Retrieval-augmented Language Models

Akari Asai

https://akariasai.github.io/

March 1, 2024 at UW CSE 517 / 447 NLP

Slides adapted from ACL 2023 tutorial by Akari Asai, Sewon Min, Zexuan Zhong, Danqi Chen

https://acl2023-retrieval-lm.github.io/
Knowledgable and versatile Language Models (LMs)

You
Write a poem about large language models, including the following three words: coffee, croissants and banana  
(February 18, 2024)

ChatGPT
In the realm where thoughts converge and flow,
Resides a wonder, a grand tableau,
Where words dance in intricate array,
In the domain of the model's sway.

From the depths of bytes, they come alive,
In the language vast, where dreams contrive,
Coffee-fueled, the model's might,
Awakens realms in the dead of night.

Croissants of syntax, buttered with care,
Layered meanings, rich and rare,
Each line a pastry, crisp and sweet,
In the banquet of words, where senses meet.

Banana-shaped metaphors, ripe and bold,
In the orchard of thoughts, they unfold,
Symbolic fruits of the mind's terrain,
Nourishing narratives, ripe with gain.
How do such parametric LMs work?

\[ P(x_n \mid x_1, x_2, \ldots, x_{n-1}) \]

The capital city of Ontario is Toronto

Large-scale pre-training corpus (e.g., 1T tokens)

Language model (Transformers)

The capital city of Ontario is

\[ x_1 \quad x_2 \quad \ldots \quad x_{n-1} \]
How do such parametric LMs work?

\[ P(x_n | x_1, x_2, \ldots, x_{n-1}) \]

Language model (Transformers)

Ontario’s capital city,

\[ x_1 \quad x_2 \quad \ldots \quad x_{n-1} \]

Large-scale pre-trained language models (e.g., 1T tokens)

No explicit access to large-scale text data

Toronto 0.52
Ottawa 0.31
Vancouver 0.13
Montreal 0.03
Calgary 0.01

⋯
How do such parametric LMs work?

The capital city of Ontario is **Toronto**

The capital city of Ontario is ____

Training time

Test time
Hallucinations in LM outputs

You

Explain Akari Asai, including the list of her famous papers. (*February 18, 2024)

ChatGPT

Akari Asai is a prominent figure in the field of natural language processing, having made significant contributions to various aspects of NLP research. She has made notable contributions particularly in the areas of question answering, dialogue systems, and machine learning.

Here are some of her famous papers:

1. "Learning to Retrieve Reasoning Paths over Wikipedia Graph for Question Answering" - In this paper, Asai and her co-authors proposed a method for learning to retrieve reasoning paths over the Wikipedia graph to improve question answering systems.

2. "Large-Scale Multi-Document Summarization Dataset and Evaluation Benchmark" - This paper introduces a large-scale dataset for multi-document summarization, along with an evaluation benchmark, providing valuable resources for advancing research in this area.
Catastrophic Errors as Results of LM Hallucinations

Humiliated lawyers fined $5,000 for submitting ChatGPT hallucinations in court: ‘I heard about this new site, which I falsely assumed was, like, a super search engine’

Air Canada must honor request invented by airline’s chatbot

Air Canada appears to have quietly killed its costly chatbot support.
Retrieval-augmented LMs

The capital city of Ontario is **Toronto**

Training time

LM

The capital city of Ontario is _____

Test time

LM

Datastore!
Inference

Datastore

Query

Index

LM

Input
Inference: Datastore

Datastore

Raw text corpus

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

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

Retrieval input (not necessarily input to the LM)
Inference: Search

Datastore

Query

Index

LM

Input

Retrieval
Inference: Incorporation

Datastore

Query

Index

LM

Input
Retrieval-augmented LMs are now widely used!
Today’s outline

- Why do we need retrieval-augmented LMs?
- Architectures of retrieval-augmented LMs (Inference)
- Training of retrieval-augmented LMs
- Limitations and future directions

Today’s outline

- Why do we need retrieval-augmented LMs?
- Architectures of retrieval-augmented LMs (Inference)
- Training of retrieval-augmented LMs
- Limitations and future directions

Question:
A: Because retrieval-augmented LMs can solve many core limitations of parametric LMs!
Core limitations of parametric LMs

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

**You**

Explain Akari Asai, including the list of her famous papers. (*February 18, 2024*)

**ChatGPT**

Akari Asai is a prominent figure in the field of natural language processing (NLP), known for her significant contributions to various aspects of NLP research. She has made notable contributions particularly in the areas of question answering, dialogue systems, and machine learning.

