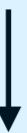


Part V

# Dense Retriever and End-to-end Training

# Key questions

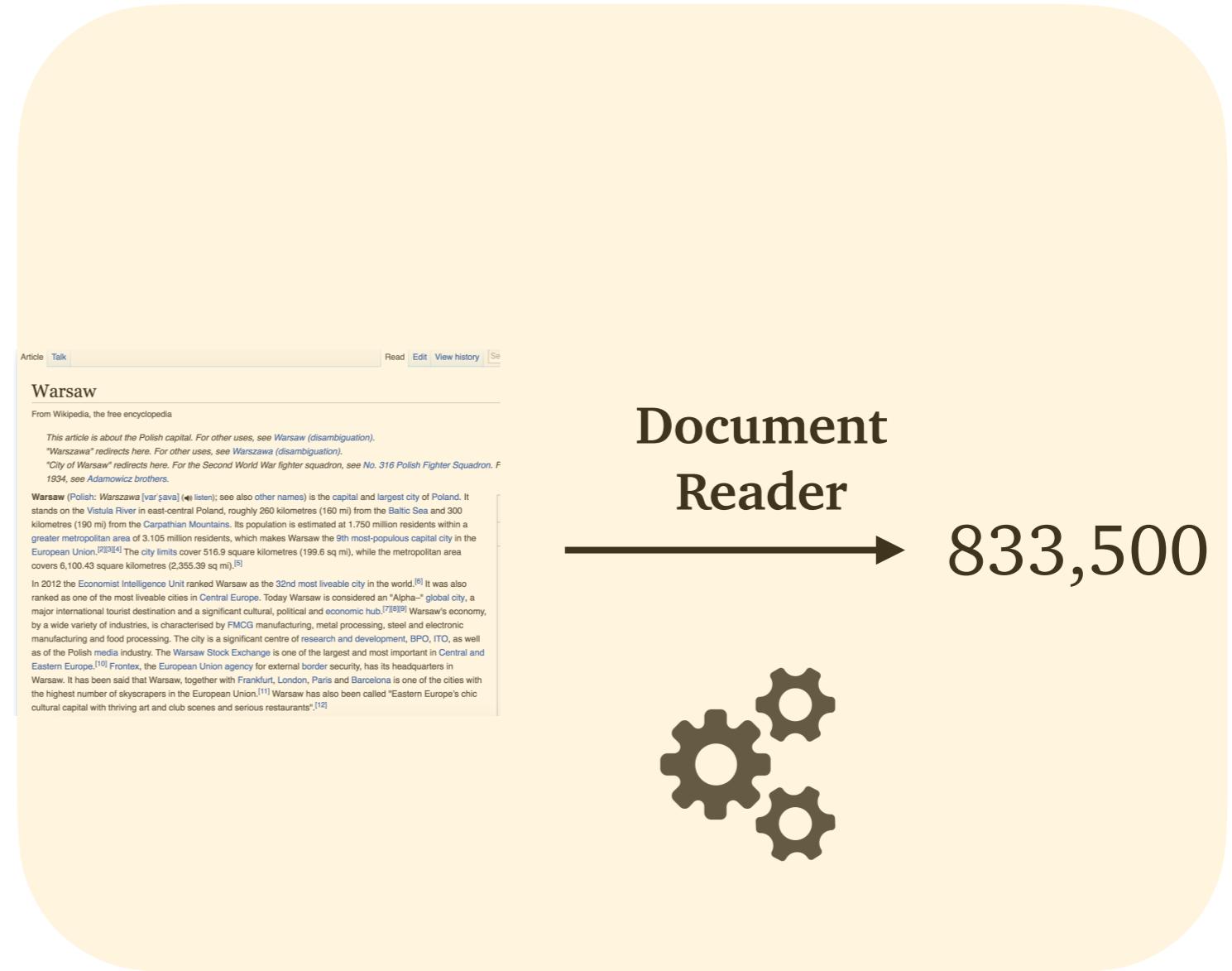
Q: How many of Warsaw's inhabitants spoke Polish in 1933?



WIKIPEDIA



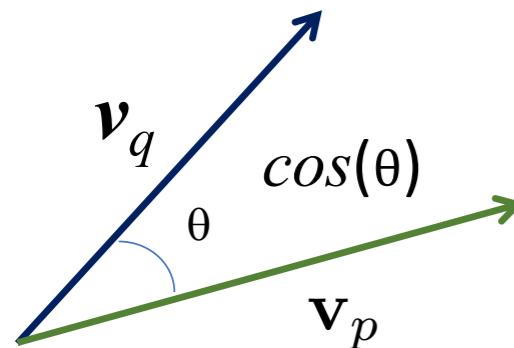
Document  
Retriever



Can we also train the retriever component?

Can we use dense representations for retrieval?

# Sparse vs dense representations for retrieval



$$d_1 \gg d_2$$

sparse repr:  $[0\dots 1 \dots 1 \dots 0.1] \in \mathbb{R}^{d_1}$

dense repr:  $[1.03, -5.72, 6.42, .., 9.91] \in \mathbb{R}^{d_2}$



sparse

“How many provinces did the Ottoman empire contain in the 17th century?”

“What part of the atom did Chadwick discover?”



dense

“Who is the **bad guy** in lord of the rings?”

*Sala Baker is an actor and stuntman from New Zealand. He is best known for portraying the **villain** Sauron in the Lord of the Rings trilogy by Peter Jackson.*

They capture complementary information; Dense representations have never been shown to outperform sparse representations in open-domain QA before 2019!

# Why dense retrieval now?

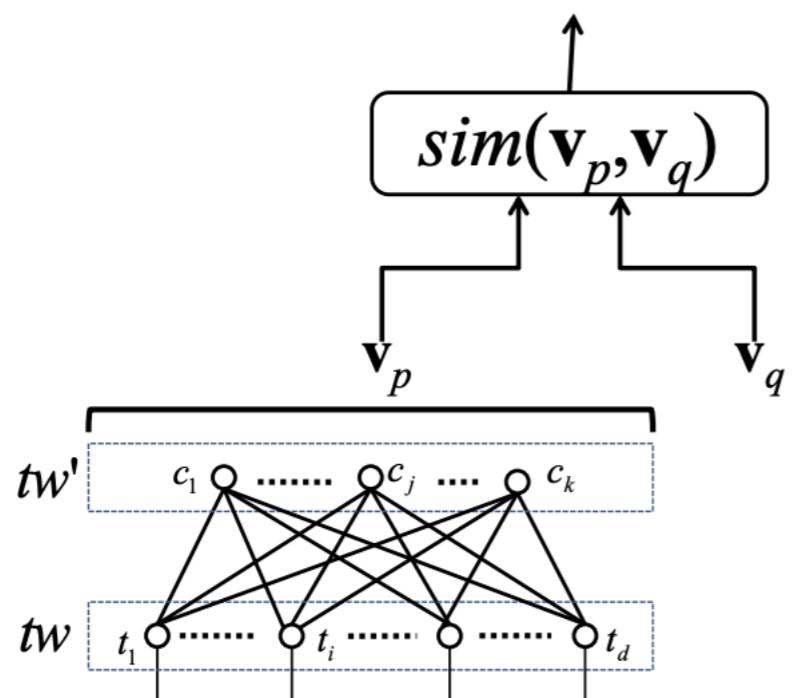
It is a difficult problem: we need to *encode*, *index* and *search* from **5M** documents or **30M** paragraphs or **60B** phrases.

## Learning Discriminative Projections for Text Similarity Measures

Wen-tau Yih    Kristina Toutanova    John C. Platt    Christopher Meek

Microsoft Research  
One Microsoft Way  
Redmond, WA 98052, USA

CoNLL'2011  
best paper



- Cross-lingual document retrieval
- Ad relevance prediction
- Web search ranking

# Why dense retrieval now?

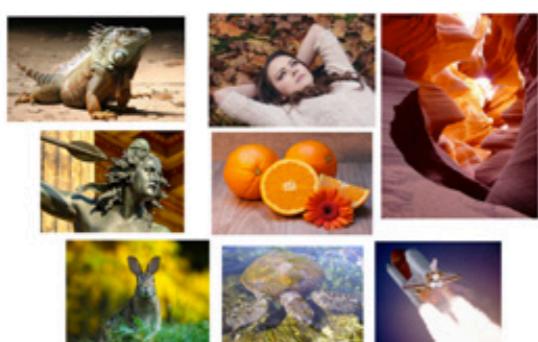
- It is actually not easy to make these dense models “work”.
  - Needs large enough labeled data (e.g., 82M query-doc pairs from user clicks).



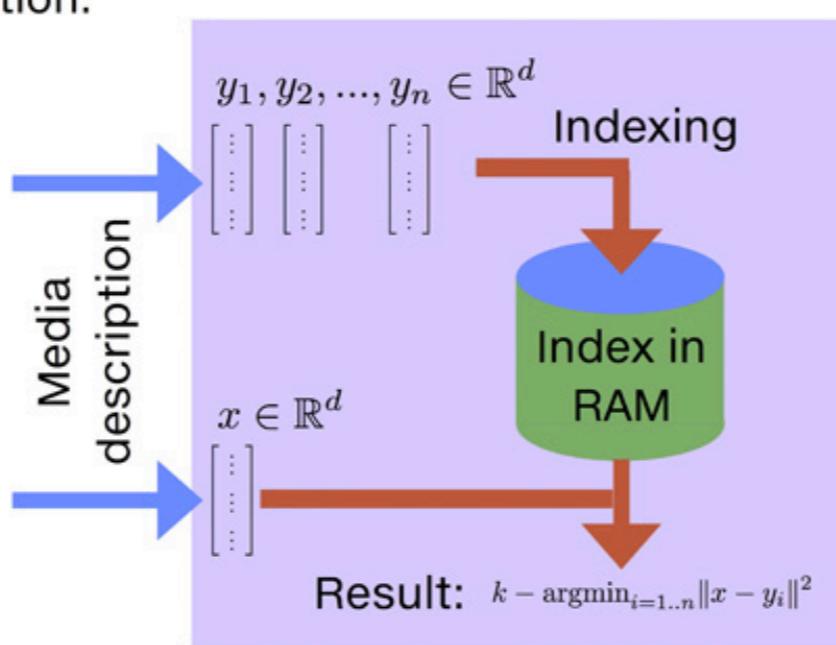
We have pre-trained models now!

