Language Models as Structured KBs

Bhuwan Dhingra
(Masked) Language Modeling

How much knowledge can you pack into the parameters of a language model? Roberts et al, 2020
Knowledge Graphs

```
SELECT ?x
WHERE {
  "Duke University" founded-by ?y
  ?y place-of-birth ?x
}
```

"Place of birth of the founder of Duke University"

```
SELECT ?x
WHERE {
  ?x instance-of "Education Inst"
  ?x part-of ?property
  ?property object "Duke University"
  ?property start-time ?start
  FILTER { ?start < 1945 }
}
```

"Duke colleges as of 1945"
Can we do this with language models?

```
SELECT ?x
WHERE {
  "Duke University" founded-by ?y
  ?y place-of-birth ?x
}
```

“Place of birth of the founder of Duke University”

```
SELECT ?x
WHERE {
  ?x instance-of "Education Inst"
  ?x part-of ?property
  ?property object "Duke University"
  ?property start-time ?start
  FILTER { ?start < 1945 }
}
```

“Duke colleges as of 1945”
Overview

1. Differentiable query language over text
   - Differentiable Reasoning over a Virtual KB Dhingra et al, ICLR 2020
   - Reasoning Over Virtual KBs With Open Predicate Relations Sun et al, ICML 2021

2. Adding temporal scopes to pretrained knowledge inside LMs
   - Time-Aware LMs as Temporal KBs Dhingra et al, 2021 (Under Review)
Setup: Factoid Question Answering

- Given a query $q$ and a corpus $C$ we need to find an entity answer $a$
- Assume access to an entity linking system over entities $E$
Retrieval augmented models

Retriever can be
- Sparse (BM25) or dense
- Trained end-to-end (REALM) or separately (DPR)

REALM: Retrieval-Augmented Language Model Pre-Training Guu et al, 2020
Dense Passage Retrieval for Open-Domain Question Answering Karpukhin et al, 2020
Retrieval augmented models

Problem
- For complex queries single-shot retrieval does not work

---

```
SELECT ?x
WHERE {
  "Duke University" founded-by ?y
  ?y place-of-birth ?x
}
```

"Place of birth of the founder of Duke University"
DrKIt
(Differentiable Reasoning over a KB of Indexed Text)

**Idea**
- Define a *query language* over an *indexed corpus*
- Parse to expressions in the query language
- Learn the parser end-to-end

**Example Query**
```
SELECT ?x
WHERE {
  "Duke University" founded-by ?y
  ?y place-of-birth ?x
}
```
“Place of birth of the founder of Duke University”

**Diagram**
- **query**
- **Virtual KB**
- **corpus**
- **Parse**
- **Executor**
- **answer**
- **loss**
Virtual KB

An entity-linked corpus

- Family Guy includes an anthropomorphic dog Brian.
- Family Guy was conceived by Seth Macfarlane.
- The voice of Brian is provided by Seth Macfarlane.
Virtual KB

1. Mention embeddings
   - Span start and end vectors from BERT
   - Pretrained on slot-filling over a KB

An entity-linked corpus

Pretraining

Family Guy includes an anthropomorphic dog \textit{Brian}. Family Guy . has character ?
Virtual KB

1. Mention embeddings

2. Sparse co-occurrence matrix
   - All mentions which co-occur with an entity

An entity-linked corpus

**Family Guy** includes an anthropomorphic dog **Brian**.

**Family Guy** was conceived by **Seth MacFarlane**.

The voice of **Brian** is provided by **Seth MacFarlane**.
Virtual KB

1. Mention embeddings

2. Sparse co-occurrence matrix

3. Sparse coreference matrix
   - All mentions of an entity

An entity-linked corpus
DrKIT
(Differentiable Reasoning over a KB of Indexed Text)

**Idea**
- Define a *query language* over an *indexed corpus*
- Parse to expressions in the query language
- Learn the parser end-to-end

**SELECT** ?x
**WHERE** {
  “Duke University” founded-by ?y
  ?y place-of-birth ?x
}

“Place of birth of the founder of Duke University”

**SELECT** ?x
**WHERE** {
  “Duke University” founded-by ?y
  ?y place-of-birth ?x
}

“Place of birth of the founder of Duke University”
Relation following

\[ Y = X.\text{follow}(R) = \{ x' \text{ s.t. } R(x, x') \text{ holds} \} \]