Here are some of her famous papers:

1. **“Large-Scale Multi-Document Summarization Dataset and Evaluation Benchmark”** - This paper introduces a large-scale dataset for multi-document summarization, along with an evaluation benchmark, providing valuable resources for advancing research in this area.
Core limitations of parametric LMs

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Explain Akari Asai, including the list of her famous papers.

Her most famous paper is “Large-Scale Multi-Document Summarization Dataset and Evaluation Benchmark”
Core limitations of parametric LMs

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Crawled Web data in 2022

Language model

ChatGPT

I'm sorry, but I don't have access to real-time information including events beyond January 2022.
Core limitations of parametric LMs

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Retrain LMs using 2024 data

Language model

ChatGPT
I'm sorry, but I don't have access to real-time information including events beyond January 2022.
Core limitations of parametric LMs

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Dodge et al., Documenting Large Webtext Corpora: A Case Study on the Colossal Clean Crawled Corpus. EMNLP 2021.
Core limitations of parametric LMs

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

New York Times lawsuits against OpenAI
Core limitations of parametric LMs

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Kandpal et al., Large Language Models Struggle to Learn Long-Tail Knowledge. ICML 2023.
Q: So how can retrieval-augmented LMs solve those challenges?
How retrieval-augmented LMs solve the issues?

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Significant improvements across model scale, with larger gain with smaller LMs

QA

How retrieval-augmented LMs solve the issues?

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Retrieved text can be used as attributions
How retrieval-augmented LMs solve the issues?

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Replacing datastores to catch up dynamically changing world without re-training

Input

Query

LM (GPT-3)

Index

2024

Output

How retrieval-augmented LMs solve the issues?

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Segregating copyright-sensitive data from pre-training data

Min* and Gururangan* et al., SILO Language Models: Isolating Legal Risk In a Nonparametric Datastore. ICLR 2024.
How retrieval-augmented LMs solve the issues?

- Hallucinations
- Lack of attributions
- Costs of adaptations
- Copyright / privacy
- Large parameter size

Smaller LMs with retrieval outperform much larger LMs e.g., GPT-3

Mallen*, Asai* et al., When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories (Best Video; Oral) 2023.
Today’s outline

Why do we need retrieval-augmented LMs?

Architectures of retrieval-augmented LMs (Inference)

Training of retrieval-augmented LMs

Limitations and future directions
Notations

Datastore $\mathcal{D}$

Input $x$ → Query $q$ → Index → Output $y$
Inference: Index

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

**sim**: a similarity score between two pieces of text

\[
\text{Example} \quad \text{sim}(i,j) = \left( \frac{\text{tf}_{i,j}}{\text{df}_i} \right) \times \log \frac{N}{\text{df}_i}
\]

# of occurrences of \( i \) in \( j \)

# of docs containing \( i \)

# of total docs

An entire field of study on how to get (or learn) the similarity function better (We’ll see some later!)

\[
\text{Example} \quad \text{sim}(i,j) = \text{Encoder}(i) \cdot \text{Encoder}(j)
\]

Maps the text into an \( h \)-dimensional vector
Inference: Index

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

**sim**: a similarity score between two pieces of text

**Index**: given $q$, return $\arg\max_{d \in \mathcal{D}} \sim(q, d)$ through fast nearest neighbor search

$k$ elements from a datastore

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

https://github.com/facebookresearch/faiss/wiki/
Categorization of 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
- The capital city of Ontario is Toronto.
- w/ retrieval w/ r w/r w/r w/r w/ r w/r w/r
- The capital city of Ontario is Toronto.
- w/ retrieval w/r w/r
- The capital city of Ontario is Toronto.
Categorization of retrieval-augmented LMs