- Now we have much better techniques and tools to support fast maximum inner product search (MIPS):
  - In-memory data structure and indexing schemes

Build index for a collection:



Query:



e.g., FAISS

[Johnson et al., 2017]

# ORQA: Open-Retriever Question Answering

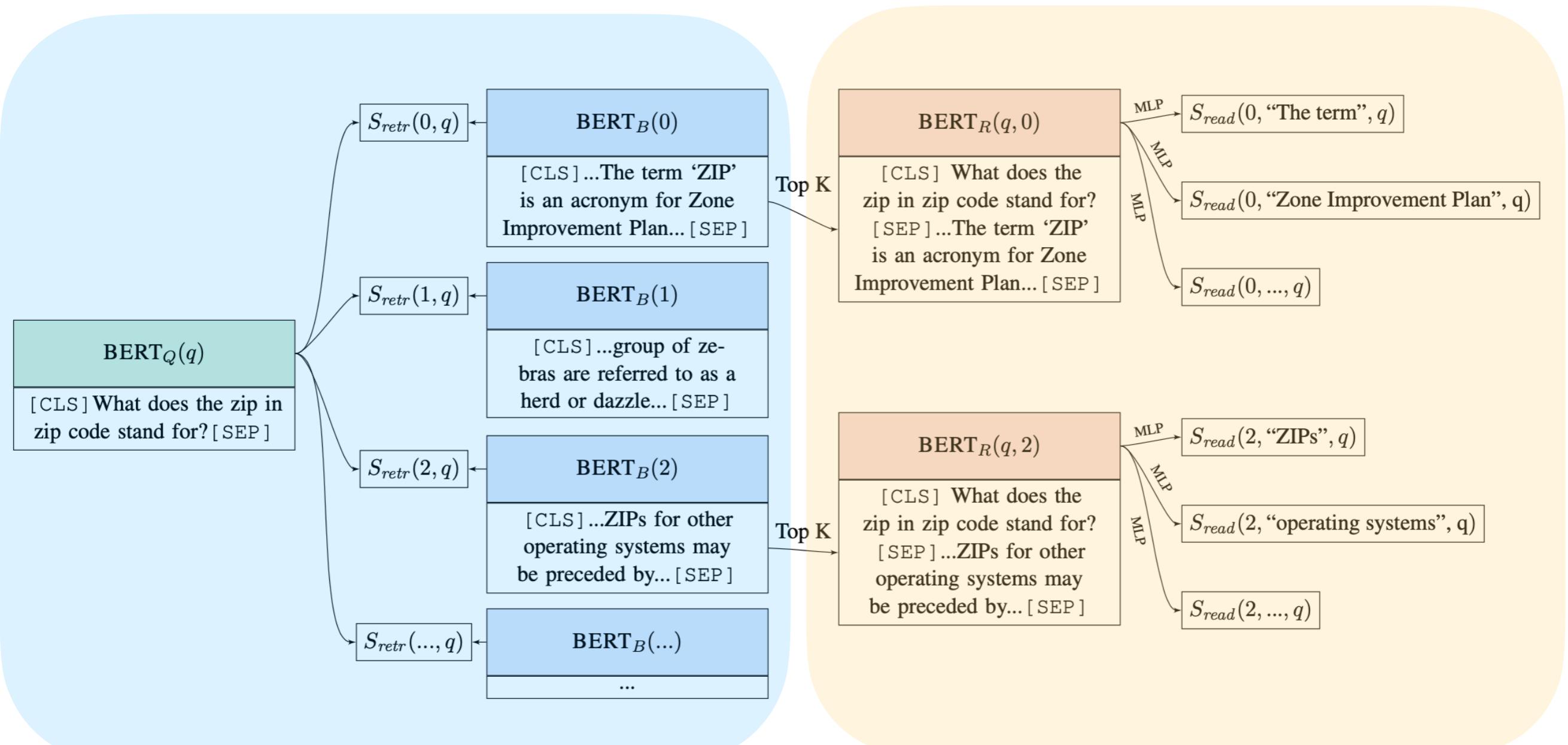
[Lee et al., 2019]

*First model to learn retriever and reader jointly*

## Key contributions:

- Both retriever and reader are learnable with NNs (= BERT)
- Only learned from question-answer pairs; No reading comprehension datasets!  $\mathcal{D}_{\text{QA}} = \{(Q_i, A_i)\}$
- A new pre-training task called **Inverse Cloze Task (ICT)** to address the challenging retrieval problem.

# ORQA: Open-Retriever Question Answering



Information Retrieval

Reading Comprehension

# ORQA: Overview

**Notations:** b - text block, s - a span of text within b, q - question

$$S(b, s, q)$$

Fixed-length blocks as “passages”: 288 wordpieces  $\Rightarrow$  13M in total

Each block has 2000 possible answer spans

**Modeling**

$$S(b, s, q) = S_{retr}(b, q) + S_{read}(b, s, q)$$

**Inference**

$$a^* = \text{TEXT}(\underset{b,s}{\operatorname{argmax}} S(b, s, q))$$

# Retriever score: $S_{retr}(b, q)$

$$h_q = \mathbf{W}_q \text{BERT}_Q(q)[\text{CLS}]$$

$$h_b = \mathbf{W}_b \text{BERT}_B(b)[\text{CLS}]$$

$$S_{retr}(b, q) = h_q^\top h_b$$

All of Wikipedia: select top K

Question  $q$

What does the zip  
in zip code stand for?



$\text{BERT}_Q$

$\text{BERT}_B$

oooooooooooo  $W_q$

oooooooooooo  $W_b$



$h_q$

$h_b$

$$S_{retr}(b, q) = h_q^\top h_b$$

Each evidence block  $b$



Evidence Block 1  $S_{retr}(b_1, q)$

The early history and context of postal codes began with postal district/zone numbers. The United States Post Office Department (USPOD) implemented postal zones for many large cities in 1943.<sup>[8]</sup> For example:

Mr. John Smith  
3256 Epiphenomenal Avenue

Minneapolis, MN 55416

The "16" is the number of the postal zone in the specific city.<sup>[citation needed]</sup> By the early 1960s, a more organized system was needed, and non-mandatory five-digit ZIP Codes were introduced nationwide on July 1, 1963. The USPOD issued its *Publication 59: Abbreviations for Use with ZIP Code* on October 1, 1963, with the list of two-letter state abbreviations which were generally written with both letters capitalized.<sup>[4]</sup> An earlier list, published in June 1963, had proposed capitalized abbreviations ranging from two to five letters.<sup>[4]</sup> According to *Publication 59*, the two-letter standard was chosen as a maximum 22-position line. The reason for this has been found to be the most universal addressable line capacity based on the needs of the time, which would be needed by a local city name combined with a two-letter state abbreviation, such as "Sacramento, Calif."<sup>[4]</sup> The abbreviations remained unchanged, with the exception of Nebraska, which was changed from NB to NE in 1969 at the request of the Canadian postal administration, to avoid confusion with the Canadian province of New Brunswick.<sup>[4]</sup>

Robert Moon is considered the father of the ZIP Code; he submitted his proposal in 1944 while working as a postal inspector.<sup>[8]</sup>

<sup>[8]</sup> The post office only credits Moon with the first three digits of the ZIP Code, which describe the sectional center facility (SCF).

precise locale within the SCF, were proposed by Henry Bentley Hahn Sr.<sup>[7]</sup> The SCF sorts mail to all post offices with those first three digits in their ZIP Codes. The mail is sorted according to the final two digits of the ZIP Code and sent to the corresponding post offices in the early morning. Sectional centers do not deliver mail and are not open to the public (although the mailing may include a post office that is open to the public), and most of their employees work the night shift. Items of mail picked up at post offices are sent to their own SCFs in the afternoon, where the mail is sorted overnight. In the case of large cities, the last two digits assigned generally coincide with the older postal zone number.<sup>[7]</sup>

For more information, see ZIP Code.

Mr. John Smith  
3256 Epiphenomenal Avenue

Minneapolis, MN 55416

In 1967, these became mandatory for second- and third-class bulk mailers, and the system was soon adopted generally. The United States Post Office used a cartoon character, which it called Mr. ZIP, to promote the use of the ZIP Code. He was often depicted with a legend such as "USE ZIP CODE" in the selvage or panes of postage stamps or on the covers or booklet panes of stamps.<sup>[citation needed]</sup>

In 1971, Elmira (NY) *Star-Gazette* reporter Dick Baumbach found out the White House was not using a ZIP Code on its envelopes. Herb Klein, special assistant to President Nixon, responded by saying the next printing of envelopes would include the ZIP Code.<sup>[8]</sup>

<sup>[8]</sup> ZIP+4 In 1983, the USPS introduced a ZIP+4 system that called ZIP+4, often "plus-four" or "add-on" code, or ZIP+4 code for short. ZIP+4 adds four additional digits to identify a geographic segment within the five-digit delivery area, such as a city block, a group of apartment buildings, a high-volume delivery point, a mail, a post office box, or any other unit that could use an extra identifier to aid in efficient mail sorting and delivery. However, initial attempts to promote universal use of the new format met with public resistance, and today the plus-four code is not required.<sup>[10]</sup> In general, mail is read by a multiline optical character reader (MOCR) that almost instantly determines the correct ZIP+4 Code from the address—along with the even more specific delivery point—and sprays an Intelligent Mail barcode (IM) on

For Post Office Boxes, the general (but not invariable) rule is that each box has its own ZIP+4 code. The add-on code is often one of the following: the last four digits of the box number (e.g., PO Box 107050, Albany, NY 12201-7050), zero plus the last three digits of the box number (e.g., PO Box 17727, Eagle River, AK 99577-0727), or, if the box number consists of fewer than four digits, enough zeros are attached to the front of the box number to produce a four-digit number (e.g., PO Box 77, Juneau, AK 99750-0077). However, there is no uniform rule, so the ZIP+4 Code must be looked up individually for each box.<sup>[citation needed]</sup>

(e.g., using the USPS's official ZIP Code Look-up tool, and being sure to enter just city and state, not the full ZIP<sup>[10]</sup>.)

