Section 5: Applications
Downstream adaptation of retrieval-based LMs

The capital city of Ontario is __

Datastore → LM → Toronto, which is known for ...
Downstream adaptation of retrieval-based LMs

What are the tasks?

Open-domain QA

What is the capital of Ontario?

Datastore

LM

Toronto
Downstream adaptation of retrieval-based LMs

What are the tasks?

Fact verification

Ottawa is the Ontario state capital.
A range of target tasks

<table>
<thead>
<tr>
<th>Question Answering</th>
<th>Fact verification</th>
<th>Dialogue</th>
</tr>
</thead>
<tbody>
<tr>
<td>RETRO (Borgeaud et al., 2021)</td>
<td>RAG (Lewis et al, 2020)</td>
<td>BlenderBot3 (Shuster et al., 2022)</td>
</tr>
<tr>
<td>REALM (Gu et al, 2020)</td>
<td>ATLAS (Izacard et al, 2022)</td>
<td>Internet-augmented generation (Komeili et al., 2022)</td>
</tr>
</tbody>
</table>

Retrieval-based LMs have been extensively evaluated on knowledge-intensive tasks
# A range of target tasks

<table>
<thead>
<tr>
<th>Question answering</th>
<th>Fact verification</th>
<th>Dialogue</th>
</tr>
</thead>
<tbody>
<tr>
<td>RETRO (Borgeaud et al., 2021)</td>
<td>RAG (Lewis et al., 2020)</td>
<td>BlenderBot3 (Shuster et al., 2022)</td>
</tr>
<tr>
<td>REALM (Gu et al, 2020)</td>
<td>ATLAS (Izacard et al, 2022)</td>
<td>Internet-augmented generation (Komeili et al., 2022)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Summarization</th>
<th>Machine translation</th>
<th>Code &amp; proof generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLARE (Jiang et al, 2023)</td>
<td>kNN-MT (Khandelwal et al., 2020)</td>
<td>DocPrompting (Zhou et al., 2023)</td>
</tr>
<tr>
<td></td>
<td>TRIME-MT (Zhong et al., 2022)</td>
<td>Natural Prover (Welleck et al., 2022)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NLI</th>
<th>Sentiment analysis</th>
<th>Commonsense reasoning</th>
</tr>
</thead>
<tbody>
<tr>
<td>kNN-Prompt (Shi et al., 2022)</td>
<td>kNN-Prompt (Shi et al., 2022)</td>
<td>Raco (Yu et al, 2022)</td>
</tr>
<tr>
<td>NPM (Min et al., 2023)</td>
<td>NPM (Min et al., 2023)</td>
<td></td>
</tr>
</tbody>
</table>

More general NLP tasks:

- Real-world association (REALM)
- Question answering (REALM, RETRO)
- Sentiment analysis (NPM, kNN-Prompt, Raco)
- Commonsense reasoning (Raco)
- Code & proof generation (DocPrompting)
- Fact verification (Evi. Generator, RAG)
- Machine translation (kNN-MT, TRIME-MT)
- Summarization (FLARE)
- Dialogue (BlenderBot3)
- Natural Prover (Welleck et al.)
A range of target tasks

Question answering
- RETRO (Borgeaud et al., 2021)
- REALM (Gu et al., 2020)
- ATLAS (Izacard et al., 2023)

Fact verification
- RAG (Lewis et al., 2020)
- ATLAS (Izacard et al., 2022)
- Evi. Generator (Asai et al., 2022)

Dialogue
- BlenderBot3 (Shuster et al., 2022)
- Internet-augmented generation (Komeili et al., 2022)

Summarization
- FLARE (Jiang et al., 2023)

Machine translation
- kNN-MT (Khandelwal et al., 2020)
- TRIME-MT (Zhong et al., 2022)

Code & proof generation
- DocPrompting (Zhou et al., 2023)
- Natural Prover (Welleck et al., 2022)

NLI
- kNN-Prompt (Shi et al., 2022)
- NPM (Min et al., 2023)

Sentiment analysis
- kNN-Prompt (Shi et al., 2022)
- NPM (Min et al., 2023)

Commonsense reasoning
- Raco (Yu et al., 2022)

More generations
A range of target tasks

- **Question answering**
  - RETRO (Borgeaud et al., 2021)
  - REALM (Gu et al., 2020)
  - ATLAS (Izacard et al., 2023)

- **Fact verification**
  - RAG (Lewis et al., 2020)
  - ATLAS (Izacard et al., 2022)
  - Evi. Generator (Asai et al., 2022)

- **Dialogue**
  - BlenderBot3 (Shuster et al., 2022)
  - Internet-augmented generation (Komeili et al., 2022)