**What** to retrieve?
- Query
  - Text chunks (passages)?
  - Tokens?
  - Something else?

**How** to use retrieval?
- Input
  - LM
  - Output

**When** to retrieve?
- Today we focus on
  - 1. What to retrieve
  - 2. How to use retrieval
Three representative architectures

What: Text chunks
How: Input

Input augmentation (RAG)

What: Tokens
How: Output

Output interpolations

What: Text chunks
How: Intermediate

Intermediate fusion

More details?
- Section 3 of our tutorial (https://acl2023-retrieval-lm.github.io/)
- Our position paper (Asai et al., 2024; https://akariasai.github.io/assets/pdf/ralm_position.pdf)
Three representative architectures

**What:** Text chunks  
**How:** Input  
REALM (Guu et al., 2020)

**What:** Text chunks  
**How:** Intermediate  
RETRO (Borgeaud et al., 2021)

**What:** Tokens  
**How:** Output  
kNN-LM (Khandelwal et al., 2020)

More details?
- Section 3 of our tutorial (https://acl2023-retrieval-lm.github.io/)
- Our position paper (Asai et al., 2024; https://akariasai.github.io/assets/pdf/ralm_position.pdf)
Three representative architectures

What: Text chunks
How: Input

REALM (Guu et al., 2020)

What: Text chunks
How: Intermediate

RETRO (Borgeaud et al., 2021)

What: Tokens
How: Output

kNN-LM (Khandelwal et al., 2020)

More details?
• Section 3 of our tutorial (https://acl2023-retrieval-lm.github.io/)
• Our position paper (Asai et al., 2024; https://akariasai.github.io/assets/pdf/ralm_position.pdf)
x = World Cup 2022 was the last with 32 teams before the increase to [MASK] in 2026.
\( x = \) World Cup 2022 was the last with 32 teams before the increase to \([\text{MASK}]\) in 2026.

FIFA World Cup 2026 will expand to 48 teams.
REALM \textsuperscript{(Guu et al 2020)}

$x = \text{World Cup 2022 was the last before the increase to [MASK] in the 2026 tournament.}$

\begin{align*}
\text{FIFA World Cup 2026 will expand to 48 teams.} \\
\text{World Cup 2022 was … the increase to [MASK] in 2026.}
\end{align*}

$k \text{ chunks of text (passages)}$

\begin{align*}
\text{Retrieve stage} \\
\text{Read stage}
\end{align*}

Guu et al. \textit{REALM: Retrieval-Augmented Language Model Pre-Training. ICML 2020.}
REALM: (1) Retrieve stage

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

\( z = \text{Encoder}(z) \)

\( x = \text{Encoder}(x) \)

\( z_1, \ldots, z_k = \text{argTop}-k (x \cdot z) \)

\( k \) retrieved chunks

---

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: (2) Read stage

\[
\sum_{z \in \mathcal{D}} P(z \mid x) P(y \mid x, z)
\]

Need to approximate

→ Consider top \( k \) chunks only

→ From the retrieve stage

→ From the read stage

Weighted average

\[P(y \mid x, z_1)\]

\[P(y \mid x, z_2)\]

\[P(y \mid x, z_k)\]
Recent trend: RAG with LLMs

Existing parametric LMs (e.g., GPT-3)

Off-the-shelf retrievers (e.g., Google search, BM25, DPR)

Simply combining existing models w/o training has shown to be successful!

Three representative architectures

What: Text chunks  
REALM (Guu et al., 2020)

What: Tokens  
kNN-LM (Khandelwal et al., 2020)

What: Text chunks  
RETRO (Borgeaud et al., 2021)

More details?
- Section 3 of our tutorial (https://acl2023-retrieval-lm.github.io/)
- Our position paper (Asai et al., 2024; https://akariasai.github.io/assets/pdf/ralm_position.pdf)
RETRO (Borgeaud et al. 2022)

- Incorporation in the “intermediate layer” instead of the “input” layer → designed for many chunks, frequently, more efficiently

- Scale the datastore (1.8T tokens)
RETRO (Borgeaud et al. 2021)

$x = \text{World Cup 2022 was the last with 32 teams, before the increase to}$

$x_1$ $x_2$ $x_3$

$(k \text{ chunks of text per split})$

$x_1$ Retrieval Encoder $\rightarrow$ Index $\rightarrow$ LM Encoder $\leftrightarrow$