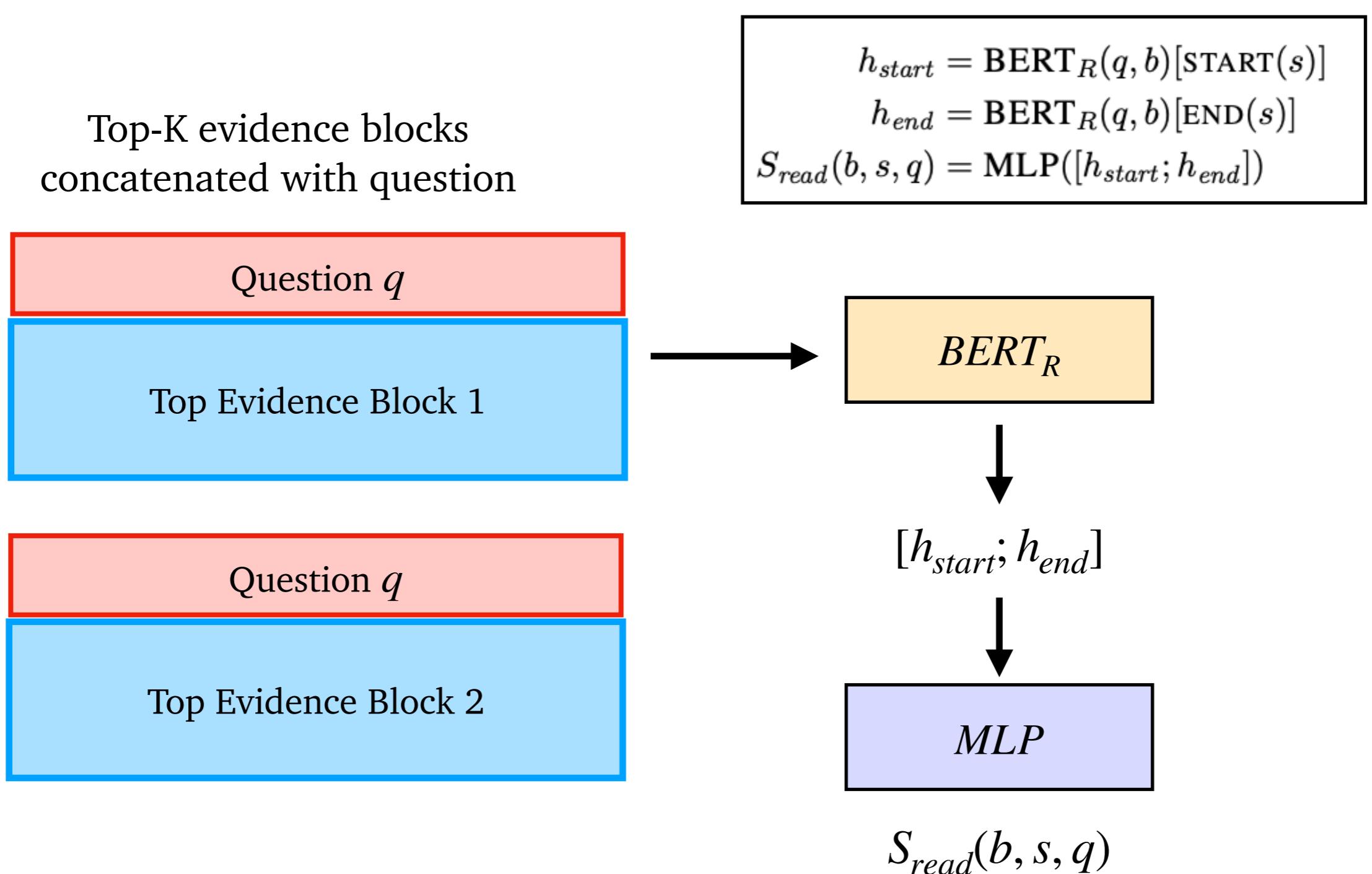
Postage and delivery costs are calculated into the Intelligent Mail barcode. It is printed on the envelope so it can be read by automated machines to sort. A barcode can be printed by the sender (some word-processing programs such as WordPad<sup>[edit]</sup>) inside the feature, but this is not recommended, as the address-to-ZIP lookup tables can be significantly out of date. It is better to let the post office put one on when it processes the piece.<sup>[citation needed]</sup> In general, the post office uses OCR technology, though in some cases a human might have to read and enter the address.<sup>[citation needed]</sup>

use mail. This requires more than just a simple 10H; mailing lists must be standardized with up-to-date Coding Accuracy Support System (CASS)-certified software that adds and verifies a full, correct ZIP+4 Code and an additional two digits representing the exact delivery point.<sup>[citation needed]</sup> Furthermore, mail must be sorted in a specific manner to an 11-digit code with at least 150 mailpieces for each qualifying ZIP Code and must be accompanied by documentation confirming this. These steps are usually done with PAVE-certified software that also prints the barcoded address labels and the barcoded sack or tray tags.<sup>[citation needed]</sup>

The assignment of delivery-point digits (the 10th and 11th digits) is intended to ensure that every single mailable point in the country has its own 12-digit number. The delivery-point digits are calculated based on the primary origin boundary number of the address. The USPS publishes the rules for calculating the delivery-point in a document called the *ASG Technical Guide*.<sup>[12]</sup> However, for two addresses in the same ZIP+4 code, the 10th digit will always be the same (2-0, 4-0, 6-0, 8-0, 0-0). The 11th digit is calculated by dividing the 10th digit by 10 (i.e., the remainder after dividing by 10) and then subtracting that from 10. (Thus, the check digit for 10001-0001 00 would be 7, since 1+1+1=3, 3=3(mod 10) and 10-3=7.)<sup>[citation needed]</sup>

<sup>[12]</sup> Structure and allocation<sup>[edit]</sup>  
Scope and international mail<sup>[edit]</sup>  
ZIP Codes designate delivery points within the United States (and its territories). There are generally no ZIP Codes for

# Reader score: $S_{read}(b, s, q)$



# How is this model learned?

$$P(b, s|q) = \frac{\exp(S(b, s, q))}{\sum_{b' \in \text{TOP}(k)} \sum_{s' \in b'} \exp(S(b', s', q))}$$

$k=5$ ,  $\text{TOP}(k)$  = the top  $k$  retrieved blocks according to  $S_{\text{retr}}(b, q)$

How to update  $\text{BERT}_B$  for all the blocks??

How if top 5 (out of 13M)  
blocks don't contain the  
answer at all?

## Loss function

$$L_{\text{full}}(q, a) = -\log \sum_{b \in \text{TOP}(k)} \sum_{s \in b, a = \text{TEXT}(s)} P(b, s | q)$$

# How is this model learned?

**Early learning: consider a larger set of evidence blocks**

$c = 5000$

$$P_{\text{early}}(b|q) = \frac{\exp(S_{\text{retr}}(b, q))}{\sum_{b' \in \text{TOP}(c)} \exp(S_{\text{retr}}(b', q))}$$
$$L_{\text{early}}(q, a) = -\log \sum_{b \in \text{TOP}(c), a \in \text{TEXT}(b)} P_{\text{early}}(b|q)$$

