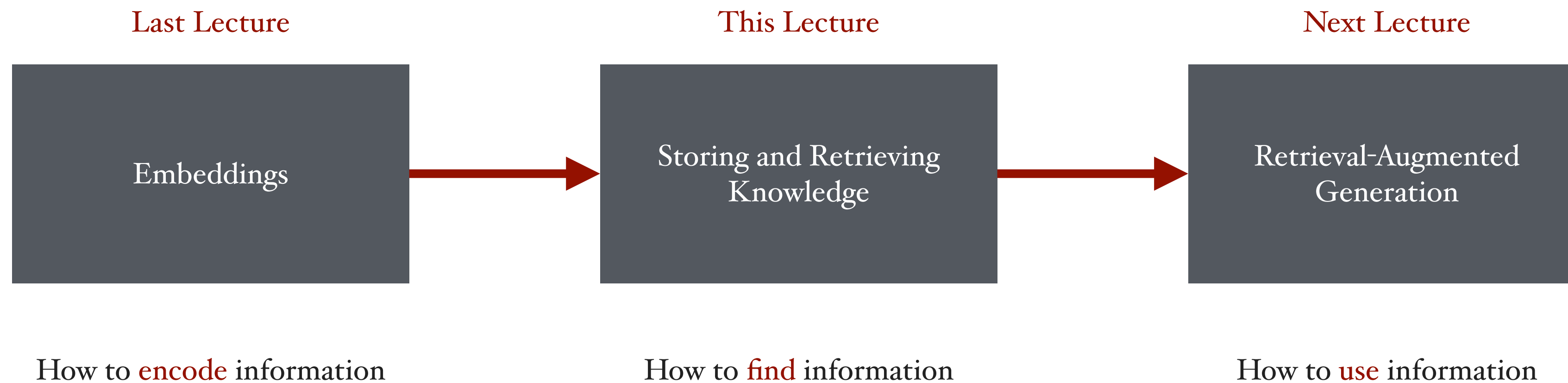


# Large Language Model Applications

Storing and Retrieving Knowledge (2)  
Retrieval Augmented Generation (1)

# From Embeddings to Knowledge Access

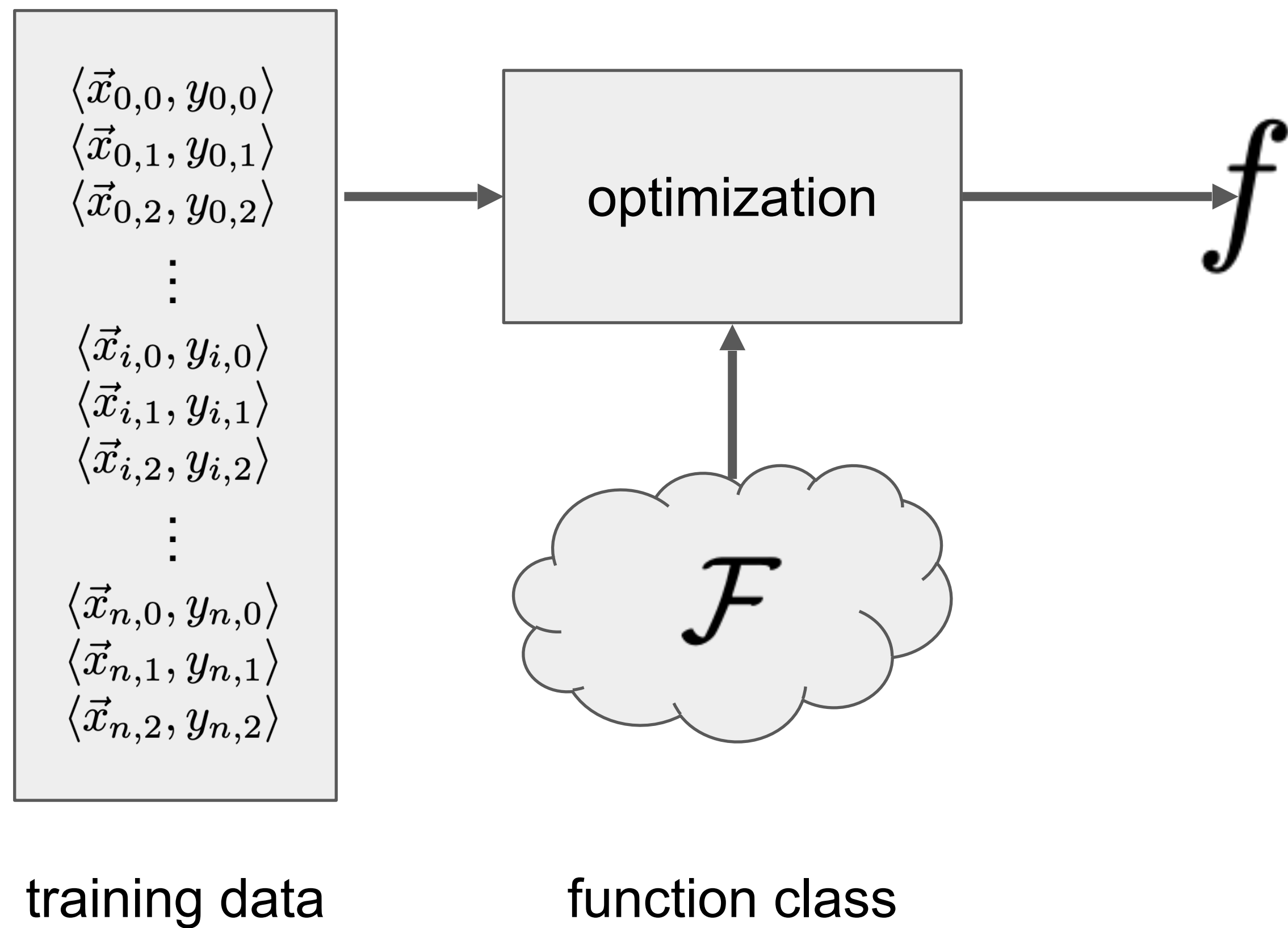


# Retrieval methods

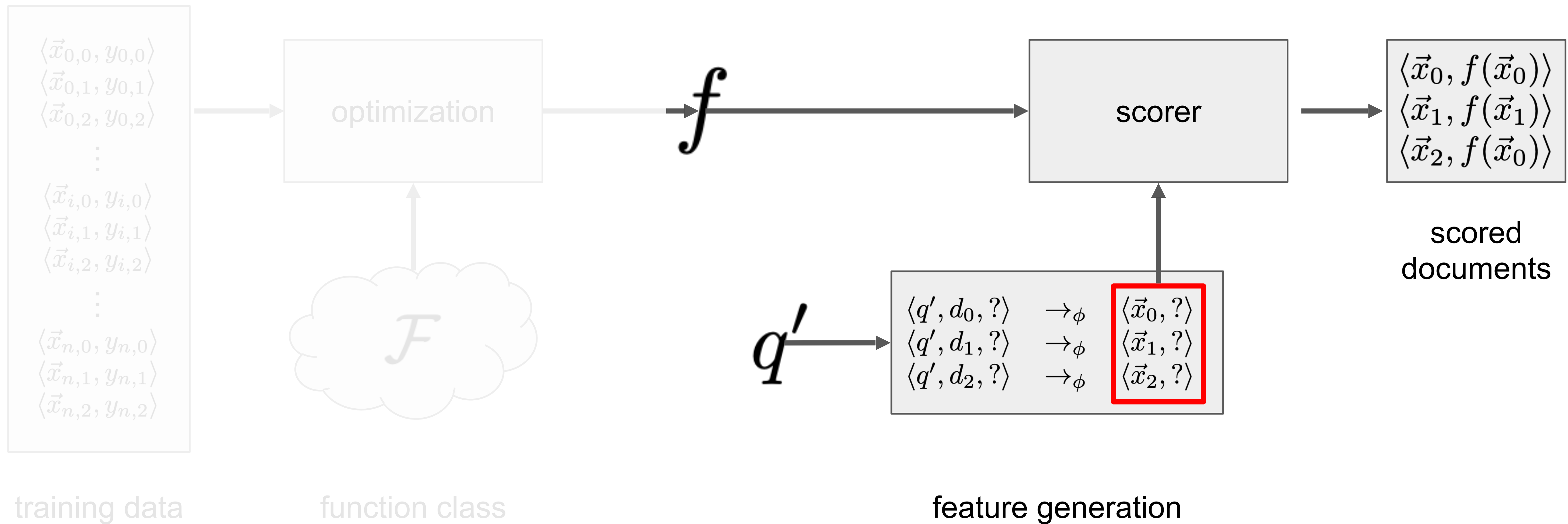
## Comparison

| Dimension     | Sparse  | Dense       | Learning to Rank                  |
|---------------|---------|-------------|-----------------------------------|
| Matching      | Lexical | Semantic    | Feature-based (hybrid)            |
| Out-of-domain | Robust  | Can degrade | Can degrade (data-dependent)      |
| Rare entities | Strong  | Weak        | Strong (if features capture them) |
| Paraphrase    | Weak    | Strong      | Strong                            |
| Training      | None    | Required    | Required (labeled data)           |

# Training (offline)



# Inference (online)



# Function class: warm up

The function class is the set of all possible relationships between feature values and the document score.

- let's start with a simple linear model with three features.
- our feature values are,
  - $x_1$  **BM25 score (float)**: match score between query and document.
  - $x_2$  **document popularity (float)**: number of clicks the document has received in the past year.
  - $x_3$  **language match (binary)**: does the page language match the user language?
- a linear model with three features would have four parameters,  $w$ : one weight for each feature and an addition 'bias' term.
  - $w_0$  bias term
  - $w_1$  BM25 weight
  - $w_2$  popularity weight
  - $w_3$  language match weight

# Function class: warm up

- given a new query  $q$  and a corpus of three documents, we generate three feature vectors

$$\vec{x}_0 = [0.90, 0.25, 1]$$

$$\vec{x}_1 = [0.80, 1.0, 0]$$

$$\vec{x}_2 = [0.20, 0.5, 0]$$

- assuming we have a weight vector,  $\vec{w} = [1, 1, 0.1, 10]$ , the document score is computed as  $f(x) = w \cdot x$
- we can then score each document using its feature vector,

$$f(\vec{x}_0) = 11.93$$

$$f(\vec{x}_1) = 1.90$$

$$f(\vec{x}_2) = 1.25$$

# Function class

- more generally, a function class bounds the types of relationships a model can learn
  - linear model: compact, fast, interpretable, less expressive (e.g., nonlinear relationships)
  - boosted decision tree: compact, fast, more interpretable, can express nonlinear relationships
  - neural network: larger, can be fast, less interpretable, extremely expressive
- appropriate function class depends on requirements of the domain
  - model size
  - latency
  - interpretability
  - effectiveness
- each function class is parameterized in such a way that a good model can be selected using a loss function

# Loss function

Loss functions provide us with a way to evaluate specific function in a function class by using labeled data. Assuming that  $y^*$  is the correct prediction,

- **binary loss:** 0 if  $y^* = f(x)$ , 1 if  $y^* \neq f(x)$ 
  - all mistakes have the same loss (all mistakes are equally important)
- **margin loss:**  $|y^* - f(x)|$  or  $(y^* - f(x))^2$ 
  - bigger mistakes have a higher loss (some mistakes are more serious than others)
- there are many types of loss function (not our focus today)

# Pointwise loss

- pointwise learning to rank: the function predicts the document label (i.e., relevance) based on the features.
- intuitive modeling problem
- interpretable prediction (i.e., relevance) on the scale provided by the system designer.

# Pointwise loss

Predicting numeric labels is difficult because *numeric labels are arbitrary*

- during *training*, different mappings from relevance categories to numeric values cause the learning algorithm to learn different models,
  - { bad, fair, good, excellent, perfect } = { 0, 1, 2, 3, 4 }; **equal differences among categories**
  - { bad, fair, good, excellent, perfect } = { 0, 0.5, 2, 4, 5 }; **preference for relevance**
  - { bad, fair, good, excellent, perfect } = { 0, 0.25, 1, 2, 10 }; **strong preference for relevance**
- what is the relative importance of different types of errors?
  - what are the right numbers to assign for bad, fair, good, excellent, perfect?
  - there isn't good theory to guide us...
  - thus, pointwise training data is inherently noisy

# Pairwise loss

**approach:** Train a model using pairs of documents

- **training data:**

$$y_i^* > y_j^* \rightarrow d_i \succ d_j \quad \text{'}d_i \text{ should be ranked above } d_j \text{'}$$

- **learned model:**

$$f(x_i) > f(x_j) \rightarrow d_i \succ d_j$$

- given a pair of documents  $d_i$  and  $d_j$ , we can treat this as a binary classification problem,

$$y_i^* > y_j^* \rightarrow +1$$

$$y_i^* < y_j^* \rightarrow -1$$

$$y_i^* = y_j^* \rightarrow 0$$

- **loss function:** 0 if a pair is ordered correctly, otherwise 1
- minimize the number of misclassified document pairs
  - focus on **preference**, not specific scores or labels.

