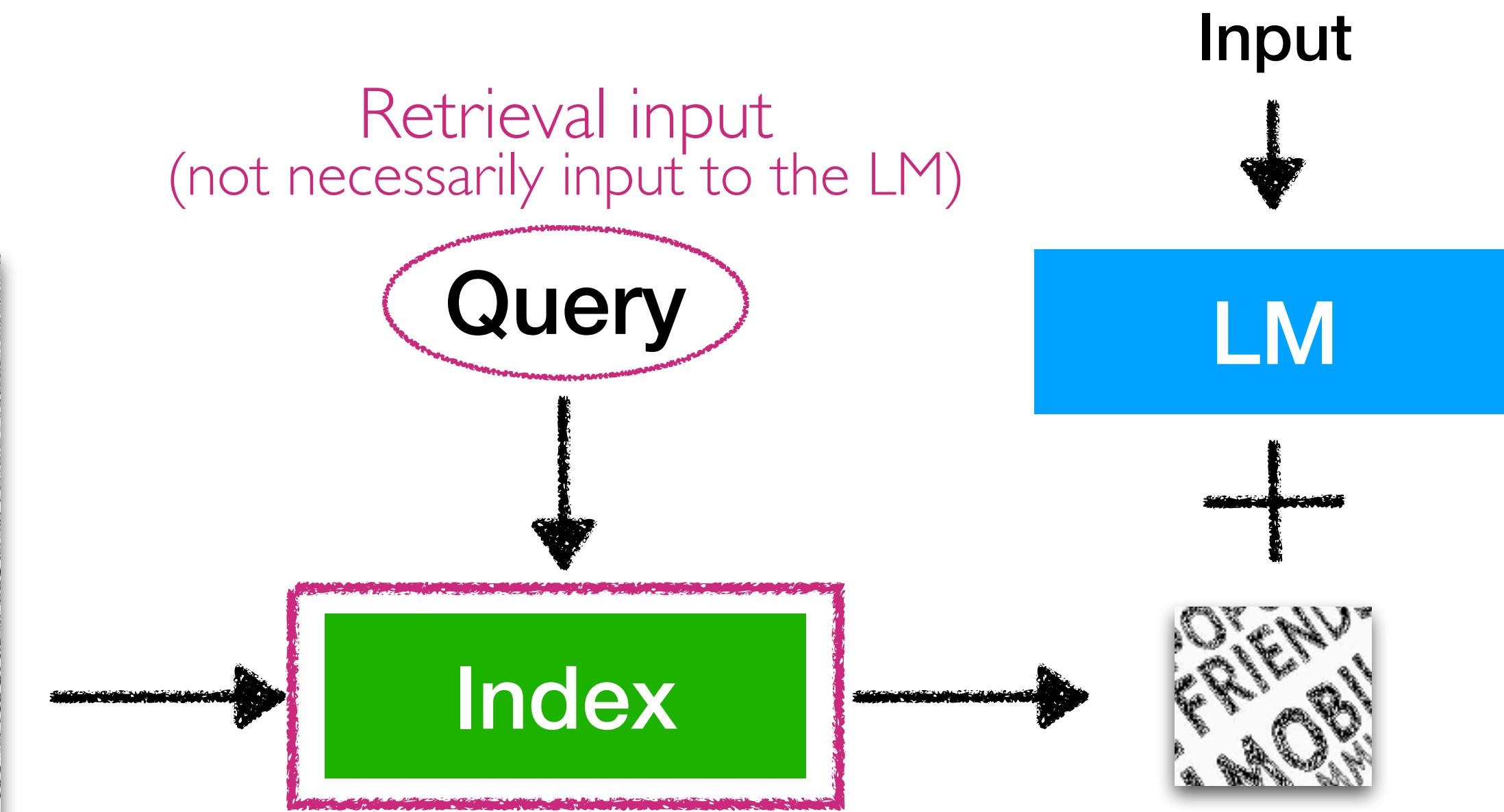
(Extending this definition is possible - some of them in a later part of Section 3)



Inference: Index



Datastore



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

Inference: Index

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

sim: a similarity score between two pieces of text

Example

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

of total docs
of docs containing i

Example

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

Maps the text into an h -dimensional vector

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

Inference: Index

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

sim: a similarity score between two pieces of text

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

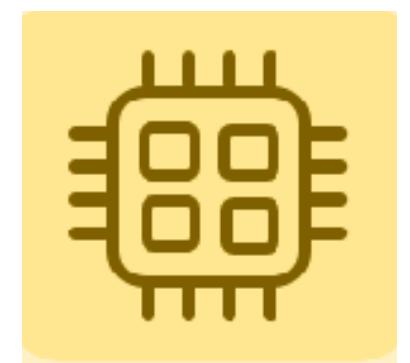
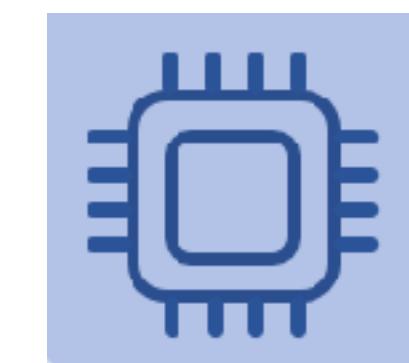
Index: given q , return $\arg \text{Top-}k_{d \in \mathcal{D}} \text{sim}(q, d)$ through fast nearest neighbor search

k elements from a datastore

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

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

Exact Search

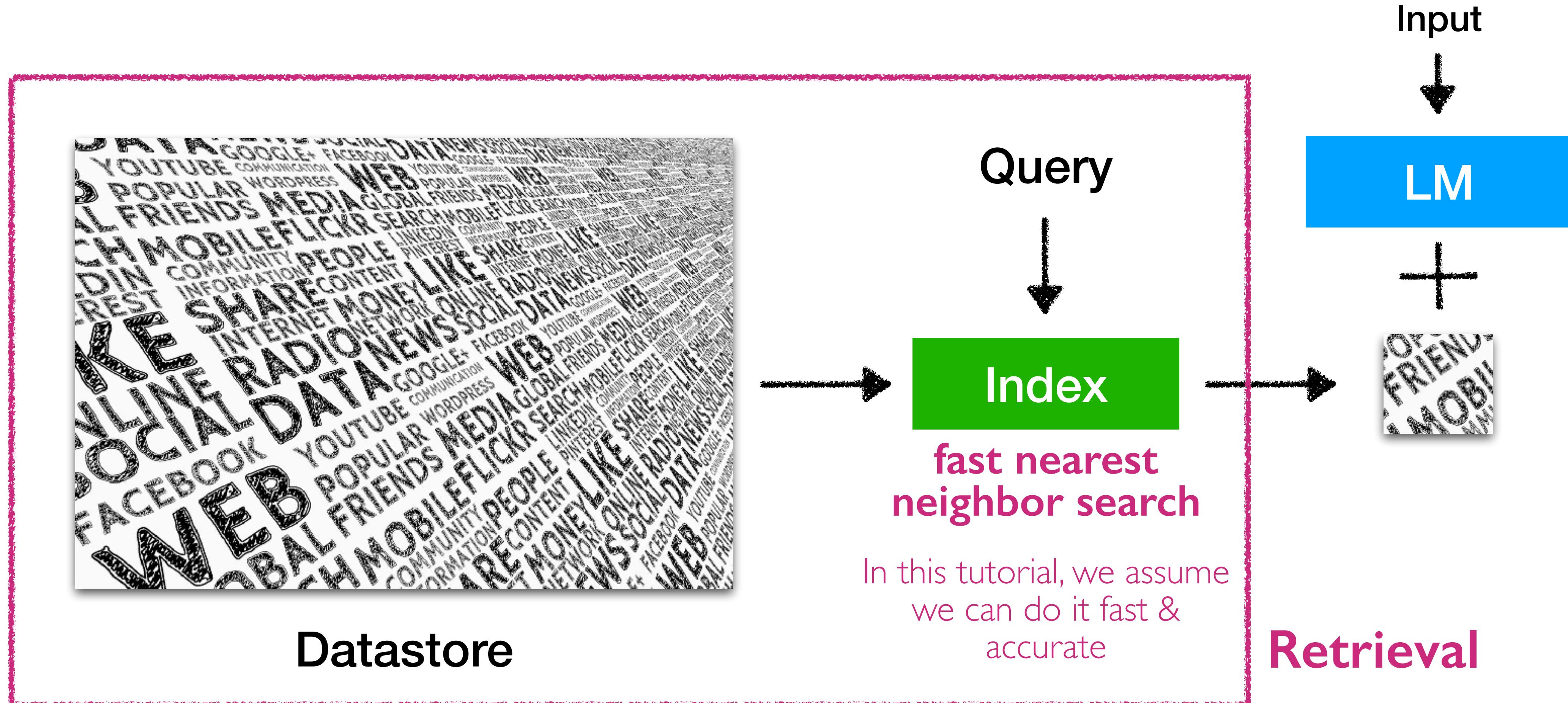


CPU vs. GPU

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

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

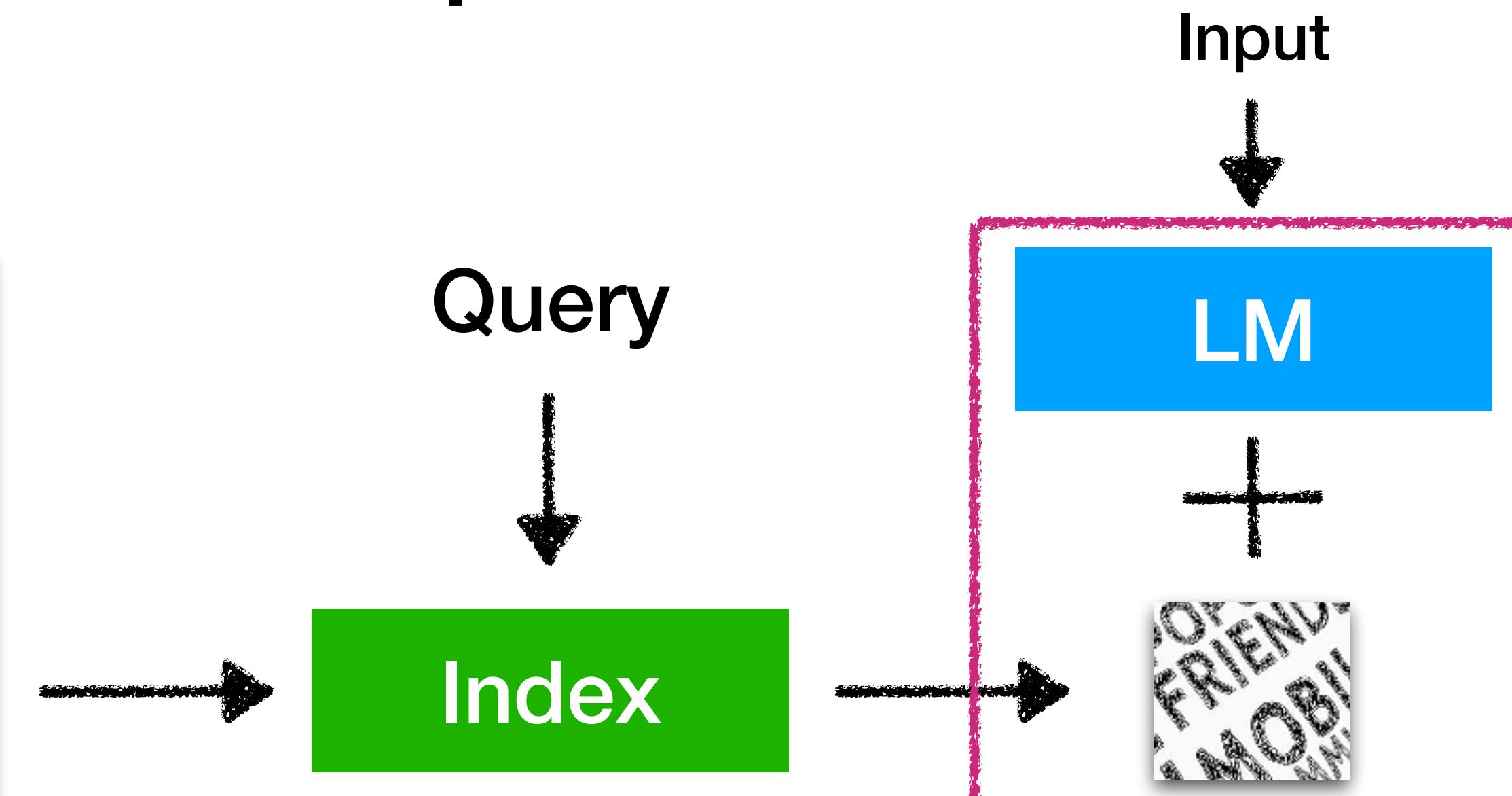
Inference: Search



Inference: Incorporation



Datastore

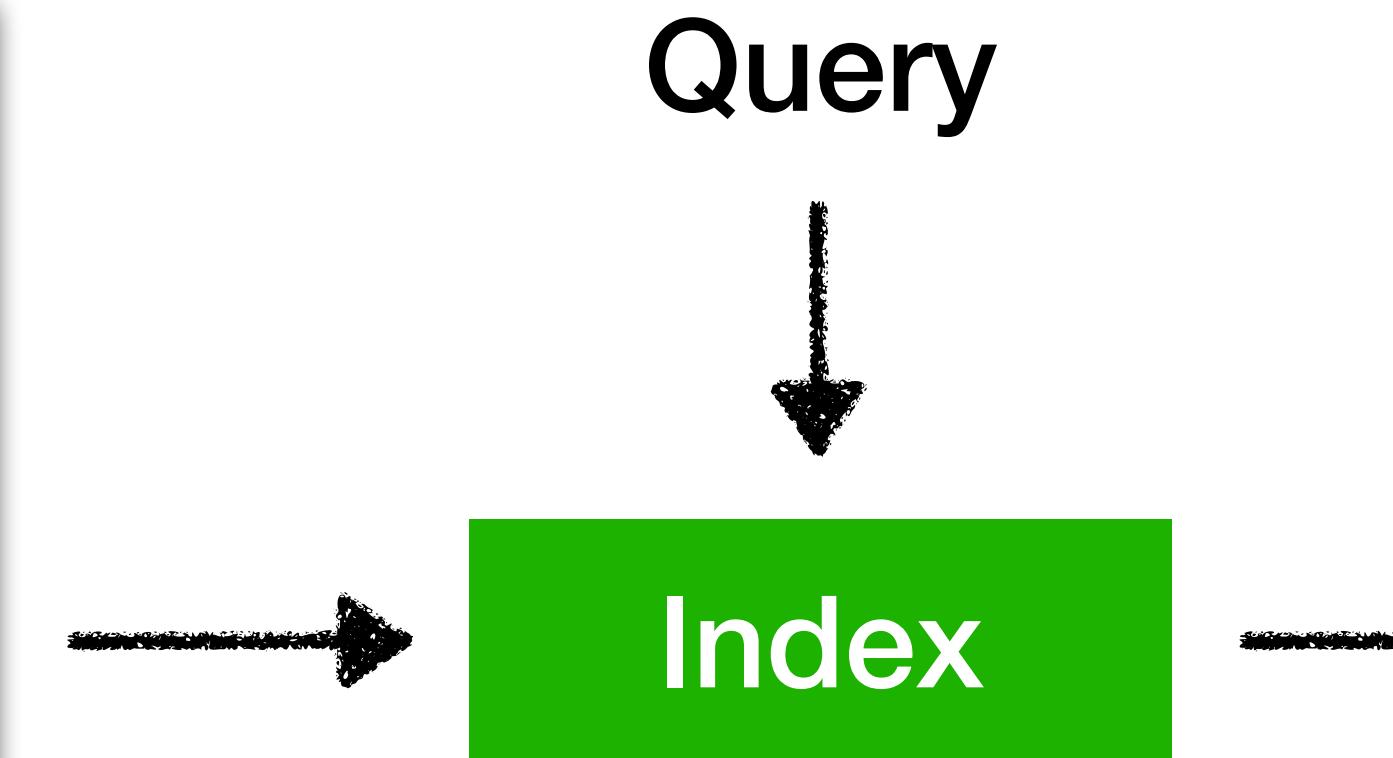


Questions to answer



Datastore

What's the query & when do we retrieve?



Input

How do we use retrieval?

What do we retrieve?

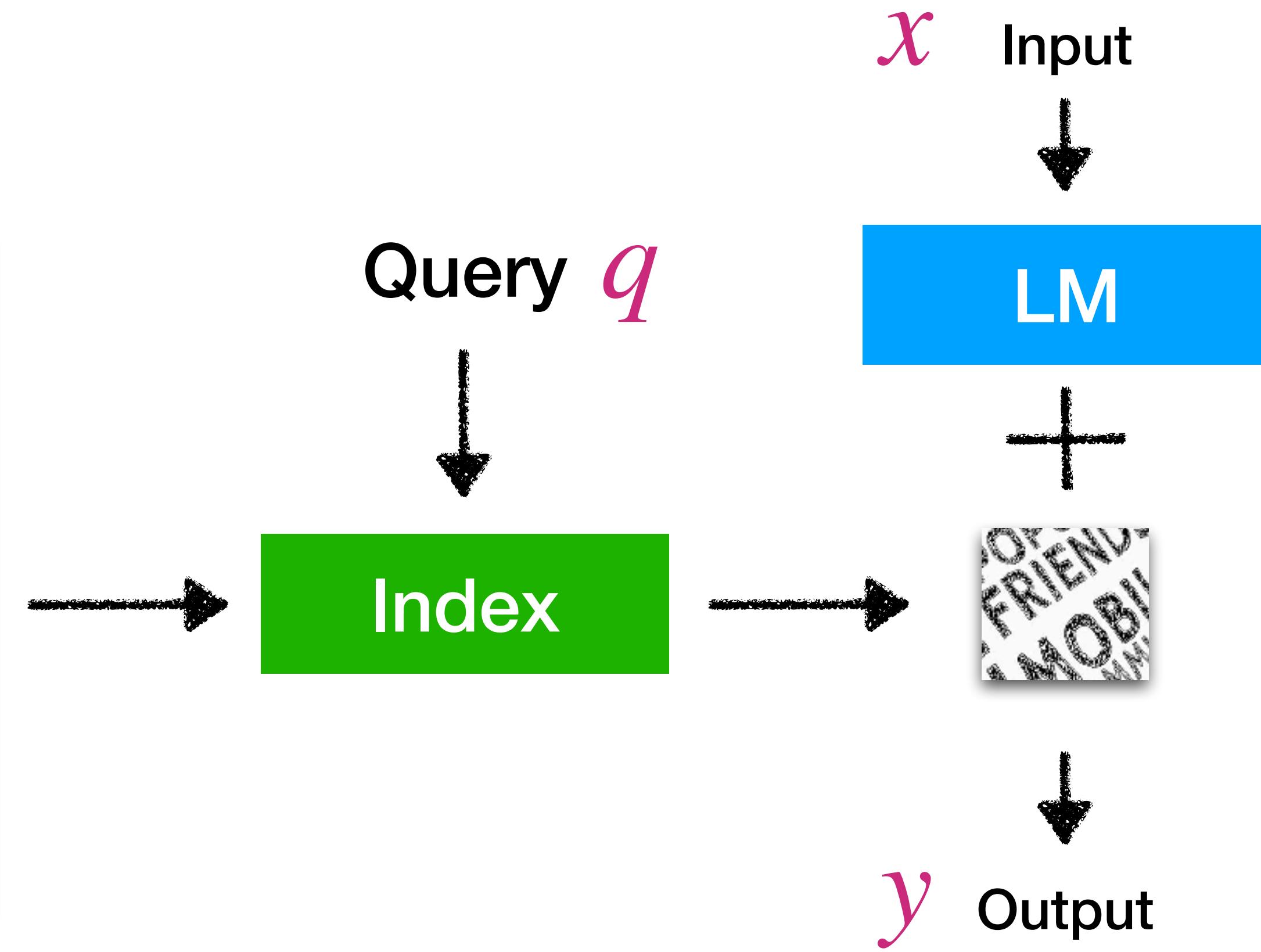
We'll answer these questions in Section 3!

Notations



Datastore

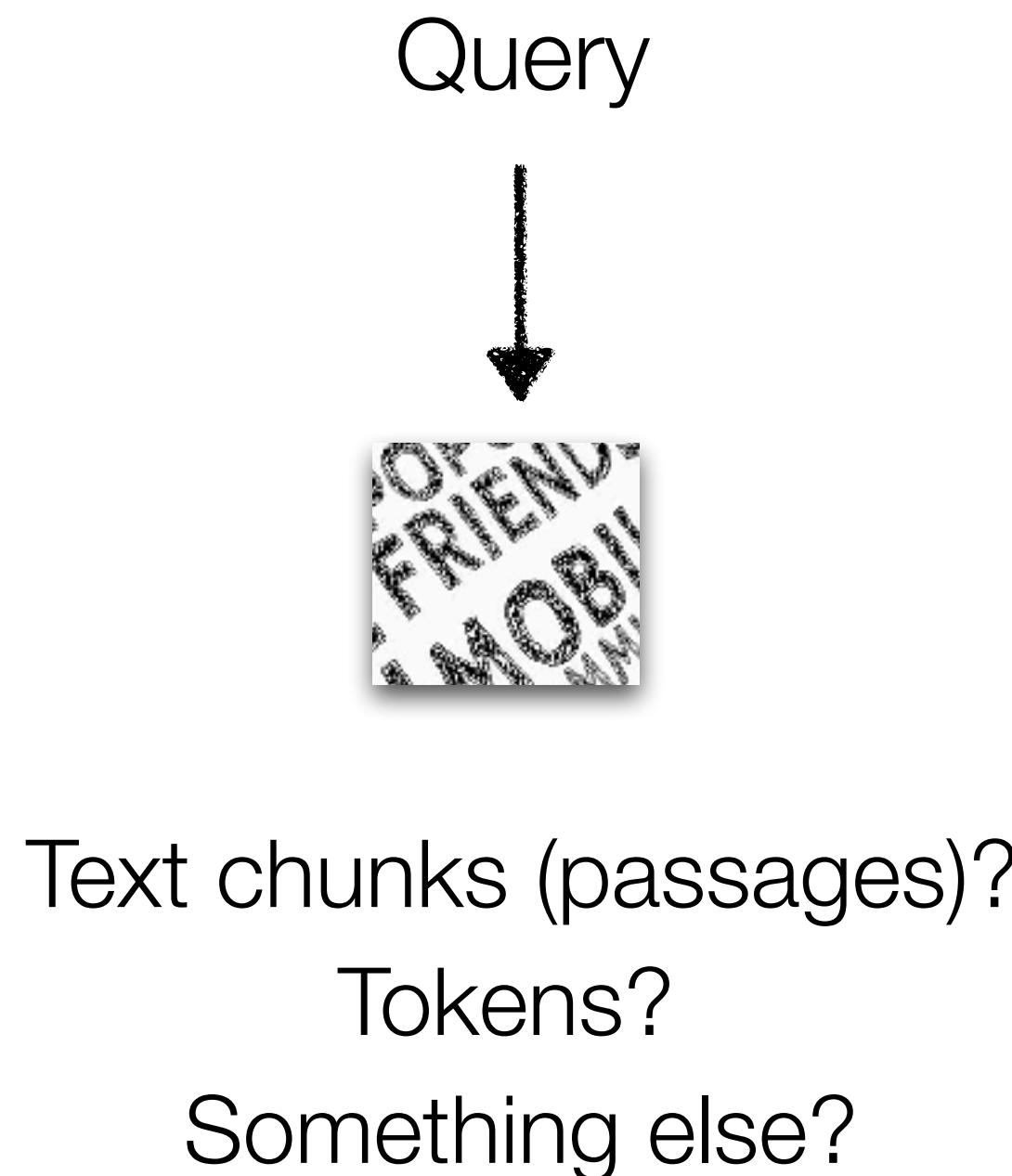
9



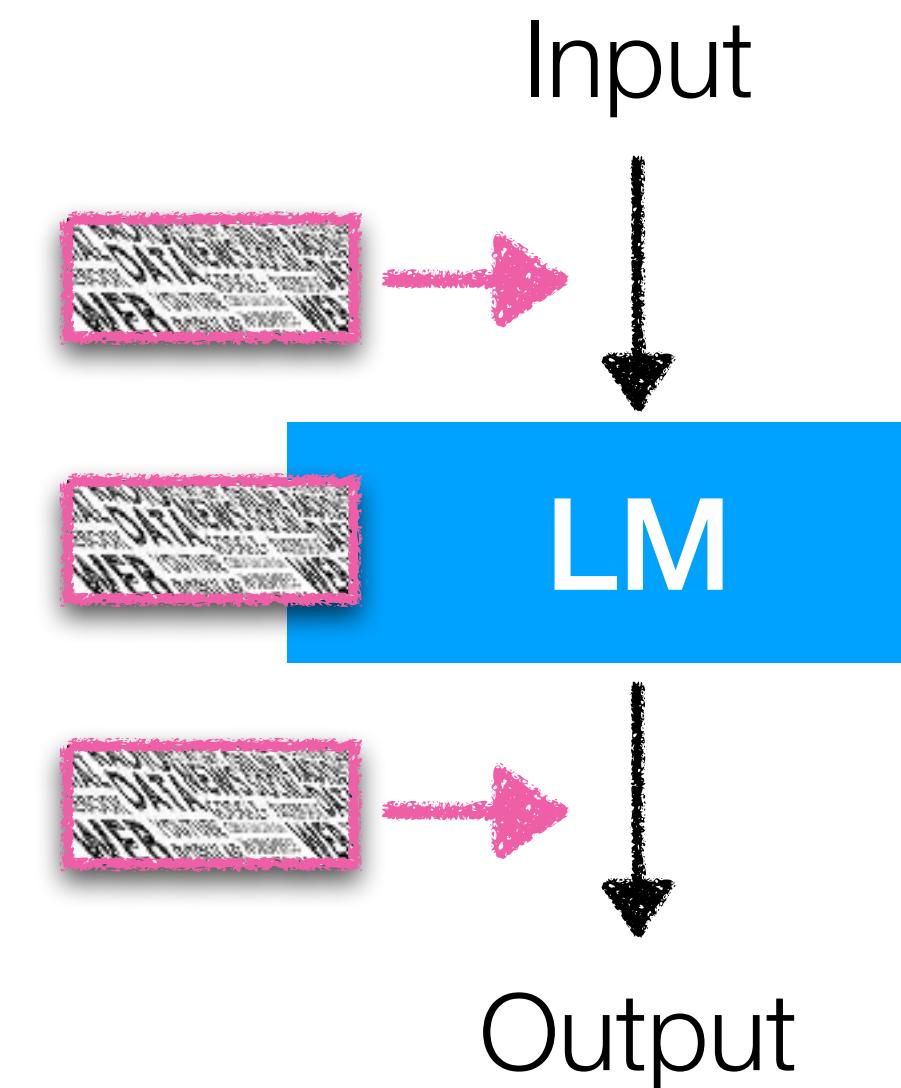
Section 3: Retrieval-based LM:Architecture

Categorization of retrieval-based LMs

What to retrieve?