$p_1^1 \ldots p_k^1$ $p_1^2 \ldots p_k^2$ $p_1^3 \ldots p_k^3$

$E_1$ $E_2$ $E_3$

$(A \ r \times k \times d \text{ matrix})$

$(r = \# \text{ tokens per text chunk})$

$(d = \text{hidden dimension})$

$(k = \# \text{retrieved chunks per split})$

Regular decoder

Transformers blocks ($xL$)

Decoder in RETRO

 Chunked Cross Attention (CCA)

Chunked Cross Attention

Outputs from the previous layer $H$

$E_1$, $E_2$

Chunked Cross Attention

Outputs from the previous layer $H$

Chunked Cross Attention

Outputs from the previous layer $H$
Chunked Cross Attention

Outputs from the previous layer \( H \)

Inputs to the next layer

## Results

Perplexity: The lower the better

<table>
<thead>
<tr>
<th>Model</th>
<th>Retrieval Set</th>
<th>#Database tokens</th>
<th>#Database keys</th>
<th>Valid</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline transformer (ours)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>21.53</td>
<td>22.96</td>
</tr>
<tr>
<td>kNN-LM (ours)</td>
<td>Wikipedia</td>
<td>4B</td>
<td>4B</td>
<td>18.52</td>
<td>19.54</td>
</tr>
<tr>
<td>RETRO</td>
<td>Wikipedia</td>
<td>4B</td>
<td>0.06B</td>
<td>18.46</td>
<td>18.97</td>
</tr>
<tr>
<td>RETRO</td>
<td>C4</td>
<td>174B</td>
<td>2.9B</td>
<td>12.87</td>
<td>10.23</td>
</tr>
<tr>
<td>RETRO</td>
<td>MassiveText (1%)</td>
<td>18B</td>
<td>0.8B</td>
<td>18.92</td>
<td>20.33</td>
</tr>
<tr>
<td>RETRO</td>
<td>MassiveText (10%)</td>
<td>179B</td>
<td>4B</td>
<td>13.54</td>
<td>14.95</td>
</tr>
<tr>
<td>RETRO</td>
<td>MassiveText (100%)</td>
<td>1792B</td>
<td>28B</td>
<td>3.21</td>
<td>3.92</td>
</tr>
</tbody>
</table>

**RETRO (w/ Wikipedia) outperforms its parametric counterpart**

## Results

### Perplexity: The lower the better

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

**RETRO w/ 1.8T datastores achieves SOTA**

Results

Gains are constant with model scale

The larger datastore is, the better

Three representative architectures

- REALM (Guu et al., 2020)
- RETRO (Borgeaud et al., 2021)
- kNN-LM (Khandelwal et al., 2020)

More details?
- Section 3 of our tutorial (https://acl2023-retrieval-lm.github.io/)
- Our position paper (Asai et al., 2024; https://akariasai.github.io/assets/pdf/ralm_position.pdf)
kNN-LM (Khandelwal et al. 2020)

✓ A different way of using retrieval, where the LM outputs a nonparametric distribution over every token in the data.

✓ Can be seen as an incorporation in the “output” layer
<table>
<thead>
<tr>
<th>Test Context</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>Obama’s birthplace is</td>
</tr>
</tbody>
</table>
**kNN-LM** (Khandelwal et al. 2020)

**Test Context**
- **Obama’s birthplace is**

**Target**
- ?

**Representation**
- $q = f(x)$

**Classification**
- $p_{LM}(y)$
  - Hawaii: 0.2
  - Illinois: 0.2
  - ...

Parametric distribution
Obama was senator for Illinois from 1997 to 2005, Barack is married to Michelle and their first daughter, Obama was born in Hawaii, and graduated from Columbia University. Obama is a native of Hawaii.
kNN-LM (Khandelwal et al. 2020)

Which tokens in a datastore are close to the next token?
kNN-LM (Khandelwal et al. 2020)

