

Natural Language Understanding, Generation, and Machine Translation

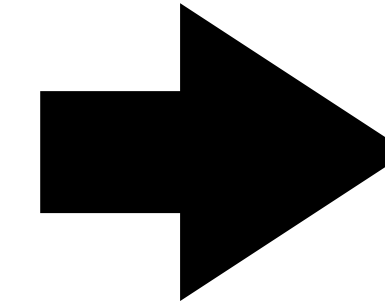
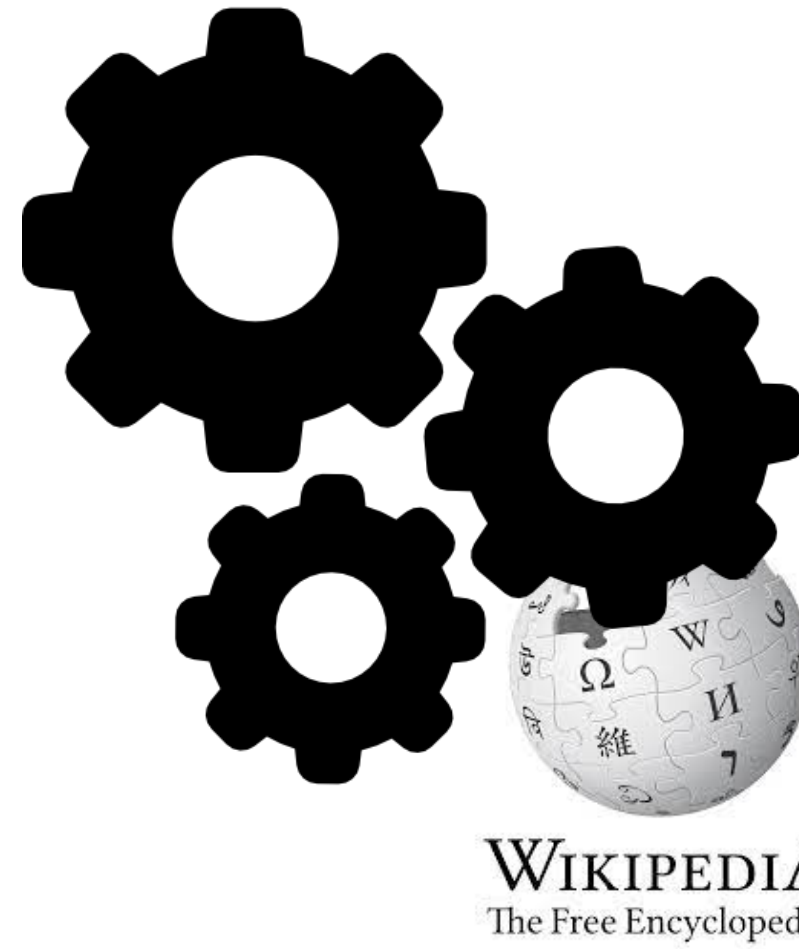
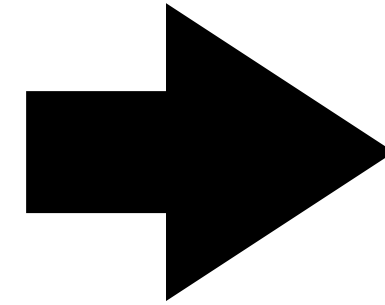
Lecture 23: Retrieval Augmented Generation

Pasquale Minervini
p.minervini@ed.ac.uk
March 13th, 2024

Open Domain Question Answering

Recap

Question (Q)

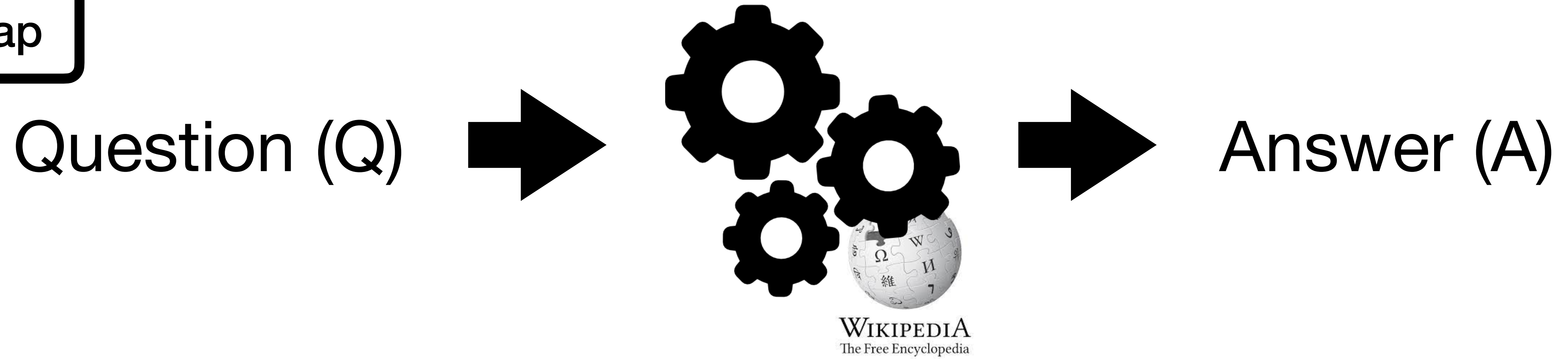


Answer (A)

Open-Domain Question Answering (ODQA):

Open Domain Question Answering

Recap

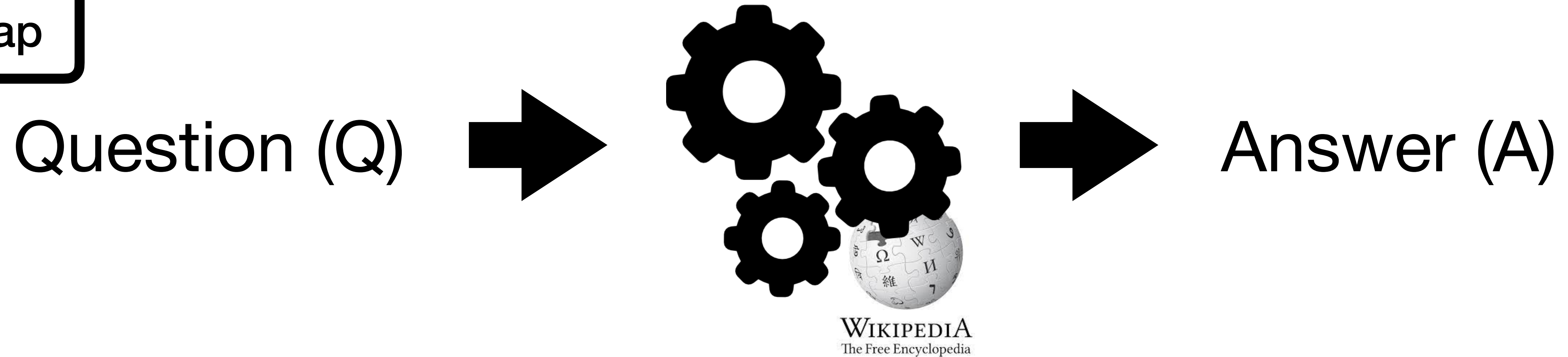


Open-Domain Question Answering (ODQA):

We do not assume we are given a passage together with the question

Open Domain Question Answering

Recap



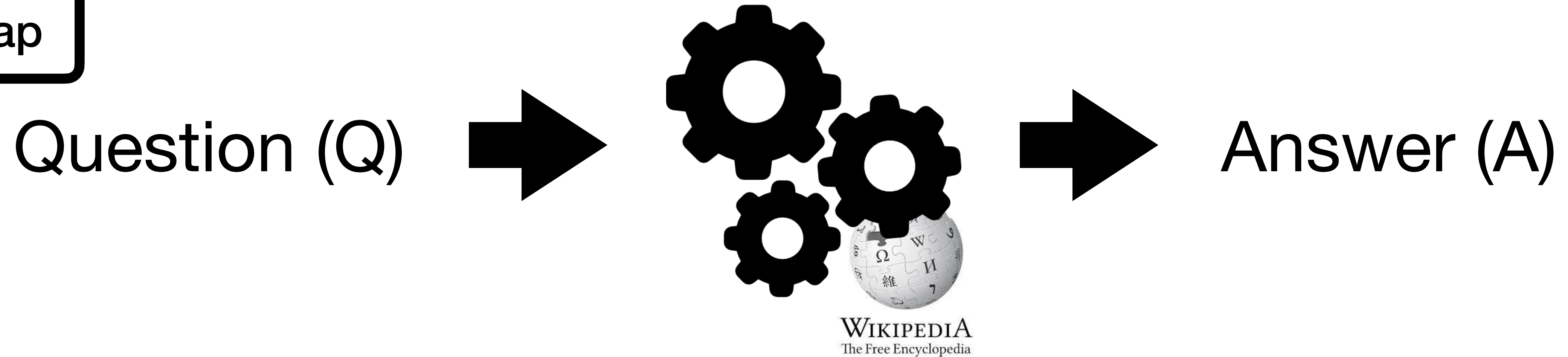
Open-Domain Question Answering (ODQA):

We do not assume we are given a passage together with the question

We can only access a large collection of documents (e.g., Wikipedia) — we don't know which document contains the answer, and the goal is to answer any open-domain questions.

Open Domain Question Answering

Recap



Open-Domain Question Answering (ODQA):

We do not assume we are given a passage together with the question

We can only access a large collection of documents (e.g., Wikipedia) — we don't know which document contains the answer, and the goal is to answer any open-domain questions.

Both more challenging and more practical/useful!

Slot Filling

INPUT:
Star Trek [SEP] creator

OUTPUT:
Gene Roddenberry

PROVENANCE:
17157886-1

zsRE

Open Domain QA

INPUT:
When did Star Trek go off the air

OUTPUT:
June 3, 1969

PROVENANCE:
17157886-5

NQ

TQA

INPUT:
Which Star Trek star directed Three Men and a Baby?

OUTPUT:
Leonard Nimoy

PROVENANCE:
17157886-4, 596639-7

INPUT:
Treklanda (formerly “TrekTrax Atlanta”) is an annual convention for what American science fiction media franchise?

OUTPUT:
Star Trek

 KILT

Knowledge source:
5.9 Million Wikipedia pages

Star Trek 17157886

Star Trek is an American media franchise based on the science fiction television series created by Gene Roddenberry.¹ [...] It followed the interstellar adventures of Captain James T. Kirk (William Shatner) and his crew aboard the starship USS "Enterprise", a space exploration vessel built by the United Federation of Planets in the 23rd century.² The "Star Trek" canon includes "The Original Series", an animated series, five spin-off television series, the film franchise, and further adaptations in several media.³ [...] The original 1966–69 series featured William Shatner as Captain James T. Kirk, Leonard Nimoy⁴ as Spock, DeForest Kelley as Dr. Leonard "Bones" McCoy, James Doohan as Montgomery "Scotty" Scott, Nichelle Nichols as Uhura, George Takei as Hikaru Sulu, and Walter Koenig as Pavel Chekov. During the series' first run, it earned several nominations for the Hugo Award for Best Dramatic Presentation, and won twice. [...] NBC canceled the show after three seasons; the last original episode aired on June 3, 1969⁵. [...]

Three Men and a Baby 596639

Three Men and a Baby is a 1987 American comedy film directed by Leonard Nimoy⁷ and starring Tom Selleck, Steve Guttenberg, Ted Danson and Nancy Travis. [...]

Treklanda 28789994

Treklanda is an annual "Star Trek" convention based in Atlanta, Georgia that

Dialogue

INPUT:
I am a big fan of Star Trek, the American franchise created by Gene Roddenberry. I don't know much about it. When did the first episode air?
It debuted in 1996 and aired for 3 seasons on NBC.
What is the plot of the show?

OUTPUT:
William Shatner plays the role of Captain Kirk. He did a great job.

PROVENANCE:
17157886-2

WoW

Fact Checking

INPUT:
Star Trek had spin-off television series.

OUTPUT:
Supports

PROVENANCE:
17157886-3

FEV

Entity Linking

INPUT:
[...] Currently the site offers five movie collections ranging from \$149 for 10 [START_ENT] Star Trek [END_ENT] films to \$1,125 for the eclectic Movie Lovers' Collection of 75 movies. [...]

OUTPUT:
Star Trek

- # The KILT Benchmark
- Open-Domain Question Answering (Natural Questions, TriviaQA, HotPotQA, ELI5)
 - Fact-Checking (FEVER)
 - Slot Filling (T-REx, zsRE)
 - Dialogue (Wizard of Wikipedia)
 - Entity Linking (AIDA, WNED-WIKI, WNED-CWEB)

LLMs and their Limitations

LLMs are Extremely Impressive —

- ✅ They can store vast amounts of knowledge in their parameters/activations
- ✅ Very strong results on many tasks, even in few-shot learning settings
- ✅ Very flexible — applicable on a variety of tasks

LLMs and their Limitations

LLMs are Extremely Impressive —

- ✅ They can store vast amounts of knowledge in their parameters/activations
- ✅ Very strong results on many tasks, even in few-shot learning settings
- ✅ Very flexible — applicable on a variety of tasks

However —

- ❌ It can be difficult to update and control their knowledge/memory
- ❌ LLMs are black-boxes — no provenance or interpretability
- ❌ Very large and expensive

LLMs and their Limitations

Input: List the top five US states with the highest per-capita GDP, in order.

[Asai et al., 2024]

LLMs and their Limitations

Input: List the top five US states with the highest per-capita GDP, in order.

Parametric LMs : Pre-trained on large-scale pre-training data



LLMs and, more generally, neural models

LLMs and their Limitations

Input: List the top five US states with the highest per-capita GDP, in order.

Parametric LMs: Pre-trained on large-scale pre-training data



LLMs and, more generally, neural models

Retrieval-augmented LMs: Incorporate data at inference



Retrieval-Augmented Generation models

[Asai et al., 2024]

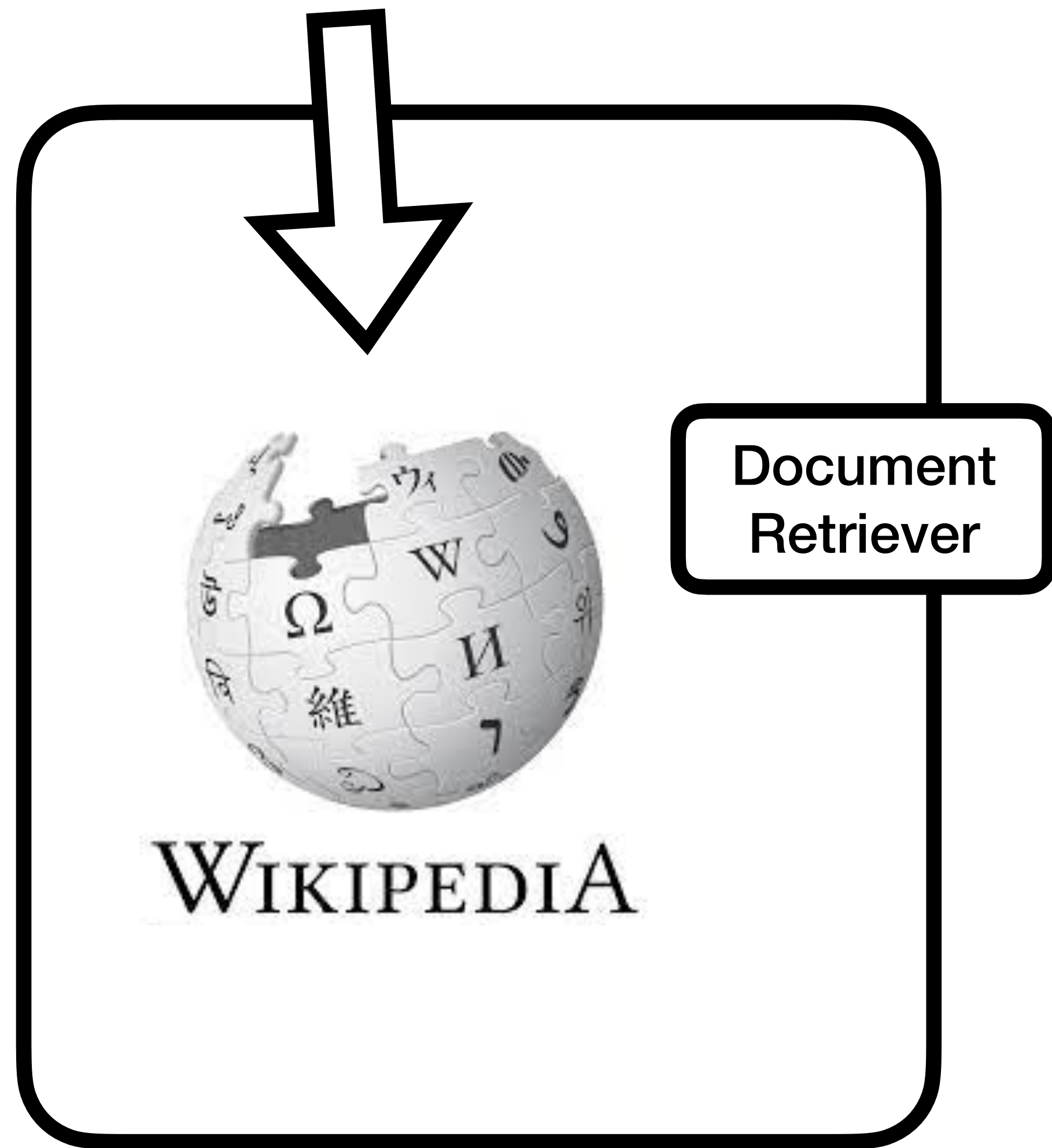
The Retriever-Reader Framework

“In what city is the University of Edinburgh located?”

