香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

COMP 4901B
Large Language Models

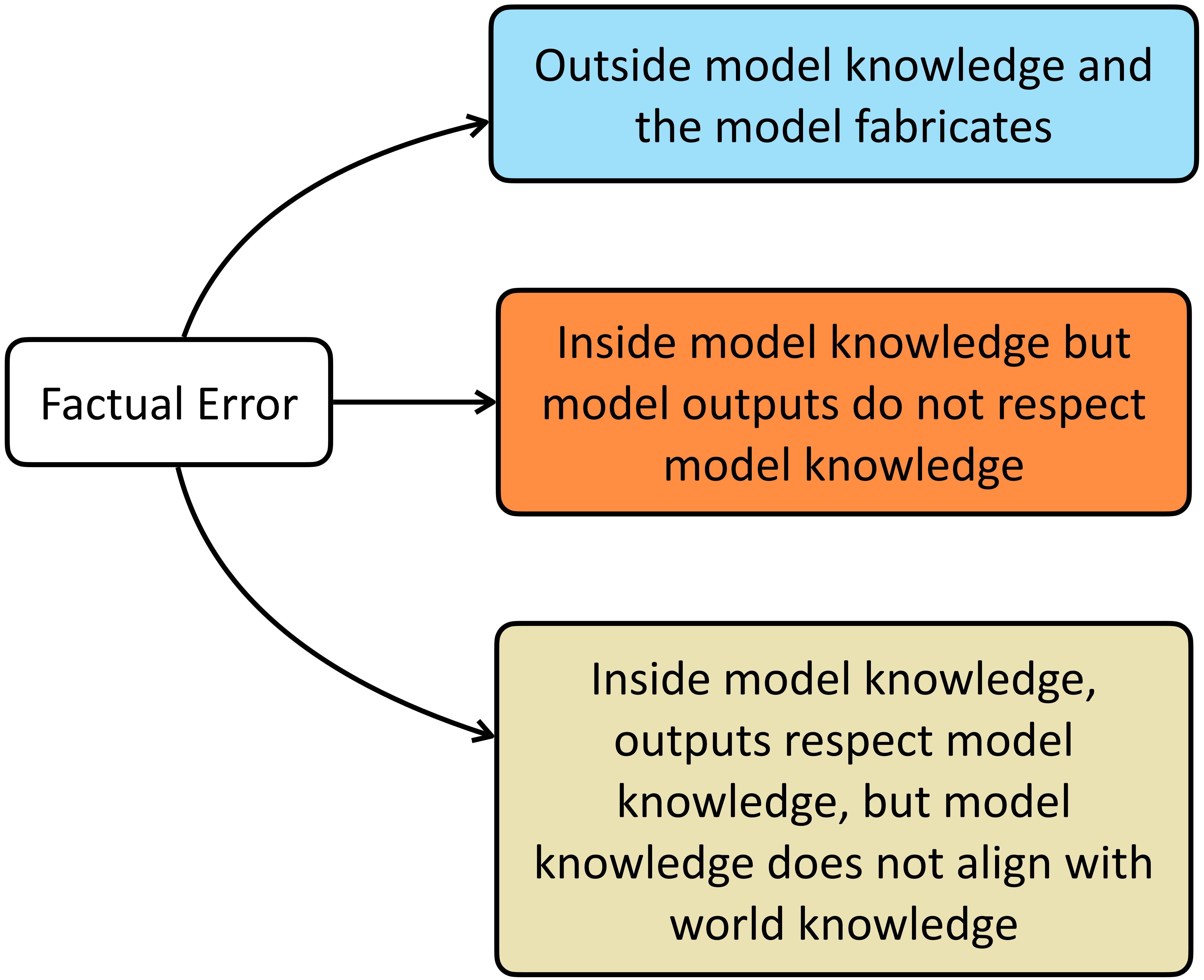
RAG and MoE LLMs

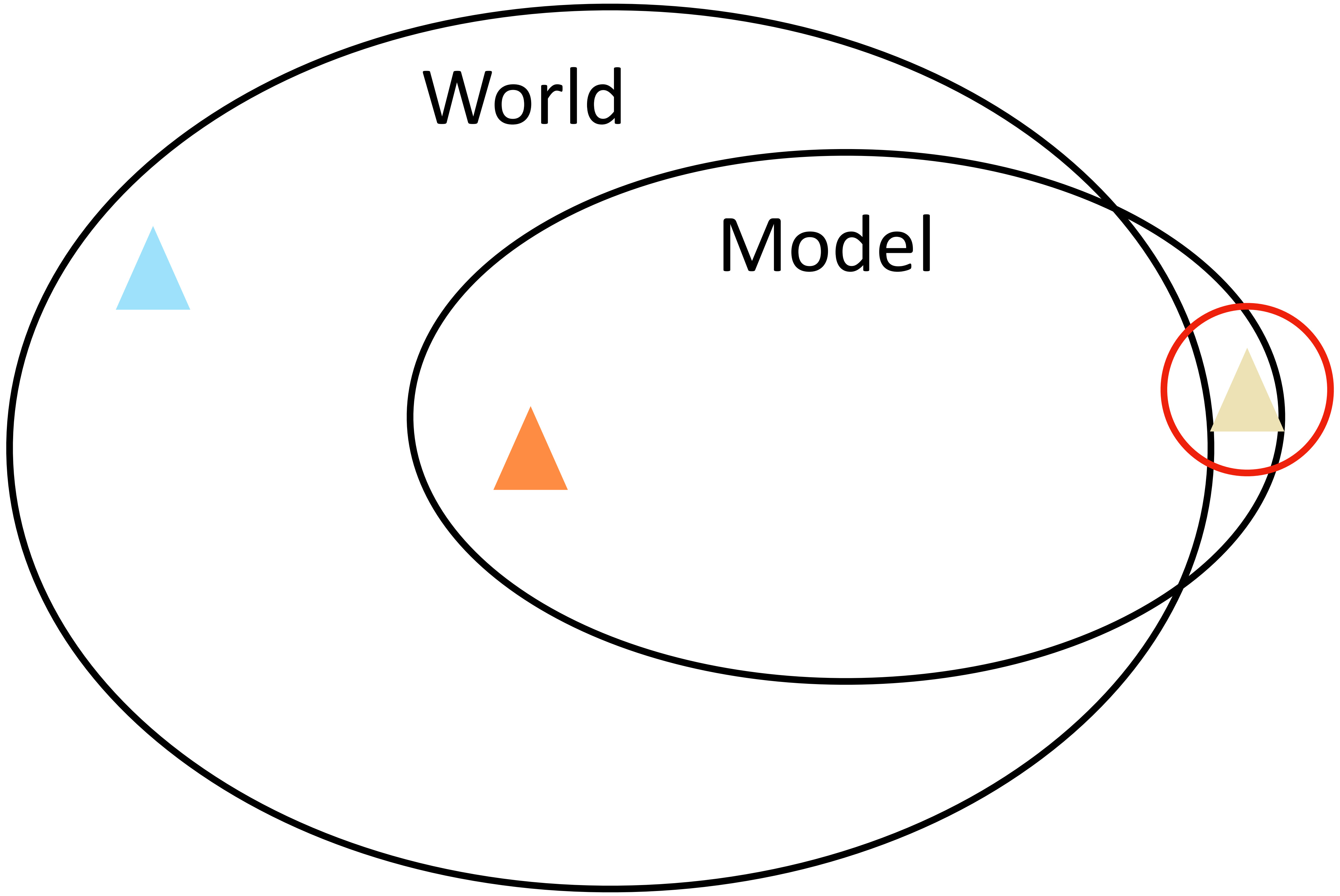
Junxian He

Nov 26, 2025

Part of slides are adapted from CMU 11711

Recap: Factual Error Sources



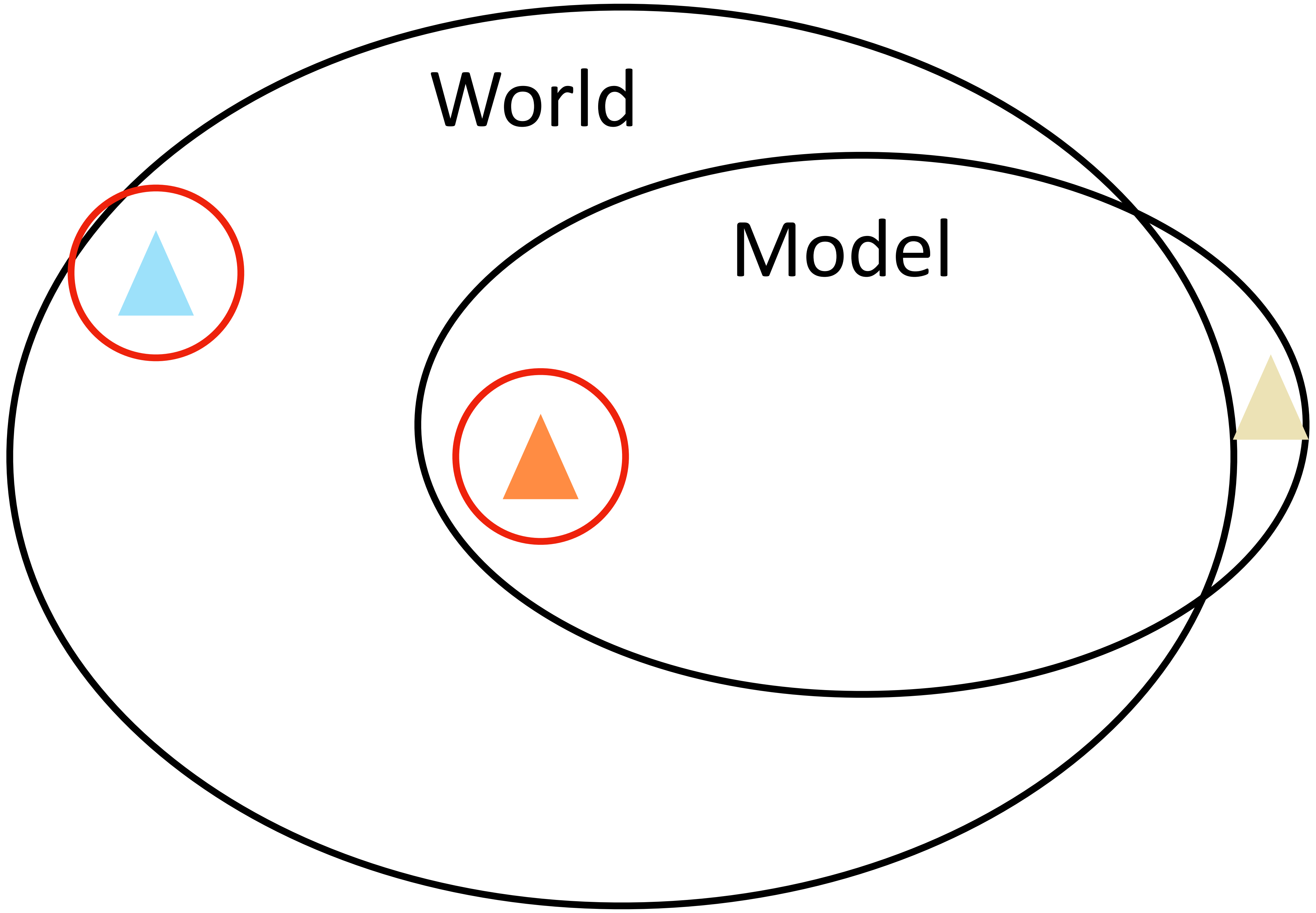


Factual error, not hallucination. No solution without relying on external knowledge

Factual Error Sources

Factual Error

- Outside model knowledge and the model fabricates