- \( X \) and \( Y \) are sets of entities
- Useful for QA, e.g.,
  “Who is the author of *On the Origin of Species*?”
  \[ X = \{ \text{On the Origin of Species} \}, \ R = \{ \text{author_of} \} \quad \Rightarrow \quad Y = \{ \text{Charles Darwin} \} \]
Relation following

1. Expand $X$ to co-occurring mentions
2. Filter mentions based on similarity to $R$
3. Aggregate over all mentions of the same entity

$Y = X.\text{follow}(R) = \{x' \text{ s.t. } R(x, x') \text{ holds}\}$
Relation following

1. Expand $X$ to co-occurring mentions

$Y = X.\text{follow}(R) = \{x' \text{ s.t. } R(x, x') \text{ holds}\}$

Complexity: $O(|X| \times \text{out-degree})$
Relation following

1. Expand $X$ to co-occurring mentions

2. Filter mentions based on similarity to $R$

$R = \text{“dog in show”}$

$Y = X.\text{follow}(R) = \{x' \text{ s.t. } R(x, x') \text{ holds}\}$

Mentions whose type matches $R$

Complexity: $O(\text{polylog}(\#\text{mentions}))$

$\text{score}(m, R) = f(m)^T q_R$
Relation following

1. Expand $X$ to co-occurring mentions

2. Filter mentions based on similarity to $R$

$X = X.\text{follow}(R) = \{x' \text{ s.t. } R(x, x') \text{ holds}\}$

$A_{E\to M \cup X}$

Complexity: $O(\text{polylog}(#\text{mentions}))$
Relation following

1. Expand $X$ to co-occurring mentions

2. Filter mentions based on similarity to $R$

3. Aggregate over all mentions of the same entity

$Y = X.\text{follow}(R) = \{x' \text{ s.t. } R(x, x') \text{ holds}\}$

$\mathcal{U}X.\text{follow}(R)$

$B_{M \rightarrow E}$

$\mathcal{T}_K(q_R) \odot A_{E \rightarrow M} \mathcal{U}X$

Sparse vector representing the weighted set of entities $Y$

Complexity: $O(K)$
Relation following

1. Expand $X$ to co-occurring mentions
2. Filter mentions based on similarity to $R$
3. Aggregate over all mentions of the same entity

$Y = X.\text{follow}(R) = \{x' \text{ s.t. } R(x, x') \text{ holds}\}$

\[
u_X.\text{follow}(R) = B_{M \to E} [T_K(q_R) \odot A_{E \to M} \nu_X]
\]

- Efficient
- Closed under composition
- Differentiable
DrKIT
(Differentiable Reasoning over a KB of Indexed Text)

SELECT ?x
WHERE {
  "Duke University" founded-by ?y
  ?y place-of-birth ?x
}

"Place of birth of the founder of Duke University"

Idea
- Define a query language over an indexed corpus
- Parse to expressions in the query language
- Learn the parser end-to-end

Differentiable Reasoning over a Virtual Knowledge Base Dhingra et al, 2020
Query Templates

Who voices the dog in the TV show Family Guy?

Entity Linker

Transformer-1

Transformer-2

X.follow(R1).follow(R2)

execute
Who voices the dog in the TV show Family Guy?

Mixing Templates

Entity Linker

Transformer-1

Transformer-2

X.follow(R1).follow(R2)

X.follow(R1)

X.follow(R1) & X.follow(R2)

execute

execute

execute

combine
## Results: Multi-Hop QA

<table>
<thead>
<tr>
<th>Model</th>
<th>MetaQA 2Hop</th>
<th>MetaQA 3Hop</th>
<th>MSF 2Hop</th>
<th>MSF 3Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>KVMem</td>
<td>7.0</td>
<td>19.5</td>
<td>3.4</td>
<td>2.6</td>
</tr>
<tr>
<td>DrQA</td>
<td>32.5</td>
<td>19.7</td>
<td>14.1</td>
<td>7.0</td>
</tr>
<tr>
<td>GRAFT-Net</td>
<td>36.2</td>
<td>40.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PullNet</td>
<td>81.0</td>
<td>78.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PIQA</td>
<td>-</td>
<td>-</td>
<td>36.9</td>
<td>18.2</td>
</tr>
<tr>
<td>DrKIT</td>
<td>86.0</td>
<td><strong>87.6</strong></td>
<td>46.9</td>
<td>24.4</td>
</tr>
</tbody>
</table>

