Megan Leszczynski
Lecture 15: Integrating Knowledge in Language Models
Lecture Plan

1. Recap: language models (LMs)
2. What does a LM know?
3. Techniques to add knowledge to LMs
   1. Add pretrained entity embeddings
   2. Use an external memory
   3. Modify the training data
4. Evaluating knowledge in LMs

Reminders:
• Project milestone due today!
• Change of grading basis/course withdrawal deadline is this Friday at 5PM PT!
• Final projects due Tuesday, March 16th at 4:30PM PT!
Recap: LMs

• **Standard language models** predict the *next word* in a sequence of text and can compute the probability of a sequence

  \[ \text{The students opened their } \underline{\text{books}}. \]

• Recently, **masked language models** (e.g., BERT) instead predict a *masked token* in a sequence of text using bidirectional context

  \[ \text{went } \underline{\text{to the store}}. \]

• Both types of language models can be trained over large amounts of unlabeled text!
Recap: LMs

- Traditionally, LMs are used for many tasks involving generating or evaluating the probability of text:
  - Summarization
  - Dialogue
  - Autocompletion
  - Machine translation
  - Fluency evaluation
  - ...

- Today, LMs are commonly used to generate pretrained representations of text that encode some notion of language understanding for downstream NLP tasks
- Can a language model be used as a knowledge base?
What does a language model know?

• iPod Touch is produced by ________.

• London Jazz Festival is located in ________.

• Dani Alves plays with ________.

• Carl III used to communicate in _________.

• Ravens can ________.

Examples taken from Petroni et al., EMNLP 2019 to test BERT-Large.
What does a language model know?

- iPod Touch is produced by __________.
- London Jazz Festival is located in __________.
- Dani Alves plays with __________.
- Carl III used to communicate in __________.
- Ravens can __________.

Examples taken from Petroni et al., EMNLP 2019 to test BERT-Large.
What does a language model know?

• Takeaway: predictions generally make sense (e.g. the correct types), but are not all factually correct.

• Why might this happen?
  • Unseen facts: some facts may not have occurred in the training corpora at all
  • Rare facts: LM hasn’t seen enough examples during training to memorize the fact
  • Model sensitivity: LM may have seen the fact during training, but is sensitive to the phrasing of the prompt
    • Correctly answers “x was made in y” templates but not “x was created in y”

• The inability to reliably recall knowledge is a key challenge facing LMs today!
  • Recent works have found LMs can recover some knowledge, but have a way to go.
The importance of knowledge-aware language models

- LM pretrained representations can benefit downstream tasks that leverage knowledge
  - For instance, extracting the relations between two entities in a sentence is easier with some knowledge of the entities
  - We’ll come back to this when talking about evaluation!

- Stretch goal: can LMs ultimately replace traditional knowledge bases?
  - Instead of querying a knowledge base for a fact (e.g. with SQL), query the LM with a natural language prompt!
    - Of course, this requires LM to have high quality on recalling facts
Querying traditional knowledge bases

- Query knowledge base with SQL statements

```
SELECT date_of_birth
WHERE person = "Franklin D. Roosevelt"
```
Querying language models as knowledge bases

- Pretrain LM over unstructured text and then query with natural language.

President Franklin <M> born <M> January 1882.
Lily couldn't <M>. The waitress had brought the largest <M> of chocolate cake <M> seen.
Our <M> hand-picked and sun-dried <M> orchard in Georgia.
D. Roosevelt was <M> in
believe her eyes <M> piece <M> she had ever
peaches are <M> at our

President Franklin D. Roosevelt was born in January 1882.

When was Franklin D. Roosevelt born?