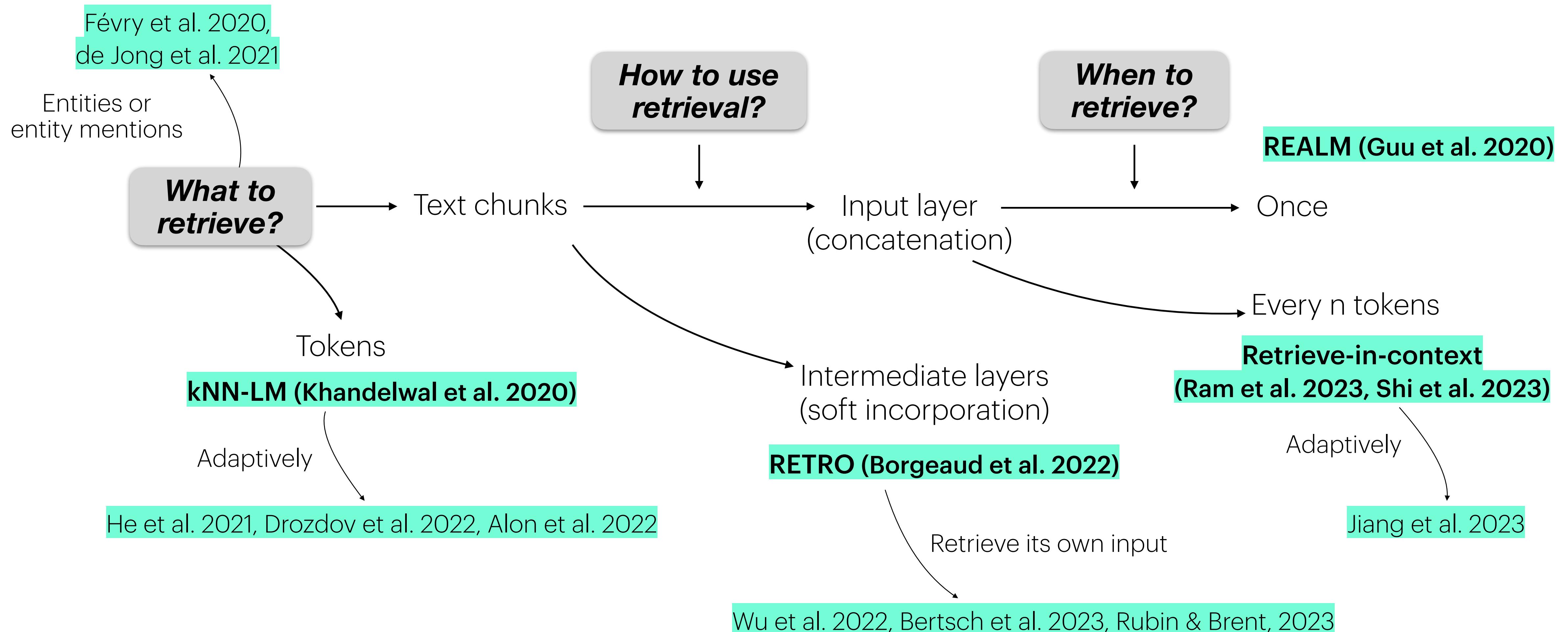
How to use retrieval?



When to retrieve?



Roadmap

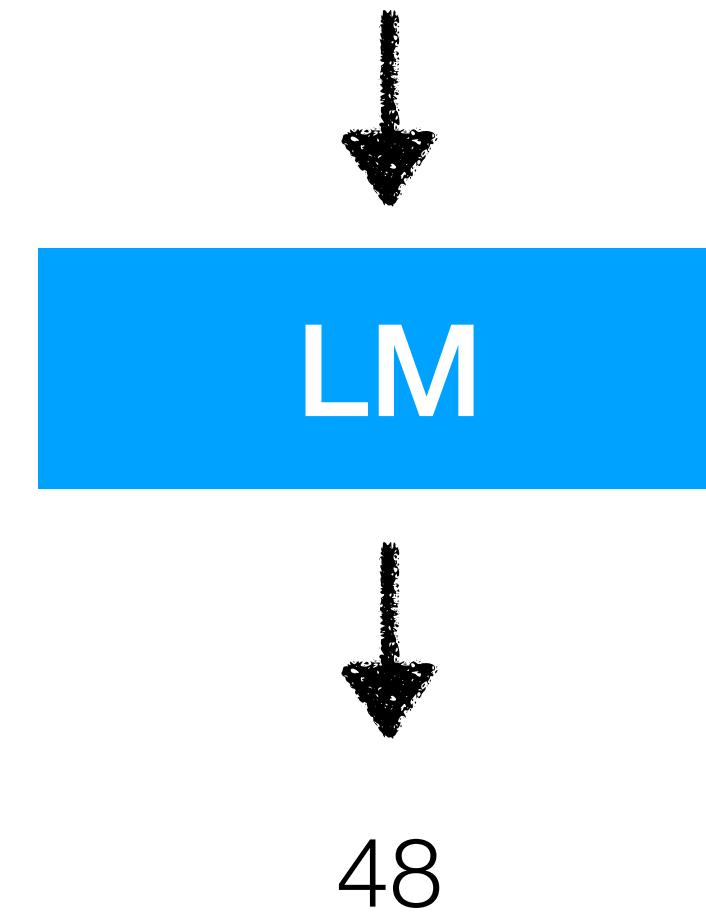


This is only about “architecture”
Section 4 will categorize & discuss “training”

REALM (Guu et al 2020)

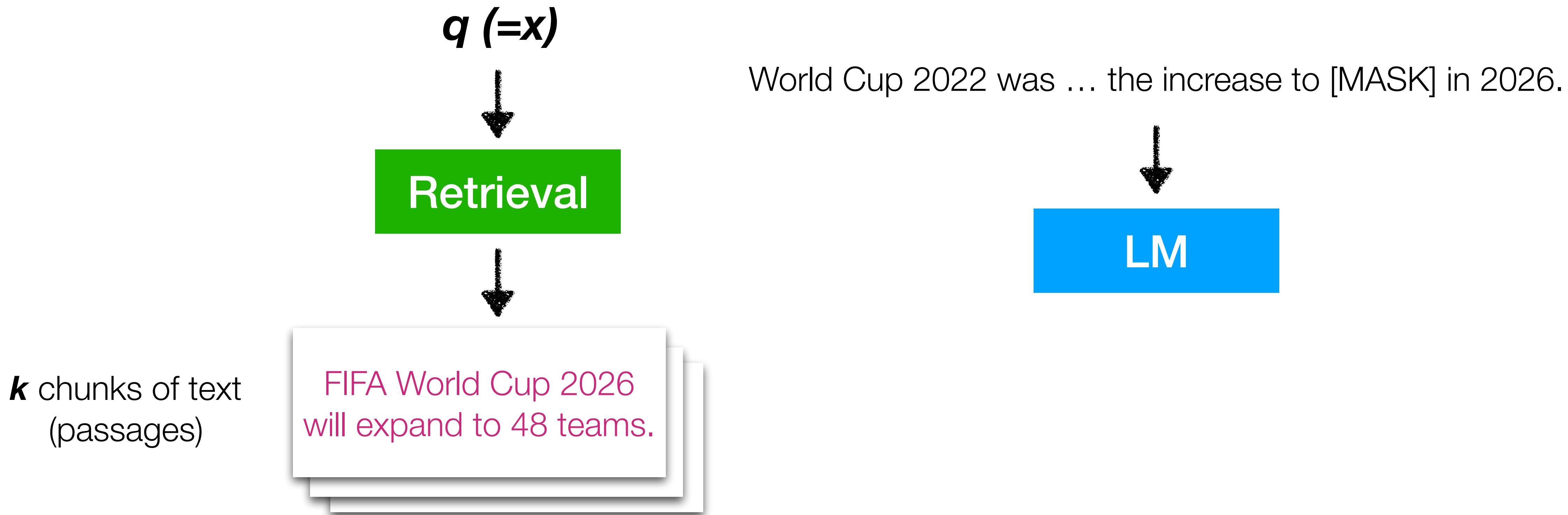
x = World Cup 2022 was the last with 32 teams before the increase to [MASK] in 2026.

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



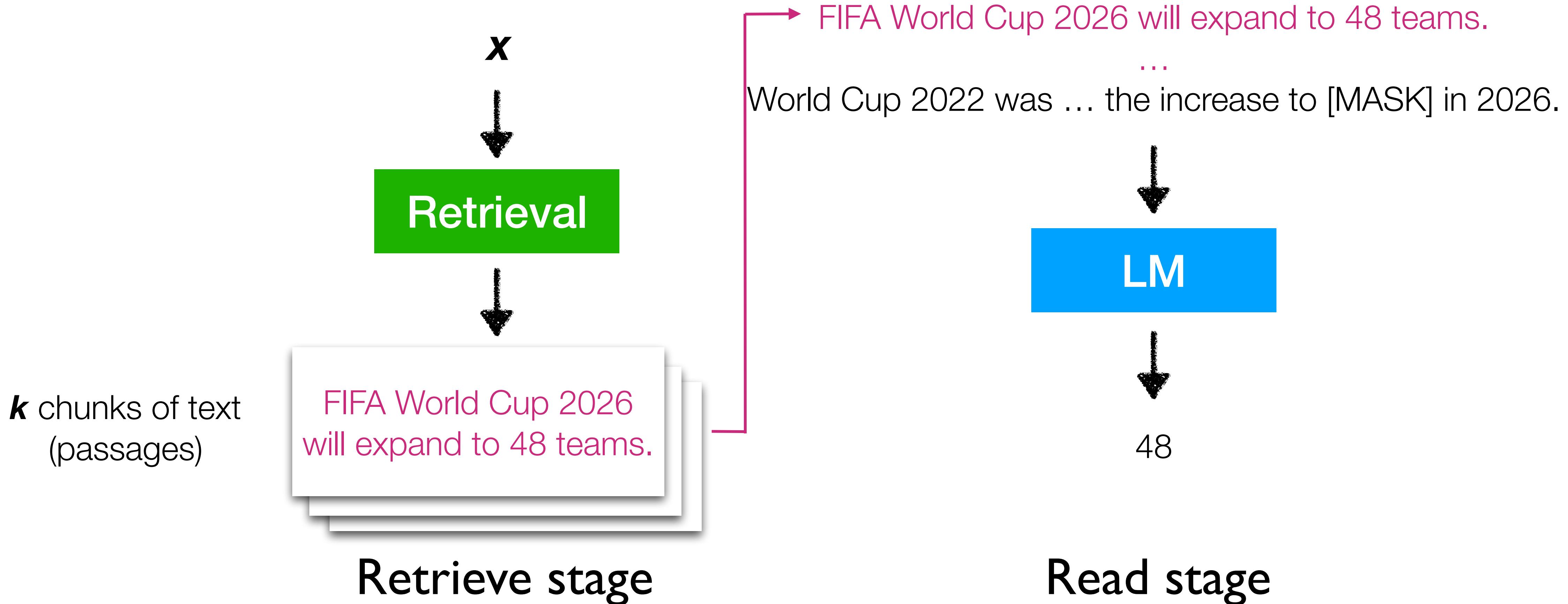
REALM (Guu et al 2020)

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



REALM (Guu et al 2020)

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

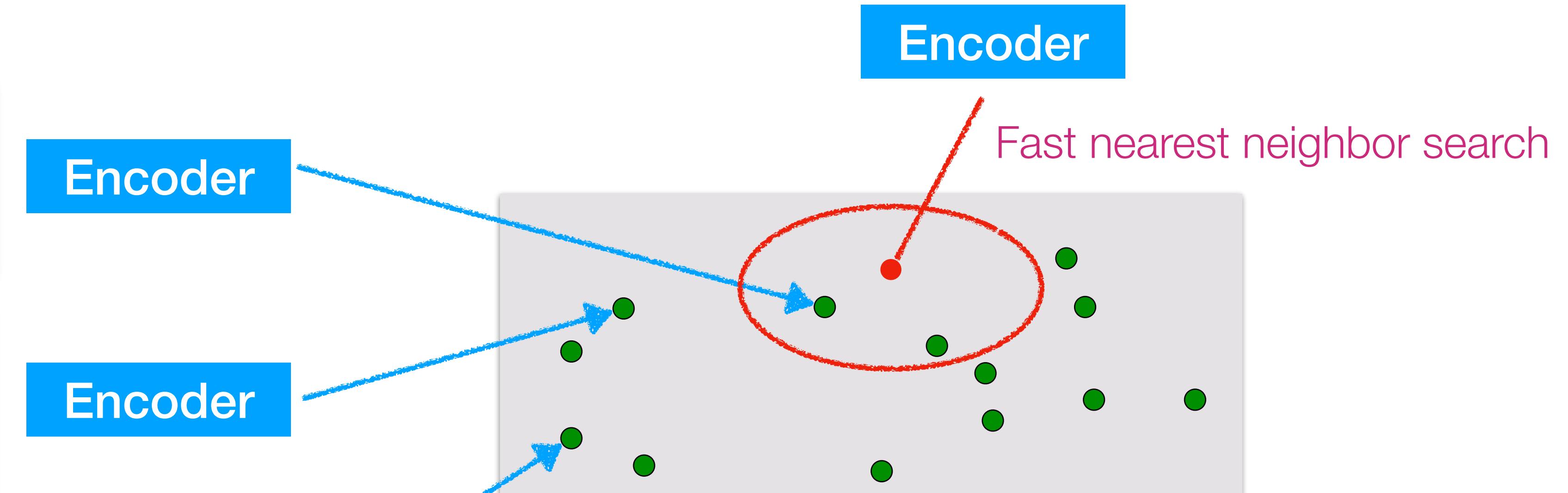


REALM: (I) Retrieve stage

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

- FIFA World Cup 2026 will expand to 48 teams.
- In 2022, the 32 national teams involved in the tournament.
- Team USA celebrated after winning its match against Iran ...

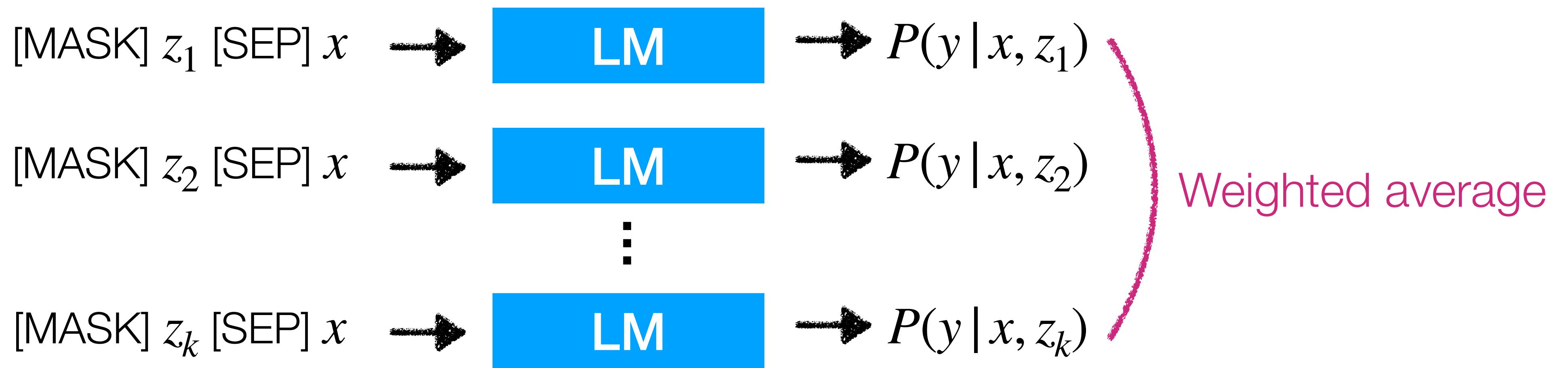
Wikipedia
13M chunks (passages)
(called *documents* in the paper)



$$\begin{aligned}\mathbf{z} &= \text{Encoder}(\mathbf{z}) \\ \mathbf{x} &= \text{Encoder}(\mathbf{x})\end{aligned}$$

$$\begin{aligned}z_1, \dots, z_k &= \text{argTop-}k(\mathbf{x} \cdot \mathbf{z}) \\ k \text{ retrieved blocks}\end{aligned}$$

REALM: (2) Read stage



Need to approximate
→ Consider top k chunks only

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

from the retrieve stage from the read stage

0 if not one of top k

REALM (Guu et al 2020)

What to retrieve?

- Chunks
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

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

REALM (Guu et al 2020)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

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

REALM (Guu et al 2020)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer ✓
- Intermediate layers
- Output layer

When to retrieve?

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

REALM (Guu et al 2020)

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer ✓
- Intermediate layers
- Output layer

When to retrieve?

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

REALM and subsequent work

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

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

REALM and subsequent work

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

Retrieval-in-context in LM

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

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



Retrieval



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

FIFA World Cup 2026 will expand to 48 teams.

Retrieval-in-context in 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.

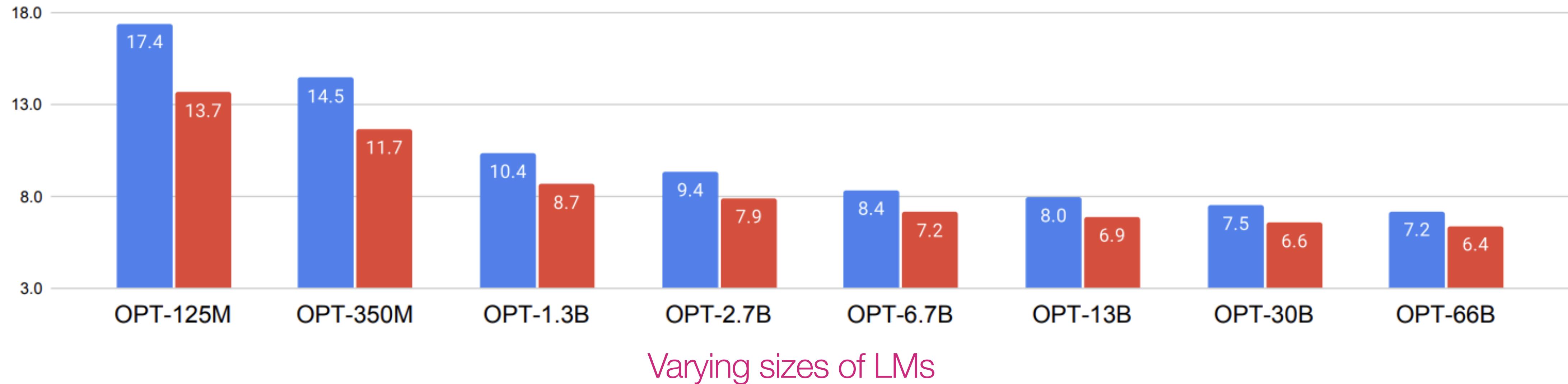
Ram et al. 2023. “In-Context Retrieval-Augmented Language Models”

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

Retrieval-in-context in LM

Perplexity: The lower the better

■ No Retrieval ■ In-Context RALM (BM25)



Varying sizes of LMs

Retrieval helps over all sizes of LMs

Graphs from Ram et al. 2023

Retrieval-in-context in LM

Is $q=x$ necessary?

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

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

Retrieval

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

Does not cover “tokens that will come next”

Retrieval-in-context in LM

Is $q=x$ necessary?

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

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



Retrieval

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

Does not cover “tokens that will come next”

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

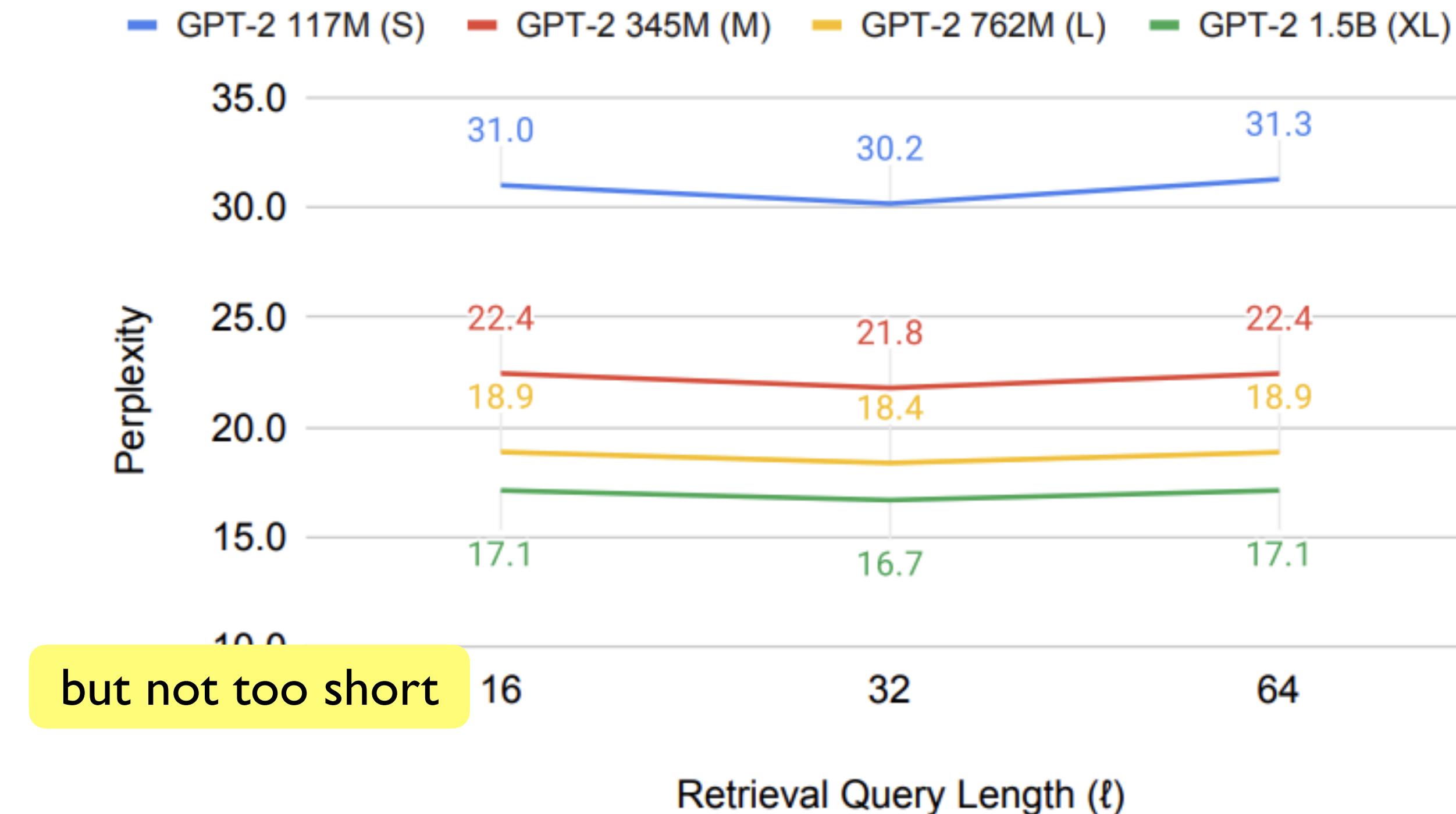


Retrieval

FIFA World Cup 2026 will expand to 48 teams.

more relevant to what will come next

Retrieval-in-context in LM



Shorter prefix (more recent tokens) as a query helps

Graphs from Ram et al. 2023

Retrieval-in-context in 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 in 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 in 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 in 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 in 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

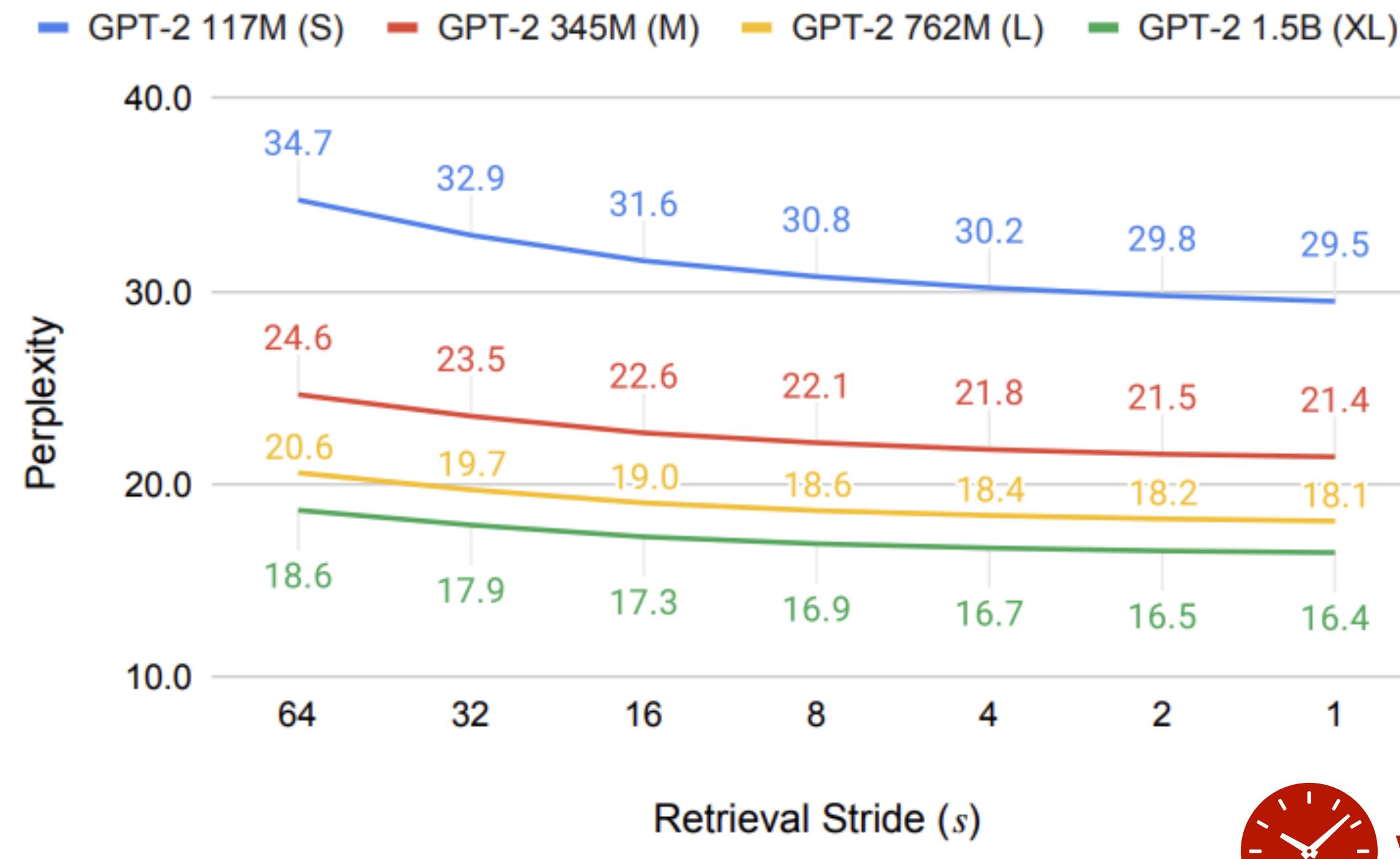


LM



to 48 in the 2026 tournament.
Retrieval results from a new query explain them!

