Natural Language Understanding, Generation, and Machine Translation

Lecture 23: Retrieval Augmented Generation

Pasquale Minervini
p.minervini@ed.ac.uk
March 13th, 2024
Open-Domain Question Answering (ODQA):

We do not assume we are given a passage together with the question. We can only access a large collection of documents (e.g., Wikipedia) — we don’t know which document contains the answer, and the goal is to answer any open-domain questions. Both more challenging and more practical/useful!
Open-Domain Question Answering (ODQA):

We do not assume we are given a passage together with the question.
Open-Domain Question Answering (ODQA):

We do not assume we are given a passage together with the question. We can only access a large collection of documents (e.g., Wikipedia) — we don’t know which document contains the answer, and the goal is to answer any open-domain questions.
Open-Domain Question Answering (ODQA):

We do not assume we are given a passage together with the question. We can only access a large collection of documents (e.g., Wikipedia) — we don’t know which document contains the answer, and the goal is to answer any open-domain questions. Both more challenging and more practical/useful!
The KILT Benchmark

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

### Slot Filling

**Input:** Star Trek (SEP) creator  
**Output:** Gene Roddenberry  
**Provenance:** 17157886-1  

**Input:** When did Star Trek go off the air?  
**Output:** June 3, 1969  
**Provenance:** 17157886-5  

**Input:** Which Star Trek star directed Three Men and a Baby?  
**Output:** Leonard Nimoy  
**Provenance:** 17157886-4, 596639-7  

**Input:** Treklanta (formerly "TrekTrax Atlanta") is an annual convention for what American science fiction media franchise?  
**Output:** Star Trek  

### Open Domain QA

**Input:** Star Trek 17157886  
**Output:** Gene Roddenberry  
**Provenance:** 17157886-1  

**Input:** When did Star Trek go off the air?  
**Output:** June 3, 1969  
**Provenance:** 17157886-5  

**Input:** Which Star Trek star directed Three Men and a Baby?  
**Output:** Leonard Nimoy  
**Provenance:** 17157886-4, 596639-7  

**Input:** Treklanta (formerly "TrekTrax Atlanta") is an annual convention for what American science fiction media franchise?  
**Output:** Star Trek  

### Fact Checking

**Input:** Star Trek had spin-off television series.  
**Output:** Supports  
**Provenance:** 17157886-3  

### Dialogue

**Input:** I am a big fan of Star Trek, the American franchise created by Gene Roddenberry. I don't know much about it. When did the first episode air? It debuted in 1966 and aired for 3 seasons on NBC. What is the plot of the show?  
**Output:** William Shatner plays the role of Captain Kirk. He did a great job.  
**Provenance:** 17157886-2  

**Input:** Star Trek had spin-off television series.  
**Output:** Supports  
**Provenance:** 17157886-3  

### Entity Linking

**Input:** Treklanta 2878994  
**Output:** Star Trek  

**Input:** Current site offers five movie collections ranging from $149 for 10 [START_ENT]Star Trek [END_ENT] films to $1,125 for the eclectic Movie Lovers' Collection of 75 movies.  
**Output:** Star Trek
LLMs and their Limitations

LLMs are Extremely Impressive —

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

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

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

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

[Asai et al., 2024]
LLMs and their Limitations

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

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

Top five states are:
1. District of Columbia (DC)
2. New York
3. Massachusetts
4. California
5. Connecticut

1. Factual inaccuracies
2. Difficulty of verification
3. Difficulty of data opt-out
4. Expensive costs to adapt
5. Large model size

[Asai et al., 2024]
LLMs and their Limitations

Asai et al., 2024

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

- Top five states are:
  1. District of Columbia (DC)
  2. New York
  3. Massachusetts
  4. California
  5. Connecticut

- LLMs and, more generally, neural models

  1. Factual inaccuracies
  2. Difficulty of verification
  3. Difficulty of data opt-out
  4. Expensive costs to adapt
  5. Large model size