T5
1882

Pre-training
Fine-tuning

Roberts et al., EMNLP 2020
Advantages of language models over traditional KBs

- LMs are pretrained over large amounts of unstructured and unlabeled text
  - KBs require manual annotation or complex NLP pipelines to populate

- LMs support more flexible natural language queries
  - Example: *What does the final F in the song U.F.O.F. stand for?*
    - Traditional KB wouldn’t have a field for “final F”; LM *may* learn this

- However, there are also many open challenges to using LMs as KBs:
  - Hard to interpret (i.e., why does the LM produce an answer)
  - Hard to trust (i.e., the LM may produce a realistic, incorrect answer)
  - Hard to modify (i.e., not easy to remove or update knowledge in the LM)

*Petroni et al., EMNLP 2019 & Roberts et al., EMNLP 2020*
Section 2: Techniques to add knowledge to LMs
Techniques to add knowledge to LMs

Add pretrained entity embeddings
- ERNIE
- KnowBERT

Use an external memory
- KGLM
- kNN-LM

Modify the training data
- WKLM
- ERNIE (another!), salient span masking
Techniques to add knowledge to LMs

Add pretrained entity embeddings
- ERNIE
- KnowBERT

Use an external memory
- KGLM
- kNN-LM

Modify the training data
- WKLM
- ERNIE (another!), salient span masking
Method 1: Add pretrained embeddings

- Facts about the world are usually in terms of entities
  - Example: Washington was the first president of the United States.

- Pretrained word embeddings do not have a notion of entities
  - Different word embeddings for “U.S.A.”, “United States of America” and “America” even though these refer to the same entity

- What if we assign an embedding per entity?
  - Single entity embedding for “U.S.A.”, “United States of America” and “America”

- Entity embeddings can be useful to LMs iff you can do entity linking well!
Aside: What is entity linking?

• Link mentions in text to entities in a knowledge base

Washington was the first president of the United States.

• Entity linking tells us which entity embeddings are relevant to the text

More resources: Orr et al., CIDR 2021 & Li et al., EMNLP 2020
Method 1: Add pretrained entity embeddings

Entity embeddings are like word embeddings, but for entities in a knowledge base!

\[
George\ Washington = \begin{pmatrix}
0.111 \\
-0.345 \\
0.876 \\
-0.201
\end{pmatrix}
\]

Many techniques for training entity embeddings:
- Knowledge graph embedding methods (e.g., TransE)
- Word-entity co-occurrence methods (e.g., Wikipedia2Vec)
- Transformer encodings of entity descriptions (e.g., BLINK)
**Method 1: Add pretrained entity embeddings**

**Question:** How do we incorporate pretrained entity embeddings from a different embedding space?

**Answer:** Learn a fusion layer to combine context and entity information.

\[ h_j = F(W_tw_j + W_e e_k + b) \]

We assume there’s a known alignment between entities and words in the sentence such that \( e_k = f(w_j) \)

- \( w_j \) is the embedding of word \( j \) in a sequence of words
- \( e_k \) is the corresponding entity embedding
ERNIE: Enhanced Language Representation with Informative Entities [Zhang et al., ACL 2019]

- **Text encoder**: multi-layer bidirectional Transformer encoder over the words in the sentence
- **Knowledge encoder**: stacked blocks composed of:
  - Two multi-headed attentions (MHAs) over entity embeddings and token embeddings
  - A fusion layer to combine the output of the MHAs

\[
\begin{align*}
    h_j &= \sigma \left( \tilde{W}_t^{(i)} \tilde{w}_j^{(i)} + \tilde{W}_e^{(i)} \tilde{e}_k^{(i)} + \tilde{b}^{(i)} \right) \\
    w_j^{(i)} &= \sigma \left( W_t^{(i)} h_j + b_t^{(i)} \right) \\
    e_k^{(i)} &= \sigma \left( W_e^{(i)} h_j + b_e^{(i)} \right)
\end{align*}
\]
**ERNIE: Enhanced Language Representation with Informative Entities** [Zhang et al., ACL 2019]

Bob Dylan wrote *Blowin' in the Wind* in 1962

(a) Model Architecture

(b) Aggregator
ERNIE: Enhanced Language Representation with Informative Entities [Zhang et al., ACL 2019]

- Pretrain with three tasks:
  - Masked language model and next sentence prediction (i.e., BERT tasks)
  - Knowledge pretraining task (dEA\(^1\)): randomly mask token-entity alignments and predict corresponding entity for a token from the entities in the sequence

\[
p(e_j | w_i) = \frac{\exp(Ww_i \cdot e_j)}{\sum_{k=1}^{m} \exp(Ww_i \cdot e_k)}
\]

\[
\mathcal{L}_{ERNIE} = \mathcal{L}_{MLM} + \mathcal{L}_{NSP} + \mathcal{L}_{dEA}
\]