- **Summarization**
  - FLARE (Jiang et al., 2023)

- **Machine translation**
  - kNN-MT (Khandelwal et al., 2020)
  - TRIME-MT (Zhong et al., 2022)

- **Code & proof generation**
  - DocPrompting (Zhou et al., 2023)
  - Natural Prover (Welleck et al., 2022)

- **NLI**
  - kNN-Prompt (Shi et al., 2022)
  - NPM (Min et al., 2023)

- **Sentiment analysis**
  - kNN-Prompt (Shi et al., 2022)
  - NPM (Min et al., 2023)

- **Commonsense reasoning**
  - Raco (Yu et al., 2022)

More classifications
Two key questions for downstream adaptations

**How** can we adapt a retrieval-based LM for a task?

**When** should we use a retrieval-based LM?
How to adapt a retrieval-based LM for a task

What are the tasks?

- Open-domain QA
- Other knowledge-intensive tasks
- Sentiment analysis
- Code generation

How to adapt?

- Fine-tuning
- Reinforcement learning
- Prompting
How to adapt a retrieval-based LM for a task

Fine-tuning (+RL)

Training LM and/or retriever on task-data & data store
How to adapt a retrieval-based LM for a task

**Fine-tuning (+RL)**
Training LM and/or retriever on task-data & data store

Should we update both retriever & LM?

**Prompting**
Prompt a frozen LM with retrieved knowledge

Where to incorporate?

How to prompt LM?
How to adapt a retrieval-based LM for a task

What are the tasks?
- Open-domain QA
- Other knowledge-intensive tasks
- Sentiment analysis
- Code generation
...

How to adapt?
- Fine-tuning
- Reinforcement learning
- Prompting

What is data store?
- Wikipedia
- Training data
- Code documentation
When to use a retrieval-based LM

Long-tail knowledge update Verifiability Parameter-efficiency
Effectiveness of retrieval-based LMs

Q: Is Toronto really cold during winter?

Yes it is.
Effectiveness of retrieval-based LMs

Q: Where is Toronto Zoo located?

LM

1361A Old Finch Avenue, in Scarborough, Ontario
Effectiveness of retrieval-based LMs

Q: Where is Toronto Zoo located?

LM

361A Old Finch Avenue, in Scarborough, Ontario

Retriever

Toronto zoo Info
Location: 361A Old Finch Avenue, Toronto, Ontario
Land Area: 287 hectares
Q: What is the population of Toronto Metropolitan area in 2023?

A: 6,255,000

Trained on the 2021 corpus
Effectiveness of retrieval-based LMs

**Q:** What is the population of Toronto Metropolitan area in 2023?

**A:** 6,372,000

LM

Trained on the 2021 corpus

Retriever

Collected in 2023
Effectiveness of retrieval-based LMs

Q: Where is Toronto Zoo located?

361A Old Finch Avenue, in Scarborough, Ontario

Retriever

LM

Toronto zoo Info
Location: 361A Old Finch Avenue, Toronto, Ontario
Land Area: 287 hectares
Effectiveness of retrieval-based LMs

Long-tail
knowledge update
Verifiability
Parameter-efficiency

Retriever

LM

LM
Two key questions for downstream adaptations

How can we adapt a retrieval-based LM for a task?

When should we use a retrieval-based LM?
Downstream adaptation of retrieval-based LMs

What are the tasks?
- Open-domain QA
- Other knowledge-intensive tasks
- General NLU
- Language Modeling & other generation tasks

How to adapt?
- **Fine-tuning**
  - Reinforcement learning
  - Prompting

What is data store?
- Unlabeled Wikipedia / CC
- Web (Google / Bing Search Results)
- Training data
Adapting retrieval-based LMs for tasks

Fine-tuning

Training LM and/or retriever on task-data & data store
Adapting retrieval-based LMs for tasks

Fine-tuning

Training LM and / or retriever on task-data & data store

Costs of retrieval-based LM training (Section 4)

Independent training (DPR)
Asynchronous updates (REALM)
…
ATLAS (Izacard et al., 2022; Section 4)

Few-shot task data (e.g., k=64)

- QA: $(x_{i}^{QA}, y_{i}^{QA})$
- Fact verification: $(x_{i}^{FV}, y_{i}^{FV})$
- Dialogue: $(x_{i}^{dial}, y_{i}^{dial})$

Izacard et al. 2022. “Few-shot learning with retrieval augmented language models”
ATLAS (Izacard et al., 2022; Section 4)

Input $x$ 

Query $q$ 

Query encoder

Output $y$

Few-shot task data (e.g., $k=64$)

- QA: $(x_{i}^{QA}, y_{i}^{QA})$
- Fact verification: $(x_{i}^{FV}, y_{i}^{FV})$
- Dialogue: $(x_{i}^{dial}, y_{i}^{dial})$

Dialog $d_i$

Docs $z$

Back-propagate

Doc encoder

Index

Query encoder

LM
ATLAS (Izacard et al., 2022; Section 4)

Fully updating index during training is expensive!