Retrieval-in-context in LM



Retrieving more frequently helps



with cost in inference time

Graphs from Ram et al. 2023

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

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer
- Intermediate layers
- Output layer

When to retrieve?

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

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

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer ✓
- Intermediate layers
- Output layer

When to retrieve?

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

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

What to retrieve?

- Chunks ✓
- Tokens
- Others

How to use retrieval?

- Input layer ✓
- Intermediate layers
- Output layer

When to retrieve?

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

Summary

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

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

Summary

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

can be very inefficient to retrieve many text chunk, frequently

RETRO (Borgeaud et al. 2021)

- ✓ Incorporation in the “intermediate layer” instead of the “input” layer
→ designed for many blocks, frequently, more efficiently
- ✓ Scale the datastore to retrieve from

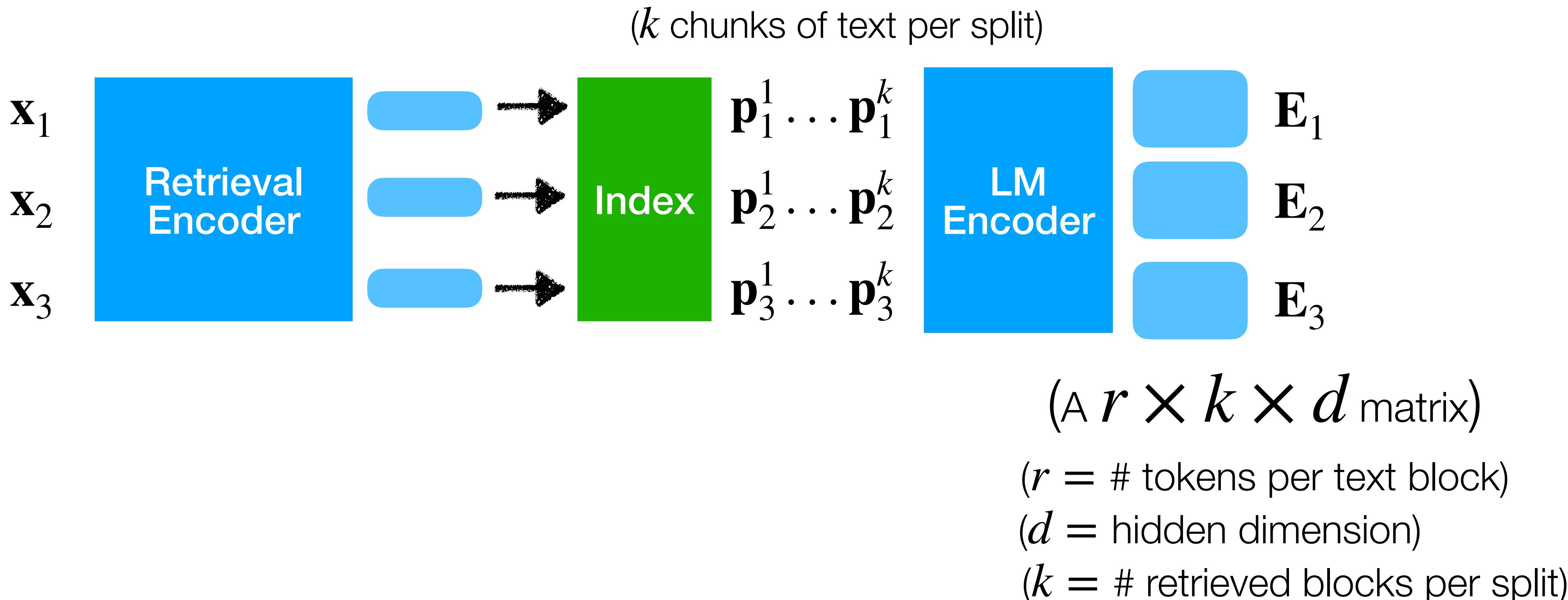
RETRO (Borgeaud et al. 2021)

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

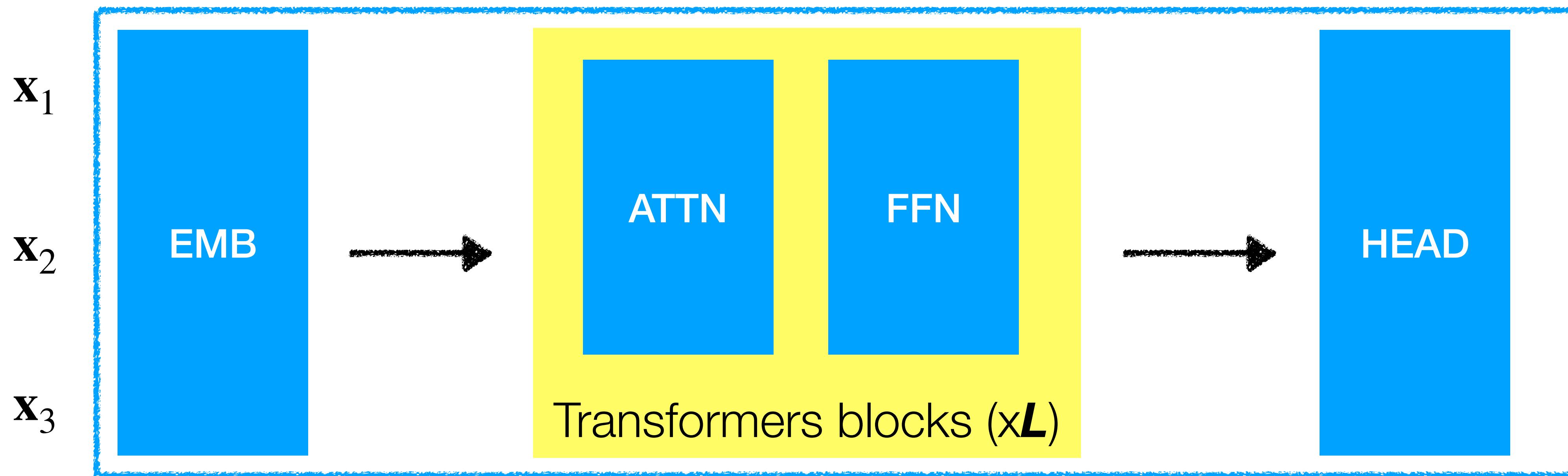
\mathbf{x}_1

\mathbf{x}_2

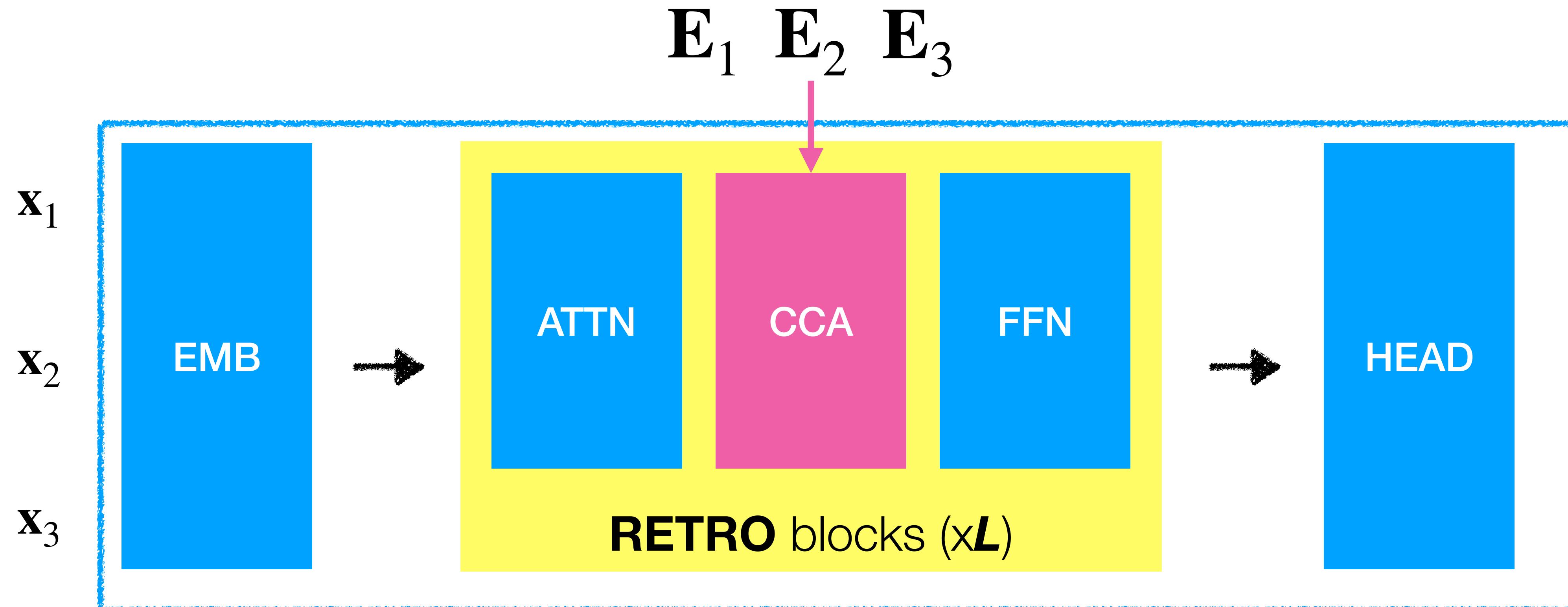
\mathbf{x}_3



Regular decoder

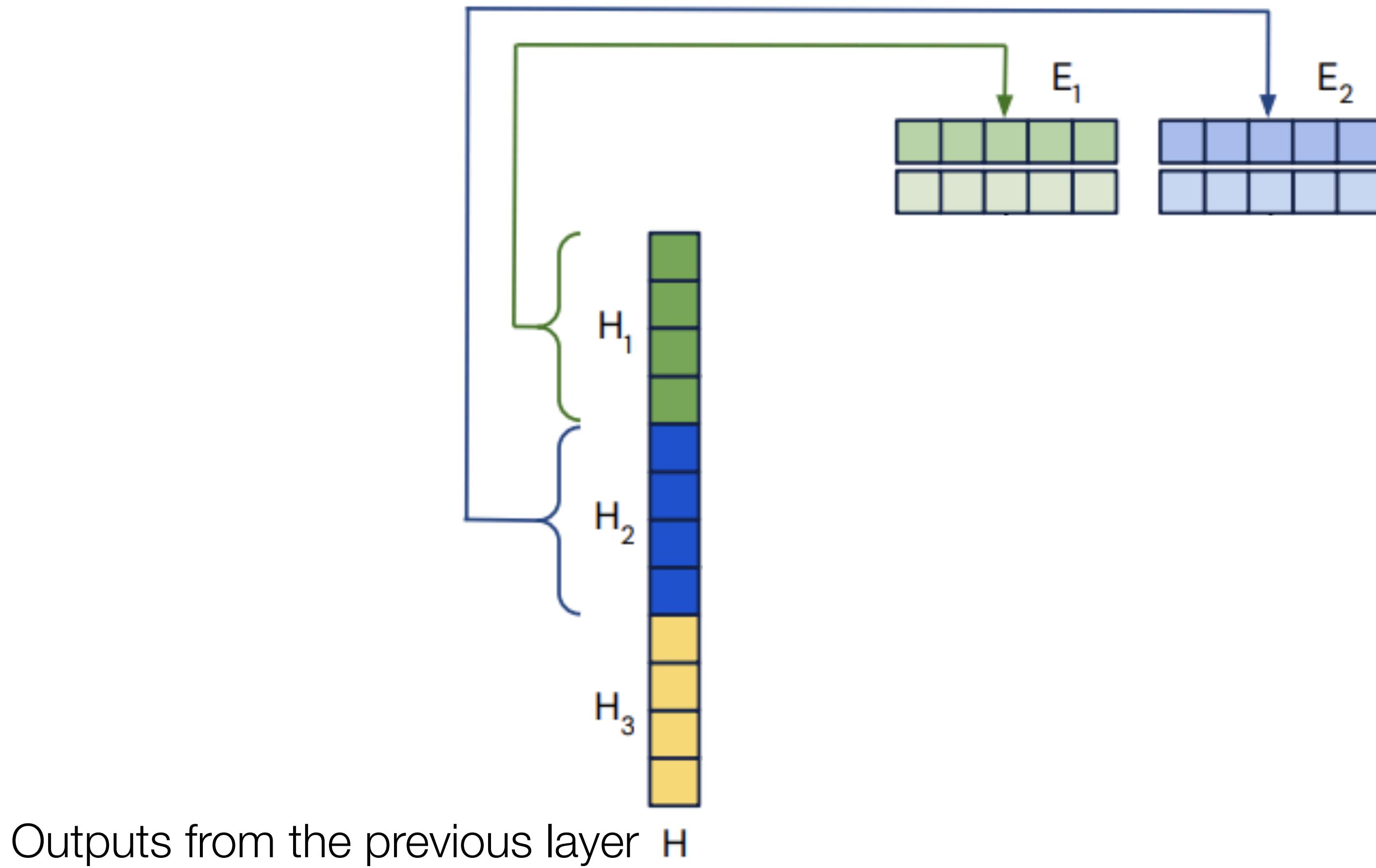


Decoder in RETRO

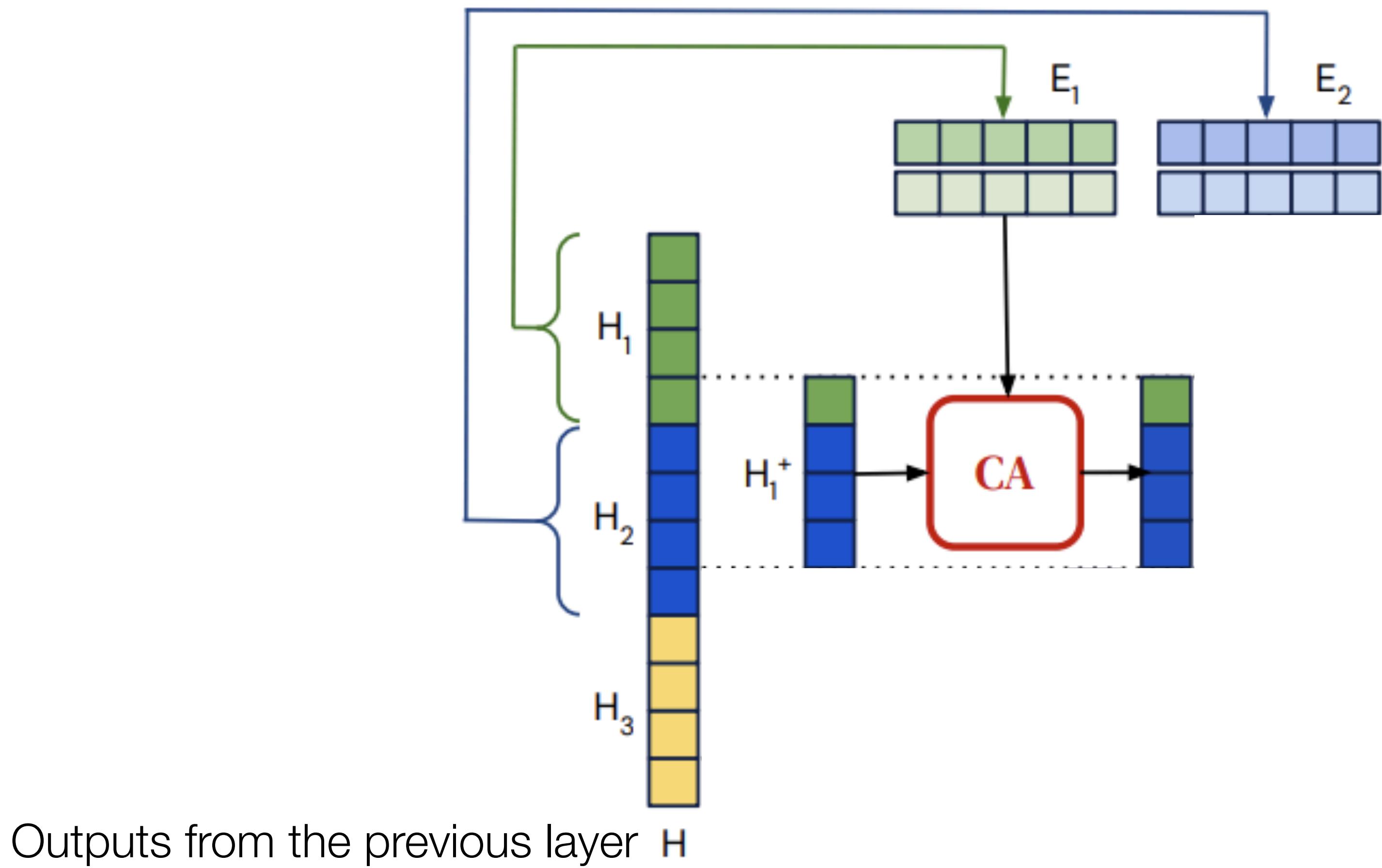


Chunked Cross Attention (CCA)