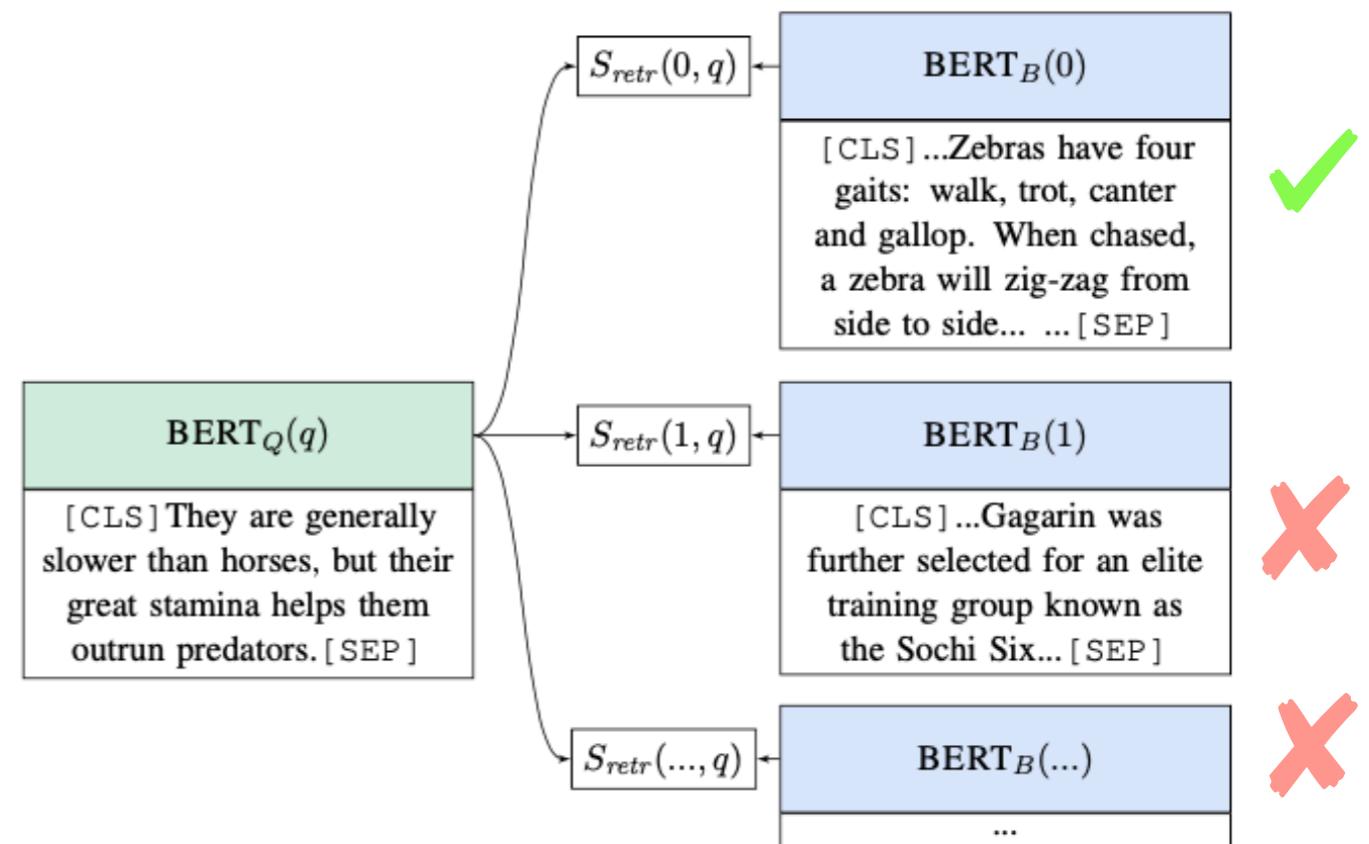
**Final loss**

$$L(q, a) = L_{\text{early}}(q, a) + L_{\text{full}}(q, a)$$

# Pre-training: Inverse Cloze Task (ICT)

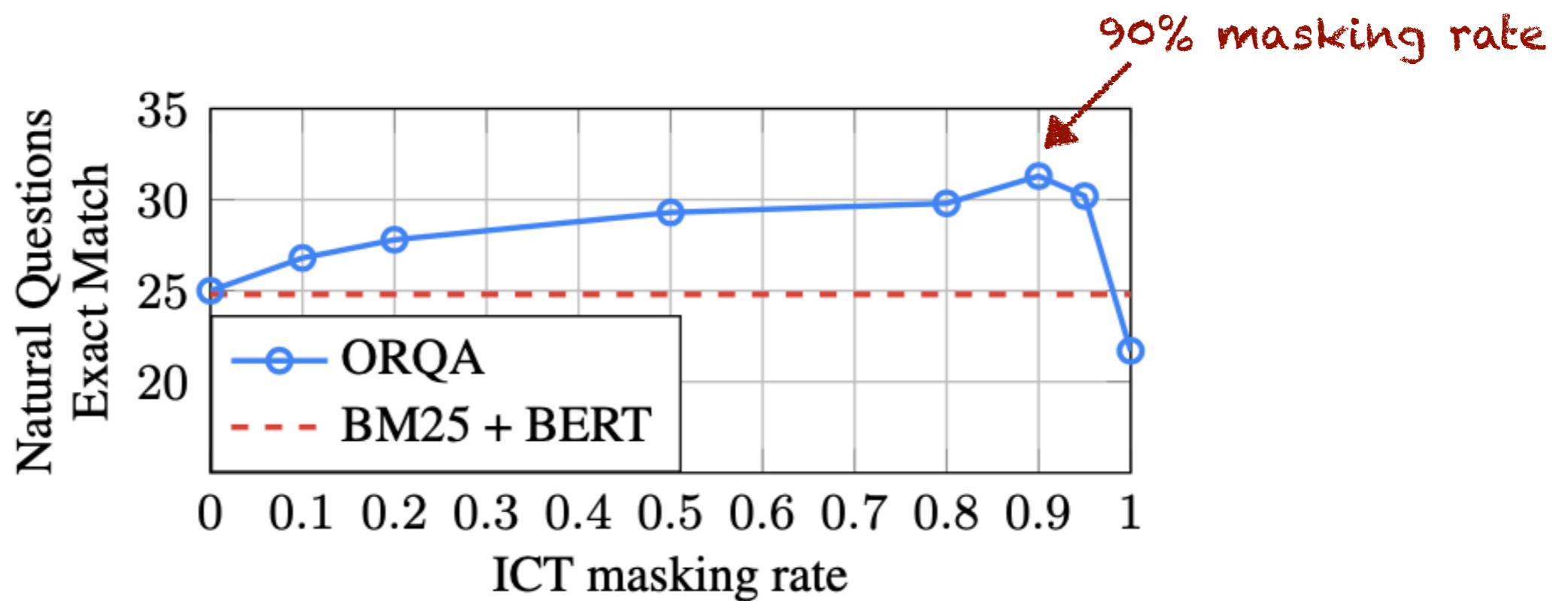
**Key idea:** a sentence is treated as a *pseudo-question*, and its context is treated as *pseudo-evidence*. The goal is to predict the correct context among a set of other random options.

...Zebras have four gaits: walk, trot, canter and gallop. **They are generally slower than horses, but their great stamina helps them outrun predators.** When chased, a zebra will zig- zag from side to side...



# Pre-training: Inverse Cloze Task (ICT)

Still encourages the model to learn word matching – only remove sentences from its context in **90%** of the examples!



**Important** - After pre-training, **BERT<sub>B</sub> is fixed** and all the block representations can be pre-computed and search efficiently using existing tools (e.g., Locality Sensitive Hashing).

# ORQA: experimental results

Model	BM25 +BERT	ORQA
Natural Questions	26.5	<b>33.3</b>
WebQuestions	17.7	<b>36.4</b>
CuratedTrec	21.3	<b>30.1</b>
TriviaQA	<b>47.1</b>	45.0
SQuAD	<b>33.2</b>	20.2

ORQA wins!

Test

TriviaQA/SQuAD

NQ/WebQ/TREC

BM25 + BERT wins!

question writer knows the answer

genuine information-seeking questions

Dataset bias? SQuAD: only 536 documents, violating IID assumption?

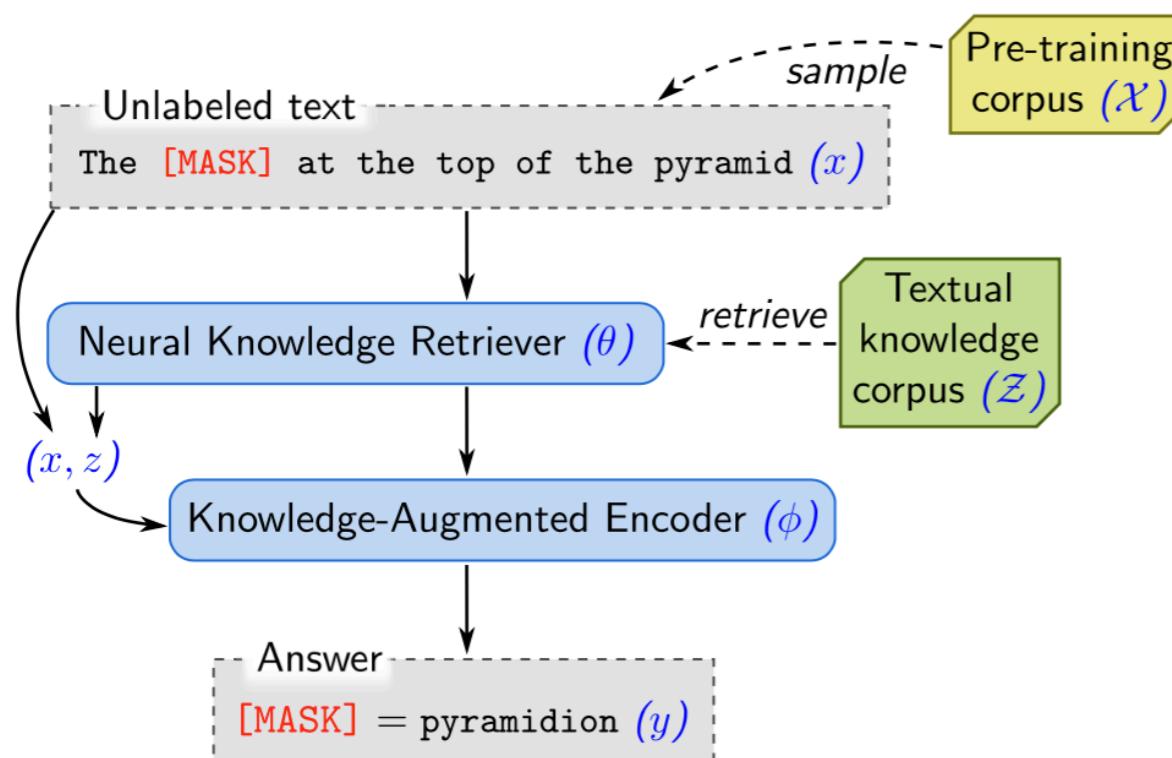
# REALM: Retrieval-augmented Language Model [Guu et al., 2020]

Pre-training on both  
retriever and reader!

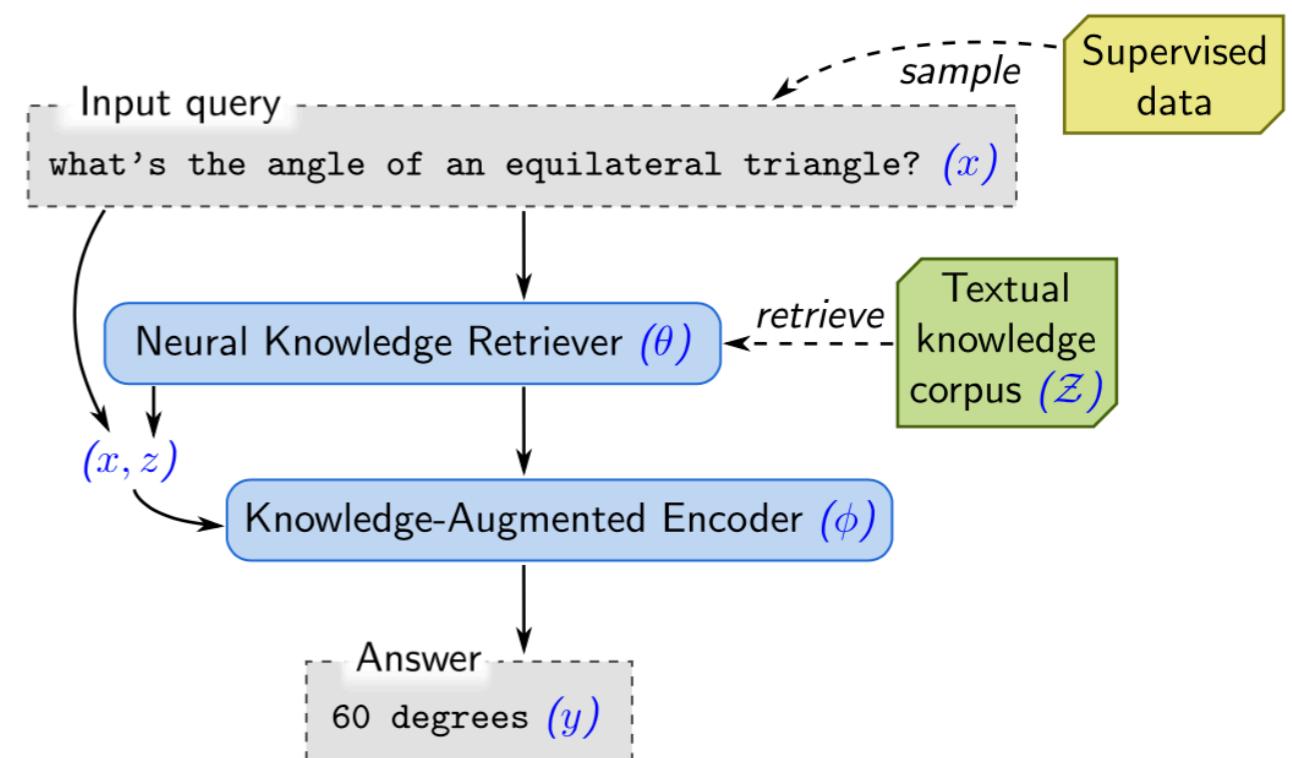
$$p(y | x) = \sum_{z \in \mathcal{Z}} p(y | z, x) p(z | x)$$