Input $x$ 

LM

Output $y$

Few-shot task data (e.g., $k=64$)

- QA $(x_{i}^{QA}, y_{i}^{QA})$
- Fact verification $(x_{i}^{FV}, y_{i}^{FV})$
- Dialogue $(x_{i}^{dial}, y_{i}^{dial})$
ATLAS: Fixed retrieval with fine-tuned LM

Index is fixed

Index

No back-prop

Doc encoder

Input $x$

Query $q$

Query encoder

LM

+ Doc encoder

Output $y$

Docs $z$

Few-shot task data (e.g., $k=64$)

QA

$(x_{i}^{QA}, y_{i}^{QA})$

Fact verification

$(x_{i}^{FV}, y_{i}^{FV})$

Dialogue

$(x_{i}^{dial}, y_{i}^{dial})$
ATLAS: Query-side fine-tuning

Input $x$ → Query $q$ → Query encoder → Index → Doc encoder → Output $y$

Few-shot task data (e.g., $k=64$)
- QA $(x_i^{QA}, y_i^{QA})$
- Fact verification $(x_i^{FV}, y_i^{FV})$
- Dialogue $(x_i^{dial}, y_i^{dial})$

No back-prop
Ablations of efficient retrieval training

Fixed FT shows large performance drop on QA.
Ablations of efficient retrieval training

Query-side fine-tuning matches or outperforms full fine-tuning
ATLAS: Few-shot v.s. full v.s. transfer setups

Few-shot task data (e.g., $k = 64$)

NQ

$((x_1^{QA}, y_1^{QA}), \ldots, (x_k^{QA}, y_k^{QA}))$

MMLU

$((x_1^{MMNU}, y_1^{MMNU}), \ldots, (x_k^{MMNU}, y_k^{MMNU}))$

Full-shot task data (e.g., $N = 70,000$)

NQ

$((x_1^{QA}, y_1^{QA}), \ldots, (x_N^{QA}, y_N^{QA}))$

Transfer task data (e.g., MCTest $\rightarrow$ MMLU)

MCTest

$((x_1^{TF}, y_1^{TF}), \ldots, (x_N^{TF}, y_N^{TF}))$


On QA, ATLAS largely outperforms other LLMs in few-shot.
Full-shot fine-tuning further improves performance
On MMLU, ATLAS few-shot largely underperforms Chinchilla / GPT-4.

Room for improvements for diverse task adaptations!
Transferring from relevant tasks give large improvements, matching Chinchilla.
### Summary of downstream adaptations

<table>
<thead>
<tr>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge-intensive</td>
<td>Fine-tuning (Retriever &amp; LM)</td>
<td>Wikipedia</td>
</tr>
</tbody>
</table>

Fine-tuning for QA & knowledge-intensive tasks often gives strong performance (even in few-shot)
## Summary of downstream adaptations

<table>
<thead>
<tr>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Izacard et al., 2022)</td>
<td>Knowledge-intensive</td>
<td>Wikipedia</td>
</tr>
</tbody>
</table>

Fine-tuning a retriever for a task matters!
Downstream adaptation of retrieval-based LMs

What are the tasks?
- Open-domain QA
- Other knowledge-intensive tasks
- General NLU
- Language Modeling & other generation tasks

How to adapt?
- Fine-tuning
- Reinforcement learning
- Prompting

What is data store?
- Unlabeled Wikipedia / CC
- Web (Google / Bing Search Results)
- Training data
GopherCite (Menick et al., 2022)

User: What kind of animal is Scooby from Scooby Doo?

GopherCite: A Great Dane dog.

Menick et al. 2022. “GopherCite: Teaching language models to support answers with verified quotes”
GopherCite (Menick et al., 2022)

User: What kind of animal is Scooby from Scooby Doo?

GopherCite: A Great Dane dog.

Wikipedia Page: Scooby-Doo
This Saturday–morning cartoon series featured teenagers Fred Jones, Daphne Blake, Velma Dinkley, and Shaggy Rogers, and their talking Great Dane named Scooby-Doo.