**Hits @1**

- 18K passages
- 43K entities
- 7 relations
- 120K passages
- 200K entities
- 888 relations
Results: Multi-Hop QA

<table>
<thead>
<tr>
<th>Model</th>
<th>MetaQA 2Hop</th>
<th>MetaQA 3Hop</th>
<th>MSF 2Hop</th>
<th>MSF 3Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>KVMem</td>
<td>7.0</td>
<td>19.5</td>
<td>3.4</td>
<td>2.6</td>
</tr>
<tr>
<td>DrQA</td>
<td>32.5</td>
<td>19.7</td>
<td>14.1</td>
<td>7.0</td>
</tr>
<tr>
<td>GRAFT-Net</td>
<td>36.2</td>
<td>40.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PullNet</td>
<td>81.0</td>
<td>78.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PIQA</td>
<td>-</td>
<td>-</td>
<td>36.9</td>
<td>18.2</td>
</tr>
<tr>
<td>DrKIT</td>
<td>86.0</td>
<td>87.6</td>
<td>46.9</td>
<td>24.4</td>
</tr>
</tbody>
</table>

Retrieved once and then find answer

Hits @1
## Results: Multi-Hop QA

<table>
<thead>
<tr>
<th>Model</th>
<th>MetaQA 2Hop</th>
<th>MetaQA 3Hop</th>
<th>MSF 2Hop</th>
<th>MSF 3Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>KVMem</td>
<td>7.0</td>
<td>19.5</td>
<td>3.4</td>
<td>2.6</td>
</tr>
<tr>
<td>DrQA</td>
<td>32.5</td>
<td>19.7</td>
<td>14.1</td>
<td>7.0</td>
</tr>
<tr>
<td>GRAFT-Net</td>
<td>36.2</td>
<td>40.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PullNet</td>
<td>81.0</td>
<td>78.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PIQA</td>
<td>-</td>
<td>-</td>
<td>36.9</td>
<td>18.2</td>
</tr>
<tr>
<td>DrKIT</td>
<td>86.0</td>
<td><strong>87.6</strong></td>
<td>46.9</td>
<td>24.4</td>
</tr>
</tbody>
</table>

Retrieve iteratively but not end-to-end

**Hits @1**
## Results: Multi-Hop QA

<table>
<thead>
<tr>
<th>Model</th>
<th>MetaQA 2Hop</th>
<th>MetaQA 3Hop</th>
<th>MSF 2Hop</th>
<th>MSF 3Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>KVMem</td>
<td>7.0</td>
<td>19.5</td>
<td>3.4</td>
<td>2.6</td>
</tr>
<tr>
<td>DrQA</td>
<td>32.5</td>
<td>19.7</td>
<td>14.1</td>
<td>7.0</td>
</tr>
<tr>
<td>GRAFT-Net</td>
<td>36.2</td>
<td>40.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PullNet</td>
<td>81.0</td>
<td>78.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PIQA</td>
<td>-</td>
<td>-</td>
<td>36.9</td>
<td>18.2</td>
</tr>
<tr>
<td><strong>DrKIT</strong></td>
<td><strong>86.0</strong></td>
<td><strong>87.6</strong></td>
<td><strong>46.9</strong></td>
<td><strong>24.4</strong></td>
</tr>
</tbody>
</table>

*3x-15x faster than baselines*

*Hits @1*
Limitations

- Pretraining the mention embeddings requires an existing KB
  - Not available in every domain
  - Lower accuracy on relations not in the KB

- Same mention participates in different relations
OPQL: Open Predicate Query Language

Virtual KB is a *key-value memory* over *pairs of entity mentions*

*Charles Darwin* published his book *Origin of the Species* after waiting ....
OPQL: Open Predicate Query Language

Charles Darwin published his book *Origin of the Species* after waiting ....

We can pretrain this without a KB by matching entity pair mentions!