- Inside model knowledge but model outputs do not respect model knowledge
- Inside model knowledge, outputs respect model knowledge, but model knowledge does not align with world knowledge

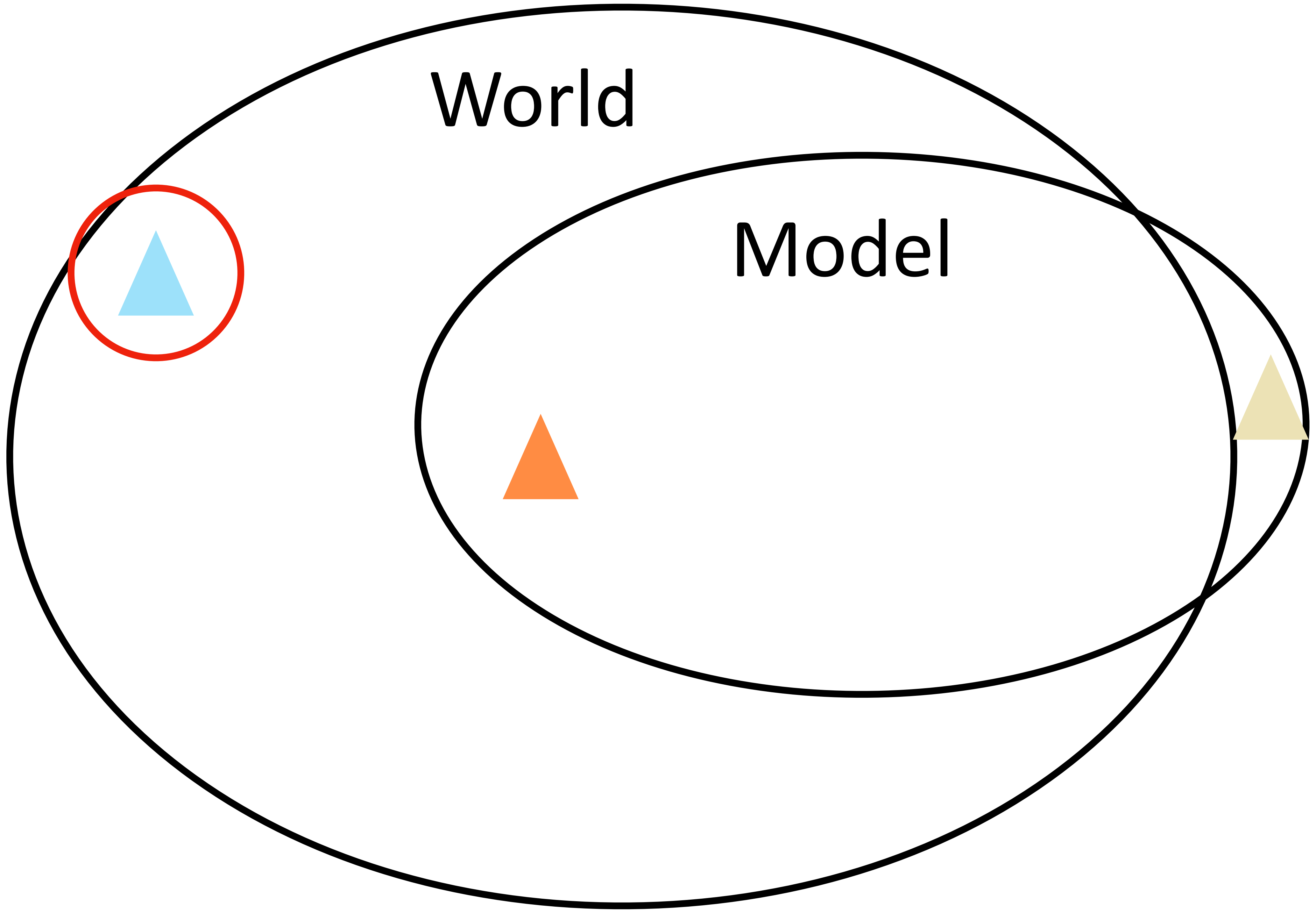


Hallucination, dishonest

Factual Error Sources

Factual Error

- Outside model knowledge and the model fabricates
- Inside model knowledge but model outputs do not respect model knowledge
- Inside model knowledge, outputs respect model knowledge, but model knowledge does not align with world knowledge