Extract and generate a quote to support an answer

Menick et al. 2022. “GopherCite: Teaching language models to support answers with verified quotes”
GopherCite: RLHF for answering with verified quotes

Index

$\mathbf{q}$

$\mathbf{z}$

LM

$x$

$y_1$

Supervised fine-tuning (SFT)

Off-the-shelf Google Search

We cannot update

$\mathbf{(x, y)}$
GopherCite: RLHF for answering with verified quotes

Model generated training data filtered by human

\[ y = \langle \text{Claim} \rangle \langle \text{Document title} \rangle \langle \text{Quote from document} \rangle \]
GopherCite: RLHF for answering with verified quotes

Reinforcement Learning with human feedback (e.g., Instruct GPT)

Index

LM

Reward Model

Off-the-shelf Google Search

Human preference data

(x, y)

(x, y^1, y^2, r)
GopherCite: RLHF for answering with verified quotes

$$(x, y^1, y^2) \rightarrow \text{Plausible as an answer to the input} \rightarrow r \in y_1, y_2$$

Human rater

Supported by accompanying evidence

33k Human preference data

$$(x, y^1, y^2, r)$$
Effects of RL

RL w/ human feedback improves the quality of top 1 generations

(S&P = Supported & Plausible)
Effects of RL

Sampling & reranking many generations using a reward model gives gains from Top 1
Summary of downstream adaptations

<table>
<thead>
<tr>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Izacard et al., 2022)</td>
<td>Knowledge-intensive</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>GopherCite (Menick et al., 2022), also WebGPT (Nakano et al., 2021)</td>
<td>Open-domain QA, Long-form QA</td>
<td>Google Search Results</td>
</tr>
<tr>
<td></td>
<td>Fine-tuning (Retriever &amp; LM)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fine-tuning + RL (LM)</td>
<td></td>
</tr>
</tbody>
</table>

Benefit of **fine-tuning**

- Customizable
- Competitive w/ more data
- Requiring training
### Summary of downstream adaptations

<table>
<thead>
<tr>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Izacard et al., 2022)</td>
<td>Knowledge-intensive</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>GopherCite (Menick et al., 2022), also WebGPT (Nakano et al., 2021)</td>
<td>Open-domain QA, Long-form QA</td>
<td>Fine-tuning + RL (LM)</td>
</tr>
</tbody>
</table>

#### Benefit of RL
- Better alignment with user preferences
- Requiring additional data collection (preference)
## Summary of downstream adaptations

<table>
<thead>
<tr>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Izacard et al., 2022)</td>
<td>Knowledge-intensive</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>GopherCite (Menick et al., 2022), also WebGPT (Nakano et al., 2021)</td>
<td>Open-domain QA, Long-form QA</td>
<td>Google Search Results</td>
</tr>
<tr>
<td></td>
<td>Fine-tuning (Retriever &amp; LM)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fine-tuning + RL (LM)</td>
<td></td>
</tr>
</tbody>
</table>

What if we cannot train LMs for downstream tasks? (e.g., lack of computational resources / proprietary LM ... etc)
Downstream adaptation of retrieval-based LMs

What are the tasks?
- Open-domain QA
- Other knowledge-intensive tasks
- General NLU
- Language Modeling & other generation tasks

How to adapt?
- Fine-tuning
- Reinforcement learning
- Prompting

What is data store?
- Wikipedia
- Web (Google / Bing Search Results)
- Training data
Prompting

$k$-shot instances (k=0, 32 … etc)

Q: who Is the US president
A: Joe Biden
##
Q: What is the capital of US?
A: Washington DC.
##
Q: what is the Ontario capital?
A:

Training instances (demonstrations)

Doesn’t require LM training on tasks!

Test instances

LM

Toronto
Retrieval-based prompting

$k$-shot instances (k=0, 32 … etc)

Q: who Is the US president
A: Joe Biden

##

Q: What is the capital of US?
A: Washington DC.

##

Q: what is the Ontario capital?
A:
Design choice of retrieval-based Prompting

**Input space:**
Incorporate retrieved context in input space

**Intermediate layers:**
N/A

**Output space:**
Interpolate token probability distributions in output space
Design choice of retrieval-based Prompting

**Input space:**
Incorporate retrieved context in input space

**Intermediate layers:**
N/A

**Output space:**
Interpolate token probability distributions in output space

Extending kNN-LM for downstream tasks
**kNN-Prompt** (Shi et al., 2022)

kNN LM with fuzzy verbalizers for zero-/few-shot **classification**

---

Shi et al. 2022. “Nearest Neighbor Zero-shot Inference”
LM predicts next token

Test example + prompt:
\[ p(x) \]
Mr. Tsai is one of world cinema's most gifted artists. It was