# Pairwise loss: training data

- assume that  $y \in \{\text{bad, fair, good, excellent, perfect}\}$ .
  - labels are *ordinal*
  - we prefer perfect documents to excellent documents
  - we prefer perfect documents to good documents
  - ...
- given a set of  $n$  judged documents for a query, we can derive a set of  $O(n^2)$  preferences
  - for each  $d_i$  such that  $y_i = \text{perfect}$ ,  $d_i \succ j$  if  $y_j \in \{\text{bad, fair, good, excellent}\}$ .
  - for each  $d_i$  such that  $y_i = \text{excellent}$ ,  $d_i \succ j$  if  $y_j \in \{\text{bad, fair, good}\}$ .
  - for each  $d_i$  such that  $y_i = \text{good}$ ,  $d_i \succ j$  if  $y_j \in \{\text{bad, fair}\}$ .
  - for each  $d_i$  such that  $y_i = \text{fair}$ ,  $d_i \succ j$  if  $y_j \in \{\text{bad}\}$ .

# Pairwise loss

how is preference data used to train a model?

- tell the learning algorithm **which** document is better
- do not tell it the true rating of the documents
  - know what the ranking order should be
  - don't know what the scores should be
- Don't train the algorithm to predict a number that we might choose arbitrarily
- algorithm is free to predict any scores that satisfy the ranking order
- goal is to minimize the number of prediction mistakes

# Pairwise loss

- when we optimize the binary classifier  $f$  with  $y \in \{-1, 1\}$  and features  $x_i - x_j$ , we can use  $f$  to produce scores  $f(x_i)$  and  $f(x_j)$  to rank documents.
  - pairwise *training*  $\neq$  pairwise *ranking*
- can generalize from linear model to much more complex pairwise models,
  - RankNet (Burges, et al., 2005)
  - Frank (Tsai, et al., 2007)
  - RankBoost (Freund, et al., 2003)
  - Ranking SVM (Herbrich, et al., 2000; Joachims, 2002)
  - MHR
  - IR-SVM

# Listwise loss

**approach:** Train a model using a list of documents

- **training data:**

$$d_1 \succ d_2 \succ \dots \succ d_n \quad \text{'target ranking for a query'}$$

- **learned model:**

$$f(x_1) > f(x_2) > \dots > f(x_n) \rightarrow d_1 \succ d_2 \succ \dots \succ d_n$$

- the value of a particular ranking is defined by some metric (e.g., NDCG, MAP)
- goal is to maximize the value of the metric
  - directly align the model with the ranking target

# Listwise loss: training data

- assume that  $y \in \{\text{bad, fair, good, excellent, perfect}\}$ .
  - labels are *ordinal*
  - we prefer perfect documents to excellent documents
  - we prefer perfect documents to good documents
  - ...
- given a set of  $n$  judged documents for a query, we can a target ranking by ranking documents in decreasing rating.

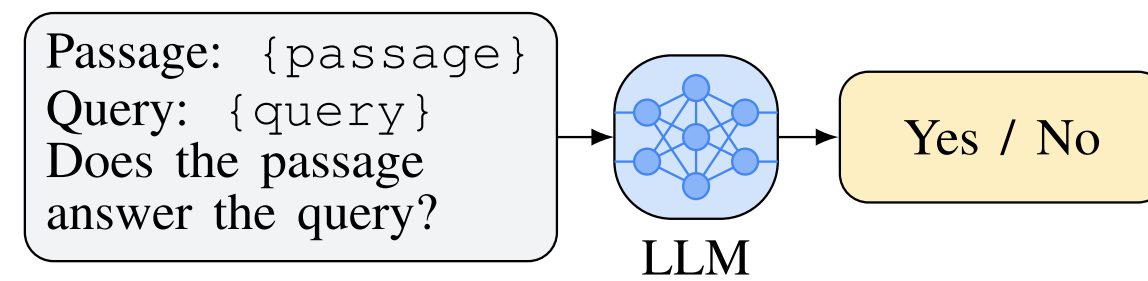
# Listwise loss: training data

- Directly optimizing some metrics is hard
  - Popular metrics (e.g., NDCG@n) are not continuous, nor convex
  - Very hard for optimization algorithms
- Two common strategies in listwise approaches:
  - Find an easier metric that is intuitive and easy to optimize (e.g. likelihood of ‘best’ rankings in training data).
  - Directly optimize ranking evaluation metrics, with approximation

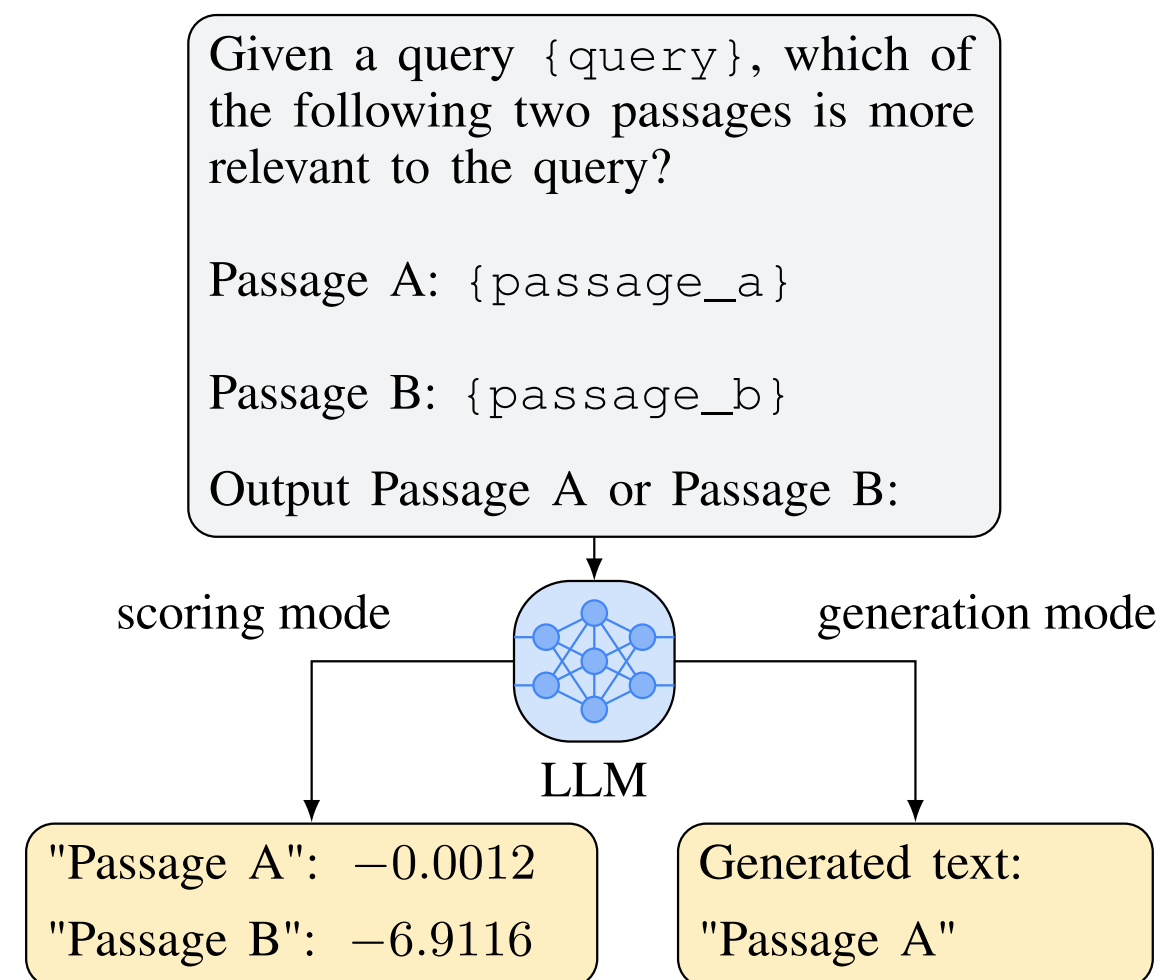
# LLM Reranking

## Pairwise Ranking Prompting (PRP)

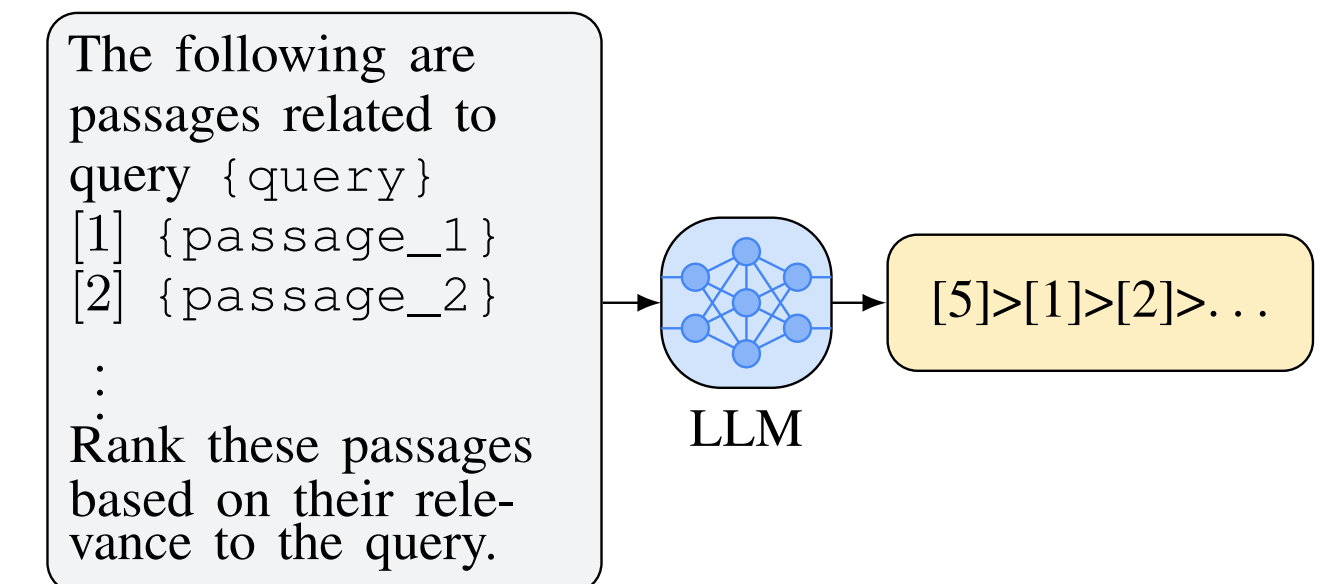
pointwise



pairwise



listwise



Instead of training a model, can we use LLMs as the rerankers.

# LLM Reranking

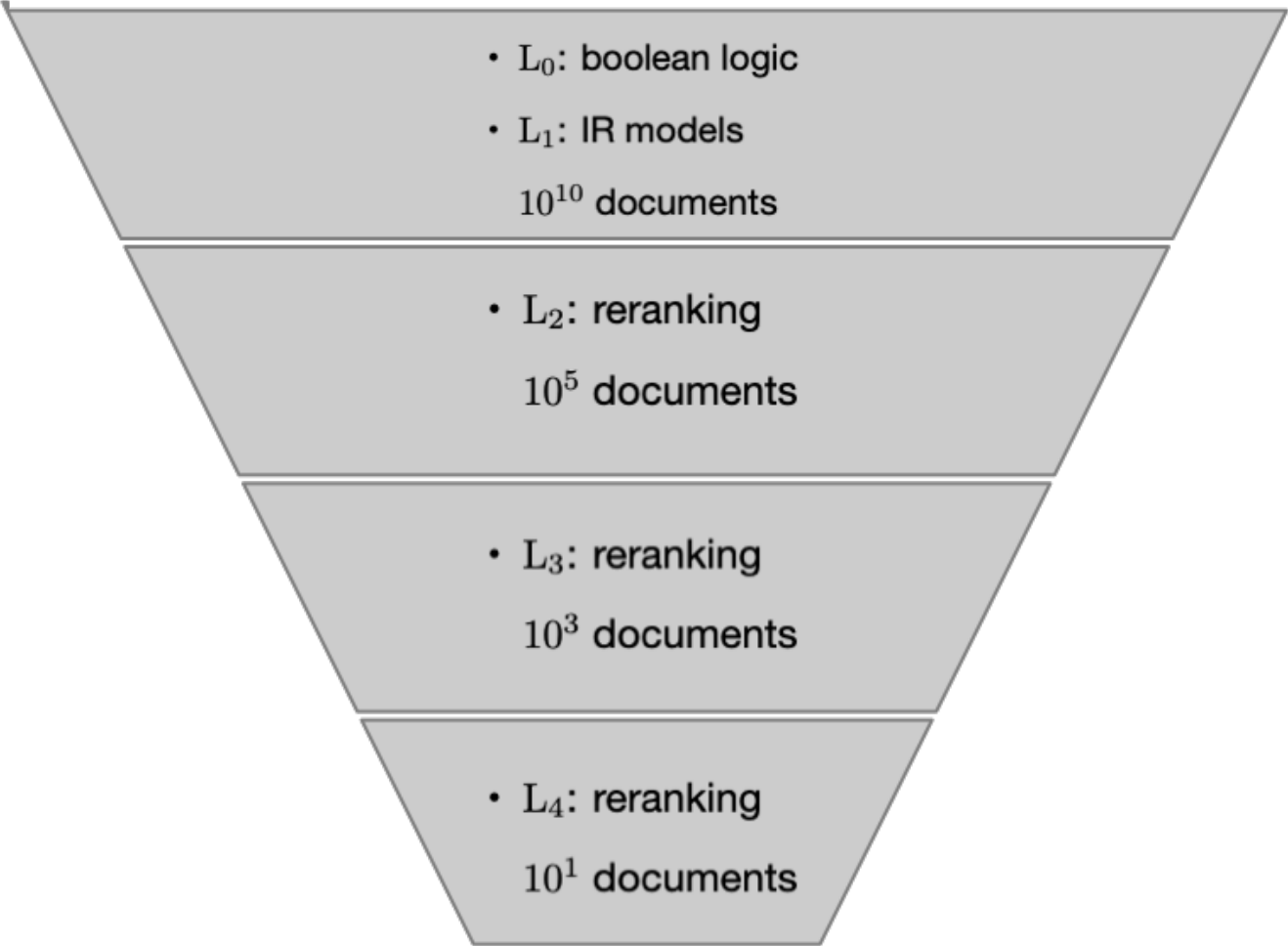
## Pairwise Ranking Prompting (PRP)