ERNie: Enhanced Language Representation with Informative Entities [Zhang et al., ACL 2019]

Additional knowledge pretraining task is necessary to make the most use of the pretrained entity embeddings.
ERNE: Enhanced Language Representation with Informative Entities [Zhang et al., ACL 2019]

• Strengths:
  • Combines entity + context info through fusion layers and a knowledge pretraining task
  • Improves performance downstream on knowledge-driven tasks

• Remaining challenges:
  • Needs text data with entities annotated as input, even for downstream tasks
    • For instance, “Bob Dylan wrote Blowin’ in the Wind” needs entities pre-linked to input entities into ERNIE
  • Requires further (expensive) pretraining of the LM

[1] Check out Poerner et al., EMNLP 2020 for a method to avoid more LM pretraining.
Jointly learn to link entities with KnowBERT \cite{Peters et al., EMNLP 2019}

- Key idea: pretrain an integrated entity linker (EL) as an extension to BERT

\[ \mathcal{L}_{\text{KnowBERT}} = \mathcal{L}_{\text{NSP}} + \mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{EL}} \]

- On downstream tasks, EL predicts entities so entity annotations aren’t required

- Learning EL may better encode knowledge - shows performance gains over ERNIE on downstream tasks

- Like ERNIE, KnowBERT uses a fusion layer to combine entity and context information and adds a knowledge pretraining task
Techniques to add knowledge to LMs

Add pretrained entity embeddings
- ERNIE
- KnowBERT

Use an external memory
- KGLM
- kNN-LM

Modify the training data
- WKLM
- ERNIE (another!), salient span masking
Method 2: Use an external memory

• Previous methods rely on the pretrained entity embeddings to encode the factual knowledge from KBs for the language model.

• **Question**: Are there more direct ways than pretrained entity embeddings to provide the model factual knowledge?

• **Answer**: Yes! Give the model access to an external memory (a key-value store with access to KG triples or context information)

• **Advantages**:
  • Can better support injecting and updating factual knowledge
    • Often without more pretraining!
  • More interpretable
Barack's Wife Hillary: Using Knowledge-Graphs for Fact-Aware Language Modeling (KGLM) [Logan et al., ACL 2019]

- Key idea: condition the language model on a knowledge graph (KG)

- Recall that language models predict the next word by computing

\[ P(x^{(t+1)}|x^{(t)}, \ldots, x^{(1)}) \text{, where } x^{(1)}, \ldots, x^{(t)} \text{ is a sequence of words} \]

- Now, predict the next word using entity information, by computing

\[ P(x^{(t+1)}, E^{(t+1)}|x^{(t)}, \ldots, x^{(1)}, E^{(t)}, \ldots, E^{(1)}) \]

where \( E^{(t)} \) is the set of KG entities mentioned at timestep \( t \)
KGLM [Logan et al., ACL 2019]

• Build a “local” knowledge graph as you iterate over the sequence
  • Local KG: subset of the full KG with only entities relevant to the sequence

**Super Mario Land** is a game developed by **Nintendo**.

Assumes entities are known during training!

- **When** should the LM use the local KG to predict the next word?
Super Mario Land is a game developed by **Nintendo**.

Classify: Is the next word...
1. **Related entity** (in the local KG)
2. **New entity** (not in the local KG)
3. **Not an entity**

- Use the LSTM hidden state to predict the type of the next word (3 classes)
- **How** does the LM predict the next entity and word in each case?
Super Mario Land is a game developed by Nintendo.

Related entity (in the local KG)

Example
Top scoring parent entity: “Super Mario Land”
Top scoring relation: “publisher”
-> Next entity is “Nintendo”, due to KG triple (Super Mario Land, publisher, Nintendo).
Super Mario Land is a game developed by Nintendo.