[Chen et al., 2017]

The Retriever-Reader Framework

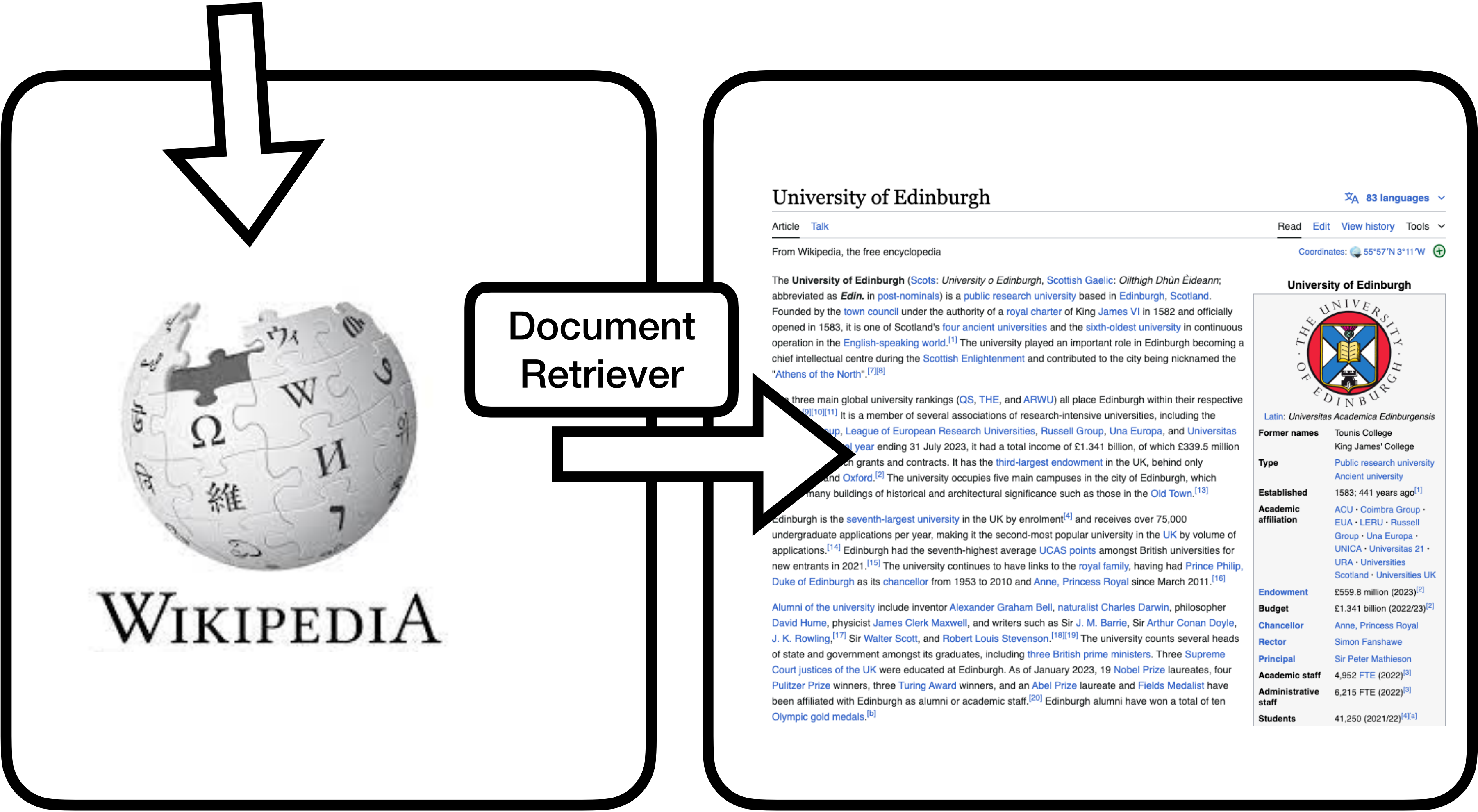
“In what city is the University of Edinburgh located?”



[Chen et al., 2017]

The Retriever-Reader Framework

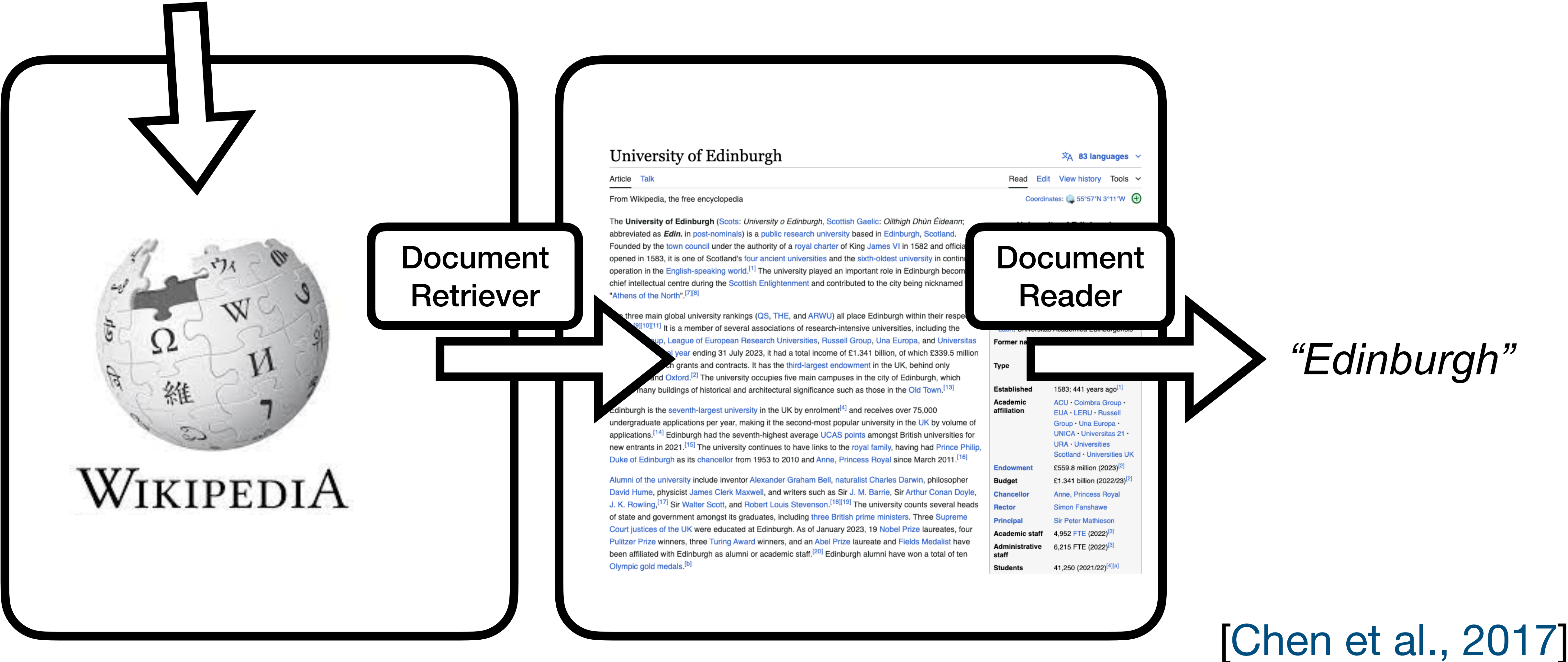
“In what city is the University of Edinburgh located?”



[Chen et al., 2017]

The Retriever-Reader Framework

“In what city is the University of Edinburgh located?”



The Retriever-Reader Framework

Input: a large collection of documents $\mathcal{D} = \{D_1, \dots, D_n\}$ and a question Q

Output: an answer A

The Retriever-Reader Framework

Input: a large collection of documents $\mathcal{D} = \{D_1, \dots, D_n\}$ and a question Q

Output: an answer A

Retriever: $\text{retriever}(\mathcal{D}, Q) \rightarrow P_1, \dots, P_k$, where $k \in \mathbb{N}$ is pre-defined (e.g., 100)

Reader: $\text{reader}(Q, \{P_1, \dots, P_k\}) \rightarrow A$, similar to reading comprehension

The Retriever-Reader Framework

Input: a large collection of documents $\mathcal{D} = \{D_1, \dots, D_n\}$ and a question Q

Output: an answer A

Retriever: $\text{retriever}(\mathcal{D}, Q) \rightarrow P_1, \dots, P_k$, where $k \in \mathbb{N}$ is pre-defined (e.g., 100)

Reader: $\text{reader}(Q, \{P_1, \dots, P_k\}) \rightarrow A$, similar to reading comprehension

An early retriever-reader system is **DrQA** [Chen et al., 2017]:

Retriever: a standard, “classic” TF-IDF information retrieval module (fixed)

Reader: a neural reading comprehension model, trained on SQuAD via distant supervision (i.e., by using retrieved paragraphs rather than gold ones)

Dense and Sparse Retrievers

Goal: find a small subset of elements (e.g., documents, paragraphs) in a datastore that are the most similar/related/relevant to the query

$\text{sim}(Q, P)$: similarity score between a query Q and a paragraph P

Dense and Sparse Retrievers

Goal: find a small subset of elements (e.g., documents, paragraphs) in a datastore that are the most similar/related/relevant to the query

$\text{sim}(Q, P)$: similarity score between a query Q and a paragraph P

Example: TF-IDF similarity (sparse)

$$\text{sim}(Q_i, P_j) = \text{cosine}(\mathbf{q}, \mathbf{p}) \text{ with } \mathbf{q}, \mathbf{p} \in \mathbb{R}^{|V|}$$

Dense and Sparse Retrievers

Goal: find a small subset of elements (e.g., documents, paragraphs) in a datastore that are the most similar/related/relevant to the query

$\text{sim}(Q, P)$: similarity score between a query Q and a paragraph P

Example: TF-IDF similarity (sparse)

$$\text{sim}(Q_i, P_j) = \text{cosine}(\mathbf{q}, \mathbf{p}) \text{ with } \mathbf{q}, \mathbf{p} \in \mathbb{R}^{|V|}$$

$$\mathbf{q}_w = \text{TF}(w, Q) \cdot \text{IDF}(w, \mathcal{D})$$

Term Frequency

Inverse Document
Frequency

$$\text{TF}(w, Q) = \frac{\text{freq}(w, Q)}{\sum_{w'} \text{freq}(w', Q)}$$

$$\text{IDF}(w, \mathcal{D}) = \log \frac{|\mathcal{D}|}{|\{P \in \mathcal{D} \wedge w \in P\}|}$$

Dense Retrieval in Practice

Goal: find a small subset of elements (e.g., documents, paragraphs) in a datastore that are the most similar/related/relevant to the query

$\text{sim}(Q, P)$: similarity score between a query Q and a paragraph P

Example: Dense Retrieval

$$\text{sim}(Q_i, P_j) = \mathbf{q}_i^\top \mathbf{p}_j \text{ with } \mathbf{q}_i, \mathbf{p}_j \in \mathbb{R}^d$$

Neural network is used to calculate the query embedding and the paragraph embedding

Dense Retrieval in Practice

Goal: find a small subset of elements (e.g., documents, paragraphs) in a datastore that are the most similar/related/relevant to the query

$\text{sim}(Q, P)$: similarity score between a query Q and a paragraph P

Example: Dense Retrieval

$$\text{sim}(Q_i, P_j) = \mathbf{q}_i^\top \mathbf{p}_j \text{ with } \mathbf{q}_i, \mathbf{p}_j \in \mathbb{R}^d$$

$$\mathbf{q}_i = \text{Encode}(Q_i)$$

$$\mathbf{p}_j = \text{Encode}(P_j)$$

Entire research on how to improve or learn the similarity function!

Dense Retrieval in Practice

Goal: find a small subset of elements (e.g., documents, paragraphs) in a datastore that are the most similar/related/relevant to the query

$\text{sim}(Q, P)$: similarity score between a query Q and a paragraph P

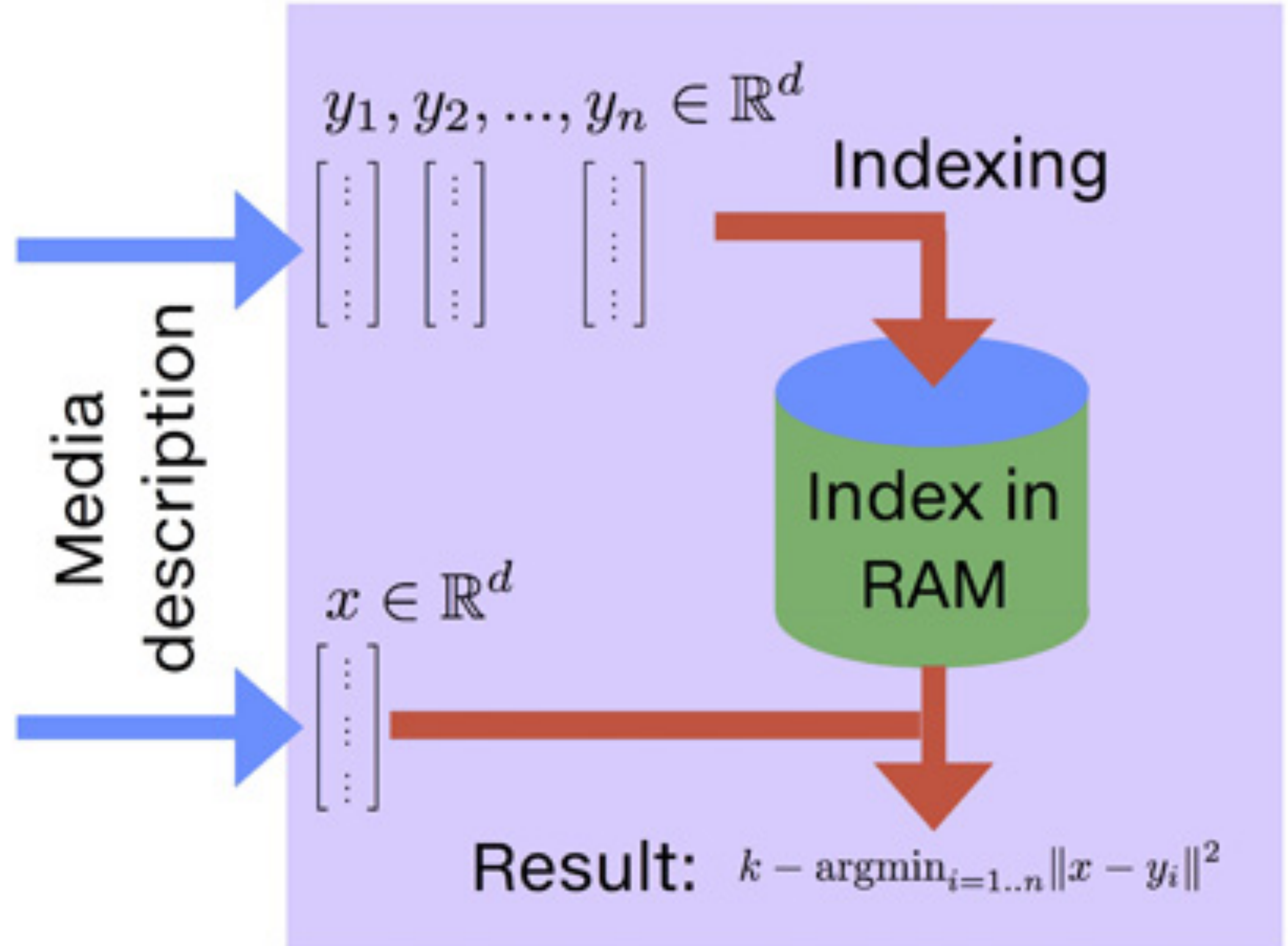
Index: given a query embedding $\mathbf{q}_i \in \mathbb{R}^d$, returns the top- k paragraph embeddings $\mathbf{p}_1, \dots, \mathbf{p}_k \in \mathbb{R}^d$ via **maximum inner-product search** (MIPS)

Software: FAISS, ScaNN, Annoy, ...

Build index for a collection:



Query:



Software: FAISS, ScaNN, Annoy, ...