Retrieval-augmented LMs: Incorporate data at inference

- Top five states are:
  1. DC
  2. New York
  3. Massachusetts
  4. Washington
  5. California

  1. Reduced factual errors
  2. Better attributions
  3. Flexible data opt-in/out
  4. Adaptivity & customizability
  5. Parameter efficiency

[Asai et al., 2024]
The Retriever-Reader Framework

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

[Chen et al., 2017]
The Retriever-Reader Framework

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

[Chen et al., 2017]
The Retriever-Reader Framework

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

[Chen et al., 2017]
“In what city is the University of Edinburgh located?”

“Edinburgh”

[Chen et al., 2017]
The Retriever-Reader Framework

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

**Output:** an answer $A$

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

- **Retriever:** a standard, “classic” TF-IDF information retrieval module (fixed)
- **Reader:** a neural reading comprehension model, trained on SQuAD via distant supervision (i.e., by using retrieved paragraphs rather than gold ones)
The Retriever-Reader Framework

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

Output: an answer $A$

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

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

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

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

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

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

Output: an answer $A$

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

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

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

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

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

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

\[ \text{sim}(Q, P): \text{similarity score between a query } Q \text{ and a paragraph } P \]
Dense and Sparse Retrievers

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

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

**Example:** TF-IDF similarity (sparse)

\[ \text{sim}(Q_i, P_j) = \text{cosine}(q, p) \text{ with } q, p \in \mathbb{R}^{|V|} \]
Dense and Sparse Retrievers

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

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

**Example:** TF-IDF similarity (sparse)

\[
\text{sim}(Q_i, P_j) = \text{cosine}(q, p) \quad \text{with} \quad q, p \in \mathbb{R}^{V}
\]

\[
q_w = \text{TF}(w, Q) \cdot \text{IDF}(w, \mathcal{D})
\]

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

\[
\text{IDF}(w, \mathcal{D}) = \log \frac{|\mathcal{D}|}{|\{P \in \mathcal{D} \land w \in P\}|}
\]
Dense Retrieval in Practice

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

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

**Example:** Dense Retrieval

\[
\sim(Q_i, P_j) = q_i^\top p_j \quad \text{with } q_i, p_j \in \mathbb{R}^d
\]
Dense Retrieval in Practice

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

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

**Example:** Dense Retrieval

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

\( \mathbf{q}_i = \text{Encode}(Q_i) \) \hspace{1cm} \text{Entire research on how to improve or learn the similarity function!}

\( \mathbf{p}_j = \text{Encode}(P_j) \)
**Dense Retrieval in Practice**

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

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

**Index:** given a query embedding \( q_i \in \mathbb{R}^d \), returns the top-\( k \) paragraph embeddings \( p_1, \ldots, p_k \in \mathbb{R}^d \) via **maximum inner-product search** (MIPS)
Software: **FAISS, SCaNN, Annoy,...**

Build index for a collection:

Query:

```
y_1, y_2, \ldots, y_n \in \mathbb{R}^d
```

Indexing

Media description

Index in RAM

Result:

```
k - \text{argmin}_{i=1\ldots n} \|x - y_i\|^2
```
## Summary of methods

The basic indexes are given hereafter:

<table>
<thead>
<tr>
<th>Method</th>
<th>Class name</th>
<th>index factory</th>
<th>Main parameters</th>
<th>Bytes/vector</th>
<th>Exhaustion</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exact Search for L2</strong></td>
<td>IndexFlatL2</td>
<td>&quot;Flat&quot;</td>
<td>d</td>
<td>4+d</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Exact Search for Inner Product</strong></td>
<td>IndexFlatIP</td>
<td>&quot;Flat&quot;</td>
<td>d</td>
<td>4+d</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Hierarchical Navigable Small World graph exploration</strong></td>
<td>IndexIVFFlat</td>
<td>&quot;IVFFlat&quot;</td>
<td>d, M</td>
<td>4d + x * M + 2 * 4</td>
<td>no</td>
</tr>
<tr>
<td><strong>Inverted file with exact post-verification</strong></td>
<td>IndexIVFFlat</td>
<td>&quot;IVFFlat&quot;</td>
<td>quantizer, d, nlists, metric</td>
<td>4d + 8</td>
<td>no</td>
</tr>
<tr>
<td><strong>Locality-Sensitive Hashing (binary flat index)</strong></td>
<td>IndexLSH</td>
<td>-</td>
<td>d, nbits</td>
<td>ceil(nbits/B)</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Scalar quantizer (SQ) in flat mode</strong></td>
<td>IndexScalarQuantizer</td>
<td>&quot;SQB&quot;</td>
<td>d</td>
<td>d</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Product quantizer (PQ) in flat mode</strong></td>
<td>IndexPQ</td>
<td>&quot;PQ&quot;, &quot;PQ&quot;*nbits,</td>
<td>d, M, nbits</td>
<td>ceil(M * nbits / B)</td>
<td>yes</td>
</tr>
</tbody>
</table>

### Software:

- **FAISS, SCaNN, Annoy,**...

### Exact Search

### Approximate Search

(Scales to Billions of vectors)

---

**CPU vs. GPU**
Early method for training the retrieval component proposed by Lee et al., 2019:

Each passage can be encoded as a vector using BERT and the retriever score can be measured as the dot product between the question and passage representations. Not easy to model as there are a huge number of passages (21M in Eng. Wikipedia)
Later, Dense Passage Retrieval [DPR, Karpukhin et al., 2020] authors propose to train the retriever using question-answer pairs:

Trainable retriever (using BERT) can produce more accurate results than traditional IR models, such as BM25 and TF-IDF
Later, Dense Passage Retrieval [DPR, Karpukhin et al., 2020] authors propose to train the retriever using question-answer pairs:

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

...although this was slightly controversial — see e.g., “A Replication Study of DPR” https://arxiv.org/abs/2104.05740
Recent works show that it can be beneficial to **generate answers** rather than **extracting them** from retrieved passages, e.g., Fusion-in-Decoder [Izacard et al., 2021]
Recent works show that it can be beneficial to **generate answers** rather than **extracting them** from retrieved passages, e.g., Fusion-in-Decoder [Izacard et al., 2021]

<table>
<thead>
<tr>
<th>Model</th>
<th>NQ EM</th>
<th>TriviaQA EM</th>
<th>SQuAD Open EM</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>DrQA (Chen et al., 2017)</td>
<td>-</td>
<td>-</td>
<td>29.8</td>
<td>-</td>
</tr>
<tr>
<td>Multi-Passage BERT (Wang et al., 2019)</td>
<td>-</td>
<td>-</td>
<td>53.0</td>
<td>60.9</td>
</tr>
<tr>
<td>Path Retriever (Asai et al., 2020)</td>
<td>31.7</td>
<td>-</td>
<td><strong>56.5</strong></td>
<td><strong>63.8</strong></td>
</tr>
<tr>
<td>Graph Retriever (Min et al., 2019b)</td>
<td>34.7</td>
<td>55.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Hard EM (Min et al., 2019a)</td>
<td>28.8</td>
<td>50.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ORQA (Lee et al., 2019)</td>
<td>31.3</td>
<td>45.1</td>
<td>20.2</td>
<td>-</td>
</tr>
<tr>
<td>REALM (Guu et al., 2020)</td>
<td>40.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DPR (Karpukhin et al., 2020)</td>
<td>41.5</td>
<td>57.9</td>
<td>36.7</td>
<td>-</td>
</tr>
<tr>
<td>SpanSeqGen (Min et al., 2020)</td>
<td>42.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>RAG (Lewis et al., 2020b)</td>
<td>44.5</td>
<td>56.1</td>
<td>68.0</td>
<td>-</td>
</tr>
<tr>
<td>T5 (Roberts et al., 2020)</td>
<td>36.6</td>
<td>60.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GPT-3 few shot (Brown et al., 2020)</td>
<td>29.9</td>
<td>-</td>
<td>71.2</td>
<td>-</td>
</tr>
<tr>
<td>Fusion-in-Decoder (base)</td>
<td><strong>48.2</strong></td>
<td>65.0</td>
<td>77.1</td>
<td>53.4</td>
</tr>
<tr>
<td>Fusion-in-Decoder (large)</td>
<td><strong>51.4</strong></td>
<td><strong>67.6</strong></td>
<td><strong>80.1</strong></td>
<td><strong>56.7</strong></td>
</tr>
</tbody>
</table>
Recent works show that it can be beneficial to **generate answers** rather than **extracting them** from retrieved passages, e.g., Fusion-in-Decoder [Izacard et al., 2021].