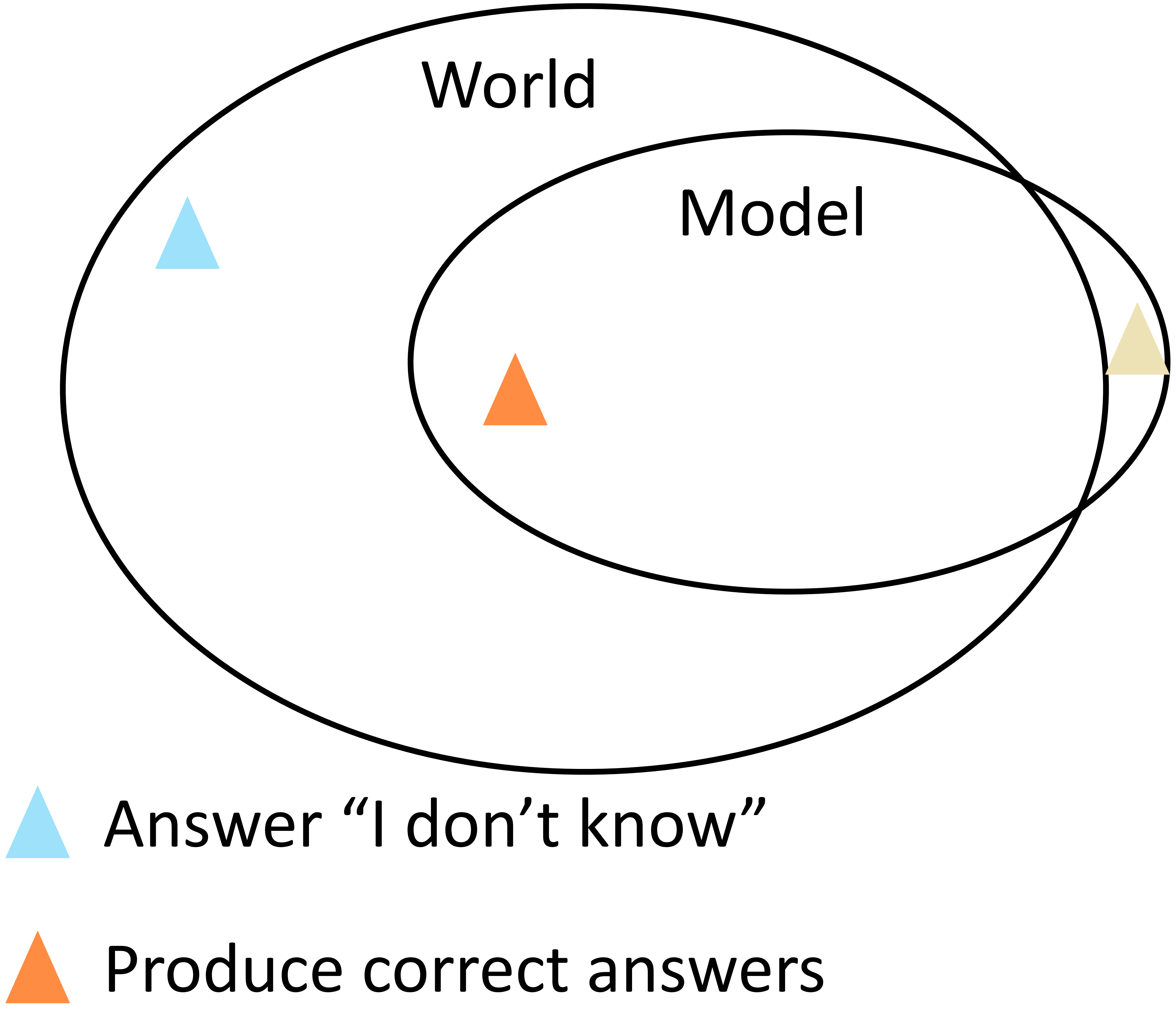


Impossible to produce correct answers without external tools

What Can We Do to Mitigate Hallucination?

Factual Error

- Outside model knowledge and the model fabricates
- Inside model knowledge but model outputs do not respect model knowledge
- Inside model knowledge, outputs respect model knowledge, but model knowledge does not align with world knowledge



Recap: How to Improve Models' Factual Correctness?

Challenges:

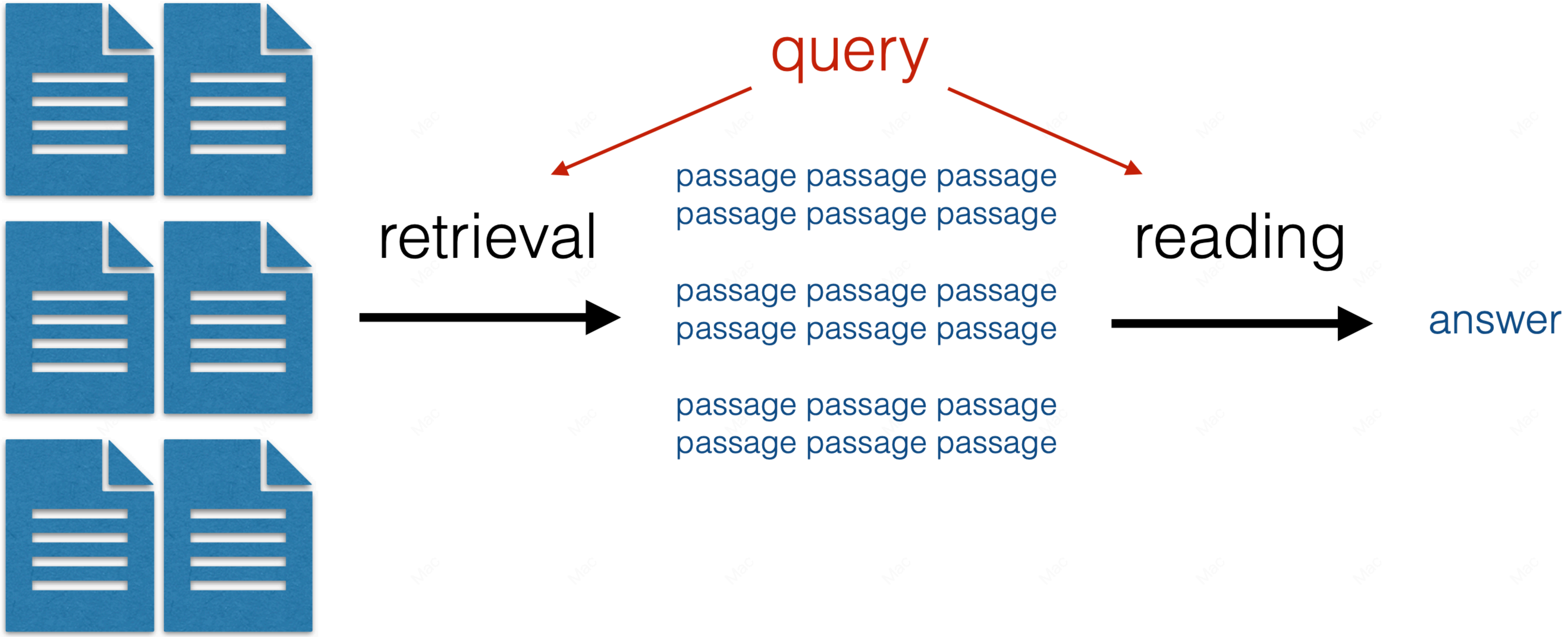
1. Difficult to update static knowledge (e.g., president of US)
2. The world is producing new knowledge everyday

Retrieval is currently considered the best solution to improve factual correctness

Retrieval also resembles how humans access knowledge

Retrieval-Augmented Generation (RAG)

- **Retrieve** relevant passages efficiently
- **Read** the passages to answer the query



Your group project is actually a standard RAG example, but in a more advanced manner with multiple steps

Chen et al. Reading Wikipedia to Answer Open-Domain Questions. 2017

Example

You

I think Vin Diesel has been a voice actor for several characters in TV series, do you know what their names are? Please list all of them.

ChatGPT

Vin Diesel has lent his voice to several characters across various TV series and shows. Here's a list of the characters he's voiced:

1. **Groot** in "Guardians of the Galaxy Vol. 3" (2023 Movie), "The Guardians of the Galaxy Holiday Special" (2022 TV Show), "I Am Groot" (2022 Short), "Thor: Love and Thunder" (2022 Movie), "Avengers: Endgame" (2019 Movie), "Ralph Breaks the Internet: Wreck-It Ralph 2" (2018 Movie), "Avengers: Infinity War" (2018 Movie), "Guardians of the Galaxy Vol. 2" (2017 Movie), and "Guardians of the Galaxy" (2014 Movie) [”].
2. **Santiago** in "ARK: The Animated Series" (2023 TV Show) [”].
3. **Dominic Toretto** in "Fast & Furious: Spy Racers" (2019 TV Show) [”].

https://www.behindthevoiceactors.com/Vin-Diesel/

Additionally, though the website for "Big Mouth" lists Vin Diesel, it appears to be a misunderstanding or an error, as Nick Kroll is credited as the voice of Vin Diesel in that show [”].

https://www.behindthevoiceactors.com/tv-shows/Big-Mouth/Vin-Diesel/

These roles showcase Diesel's versatility and contribution to voice acting, particularly in animated series and specials, where he brings iconic characters to life with his distinctive voice.

Retrieval Methods

- **Sparse retrieval**
- **Document-level dense retrieval**
- **Token-level dense retrieval**
- **Cross-encoder reranking**
- **Black-box retrieval** (just ask Google/Bing)

Sparse Retrieval

- Express the query and document as a sparse word frequency vector (usually normalized by length)

q=what is nlp

$d_1 =$ what is life ? candy is life !