**Verbalizer:**
\[ V(y^+) = \text{great} \]
\[ V(y^-) = \text{terrible} \]

<table>
<thead>
<tr>
<th>token</th>
<th>( P_{LM} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>great</td>
<td>0.2</td>
</tr>
<tr>
<td>terrible</td>
<td>0.6</td>
</tr>
<tr>
<td>good</td>
<td>0.1</td>
</tr>
</tbody>
</table>
**kNN-Prompt** (Shi et al., 2022)

<table>
<thead>
<tr>
<th>Leftward Contexts $C_i$</th>
<th>Datastore</th>
<th>Nearest $k_i$</th>
<th>$P_{kNN}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watching this movie makes me feel dumb. It is...</td>
<td>terrible</td>
<td>great</td>
<td>0.4</td>
</tr>
<tr>
<td>One of the best movies I've ever seen. The story is...</td>
<td>good</td>
<td>good</td>
<td>0.4</td>
</tr>
<tr>
<td>Performed by the greatest artist. Thriller is the...</td>
<td>great</td>
<td>terrible</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Test example + prompt:

$p(x)$

Mr. Tsai is one of world cinema’s most gifted artists. It was

**Verbalizer:**

$V(y^+) = great$

$V(y^-) = terrible$

**kNN predicts next tokens**

<table>
<thead>
<tr>
<th>token</th>
<th>$P_{LM}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>great</td>
<td>0.2</td>
</tr>
<tr>
<td>terrible</td>
<td>0.6</td>
</tr>
<tr>
<td>good</td>
<td>0.1</td>
</tr>
</tbody>
</table>
kNN-Prompt (Shi et al., 2022)

<table>
<thead>
<tr>
<th>Left-ward Contexts</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_i$</td>
<td></td>
</tr>
</tbody>
</table>

- Watching this movie makes me feel dumb. It is terrible.
- One of the best movies I’ve ever seen. The story is good.
- Performed by the greatest artist. Thriller is the great.

Datastore:

<table>
<thead>
<tr>
<th>Key $f(C_i)$</th>
<th>Value $v_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>terrible</td>
</tr>
<tr>
<td></td>
<td>good</td>
</tr>
<tr>
<td></td>
<td>great</td>
</tr>
</tbody>
</table>

Nearest $k_i$ $P_{kNN}$

- great: 0.4
- good: 0.4
- terrible: 0.2

Interpolate output token probability

Test example + prompt:
$p(x)$
Mr. Tsai is one of world cinema’s most gifted artists. It was
great

Verbalizer:

$V(y^+)$ = great

$V(y^-)$ = terrible

LM

<table>
<thead>
<tr>
<th>token</th>
<th>$P_{LM}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>great</td>
<td>0.2</td>
</tr>
<tr>
<td>terrible</td>
<td>0.6</td>
</tr>
<tr>
<td>good</td>
<td>0.1</td>
</tr>
</tbody>
</table>

kNN-LM (Khandelwal et al., 2020)
The kNN token distributions are quite sparse!

kNN-Prompt (Shi et al., 2022)

Left-ward Contexts $c_i$

- Watching this movie makes me feel dumb. It is terrible.
- One of the best movies I've ever seen. The story is good.
- Performed by the greatest artist. Thriller is the great.

Datastore

<table>
<thead>
<tr>
<th>Key $f(c_i)$</th>
<th>Value $v_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watching</td>
<td>terrible</td>
</tr>
<tr>
<td>One</td>
<td>good</td>
</tr>
<tr>
<td>Performed</td>
<td>great</td>
</tr>
</tbody>
</table>

Nearest $k_i$ $p_{kNN}$

<table>
<thead>
<tr>
<th>$k_i$</th>
<th>$p_{kNN}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>great</td>
<td>0.4</td>
</tr>
<tr>
<td>good</td>
<td>0.4</td>
</tr>
<tr>
<td>terrible</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Test example + prompt:

$p(x)$ Mr. Tsai is one of world cinema’s most gifted artists. It was

Verbalizer:

$V(y^+) = \text{great}$
$V(y^-) = \text{terrible}$

LM

<table>
<thead>
<tr>
<th>token</th>
<th>$p_{LM}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>great</td>
<td>0.2</td>
</tr>
<tr>
<td>terrible</td>
<td>0.6</td>
</tr>
<tr>
<td>good</td>
<td>0.1</td>
</tr>
</tbody>
</table>
### kNN-Prompt (Shi et al., 2022)

The kNN token distributions are quite sparse!