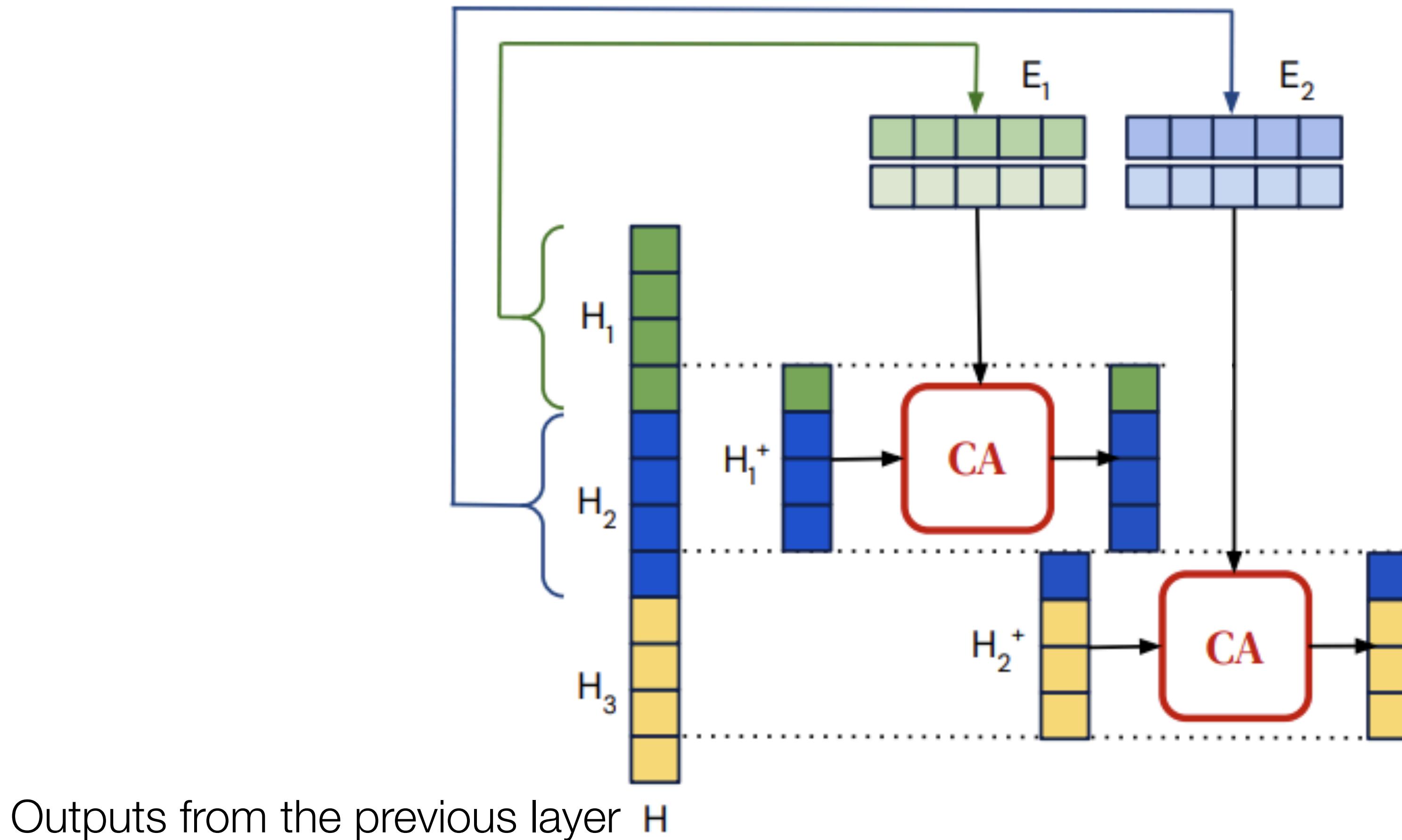
Chunked Cross Attention



Chunked Cross Attention

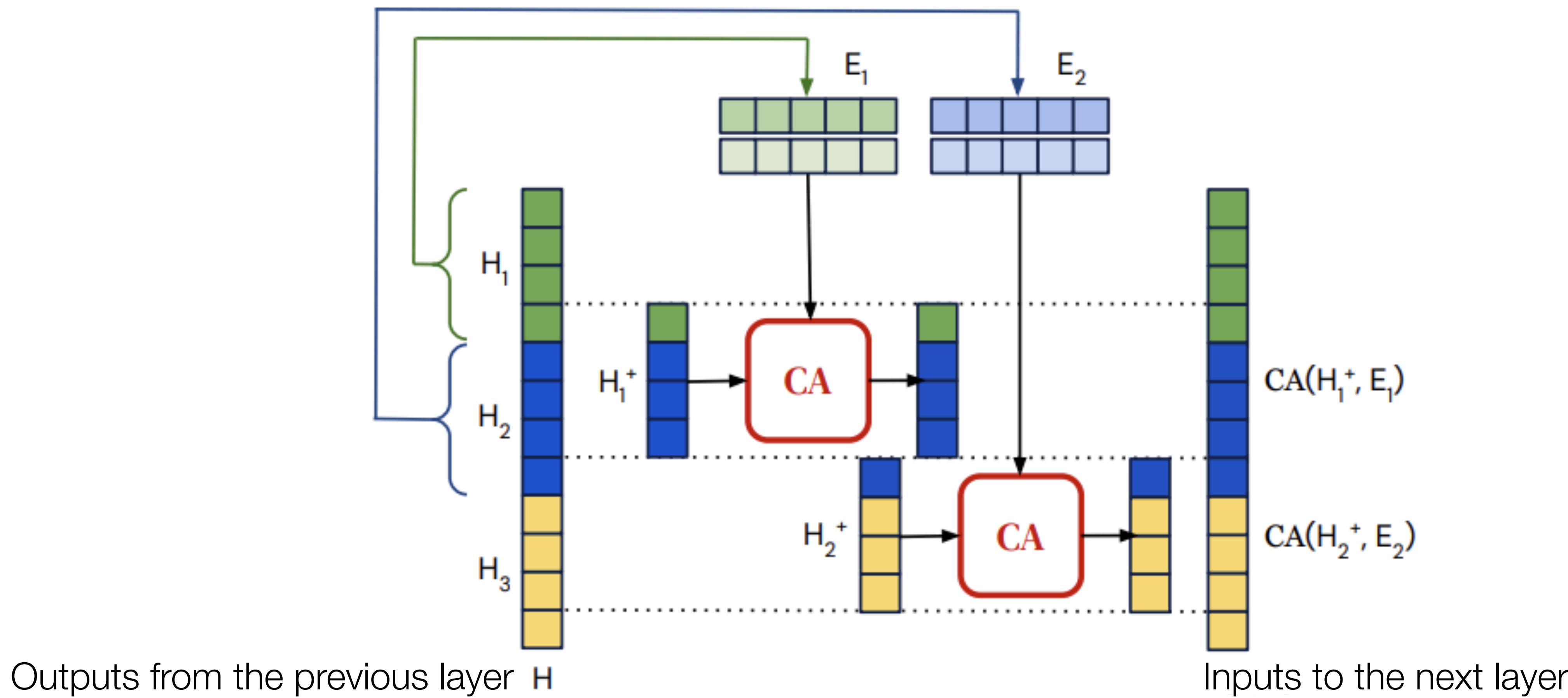


Chunked Cross Attention

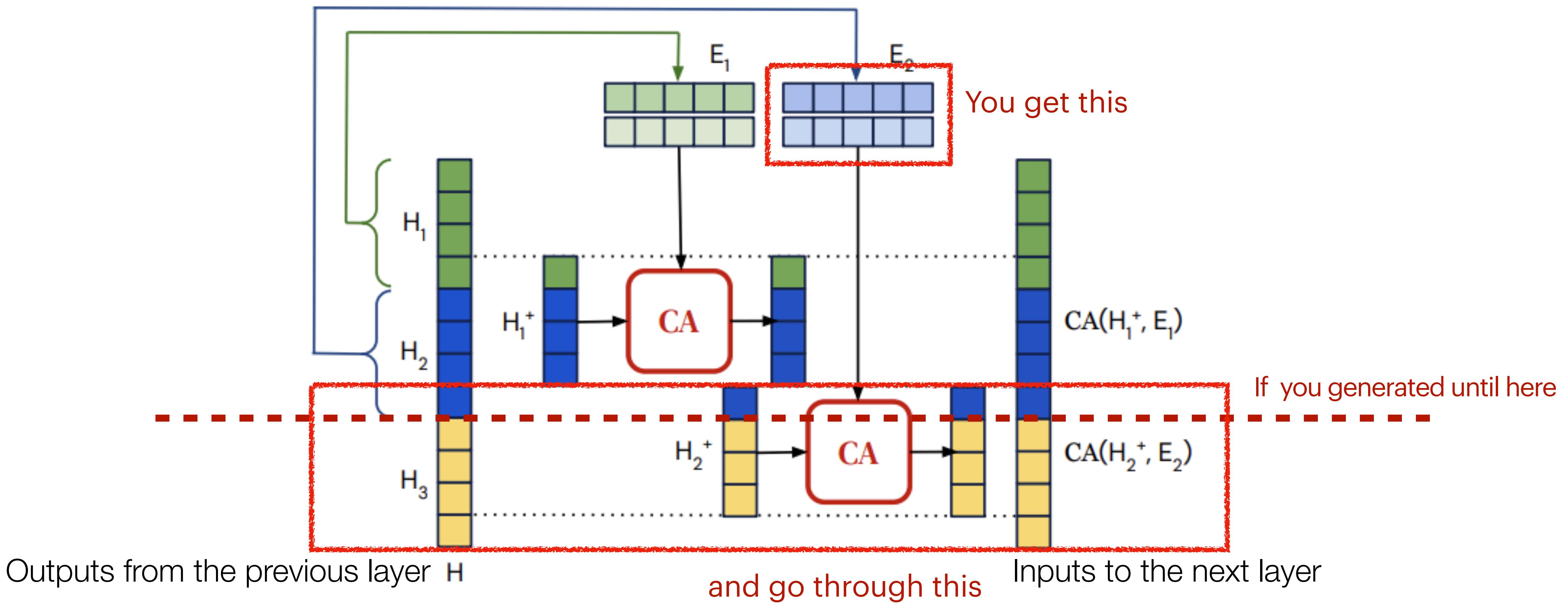


Outputs from the previous layer H

Chunked Cross Attention

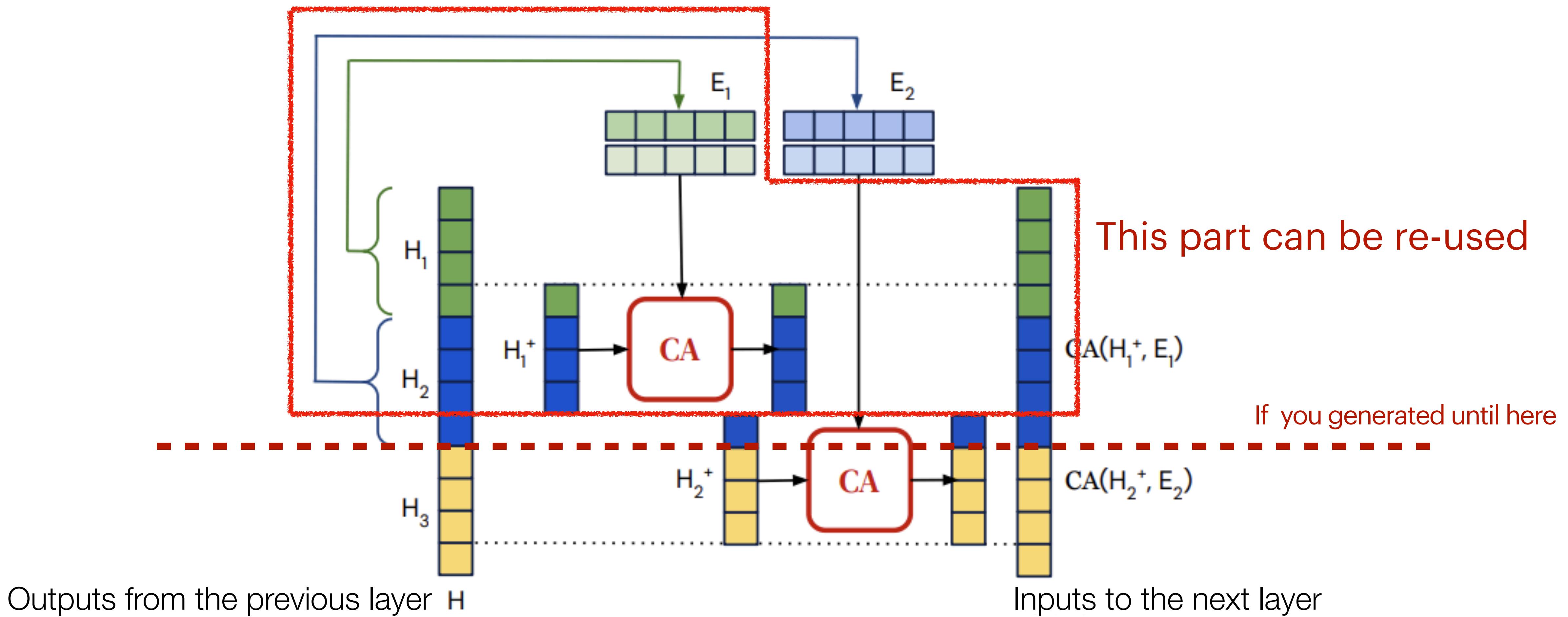


Chunked Cross Attention



✓ Cross-attention can be computed in parallel

Chunked Cross Attention



✓ Cross-attention can be computed in parallel

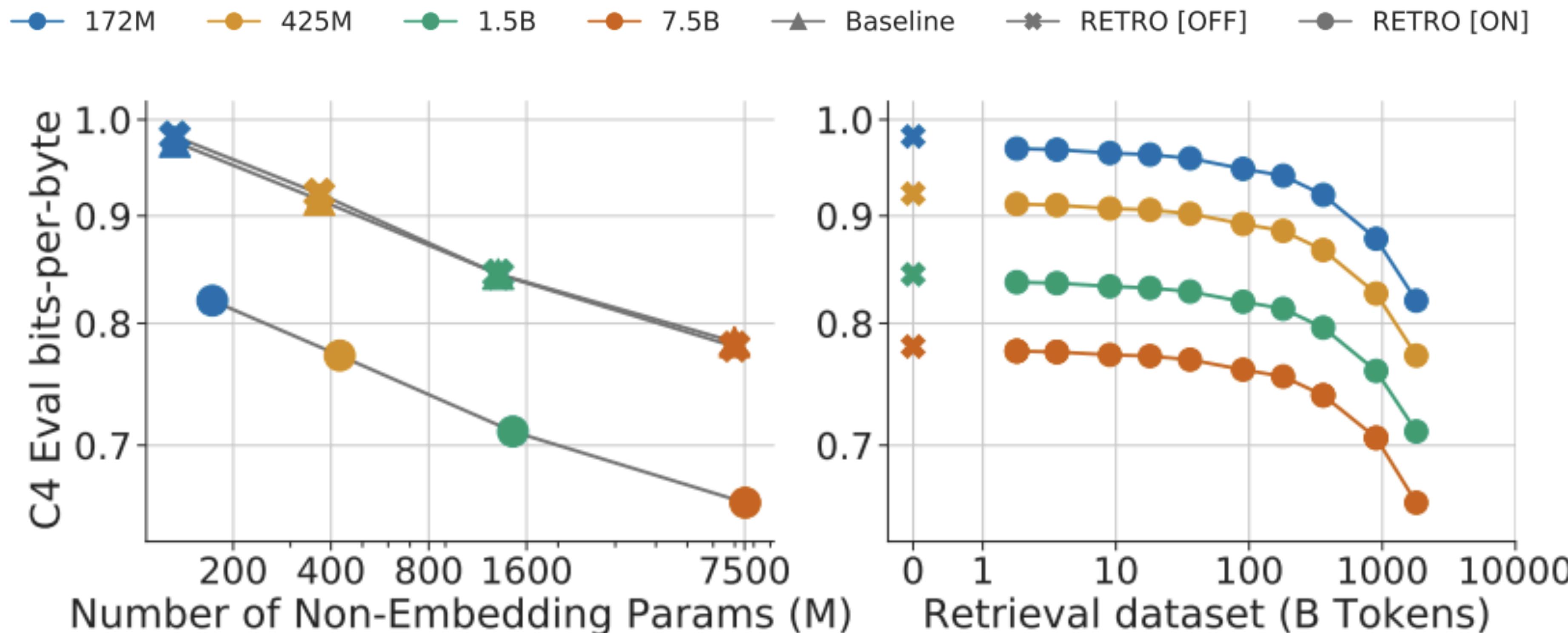
Results

Perplexity: The lower the better

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

Significant improvements by retrieving from 1.8 trillion tokens

Results



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	Input layer Intermediate layers	Every n tokens



Can use many blocks, more frequently, more efficiently



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

Summary

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

What else?

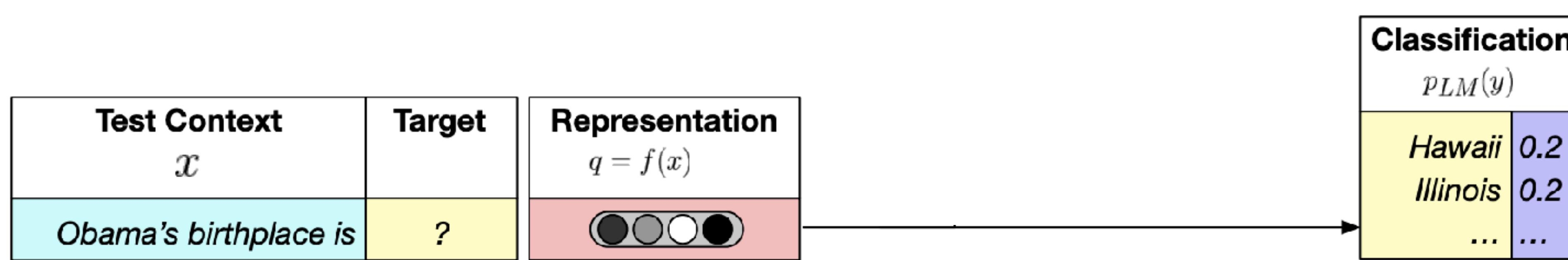
kNN-LM (Khandelwal et al. 2020)

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

kNN-LM (Khandelwal et al. 2020)

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

kNN-LM (Khandelwal et al. 2020)



kNN-LM (Khandelwal et al. 2020)

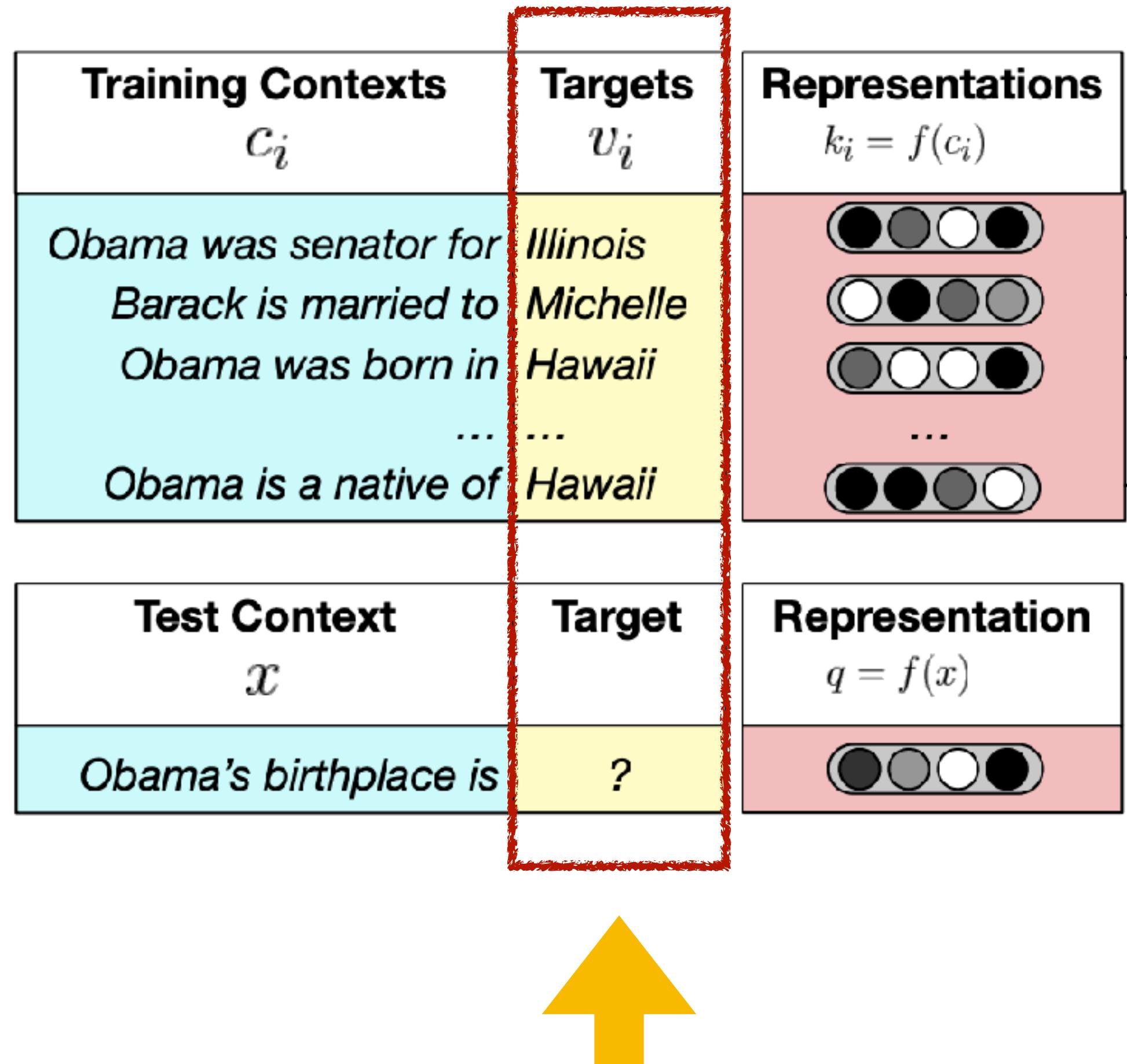
The size of the datastore = # of tokens in the corpus (>1B)

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

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

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

kNN-LM (Khandelwal et al. 2020)



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

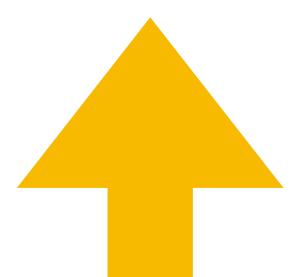
kNN-LM (Khandelwal et al. 2020)

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

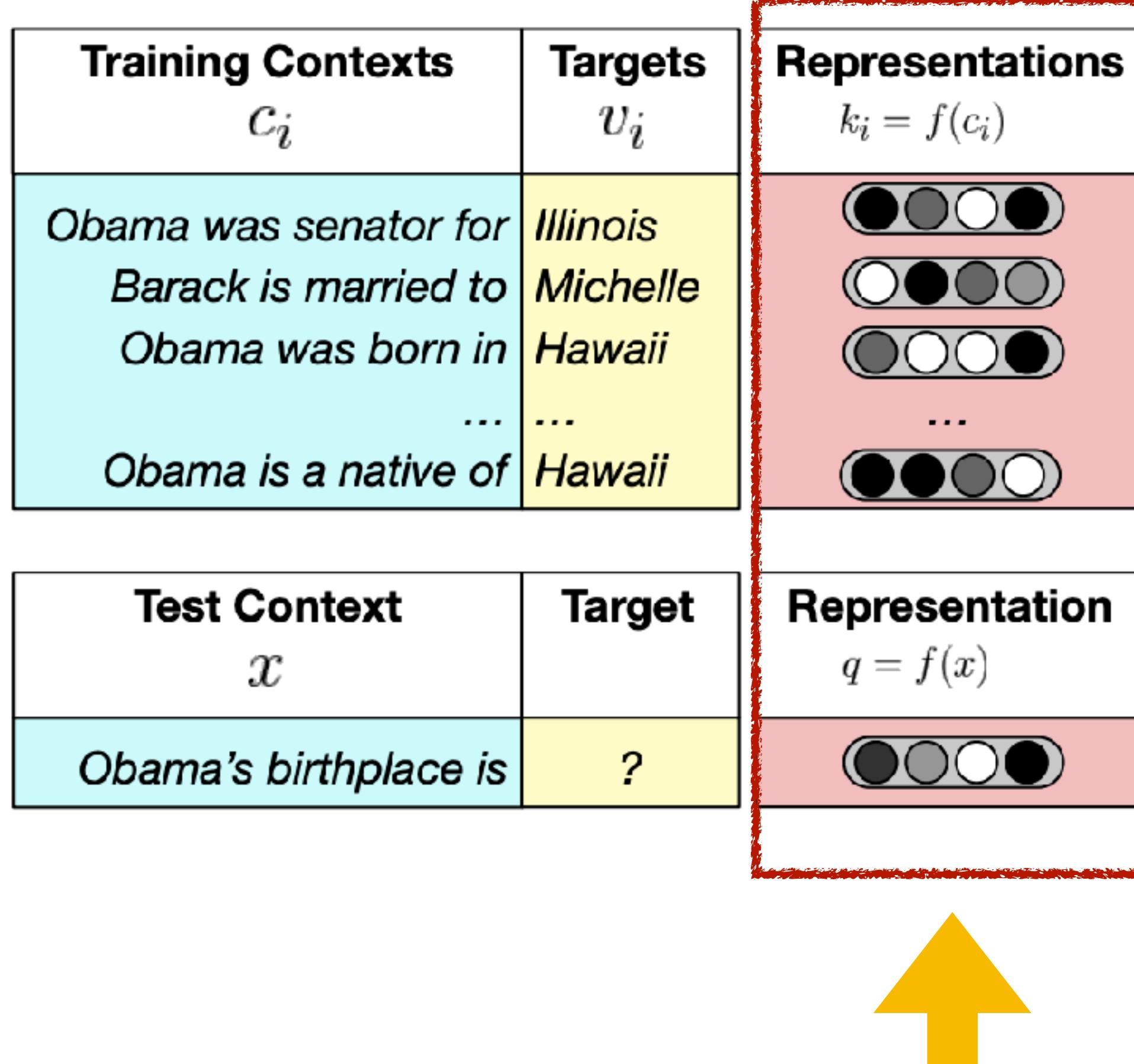
Which tokens in a datastore are close to the next token?

=

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



kNN-LM (Khandelwal et al. 2020)



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

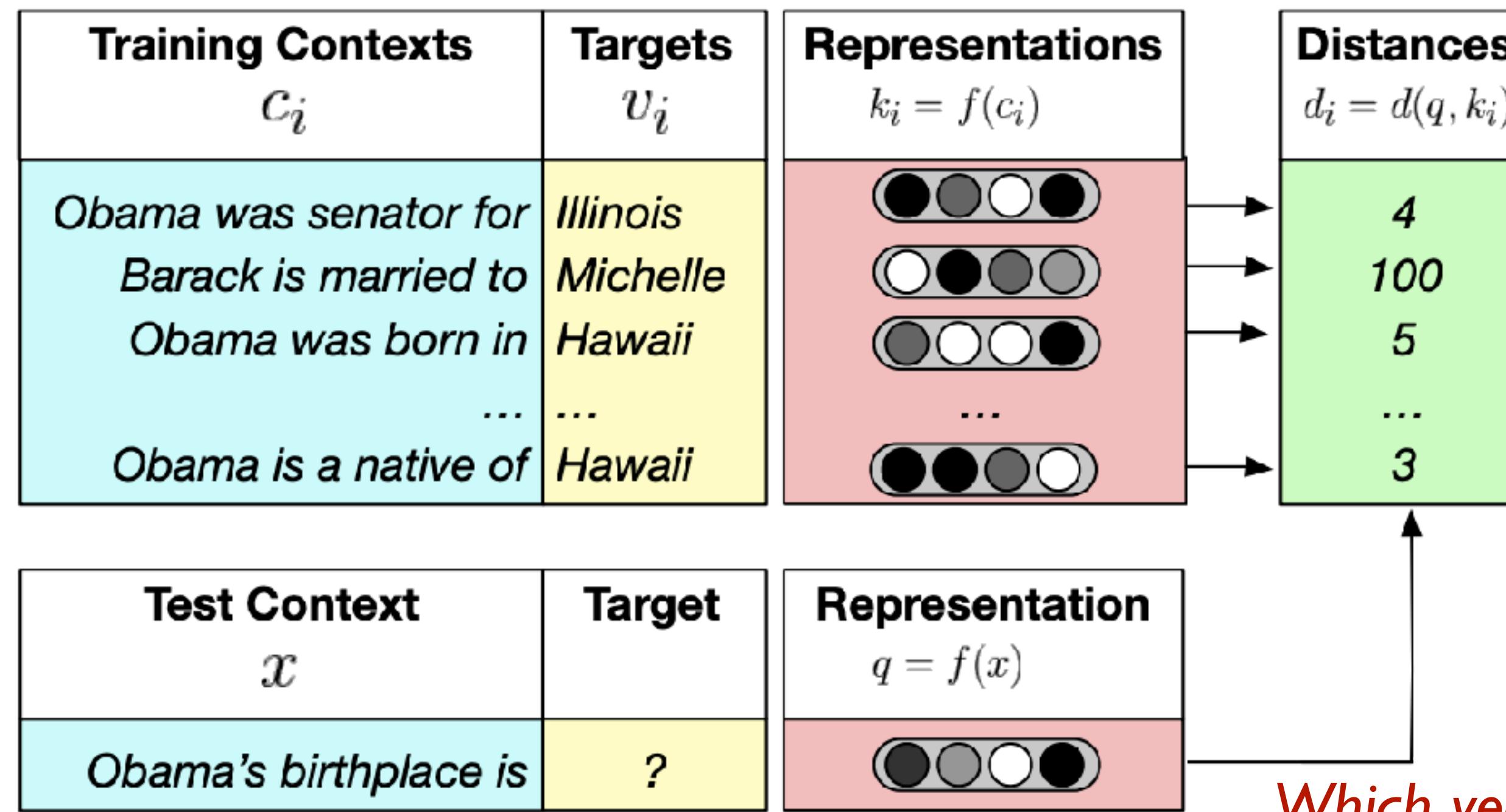
=

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

=

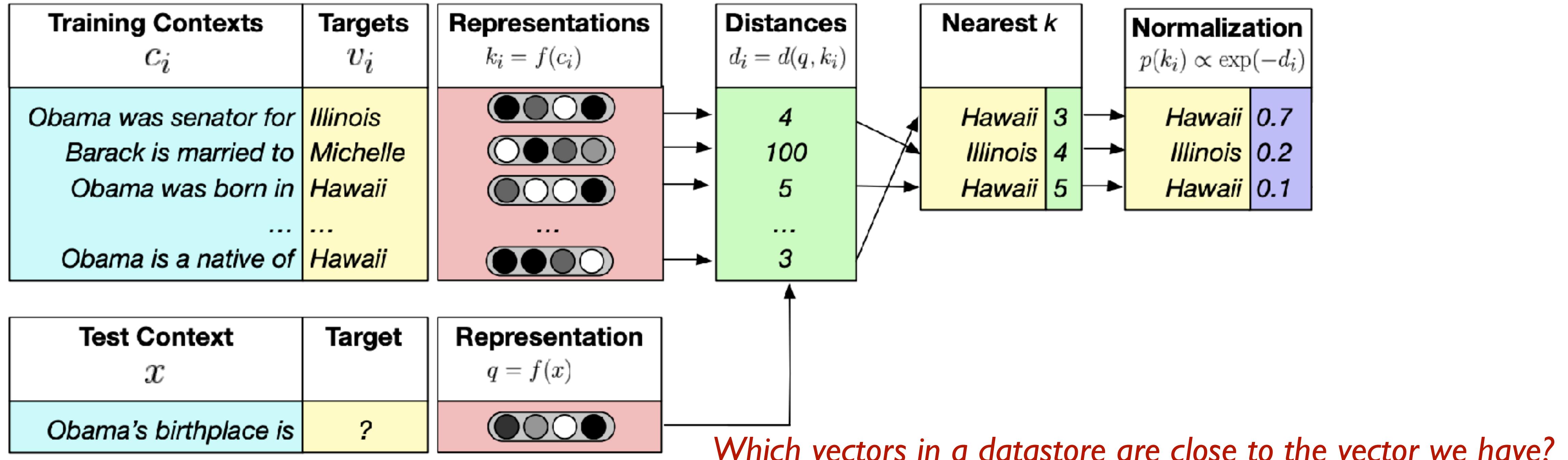
Which vectors in a datastore are close to the vector we have?

kNN-LM (Khandelwal et al. 2020)

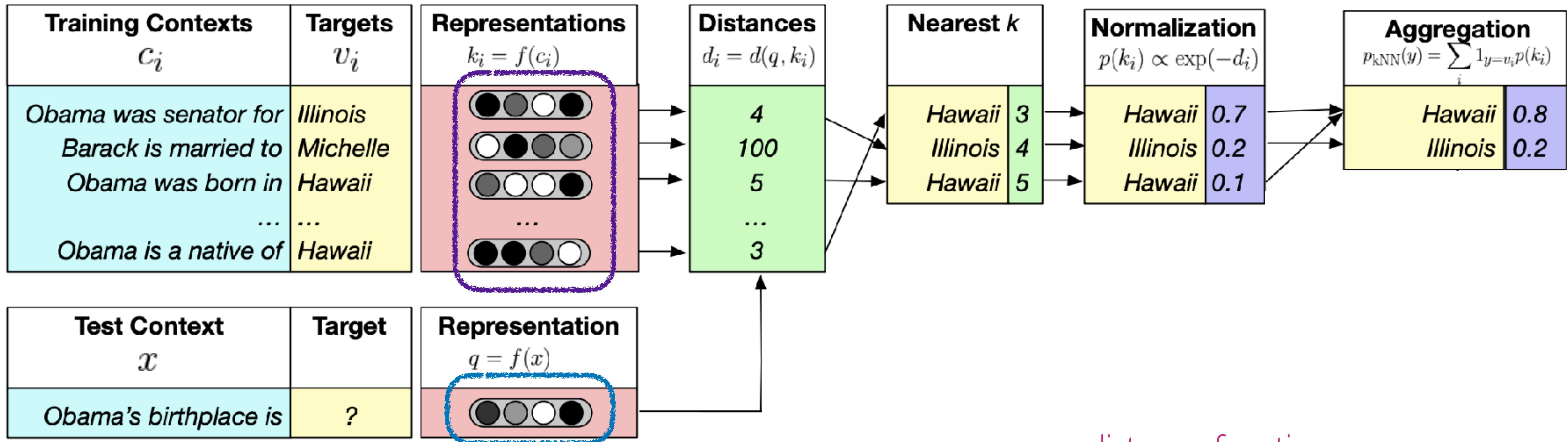


Which vectors in a datastore are close to the vector we have?

kNN-LM (Khandelwal et al. 2020)



kNN-LM (Khandelwal et al. 2020)

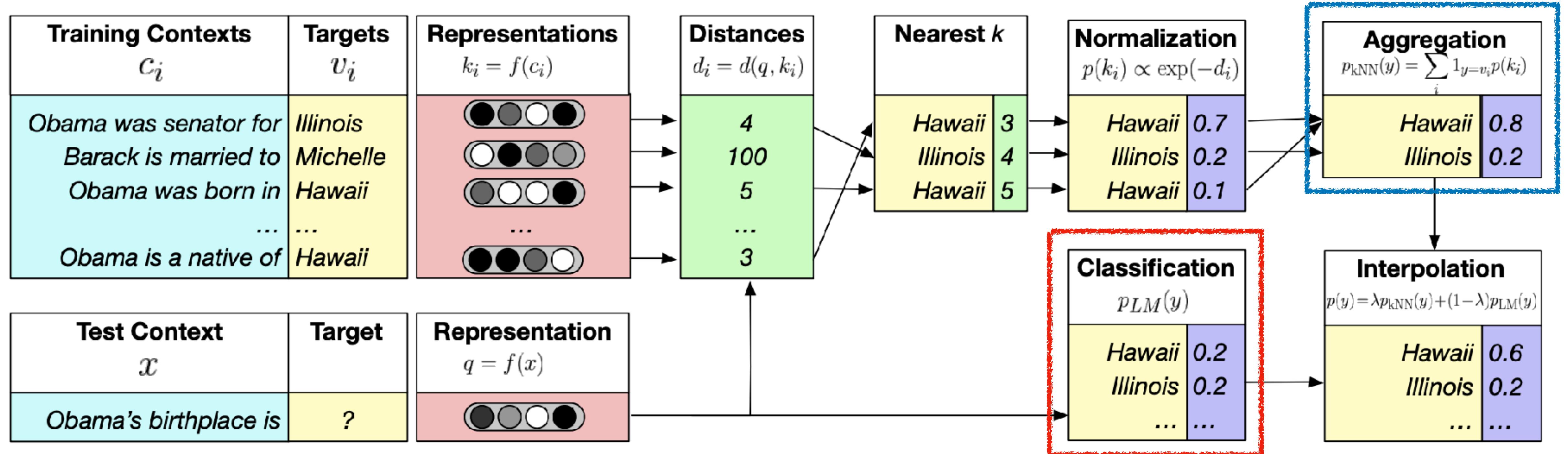


$$P_{kNN}(y | x) \propto \sum_{(k,v) \in \mathcal{D}} \mathbb{I}[v = y] \text{sim}(k, x)$$

distance function

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

kNN-LM (Khandelwal et al. 2020)



λ : hyperparameter

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

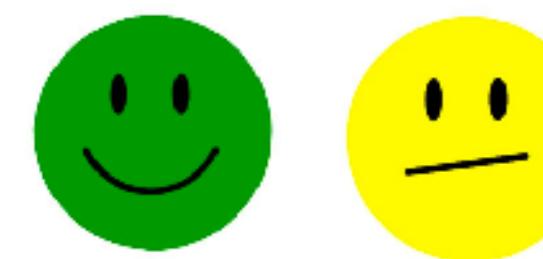
Later work, e.g., NPM (Min et al. 2023) removed interpolation (more in Section 4)

kNN-LM - why?