| Method                   | LLM      | Size | Covid        | Touche       | DBPedia      | SciFact      | Signal | News         | Robust04 | Avg          |
|--------------------------|----------|------|--------------|--------------|--------------|--------------|--------|--------------|----------|--------------|
| BM25                     | NA       | NA   | 59.47        | <b>44.22</b> | 31.80        | 67.89        | 33.05  | 39.52        | 40.70    | 45.23        |
| Supervised Methods       |          |      |              |              |              |              |        |              |          |              |
| monoBERT                 | BERT     | 340M | 70.01        | 31.75        | 41.87        | 71.36        | 31.44  | 44.62        | 49.35    | 48.63        |
| monoT5                   | T5       | 220M | 78.34        | 30.82        | 42.42        | 73.40        | 31.67  | 46.83        | 51.72    | 50.74        |
| monoT5                   | T5       | 3B   | 80.71        | 32.41        | 44.45        | <b>76.57</b> | 32.55  | 48.49        | 56.71    | 53.13        |
| RankT5                   | T5       | 3B   | 82.00        | 37.62        | 44.19        | 76.86        | 31.80  | 48.15        | 52.76    | 53.34        |
| TART-Rerank              | T5       | 3B   | 75.10        | 27.46        | 42.53        | 74.84        | 25.84  | 40.01        | 50.75    | 48.08        |
| Unsupervised LLM Methods |          |      |              |              |              |              |        |              |          |              |
| PRP-Allpair              | FLAN-UL2 | 20B  | <b>82.30</b> | 29.71        | 45.94        | 75.70        | 32.26  | 48.04        | 55.49    | 52.78        |
| PRP-Sorting              | FLAN-UL2 | 20B  | 82.29        | 25.80        | 44.53        | 67.07        | 32.04  | 45.37        | 51.45    | 49.79        |
| PRP-Sliding-10           | FLAN-UL2 | 20B  | 79.45        | 37.89        | <b>46.47</b> | 73.33        | 35.20  | <b>49.11</b> | 53.43    | <b>53.55</b> |

# Retrieval methods

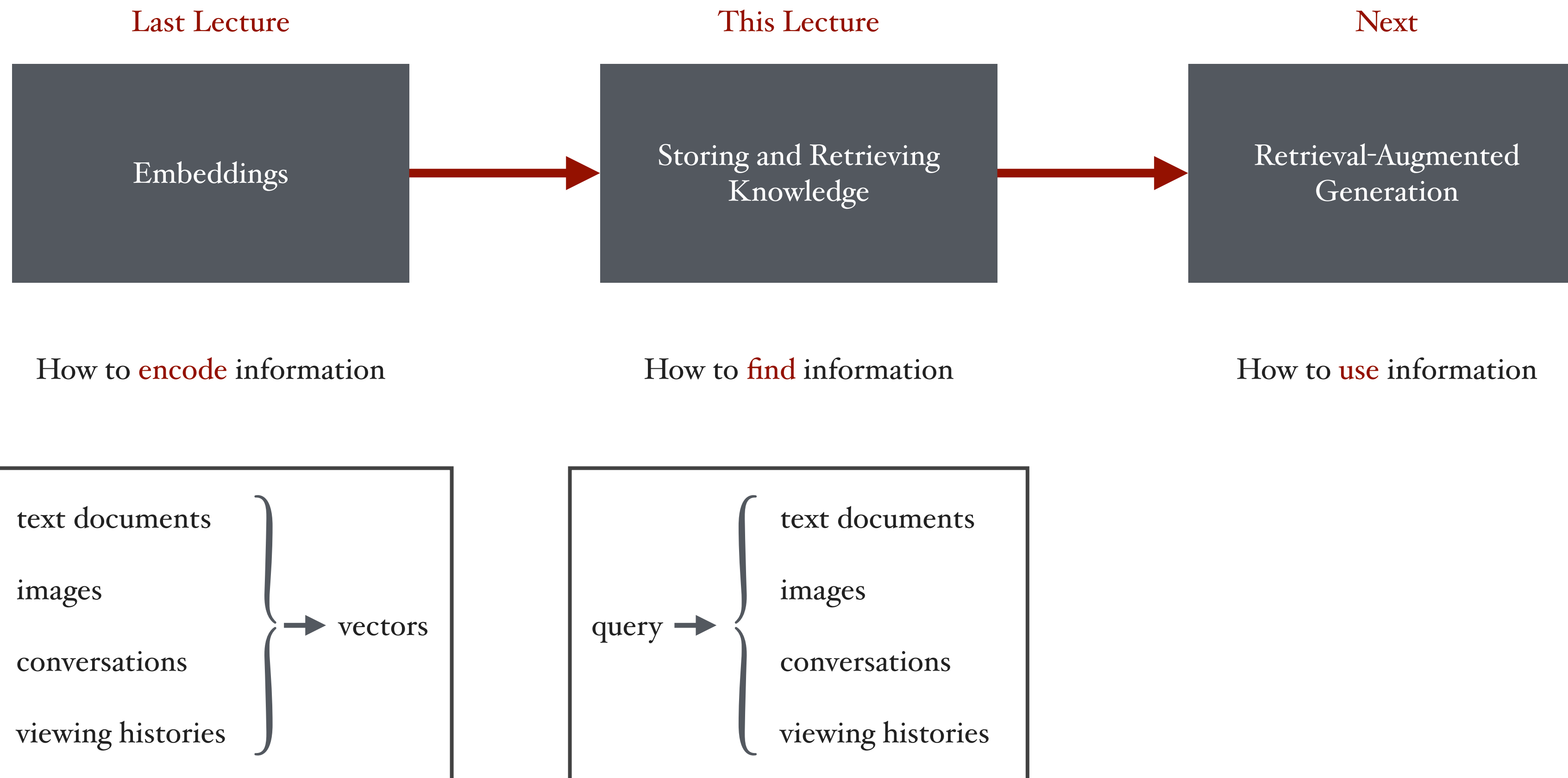
## Comparison

| Dimension       | Sparse      | Dense       | Learning to Rank                  |
|-----------------|-------------|-------------|-----------------------------------|
| Matching        | Lexical     | Semantic    | Feature-based (hybrid)            |
| Out-of-domain   | Robust      | Can degrade | Can degrade (data-dependent)      |
| Rare entities   | Strong      | Weak        | Strong (if features capture them) |
| Paraphrase      | Weak        | Strong      | Strong                            |
| Training        | None        | Required    | Required (labeled data)           |
| Data type       | Text        | Text        | Heterogeneous                     |
| Design stage    | Prototyping | Developing  | Polishing                         |
| Retrieval stage |             |             |                                   |



poll: comparing retrieval  
approaches

# From Embeddings to Knowledge Access

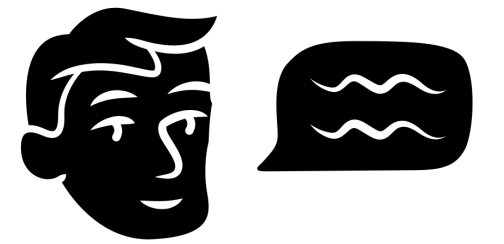


# Retrieval as a Tool

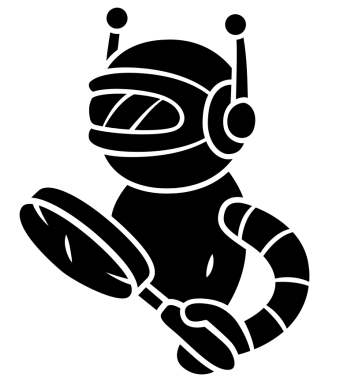
- Search engines and recommender systems are **tools** to accomplish tasks.
- People use search to find relevant knowledge to help them accomplish tasks
  - cooking dinner
  - literature review
  - shopping
  - ...
- LLMs can too!
  - answering questions
  - coding
  - ...



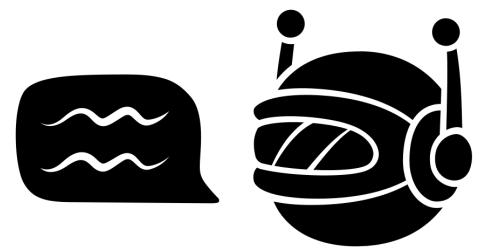
start  
task



search



finish  
task

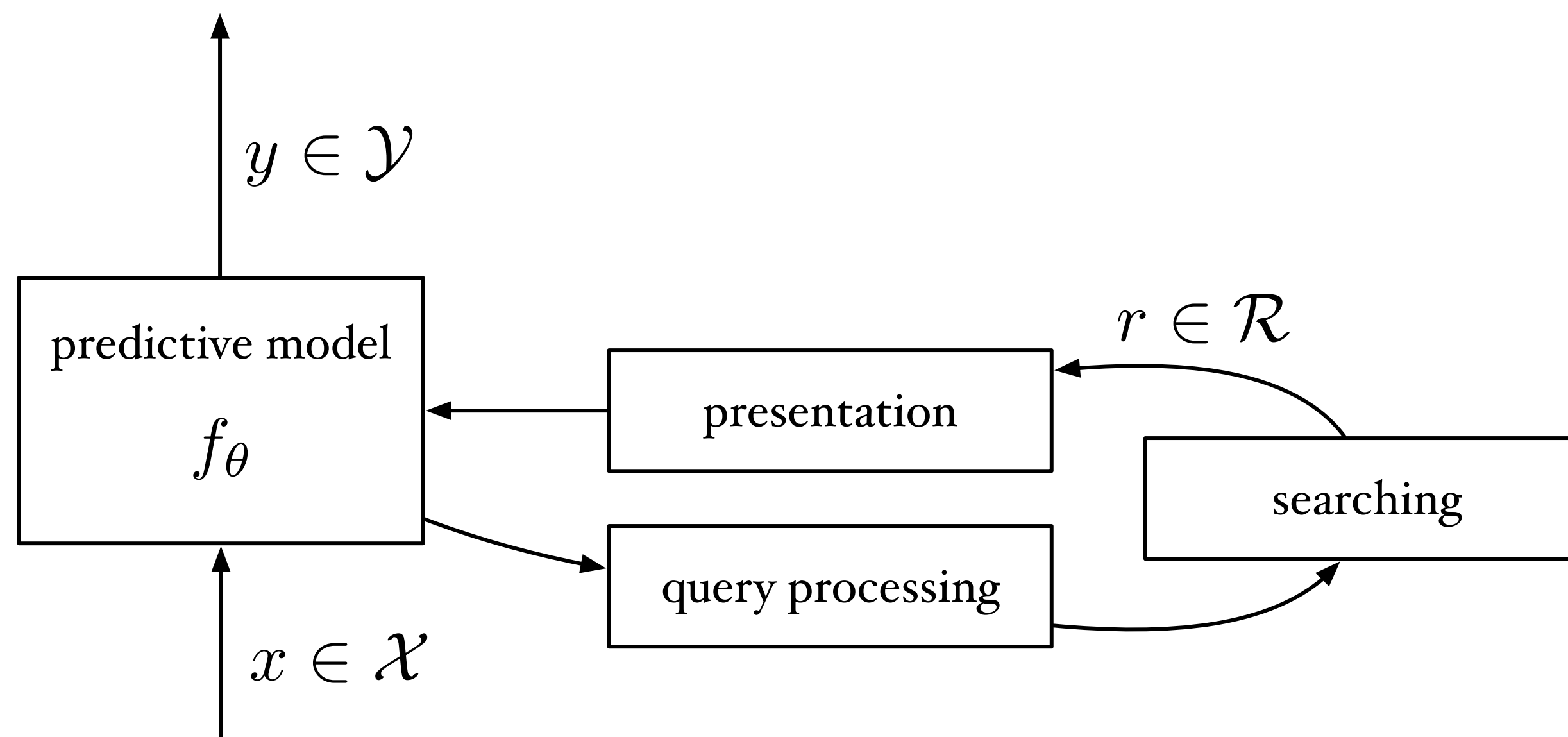


receive  
feedback



# Retrieval-enhanced machine learning

In LLMs the presented results are incorporated into the generator's prompt for the final answer