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

**Fusion-in-Decoder (FiD): DPR & T5**
LLMs Can Do Open-Domain QA

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

- [Roberts et al., 2020]
LLMs — without an (explicit) retrieval component — can be used to solve Open-Domain Question Answering tasks; knowledge about the world is encoded in their parameters and activations, rather than in a corpus:

[Roberts et al., 2020]
# LLMs vs. RAG — Generalisation

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

<table>
<thead>
<tr>
<th>Model</th>
<th>Open Natural Questions</th>
<th>TriviaQA</th>
<th>WebQuestions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Question Overlap</td>
<td>Answer Overlap Only</td>
</tr>
<tr>
<td></td>
<td>![No Overlap]</td>
<td>![No Overlap]</td>
<td>![No Overlap]</td>
</tr>
</tbody>
</table>

- The question appears in the training set
- The answer appears in the training set

[Lewis et al., 2020]
LLMs vs. RAG — Generalisation

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

<table>
<thead>
<tr>
<th>Model</th>
<th>Open Natural Questions</th>
<th>TriviaQA</th>
<th>WebQuestions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Question Overlap</td>
<td>Answer Overlap Only</td>
</tr>
<tr>
<td>Closed book</td>
<td>36.6</td>
<td>77.2</td>
<td>22.2</td>
</tr>
<tr>
<td>BART</td>
<td>26.5</td>
<td>67.6</td>
<td>10.2</td>
</tr>
</tbody>
</table>

[Lewis et al., 2020]
LLMs vs. RAG — Generalisation

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

<table>
<thead>
<tr>
<th>Model</th>
<th>Open Natural Questions</th>
<th>TriviaQA</th>
<th>WebQuestions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Question Overlap</td>
<td>Answer Overlap Only</td>
</tr>
<tr>
<td>Open book</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAG</td>
<td>44.5</td>
<td>70.7</td>
<td>34.9</td>
</tr>
<tr>
<td>DPR</td>
<td>41.3</td>
<td>69.4</td>
<td>34.6</td>
</tr>
<tr>
<td>FID</td>
<td>51.4</td>
<td>71.3</td>
<td>48.3</td>
</tr>
<tr>
<td>Closed book</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T5-11B+SSM</td>
<td>36.6</td>
<td>77.2</td>
<td>22.2</td>
</tr>
<tr>
<td>BART</td>
<td>26.5</td>
<td>67.6</td>
<td>10.2</td>
</tr>
</tbody>
</table>

[Lewis et al., 2020]
How do LLMs **really** compare with RAG models, in terms of accuracy and generalisation, on open-domain question answering tasks?