$\mathcal{Z} = \text{Top}(K)$  passages

## Pre-training: masked language model (MLM)

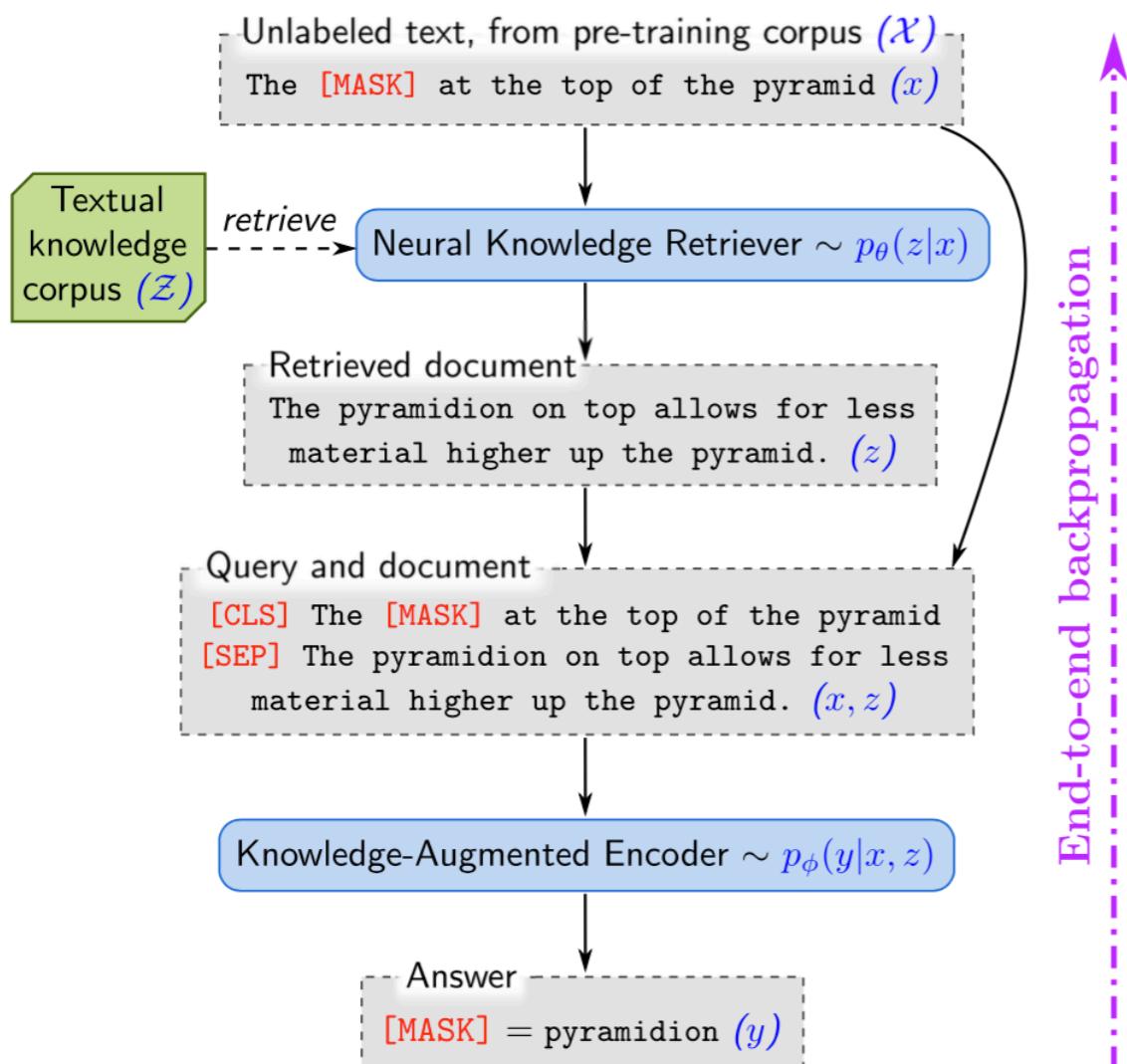


## Fine-tuning: QA



# REALM: MLM pre-training

## MLM pre-training



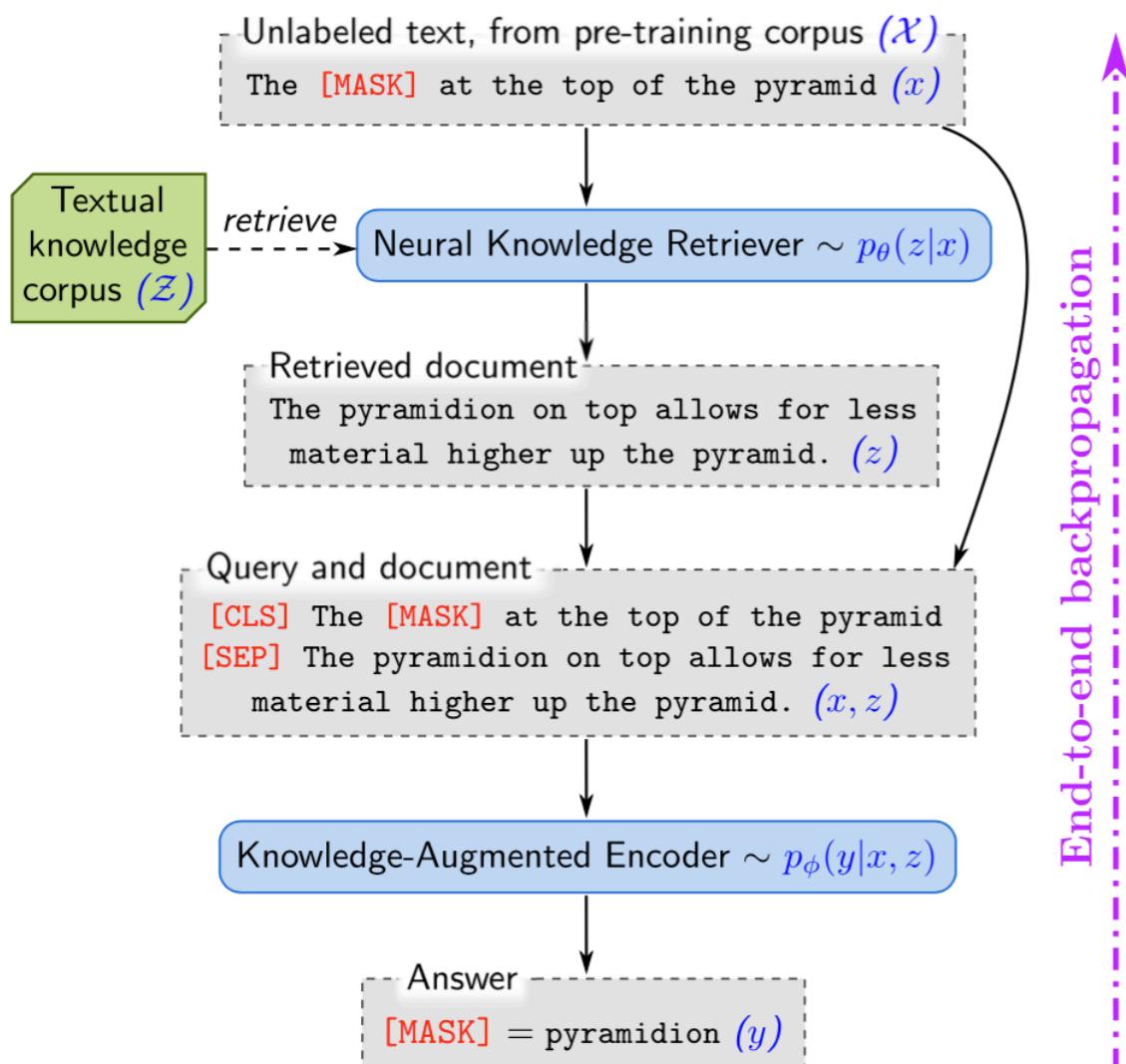
- Cold-start is hard => Use ICT as a first-stage pre-training
- **Salient span masking:** similar to span masking [Joshi et al., 2020] but only mask named entities + dates.

## Evaluated on NQ

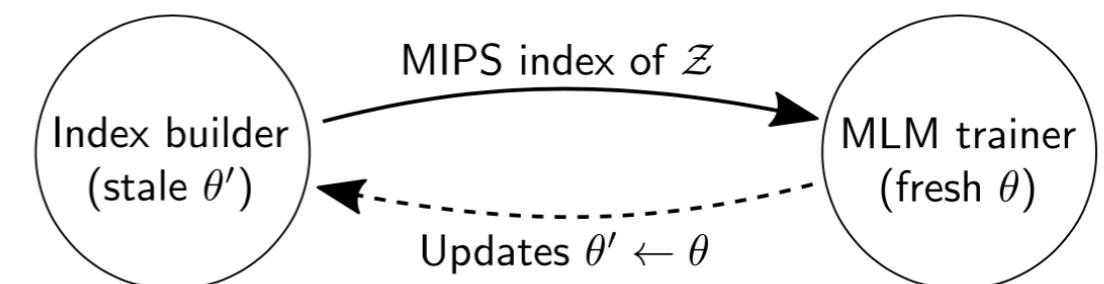
Random uniform masks: 32.3
Random span masks: 35.3
<b>Salient span masks: 38.2</b>

# REALM: MLM pre-training

## MLM pre-training



- Can use corpora even larger than Wikipedia for pre-training
  - CC-News vs Wikipedia: **40.4** vs **39.2**
- Can allow updating evidence encoder (**BERT<sub>B</sub>**) asynchronously



**ORQA vs REALM: 33.3 vs 40.4 on NQ**

# Take-aways from ORQA/REALM

- It is possible to jointly train the retriever and reader, without any sparse IR components.
- The pre-training makes this retrieval process feasible and pre-training strategies matter.
- However, pre-training is very expensive.