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

kNN-LM - why?

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



kNN-LM - why?

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



Dense vector space



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

... nice

... affordable

... bad

... good

... poor

... terrible

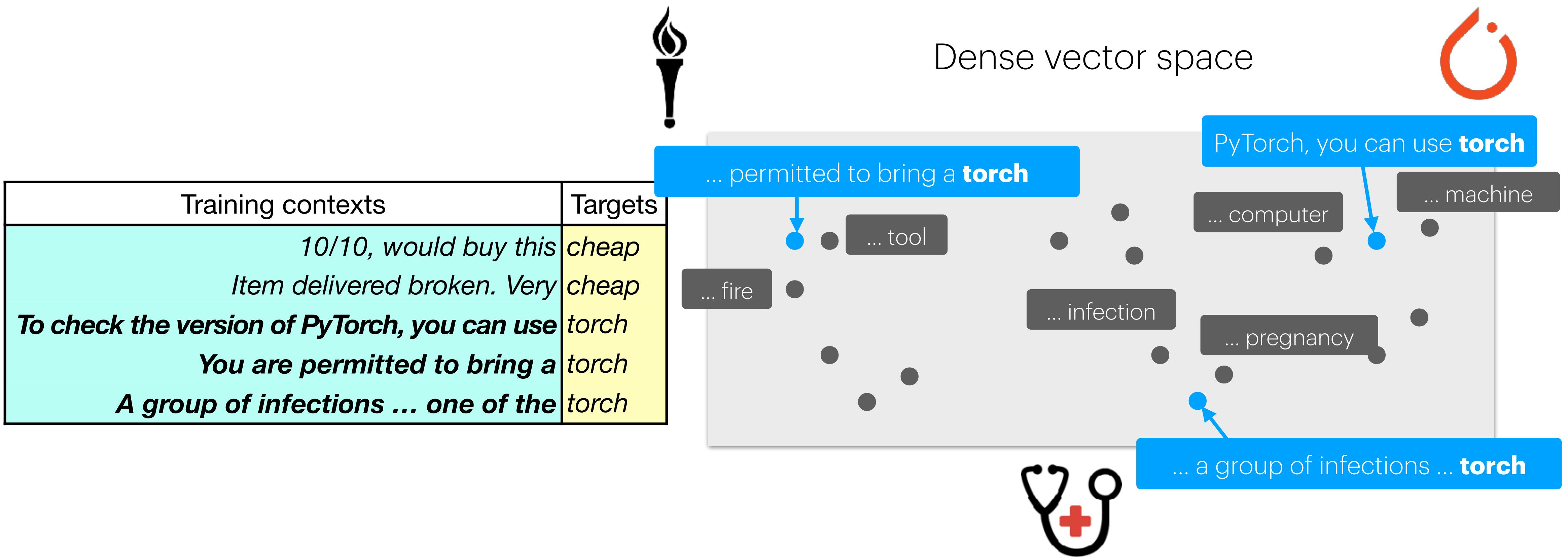
Item delivered broken. Very **cheap**

kNN-LM - why?

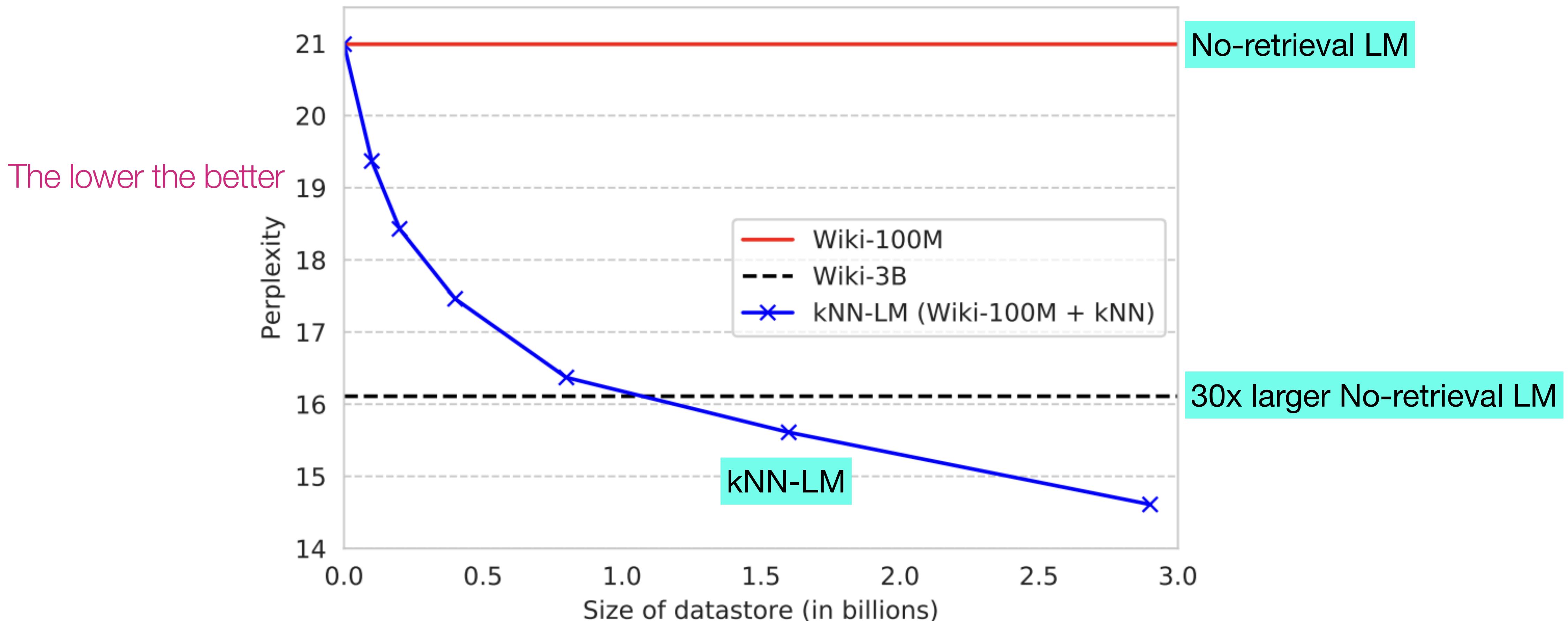
Training contexts	Targets
<i>10/10, would buy this item delivered broken. Very</i>	<i>cheap</i>
To check the version of PyTorch, you can use	<i>cheap</i>
You are permitted to bring a	<i>torch</i>
A group of infections ... one of the	<i>torch</i>
	<i>torch</i>



kNN-LM - why?



kNN-LM - results

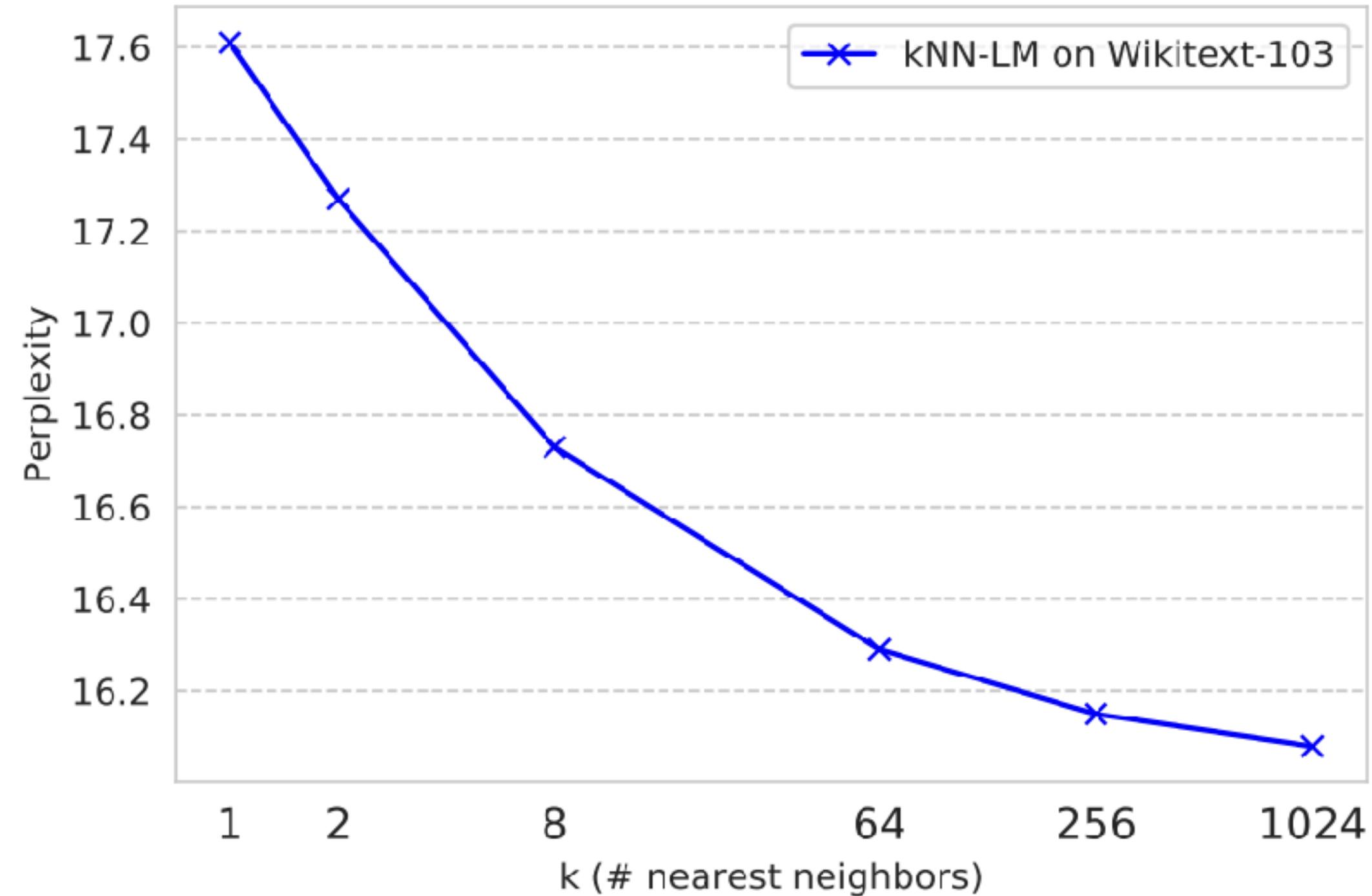


Outperforms no-retrieval LM

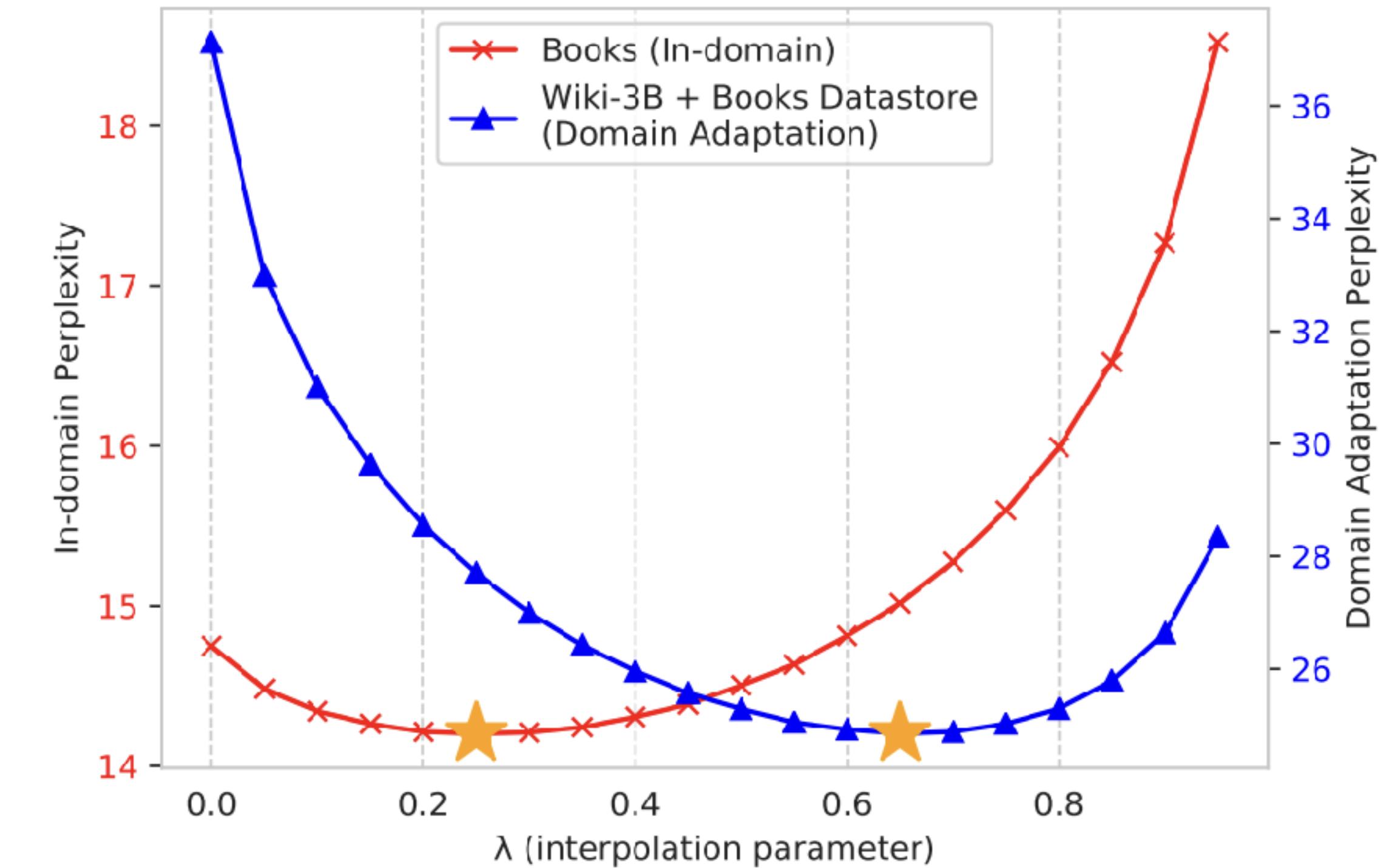
Better with bigger datastore

kNN-LM

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



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

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

(Wikipedia) 13M vs. 4B



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

No cross attention between input and retrieval results

Extensions

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

It's fixed! Can we do adaptively?

Adaptive retrieval for efficiency

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

Adaptive retrieval of
tokens
(following kNN-LM)

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.



Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

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



I am confident!

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

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



I am confident!

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

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



Unsure...

Adaptive retrieval of chunks

- *Judge necessity*

Input: Generate a summary about Joe Biden.

FLARE (Jiang et al. 2023)

Retrieval (Datastore + Index)

Language Model

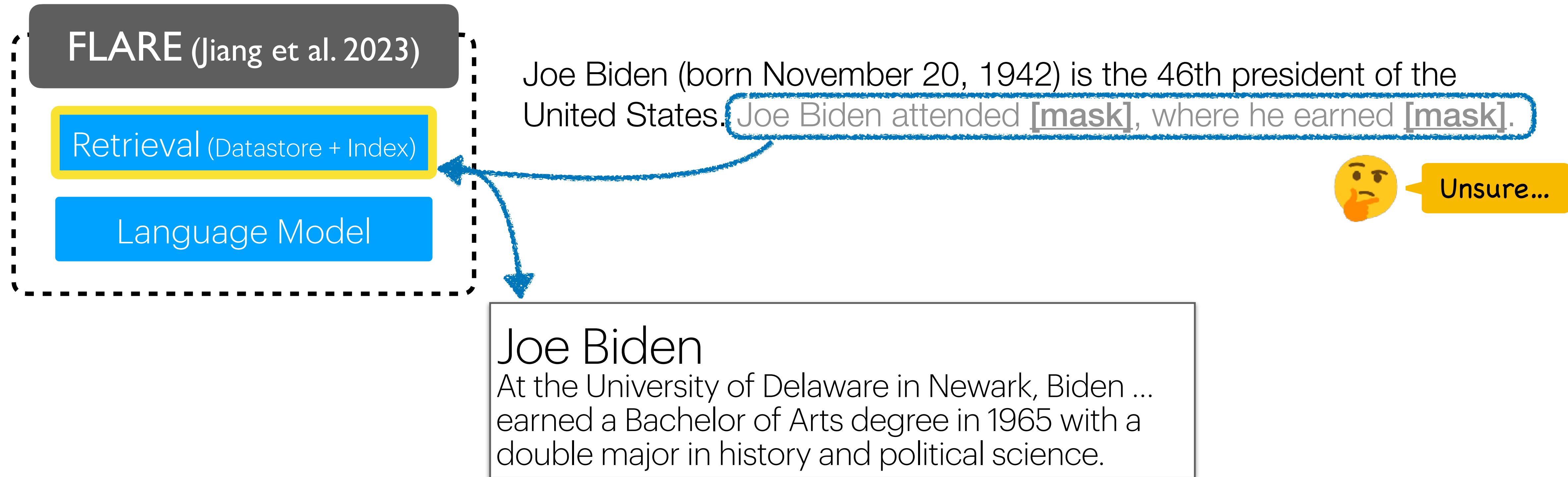
Joe Biden (born November 20, 1942) is the 46th president of the United States. Joe Biden attended [mask], where he earned [mask].



Unsure...

Adaptive retrieval of chunks

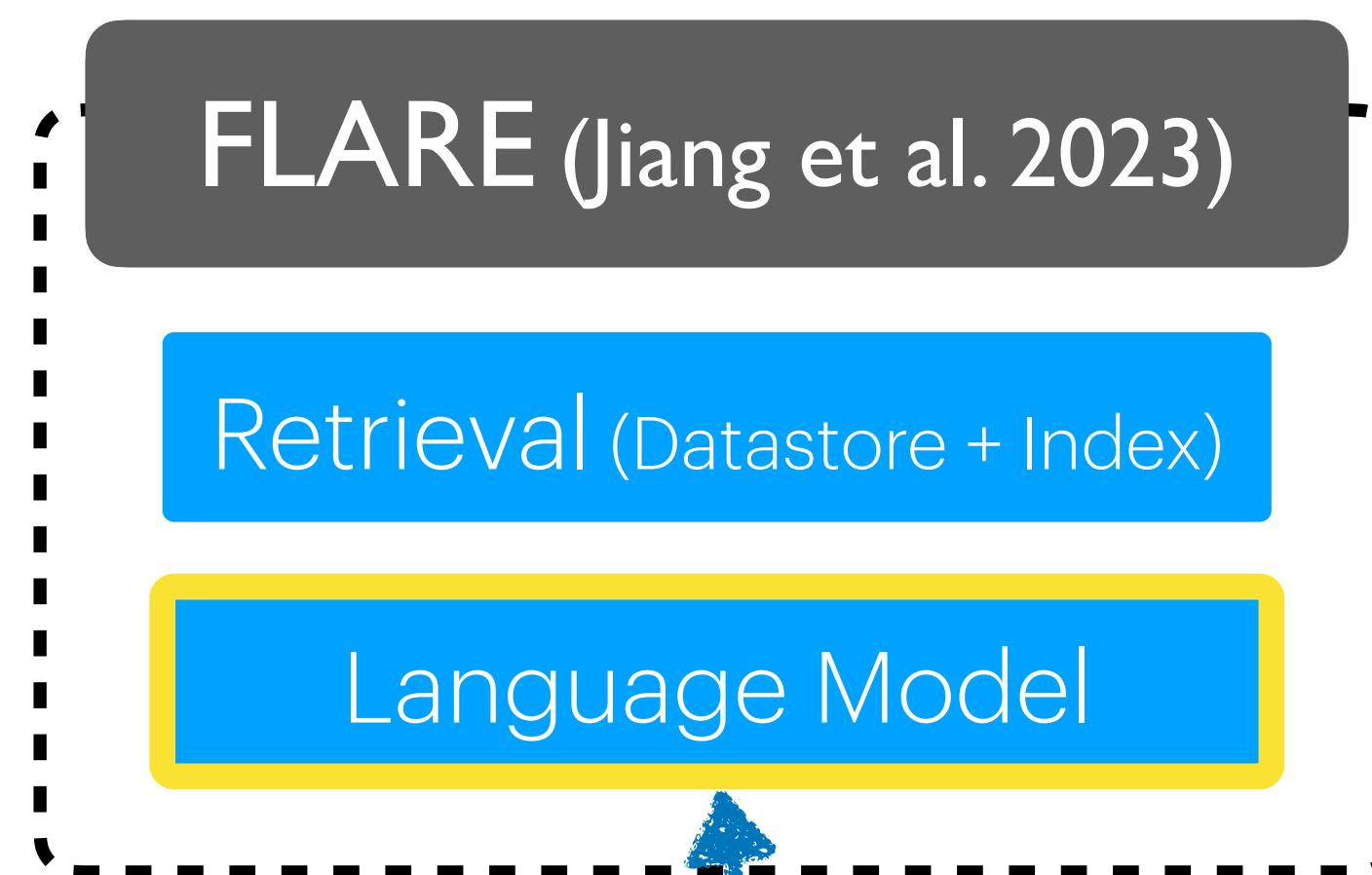
- Judge necessity



Adaptive retrieval of chunks

- Judge necessity

Input: Generate a summary about Joe Biden.



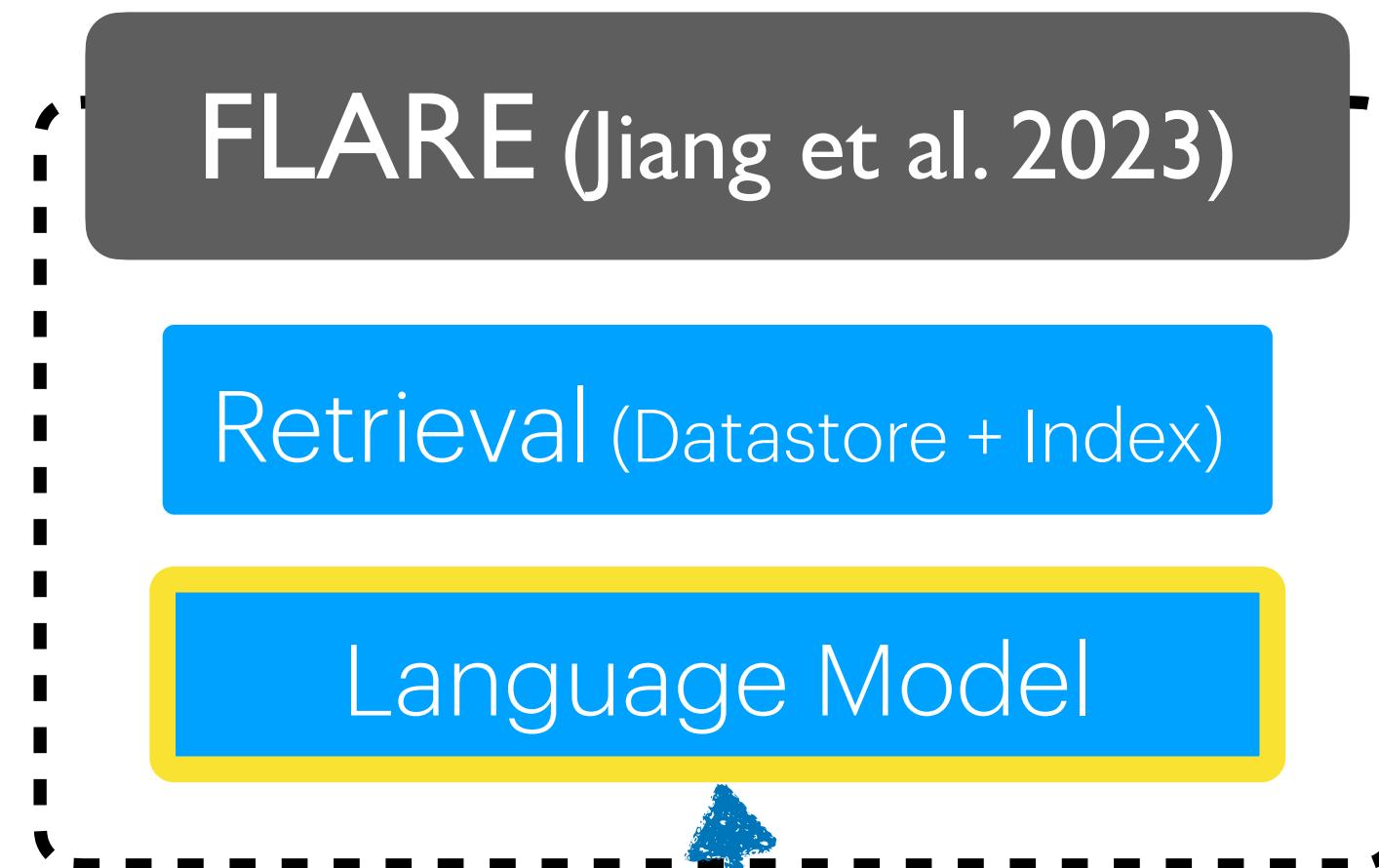
Joe Biden (born November 20, 1942) is the 46th president of the United States. ~~Joe Biden attended [mask], where he earned [mask].~~



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.

Adaptive retrieval of chunks

- Judge necessity



Input: Generate a summary about Joe Biden.