<table>
<thead>
<tr>
<th>Model</th>
<th>Natural Questions</th>
<th>TriviaQA</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total Overlap Comp-gen Novel-entity</td>
<td>Total Overlap Comp-gen Novel-entity</td>
<td></td>
</tr>
<tr>
<td>T5-11B+SSM</td>
<td>36.59 81.48</td>
<td>17.47 12.56</td>
<td>-</td>
</tr>
<tr>
<td>BART</td>
<td>26.54 76.34</td>
<td>5.88 3.35</td>
<td>26.78 78.38 11.37 10.09</td>
</tr>
</tbody>
</table>

[![Unseen entity-relation pair](image1.png)](image1.png)

[![New entity](image2.png)](image2.png)

[Liu et al., 2021]
How do LLMs **really** compare with RAG models, in terms of accuracy and generalisation, on open-domain question answering tasks?

<table>
<thead>
<tr>
<th>Model</th>
<th>Natural Questions</th>
<th>TriviaQA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Comp-gen</td>
</tr>
<tr>
<td></td>
<td>Overlap</td>
<td></td>
</tr>
<tr>
<td>Non-parametric</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAG</td>
<td>44.49</td>
<td>30.41</td>
</tr>
<tr>
<td>FiD</td>
<td><strong>53.13</strong></td>
<td><strong>40.00</strong></td>
</tr>
<tr>
<td>DPR</td>
<td>41.27</td>
<td>25.88</td>
</tr>
<tr>
<td>RePAQ</td>
<td>47.26</td>
<td>34.21</td>
</tr>
<tr>
<td>Parametric</td>
<td>T5-11B+SSM</td>
<td>36.59</td>
</tr>
<tr>
<td></td>
<td>BART</td>
<td>26.54</td>
</tr>
</tbody>
</table>

[Liu et al., 2021]
## LLMs vs. RAG — Updateability

<table>
<thead>
<tr>
<th>Train Set</th>
<th>Test-time Index</th>
<th>2017 Test Set Acc.</th>
<th>2020 Test Set Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Closed-book</td>
<td>ATLAS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>T5-XXL</td>
<td>Retrieval-augmented</td>
</tr>
<tr>
<td>2017 answers</td>
<td>2017</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2017 answers</td>
<td>2020</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2020 answers</td>
<td>2017</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2020 answers</td>
<td>2020</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Izacard et al., 2022]
## LLMs vs. RAG — Updateability

<table>
<thead>
<tr>
<th>Train Set</th>
<th>Test-time Index</th>
<th>2017 Test Set Acc.</th>
<th>2020 Test Set Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Closed-book</td>
<td>ATLAS</td>
</tr>
<tr>
<td>2017 answers</td>
<td>2017</td>
<td>12.1</td>
<td>Retrieval-augmented</td>
</tr>
<tr>
<td>2017 answers</td>
<td>2020</td>
<td>12.1</td>
<td>Retrieval-augmented</td>
</tr>
<tr>
<td>2020 answers</td>
<td>2017</td>
<td>4.8</td>
<td>Retrieval-augmented</td>
</tr>
<tr>
<td>2020 answers</td>
<td>2020</td>
<td>4.8</td>
<td>Retrieval-augmented</td>
</tr>
</tbody>
</table>

[Izacard et al., 2022]
## LLMs vs. RAG — Updateability

<table>
<thead>
<tr>
<th>Train Set</th>
<th>Test-time Index</th>
<th>2017 Test Set Acc.</th>
<th>2020 Test Set Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Closed-book</td>
<td>ATLAS</td>
</tr>
<tr>
<td>2017 answers</td>
<td>2017</td>
<td>12.1</td>
<td>57.7</td>
</tr>
<tr>
<td></td>
<td>2020</td>
<td>12.1</td>
<td>10.2</td>
</tr>
<tr>
<td>2020 answers</td>
<td>2017</td>
<td>4.8</td>
<td>50.1</td>
</tr>
<tr>
<td></td>
<td>2020</td>
<td>4.8</td>
<td>3.5</td>
</tr>
</tbody>
</table>