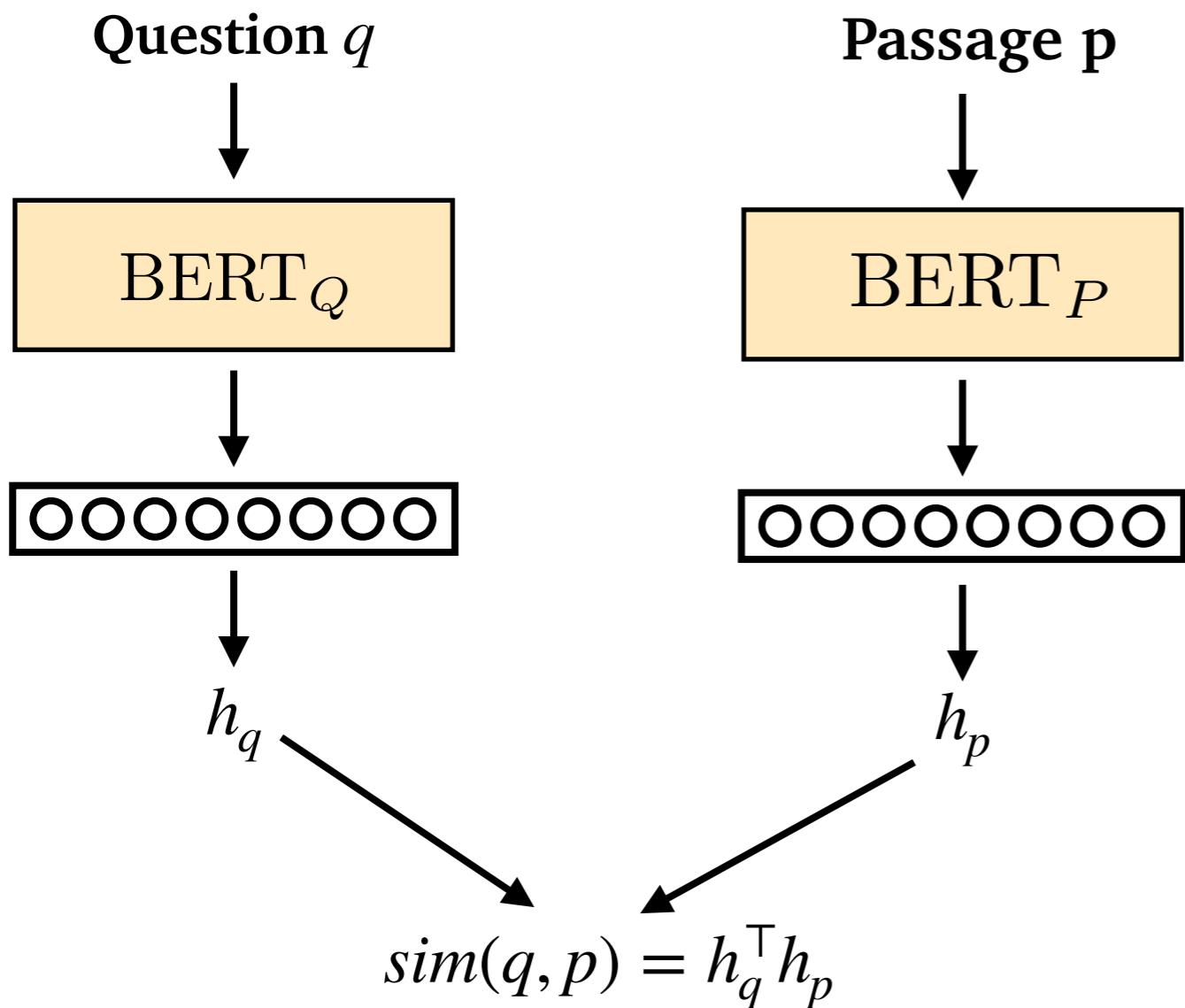
*“We pre-train for 200k steps on 64 Google Cloud TPUs, with a batch size of 512”*

Next question: Is pre-training really necessary?

# Dense Passage Retrieval (DPR)

[Karpukhin et al., 2020]

**Key contribution:** you can train a dense retrieval only from a small number of Q/A pairs, without any pre-training!



How to get positives and negatives?

$$\mathcal{D} = \{\langle q_i, p_i^+, p_{i,1}^-, \dots, p_{i,n}^- \rangle\}_{i=1}^m$$

$$L(q_i, p_i^+, p_{i,1}^-, \dots, p_{i,n}^-) = -\log \frac{e^{\text{sim}(q_i, p_i^+)}}{e^{\text{sim}(q_i, p_i^+)} + \sum_{j=1}^n e^{\text{sim}(q_i, p_{i,j}^-)}}$$

# DPR: training examples

## Positives

- (1) Provided in the reading comprehension datasets
- (2) Passages of high BM25 scores that contain the answer string

## Negatives

- (1) Random passages from the corpus
- (2) Passages of high BM25 scores that DO NOT contain the answer string
- (3) Positive passages of **OTHER** questions

*The best model uses (3) from the same mini-batch [in-batch negatives] and one passage from (2) [hard negatives].*

# DPR: in-batch negatives

- A small trick to effectively generate more training pairs
- Suppose we have  $n$  pairs of relevant questions and passages. Let  $Q_{d \times n}$  and  $P_{d \times n}$  be the question and passage embeddings.
- $S = Q^T P$  is a  $n$ -by- $n$  matrix of the similarity scores. Scores of  $n^2$  pairs of questions and passages. For each question, 1 positive passage and  $n-1$  negative passages

in-batch negatives

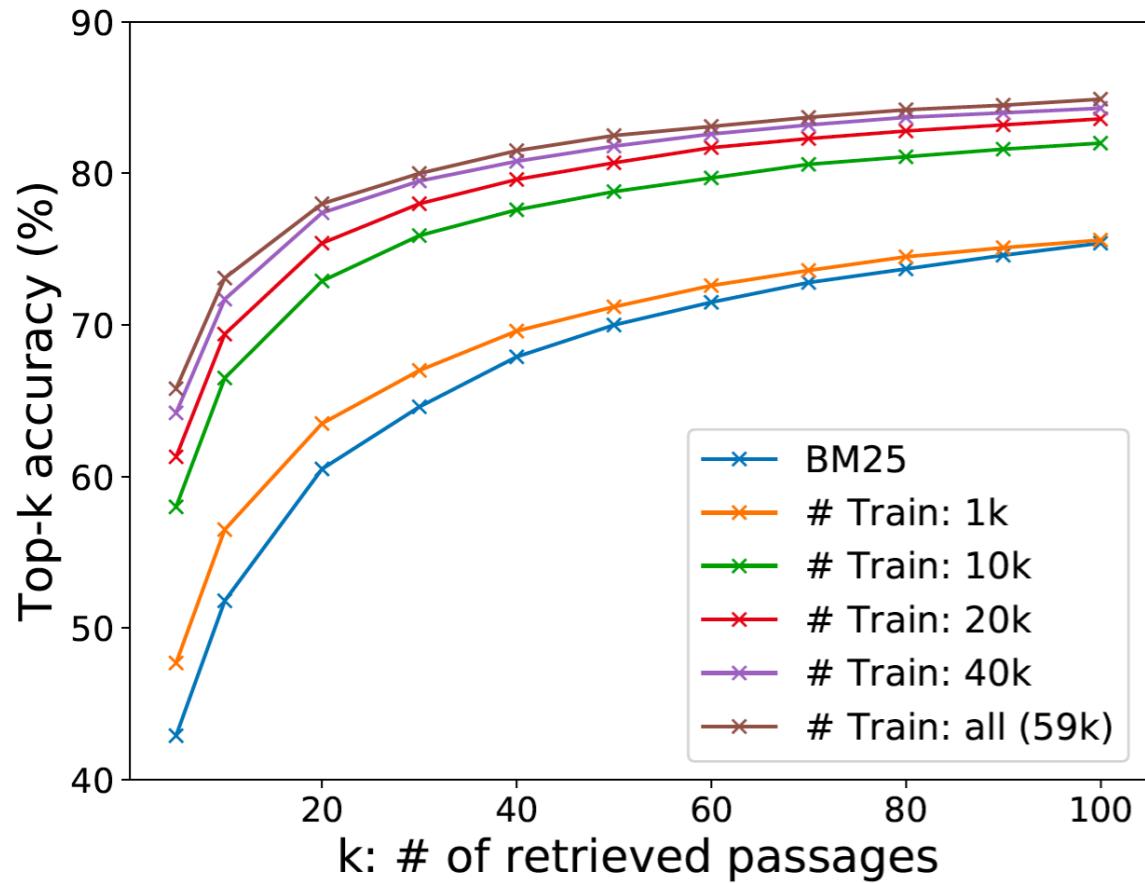
Type	#N	IB	Top-5	Top-20
Random	7	✗	47.0	64.3
BM25	7	✗	50.0	63.3
Gold	7	✗	42.6	63.1
Gold	7	✓	51.1	69.1
Gold	31	✓	52.1	70.8
Gold	127	✓	55.8	73.0
G.+BM25 <sup>(1)</sup>	31	✓	65.0	77.3
G.+BM25 <sup>(2)</sup>	31	✓	64.5	76.4
G.+BM25 <sup>(1)</sup>	127	✓	<b>65.8</b>	<b>78.0</b>

Retriever performance  
(Natural Questions)

# DPR: experimental results

## Retriever performance on NQ

1k Q/A pairs beat BM25!



## End-to-end QA performance

With a *multi-passage BERT reader* trained on the retrieved passages from the retriever:

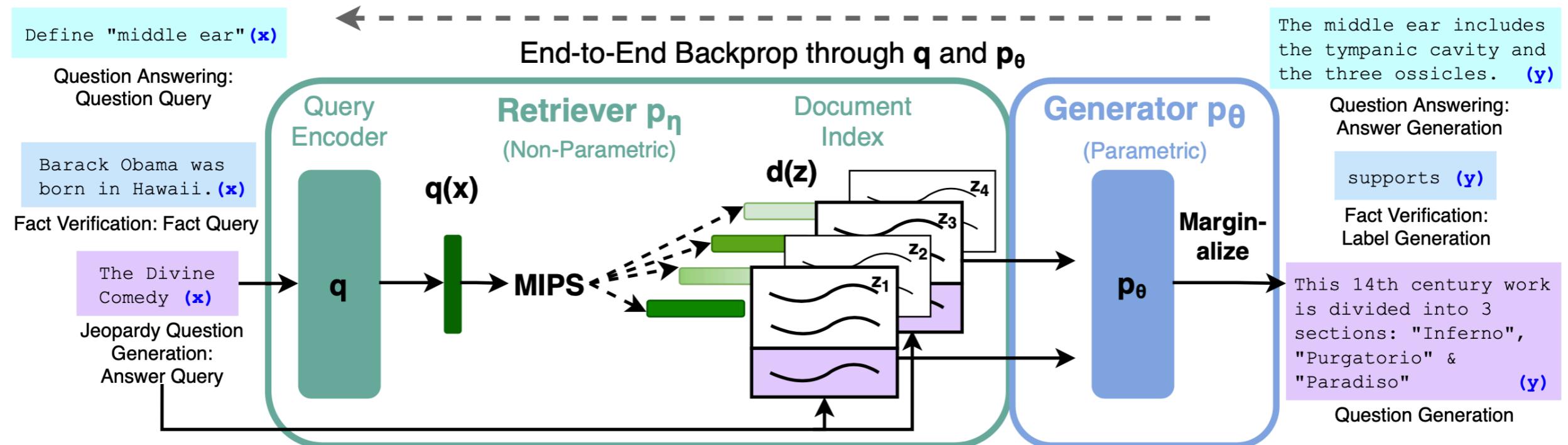
- DPR is better than BM25 on NaturalQuestions, WebQuestions, TREC, [TriviaQA](#) but not SQuAD.
- DPR is better than REALM on bigger datasets ([41.5](#) vs [39.2](#) on NQ). For smaller datasets (WebQ, TREC), it needs a mixed training with bigger datasets to outperform REALM.