Relation embedding

Query for top-K search

Entity embedding (from previous hop)

\[ q_{X,Y} = W_q^T [e_X; W_t^T r_{X,Y}] \]

Matching the Blanks: Distributional Similarity for Relation Learning Soares et al, 2019
Results: Multi-Hop QA

<table>
<thead>
<tr>
<th>Model</th>
<th>MetaQA 2Hop</th>
<th>MetaQA 3Hop</th>
<th>MSF 2Hop</th>
<th>MSF 3Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>KVMem</td>
<td>7.0</td>
<td>19.5</td>
<td>3.4</td>
<td>2.6</td>
</tr>
<tr>
<td>DrQA</td>
<td>32.5</td>
<td>19.7</td>
<td>14.1</td>
<td>7.0</td>
</tr>
<tr>
<td>GRAFT-Net</td>
<td>36.2</td>
<td>40.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PullNet</td>
<td>81.0</td>
<td>78.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PIQA</td>
<td>-</td>
<td>-</td>
<td>36.9</td>
<td>18.2</td>
</tr>
<tr>
<td>DrKIT</td>
<td>86.0</td>
<td><strong>87.6</strong></td>
<td>46.9</td>
<td>24.4</td>
</tr>
<tr>
<td>OPQL-pretrained</td>
<td><strong>84.7</strong></td>
<td><strong>84.3</strong></td>
<td>48.5</td>
<td>28.1</td>
</tr>
<tr>
<td><strong>OPQL</strong></td>
<td><strong>88.5</strong></td>
<td>87.1</td>
<td><strong>49.2</strong></td>
<td><strong>29.7</strong></td>
</tr>
</tbody>
</table>

Relation encoder is tuned on domain specific relations.
More results: Open-domain QA

<table>
<thead>
<tr>
<th>Model</th>
<th>WebQSP</th>
<th>ComplexWebQ (dev)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRAFT-NET</td>
<td>25.3</td>
<td>10.6</td>
</tr>
<tr>
<td>PullNet</td>
<td>24.8</td>
<td>13.1</td>
</tr>
<tr>
<td>BART-Large</td>
<td>30.4</td>
<td>-</td>
</tr>
<tr>
<td>EaE</td>
<td>47.4</td>
<td>31.3</td>
</tr>
<tr>
<td>DPR</td>
<td>48.6</td>
<td>24.6</td>
</tr>
<tr>
<td>DPR-cascade</td>
<td>-</td>
<td>25.1</td>
</tr>
<tr>
<td>T5</td>
<td>49.7</td>
<td>38.7</td>
</tr>
<tr>
<td>OPQL-follow</td>
<td>46.6</td>
<td>18.5</td>
</tr>
<tr>
<td><strong>OPQL-LM</strong></td>
<td><strong>51.9</strong></td>
<td><strong>40.7</strong></td>
</tr>
</tbody>
</table>

Mixing relation following results with a language model
Summary

● How can we perform explicit reasoning in a neural network?
  ○ By defining differentiable query languages over preprocessed corpora
  ○ Much related work over structured KBs (NQL, EmQL, Query2Box)

● How can we make the query language more expressive?
  ○ Conjunctions and disjunctions are possible but not well tested
  ○ Numerical operations are more tricky
Overview

1. Differentiable query language over text
   - Differentiable Reasoning over a Virtual KB Dhingra et al, ICLR 2020
   - Reasoning Over Virtual KBs With Open Predicate Relations Sun et al, ICML 2021

2. Adding temporal scopes to pretrained knowledge inside LMs
   - Time-Aware LMs as Temporal KBs Dhingra et al, 2021 (Under Review)
Knowledge changes with time

- Do LMs learn the temporal scope of the facts they encode?
- How can we update temporally-scoped knowledge in trained models?
Training Data Timeline


(2017) Lebron James plays for Cleveland Cavaliers.


(2024) Lebron James plays for ???.
TempLAMA: A diagnostic dataset

- Identify Wikidata facts with multiple objects across time

- Convert to masked LM queries
  \[ t = 2012 \]
  \[ x = \text{“Lebron James plays for ____”} \]
  \[ y = \text{“Miami Heat”} \]
Time-aware pretraining

- Instead of $Pr(y|x)$ model $Pr(y|x, t)$

- Pretraining data: CustomNews (Lazaridou et al, 2020)
  - 1M news articles each from 2010-2018
  - Mask out salient spans (entities)

- Start with T5 pretrained for 1M steps on C4 (April 2019 crawl)
T5 closed-book QA model struggles on time-sensitive questions