# What does retrieval provide as a tool?

## Parametric memory

### Hallucination

Confident generation of false information

### Staleness

Knowledge frozen at training time

### Unverifiability

No source for claims

### Cost of updates

Retraining is expensive

## Non-parametric memory (RAG)

### Grounds generation

Retrieval provides evidence

### Updatable corpus

No retraining needed

### Provides citations

Retrieved passages as sources

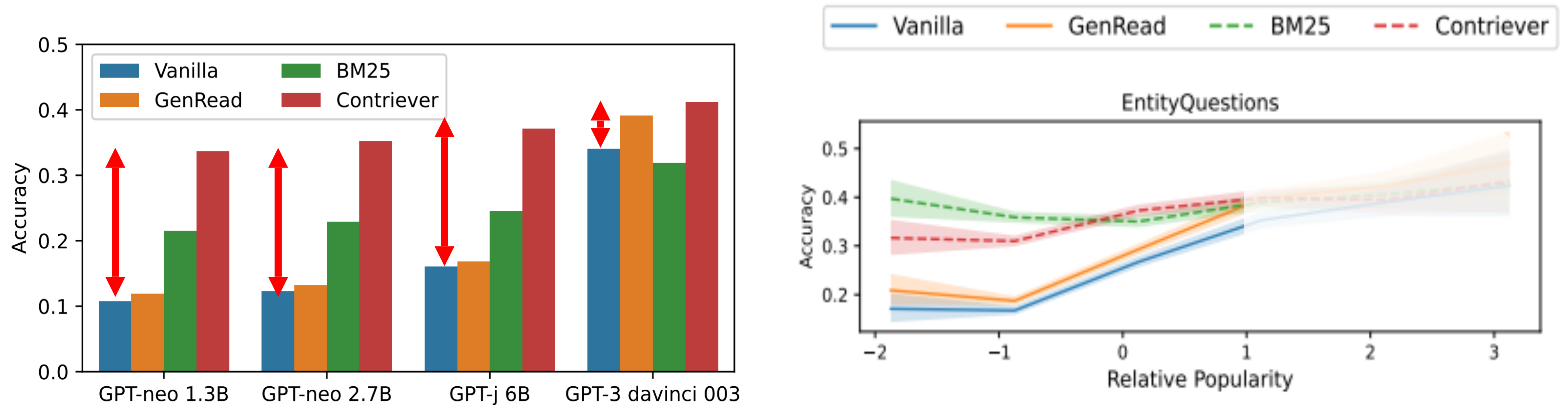
### Cost-effective

Knowledge scales independently



# Hallucinations

Retrieval-augmented LMs can reduce hallucinations, especially in long-tail knowledge





# Staleness

Parametric LMs' knowledge gets obsolete quickly & requires continuous training



The current Prime Minister of the United Kingdom is Rishi Sunak, who took office on **October 25, 2022**, following the resignation of Boris Johnson. Sunak is a member of the Conservative Party and previously served as Chancellor of the Treasury.



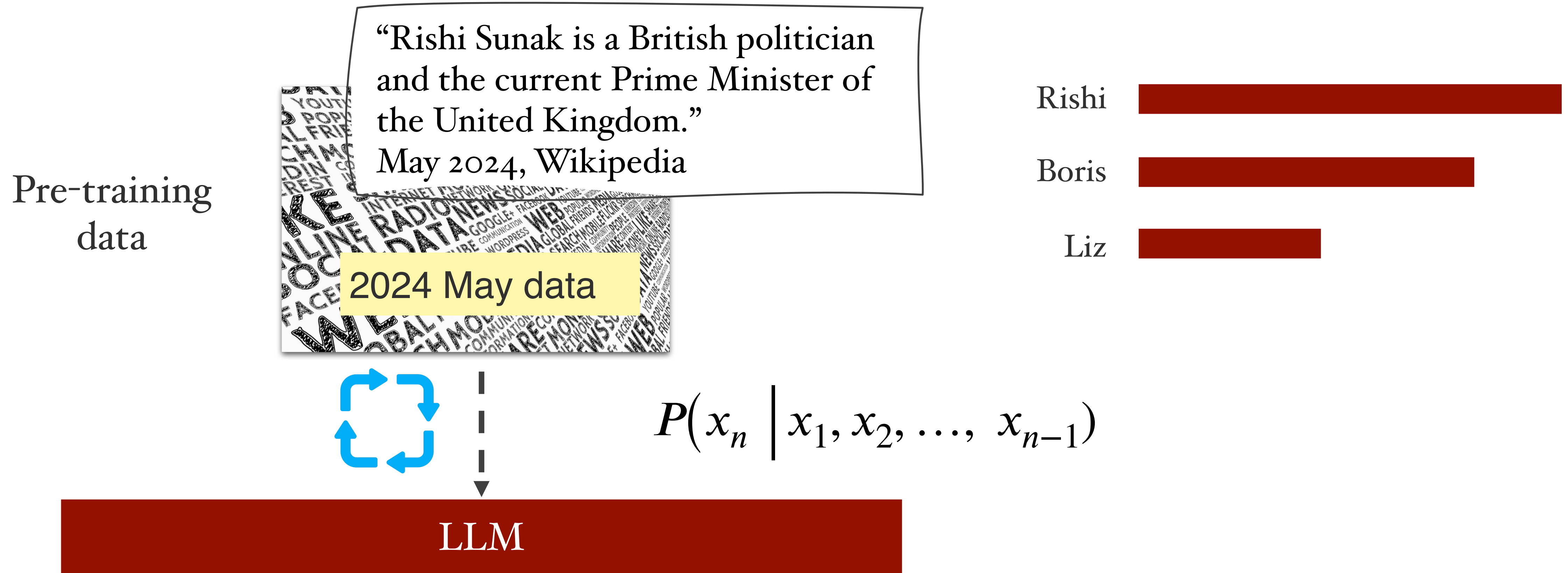
| Portrait                                                                              | Prime minister<br>Office<br>(Lifespan)                                                                   | Term of office  |                  |                  | Mandate <sup>[a]</sup> ↕ |
|---------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------|-----------------|------------------|------------------|--------------------------|
|                                                                                       |                                                                                                          | Start ↕         | End ↕            | Duration ↕       |                          |
|   | <b>Rishi Sunak</b><br>[98]<br>MP for <b>Richmond (Yorks)</b><br>(born 1980)<br><i>Premiership</i>        | 25 October 2022 | 5 July 2024      | 1 year, 255 days | —                        |
|  | <b>Keir Starmer</b><br>[99]<br>MP for <b>Holborn and St Pancras</b><br>(born 1962)<br><i>Premiership</i> | 5 July 2024     | <i>Incumbent</i> | 73 days          | 2024                     |



# Staleness

RAG can adapt without training

“The current Prime Minister UK is...”

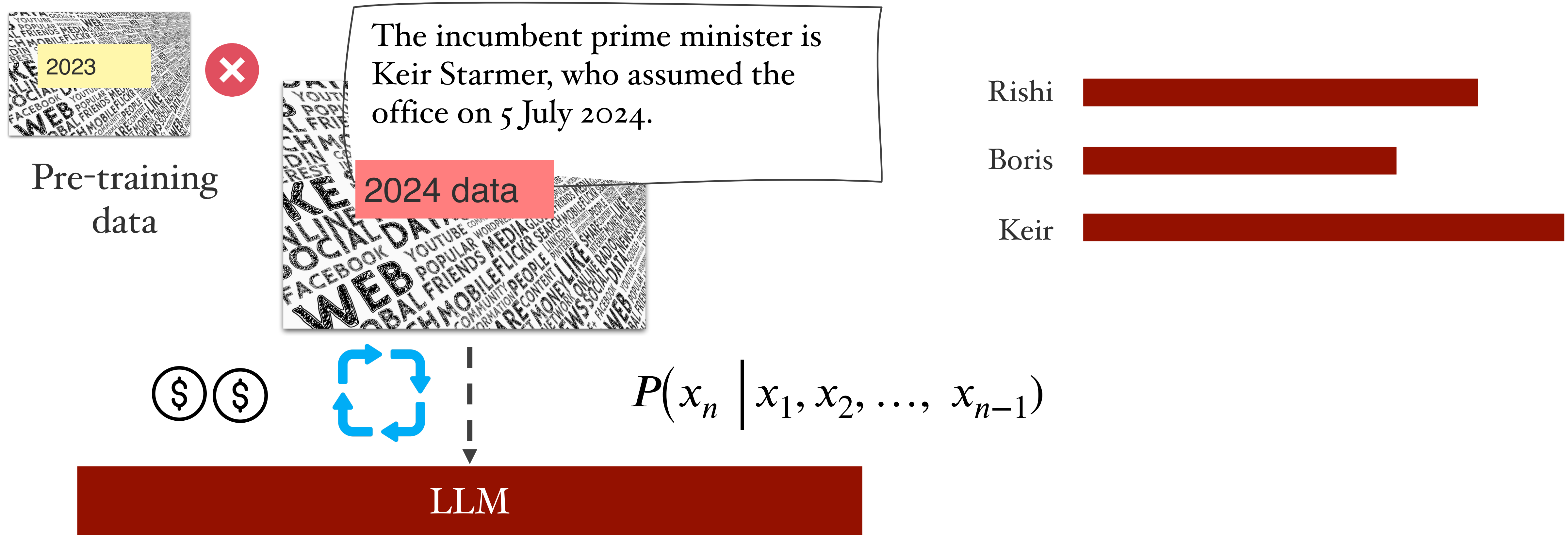




# Staleness

RAG can adapt without training

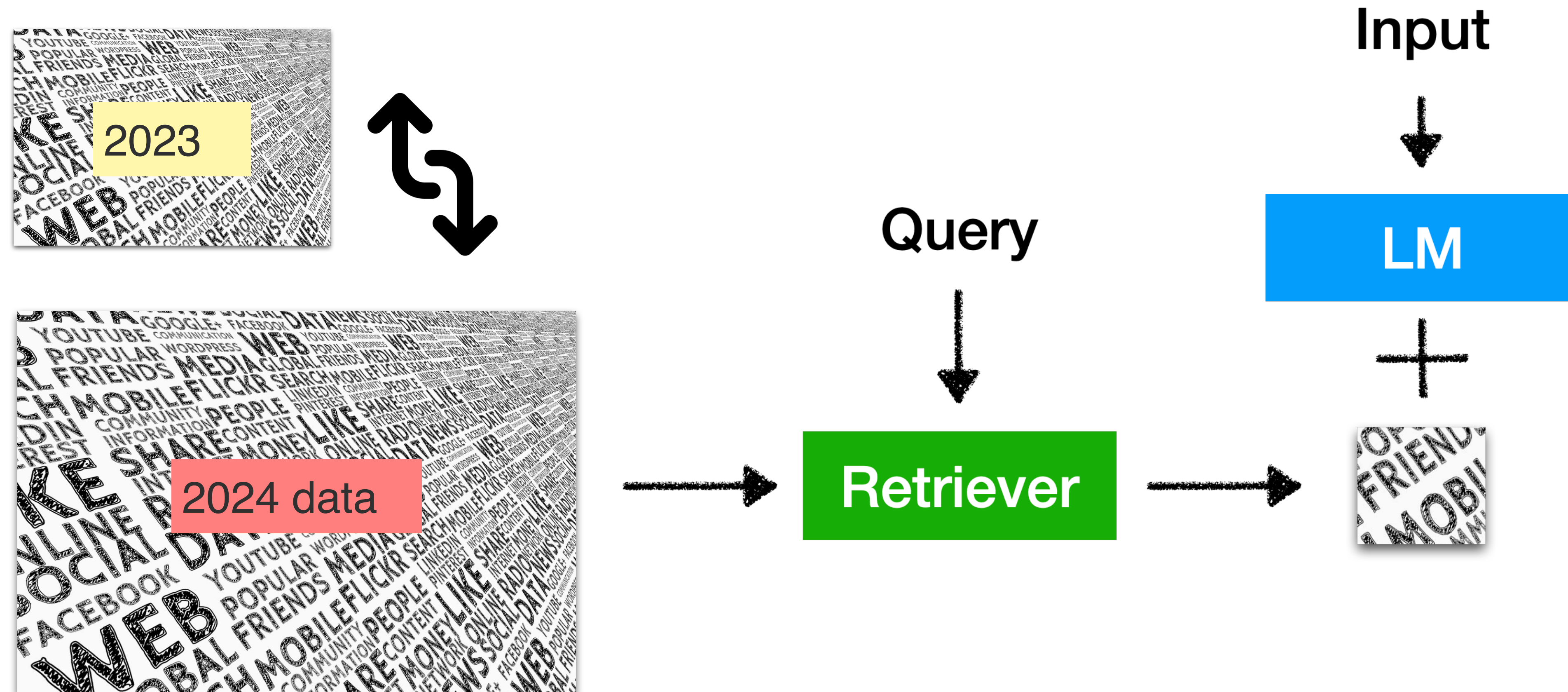
“The current Prime Minister UK is...”





# Staleness

We can easily swap datastores for retrieval-augmented LMs for new data distributions





# Unverifiability

Retrieval-augmented LMs can provide a small number of documents as attributions

Who is the current prime minister of United Kingdom?