$d_2 =$ nlp is an acronym for natural language processing

$d_3 =$ I like to do good research on nlp

$$\begin{array}{c} \text{what}\\ \text{candy}\\ \text{nlp}\\ \text{is}\\ \text{language}\\ \dots \end{array} \quad \begin{pmatrix}0.33\\0\\0.33\\0.33\\0\\\dots\end{pmatrix} \quad \begin{pmatrix}0.25\\0.125\\0\\0.25\\0\\\dots\end{pmatrix} \quad \begin{pmatrix}0\\0\\0.125\\0.125\\0\\\dots\end{pmatrix} \quad \begin{pmatrix}0\\0\\0.125\\0\\0\\\dots\end{pmatrix}$$

$q{*}d_1 = 0.165$ $\quad q{*}d_2 = 0.0825$ $\quad q{*}d_3 = 0.0413$

- Find the document with the highest inner-product or cosine similarity in the document collection

Term-Weighting

- Some terms are more important than others; low-frequency words are often more important

TF-IDF: Term frequency - Inverse document frequency

$$\mathrm{TF}(t,d) = \frac{\mathrm{freq}(t,d)}{\sum_{t'} \mathrm{freq}(t',d)} \qquad \mathrm{IDF}(t) = \log\left(\frac{|D|}{\sum_{d' \in D} \delta(\mathrm{freq}(t,d') > 0)}\right)$$

$$\mathrm{TF\text{-}IDF}(t,d) = \mathrm{TF}(t,d) \times \mathrm{IDF}(t)$$

For example, the df (document frequency) and idf for some words in Shakespeare's 37 plays are as follows:[5]

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.966
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

BM25 (Best-Matching 25)

Given a query Q, containing keywords $q_1, \ldots, q_n$, the BM25 score of a document D is:

$$\text{score}(D, Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)}$$

where $f(q_i, D)$ is the number of times that the keyword q_i occurs in the document D, $|D|$ is the length of the document D in words, and avgdl is the average document length in the text collection from which documents are drawn. k_1 and b are free parameters, usually chosen, in absence of an advanced optimization, as $k_1 \in [1.2, 2.0]$ and $b = 0.75$.[3] $\text{IDF}(q_i)$ is the IDF (inverse document

https://en.wikipedia.org/wiki/Okapi_BM25

Inverted Index

- A data structure that allows for efficient sparse lookup of vectors

Sparse Vectors

$$
\begin{array}{r ccc}
 & d_1 & d_2 & d_3 \\
\begin{array}{r} \text{what} \\ \text{candy} \\ \text{nlp} \\ \text{is} \\ \text{language} \\ \\ \end{array}
& \begin{pmatrix} 2 \\ 1 \\ 0 \\ 2 \\ 0 \\ \dots \end{pmatrix}
& \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ \dots \end{pmatrix}
& \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ \dots \end{pmatrix}
\end{array}
$$

$\rightarrow$

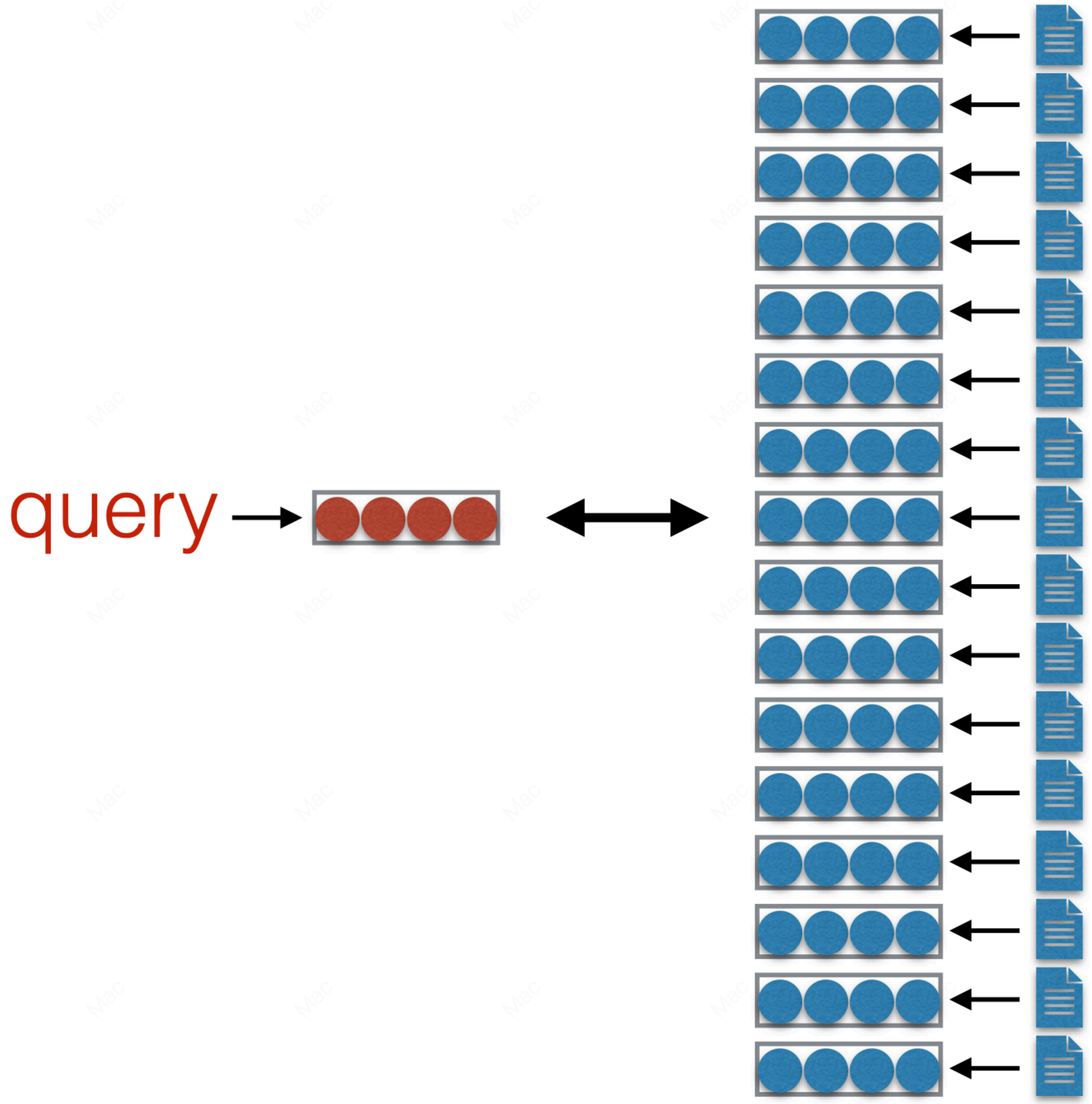
Index

```
{
  “what”: [1],
  “candy”: [1],
  “nlp”: [2,3],
  “is”: [1,2],
  “language”: [2],
  …
}
```

We can quickly look up which documents contain the keywords

Dense Retrieval

- Encode document/query and find nearest neighbor
- Can use:
  - Out-of-the-box embeddings
  - Learned embeddings



Even though dense retrieval seems more advanced, training-free sparse retrieval tends to be more robust in open-ended scenarios

Learning Retrieval-Oriented Embeddings

- Select positive and negative documents, train using a contrastive loss (e.g. hinge loss)

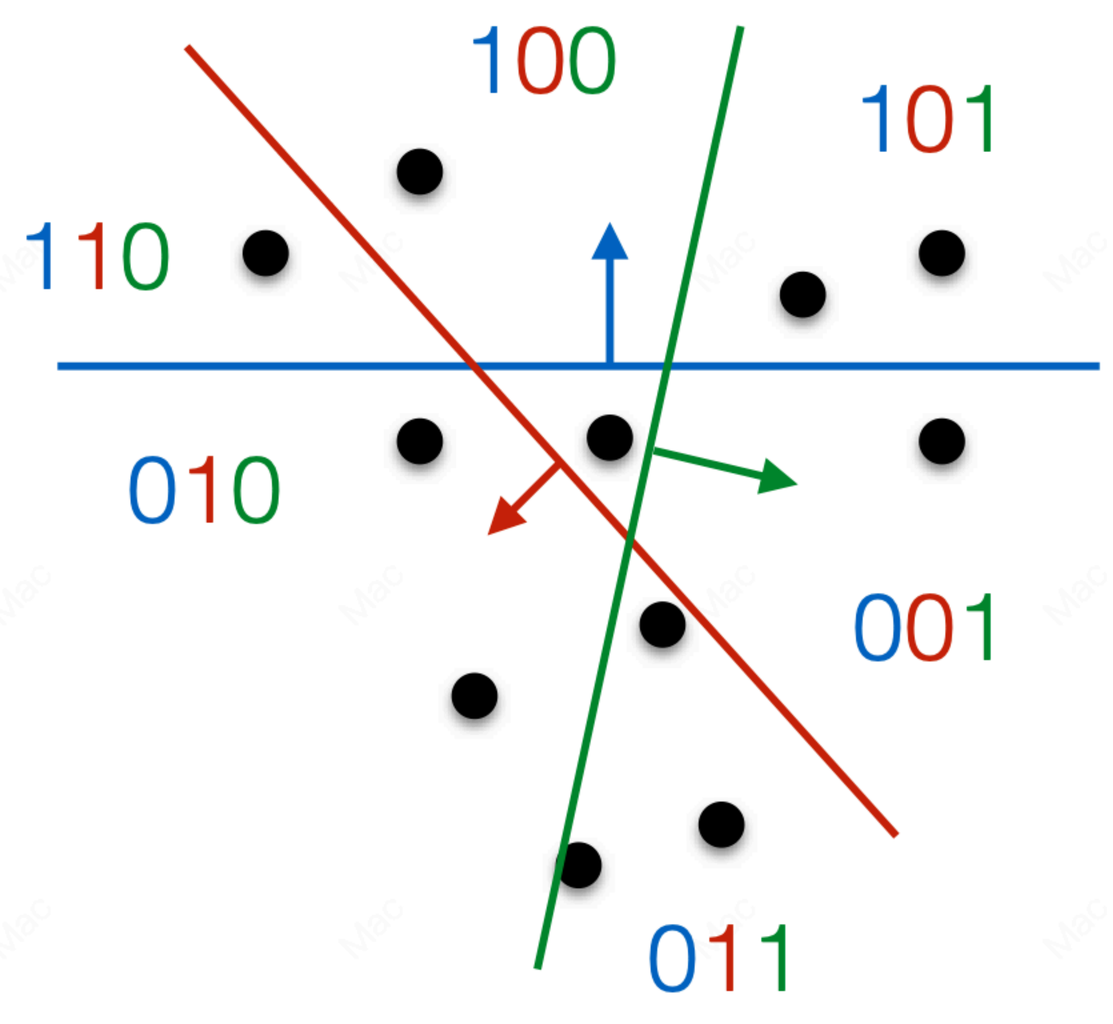
$$\mathcal{L}(\theta, q) = \sum_{d_{\text{pos}} \in D_{\text{pos}}} \sum_{d_{\text{neg}} \in D_{\text{neg}}} \max(0, s(q, d_{\text{neg}}; \theta) - s(q, d_{\text{pos}}; \theta))$$