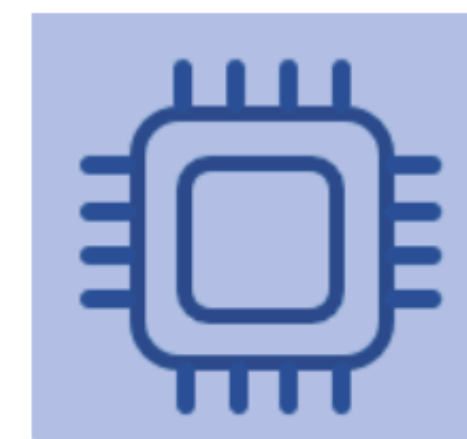
Summary of methods

The basic indexes are given hereafter:

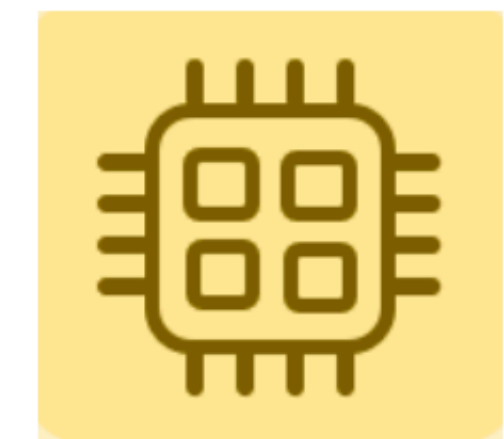
Method	Class name	index_factory	Main parameters	Bytes/vector	Exhaustive
Exact Search for L2	IndexFlatL2	"Flat"	d	4*d	yes
Exact Search for Inner Product	IndexFlatIP	"Flat"	d	4*d	yes
Hierarchical Navigable Small World graph exploration	IndexHNSWFlat	"HNSW,Flat"	d, M	$4*d + x * M * 2 * 4$	no
Inverted file with exact post-verification	IndexIVFFlat	"IVFx,Flat"	quantizer, d, nlists, metric	$4*d + 8$	no
Locality-Sensitive Hashing (binary flat index)	IndexLSH	-	d, nbits	$\text{ceil}(\text{nbits}/8)$	yes
Scalar quantizer (SQ) in flat mode	IndexScalarQuantizer	"SQ8"	d	d	yes
Product quantizer (PQ) in flat mode	IndexPQ	"PQx", "PQM"x"nbits"	d, M, nbits	$\text{ceil}(M * \text{nbits} / 8)$	yes

Exact Search

Approximate Search
(Scales to Billions of vectors)

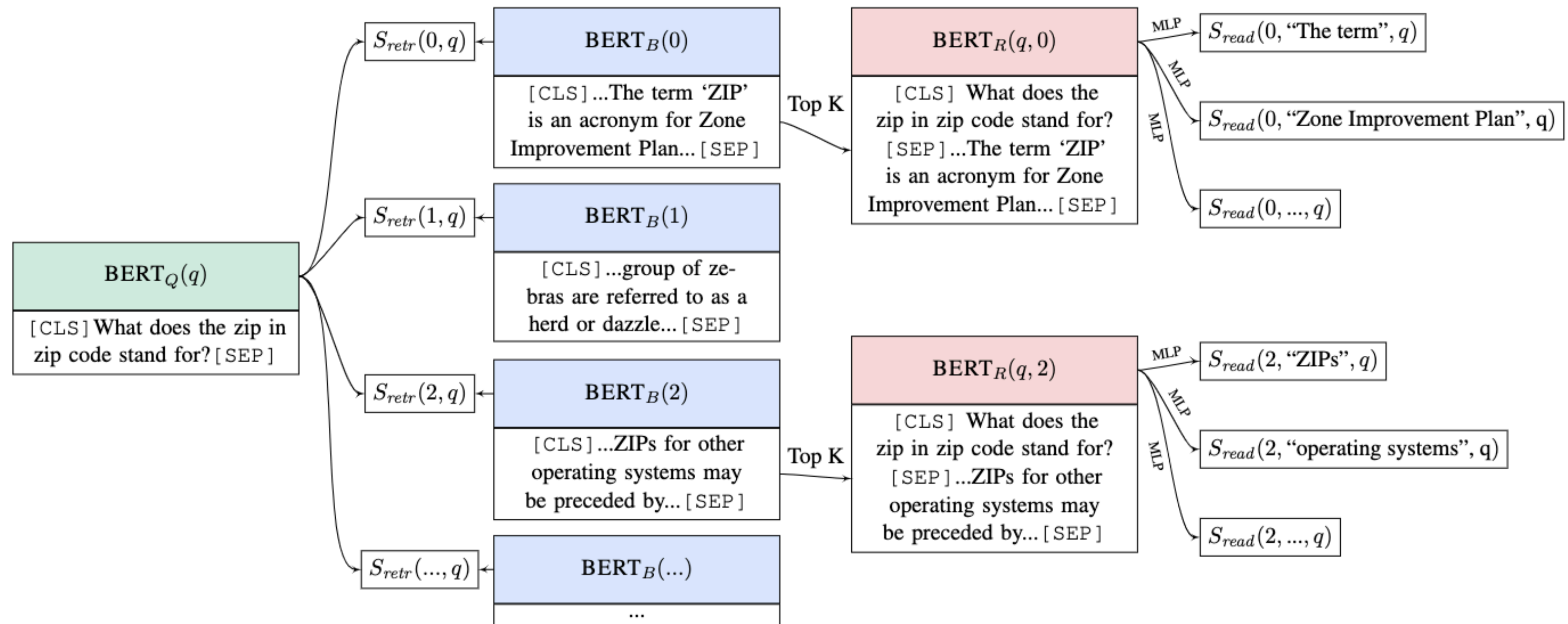


CPU vs. GPU



Early End-to-End Trainable Reader-Retriever Models

Early method for training the retrieval component proposed by [Lee et al., 2019](#):

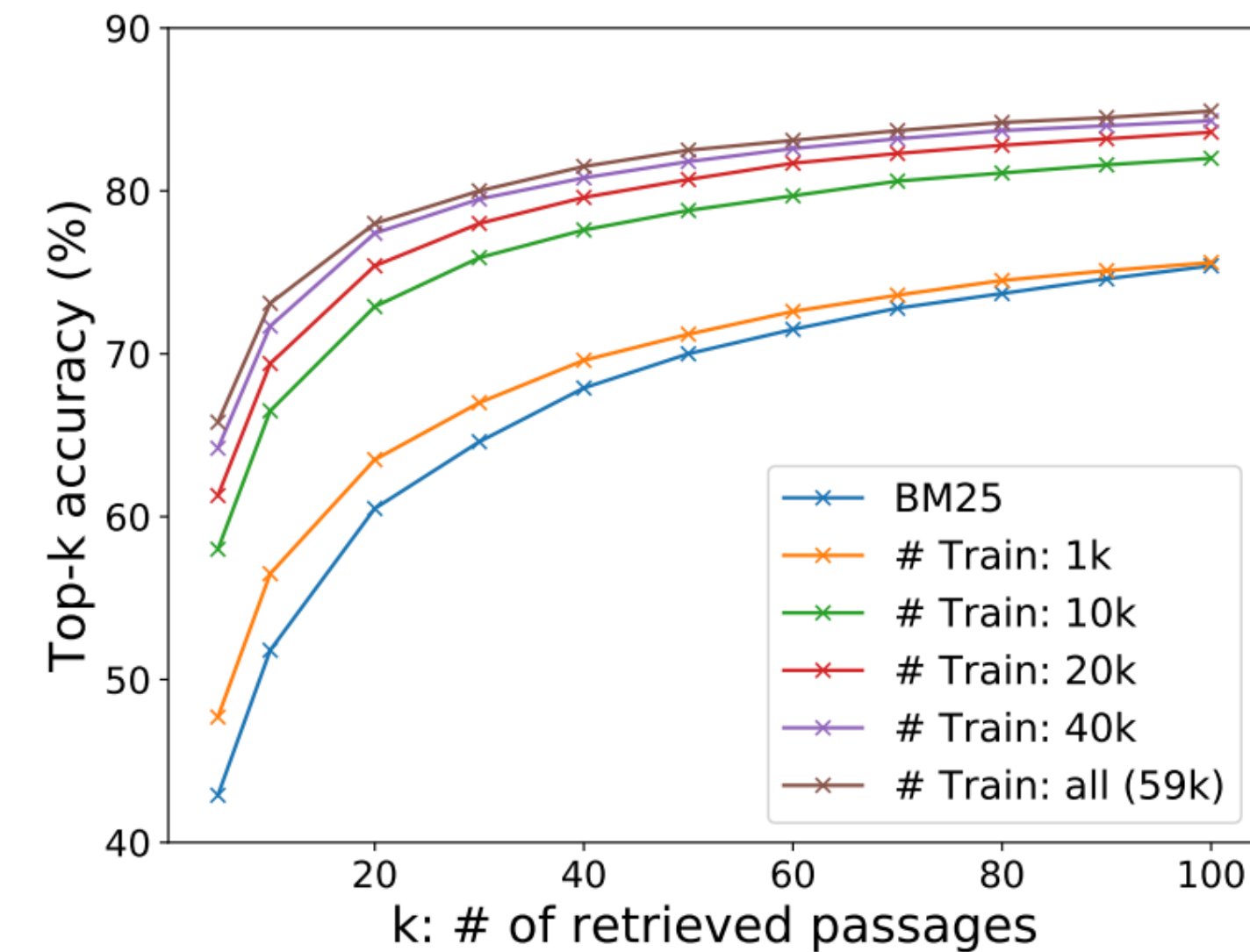
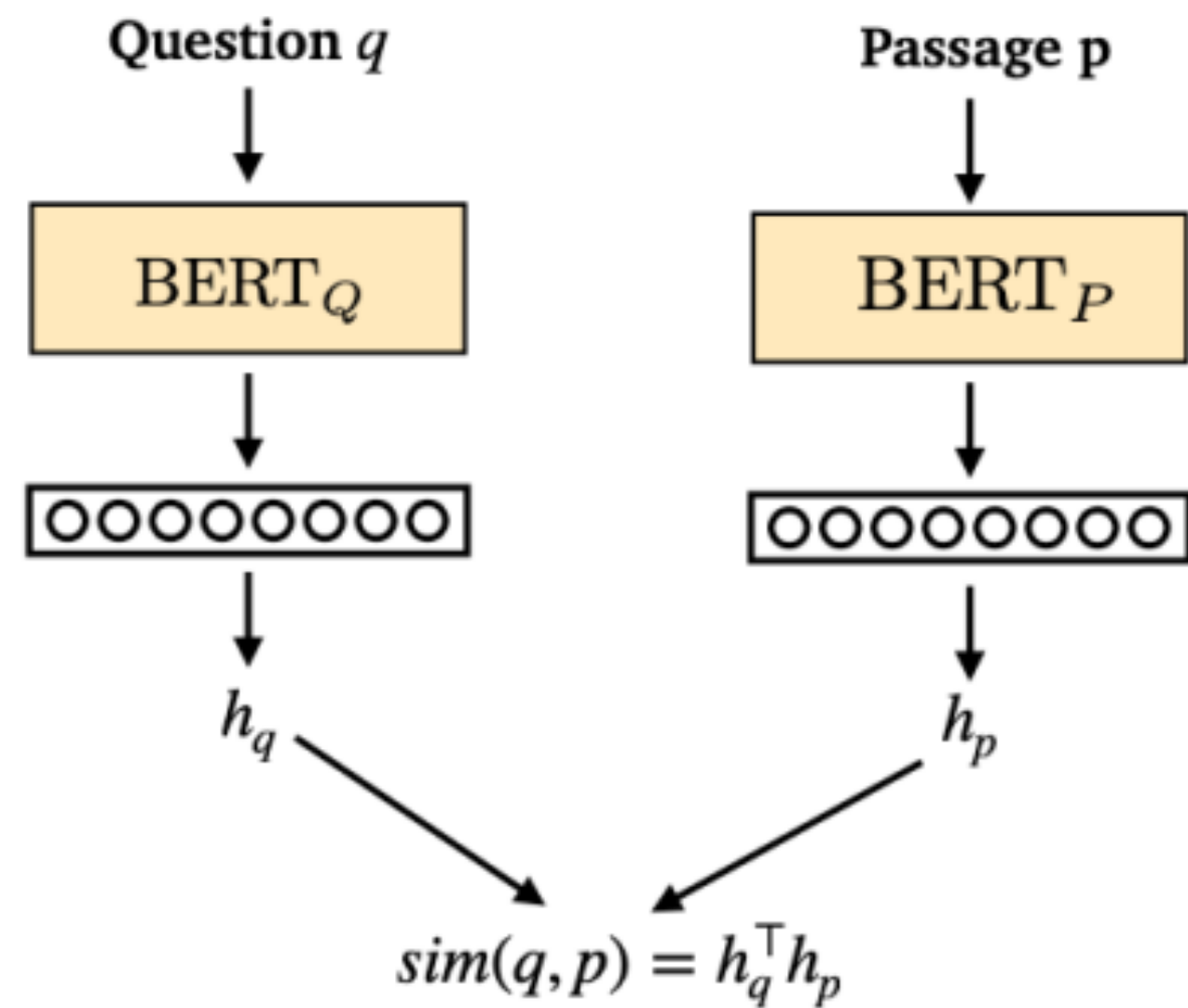


Each passage can be encoded as a vector using BERT and the retriever score can be measured as the dot product between the question and passage representations

Not easy to model as there are a huge number of passages (21M in Eng. Wikipedia)

Early End-to-End Trainable Reader-Retriever Models

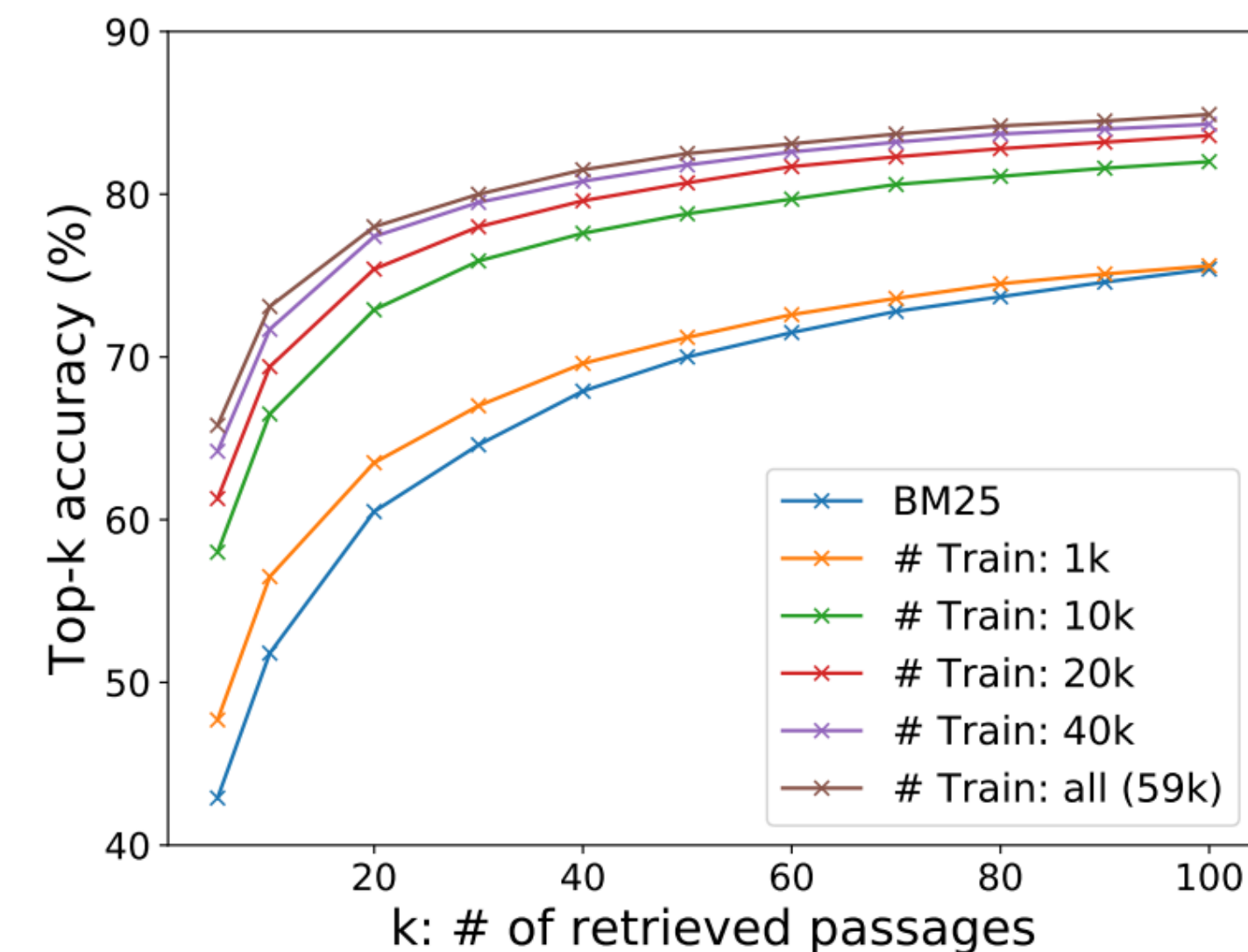
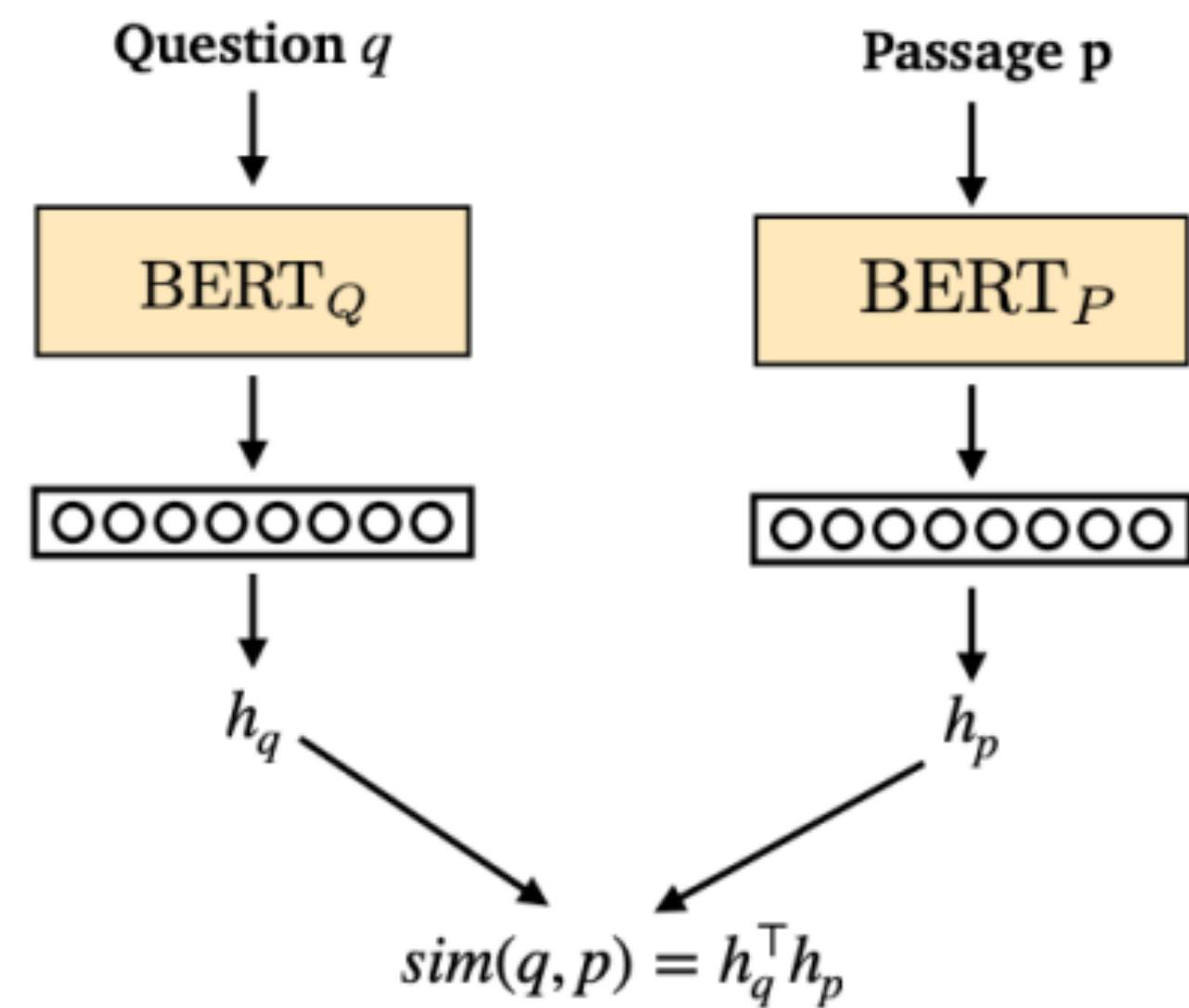
Later, Dense Passage Retrieval [DPR, [Karpukhin et al., 2020](#)] authors propose to **train the retriever using question-answer pairs**:



Trainable retriever (using BERT) can produce more accurate results than traditional IR models, such as BM25 and TF-IDF

Early End-to-End Trainable Reader-Retriever Models

Later, Dense Passage Retrieval [DPR, [Karpukhin et al., 2020](#)] authors propose to **train the retriever using question-answer pairs**:



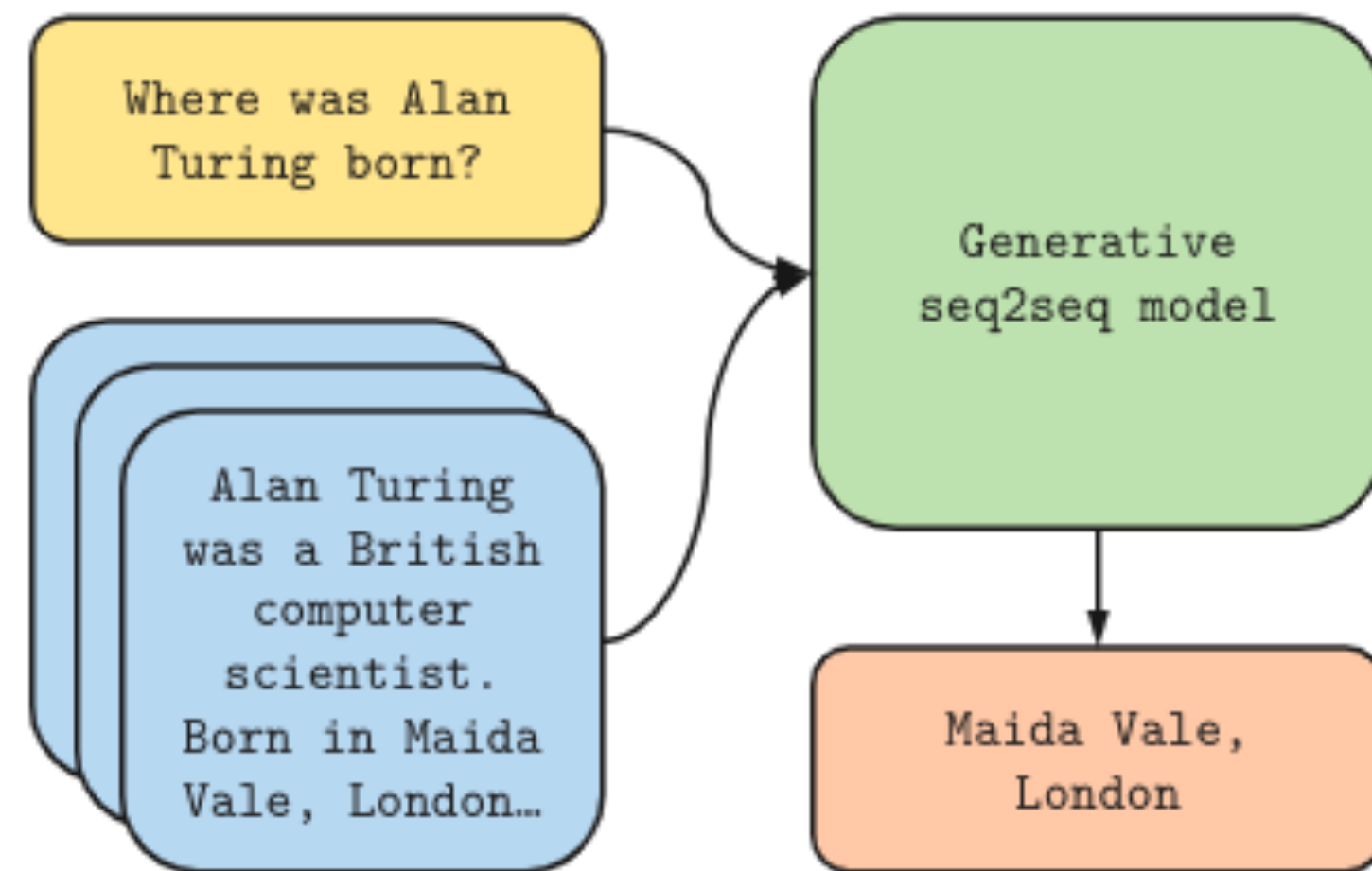
Trainable retriever (using BERT) can produce more accurate results than traditional IR models, such as BM25 and TF-IDF

...although this was slightly controversial — see e.g., “A Replication Study of DPR”

<https://arxiv.org/abs/2104.05740>

Dense Retrieval and Generative Models

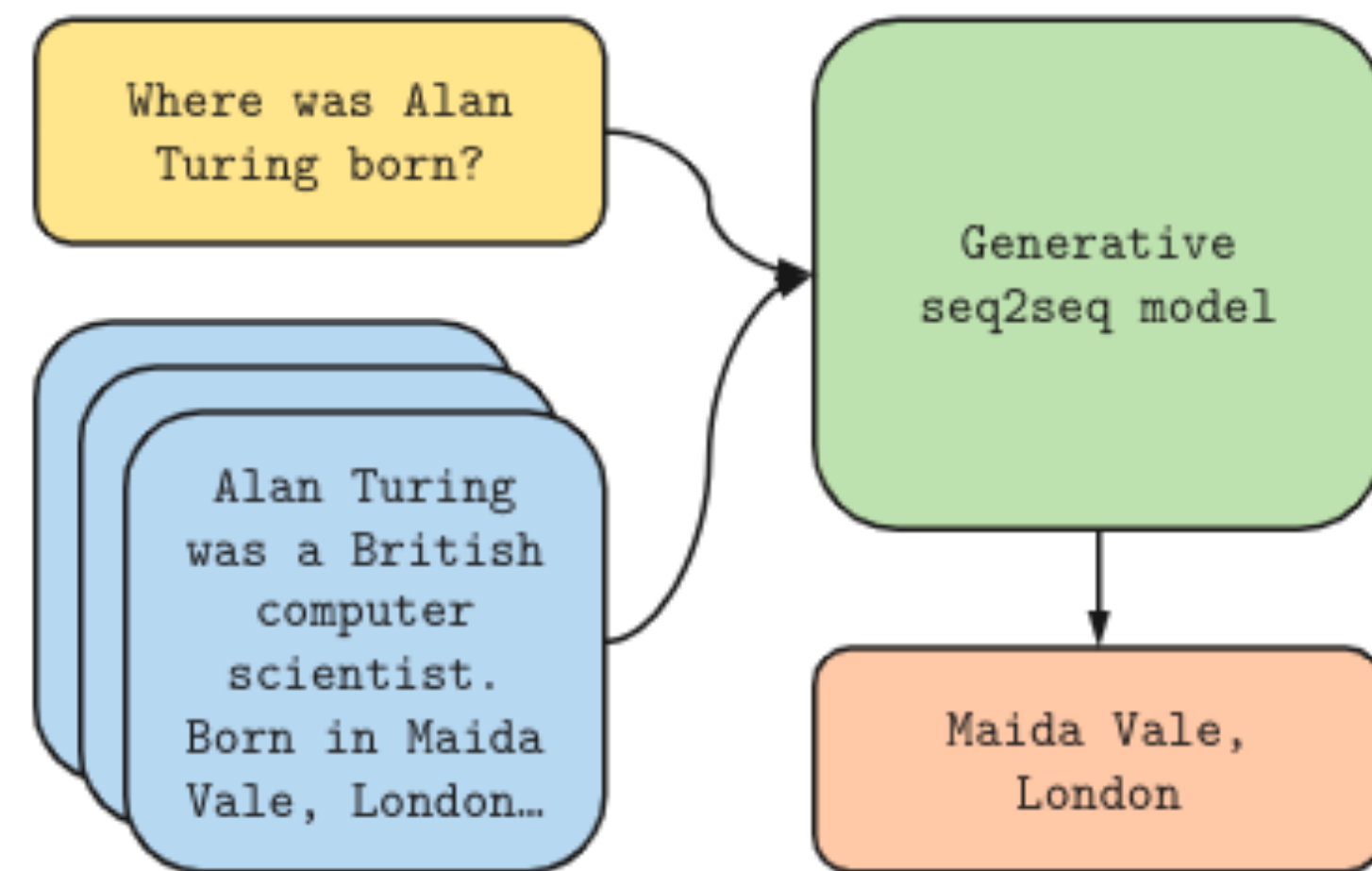
Recent works show that it can be beneficial to **generate answers** rather than **extracting them** from retrieved passages, e.g., Fusion-in-Decoder [[Izacard et al., 2021](#)]



Fusion-in-Decoder (FiD):
DPR & T5

Dense Retrieval and Generative Models

Recent works show that it can be beneficial to **generate answers** rather than **extracting them** from retrieved passages, e.g., Fusion-in-Decoder [[Izacard et al., 2021](#)]

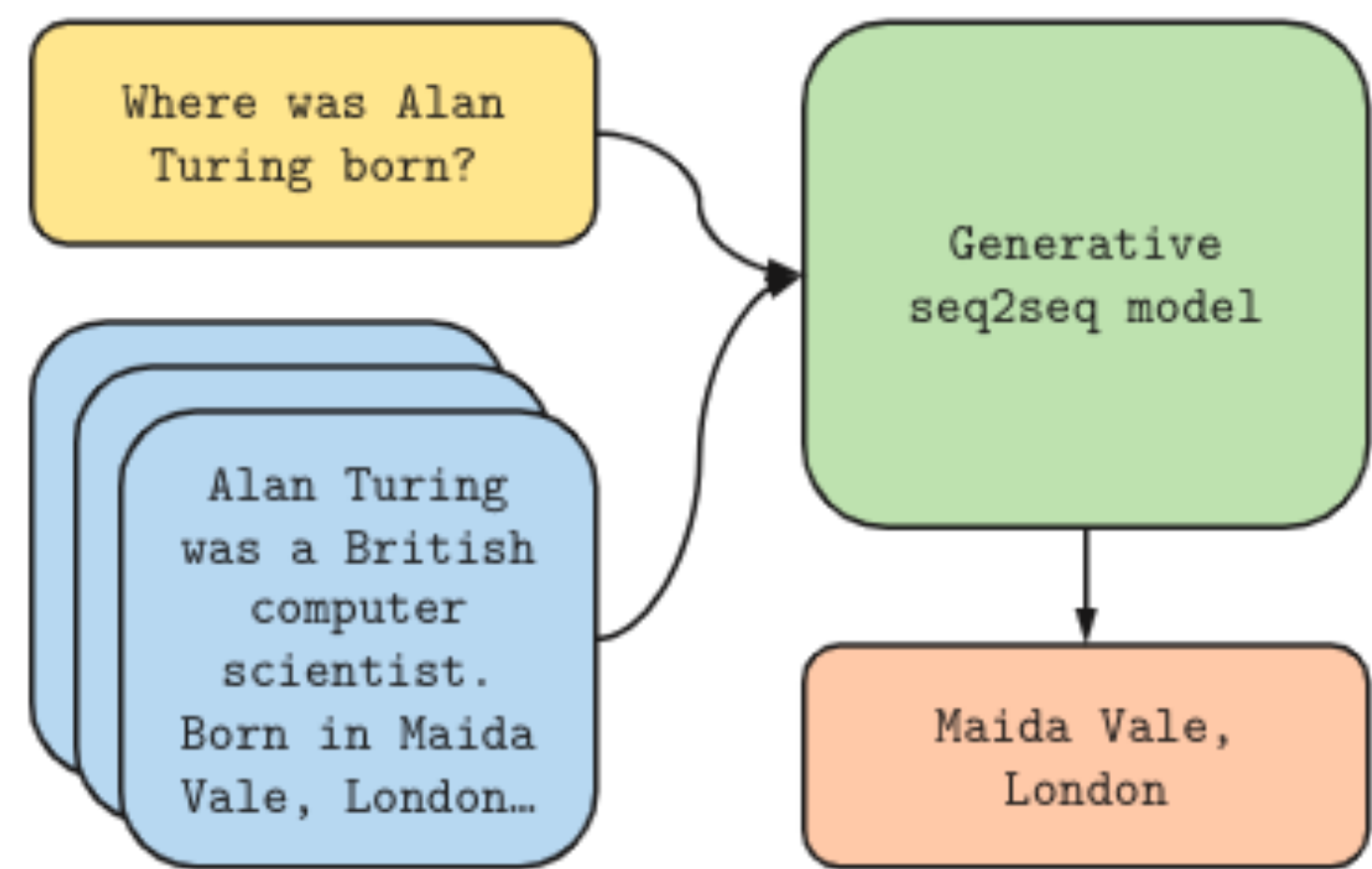


Fusion-in-Decoder (FiD):
DPR & T5

Model	NQ EM	TriviaQA EM	EM	SQuAD Open EM	F1
DrQA (Chen et al., 2017)	-	-	-	29.8	-
Multi-Passage BERT (Wang et al., 2019)	-	-	-	53.0	60.9
Path Retriever (Asai et al., 2020)	31.7	-	-	56.5	63.8
Graph Retriever (Min et al., 2019b)	34.7	55.8	-	-	-
Hard EM (Min et al., 2019a)	28.8	50.9	-	-	-
ORQA (Lee et al., 2019)	31.3	45.1	-	20.2	-
REALM (Guu et al., 2020)	40.4	-	-	-	-
DPR (Karpukhin et al., 2020)	41.5	57.9	-	36.7	-
SpanSeqGen (Min et al., 2020)	42.5	-	-	-	-
RAG (Lewis et al., 2020b)	44.5	56.1	68.0	-	-
T5 (Roberts et al., 2020)	36.6	-	60.5	-	-
GPT-3 few shot (Brown et al., 2020)	29.9	-	71.2	-	-
Fusion-in-Decoder (base)	48.2	65.0	77.1	53.4	60.6
Fusion-in-Decoder (large)	51.4	67.6	80.1	56.7	63.2