Searched 6 sites ▾

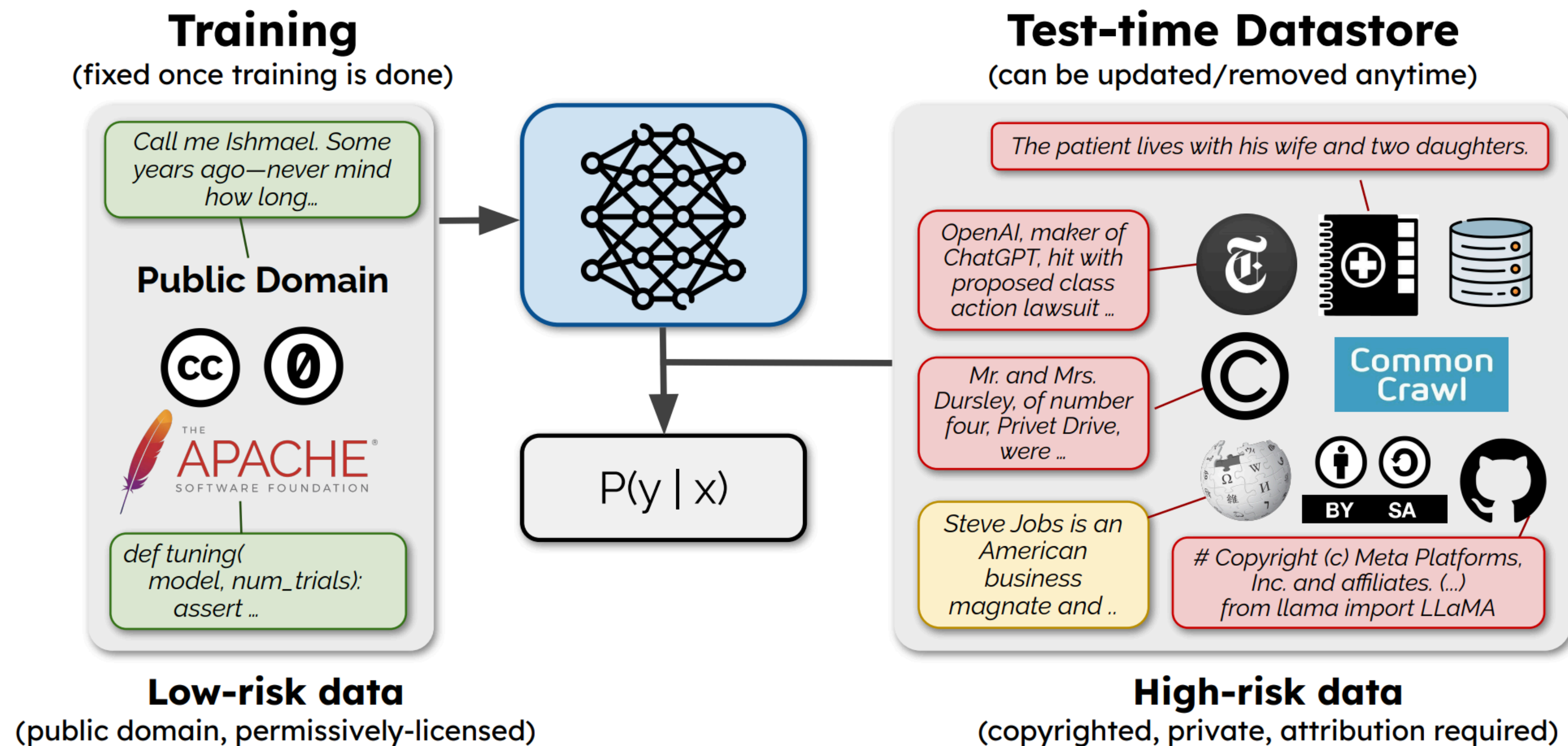
As of September 2024, the current Prime Minister of the United Kingdom is **Keir Starmer**, leader of the Labour Party. He took office following a general election in July 2024, where Labour secured a significant victory, unseating the Conservative Party's Rishi Sunak, who had been Prime Minister since 2022(Northeastern Global News)(Wikipedia).

| Government of the United Kingdom <a href="#">[edit]</a> |                                                                                       |                               |                        |                     |
|---------------------------------------------------------|---------------------------------------------------------------------------------------|-------------------------------|------------------------|---------------------|
| Prime Minister                                          | Portrait                                                                              | Since                         | Party                  | Ref                 |
| <a href="#">Keir Starmer</a>                            |  | 5 July 2024<br>(2 months ago) | <a href="#">Labour</a> | <a href="#">[1]</a> |



# Efficiency

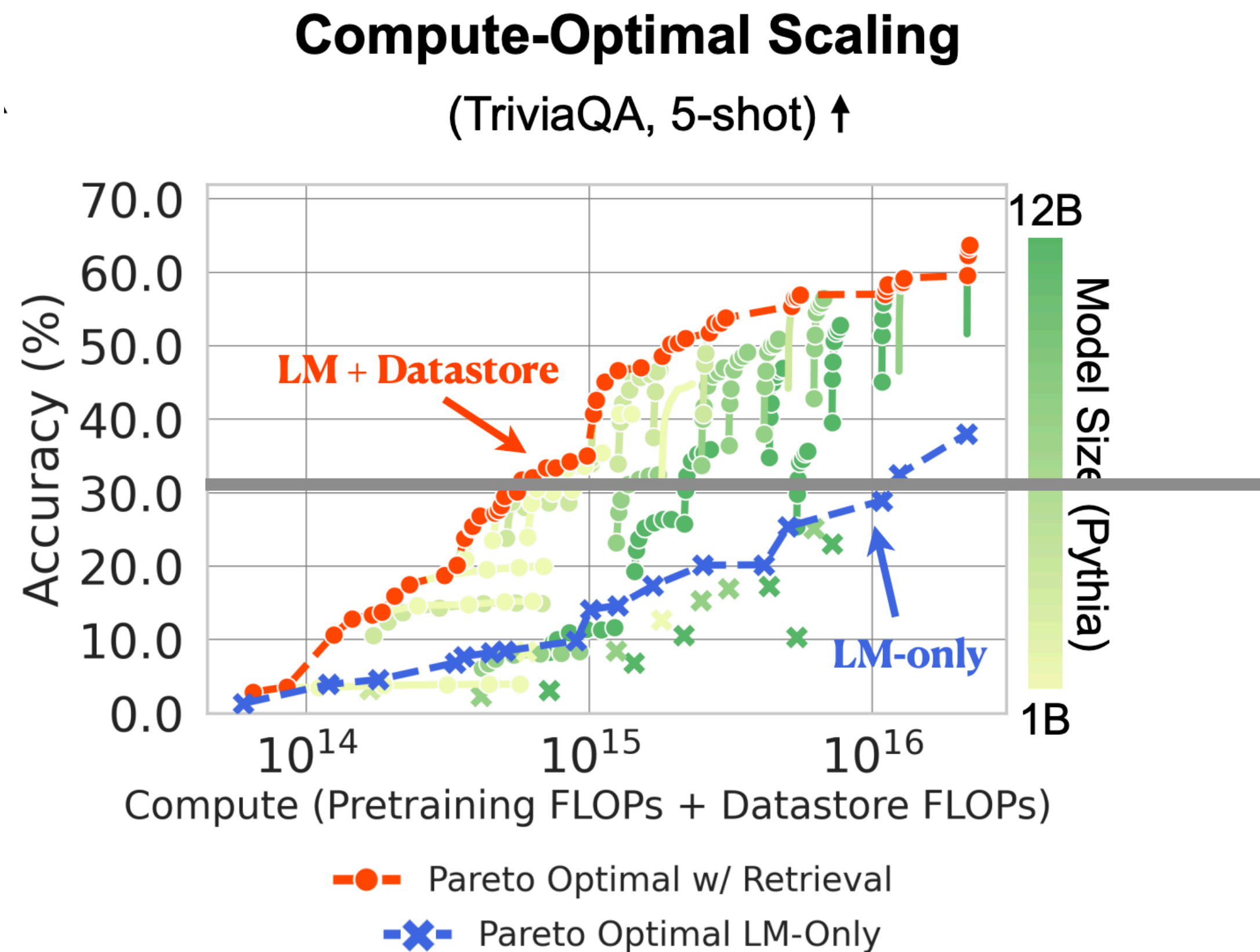
We can incorporate or remove high-risk data dynamically at inference, not training time





# Efficiency

Retrieval-augmented LMs can be much more parameter efficient and compute-optimal



FLOPs for pre-training large LMs

>

FLOPs for pre-training small LMs + construct datastore



# Widespread adoption

Retrieval-augmented LMs have been widely used both in academia and industry



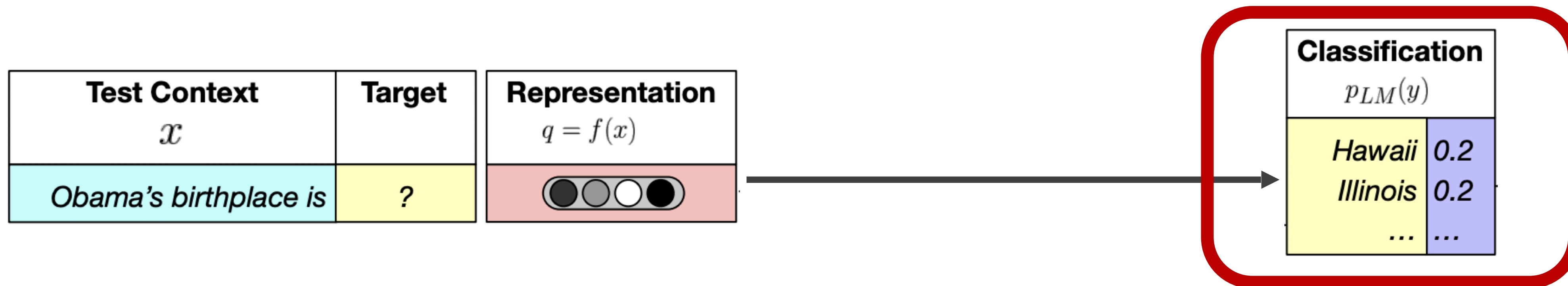
warm up: kNN-LM



# kNN LM: directly interpolate output token distributions

- Given a context  $x$ , a model predicts **parametric distributions** for next token

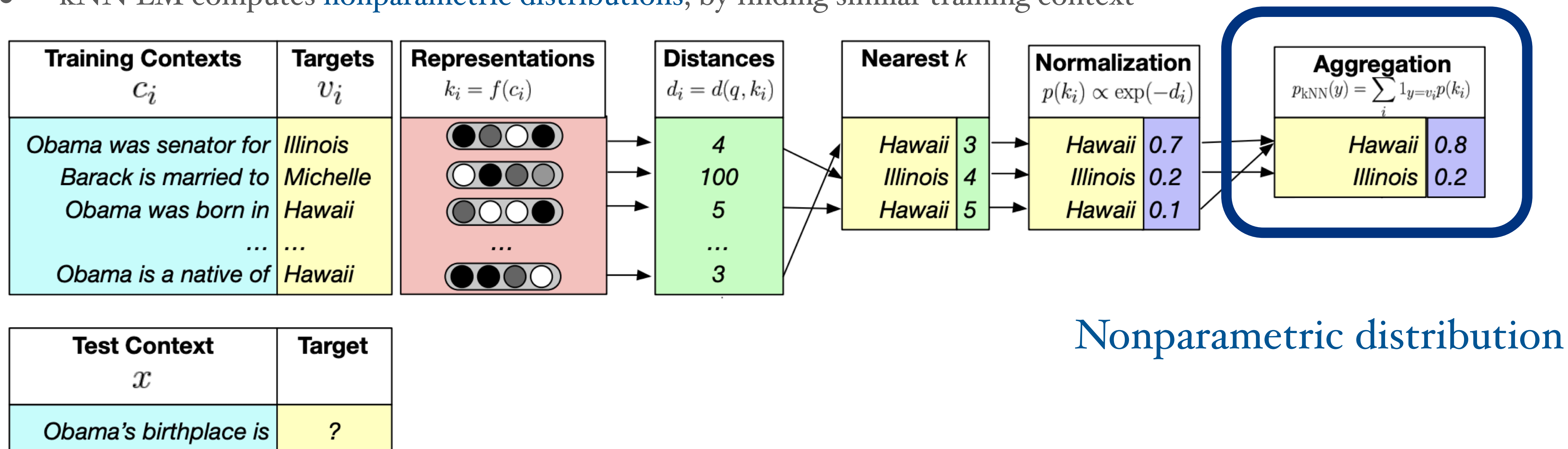
Parametric distribution  
(LM output distribution)





# kNN LM: directly interpolate output token distributions

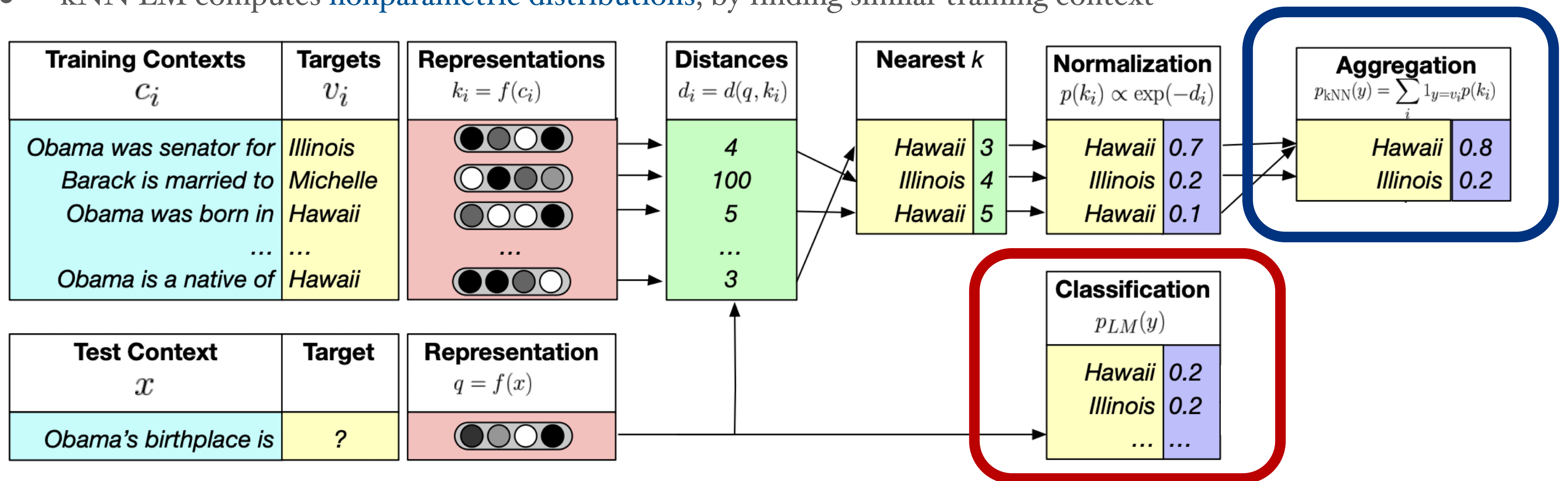
- Given a context  $x$ , a model predicts **parametric distributions** for next token
- kNN LM computes **nonparametric distributions**, by finding similar training context