Optimize so that the similarity between q and d_{neg} is smaller than that between q and d_{pos}

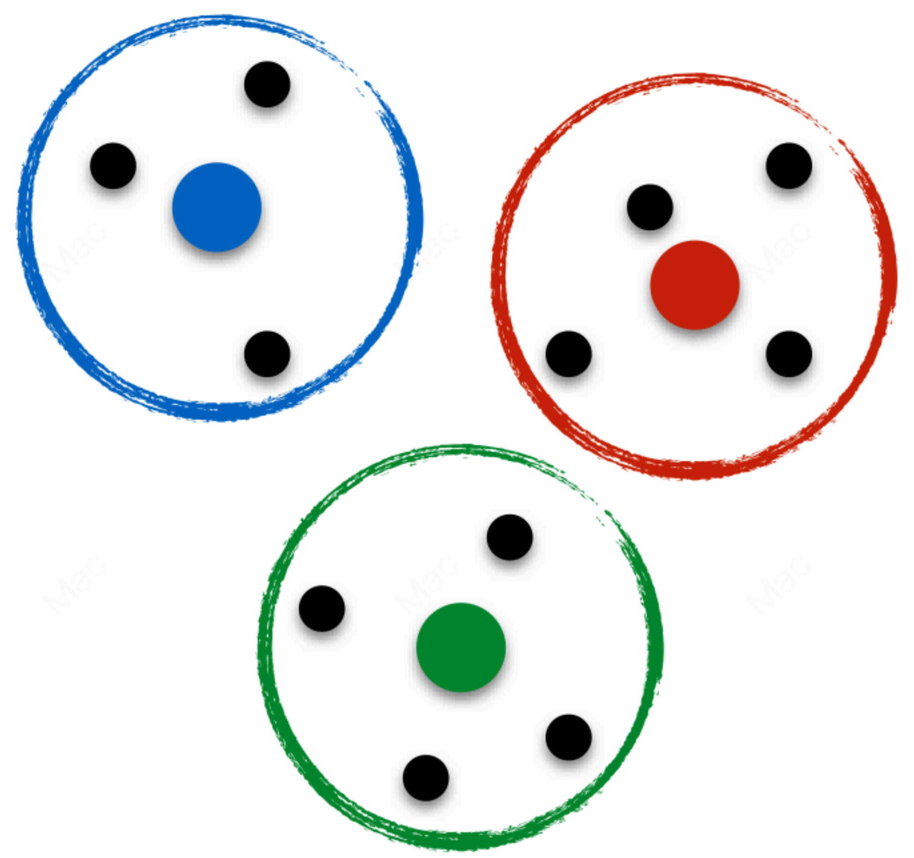
Approximate Nearest Neighbo Search

- Methods to retrieve embeddings in sub-linear time

Locality sensitive hashing: make partitions in continuous space, use like inverted index



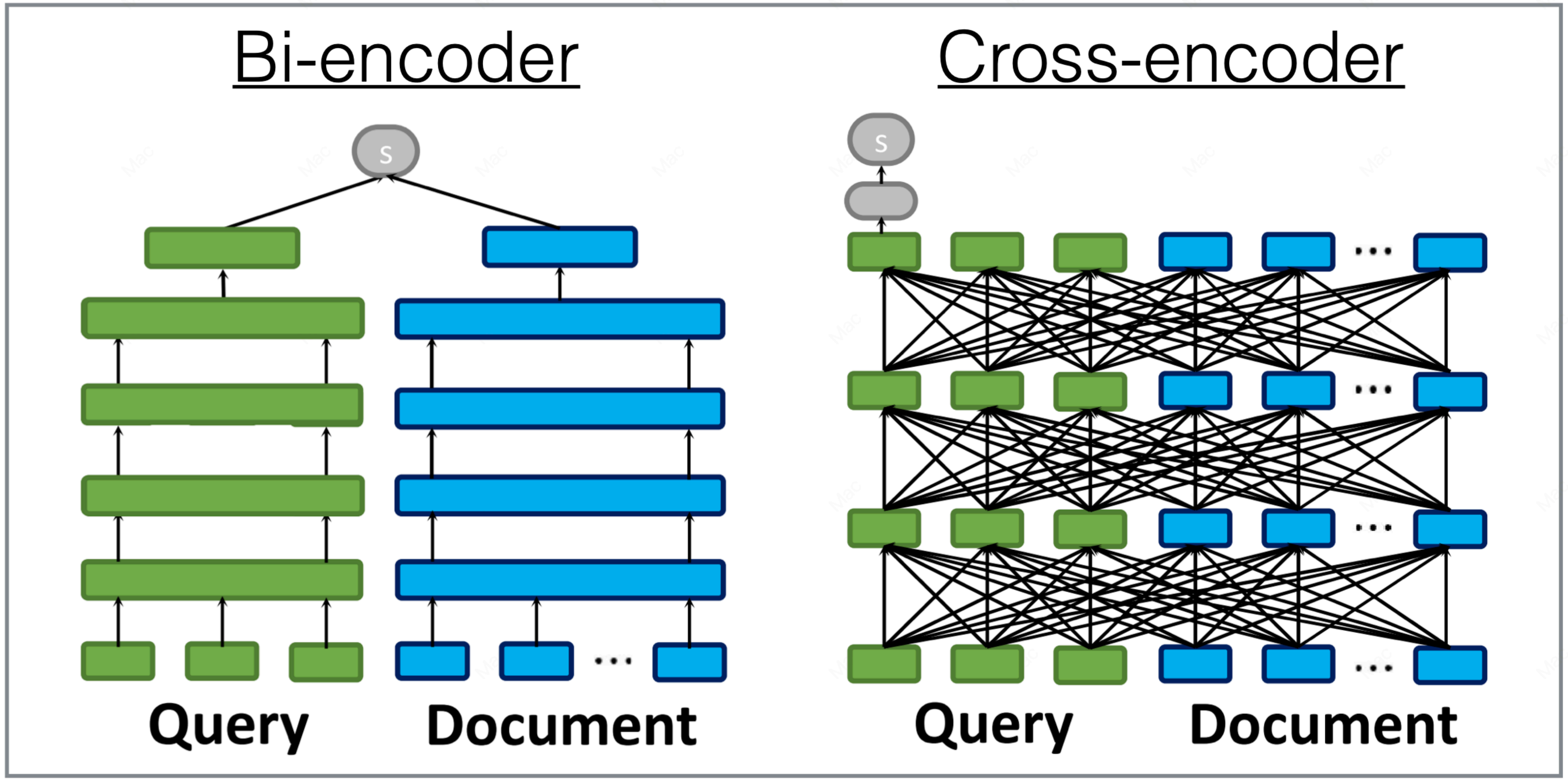
Graph-based search: create "hubs" and search from there