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

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

Adaptive retrieval of *tokens*

- Judge necessity

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

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

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

A function of the input \mathbf{x}

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

Adaptive retrieval of *tokens*

- *Use local info*

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

Joe Biden graduated from

Adaptive retrieval of tokens

- Use local info

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

retrieve
Joe Biden graduated from the

Adaptive retrieval of tokens

- Use local info

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

retrieve retrieve
Joe Biden graduated from the University

Adaptive retrieval of tokens

- Use local info

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

Joe Biden graduated from the University of

retrieve retrieve retrieve



Adaptive retrieval of tokens

- Use local info

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

Joe Biden graduated from the University of Delaware.

retrieve retrieve retrieve retrieve



Adaptive retrieval of tokens

- Use local info

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

Joe Biden graduated from the University of Delaware.

retrieve retrieve retrieve retrieve



Adaptive retrieval of tokens

- Use local info

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

retrieve
Joe Biden graduated from the

Adaptive retrieval of tokens

- Use local info

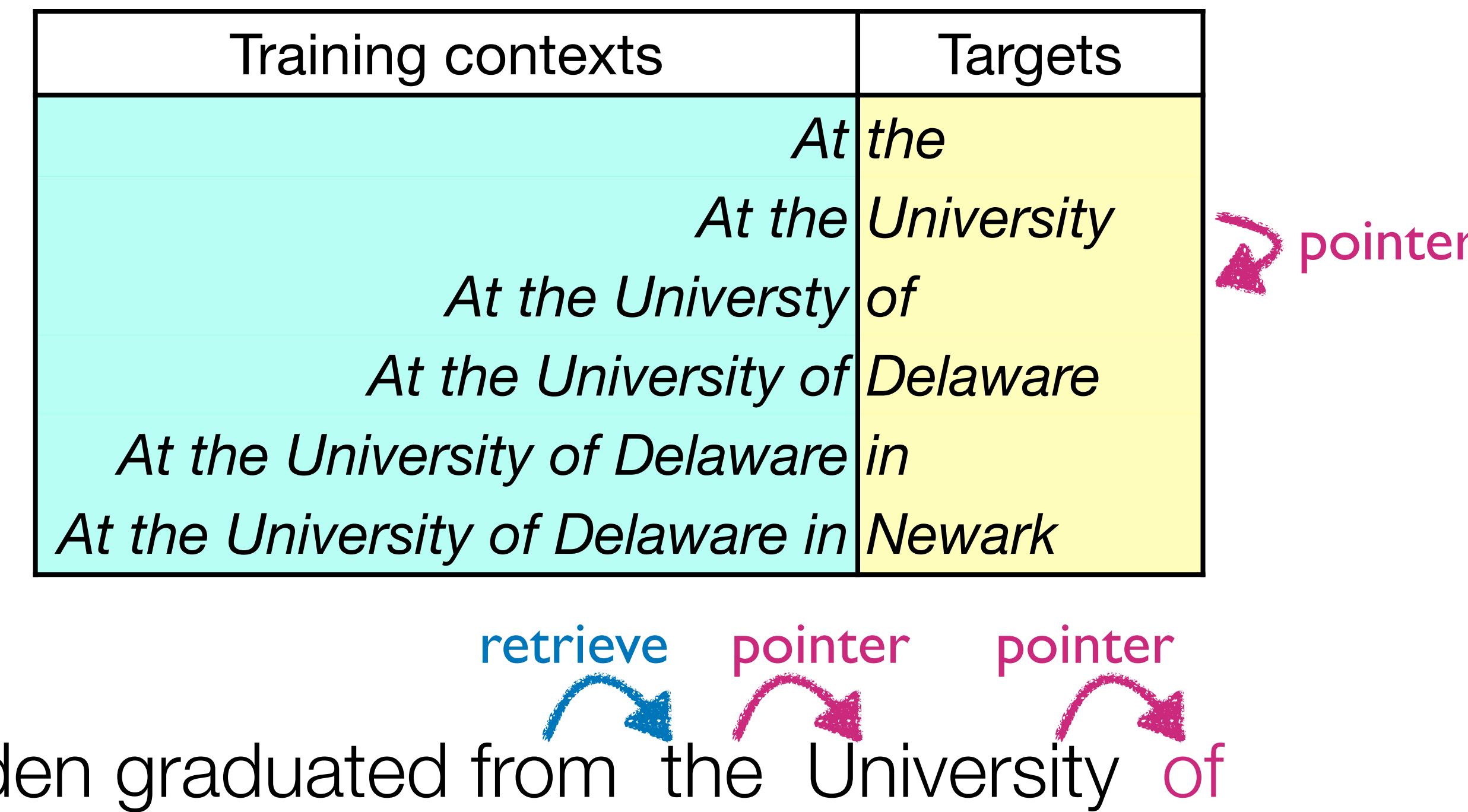
Training contexts	Targets
	<i>At the</i>
	<i>At the</i>
	<i>At the University</i>
	<i>At the University of</i>
<i>At the University of Delaware</i>	<i>Delaware</i>
<i>At the University of Delaware in</i>	<i>in</i>
<i>At the University of Delaware in Newark</i>	<i>Newark</i>

retrieve pointer

Joe Biden graduated from the University

Adaptive retrieval of tokens

- Use local info



Adaptive retrieval of tokens

- Use local info

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

Joe Biden graduated from the University of Delaware.

retrieve pointer pointer pointer



Retrieve once, and save other searches!

Summary

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



More efficient



Decision may not always be optimal

Soft adaptive instead of hard adaptive (Drozdov et al. 2022 & more):



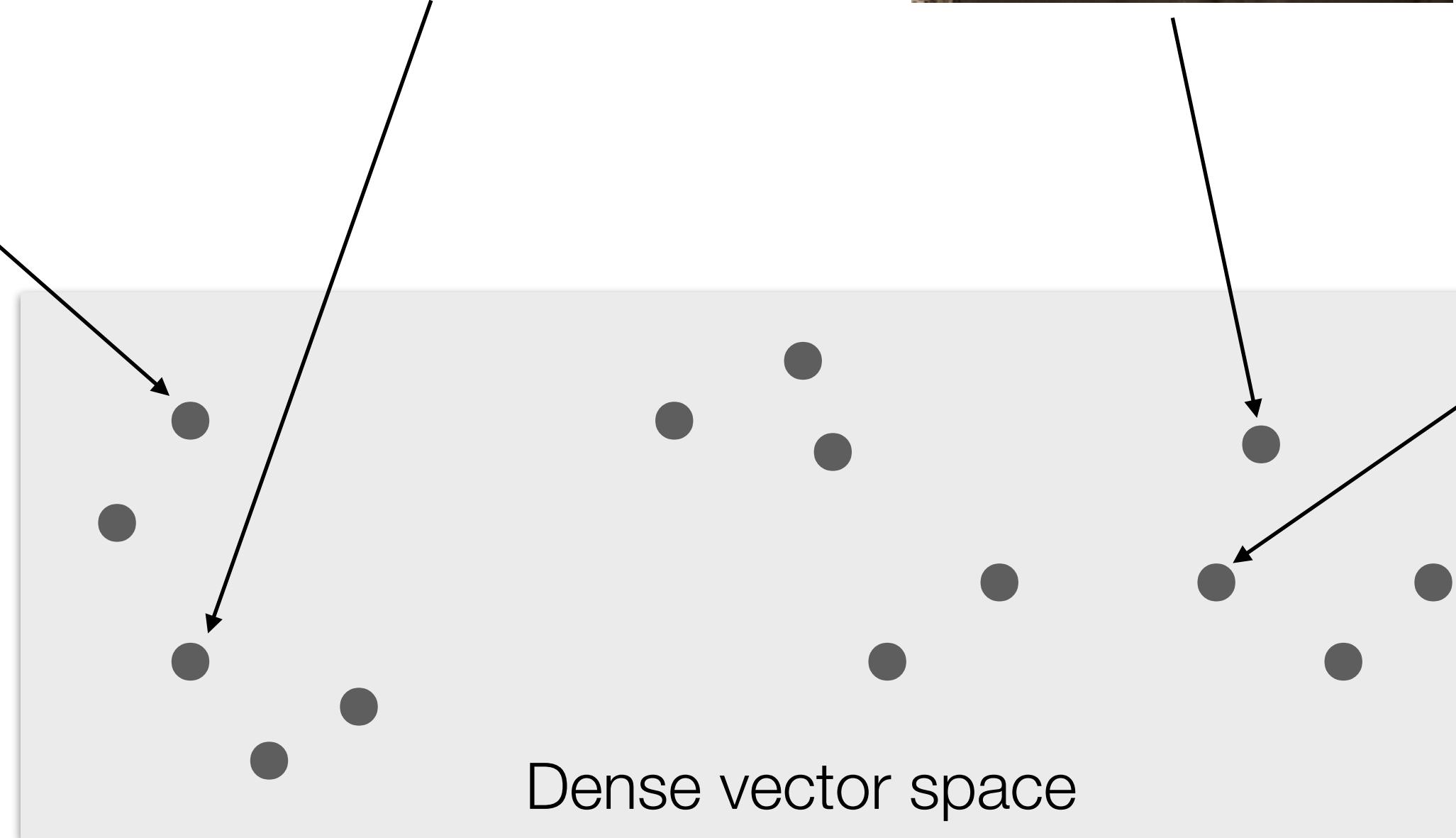
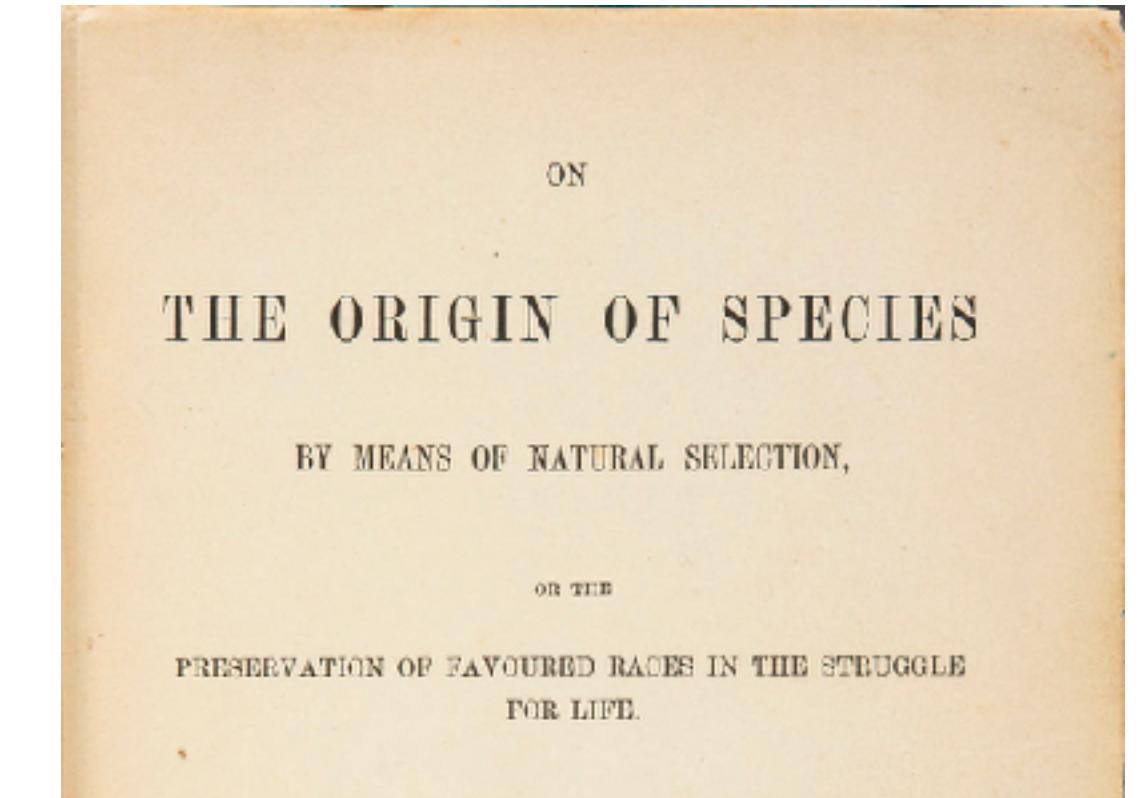
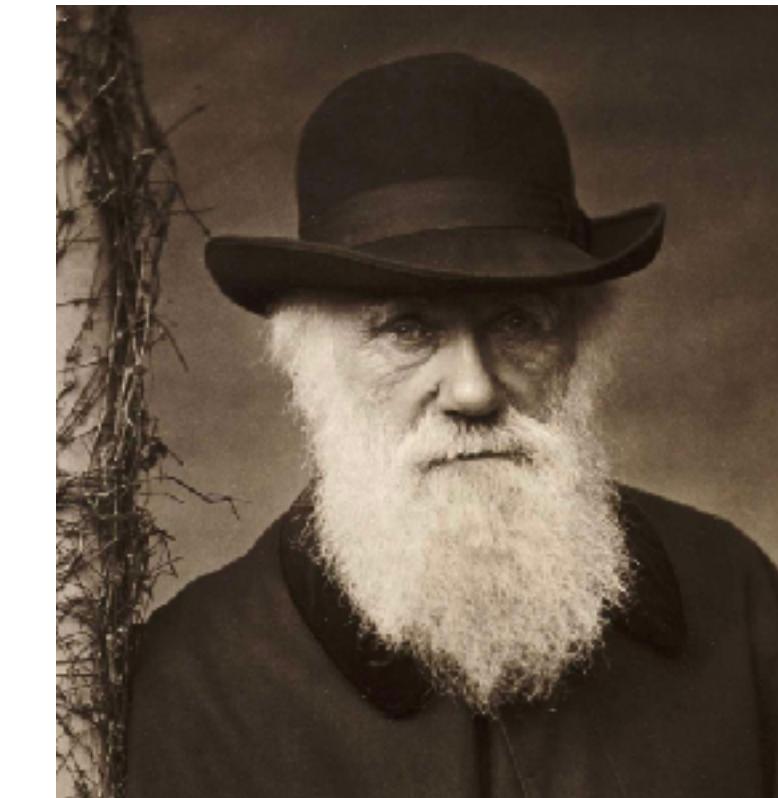
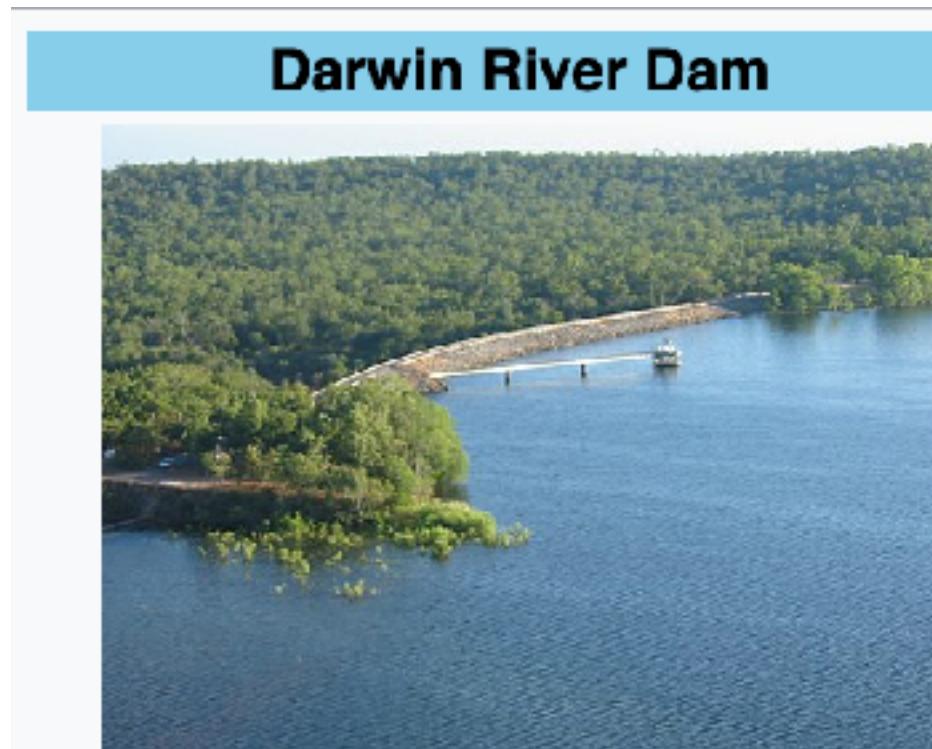
More expressivity, Better performance

Summary

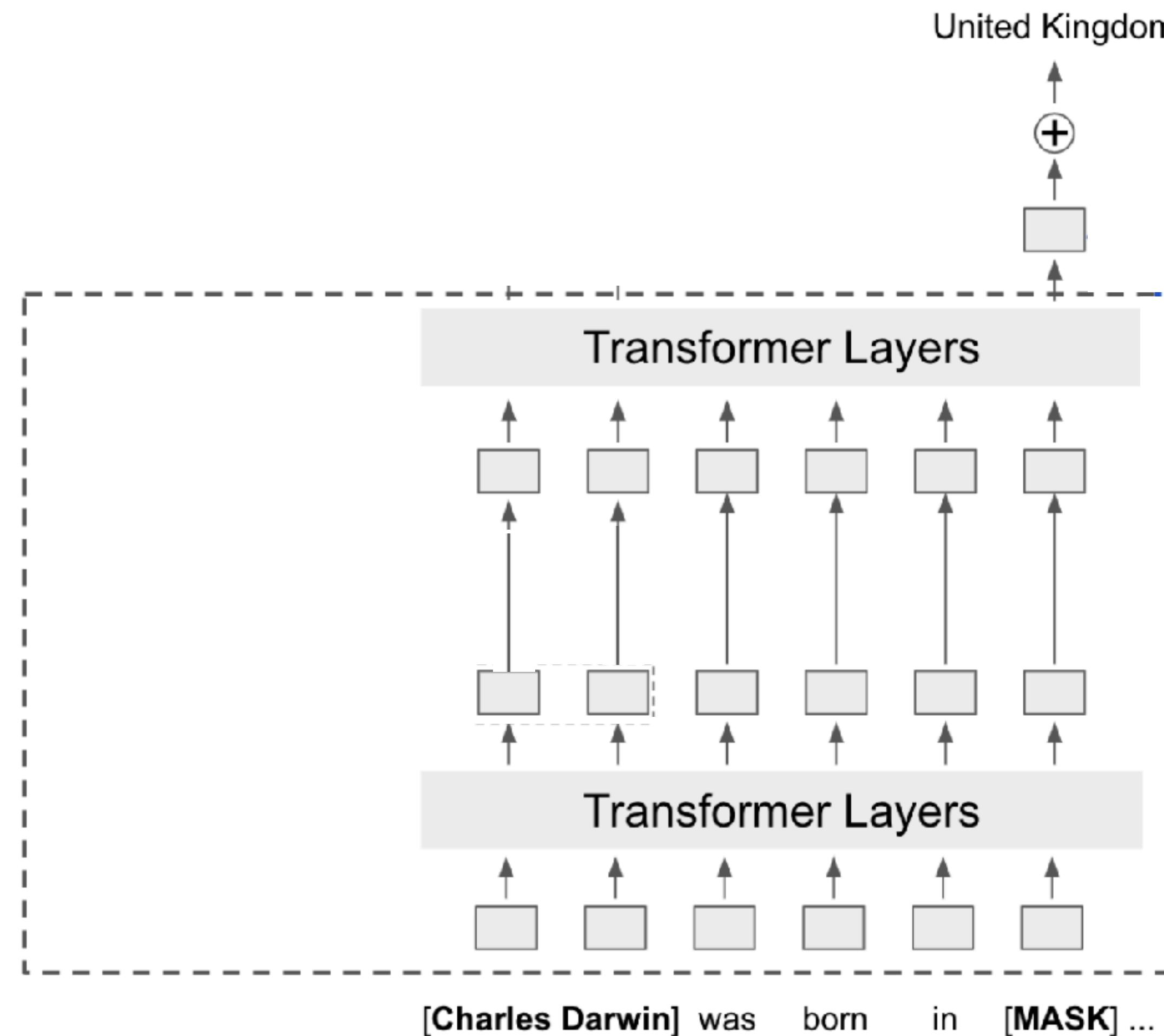
	What do retrieve?	How to use retrieval?	When to retrieve?
REALM (Guu et al 2020)	Text chunks	Input layer	Once
Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)	Text chunks	Input layer	Every n tokens
RETRO (Borgeaud et al. 2021)	Text chunks	Intermediate layers	Every n tokens
kNN-LM (Khandelwal et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

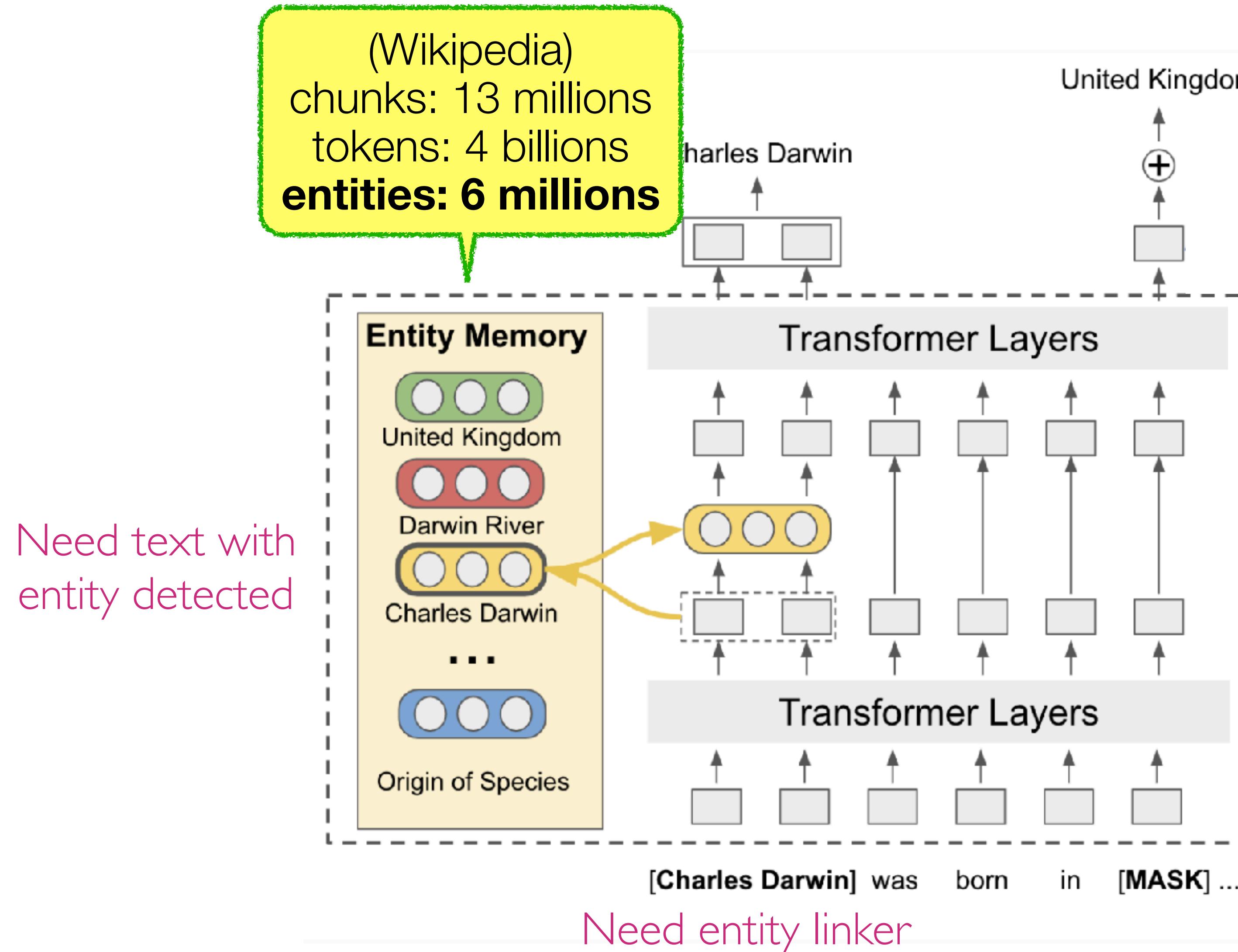
Entities as Experts (Fevry et al. 2020)



Entities as Experts (Fevry et al. 2020)



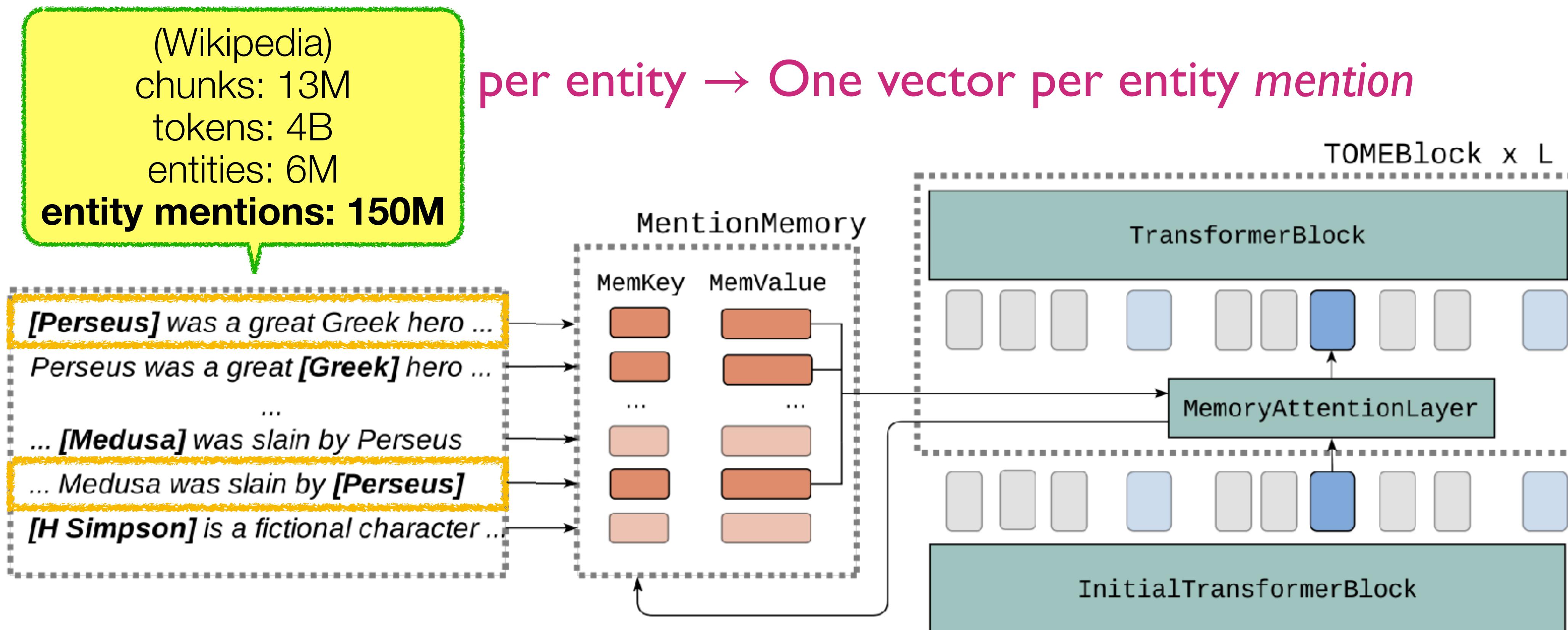
Entities as Experts (Fevry et al. 2020)