# kNN LM: directly interpolate output token distributions

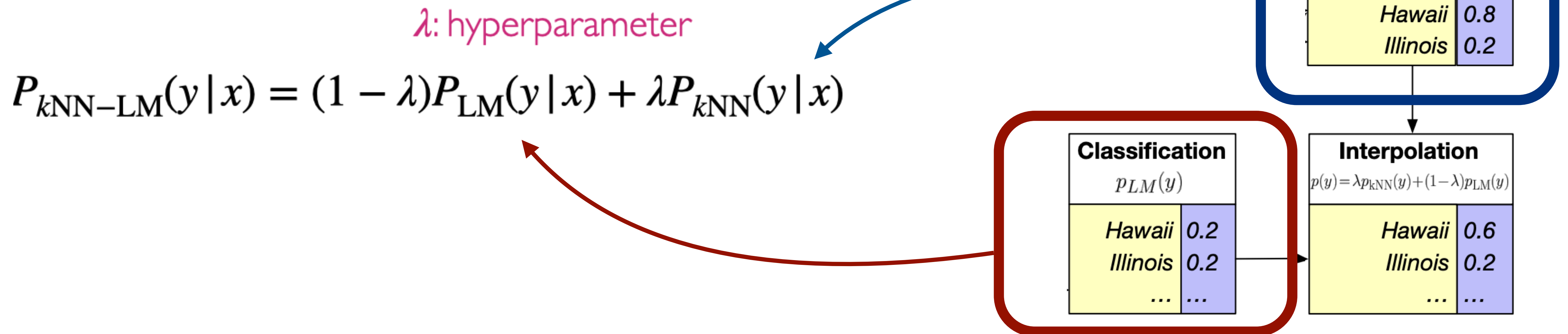
- Given a context  $x$ , a model predicts **parametric distributions** for next token
- kNN LM computes **nonparametric distributions**, by finding similar training context





# kNN LM: directly interpolate output token distributions

- Given a context  $x$ , a model predicts **parametric distributions** for next token
- kNN LM computes **nonparametric distributions**, by finding similar training context
- Interpolates two token distributions, adjusting the balance using a hyperparameter

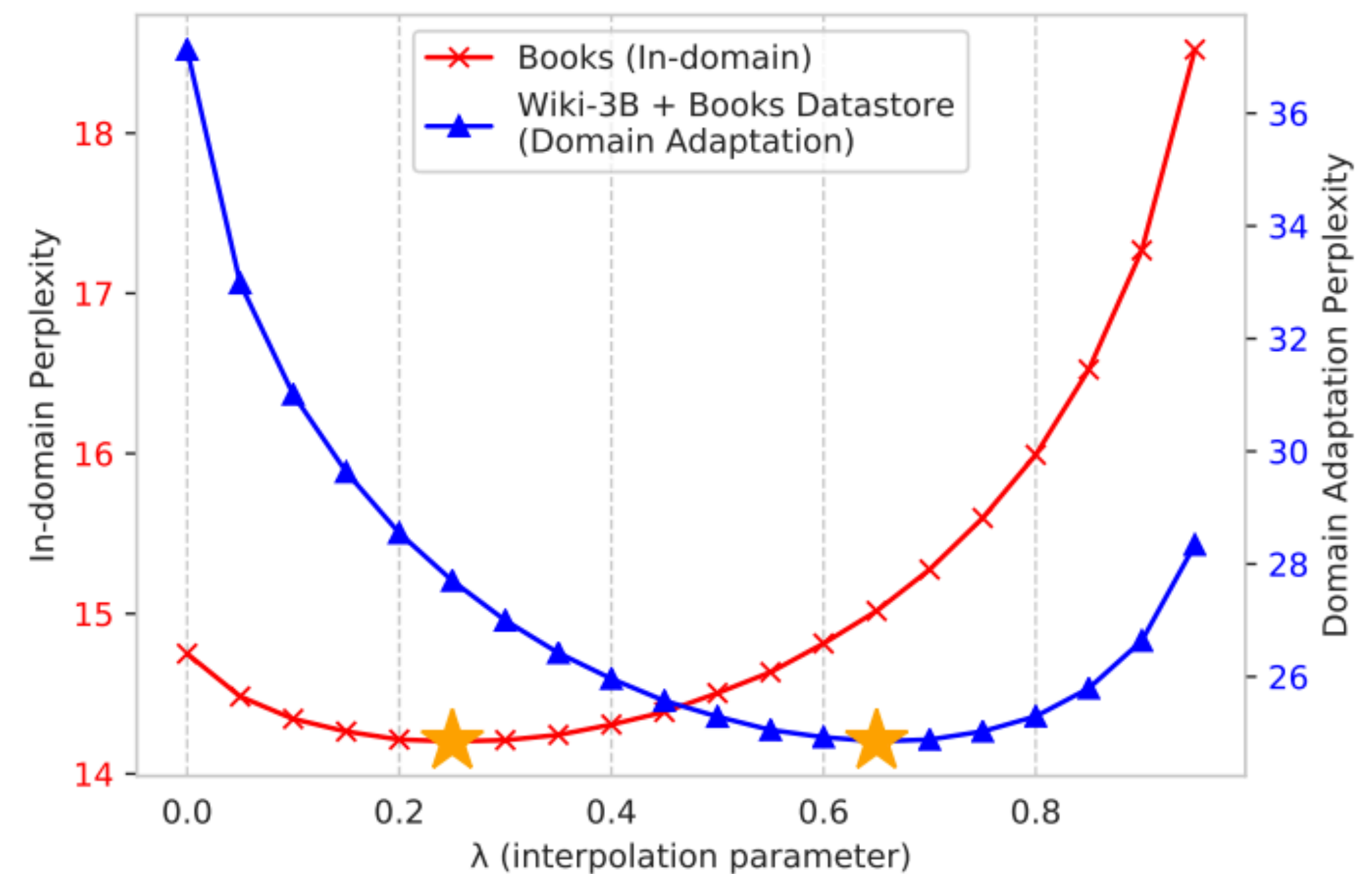
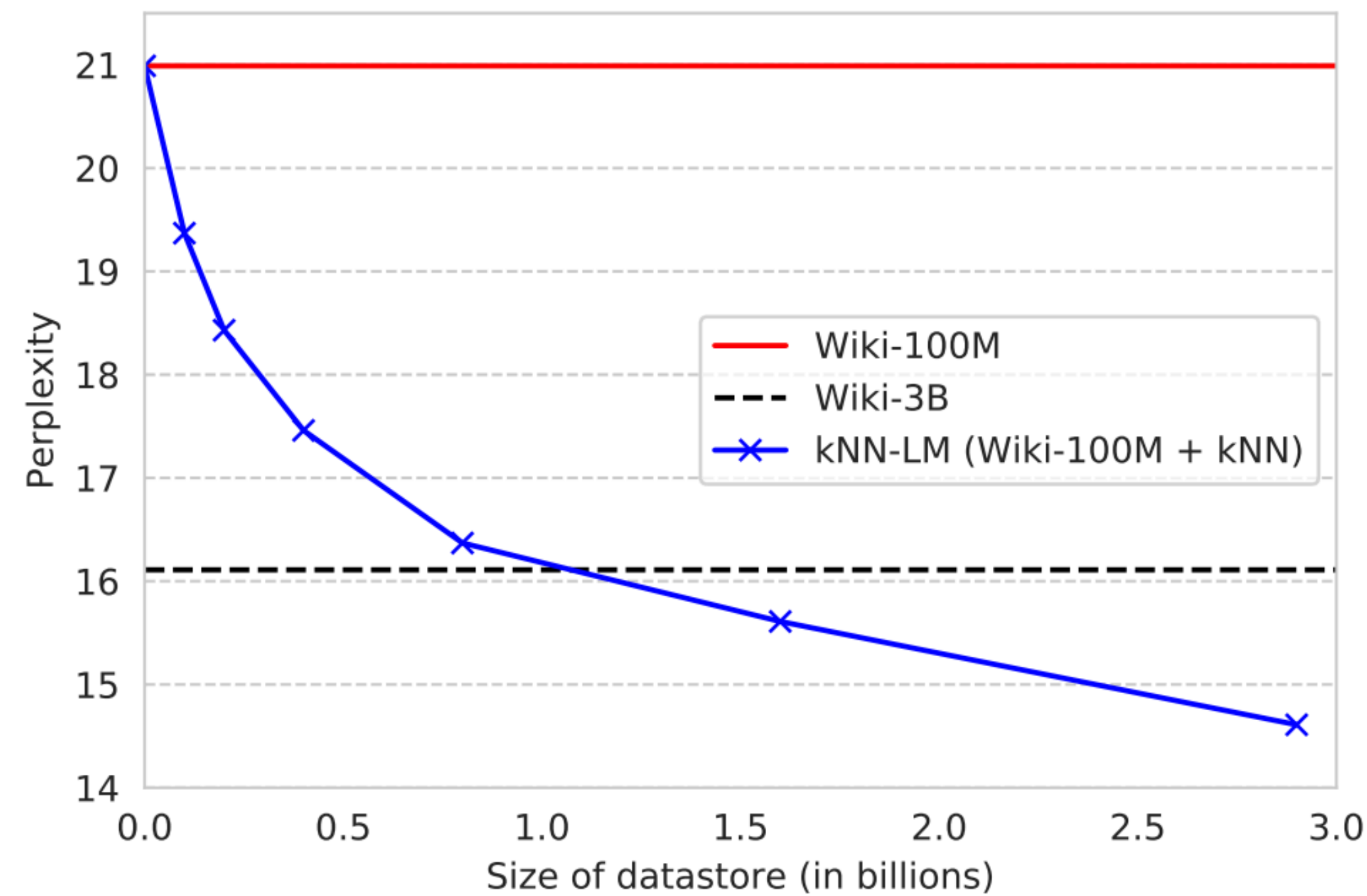




# kNN LM: directly interpolate output token distributions

## kNN LM outperforms much larger parametric LMs by large margin

- kNN LM constantly outperforms parametric LMs and 30x larger 3B LMs with larger datastore
- kNN LM also enables efficient & controlled domain adaptations





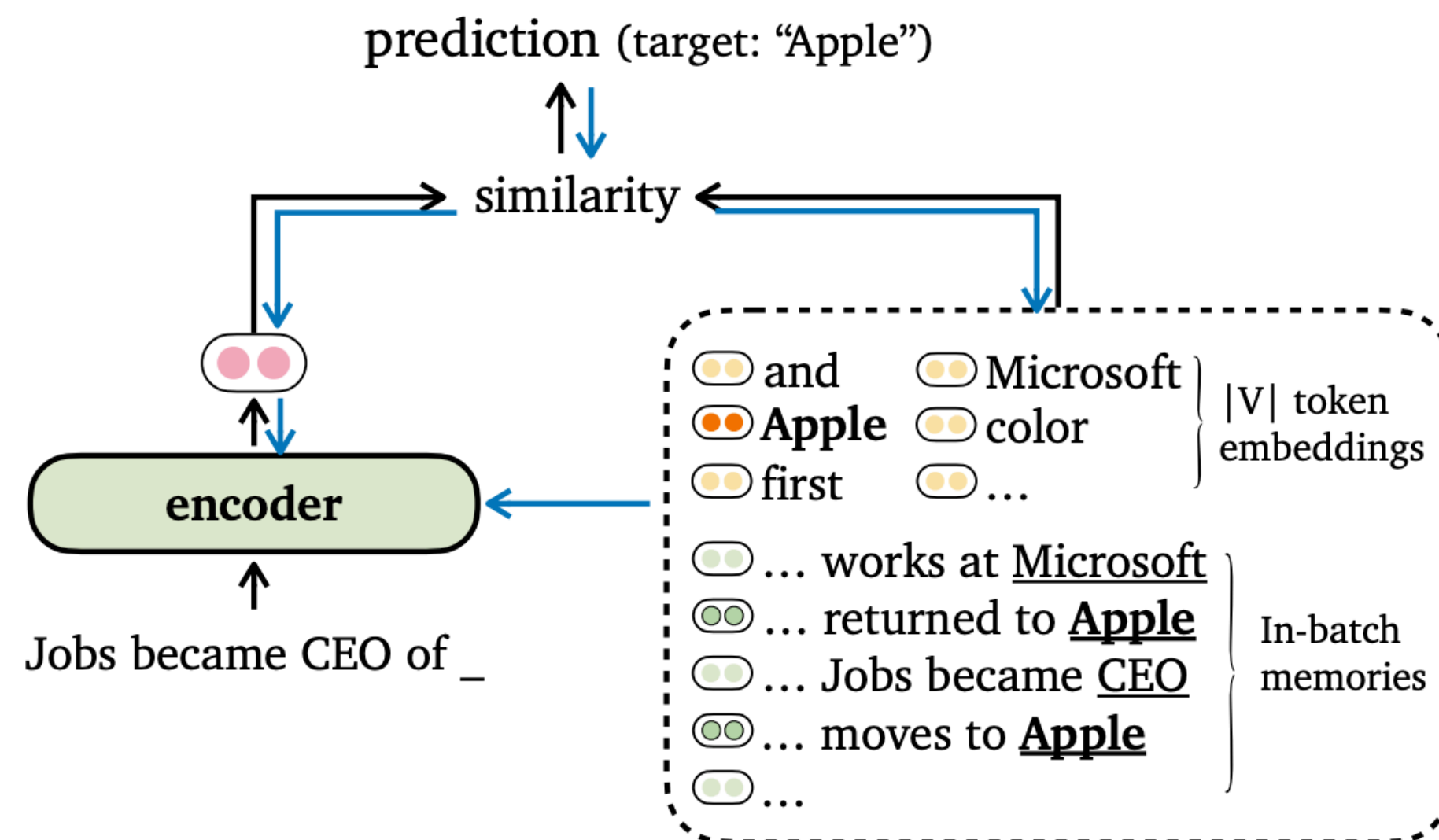
# Training with In-batch Memories (TRIME)