- Software: FAISS, ChromaDB

Cross-encoder Reranking

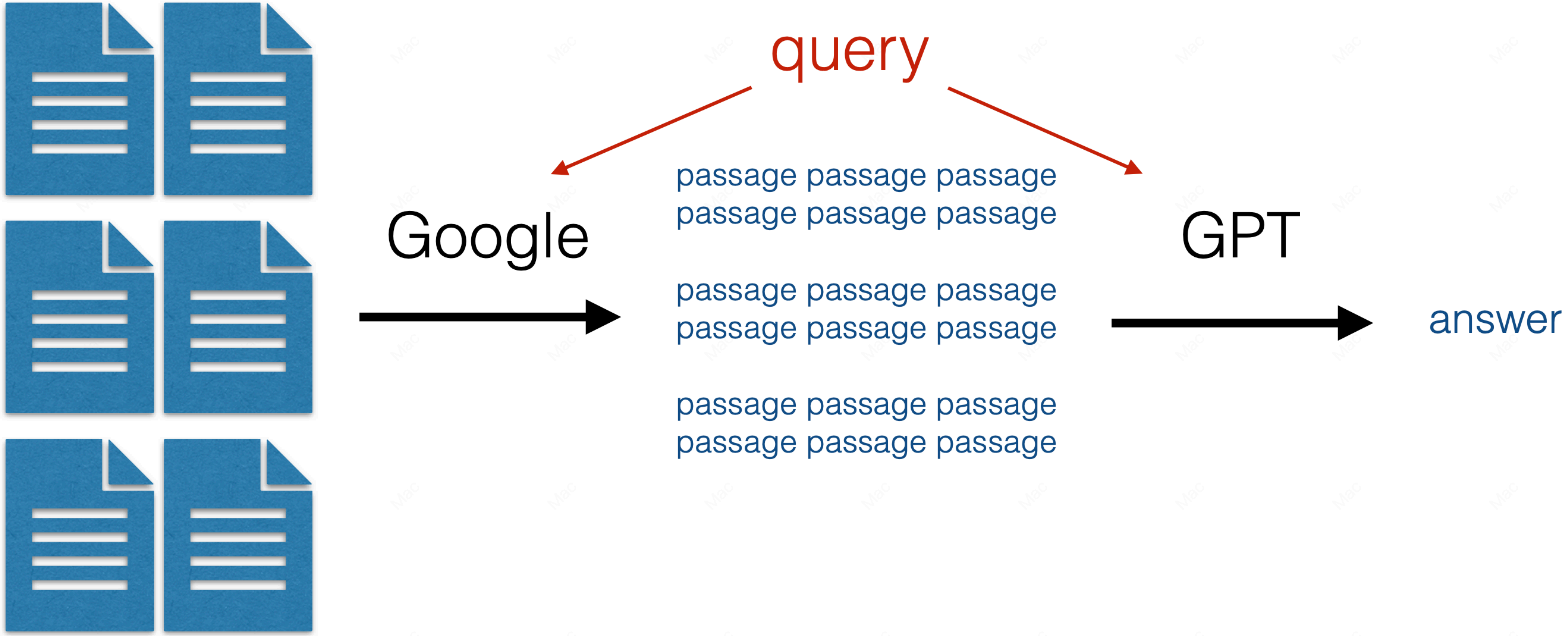
- Jointly encode both queries and documents using neural model (Nogueira et al. 2019)



can only be used on small number of candidates

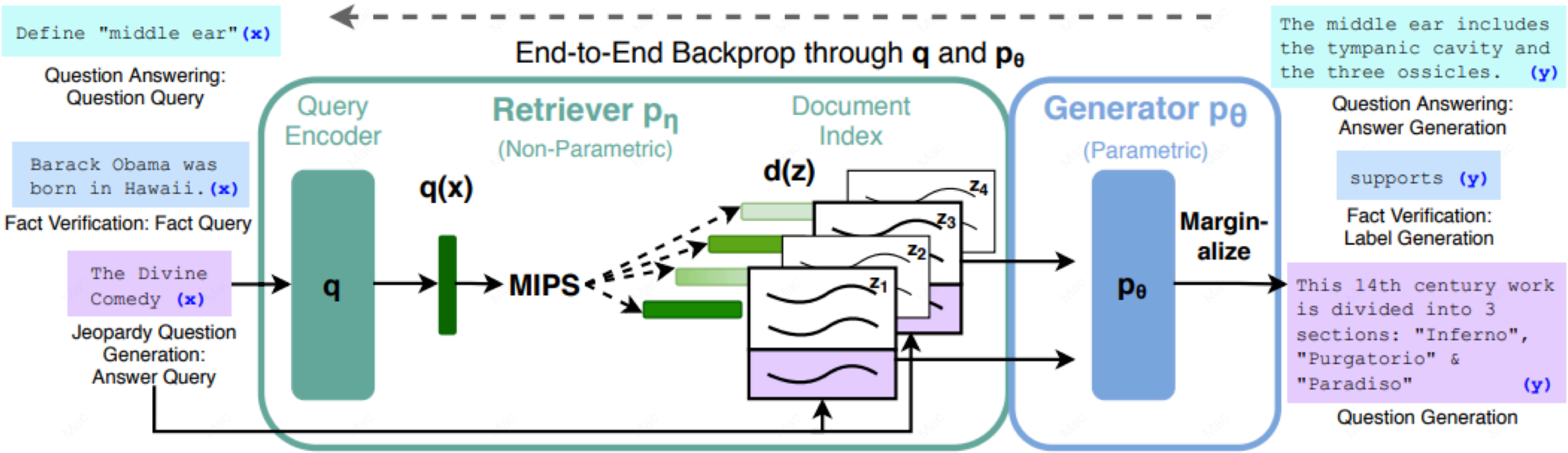
Retriever-Reader Models

Use an out-of-the-box retriever and out-of-the-box reader



Passages are concatenated to the context

Retriever+Generator End-to-End Training ("RAG")



- Train the retriever and reader to improve accuracy
- **Reader:** Maximize generation likelihood given single retrieved document
- **Retriever:** Maximize overall likelihood by optimizing mixture weights over documents

Lewis et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. 2020

End-to-End Training Equations

- Generation is a mixture model: pick a document, generate from the document

$$P_{\text{RAG}}(y|x) \approx \prod_i \sum_{z \in \text{top-k}(p(\cdot|x))} \underbrace{p_\eta(z|x)}_{\text{Retriever}} \underbrace{p_\theta(y_i|x, z, y_{1:i-1})}_{\text{Generator}}$$

- Probability of the retriever is based on embeddings

$$p_\eta(z|x) \propto \exp(\mathbf{d}(z)^\top \mathbf{q}(x)) \quad \mathbf{d}(z) = \text{enc}_d(z), \quad \mathbf{q}(x) = \text{enc}_q(x)$$

- Adjusts retriever to give higher similarities helpful docs

Issue: search index becomes stale → can only train q(x)

Lewis et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. 2020

When Do We Retrieve?

- **Once, at the beginning of generation**
  - Default method used by most systems (Lewis et al. 2020)
- **Several times during generation, as necessary**
  - Generate a search token (Schick et al. 2023)
  - Search when the model is uncertain (Jiang et al. 2023)
- **Every token**
  - Find similar final embeddings (Khandelwal et al. 2019)
  - Approximate attention with nearest neighbors (Bertsch et al. 2023)

Lewis et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. 2020

Schick et al. Toolformer: Language Models Can Teach Themselves to Use Tools. 2023

Jiang et al. Active Retrieval Augmented Generation. 2023

Khandelwal et al. Generalization through Memorization: Nearest Neighbor Language Models. 2019

Bertsch et al. Unlimiformer: Long-Range Transformers with Unlimited Length Input. 2023

Triggering Retrieval w/ Tokens (Agentic)

- Toolformer (Schick et al. 2023) generates tokens that trigger retrieval (or other tools)
- Training is done in an iterative manner - generate and identify successful retrievals

The New England Journal of Medicine is a registered trademark of **[QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society]** the MMS.

Out of 1400 participants, 400 (or **[Calculator(400 / 1400) → 0.29]** 29%) passed the test.

The name derives from "la tortuga", the Spanish word for **[MT("tortuga") → turtle]** turtle.

The Brown Act is California's law **[WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.]** that requires legislative bodies, like city councils, to hold their meetings open to the public.