<table>
<thead>
<tr>
<th>Left-ward Contexts $c_i$</th>
<th>Datastore</th>
<th>Nearest $k_i$</th>
<th>$P_{kNN}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watching this movie makes me feel dumb. It is terrible.</td>
<td>terrible</td>
<td>great 0.4</td>
<td></td>
</tr>
<tr>
<td>One of the best movies I’ve ever seen. The story is good.</td>
<td>good</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>Performed by the greatest artist. Thriller is the great.</td>
<td>great</td>
<td>terrible 0.2</td>
<td></td>
</tr>
</tbody>
</table>

**Test example + prompt:**

$p(x)$

Mr. Tsai is one of world cinema’s most gifted artists. It was

**Verbalizer:**

$V(y^+) = \text{great}$

$V(y^-) = \text{terrible}$

<table>
<thead>
<tr>
<th>token</th>
<th>$P_{LM}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>great</td>
<td>0.2</td>
</tr>
<tr>
<td>terrible</td>
<td>0.6</td>
</tr>
<tr>
<td>good</td>
<td>0.1</td>
</tr>
</tbody>
</table>
kNN-Prompt (Shi et al., 2022)

Fuzzy verbalizer maps token probability to target class labels

\[ P_{\text{FV}}(y \mid x) \propto \sum_{v_i \in N(v)} P(v_i \mid p(x)) \]

Find similar tokens using GloVe & ConceptNet
Results on diverse classification tasks

Accuracy

<table>
<thead>
<tr>
<th></th>
<th>LM</th>
<th>kNN-LM</th>
<th>kNN-Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTE</td>
<td>45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SST-2</td>
<td>67.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AGNews</td>
<td>90</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Significant gains from kNN-LM

NLI / entailment
Sentiment analysis
Topic classification
Results on diverse classification tasks

Accuracy

- **LM**
- **kNN-LM**
- **kNN-Prompt**

**RTE**
**SST-2**
**AGNews**

- **NLI / entailment**
- **Sentiment analysis**
- **Topic classification**

kNN-prompt largely outperforms vanilla LM in zero-shot classification
**Summary of downstream adaptations**

<table>
<thead>
<tr>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Izacard et al., 2022)</td>
<td>Knowledge-intensive</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>GopherCite (Menick et al., 2022)</td>
<td>Open-domain QA, Long-form QA</td>
<td>Google Search Results</td>
</tr>
<tr>
<td>kNN-prompt (Shi et al., 2022)</td>
<td>Classification</td>
<td>Prompting (output)</td>
</tr>
<tr>
<td></td>
<td>Fine-tuning (Retriever &amp; LM)</td>
<td>Wikipedia</td>
</tr>
<tr>
<td></td>
<td>Fine-tuning + RL (LM)</td>
<td></td>
</tr>
</tbody>
</table>

Retrieval-based LMs are effective in general NLU tasks!
Retrieval-based Prompting

**Input space:**  
Incorporate retrieved context in input space

**Intermediate layers:**  
N/A

**Output space:**  
Interpolate token probability distributions in output space
What is the capital of Ontario?

Toronto is in Canada. It is the capital city of the province of Ontario.

(Shi et al., 2023; Ram et al., 2022; Mallen et al., 2022; Yu et al., 2022; Press et al., 2022; inter alia)
What is the capital of Ontario?

- Toronto is in Canada. It is the capital city of the province of Ontario.
What is the capital of Ontario?

Retriever

\( \mathcal{X} \)

Toronto is in Canada. It is the capital city of the province of Ontario.

Ontario is home to the nation’s capital city, Ottawa, and the most populous city Toronto.

Top 10 documents
What is the capital of Ontario?

- Toronto is in Canada. It is the capital city of the province of Ontario.
- Ontario is home to the nation’s capital city, Ottawa, and the most populous city Toronto.

\[ \sum_i p(\text{Toronto} \mid x, p_i) > \sum_i p(\text{Ottawa} \mid x, p_i) \]

Top 10 documents
REPLUG: Results on QA & MMLU

Large performance gain from base LM

Base LM (CodeX) vs + REPLUG LSR

MMLU: +3.5
NQ: +4.6
REPLUG: Comparison with ATLAS

Outperforms ATLAS in few-shot, especially in MMLU

Acc.

<table>
<thead>
<tr>
<th></th>
<th>MMLU</th>
<th>NQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Few-shot)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPLUG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATLAS (Full / Transfer)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- REPLUG: Comparison with ATLAS
- Outperforms ATLAS in few-shot, especially in MMLU
REPLUG: Comparison with ATLAS

ATLAS (Full / Transfer) outperforms REPLUG

Acc.