## Training kNN LM to better learn interpolations

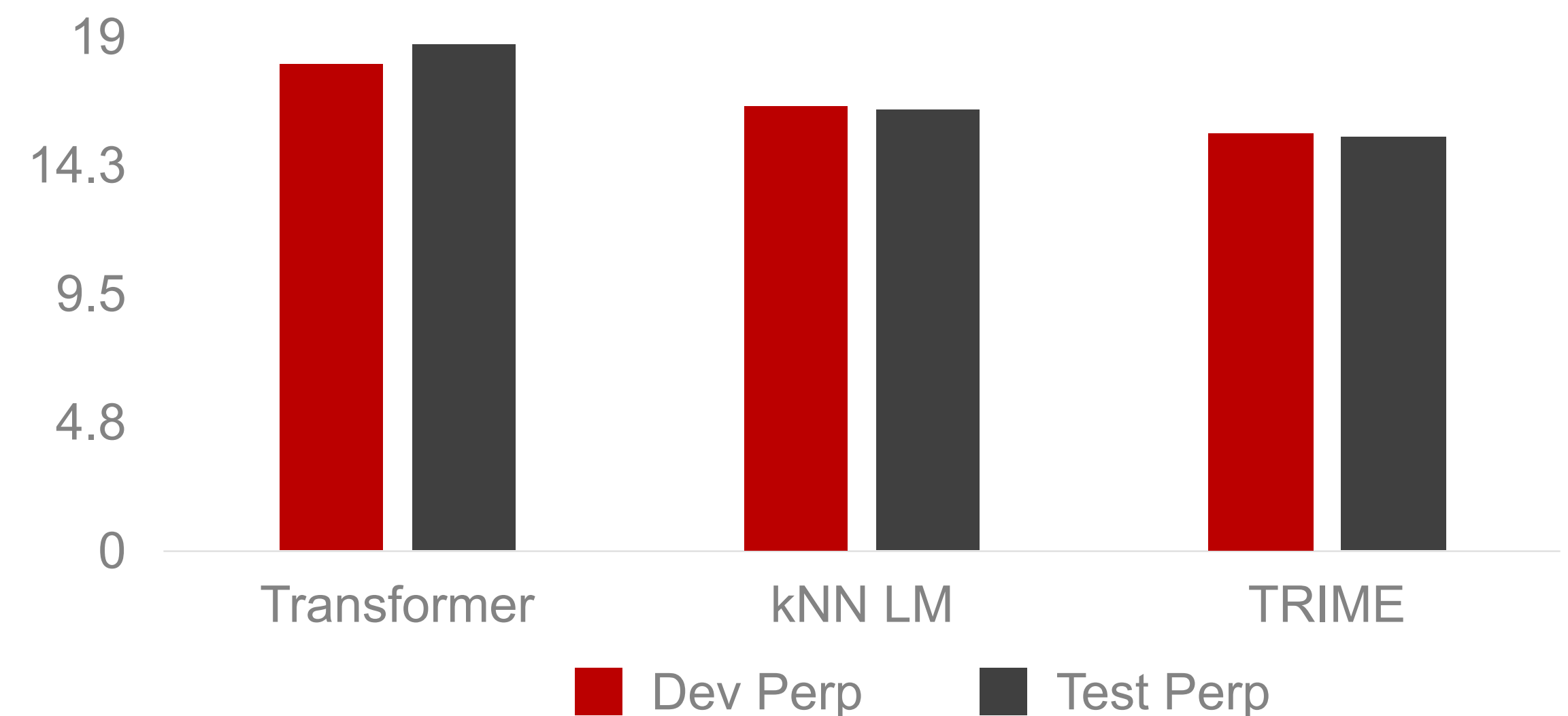
- kNN LM uses pre-trained LMs without any training
- TRIME introduces an efficient training method, outperforming kNN LM

- Target token's embedding (orange circle with two dots)
- Positive in-batch memory (green circle with two dots)
- Other token embeddings (yellow circle with two dots)
- Negative in-batch memory (light green circle with two dots)

↑ Forward pass ↓ Back-propagation



Wikitext 103 (Perplexity)



poll: cheating (1)



# Pros and cons of output interpolation

**$k$ NN LM & variants have unique advantages but have several empirical challenges**

- Pros
  - Provides token-level attributions
  - Enables explicit control between parametric and non-parametric memories
- Cons
  - Difficult to scale to large retrieval corpora (i.e., the number of embeddings equals the number of tokens)
  - Empirically shows limited effectiveness outside of upstream language modeling tasks

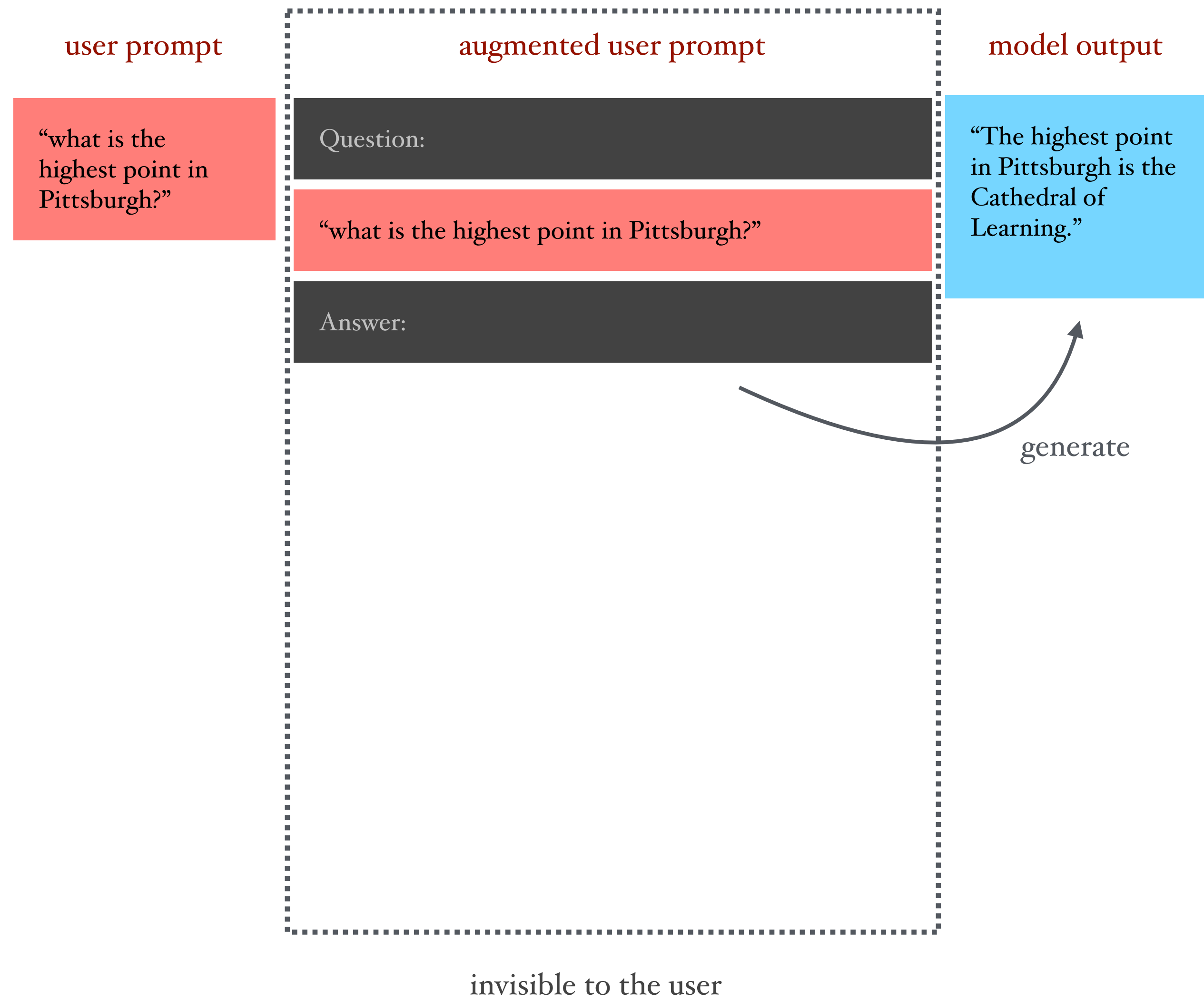
## **Great Memory, Shallow Reasoning: Limits of $k$ NN-LMs**

**Shangyi Geng   Wenting Zhao   Alexander M Rush**  
Cornell University  
{sg2323, wz346, arush}@cornell.edu

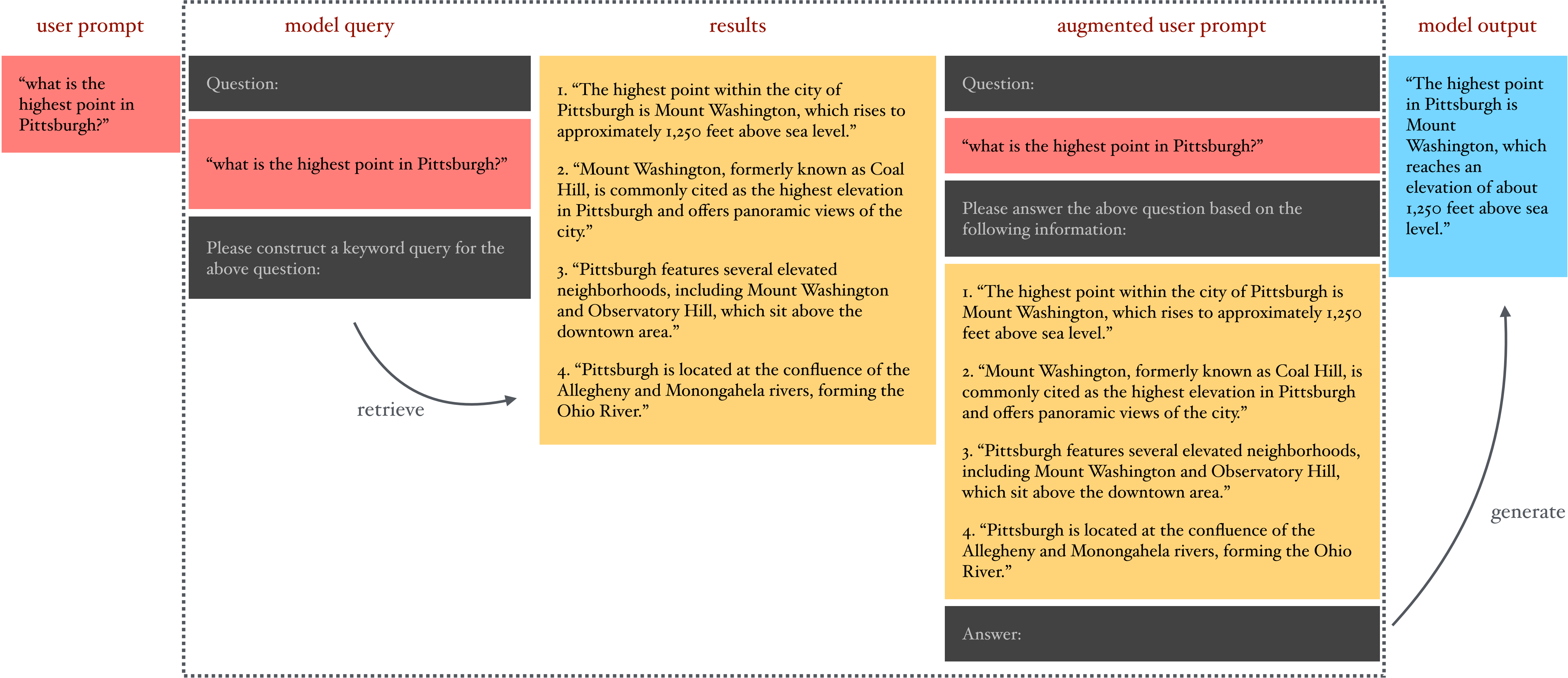
## **$k$ NN-LM Does Not Improve Open-ended Text Generation**