Related entity (in the local KG)
- Find the top-scoring parent and relation in the local KG using the LSTM hidden state and pretrained entity and relation embeddings
  - \( P(p_t) = \text{softmax}(v_p \cdot h_t) \), where \( p_t \) is the “parent” entity, \( v_p \) is the corresponding entity embedding, and \( h_t \) is from the LSTM hidden state
- **Next entity:** tail entity from KG triple of (top parent entity, top relation, tail entity)
- **Next word:** most likely next token over vocabulary expanded to include entity aliases\(^1\)

---

\(^1\) Phrases that could refer to Nintendo (e.g. Nintendo, Nintendo Co., Koppai)
New entity (not in the local KG)

- Find the top-scoring entity in the full KG using the LSTM hidden state and pretrained entity embeddings
- **Next entity**: directly predict top-scoring entity
- **Next word**: most likely next token over vocabulary expanded to include entity aliases

Not an entity

- **Next entity**: None
- **Next word**: most likely next token over standard vocabulary
Super Mario Land is a 1989 side-scrolling platform video game developed and published by **Nintendo**
KGLM [Logan et al., ACL 2019]

• Outperforms GPT-2 and AWD-LSTM\(^1\) on a fact completion task

• Qualitatively, compared to GPT-2, KGLM tends to predict more specific tokens (GPT-2 predicts more popular, generic tokens)

• Supports modifying/updating facts!
  • Modifying the KG has a direct change in the predictions

Barack Obama was born on _________.

**KG triples:**

(Barack Obama, *birthDate*, 1961-08-04)  
(Barack Obama, *birthDate*, 2013-03-21)

**Most likely next word:**

“August”, “4”, “1961”  
“March”, “21”, “2013”

[1] Merity et al., ICLR 2018
More recent takes: Nearest Neighbor Language Models (kNN-LM) [Khandelwal et al., ICLR 2020]

• Key idea: learning similarities between text sequences is easier than predicting the next word
  • Example: “Dickens is the author of _______” ≈ “Dickens wrote_______”
  • Qualitatively, researchers find this is especially true for “long-tail patterns”, such as rare facts
• So, store all representations of text sequences in a nearest neighbor datastore!
• At inference:
  1. Find the $k$ most similar sequences of text in the datastore
  2. Retrieve the corresponding values (i.e. the next word) for the $k$ sequences
  3. Combine the kNN probabilities and LM probabilities for the final prediction

$$P(y|x) = \lambda P_{kNN}(y|x) + (1 - \lambda)P_{LM}(y|x)$$

[1] $\lambda$ is a tuned hyperparameter
More recent takes: **Nearest Neighbor Language Models (kNN-LM)**

[Khandelwal et al., ICLR 2020]

Example: Shakespeare’s play _____ ....

Task: Predict the next word with kNN-LM
Techniques to add knowledge to LMs

Add pretrained entity embeddings
- ERNIE
- KnowBERT

Use an external memory
- KGLM
- kNN-LM

Modify the training data
- WKLM
- ERNIE (another!), salient span masking
Method 3: Modify the training data

- Previous methods incorporated knowledge explicitly through pretrained embeddings and/or an external memory.

- **Question**: Can knowledge also be incorporated implicitly through the unstructured text?

- **Answer**: Yes! Mask or corrupt the data to introduce additional training tasks that require factual knowledge.

- **Advantages**:
  - No additional memory/computation requirements
  - No modification of the architecture required
Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model (WKLM) [Xiong et al., ICLR 2020]

• Key idea: train the model to distinguish between true and false knowledge

• Replace mentions in the text with mentions that refer to different entities of the same type to create negative knowledge statements
  • Model predicts if entity as been replaced or not
  • Type-constraint is intended to enforce linguistically correct sentences

True knowledge statement: J.K. Rowling is the author of Harry Potter.

Negative knowledge statement: J.R.R. Tolkien is the author of Harry Potter.
Original Article:
Spider-Man is a fictional superhero created by writer-editor Stan Lee and writer-artist Steve Ditko. He first appeared in the anthology comic book American comic books published by Marvel Comics.