Mention Memory (de Jong et al. 2022)

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

Mention Memory (de Jong et al. 2022)



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

Summary

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



Most effective for entity-centric tasks & space-efficient



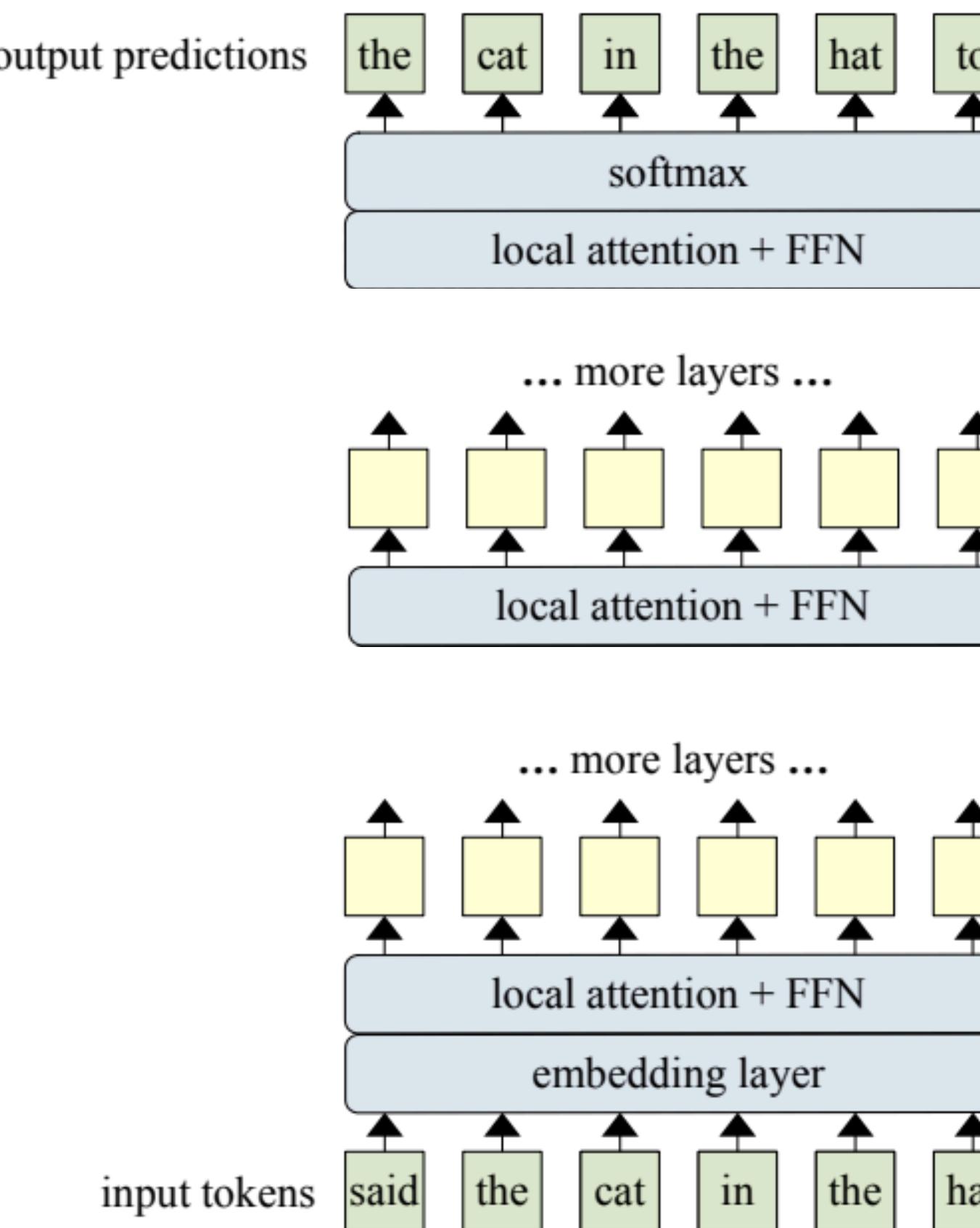
Additional entity detection required

Summary

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

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

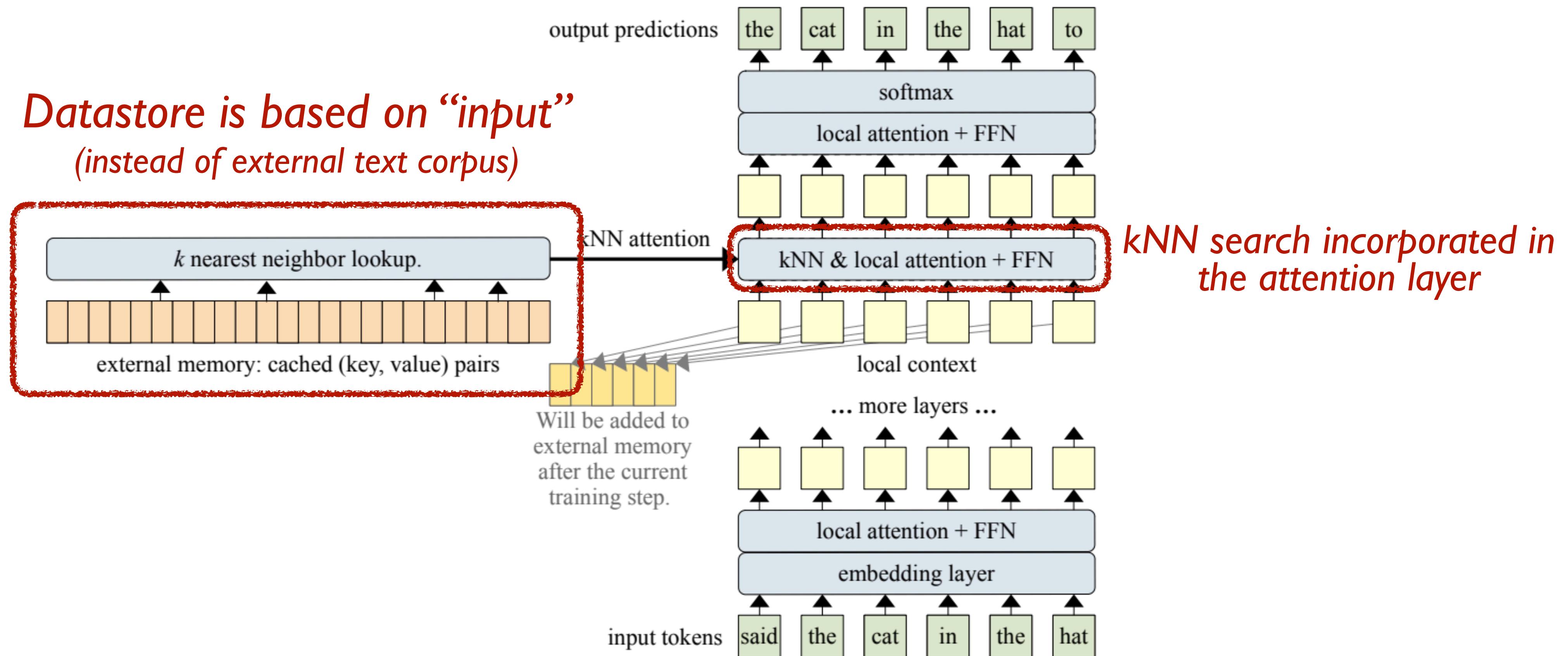
Retrieval for long-range LM



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

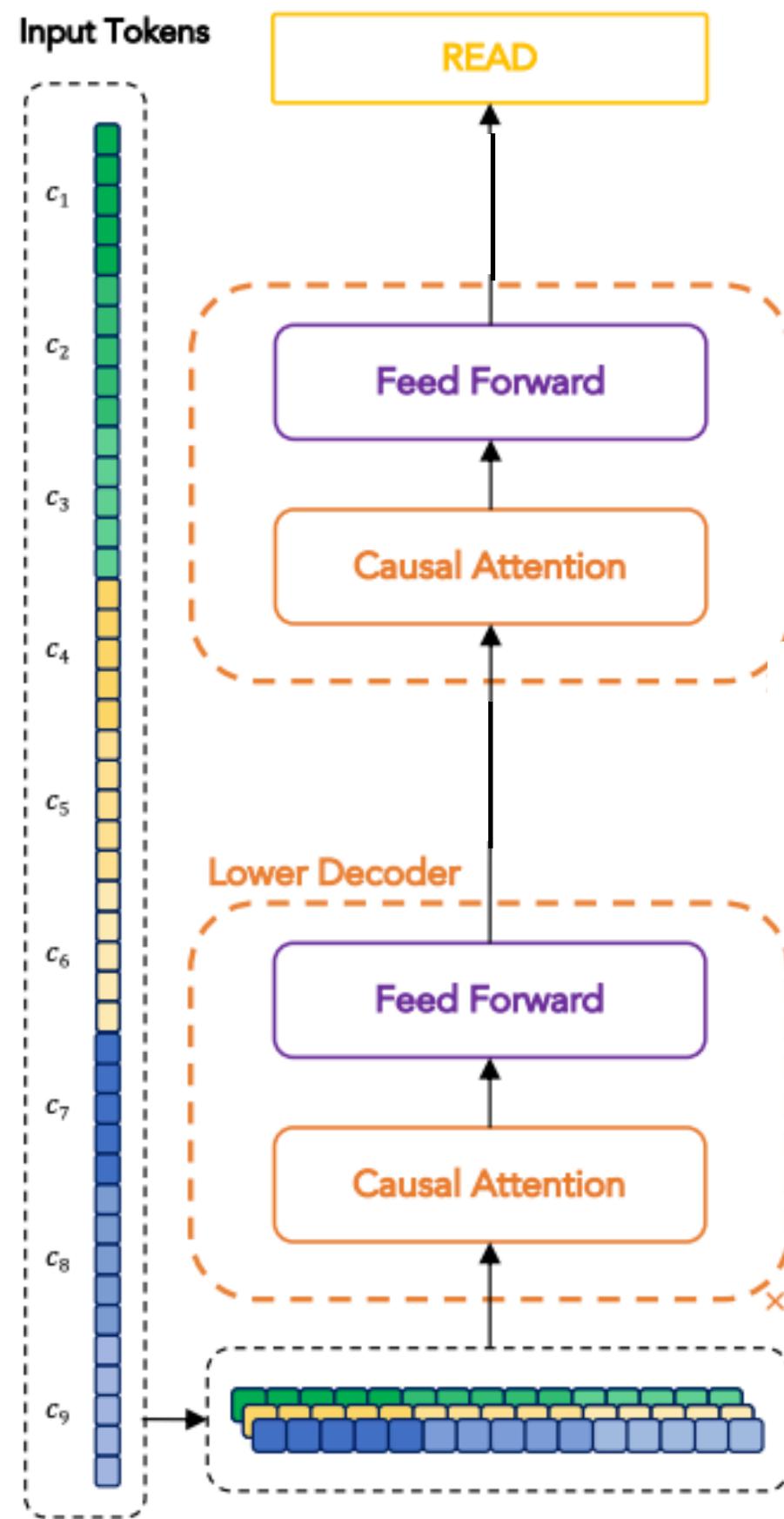
Retrieval for long-range LM

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



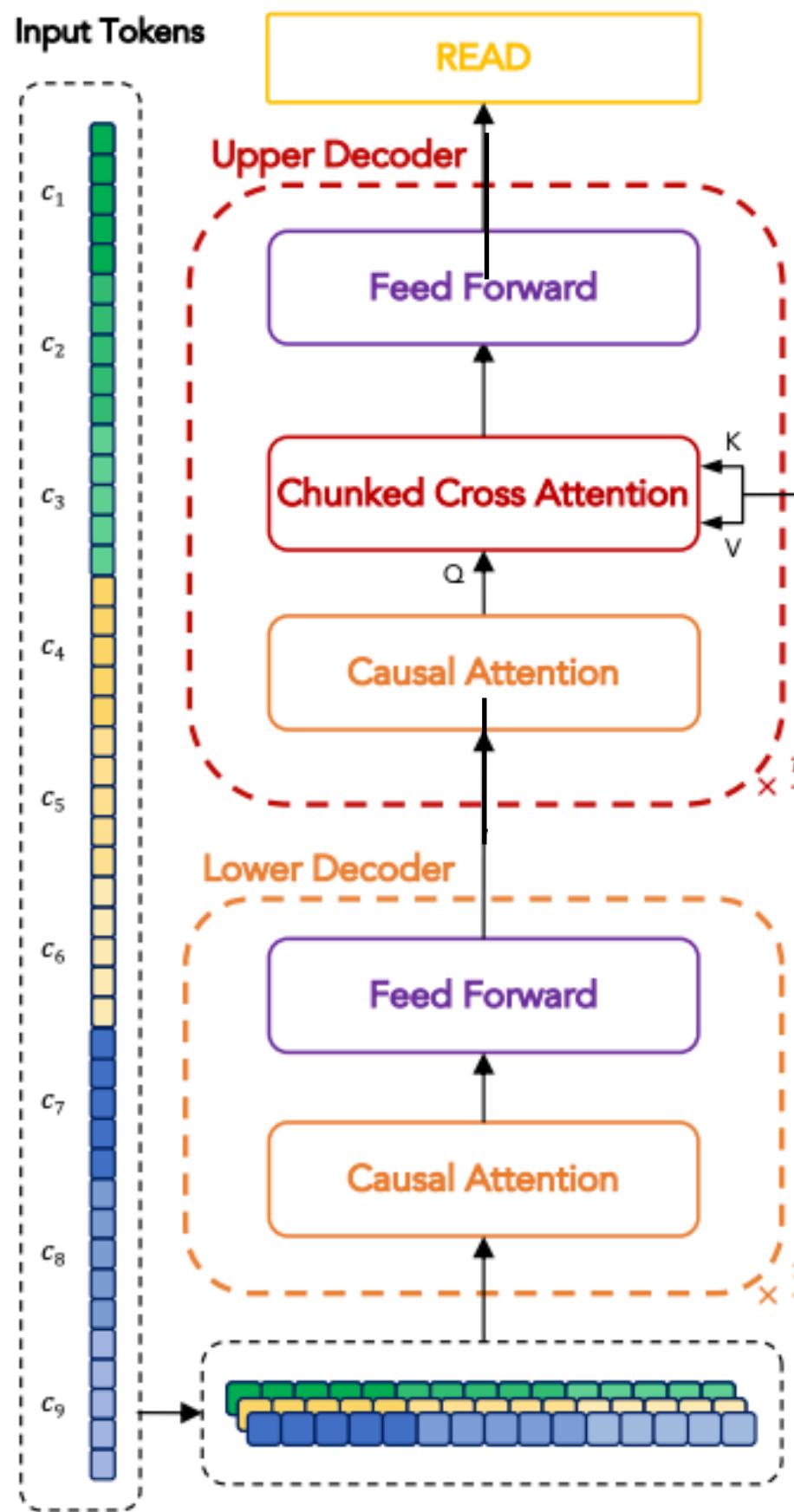
Wu et al. 2022. Memorizing Transformers (**Figure source**)
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval

Retrieval for long-range LM



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

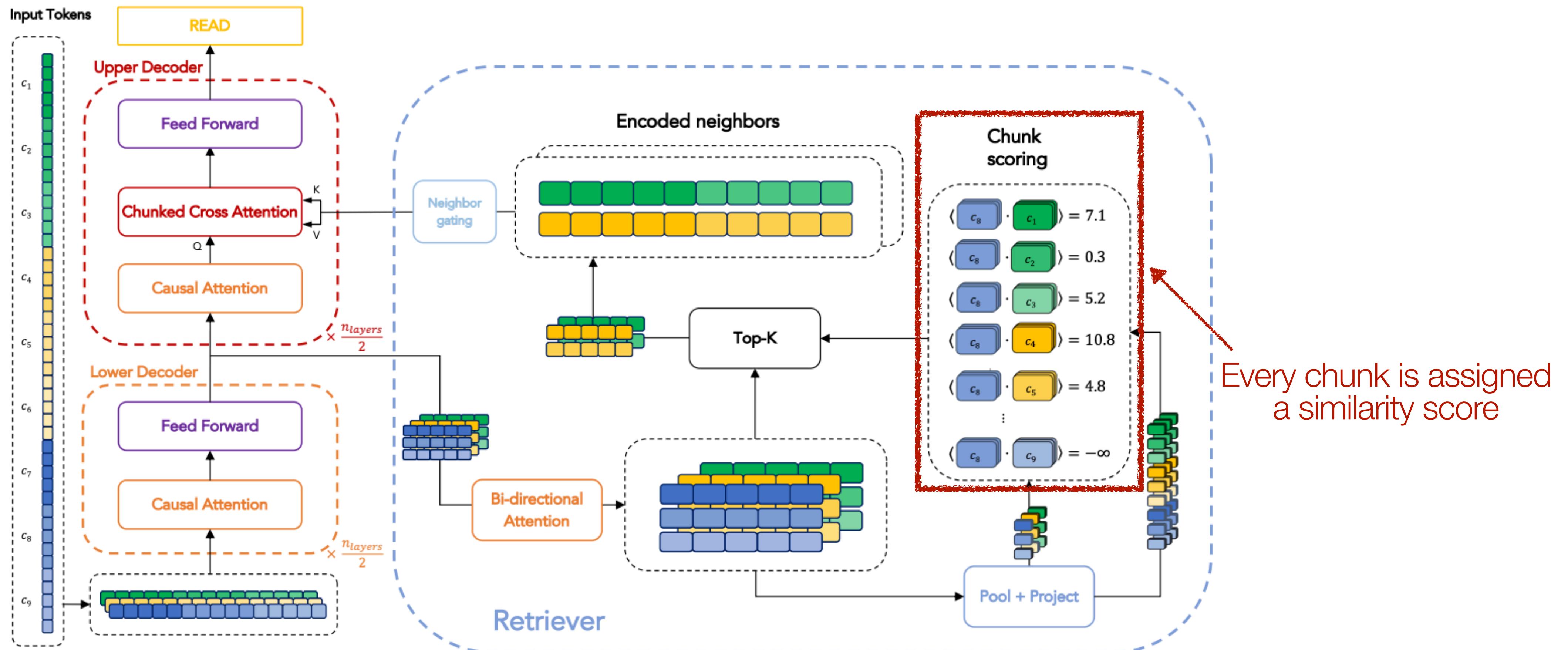
Retrieval for long-range LM



Chunked Cross Attention

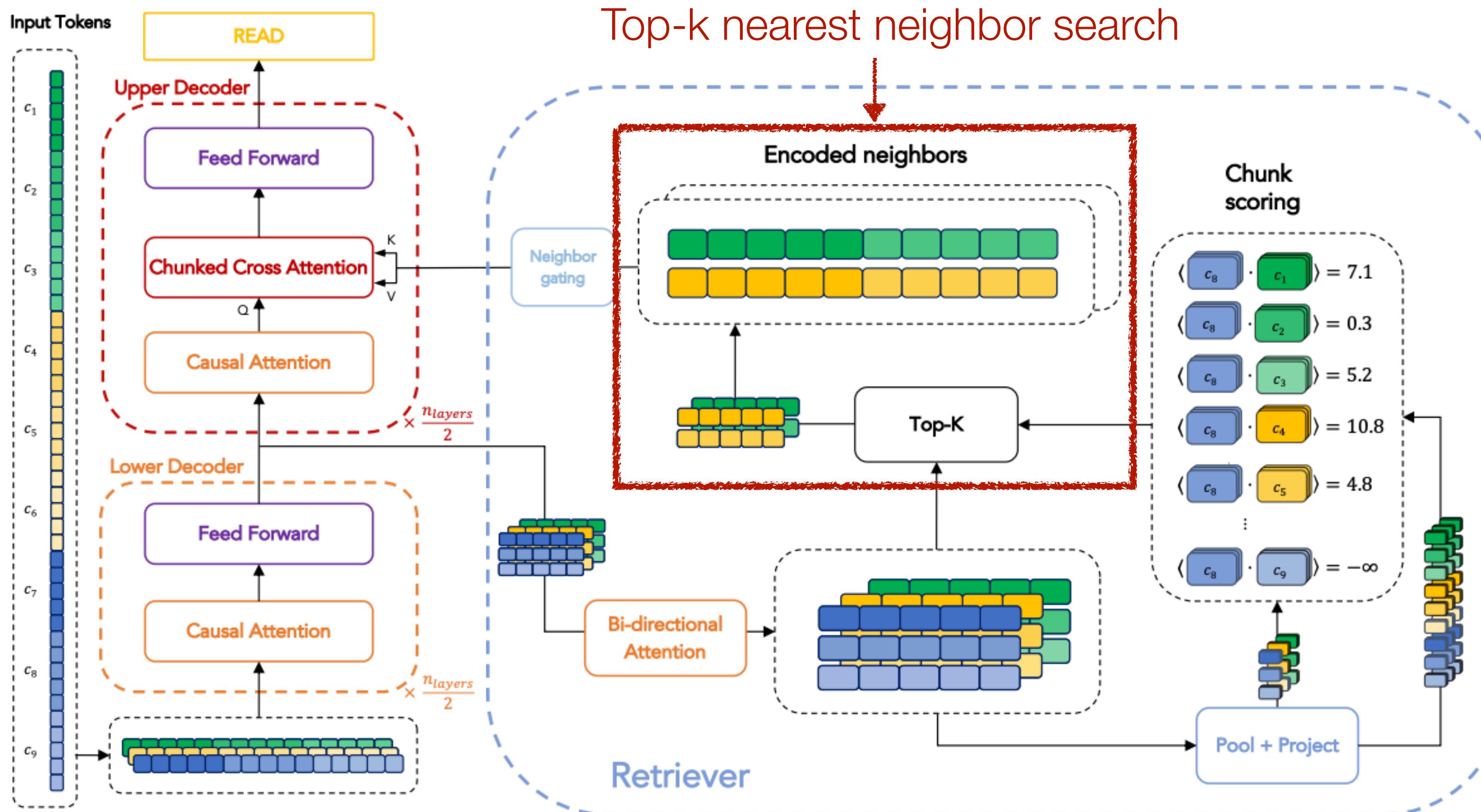
Wu et al. 2022. Memorizing Transformers
Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval (**Figure source**)

Retrieval for long-range LM



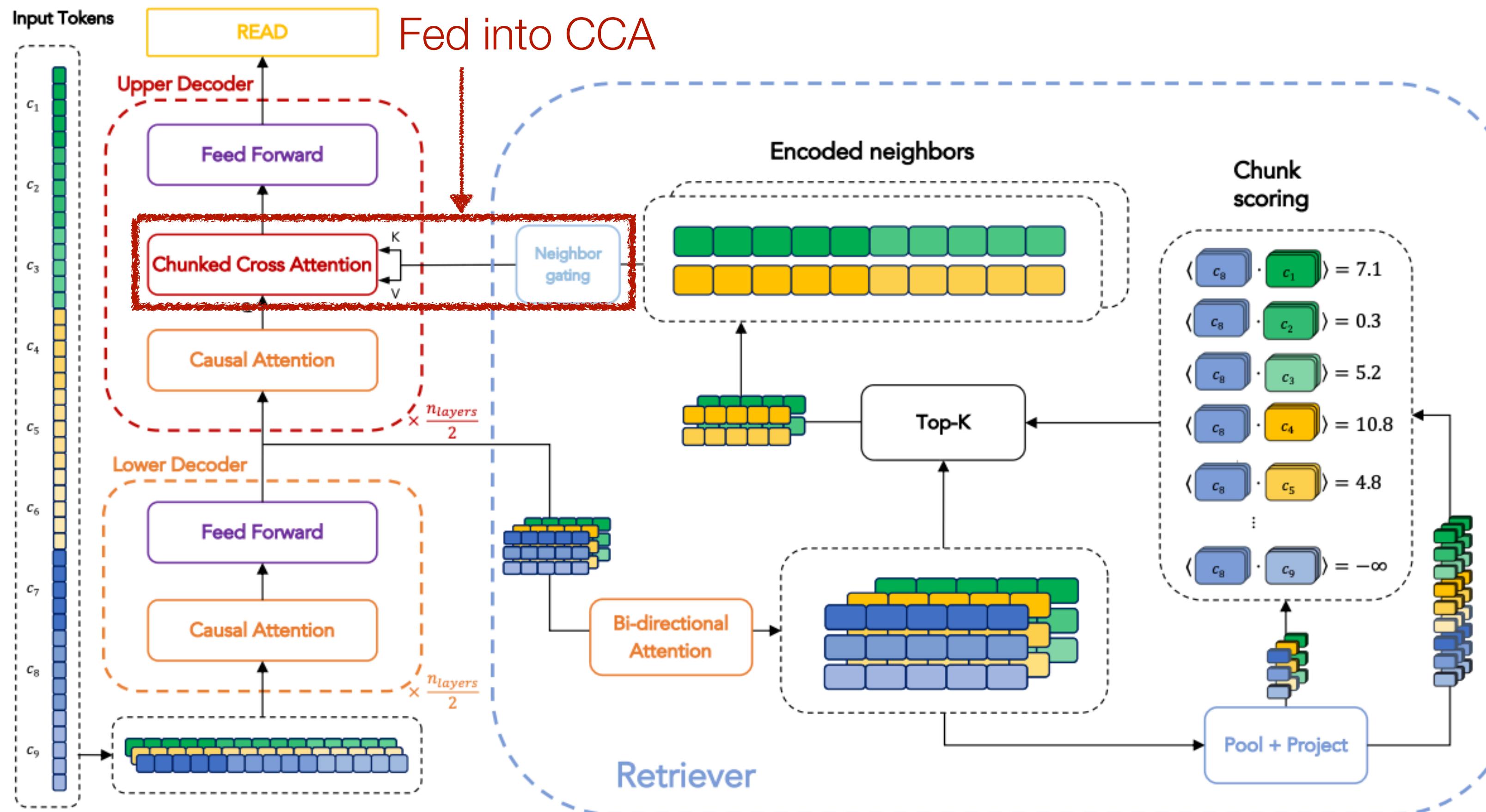
Wu et al. 2022. Memorizing Transformers
 Bertsch et al. 2023. Unlimiformer: Long-Range Transformers with Unlimited Length Input
 Rubin & Berant. 2023. Long-range Language Modeling with Self-retrieval (**Figure source**)