MMLU  NQ

ATLAS (Few-shot)  REPLUG  ATLAS (Full / Transfer)
## Summary of downstream adaptations

<table>
<thead>
<tr>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Izacard et al., 2022)</td>
<td>Knowledge-intensive Fine-tuning (Retriever &amp; LM)</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>GopherCite (Menick et al., 2022)</td>
<td>Open-domain QA, Long-form QA Fine-tuning + RL (LM)</td>
<td>Google Search Results</td>
</tr>
<tr>
<td>kNN-prompt (Shi et al., 2022)</td>
<td>Classification Promoting (output)</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>REPLUG (Shi et al., 2023)</td>
<td>Knowledge-intensive Promoting (input)</td>
<td>Wikipedia</td>
</tr>
</tbody>
</table>

**Benefit of retrieval-based prompting:**

- No training & strong performance

**Disadvantage:**

- Hard to control, underperforming full FT model
## Summary of downstream adaptations

<table>
<thead>
<tr>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Izacard et al., 2022)</td>
<td>Knowledge-intensive Fine-tuning (Retriever &amp; LM)</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>GopherCite (Menick et al., 2022)</td>
<td>Open-domain QA, Long-form QA Fine-tuning + RL (LM)</td>
<td>Google Search Results</td>
</tr>
<tr>
<td>kNN-prompt (Shi et al., 2022)</td>
<td>Classification Prompting (output)</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>REPLUG (Shi et al., 2023)</td>
<td>Knowledge-intensive Prompting (input)</td>
<td>Wikipedia</td>
</tr>
</tbody>
</table>

What can be other types of datastores?
Retrieve **code documentations** about related functions

DocPrompting (Zhou et al., 2023)

A lexer splits the source into tokens, fragments …

class PythonLexer
For Python source code

A formatter takes the token stream and writes it to an output file …

class HtmlFormatter
Format tokens as HTML 4 <span> tags with …

NL
Show slurm jobs queued by a user "xyz" every 5 seconds

Code
queue -u xyz -i 5
queue is used to view job and job step for Slurm jobs
-u Request jobs or job steps from a list of users.
DocPrompting (Zhou et al., 2023)

Retriever \[\rightarrow\] LM

A lexer splits the source into tokens, fragments …

```python
class PythonLexer
For Python source code
```

A formatter takes the token stream and writes it to an output file …

```python
class HtmlFormatter
Format tokens as HTML 4 <span> tags with …
```