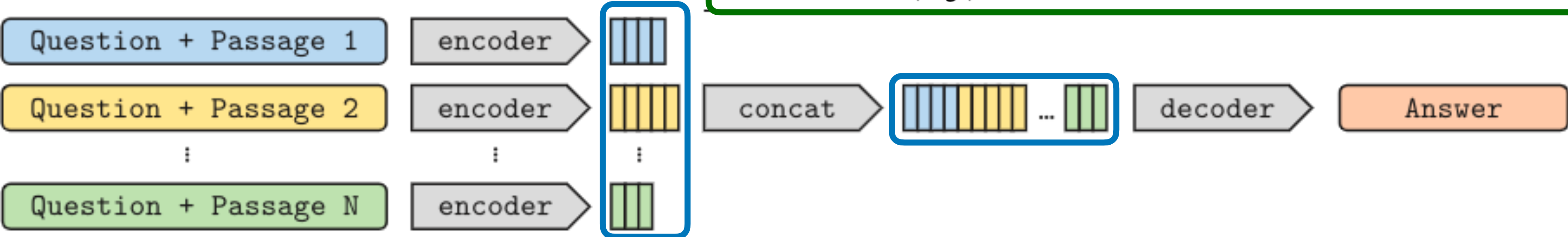
Dense Retrieval and Generative Models

Recent works show that it can be beneficial to **generate answers** rather than **extracting them** from retrieved passages, e.g., Fusion-in-Decoder [[Izacard et al., 2021](#)]



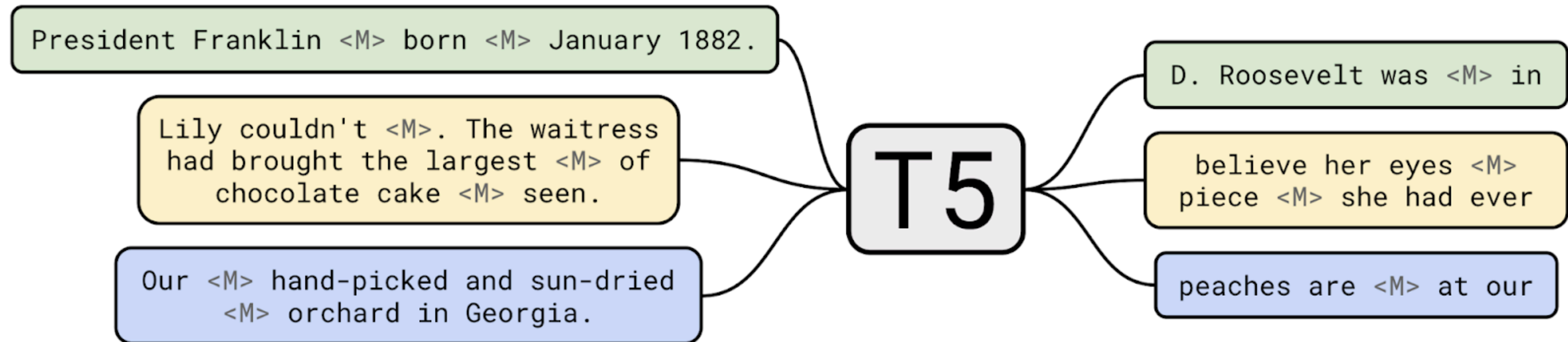
Fusion-in-Decoder (FiD):
DPR & T5

Model	NQ EM	TriviaQA EM		SQuAD Open EM F1	
DrQA (Chen et al., 2017)	-	-	-	29.8	-
Multi-Passage BERT (Wang et al., 2019)	-	-	-	53.0	60.9
Path Retriever (Asai et al., 2020)	31.7	-	-	56.5	63.8
Graph Retriever (Min et al., 2019b)	34.7	55.8	-	-	-
Hard EM (Min et al., 2019a)	28.8	50.9	-	-	-
ORQA (Lee et al., 2019)	31.3	45.1	-	20.2	-
REALM (Guu et al., 2020)	40.4	-	-	-	-
DPR (Karpukhin et al., 2020)	41.5	57.9	-	36.7	-
SpanSeqGen (Min et al., 2020)	42.5	-	-	-	-
RAG (Lewis et al., 2020b)	44.5	56.1	68.0	-	-
T5 (Roberts et al., 2020)	36.6	-	60.5	-	-
GPT-3 few shot (Brown et al., 2020)	29.9	-	71.2	-	-
Fusion-in-Decoder (base)	48.2	65.0	77.1	53.4	60.6
Fusion-in-Decoder (large)	51.4	67.6	80.1	56.7	63.2



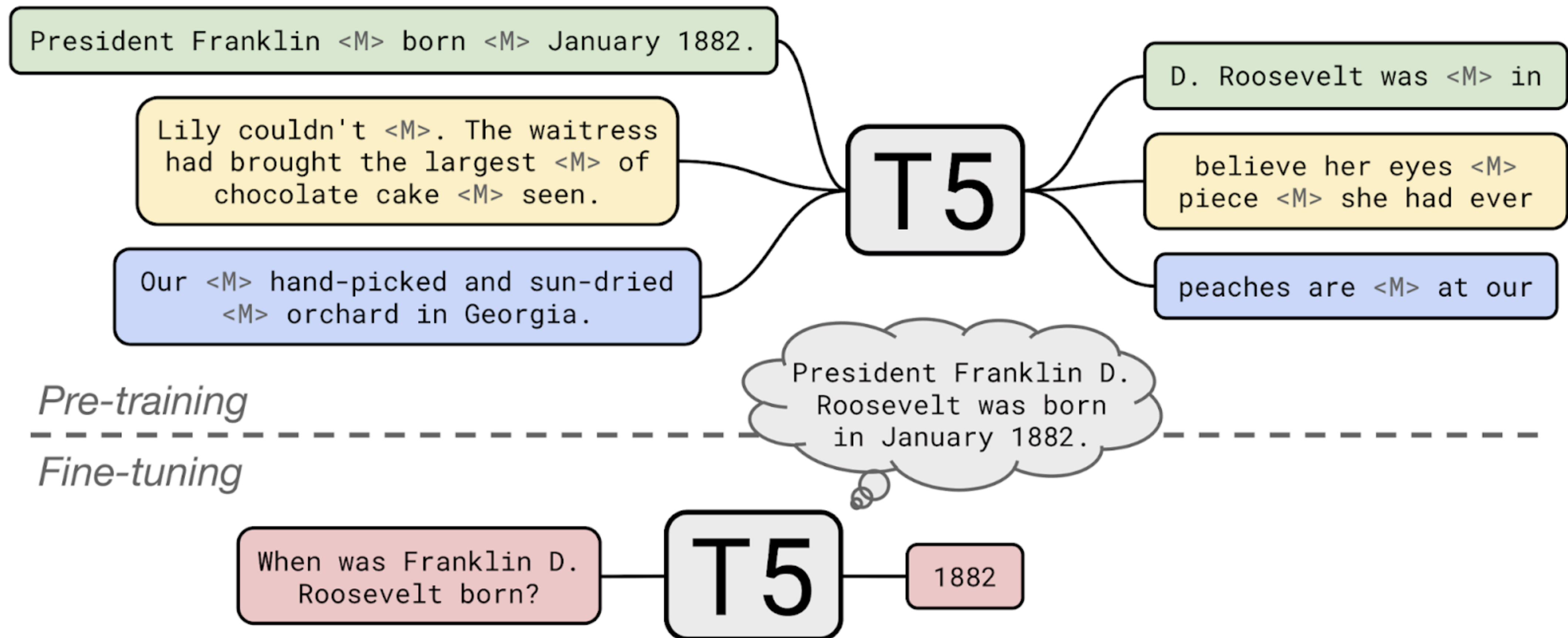
LLMs Can Do Open-Domain QA

LLMs — without an (explicit) retrieval component — can be used to solve Open-Domain Question Answering tasks; knowledge about the world is encoded in their parameters and activations, rather than in a corpus:



LLMs Can Do Open-Domain QA

LLMs — without an (explicit) retrieval component — can be used to solve Open-Domain Question Answering tasks; knowledge about the world is encoded in their parameters and activations, rather than in a corpus:



[Roberts et al., 2020]

LLMs vs. RAG — Generalisation

How do LLMs **really** compare with RAG models, in terms of accuracy and generalisation, on open-domain question answering tasks?

Model	Open Natural Questions				TriviaQA				WebQuestions			
	Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap
The question appears in the training set				The answer appears in the training set								

LLMs vs. RAG — Generalisation

How do LLMs **really** compare with RAG models, in terms of accuracy and generalisation, on open-domain question answering tasks?

Model		Open Natural Questions				TriviaQA				WebQuestions			
		Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap
		The question appears in the training set		The answer appears in the training set									
Closed book	T5-11B+SSM	36.6	77.2	22.2	9.4	-	-	-	-	44.7	82.1	44.5	22.0
	BART	26.5	67.6	10.2	0.8	26.7	67.3	16.3	0.8	27.4	71.5	20.7	1.6

LLMs vs. RAG — Generalisation

How do LLMs **really** compare with RAG models, in terms of accuracy and generalisation, on open-domain question answering tasks?

Model		Open Natural Questions				TriviaQA				WebQuestions			
		Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap
Open book	RAG	44.5	70.7	34.9	24.8	56.8	82.7	54.7	29.2	45.5	81.0	45.8	21.1
	DPR	41.3	69.4	34.6	19.3	57.9	80.4	59.6	31.6	42.4	74.1	39.8	22.2
	FID	51.4	71.3	48.3	34.5	67.6	87.5	66.9	42.8	-	-	-	-
Closed book	T5-11B+SSM	36.6	77.2	22.2	9.4	-	-	-	-	44.7	82.1	44.5	22.0
	BART	26.5	67.6	10.2	0.8	26.7	67.3	16.3	0.8	27.4	71.5	20.7	1.6

LLMs vs. RAG — Generalisation

How do LLMs **really** compare with RAG models, in terms of accuracy and generalisation, on open-domain question answering tasks?

Model		Natural Questions				TriviaQA			
		Total	Overlap	Comp -gen	Novel -entity	Total	Overlap	Comp -gen	Novel -entity
				Unseen entity- relation pair				New entity	
Parametric	T5-11B+SSM	36.59	81.48	17.47	12.56	-	-	-	-
	BART	26.54	76.34	5.88	3.35	26.78	78.38	11.37	10.09

[Liu et al., 2021]

LLMs vs. RAG — Generalisation

How do LLMs **really** compare with RAG models, in terms of accuracy and generalisation, on open-domain question answering tasks?

Model		Natural Questions				TriviaQA			
		Total	Overlap	Comp -gen	Novel -entity	Total	Overlap	Comp -gen	Novel -entity
Non-parametric	RAG	44.49	75.75	30.41	37.69	56.83	87.12	47.58	47.81
	FiD	53.13	78.85	40.00	47.74	67.69	90.39	58.10	66.23
	DPR	41.27	71.33	25.88	33.84	57.91	82.31	46.11	58.99
	RePAQ	47.26	78.61	34.21	36.85	52.06	89.08	42.95	38.38
Parametric	T5-11B+SSM	36.59	81.48	17.47	12.56	-	-	-	-
	BART	26.54	76.34	5.88	3.35	26.78	78.38	11.37	10.09

LLMs vs. RAG — Updateability

Train Set	Test-time Index	2017 Test Set Acc.		2020 Test Set Acc.	
		Closed-book	ATLAS	Closed-book	ATLAS
2017 answers	2017	T5-XXL	Retrieval-augmented		
	2020				
2020 answers	2017				
	2020				

LLMs vs. RAG — Updateability

Train Set	Test-time Index	2017 Test Set Acc.		2020 Test Set Acc.	
		Closed-book	ATLAS	Closed-book	ATLAS
2017 answers	2017	12.1	Retrieval-augmented	2.9	
	2020	12.1		2.9	
2020 answers	2017	4.8		3.6	
	2020	4.8		3.6	

LLMs vs. RAG — Updateability

Train Set	Test-time Index	2017 Test Set Acc.		2020 Test Set Acc.	
		Closed-book	ATLAS	Closed-book	ATLAS
2017 answers	2017	12.1	57.7	2.9	1.5
	2020	12.1	10.2	2.9	53.1
2020 answers	2017	4.8	50.1	3.6	4.2
	2020	4.8	3.5	3.6	60.5

LLMs vs. RAG — Accuracy

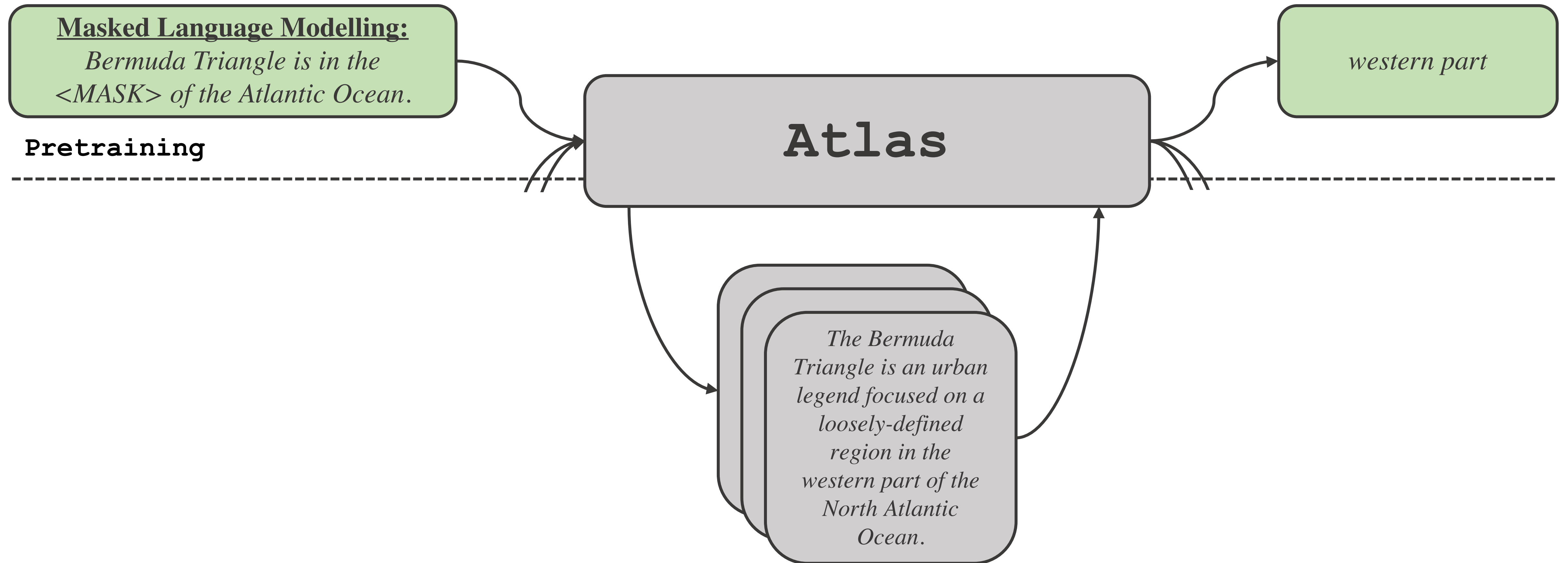
Model	NQ		TriviaQA filtered		TriviaQA unfiltered	
	64-shot	Full	64-shot	Full	64-shot	Full
GPT-3 (Brown et al., 2020)	29.9	-	-	-	71.2	-
Gopher (Rae et al., 2021)	28.2	-	57.2	-	61.3	-
Chinchilla (Hoffmann et al., 2022)	35.5	-	64.6	-	72.3	-
PaLM (Chowdhery et al., 2022)	39.6	-	-	-	81.4	-
RETRO (Borgeaud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

LLMs vs. RAG — Accuracy

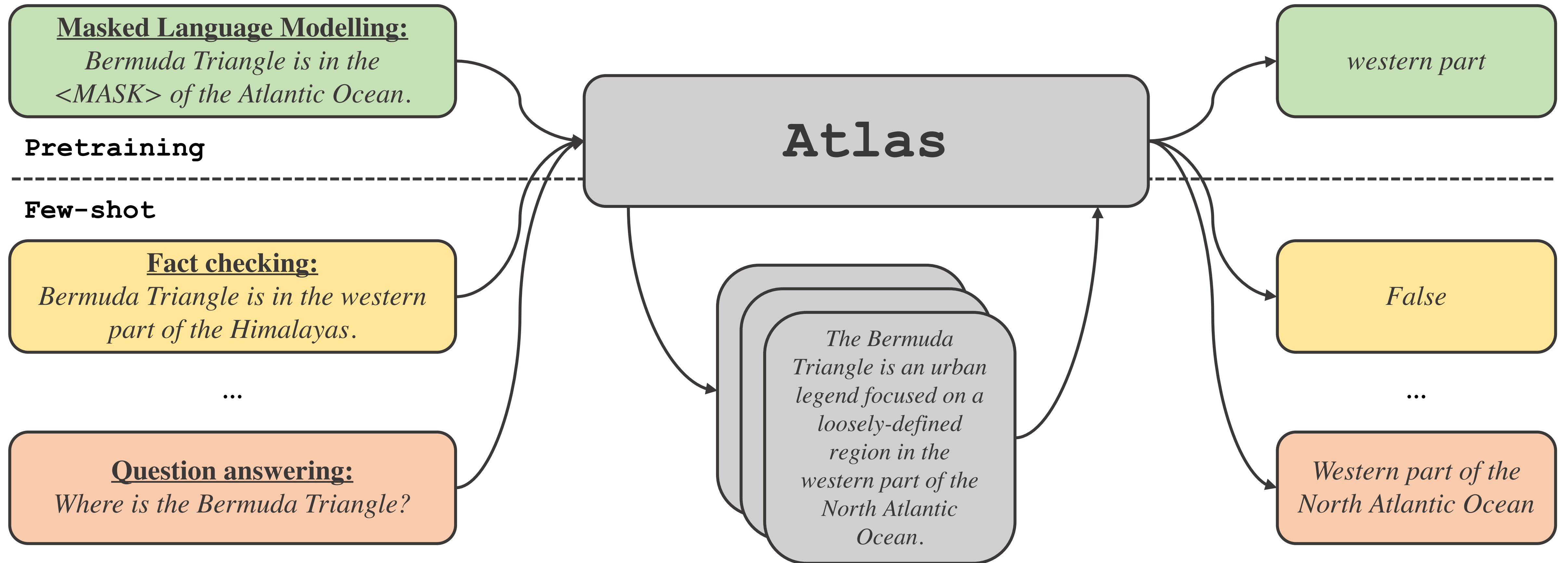
Model	NQ		TriviaQA filtered		TriviaQA unfiltered	
	64-shot	Full	64-shot	Full	64-shot	Full
GPT-3 (Brown et al., 2020)	29.9	-	-	-	71.2	-
Gopher (Rae et al., 2021)	28.2	-	57.2	-	61.3	-
Chinchilla (Hoffmann et al., 2022)	35.5	-	64.6	-	72.3	-
PaLM (Chowdhery et al., 2022)	39.6	-	-	-	81.4	-
RETRO (Borgeaud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

[Izacard et al., 2022]

ATLAS — A Retrieval-Augmented LM



ATLAS — A Retrieval-Augmented LM



Retrieval-Augmented LMs

What to retrieve?