Retrieval for long-range LM



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

Retrieval for long-range LM



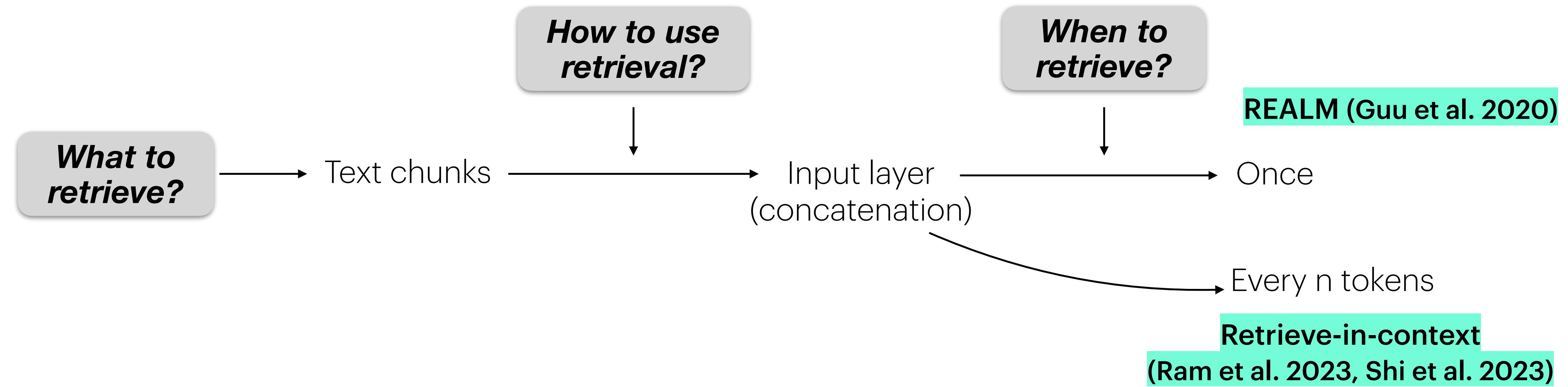
Wu et al. 2022. Memorizing Transformers

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

Summary

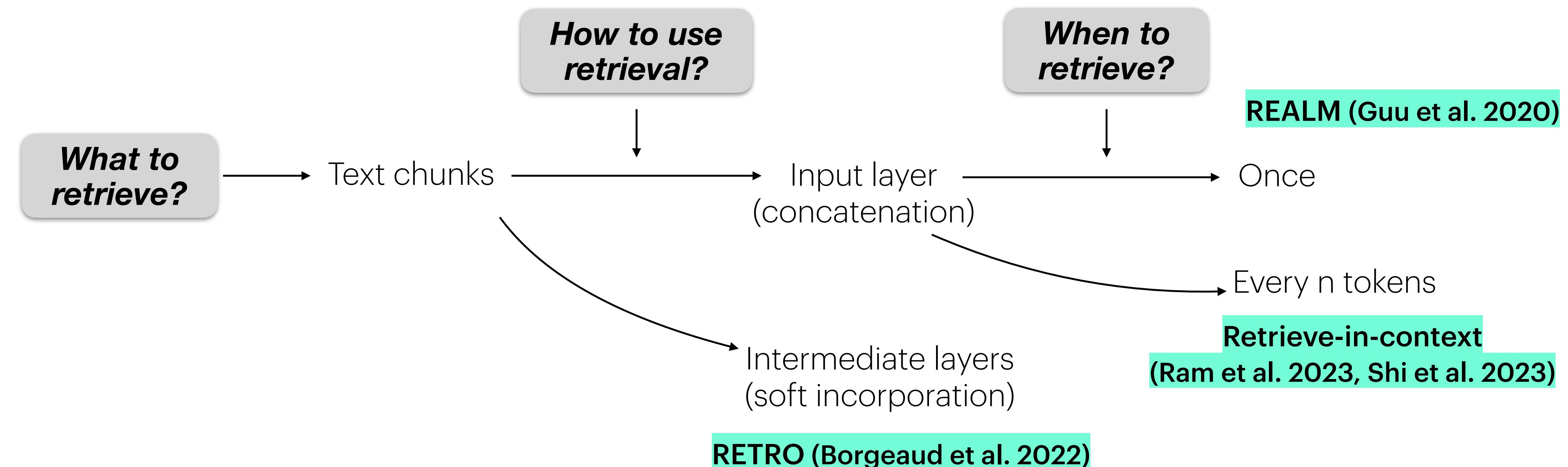
	What do retrieve?	How to use retrieval?	When to retrieve?
REALM (Guu et al 2020)	Text chunks	Input layer	Once
Retrieve-in-context LM (Shi et al 2023, Ram et al 2023)	Text chunks	Input layer	Every n tokens
RETRO (Borgeaud et al. 2021)	Text chunks	Intermediate layers	Every n tokens
kNN-LM (Khandelwal et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



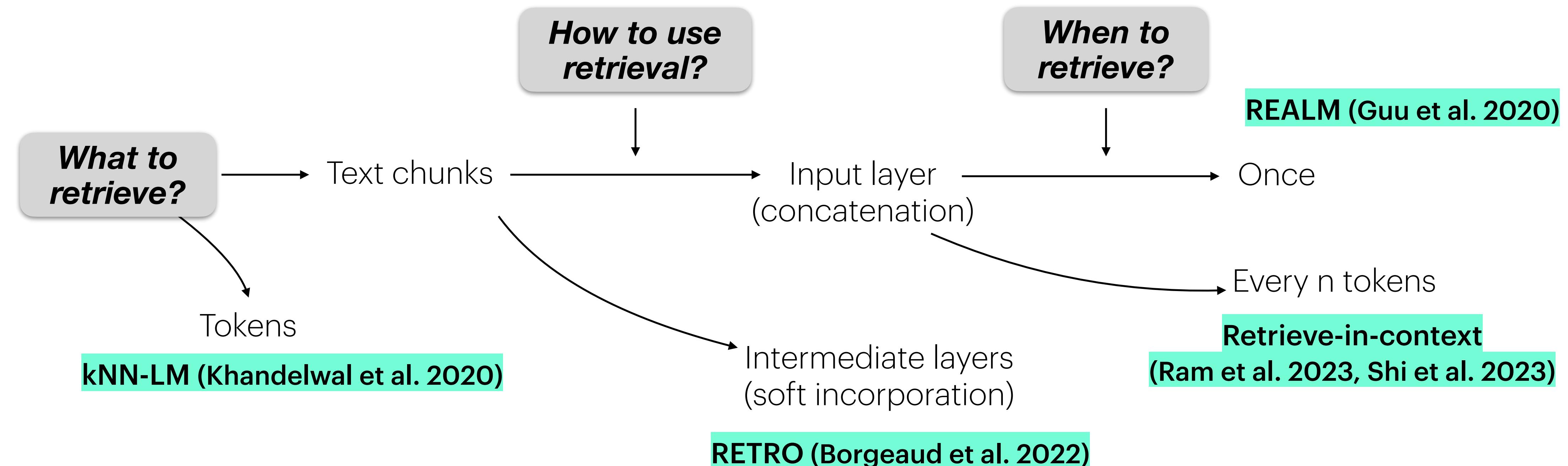
More frequent retrieval = better in performance, but slower

Wrapping up



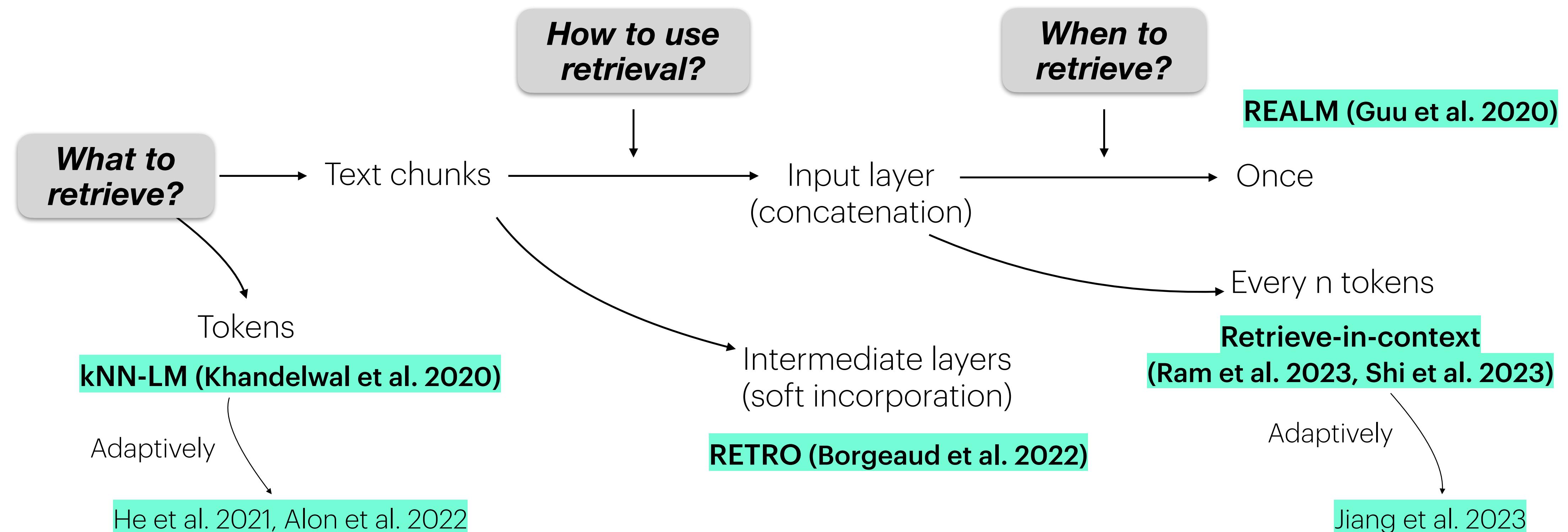
- Input layer: Simple but can be slower
- Intermediate layers: More complex (need training) but can be designed to be more efficient

Wrapping up



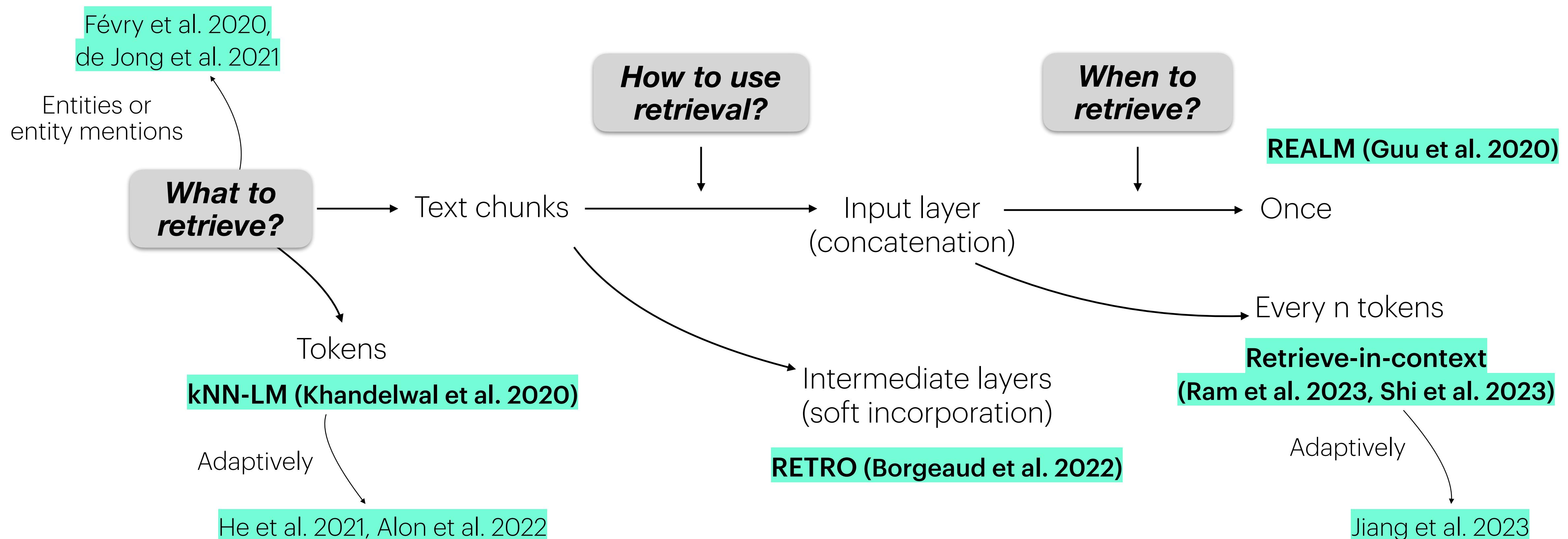
- Text blocks: Datastore can be space-efficient, more computation
- Tokens: More fine-grained, compute-efficient, but datastore can be space-expensive

Wrapping up



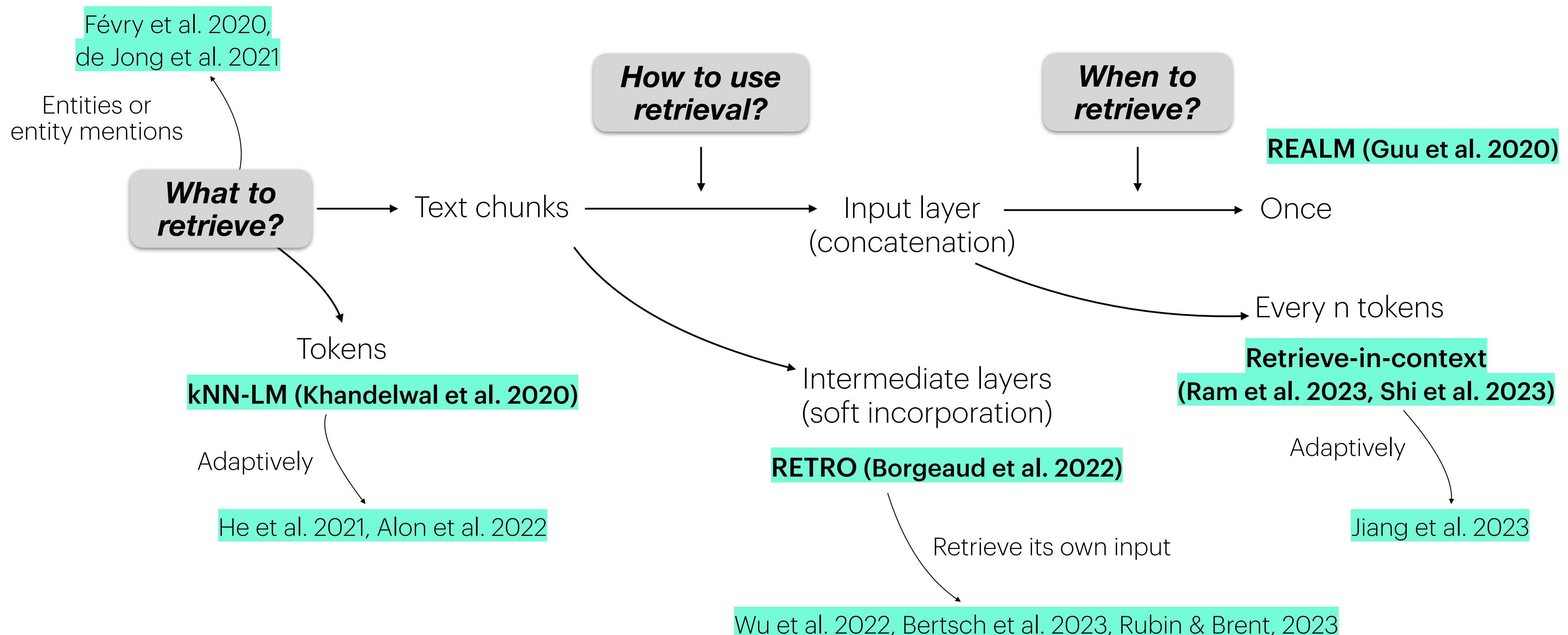
Adaptive retrieval can improve efficiency

Wrapping up



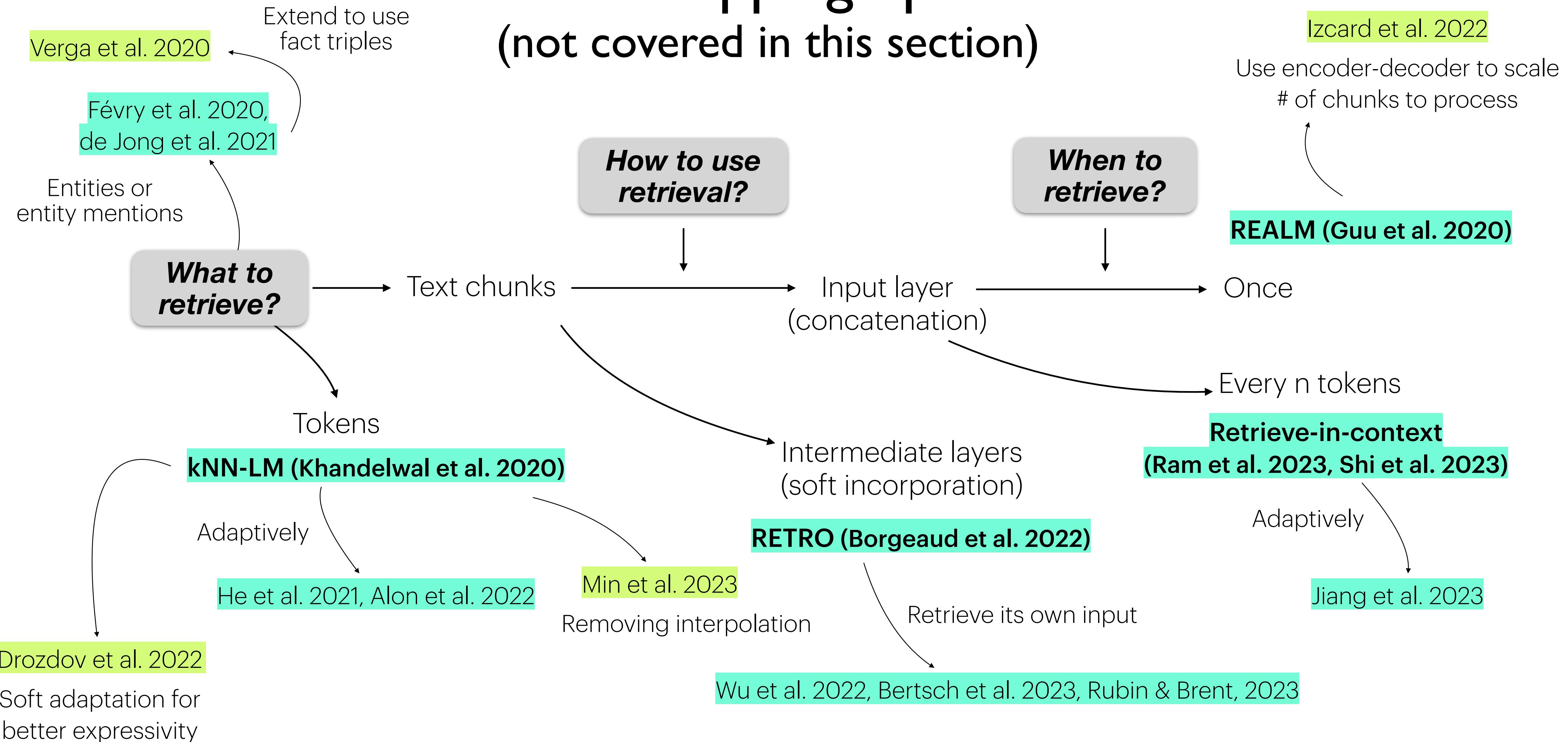
Entities or entity mentions instead of every token or chunk

Wrapping up

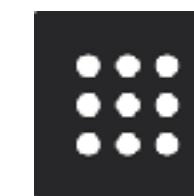
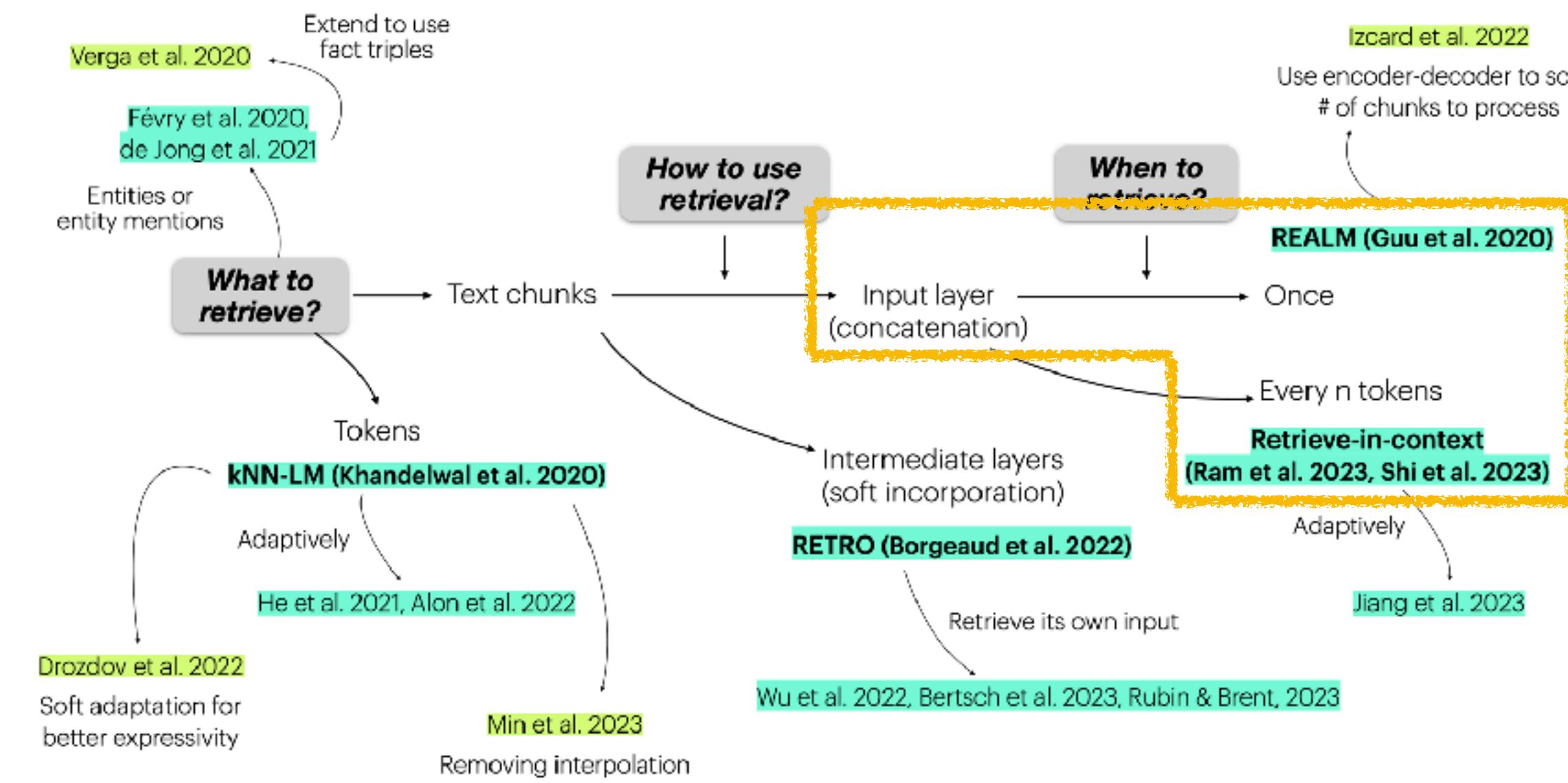


We can use a similar approach for long-sequence modeling

Wrapping up (not covered in this section)



Wrapping up



Perplexity

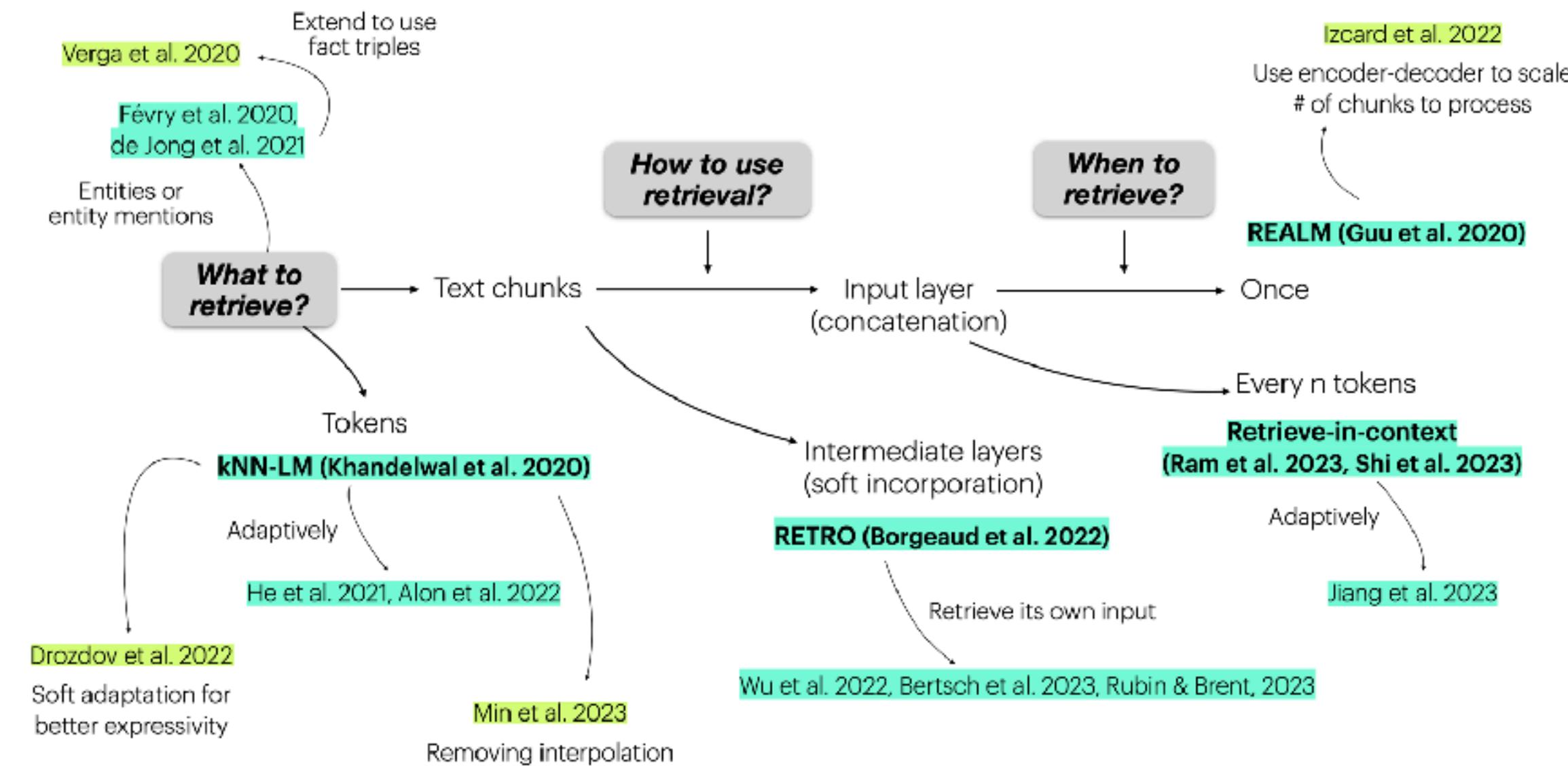
WebGPT

YOU

Still largely under-explored!

Chat GPT
Extension

Wrapping up



We didn't cover anything about training →
Section 4!

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