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

- **Training Contexts** $C_i$
  - Obama was senator for
  - Barack is married to
  - Obama was born in
  - Obama is a native of
- **Targets** $v_i$
  - Illinois
  - Michelle
  - Hawaii
  - ...
  - Hawaii
- **Representations** $k_i = f(v_i)$

- **Test Context** $x$
- **Target**
- **Representation** $q = f(x)$

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

Which prefixes in a datastore are close to the prefix we have?
# kNN-LM (Khandelwal et al. 2020)

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

<table>
<thead>
<tr>
<th>Training Contexts $C_i$</th>
<th>Targets $V_i$</th>
<th>Representations $k_i = f(v_i)$</th>
<th>Distances $d_i = d(q, k_i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obama was senator for</td>
<td>Illinois</td>
<td><img src="image1" alt="Vector Representation" /></td>
<td>4</td>
</tr>
<tr>
<td>Barack is married to</td>
<td>Michelle</td>
<td><img src="image2" alt="Vector Representation" /></td>
<td>100</td>
</tr>
<tr>
<td>Obama was born in</td>
<td>Hawaii</td>
<td><img src="image3" alt="Vector Representation" /></td>
<td>5</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td><img src="image4" alt="Vector Representation" /></td>
<td>...</td>
</tr>
<tr>
<td>Obama is a native of</td>
<td>Hawaii</td>
<td><img src="image5" alt="Vector Representation" /></td>
<td>3</td>
</tr>
</tbody>
</table>

Test Context $X$

<table>
<thead>
<tr>
<th>Target</th>
<th>Representation $q = f(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obama’s birthplace is</td>
<td><img src="image6" alt="Vector Representation" /></td>
</tr>
</tbody>
</table>
Which vectors in a datastore are close to the vector we have?
**kNN-LM** (Khandelwal et al. 2020)

\[
P_{\text{kNN-LM}}(y | x) = (1 - \lambda)P_{\text{LM}}(y | x) + \lambda P_{\text{kNN}}(y | x)
\]

\[P(k_i) \propto \exp(-d_i)
\]

\[
\text{Aggregation} \quad p_{\text{kNN}}(y) = \sum_{k=1}^{K} p(k_i)
\]

\[
\text{Interpolation} \quad p(u) = \lambda p_{\text{kNN}}(u) + (1 - \lambda)P_{\text{LM}}(u)
\]

kNN-LM - results

The lower the better

- Outperforms no-retrieval LM
- Better with bigger datastore

Three representative architectures

What: Text chunks  
How: Input

REALM (Guu et al., 2020)

What: Tokens  
How: Output

kNN-LM (Khandelwal et al., 2020)

What: Text chunks  
How: Intermediate

RETRo (Borgeaud et al., 2021)

More details? 
- Section 3 of our tutorial (https://acl2023-retrieval-lm.github.io/)
- Our position paper (Asai et al., 2024; https://akariasai.github.io/assets/pdf/ralm_position.pdf)
Today’s outline

Why do we need retrieval-augmented LMs?

Architectures of retrieval-augmented LMs (Inference)

Training of retrieval-augmented LMs

Limitations and future directions

[Question: bit.ly/akari_ralm_lec]
Training retrieval-augmented LMs

Datastore

Query

Index

LM

Input

Output

Back-propagate
Why is training challenging?

Training LMs can be very expensive!
Why is training challenging?

Too large! Expensive to update index during training!

Training LMs can be very expensive!
Challenges of updating retrieval models

We may encode a lot of (>100M) text chunks using the encoder!
Challenges of updating retrieval models

During training, we will update the encoder
Challenges of updating retrieval models

Datastore

\[ x_1 \]

\[ x_2 \]

\[ x_3 \]

…

Encoder

Index

Re-indexing will be very expensive!
Training methods for retrieval-augmented LMs

- Independent training
- Sequential training
- Joint training w/ asynchronous index update
- Joint training w/ in-batch approximation
Training methods for retrieval-augmented LMs

- Independent training
  - Sequential training
  - Joint training w/ asynchronous index update
  - Joint training w/ in-batch approximation
Independent training

Retrieval models and language models are trained independently.