Query



Text chunks (passages)?

Tokens?

Something else?

Retrieval-Augmented LMs

What to retrieve?

Query

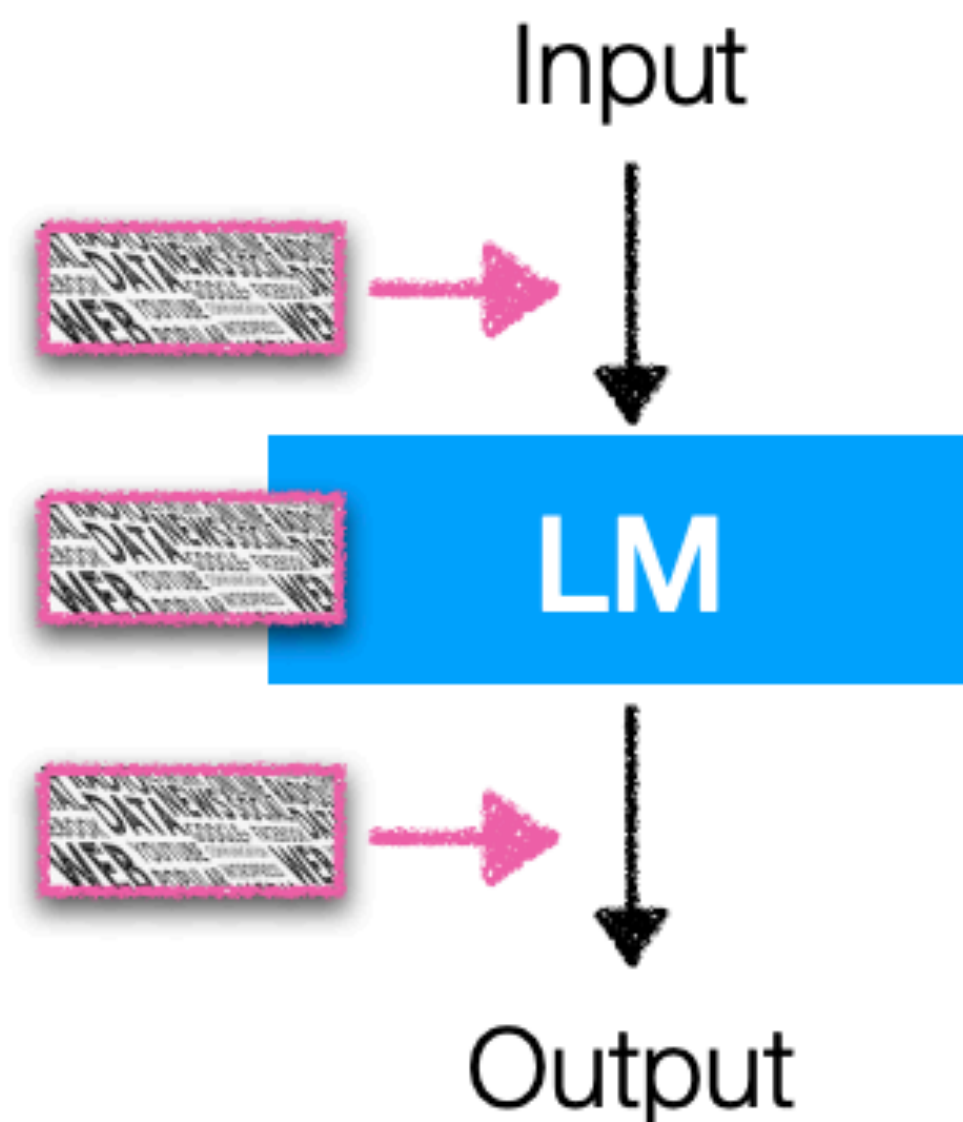


Text chunks (passages)?

Tokens?

Something else?

How to use retrieval?



Retrieval-Augmented LMs

What to retrieve?

Query

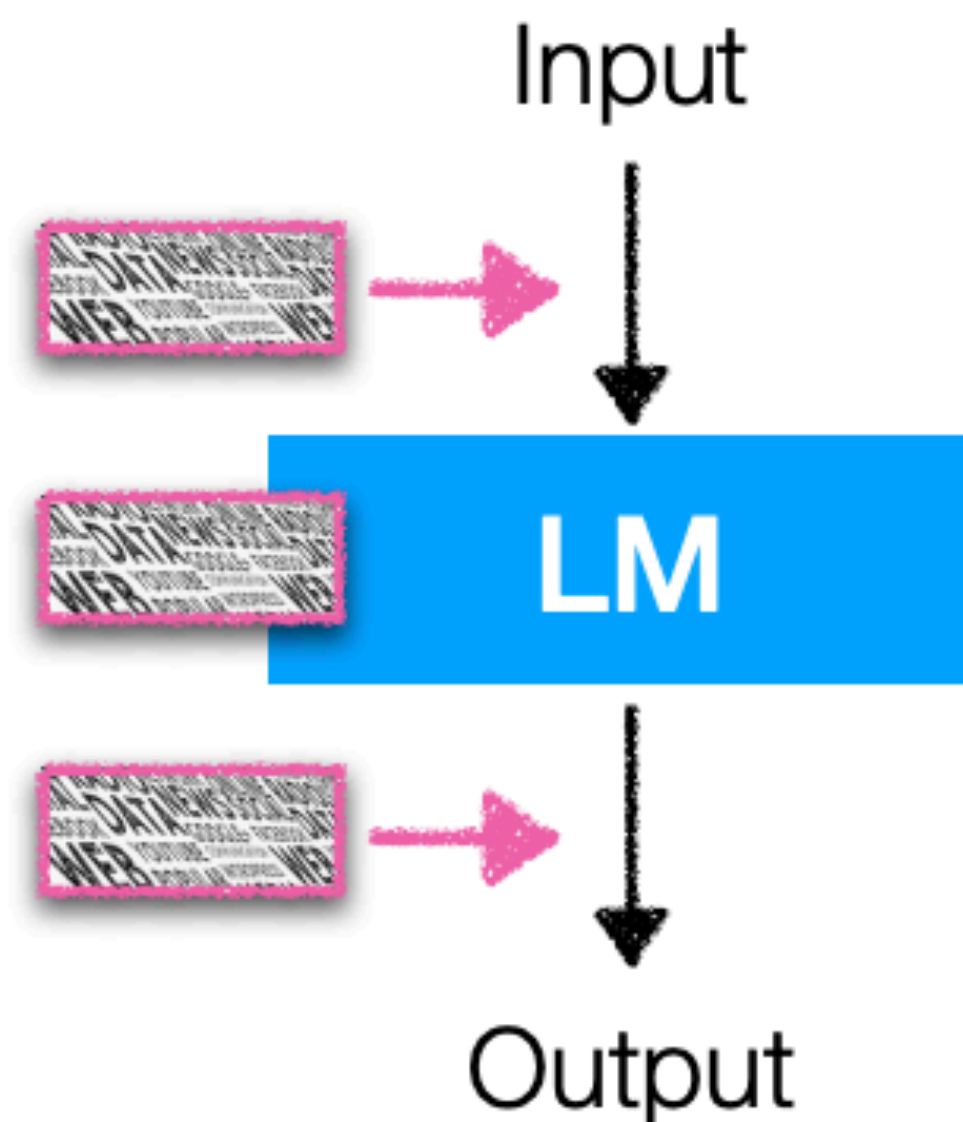


Text chunks (passages)?

Tokens?

Something else?

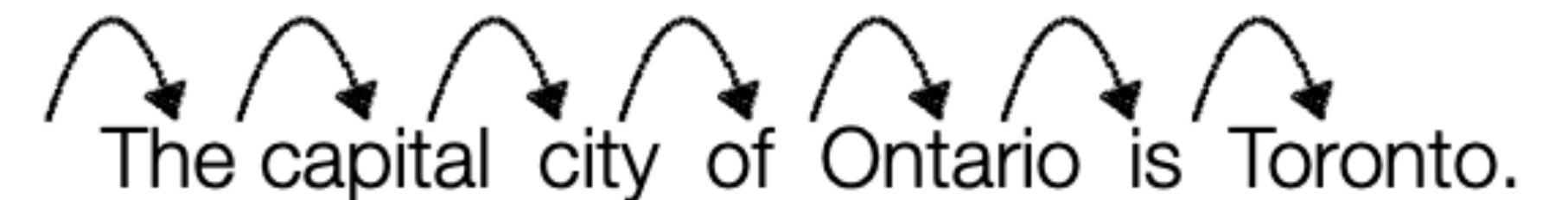
How to use retrieval?



When to retrieve?

w/ retrieval

The capital city of Ontario is Toronto.

A diagram showing the sentence 'The capital city of Ontario is Toronto.' with curved arrows connecting the words: 'The' to 'capital', 'capital' to 'city', 'city' to 'of', 'of' to 'Ontario', 'Ontario' to 'is', and 'is' to 'Toronto'. This illustrates the sequential nature of the text and how retrieval might be triggered at different points.

Retrieval-Augmented LMs

What to retrieve?

Query



Text chunks (passages)?

Tokens?

Something else?

How to use retrieval?

Input



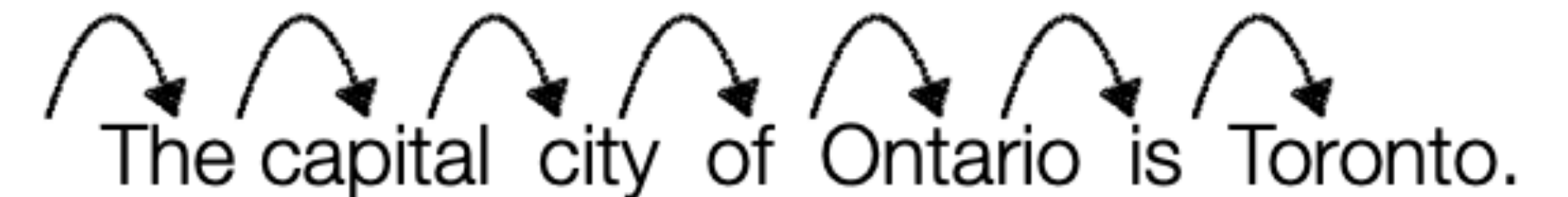
LM



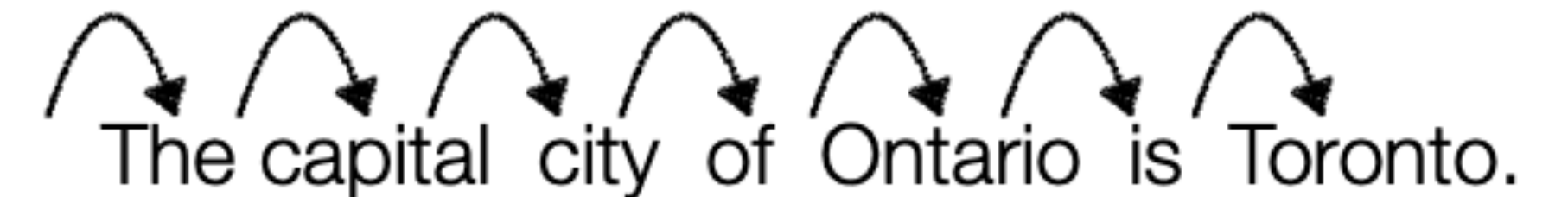
Output

When to retrieve?