All models are trained on data prior to 2019

<table>
<thead>
<tr>
<th>Model</th>
<th>#Parameters</th>
<th>TempLAMA 2010-18</th>
<th>TempLAMA 2019-20</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5-CBQA</td>
<td>737M</td>
<td>5.4</td>
<td>4.3</td>
<td>5.2</td>
</tr>
<tr>
<td>T5-CBQA-ft</td>
<td>737M</td>
<td>17.8</td>
<td>15.3</td>
<td>17.3</td>
</tr>
<tr>
<td>Uniform</td>
<td>737M</td>
<td>28.1</td>
<td>19.8</td>
<td>26.6</td>
</tr>
<tr>
<td>Yearly</td>
<td>6.6B</td>
<td>28.5</td>
<td>21.8</td>
<td>27.3</td>
</tr>
<tr>
<td>Temporal</td>
<td>737M</td>
<td><strong>29.6</strong></td>
<td><strong>22.2</strong></td>
<td><strong>28.2</strong></td>
</tr>
</tbody>
</table>

Token-level F1
### Results

Pretraining on uniformly sampled news helps (~4% inputs mention a date)

All models are trained on data prior to 2019

<table>
<thead>
<tr>
<th>Model</th>
<th>#Parameters</th>
<th>TempLAMA 2010-18</th>
<th>TempLAMA 2019-20</th>
<th>TempLAMA Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5-CBQA</td>
<td>737M</td>
<td>5.4</td>
<td>4.3</td>
<td>5.2</td>
</tr>
<tr>
<td>T5-CBQA-ft</td>
<td>737M</td>
<td>17.8</td>
<td>15.3</td>
<td>17.3</td>
</tr>
<tr>
<td>Uniform</td>
<td>737M</td>
<td>28.1</td>
<td>19.8</td>
<td>26.6</td>
</tr>
<tr>
<td>Yearly</td>
<td>6.6B</td>
<td>28.5</td>
<td>21.8</td>
<td>27.3</td>
</tr>
<tr>
<td>Temporal</td>
<td>737M</td>
<td><strong>29.6</strong></td>
<td><strong>22.2</strong></td>
<td><strong>28.2</strong></td>
</tr>
</tbody>
</table>
## Results

### Token-level F1

<table>
<thead>
<tr>
<th>Model</th>
<th>#Parameters</th>
<th>TempLAMA 2010-18</th>
<th>TempLAMA 2019-20</th>
<th>TempLAMA Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5-CBQA</td>
<td>737M</td>
<td>5.4</td>
<td>4.3</td>
<td>5.2</td>
</tr>
<tr>
<td>T5-CBQA-ft</td>
<td>737M</td>
<td>17.8</td>
<td>15.3</td>
<td>17.3</td>
</tr>
<tr>
<td>Uniform</td>
<td>737M</td>
<td>28.1</td>
<td>19.8</td>
<td>26.6</td>
</tr>
<tr>
<td>Yearly</td>
<td>6.6B</td>
<td>28.5</td>
<td>21.8</td>
<td>27.3</td>
</tr>
<tr>
<td>Temporal</td>
<td>737M</td>
<td><strong>29.6</strong></td>
<td><strong>22.2</strong></td>
<td><strong>28.2</strong></td>
</tr>
</tbody>
</table>

Single model with time prefixes does better than ensemble.

All models are trained on data prior to 2019.
# Results

## Token-level F1

<table>
<thead>
<tr>
<th>Model</th>
<th>#Parameters</th>
<th>TempLAMA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2010-18</td>
</tr>
<tr>
<td>T5-CBQA</td>
<td>737M</td>
<td>5.4</td>
</tr>
<tr>
<td>T5-CBQA-ft</td>
<td>737M</td>
<td>17.8</td>
</tr>
<tr>
<td>Uniform</td>
<td>737M</td>
<td>28.1</td>
</tr>
<tr>
<td>Yearly</td>
<td>6.6B</td>
<td>28.5</td>
</tr>
<tr>
<td>Temporal</td>
<td>737M</td>
<td><strong>29.6</strong></td>
</tr>
</tbody>
</table>

On future slice models only get unchanged facts correct

All models are trained on data prior to 2019
Updates without forgetting

- Setup: finetune models trained on 2010-28 on new data from 2019
- To avoid forgetting mix old and new data and train for 50K steps
- Temporal prefixes lead to less forgetting

Fraction of new data when finetuning
Summary

- **Time-aware pretraining helps**
  - Organize temporal knowledge inside an LM
  - Add new knowledge to the LM

- **What is the best way of modeling time in LMs?**
  - String prefixes are easy but don’t have any inductive bias about the continuity of time
  - Lots of related work on temporal knowledge graphs (e.g. HyTE; Dasgupta et al, 2018)

- **What other metadata can be useful?**
Thank you.