[Izacard et al., 2022]
## LLMs vs. RAG — Accuracy

<table>
<thead>
<tr>
<th>Model</th>
<th>NQ</th>
<th>TriviaQA filtered</th>
<th>TriviaQA unfiltered</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>64-shot</td>
<td>Full</td>
<td>64-shot</td>
</tr>
<tr>
<td>GPT-3 (Brown et al., 2020)</td>
<td>29.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Gopher (Rae et al., 2021)</td>
<td>28.2</td>
<td>-</td>
<td>57.2</td>
</tr>
<tr>
<td>Chinchilla (Hoffmann et al., 2022)</td>
<td>35.5</td>
<td>-</td>
<td>64.6</td>
</tr>
<tr>
<td>PaLM (Chowdhery et al., 2022)</td>
<td>39.6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>RETRO (Borgeaud et al., 2021)</td>
<td>-</td>
<td>45.5</td>
<td>-</td>
</tr>
<tr>
<td>FiD (Izacard &amp; Grave, 2020)</td>
<td>-</td>
<td>51.4</td>
<td>-</td>
</tr>
<tr>
<td>FiD-KD (Izacard &amp; Grave, 2021)</td>
<td>-</td>
<td>54.7</td>
<td>-</td>
</tr>
<tr>
<td>R2-D2 (Fajcik et al., 2021)</td>
<td>-</td>
<td>55.9</td>
<td>-</td>
</tr>
</tbody>
</table>

[Izacard et al., 2022]
## LLMs vs. RAG — Accuracy

<table>
<thead>
<tr>
<th>Model</th>
<th>NQ</th>
<th>TriviaQA filtered</th>
<th>TriviaQA unfiltered</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>64-shot</td>
<td>Full</td>
<td>64-shot</td>
</tr>
<tr>
<td>GPT-3 (Brown et al., 2020)</td>
<td>29.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Gopher (Rae et al., 2021)</td>
<td>28.2</td>
<td>-</td>
<td>57.2</td>
</tr>
<tr>
<td>Chinchilla (Hoffmann et al., 2022)</td>
<td>35.5</td>
<td>-</td>
<td>64.6</td>
</tr>
<tr>
<td>PaLM (Chowdhery et al., 2022)</td>
<td>39.6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>RETRO (Borgeaud et al., 2021)</td>
<td>-</td>
<td>45.5</td>
<td>-</td>
</tr>
<tr>
<td>FiD (Izacard &amp; Grave, 2020)</td>
<td>-</td>
<td>51.4</td>
<td>-</td>
</tr>
<tr>
<td>FiD-KD (Izacard &amp; Grave, 2021)</td>
<td>-</td>
<td>54.7</td>
<td>-</td>
</tr>
<tr>
<td>R2-D2 (Fajcik et al., 2021)</td>
<td>-</td>
<td>55.9</td>
<td>-</td>
</tr>
<tr>
<td><strong>ATLAS</strong></td>
<td>42.4</td>
<td>60.4</td>
<td>74.5</td>
</tr>
</tbody>
</table>

[Izacard et al., 2022]
ATLAS — A Retrieval-Augmented LM

Masked Language Modelling:
Bermuda Triangle is in the <MASK> of the Atlantic Ocean.

The Bermuda Triangle is an urban legend focused on a loosely-defined region in the western part of the North Atlantic Ocean.

[Izacard et al., 2022]
Fact checking: Bermuda Triangle is in the western part of the Himalayas.

Masked Language Modelling: Bermuda Triangle is in the <MASK> of the Atlantic Ocean.

Question answering: Where is the Bermuda Triangle?

The Bermuda Triangle is an urban legend focused on a loosely-defined region in the western part of the North Atlantic Ocean.

Pretraining

Few-shot

ATLAS — A Retrieval-Augmented LM

[2022] Izacard et al.
Retrieval-Augmented LMs

**What** to retrieve?

- Query

Text chunks (passages)?
- Tokens?
- Something else?
Retrieval-Augmented LMs

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

How to use retrieval?
- Input
- LM
- Output
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.
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.
Retrieval-Augmented LMs

**What** to retrieve?

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

**How** to use retrieval?

1. **Input**
2. **LM**
3. **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**
  - The capital city of Ontario is Toronto.
Retrieval-Augmented LMs

What to retrieve? → Text chunks → Input layer (concatenation) → When to retrieve?