- Training language models

  \[
  \text{Input} \rightarrow \text{LM} \rightarrow \text{Output}
  \]

- Training retrieval models

  \[
  \text{Datastore} \rightarrow \text{Retriever} \rightarrow \text{Chunks/tokens}
  \]

  Query
Independent training

Retrieval models and language models are trained independently

- Training language models

  Input → **LM** → Output

- Training retrieval models

  Datastore → **Retriever** → Chunks/tokens

  Query
Sparse retrieval models: TF-IDF / BM25

In 1997, Apple merged with NeXT, and Steve Jobs became CEO of …

Jobs returned to Apple as CEO after the company’s acquisition …

[0, 0, 0.4, 0, 0.8, 0.7, …]

[0, 1.2, 0.4, 0, 0.8, 0, …]

Text chunks

Sparse vectors

Lexical overlap

No training needed!

Dense retrieval models: **DPR** (Karpukhin et al. 2020)

Inner Product Similarity

Dense vectors

Encoder

$q$

Query

Encoder

Text chunks

Karpukhin et al. Dense Passage Retrieval for Open-Domain Question Answering. EMNLP 2020.
Training dense retrieval models: DPR

Inner Product Similarity

Encoder

$q$

Query

Encoder

Text chunks

Karpukhin et al. Dense Passage Retrieval for Open-Domain Question Answering. EMNLP 2020.
Training dense retrieval models: DPR

\[ L(q, p^+, p_1^-, p_2^-, \ldots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^{n} \exp(\text{sim}(q, p_j^-))} \]
Training dense retrieval models: DPR

\[
L(q, p^+, p_1^-, p_2^-, \ldots, p_n^-) = -\log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^{n} \exp(\text{sim}(q, p_j^-))}
\]

Contrastive learning

Karpukhin et al. Dense Passage Retrieval for Open-Domain Question Answering. EMNLP 2020.
Training dense retrieval models: DPR

Inner Product Similarity

$$L(q, p^+, p_1^-, p_2^-, \ldots, p_n^-) = -\log \frac{\exp(sim(q, p^+))}{\exp(sim(q, p^+)) + \sum_{j=1}^{n} \exp(sim(q, p_j^-))}$$

Contrastive learning

Karpukhin et al. Dense Passage Retrieval for Open-Domain Question Answering. EMNLP 2020.
Training dense retrieval models: DPR

Inner Product Similarity

Encoder

Query

Text chunks

$$L(q, p^+, p_1^-, p_2^-, \ldots, p_n^-)$$

Positive passage

$$= - \log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^{n} \exp(\text{sim}(q, p_j^-))}$$

Karpukhin et al. Dense Passage Retrieval for Open-Domain Question Answering. EMNLP 2020.
Training dense retrieval models: DPR

Inner Product Similarity

$\text{Encoder}$

$\text{Query}$

$\text{Text chunks}$

$\mathbf{L}(q, p^+, p_1^-, p_2^-, \ldots, p_n^-)$

Positive passage

Too expensive to consider all negatives!

$= - \log \frac{\exp(\text{sim}(q, p^+))}{\exp(\text{sim}(q, p^+)) + \sum_{j=1}^{n} \exp(\text{sim}(q, p_j^-))}$
RAG with LMs using different retrievers

Better retrieval model
Better base LMs
Better retrieval-based LMs
Each component can be improved separately

Independent training

👍 Work with off-the-shelf models (no extra training required)

👍 Each part can be improved independently
Independent training

- Work with off-the-shelf models (no extra training required)
- Each part can be improved independently
- LMs are not trained to leverage retrieval
- Retrieval models are not optimized for LM tasks/domains
Training methods for retrieval-augmented LMs

- Independent training
- **Sequential training**
- Joint training w/ asynchronous index update
- Joint training w/ in-batch approximation
Sequential training

- One component is first trained independently and then fixed
- The other component is trained with an objective that depends on the first one
Sequential training

- **One component** is first trained independently and then fixed
- **The other component** is trained with an objective that depends on the **first one**
Sequential training

- **One component** is first trained independently and then fixed.
- **The other component** is trained with an objective that depends on the **first one**.
Sequential training

- **One component** is first trained independently and then fixed
- **The other component** is trained with an objective that depends on the first one