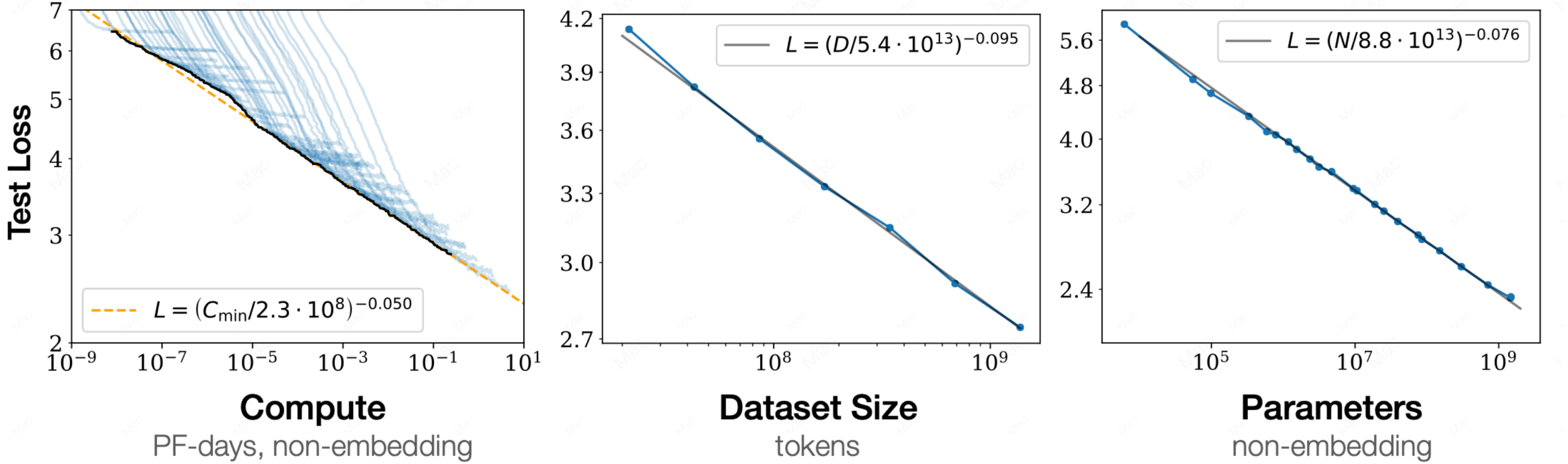
Schick et al. Toolformer: Language Models Can Teach Themselves to Use Tools. 2023

香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

Mixture-of-Experts Transformer Language Models — the way to scaling

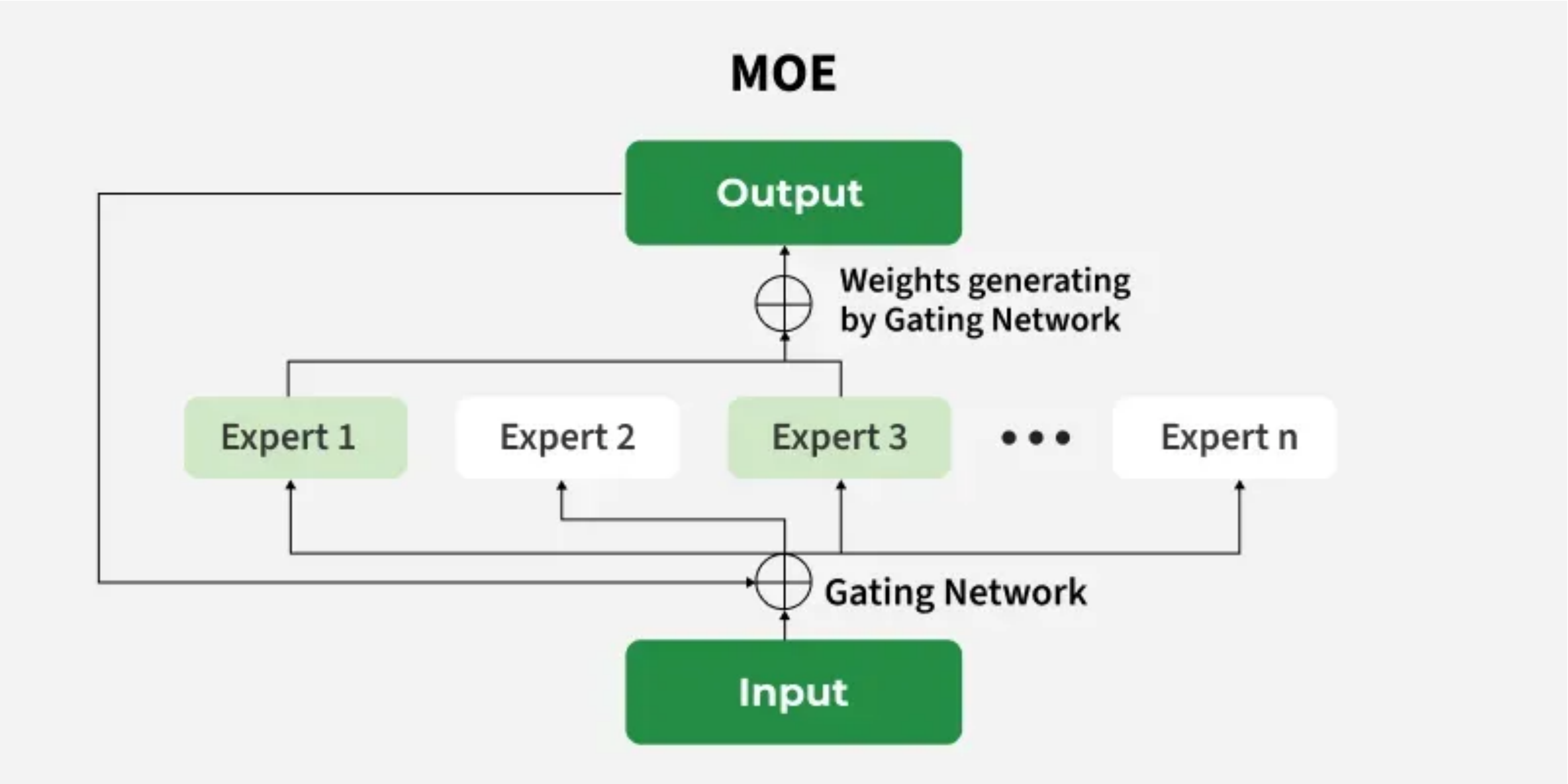
Challenges of Scaling Model Sizes

Scaling law tells us to scale up model sizes

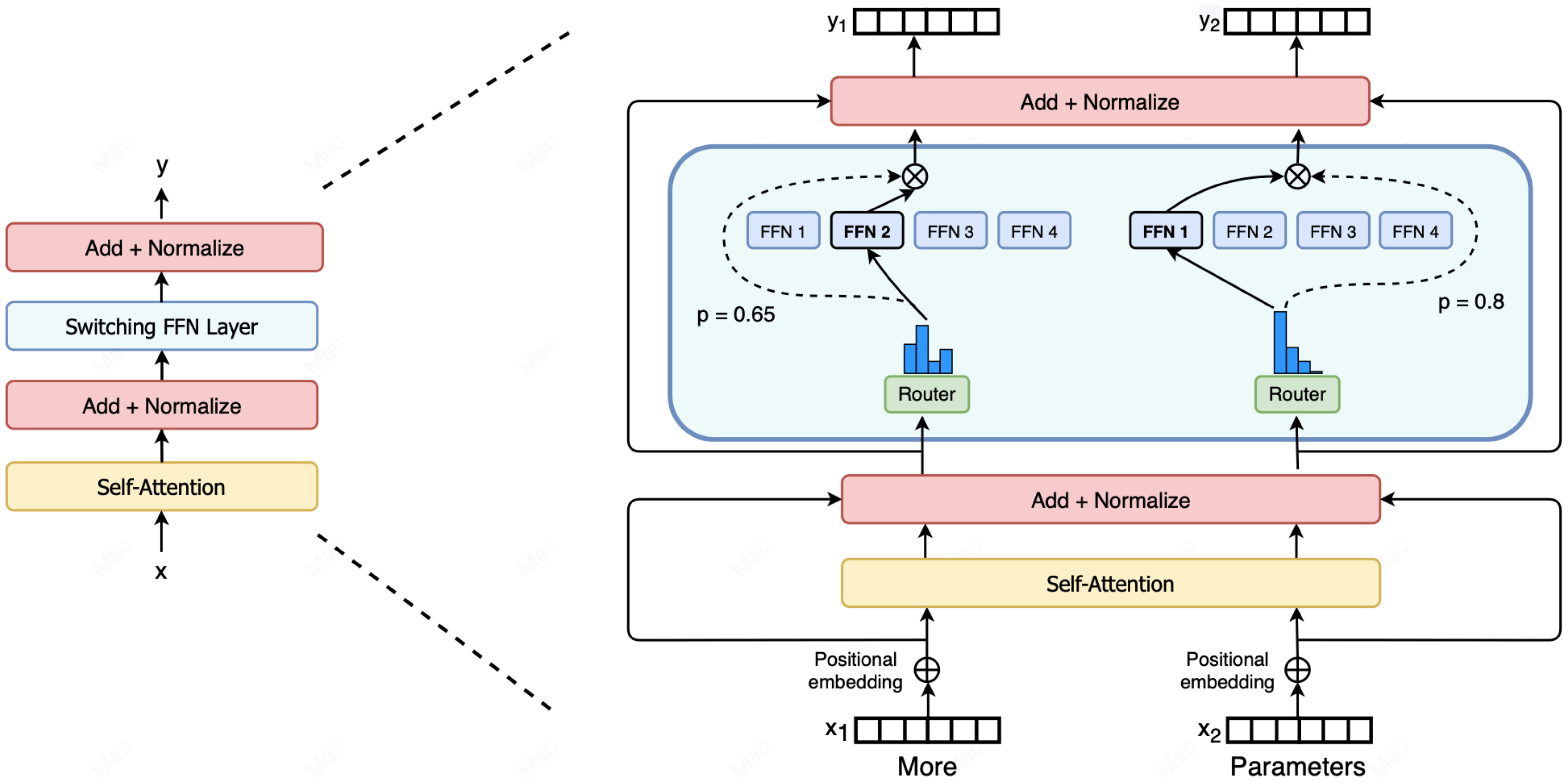


However, larger model sizes require more compute to train and causes higher latency