code = 'print("reading docs")'
```
```
s = highlight(code, PythonLexer(), HtmlFormatter())
```

Fine-tuning (e.g., CodeT5)

Prompting (CodeX)

\[(x, y)\]
DocPrompting (Zhou et al., 2023)

TLDR (NL → bash)

Large gain given by DocPrompting for both CodeT5 & CodeX

BLEU

- CodeT5
- + DocPrompting
- CodeX
- + DocPrompting
**DocPrompting** (Zhou et al., 2023)

Room for improvement in the retrieval component

Active research in OOD / Zero-shot retrieval!
(BEIR; Thakur et al., 2021)
## Summary of downstream adaptations

<table>
<thead>
<tr>
<th>Method</th>
<th>Target task</th>
<th>Adaptation method</th>
<th>Datastore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLAS (Izacard et al., 2022)</td>
<td>Knowledge-intensive</td>
<td>Fine-tuning (Retriever &amp; LM)</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>GopherCite (Menick et al., 2022)</td>
<td>Open-domain QA, Long-form QA</td>
<td>Fine-tuning + RL (LM)</td>
<td>Google Search Results</td>
</tr>
<tr>
<td>kNN-prompt (Shi et al., 2022)</td>
<td>Classification</td>
<td>Prompting (output)</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>REPLUG (Shi et al., 2023)</td>
<td>Knowledge-intensive</td>
<td>Prompting (input)</td>
<td>Wikipedia</td>
</tr>
</tbody>
</table>
How to adapt a retrieval-based LM for a task

- **Retrieval-based prompting** is easy and simple; no need to train but has higher variance.
- **Fine-tuning (+ RL)** requires training but less variance & is competitive with more data.
How to adapt a retrieval-based LM for a task

Training a retriever on downstream tasks helps

Datastore can be diverse (also in Section 6) while challenges remain in OOD retrieval
Two key questions for downstream adaptations

**How** can we adapt a retrieval-based LM for a task?

**When** should we use a retrieval-based LM?
When to use a retrieval-based LM

- Long-tail
- Knowledge update
- Verifiability
- Parameter-efficiency
Key effectiveness in downstream tasks

Long-tail

LLMs often struggle in long-tail/less frequent entities

Kandpal et al. 2023. “Large language models struggle to learn long-tail knowledge”
Key effectiveness in downstream tasks

Scaling LLMs only helps for **popular knowledge**; for long tail, scaling gives marginal performance improvements.

Performance on less popular questions (blue) doesn’t improve over scale.

Key effectiveness in downstream tasks

Retrieval gives large performance gain in such **long-tail**

Retrieval-in-context

Key effectiveness in downstream tasks

Long-tail

Retrieval gives large performance gain in such long-tail

Google RE

Joshua Mathiot died in [MASK].

<— more popular

Less popular —>

Key effectiveness in downstream tasks

Largely reduce hallucinations in **long-form generations**

---

**FactScore**

- **InstructGPT**
- **ChatGPT**
- **Perplexity**

---

Key effectiveness in downstream tasks

**Update**

Standard LLMs need to be *trained again* to adapt to evolving world knowledge.

**Temp LAMA**

<table>
<thead>
<tr>
<th>Year</th>
<th>Prompt</th>
<th>Knowledge Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012</td>
<td>Cristiano Ronaldo plays for <em>X</em>.</td>
<td>Real Madrid</td>
</tr>
<tr>
<td>2019</td>
<td>Cristiano Ronaldo plays for <em>X</em>.</td>
<td>Juventus FC</td>
</tr>
</tbody>
</table>

Huge performance drop when test knowledge needs to be updated.

Izacard et al. 2022. “Few-shot learning with retrieval augmented language models”
Key effectiveness in downstream tasks

Swapping the knowledge corpus to **accommodate temporal changes** without additional training.

**Temp LAMA**

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
<th>Team</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012</td>
<td>Cristiano Ronaldo plays for <em>X</em>.</td>
<td>Real Madrid</td>
</tr>
<tr>
<td>2019</td>
<td>Cristiano Ronaldo plays for <em>X</em>.</td>
<td>Juventus FC</td>
</tr>
</tbody>
</table>

Swapping test datastore only retains strong performance

Izacard et al. 2022. “Few-shot learning with retrieval augmented language models”
Key effectiveness in downstream tasks

**Parameter-efficiency**

**Much smaller LMs with retrieval** can outperform much larger LMs in fact completions.

Retrieval + GPT-Neo 1.3B outperforms vanilla GPT3 on PopQA

Key effectiveness in downstream tasks

Parameter-efficiency

**Much smaller LMs with retrieval** can outperform much larger LMs in fact completions.

NPM (354 M) outperforms GPT-3 on T-Rex.

Min et al. 2023. “Nonparametric Masked Language Modeling”

Key effectiveness in downstream tasks

Parameter-efficiency

**Much smaller LMs with retrieval** can outperform much larger LMs in fact completions.

Room for improvements for diverse task adaptations!

Izacard et al. 2022. “Few-shot learning with retrieval augmented language models”
Key effectiveness in downstream tasks

Verifiability

Human and model can reliably assess the **factuality of the generations** using the retrieved evidence.

Attributions: AttributedQA (Bohnet et al., 2022)

Q: The population of Toronto is _

Answer:

Attribution: The current metro area population of Toronto in 2021 is 6,255,000.

Expected Model Output

\[ g(x) = (a, c) \]

Human Evaluation
(AIS)

1. Are all (a,c) interpretable?
2. Is any information in a supported by c?

Automatic Evaluation
(AutoAIS)

\[ E^A[g] = \frac{1}{n} \sum_{i=1}^{n} \text{AutoAIS}(x_i, g(x_i)) \]
AttributedQA (Bohnet et al., 2022)

Retrieval-based LM

Q: The population of Toronto in 2023 is

Evidence: The current metro area population of Toronto in 2023 is 6,372,000.

LM → 6,372,000

Datastore

Post-hoc retrieval

LM → 6,372,000

Datastore

Attribution: The current metro area population of Toronto in 2021 is 6,255,000.
AttributedQA (Bohnet et al., 2022)

Retrieval in context yields higher AIS than post-hoc retrieval

- Retrieval-based LM
- Post-hoc retrieval

Attributions
When to use a retrieval-based LM

- Long-tail
- Knowledge update
- Verifiability
- Parameter-efficiency
When to use a retrieval-based LM

- Long-tail
- Knowledge update
- Verifiability
- Parameter efficiency

Out of domain adaptations

(Shi et al., 2022; Zheng et al., 2021)


Shi et al. 2022. “Nearest Neighbor Zero-shot Inference”
**When** to use a retrieval-based LM

- Long-tail knowledge update
- Verifiability
- Parameter-efficiency
- Out of domain adaptations
  - (Shi et al., 2022; Zheng et al., 2021)
- Privacy
  - (Huang et al., 2023)


Shi et al. 2022. “Nearest Neighbor Zero-shot Inference”

Huang et al. 2023. “Privacy Implications of Retrieval-Based Language Models”