![Diagram showing sequential training between Retriever and LM, with Retriever first trained independently and then fixed, while LM is trained with an objective that depends on the first one, e.g., RETRO; WebGPT.]
RETRO: Training

- Retrieval Encoder
- Index
- LM Encoder

\[ x_1 \rightarrow \text{Retrieval Encoder} \rightarrow \text{Index} \rightarrow p_1 \ldots p_k \rightarrow \text{LM Encoder} \]

- EMB
- ATTN
- CCA
- FFW
- HEAD

\[ x_1 \rightarrow \text{EMB} \rightarrow \text{ATTN} \rightarrow \text{CCA} \rightarrow \text{FFW} \rightarrow \text{HEAD} \]

\[ x_1 \rightarrow \text{EMB} \rightarrow \text{ATTN} \rightarrow \text{CCA} \rightarrow \text{FFW} \rightarrow \text{HEAD} \]

Back-propagate
RETRO: Training

Updating an index with 600B is extremely expensive!!
Fix the **retrieval encoder** and the **index** during training!
Sequential training

Work with off-the-shelf components (either a large index or a powerful LM)

LMs are trained to effectively leverage retrieval results

Retrievers are trained to provide text that helps LMs the most

One component is still fixed and not trained
Sequential training

✔ Work with off-the-shelf components (either a large index or a powerful LM)

✔ LMs are trained to effectively leverage retrieval results

✔ Retrievers are trained to provide text that helps LMs the most

✔ One component is still fixed and not trained

Let’s jointly train retrieval models and LMs!
Training methods for retrieval-augmented LMs

- Independent training
- Sequential training
- Joint training w/ synchronous index update
- Joint training w/ in-batch approximation
Training methods for retrieval-augmented LMs

- Independent training
- Sequential training
- Joint training w/ asynchronous index update
- Joint training w/ in-batch approximation
Joint training w/ asynchronous index update

- Retrieval models and language models are trained jointly
- Allow the index to be “stale”; rebuild the retrieval index every T steps
Asynchronous index update

Datastore

Encoder

Index

\[ x_1 \]
\[ x_2 \]
\[ x_3 \]
\[ \ldots \]
Asynchronous index update

Datastore

\[ x_1 \]
\[ x_2 \]
\[ x_3 \]
\[ \ldots \]

Updated Encoder

Stale Index
Asynchronous index update

Datastore

\[ x_1, x_2, x_3, \ldots \]

Updated Encoder

Updated Index

\( T \) updates

Refresh
REALM (Guu et al. 2020)

\( x = \text{The [MASK] at the top of the pyramid.} \)

\[ \begin{align*}
q (=x) & \quad \text{The pyramidion on top \ldots the pyramid.} \\
\text{Index} & \quad \ldots \\
& \quad \text{The [MASK] at the top of the pyramid.}
\end{align*} \]

\[ P(z \mid x) \]

\[ P(y \mid x, z) \]

REALM: Training

Objective: \( \maximize \sum_{\mathcal{z} \in \mathcal{I}_\theta} P_\theta(z \mid q)P_\theta(y \mid q, z) \)

\( q (=x) \)

\( \mathcal{I}_\theta \): top-K retrieved chunks

The pyramidion on top allows for less material higher up the pyramid.

The pyramidion on top … the pyramid.

The [MASK] at the top of the pyramid.

\( P_\theta(z \mid x) \)

\( P_\theta(y \mid x, z) \)

REALM: Training

Objective: \( \text{maximize } \sum_{z \in \mathcal{I}_\theta} P_{\theta}(z \mid q)P_{\theta}(y \mid q, z) \)

\( q (=x) \)

The pyramidion on top allows for less material higher up the pyramid.

\( \mathcal{I}_\theta : \text{top-K retrieved chunks} \)

Index

Back-propagation

\( P_{\theta}(z \mid x) \)

The pyramidion on top … the pyramid.

…

The [MASK] at the top of the pyramid.

\( P_{\theta}(y \mid x, z) \)

REALM: Training

Objective: maximize \( \sum_{z \in \mathcal{Z}_\theta} P_\theta(z \mid q)P_\theta(y \mid q, z) \)

Stale index; Update every T steps

\( \mathcal{Z}_\theta \): top-K retrieved chunks

Up-to-date parameters

The pyramidion on top allows for less material higher up the pyramid.