Mixture of Experts (MoE) in Traditional Machine Learning



MoE Transformer Language Models



Mixture of FFN Blocks

For each token at each layer, only a small fraction (e.g., 2 or 3) experts are activated by the router, thus this is also referred to as SPARSE models

Fedus et al. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. 2021

Sparse Routing

$$h(x) = W_r \cdot x$$ Logits of different experts

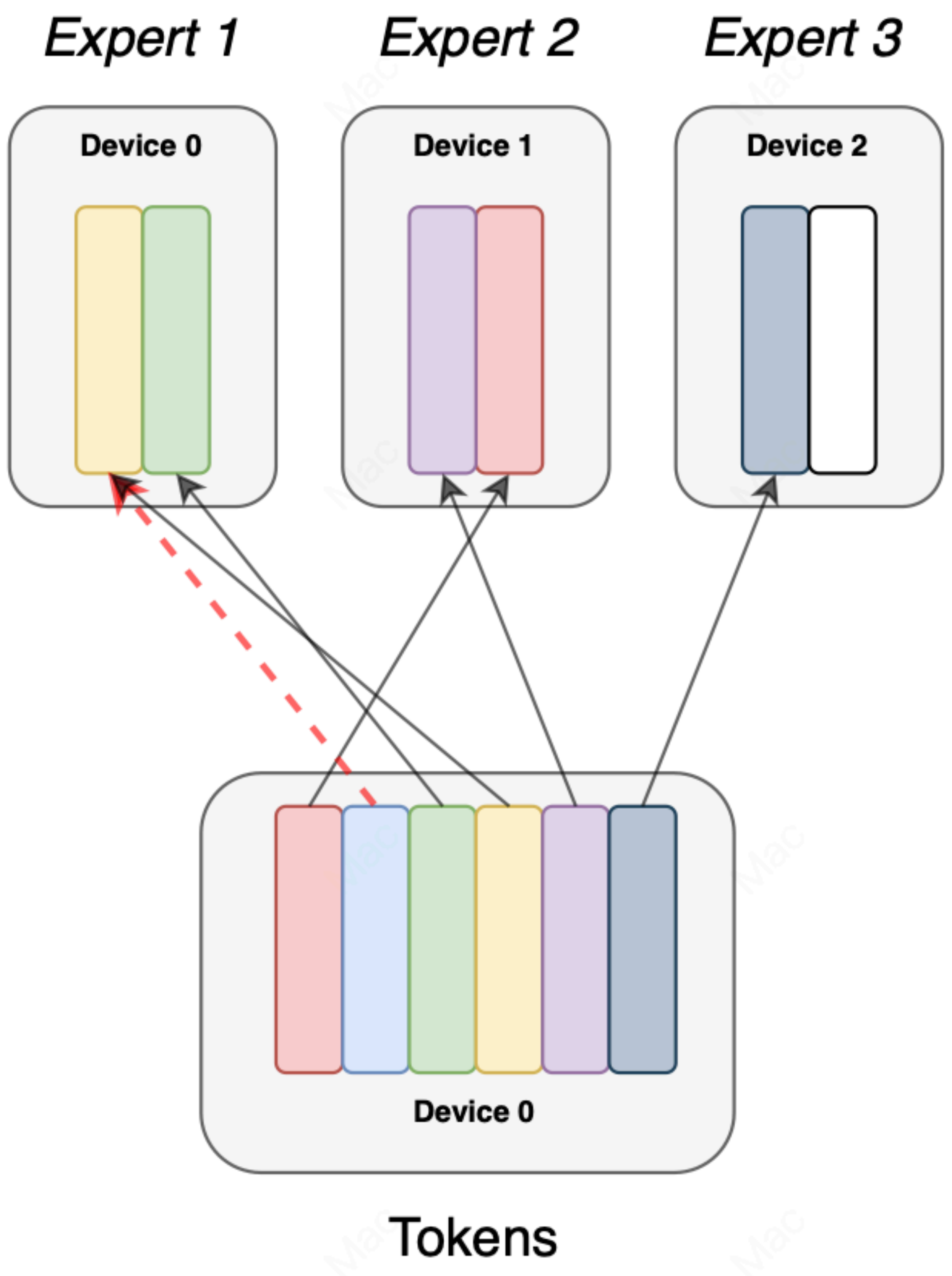
$$p_i(x) = \frac{e^{h(x)_i}}{\sum_j^N e^{h(x)_j}}.$$ Gate value, this is softmax

Sparse Routing: only top-k experts are used during both training and test time

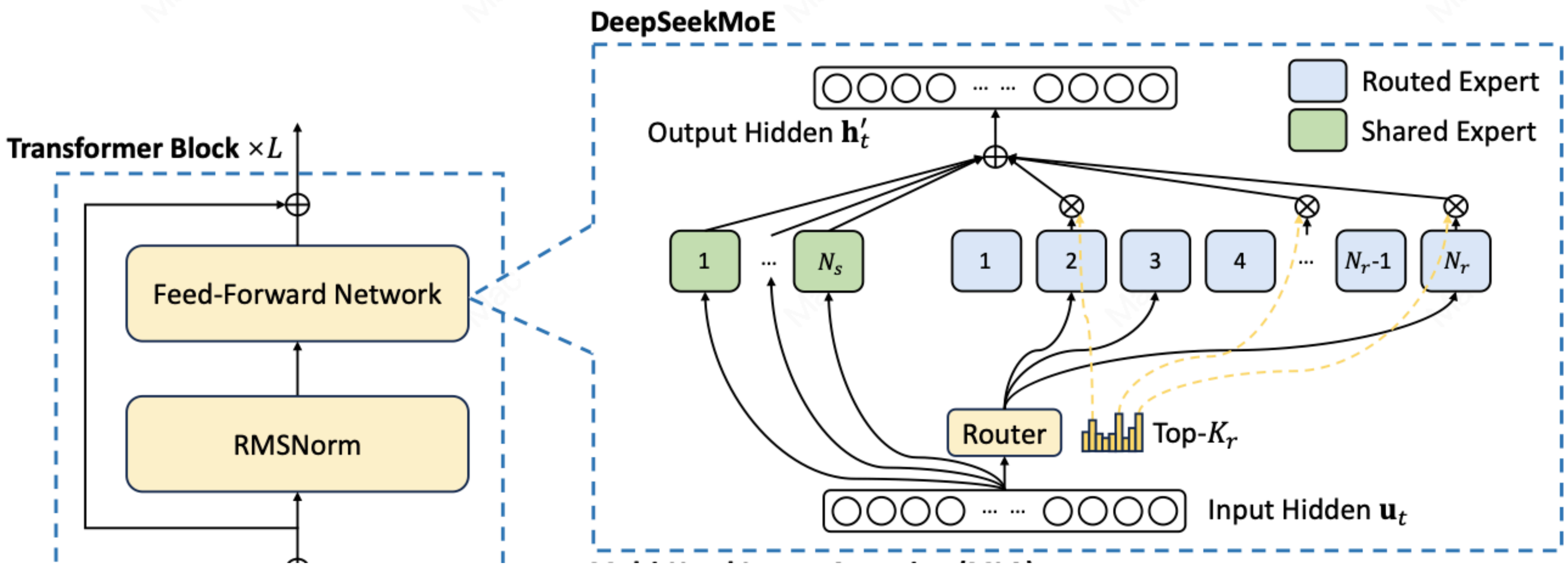
Fedus et al. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. 2021

Why MoE?

MoE models support easier model parallel across different GPUs. It can easily split models



DeepSeek MoE



Shared Experts + Routed Expert

For DS-V3, 1 shared expert + 256 routed experts, each token 8 experts are activated

DeepSeek V3 Technical Report

Thank You!