**Shufan Wang<sup>1</sup>   Yixiao Song<sup>1</sup>   Andrew Drozdov<sup>1</sup>**  
**Aparna Garimella<sup>2</sup>   Varun Manjunatha<sup>2</sup>   Mohit Iyyer<sup>1</sup>**  
University of Massachusetts Amherst<sup>1</sup>   Adobe Research<sup>2</sup>  
{shufanwang, yixiaosong, adrozdov, miyyer}@umass.edu  
{garimell, vmanjuna}@adobe.com

in-context RAG



# Retrieval becomes prompt engineering: generator sees retrieved text in its context window

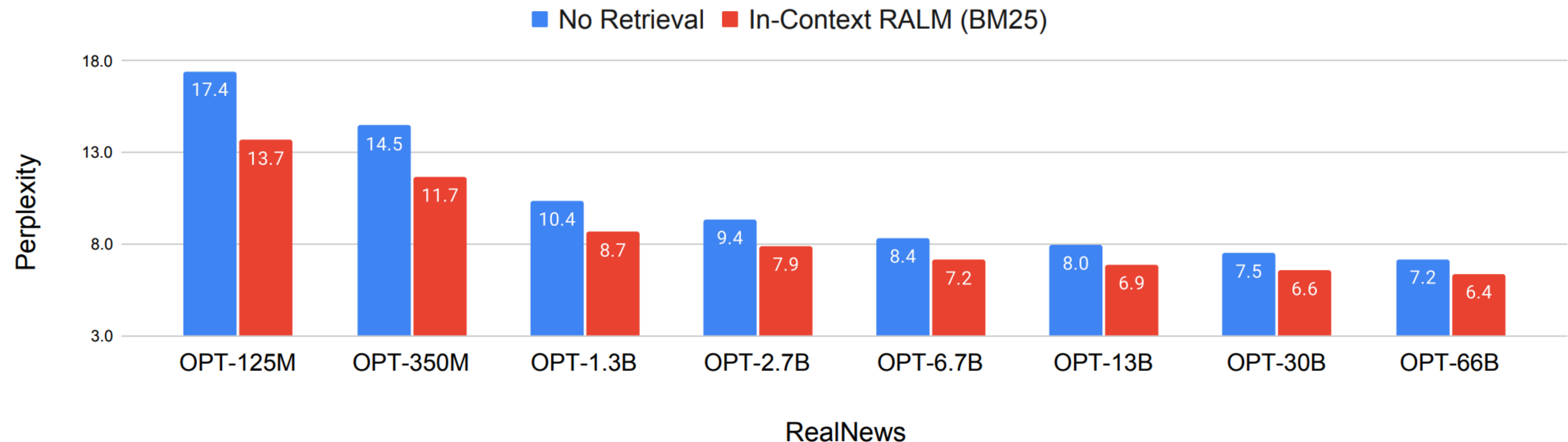




# In-context RAG

## Simply augmenting input-space of LMs give significant gain across different tasks

- In upstream language modeling task, simply adding retrieved context gives large gains, especially smaller models
- Similar significant gains in downstream tasks such as Question Answering

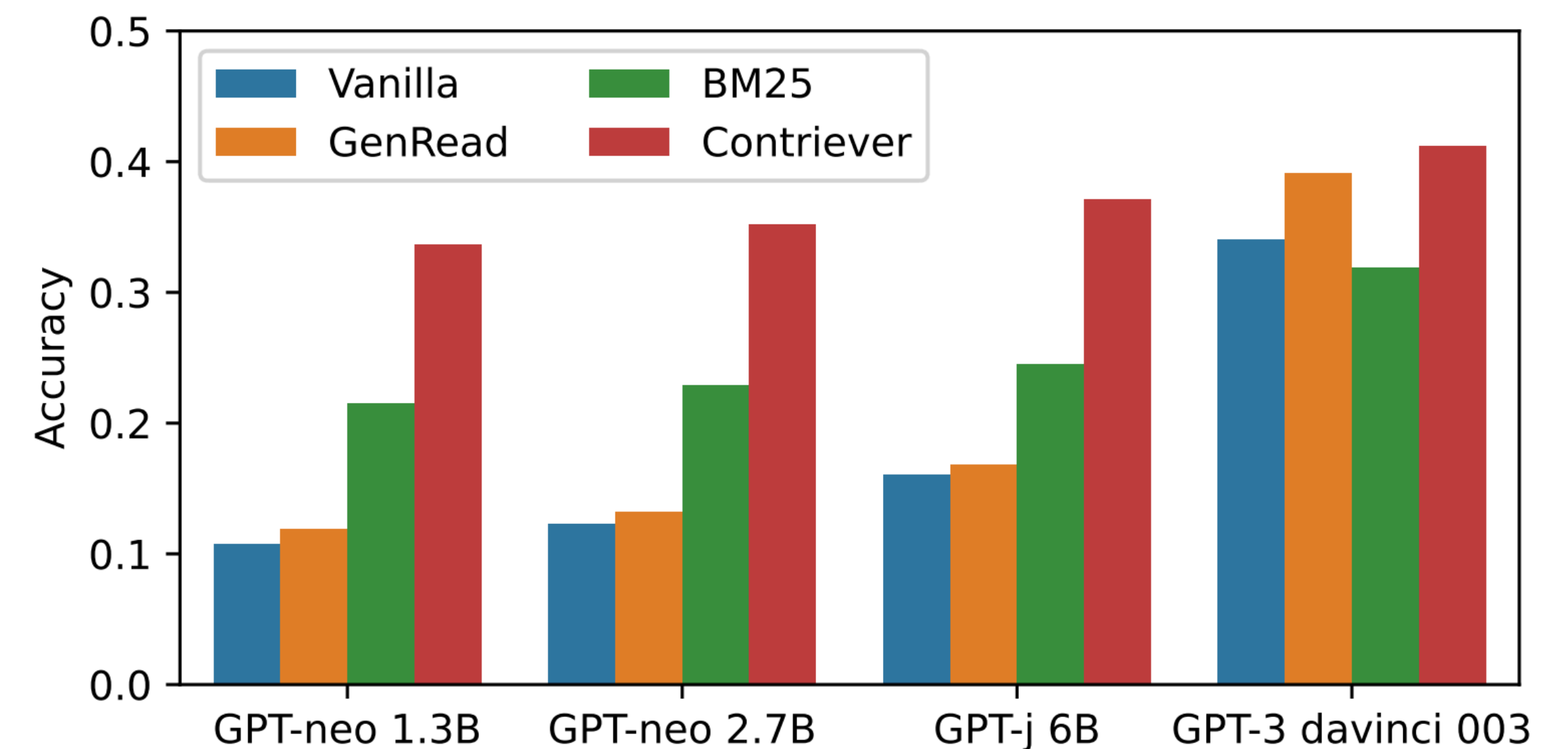
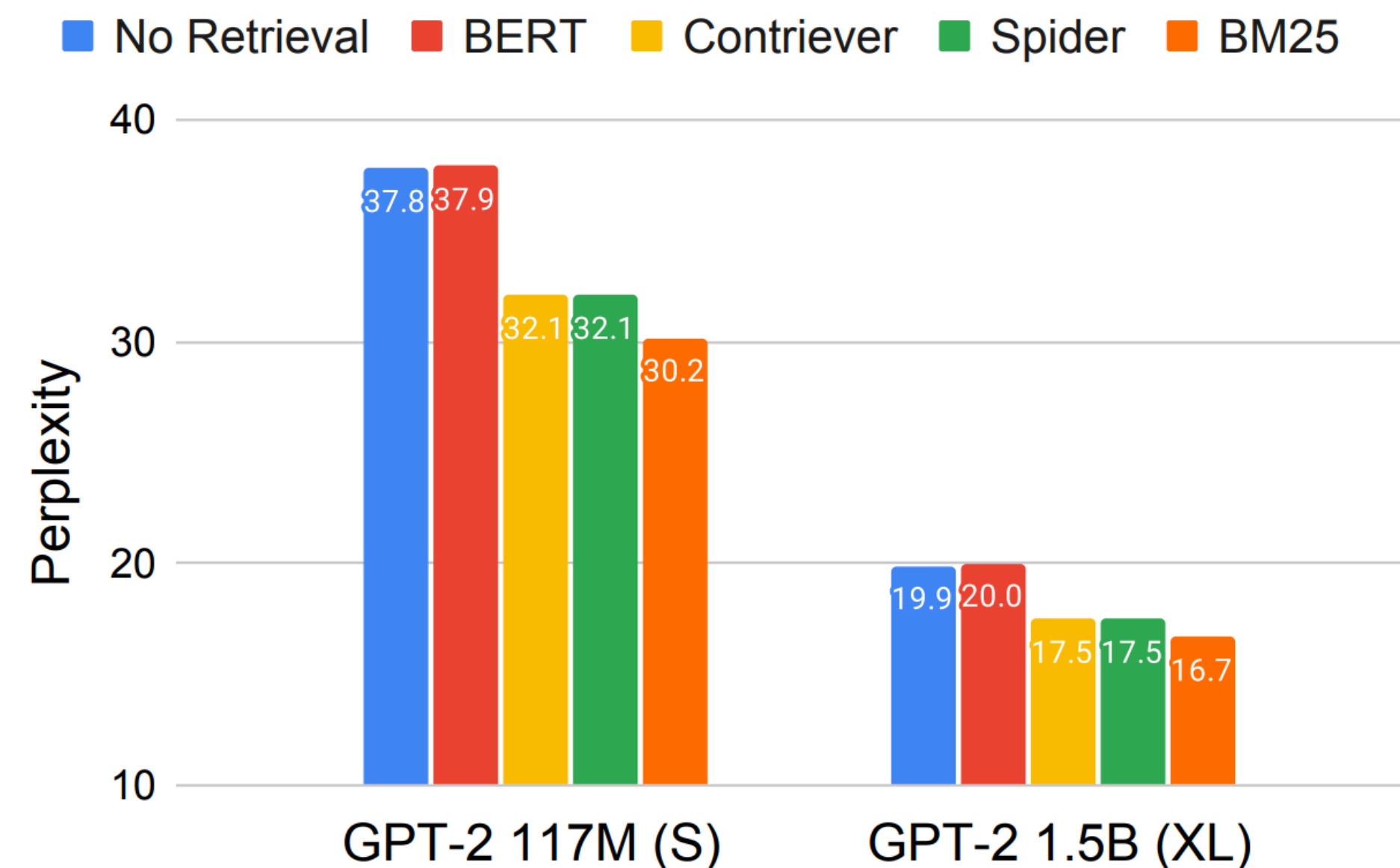




# In-context RAG

## Effects of retrieval systems for downstream task performance

- On language modeling, BM 25 results in best performance
- On downstream QA tasks, trained retrieval models eg Contriever results in best performance



poll: cheating (2)

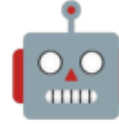


# Limitations of naïve RAG

## Effects of retrieval systems for downstream task performance

- In-context retrieval-augmented LMs sometimes generate content that is not fully supported by their citations

 What are the latest discoveries from the James Webb Space Telescope?



The James Webb Space Telescope is designed to peer into the dusty clouds of gas where stars and planetary systems are born. Webb has captured the first direct image of an exoplanet, and the Pillars of Creation in the Eagle Nebula[1][2]. Additionally, the telescope will be used to study the next interstellar interloper[3].

(\*Some generated statements may *not be fully supported* by citations, while others are *fully supported*.)

**Cited Webpages**

[1]:  nasa.gov (✗ citation does not support its associated statement)  
[NASA's Webb Confirms Its First Exoplanet](#)  
... Researchers confirmed an exoplanet, a planet that orbits another star, using NASA's James Webb Space Telescope for the first time. ...

[2]:  cnn.com (⚠ citation partially supports its associated statement)  
[Pillars of Creation: James Webb Space Telescope ...](#)  
... The Pillars of Creation, in the Eagle Nebula, is a star-forming region captured in a new image (right) by the James Webb Space Telescope that reveals more detail than a 2014 image (left) by Hubble ...

[3]:  nasa.gov (✓ citation fully supports its associated statement)  
[Studying the Next Interstellar Interloper with Webb](#)  
...Scientists have had only limited ability to study these objects once discovered, but all of that is about to change with NASA's James Webb Space Telescope...The team will use Webb's spectroscopic capabilities in both the near-infrared and mid-infrared bands to study two different aspects of the interstellar object.



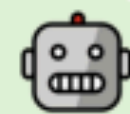
# Limitations of naïve RAG

## Effects of retrieval systems for downstream task performance

- In-context retrieval-augmented LMs sometimes generate content that is not fully supported by their citations
- They can easily be distracted by unhelpful context

**Q: Who is the actor playing Jason on general hospital?**

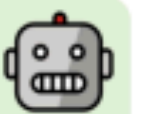
**Large Language Model (no retrieval)**



The answer is: Steve Burton



**Retrieval Augmented Language Model**



E: Jason Gerhardt (born April 21, 1974) is an American actor. He is known for playing the role of Cooper Barrett in General Hospital and Zack Kilmer in Mistresses.

The answer is: Jason Gerhardt





# Limitations of naïve RAG

## Effects of retrieval systems for downstream task performance

- In-context retrieval-augmented LMs sometimes generate content that is not fully supported by their citations
- They can easily be distracted by unhelpful context
- Diverse tasks require different retrieval needs e.g., content, frequency

Who is the current PM of UK?

Can be easily answered based on top documents retrieved at the beginning

Create a table listing all previous UK Prime Ministers, including their terms in office, political party, alma mater, and notable achievements.

This may require iterative retrieval, based on the current generation

The equation  $x^2 + 2x = i$  has two complex solutions. Determine the product of their real parts (from MATH)

Questions with similar solutions may have limited semantic similarities in embedding space