# Take-aways from DPR

- No need of expensive pre-training?
  - At least for modest-sized QA datasets
- **Joint** training vs **pipeline** training of retriever and reader?
  - Joint training didn't help DPR ( $41.5 \rightarrow 39.8$  on NQ)
  - Pipeline training is more efficient. You only build the index once!
- **Reading comprehension** vs **QA** datasets
  - Doesn't make a much difference ( $41.5$  vs  $41.0$  on NQ).
  - You can train the system using only Q/A pairs!
- BM25 + DPR is slightly better than DPR but the difference is small.

# Retrieval-Augmented Generation (RAG)

[Lewis et al., 2020]



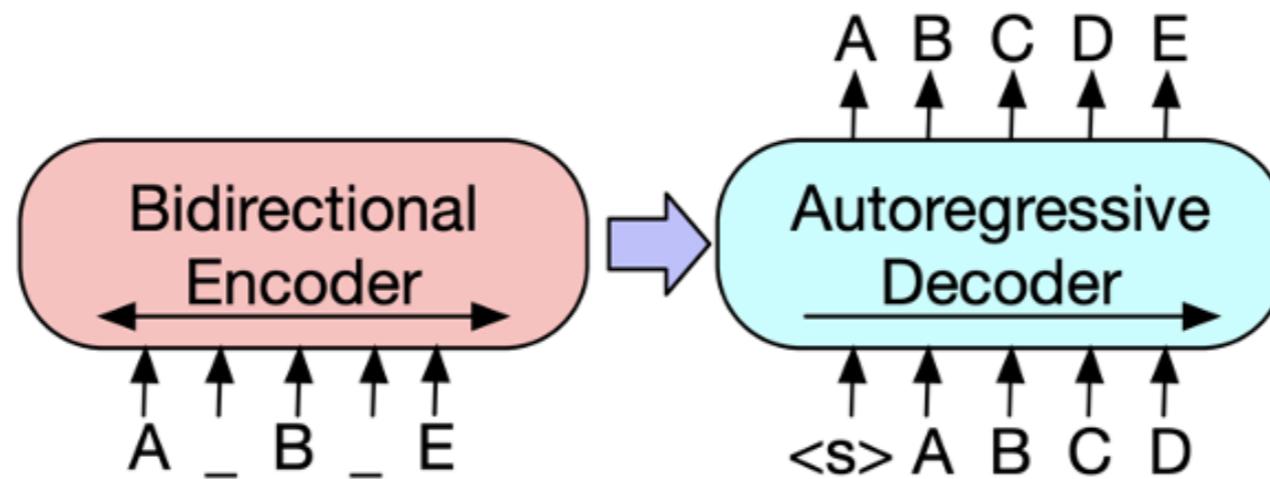
$$p_{\text{RAG-Sequence}}(y|x) = \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) \prod_i^N p_\theta(y_i|x, z, y_{1:i-1})$$

↑  
retrieval model      ↑  
seq2seq model

# Retrieval-Augmented Generation (RAG)

- Retrieval model:  $p(z | x)$  = Dense passage retrieval (DPR)
- seq2seq model:  $p(y | x, z)$  = BART [Lewis et al., 2020]

Pre-trained on large text corpora, simply concatenates  $z$  to question  $x$



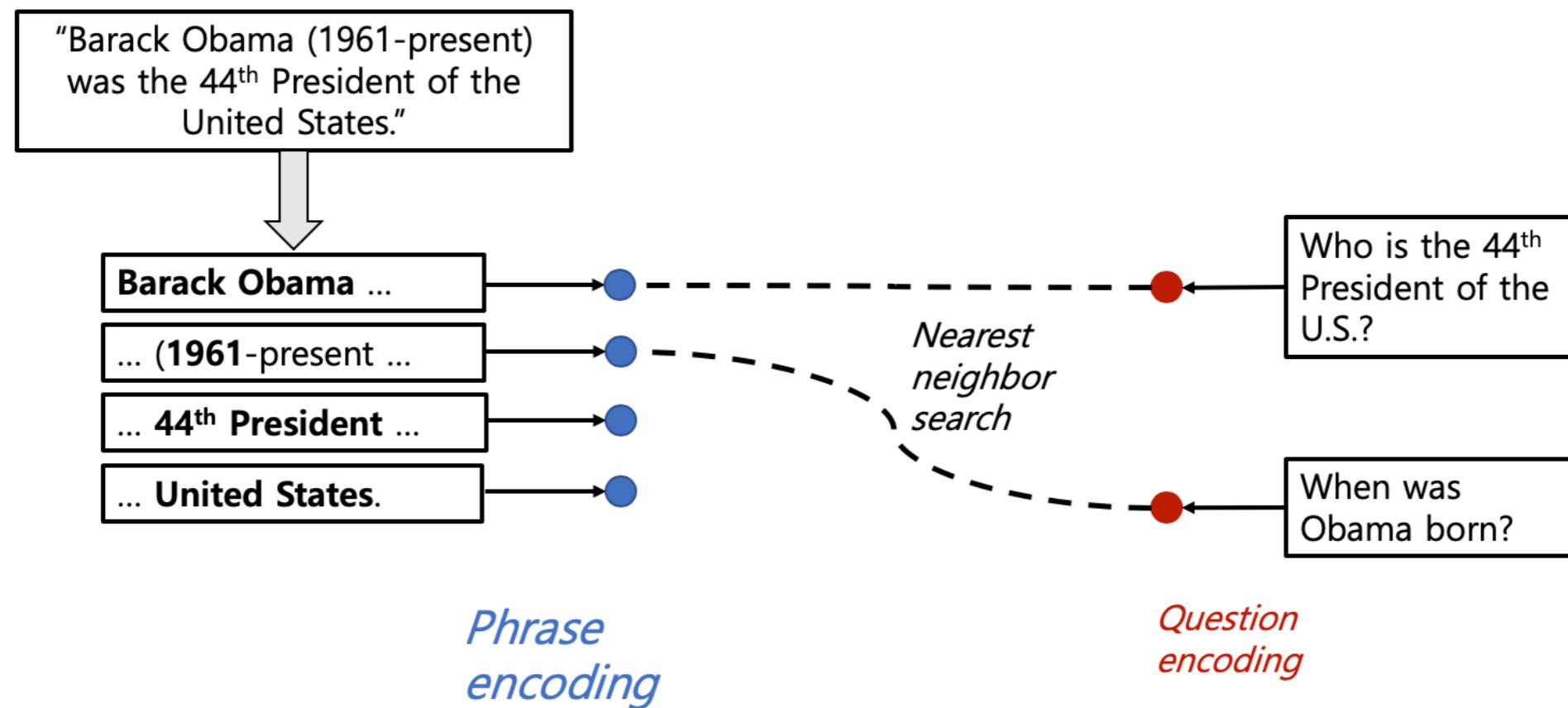
- Joint training of the two components
- Improved performance on open-domain QA evaluation and easy to extend to other generation tasks

# DenSPI: Dense-Sparse Phrase Index

[Seo et al., 2019]

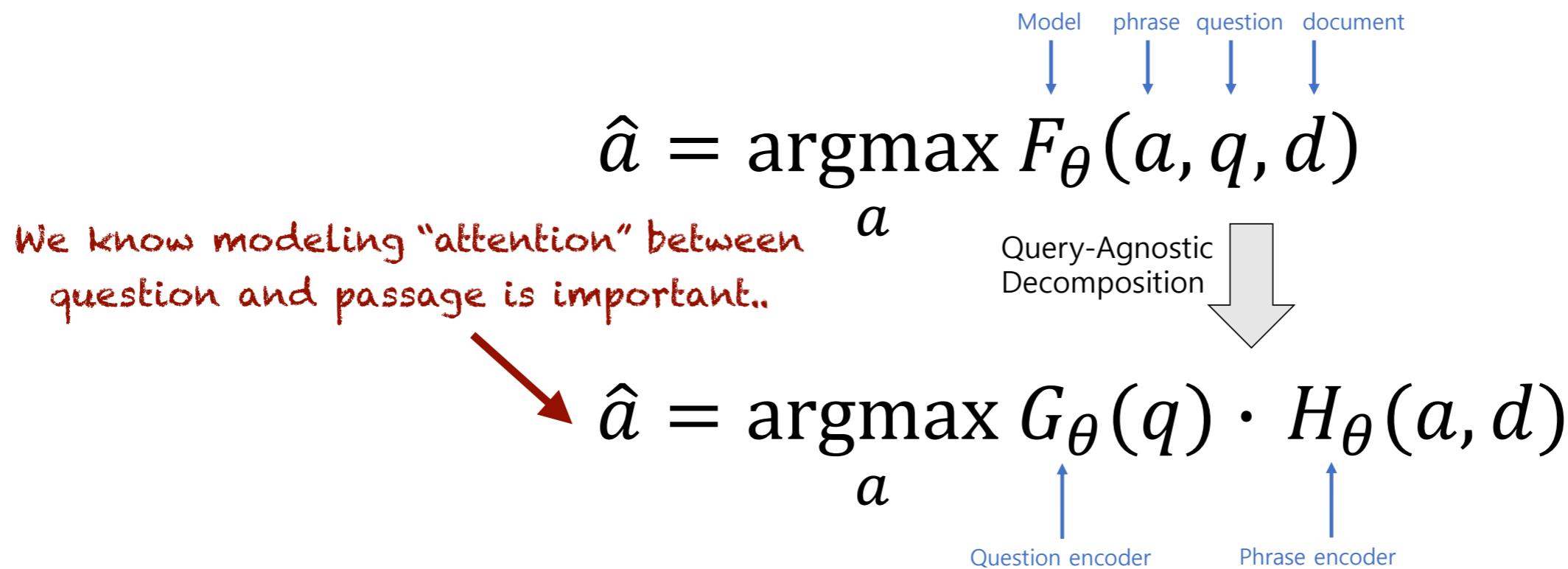
**Key question:** Is it possible to encode/index at phrase level instead of paragraphs or documents so retriever + reader will be reduced to a harder “retriever” problem?