w/ retrieval

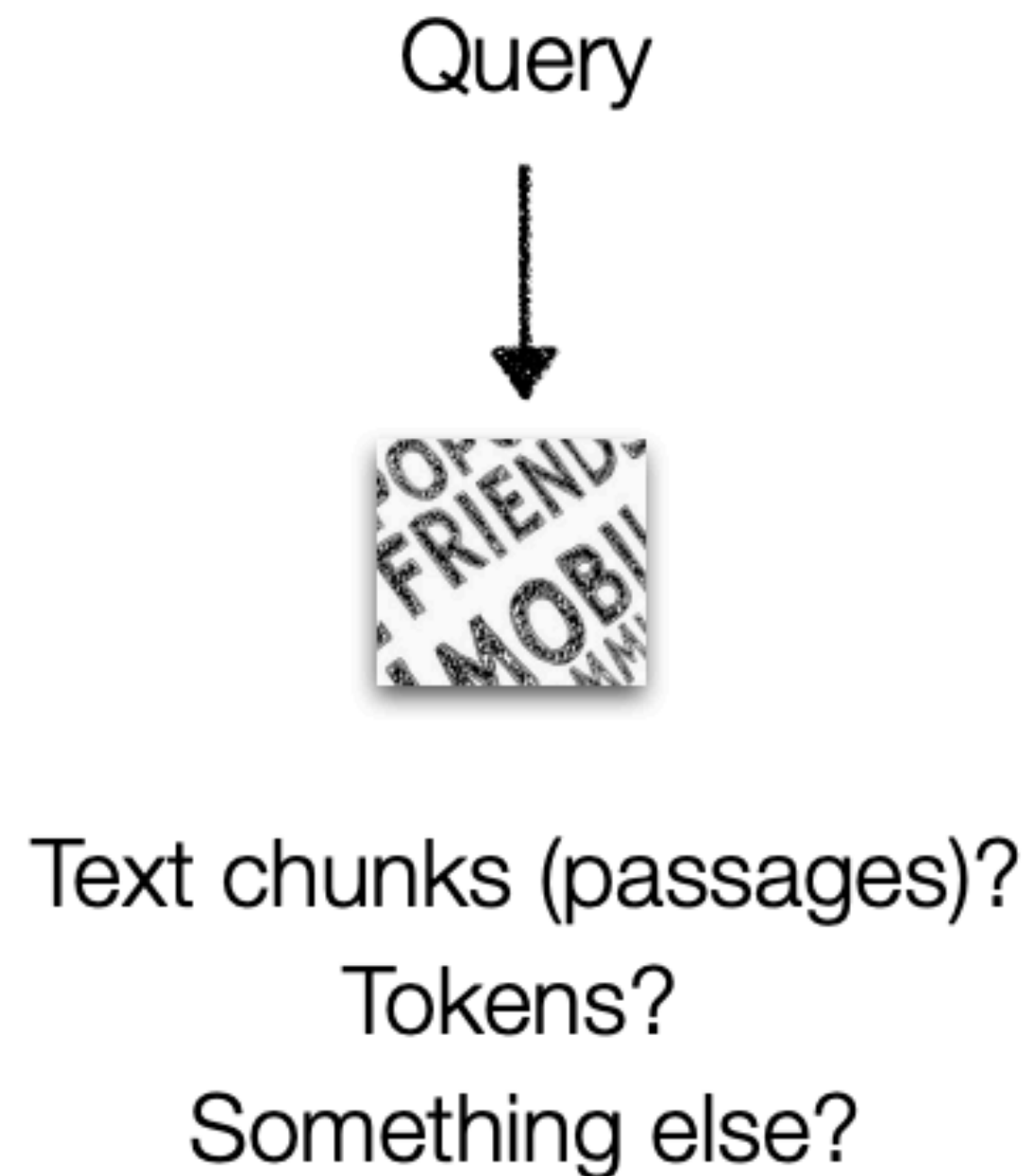


w/ retrieval w/ r w/r w/r w/ r w/r w/r

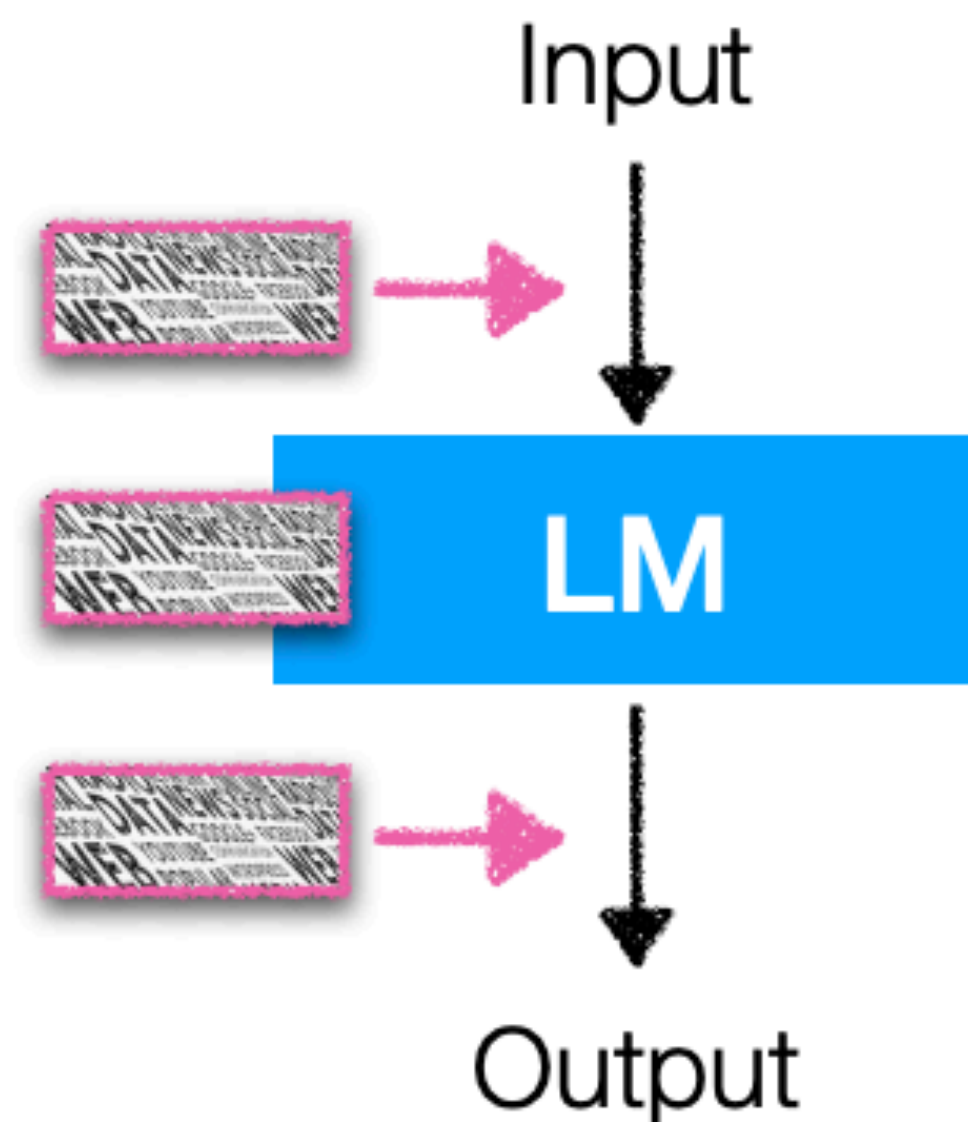


Retrieval-Augmented LMs

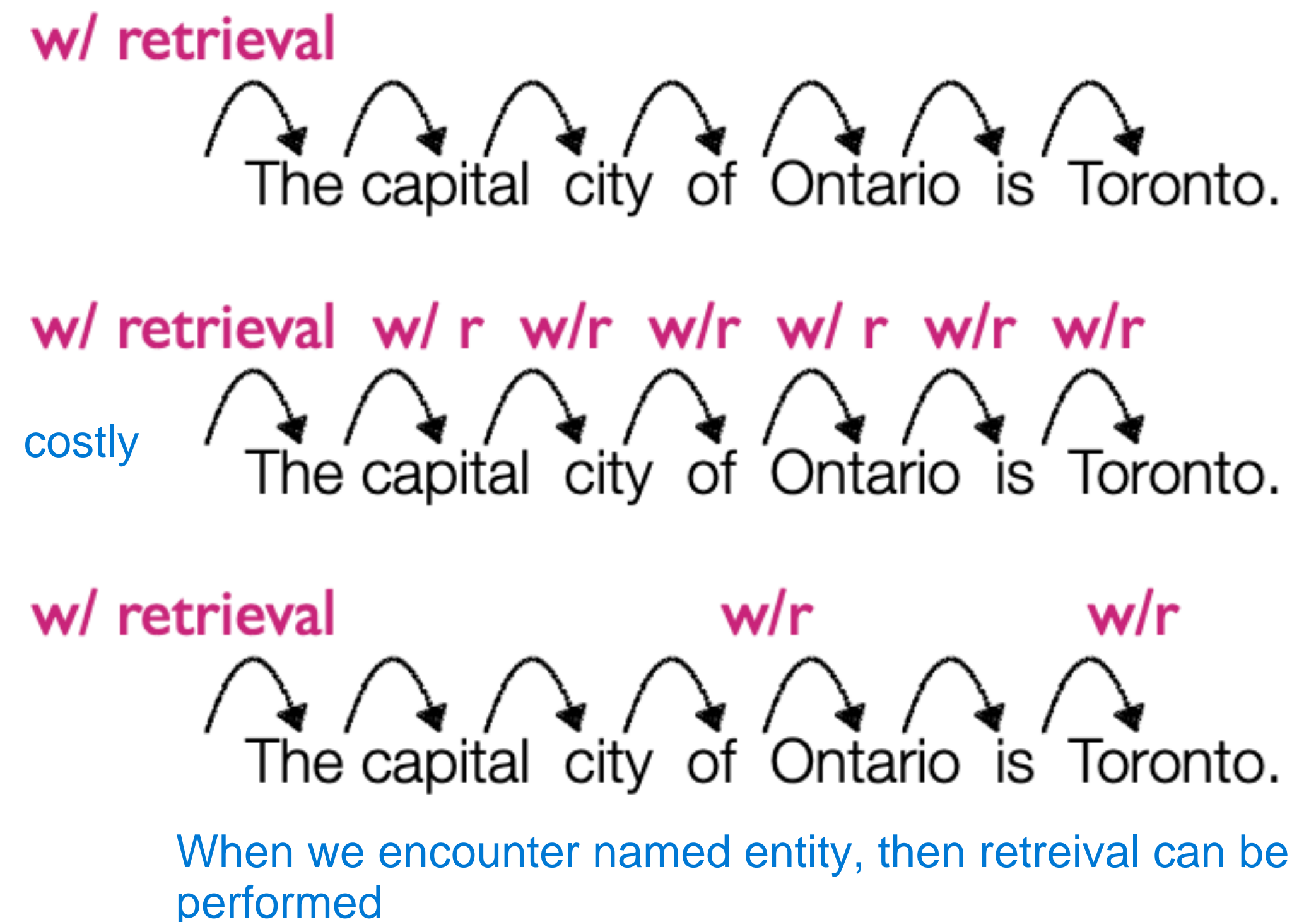
What to retrieve?



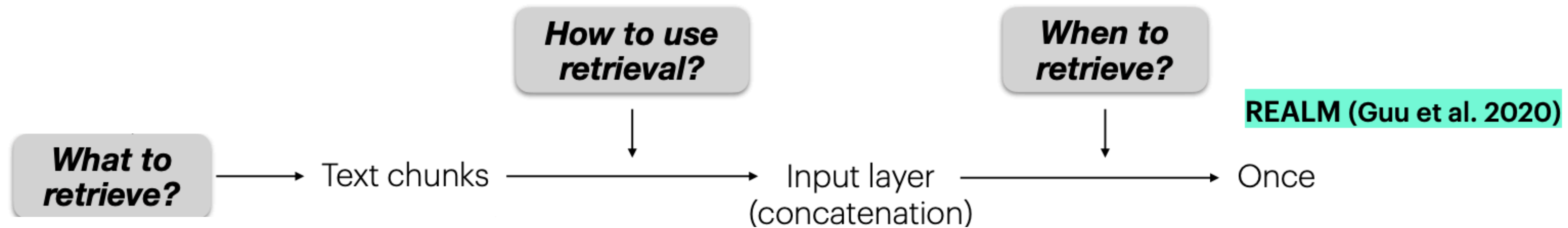
How to use retrieval?



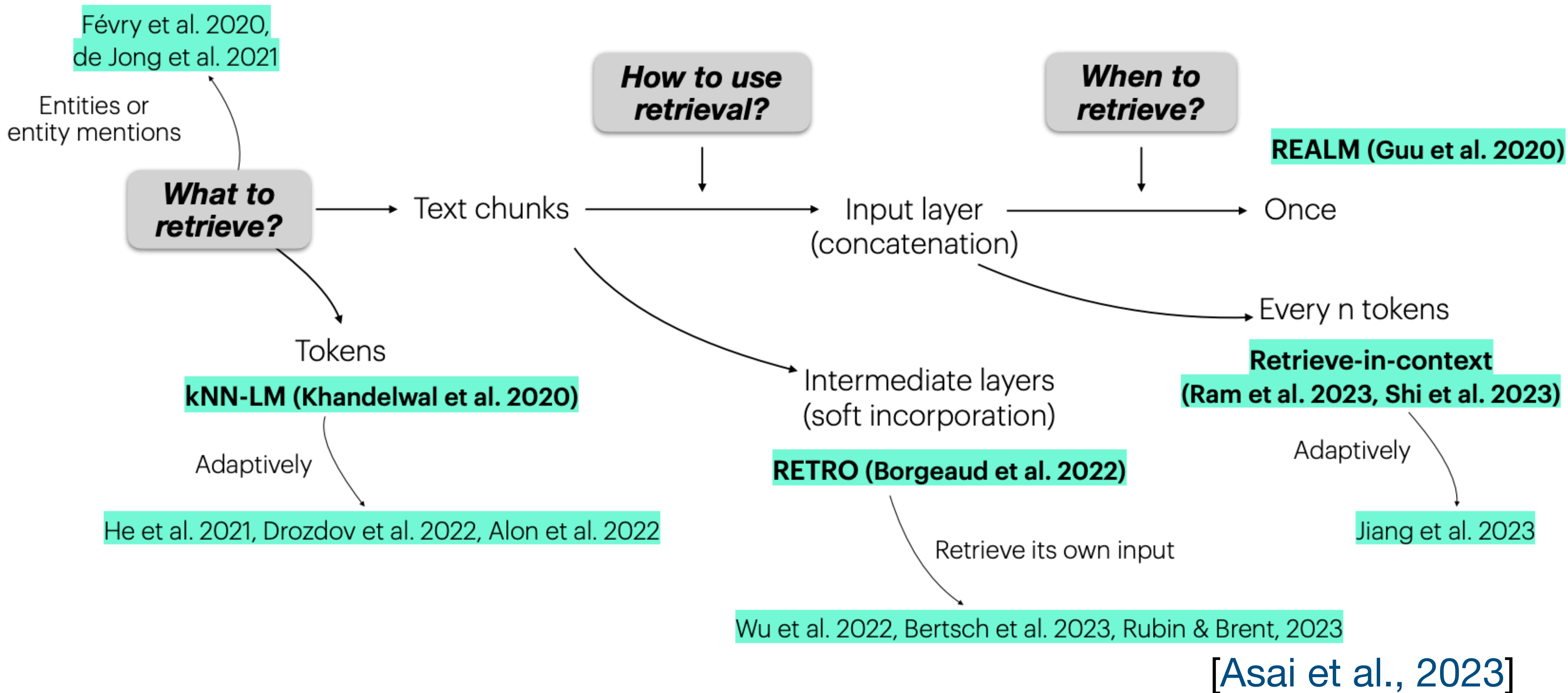
When to retrieve?



Retrieval-Augmented LMs



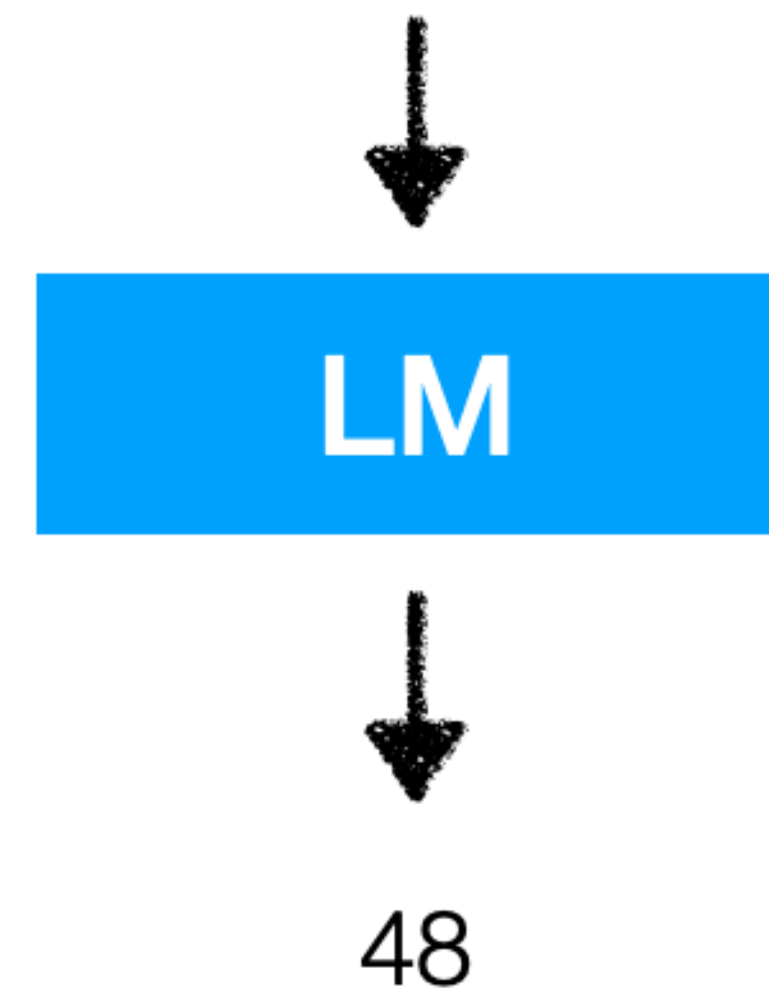
Retrieval-Augmented LMs



REALM [Guu et al., 2020]

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

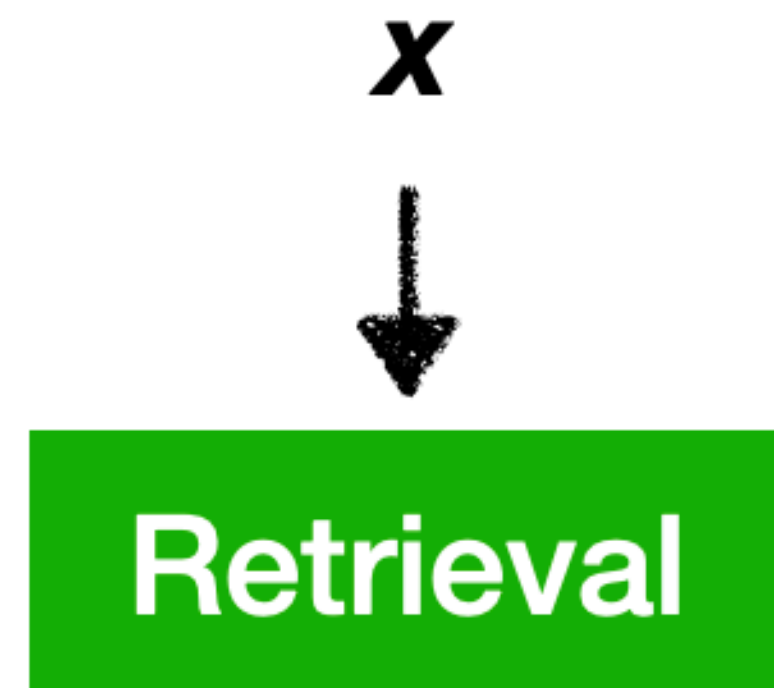
World Cup 2022 was ... the increase to [MASK] in 2026.