How to use retrieval? → Once

REALM (Guu et al. 2020)

[Asai et al., 2023]
REALM [Guu et al., 2020]

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

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

LM

48

Read stage
REALM [Guu et al., 2020]

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

$x$

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

Retrieval

LM

48

Read stage
REALM [Guu et al., 2020]

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

\[x\]

\(x\)

\[\text{Retrieval}\]

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

\(k\) chunks of text (passages)

FIFA World Cup 2026 will expand to 48 teams.

\[\text{Retrieving stage} \]

\[\text{Read stage}\]

48
REALM [Guu et al., 2020]

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

$\rightarrow$ FIFA World Cup 2026 will expand to 48 teams.

$\rightarrow$ World Cup 2022 was … the increase to [MASK] in 2026.

$k$ chunks of text (passages)

Retrieval

FIFA World Cup 2026 will expand to 48 teams.

Retrieve stage

LM

48

Read stage
REALM — Retrieval

FIFA World Cup 2026 will expand to 48 teams.

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

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

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

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

\[ z = \text{Encoder}(z) \]
\[ x = \text{Encoder}(x) \]
REALM — Retrieval

$x =$ World Cup 2022 was … the increase to [MASK] in 2026.

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 …

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

$k$ retrieved chunks

Wikipedia
13M chunks (passages) (called documents in the paper)
REALM — Reading

\[[\text{MASK}] \, z_1 \, [\text{SEP}] \, x \quad \rightarrow \quad \text{LM} \quad \rightarrow \quad P(y \mid x, z_1)\]

\[[\text{MASK}] \, z_2 \, [\text{SEP}] \, x \quad \rightarrow \quad \text{LM} \quad \rightarrow \quad P(y \mid x, z_2)\]

\[\vdots\]

\[[\text{MASK}] \, z_k \, [\text{SEP}] \, x \quad \rightarrow \quad \text{LM} \quad \rightarrow \quad P(y \mid x, z_k)\]
REALM — Reading

\[ [\text{MASK}] z_1 \ [\text{SEP}] x \rightarrow \text{LM} \rightarrow P(y | x, z_1) \]

\[ [\text{MASK}] z_2 \ [\text{SEP}] x \rightarrow \text{LM} \rightarrow P(y | x, z_2) \]

\[ \vdots \]

\[ [\text{MASK}] z_k \ [\text{SEP}] x \rightarrow \text{LM} \rightarrow P(y | x, z_k) \]

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

Weighted average

Need to approximate
→ Consider top \( k \) chunks only

0 if not one of top \( k \)

from the retrieve stage
from the read stage
REALM [Guu et al., 2020]

What to retrieve?

- Chunks ✓
- Tokens
- Others
REALM [Guu et al., 2020]

**What to retrieve?**
- Chunks ✔
- Tokens
- Others

**How to use retrieval?**
- Input layer ✔
- Intermediate layers
- Output layer
# REALM [Guu et al., 2020]

<table>
<thead>
<tr>
<th><strong>What to retrieve?</strong></th>
<th><strong>How to use retrieval?</strong></th>
<th><strong>When to retrieve?</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>- Chunks ✓</td>
<td>- Input layer ✓</td>
<td>- Once ✓</td>
</tr>
<tr>
<td>- Tokens</td>
<td>- Intermediate layers</td>
<td>- Every $n$ tokens ($n&gt;1$)</td>
</tr>
<tr>
<td>- Others</td>
<td>- Output layer</td>
<td>- Every token</td>
</tr>
</tbody>
</table>
Overview — Retrieval-Augmented Models

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

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

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

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