## Phrase Indexing



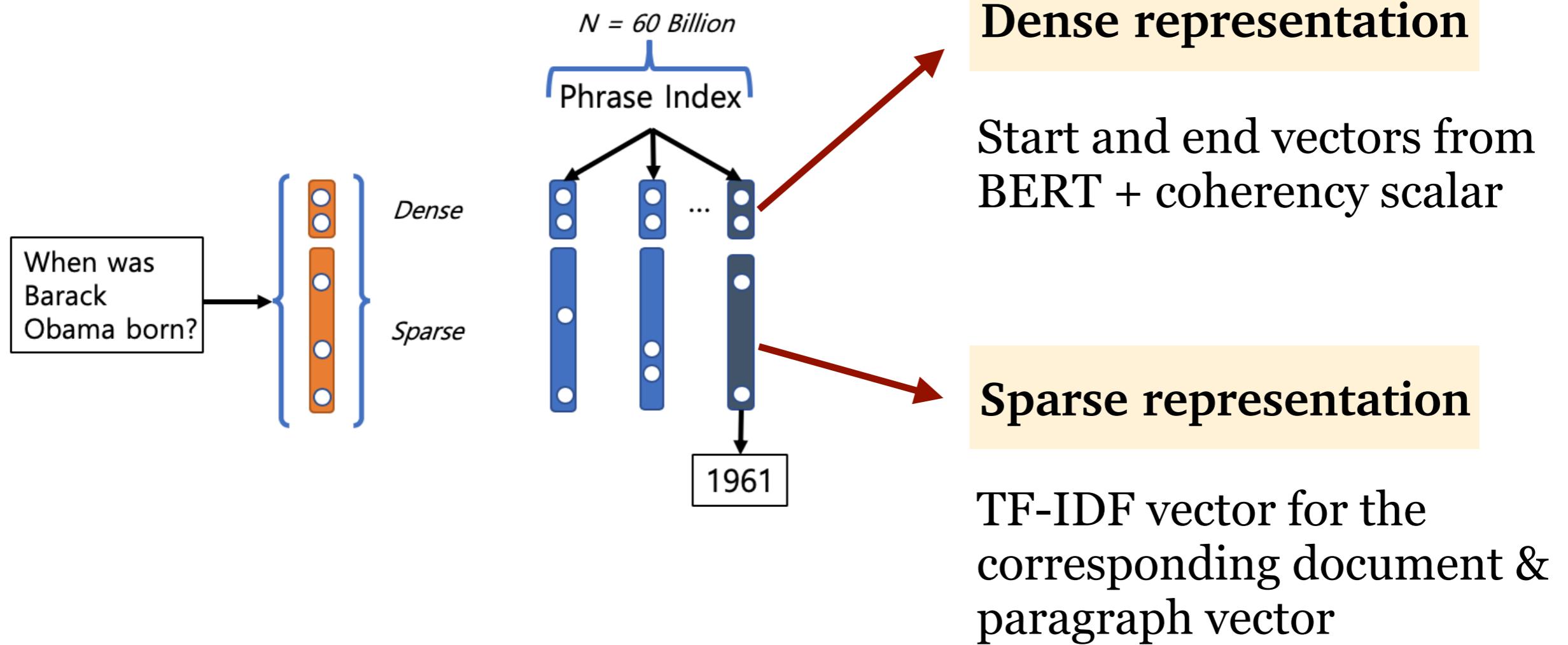
# DenSPI: Challenges

**Decomposability gap:** question and passage/answer have to be encoded independently



**Scalability:** 60 billion phrases in Wikipedia! Storage, indexing and search all remain challenging.

# DenSPI: Dense & sparse representations



# DenSPI: Results

**Storage:** pointer, filter, scalar quantization: 240T → 1.5T

**Search:** Dense-first search (DFS), sparse-first search (SFS), Hybrid

Performance on SQuAD

	F1	EM	s/Q
DrQA	-	29.8	35
R <sup>3</sup>	37.5	-	-
Paragraph ranker	-	30.2	-
Multi-step reasoner	39.2	31.9	-
MINIMAL	42.5	34.7	-
BERTserini	46.1	38.6	115
Weaver	-	42.3	-
DENSPI-SFS	42.5	33.3	0.60
DENSPI-DFS	35.9	28.5	0.51
-sparse scale=0	16.3	11.2	0.40
DENSPI-Hybrid	44.4	36.2	0.81

Removing sparse vector  
performs much worse

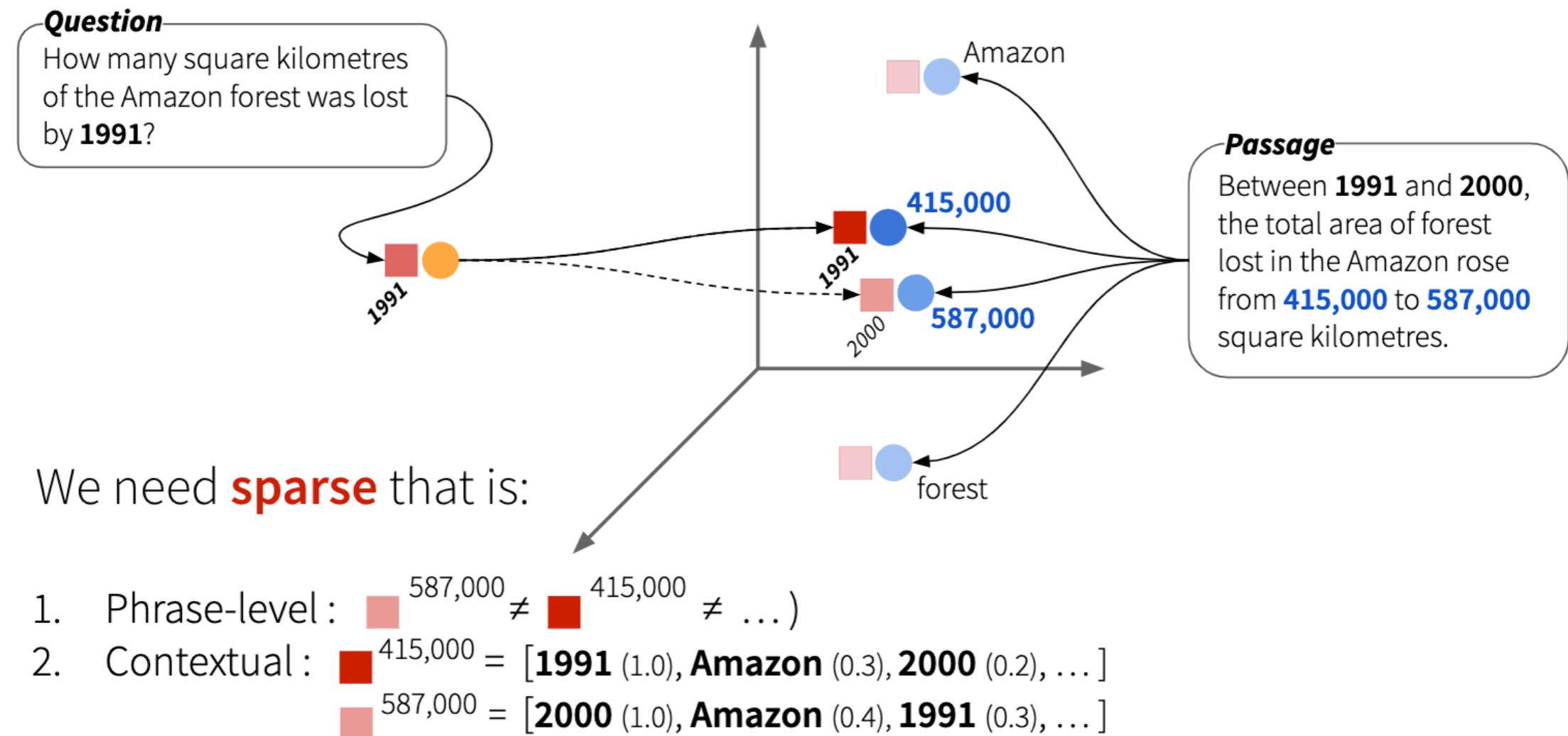


Very fast inference time



# DenSPI: Contextualized sparse representations

[Lee et al., 2020]



- **Sparc**: use different weights for different sparse terms
- Improved DenSPI by **+4.1%** on TREC and **+4.5%** on SQuAD

# Comparison of all models

		SQuAD	TriviaQA	NaturalQ	TREC	WebQ
2017/04	DrQA	28.4	-	-	25.7	19.5
2017/08	R <sup>3</sup>	29.1	-	-	28.4	17.1
2018/04	DrQA*	40.4	-	-	28.8	24.3
2019/02	BERTserini	38.6	-	-	-	-
2019/08	Multi-passage BERT@	53.0	-	-	-	-
2019/09	Hard-EM	-	50.9	28.1	-	-
2019/06	ORQA	20.2	45.0	33.3	30.1	36.4
2020/02	REALM (Wiki)	-	-	39.2	46.8	40.2
2020/02	REALM (CC-News)	-	-	40.4	42.9	40.7
2020/04	DPR	29.8	56.8	41.5	49.4*	42.4*
2020/05	RAG@	-	56.1	44.5	52.2*	45.2*
2019/06	DenSPI	36.2	-	-	31.6#	-
2019/11	DenSPI + Sparc	40.7	-	-	35.7#	-
2020/04	BM25 + DPR	36.7	57.0	39.0	50.6*	41.1*

#: no supervision using target training data / \*: jointly trained with bigger datasets (e.g., NQ) / @: BERT-large models

# Summary

- ORQA/REALM: First demonstrate that it is possible jointly train the retriever and reader without any sparse IR components; requires novel ways of pre-training
- DPR/RAG: It is possible to learn the dense retrieval only from Q/A pairs without pre-training!
- DenSPI /Sparc: It is possible to index and retrieve at phrase level without requiring an explicit “reader”. Very fast inference time but slightly worse performance. Sparse features still matter.

Questions?