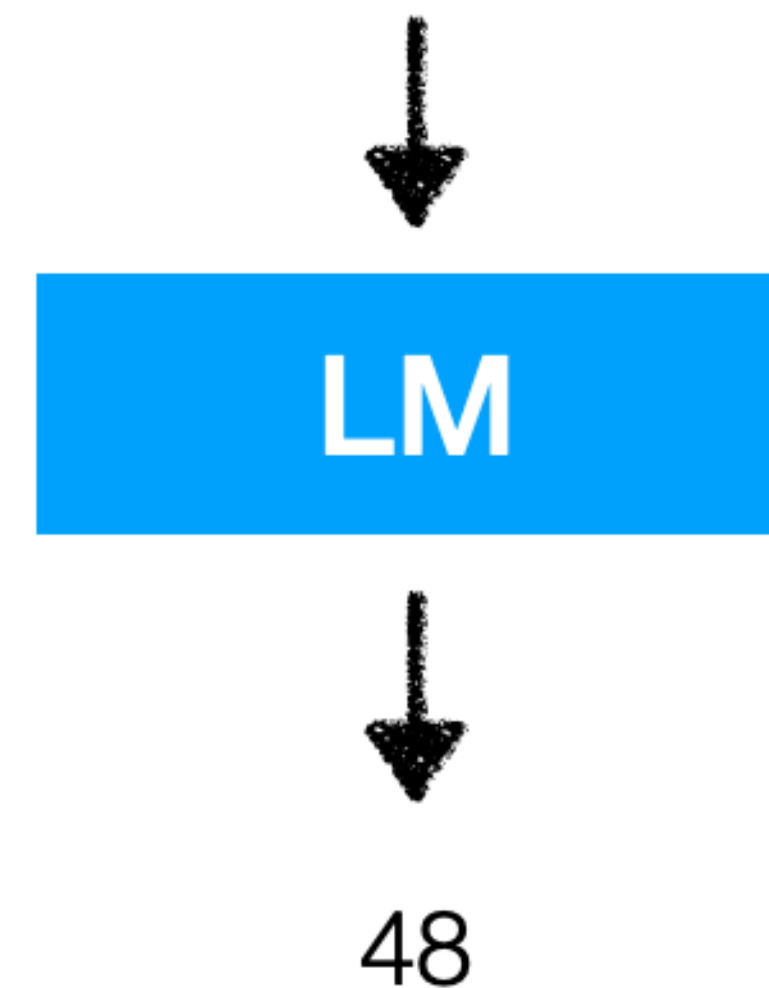
Read stage

REALM [Guu et al., 2020]

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



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



Read stage

REALM [Guu et al., 2020]

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

x



Retrieval



FIFA World Cup 2026
will expand to 48 teams.

Retrieve stage

k chunks of text
(passages)

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



LM

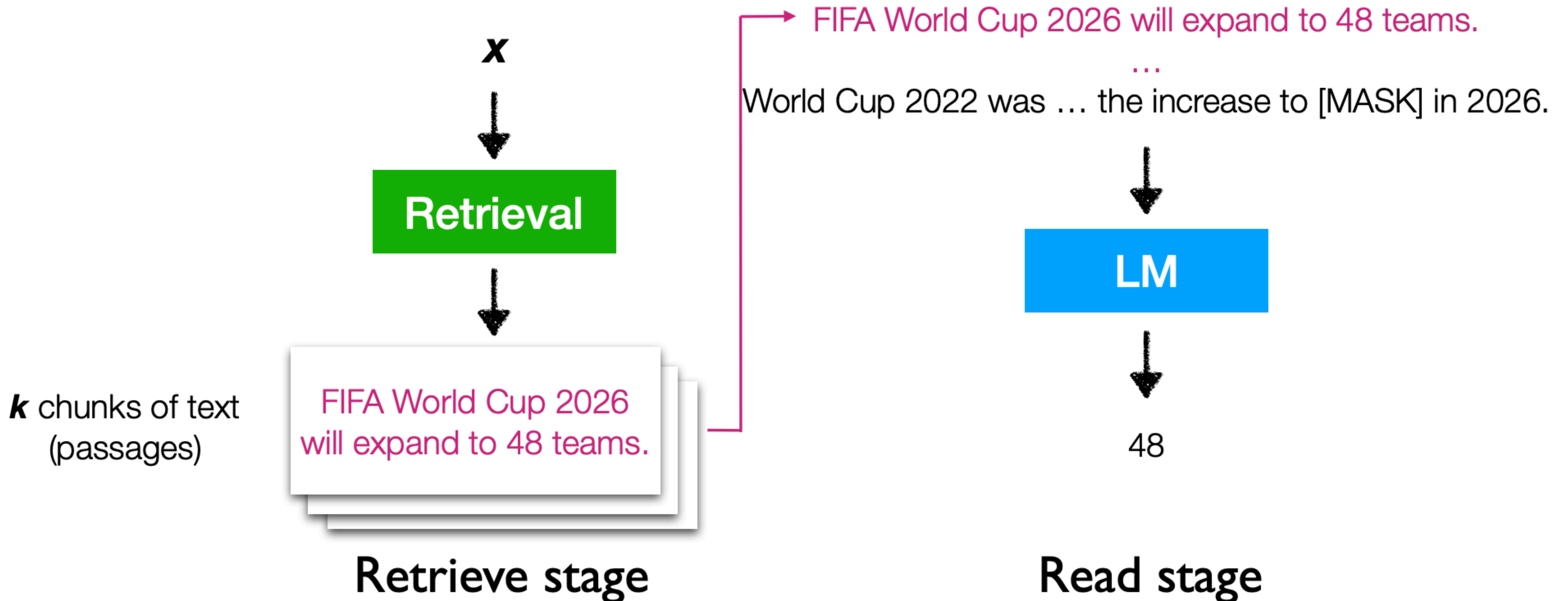


48

Read stage

REALM [Gua et al., 2020]

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



REALM — Retrieval

FIFA World Cup 2026
will expand to 48 teams.

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

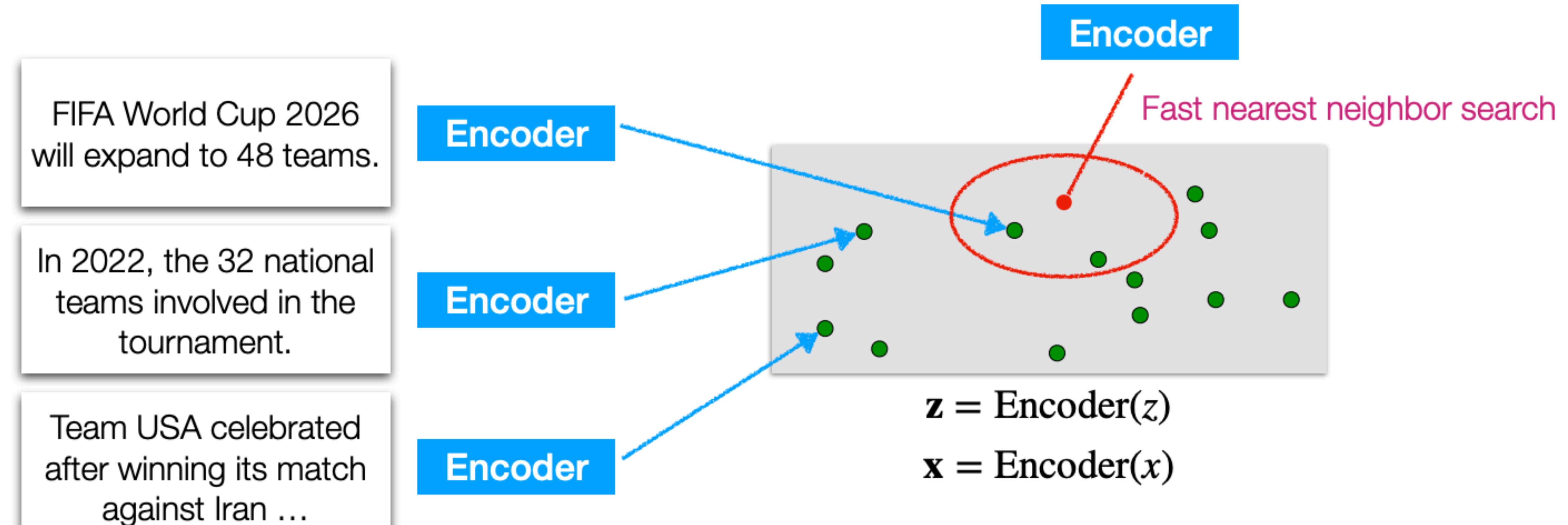
Team USA celebrated
after winning its match
against Iran ...

Wikipedia

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

REALM — Retrieval

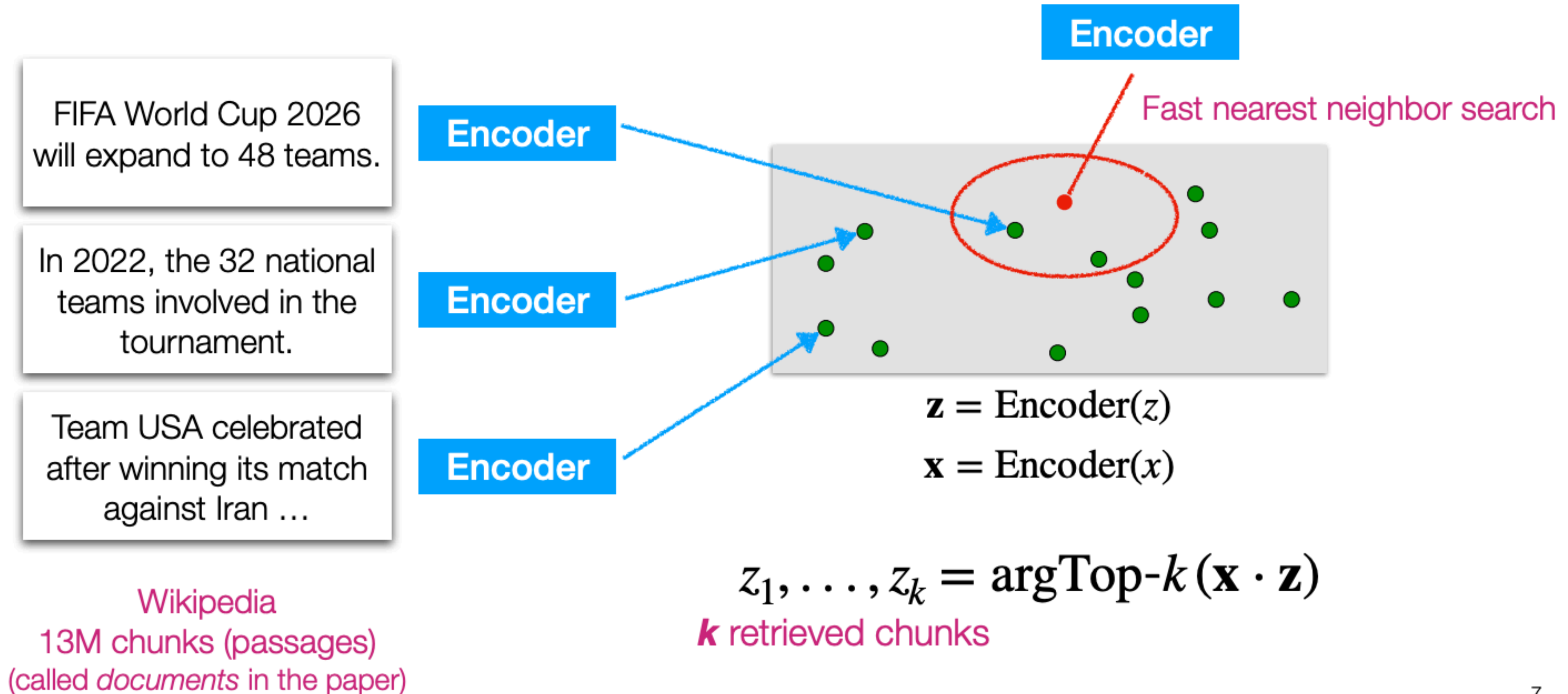
\mathbf{x} = World Cup 2022 was ... the increase to [MASK] in 2026.



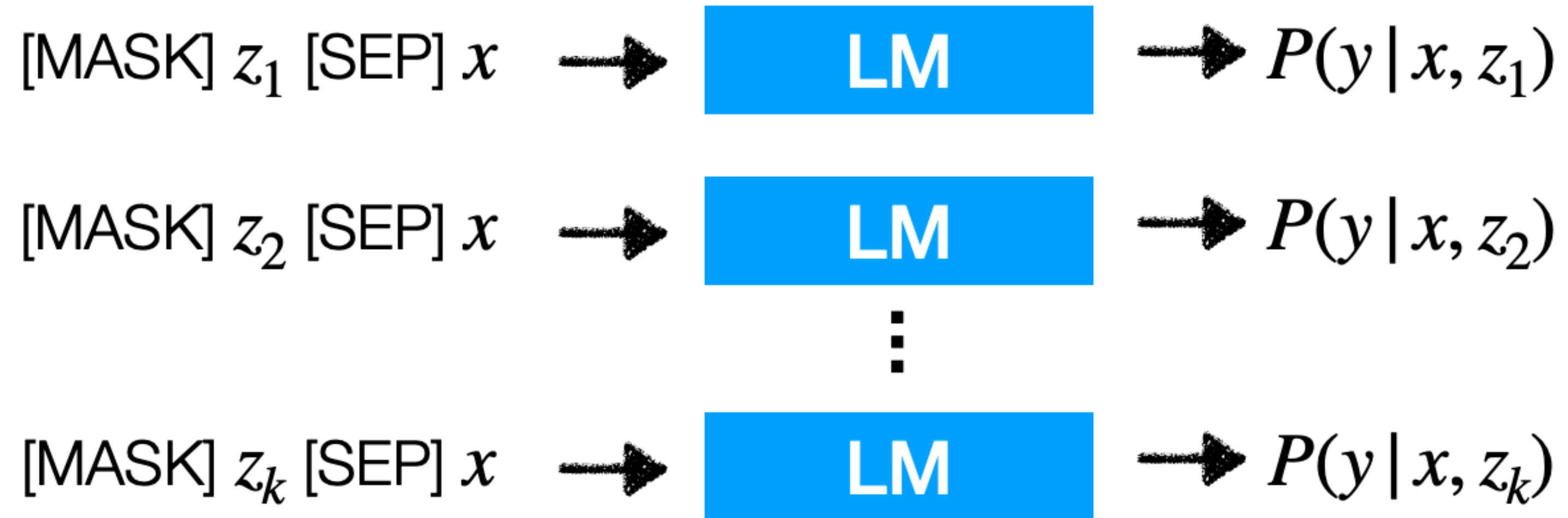
Wikipedia
13M chunks (passages)
(called *documents* in the paper)

REALM — Retrieval

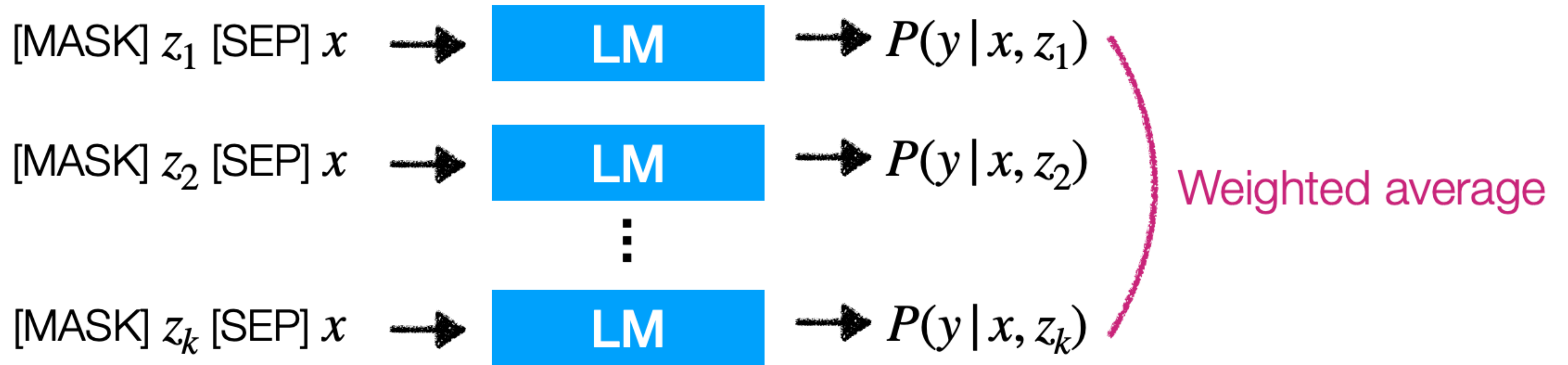
\mathbf{x} = World Cup 2022 was ... the increase to [MASK] in 2026.



REALM — Reading



REALM — Reading



Need to approximate
 \rightarrow Consider top k chunks only

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

0 if not one of top k

REALM [Guu et al., 2020]

What to retrieve?

- **Chunks** ✓
- Tokens
- Others

REALM [Guu et al., 2020]

What to retrieve?

- **Chunks** ✓
- Tokens
- Others

How to use retrieval?

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

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

Overview — Retrieval-Augmented Models

REALM [Guu et al., 2020] — Masked Language Modeling (MLM) pre-training objective followed by fine-tuning, focusing on ODQA

DPR [Karpukhin et al., 2020] — pipeline training rather than joint training, focusing on ODQA with no explicit LM training objective

RAG [Lewis et al., 2020] — Generative training objective rather than MLM, focusing on ODQA and knowledge-intensive tasks (no explicit LM objective)

ATLAS [Izacard et al., 2022] — Combine RAG with a retrieval-based LM pre-training objective and an encoder-decoder architecture, focusing on ODQA and knowledge-intensive tasks

Reading List

Reliable, Adaptable, and Attributable Language Models with Retrieval,

<https://arxiv.org/abs/2403.03187>

ATLAS: Few-shot Learning with Retrieval Augmented Language Models,

<https://arxiv.org/abs/2208.03299>

REALM: Retrieval-Augmented Language Model Pre-Training,

<https://arxiv.org/abs/2002.08909>

Reading Wikipedia to Answer Open-Domain Questions,

<https://arxiv.org/abs/1704.00051>

Question and Answer Test-Train Overlap in Open-Domain Question Answering Datasets, **<https://arxiv.org/abs/2008.02637>**

Challenges in Generalisation in Open Domain Question Answering,

<https://arxiv.org/abs/2109.01156>