The pyramidion on top … the pyramid.

The [MASK] at the top of the pyramid.

LM
REALM: Index update rate

How often should we update the retrieval index?
- Frequency too high: expensive
- Frequency too slow: out-dated

REALM: updating the index every 500 training steps

Joint training

End-to-end trained — each component is optimized

Good performance

Training is more complicated
   (async update, overhead, data batching, etc)

Train-test discrepancy still remains
Today’s outline

Why do we need retrieval-augmented LMs?

Architectures of retrieval-augmented LMs (Inference)

Training of retrieval-augmented LMs

Limitations and future directions
Challenge: retrieval-augmented LMs for applications

Open-ended text generation? Reasoning?

Doesn’t improve open-ended generation

Failure of retrieval in reasoning task


Challenge: efficiency retrieval-augmented LMs

Additional costs from retrieval augmentation

Retrieval-augmented LMs add inference costs

Mallen*, Asai* et al., When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories (Best Video; Oral) 2023.
Challenge: scaling retrieval-augmented LMs

A small LM + a large datastore \(\approx\) a large parametric LM?

<table>
<thead>
<tr>
<th></th>
<th>LM</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># of parameters</td>
<td># of tokens</td>
</tr>
<tr>
<td>kNN-LM (Khandelwal et al., 2020)</td>
<td>250M</td>
<td>≤ 3B</td>
</tr>
<tr>
<td>NPM (Min et al., 2023)</td>
<td>350M</td>
<td>1B</td>
</tr>
<tr>
<td>Atlas (Izacard et al., 2022)</td>
<td>11B</td>
<td>~30B</td>
</tr>
<tr>
<td>RETRO (Borgeaud et al., 2021)</td>
<td>7B</td>
<td>2T</td>
</tr>
<tr>
<td>REPLUG (Shi et al., 2023)</td>
<td>≤ 175B</td>
<td>~5B</td>
</tr>
</tbody>
</table>
Challenge: robustness and controllability

Retrieval-augmented LMs can still hallucinate

Roadmap to advance retrieval-augmented LMs

Rethink Retrieval and Datastore

Advance Architectures & Retrieval-aware Training

Investment Infrastructures for Training and Inference at Scale
Beyond semantic and lexical-similarity based search

Training retrievers to optimize end-to-end retrieval-augmented LM performance in diverse tasks tasks

Asai et al., Task-aware Retrieval with Instruction. Findings of ACL 2023.

Lin et al., RA-DIT: Retrieval-Augmented Dual Instruction Tuning. ICLR 2024.
Roadmap to advance retrieval-augmented LMs

Rethink Retrieval and Datastore

Investment Infrastructures for Training and Inference at Scale

Advance Architectures & Retrieval-aware Training
New architectures for performance and efficiency

Further explorations of unified architectures & caching

Muennighoff et al. Generative Representational Instruction Tuning. 2024.

Training LMs with Retrieval

Training LMs to learn to use retrieval during pre-training or instruction-tuning

Instruction-tuning with retrieval


Retrieval-aware pre-training

Roadmap to advance retrieval-augmented LMs

- Rethink Retrieval and Datastore
- Advance Architectures & Retrieval-aware Training
- Investment Infrastructures for Training and Inference at Scale
Retrieval-augmented LMs can be really expensive!

Scaling up DS to trillion tokens

End-to-end training w/ Index updates

Inference with many documents
More open-sourced and collaborative opportunities

- System / Algorithmic improvements for massive Datastore
- Standardized implementations for efficient training
- Fast inference algorithms for retrieval-augmented LMs
Retrieval-augmented LMs can solve many issues e.g., hallucinations

Various architectures (not just RAG) exist with different pros & cons

Jointly training retrieval-augmented LMs is important but hard

Many interesting research opportunities — let’s work together!

ACL 2023 tutorial:  https://acl2023-retrieval-lm.github.io/

Contact:  akari@cs.washington.edu
Website:  https://akariasai.github.io/
Twitter:  @AkariAsai
References (1)


References (2)


References (3)


References (5)