Replaced Article:
Spider-Man is a fictional superhero created by writer-editor Bryan Johnson and writer-artist Steve Ditko. He first appeared in the anthology comic book American comic books published by DC Comics.
**WKLM** [Xiong et al., ICLR 2020]

- Uses an entity replacement loss to train the model to distinguish between true and false mentions

\[
\mathcal{L}_{entRep} = \mathbb{1}_{e \in \mathcal{E}^+} \log P(e \mid C) + (1 - \mathbb{1}_{e \in \mathcal{E}^+}) \log (1 - P(e \mid C))
\]

where $e$ is an entity, $C$ is the context, and $\mathcal{E}^+$ represents a true entity mention

- Total loss is the combination of standard masked language model loss (MLM) and the entity replacement loss.

\[
\mathcal{L}_{WKLM} = \mathcal{L}_{MLM} + \mathcal{L}_{entRep}
\]

- MLM is defined at the **token-level**; entRep is defined at the **entity-level**
**WKLM** [Xiong et al., ICLR 2020]

- Improves over BERT and GPT-2 in fact completion tasks
- Improves over ERNIE on a downstream task (entity typing)
- Ablation experiments
  - MLM loss is essential for downstream task performance
  - WKLM outperforms training longer with just MLM loss

<table>
<thead>
<tr>
<th>Model</th>
<th>SQuAD (F1)</th>
<th>TriviaQA (F1)</th>
<th>Quasar-T (F1)</th>
<th>FIGER (acc)</th>
</tr>
</thead>
<tbody>
<tr>
<td>WKLM</td>
<td>91.3</td>
<td>56.7</td>
<td>49.9</td>
<td>60.21</td>
</tr>
<tr>
<td>WKLM w/o MLM</td>
<td>87.6</td>
<td>52.5</td>
<td>48.1</td>
<td>58.44</td>
</tr>
<tr>
<td>BERT + 1M Updates</td>
<td>91.1</td>
<td>56.3</td>
<td>48.2</td>
<td>54.17</td>
</tr>
</tbody>
</table>

Much worse without MLM

Much worse training for longer, compared to using the entity replacement loss
Learn inductive biases through masking

• Can we encourage the LM to learn factual knowledge by being clever about masking?
• Thread in several recent works:
  • **ERNIE**: Enhanced Representation through Knowledge Integration, Sun et al., arXiv 2019
    • Shows improvements on downstream Chinese NLP tasks with phrase-level and entity-level masking
  • How Much Knowledge Can You Pack Into the Parameters of a Language Model?, Roberts et al., EMNLP 2020
    • Uses “salient span masking” (Guu et al., ICML 2020) to mask out salient spans (i.e. named entities and dates)
    • Shows that salient span masking helps T5 performance on QA

[1] Yes, another ERNIE paper!
ERNIE\textsuperscript{1}: Enhanced Representation through Knowledge Integration

\cite{Sun2019}

\[1\] Yes, another ERNIE paper!
Salient span masking

Salient span masking has been shown to outperform other masking/corruption strategies on retrieval and QA tasks.

REALM on Natural Questions

<table>
<thead>
<tr>
<th>Masking technique</th>
<th>Exact Match</th>
<th>Retrieval Recall @5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Salient span masking</td>
<td>38.2</td>
<td>38.5</td>
</tr>
<tr>
<td>Random uniform masks</td>
<td>32.3</td>
<td>24.2</td>
</tr>
<tr>
<td>Random span masks</td>
<td>35.3</td>
<td>26.1</td>
</tr>
</tbody>
</table>

Guu et al., ICML 2020

Roberts et al., EMNLP 2020
Recap: Techniques to add knowledge to LMs

1. Use pretrained entity embeddings
   - Often not too difficult to apply to existing architectures to leverage KG pretraining
   - Indirect way of incorporating knowledge and can be hard to interpret

2. Add an external memory
   - Can support some updating of factual knowledge and easier to interpret
   - Tend to be more complex in implementation and require more memory

3. Modify the training data
   - Requires no model changes or additional computation. May also be easiest to theoretically analyze! Active area of research
   - Still open question if this is always as effective as model changes
Section 3: Evaluating knowledge in LMs
LAngeue Model Analysis (LAMA) Probe [Petroni et al., EMNLP 2019]

• How much relational (commonsense and factual) knowledge is already in off-the-shelf language models?
  • Without any additional training or fine-tuning

• Manually constructed a set of cloze statements to assess a model’s ability to predict a missing token.
  Examples:

  The theory of relativity was developed by [MASK].
  The native language of Mammootty is [MASK].
  Ravens can [MASK].
  You are likely to find a overflow in a [MASK].
LLanguage Model Analysis (LAMA) Probe [Petroni et al., EMNLP 2019]

- Generate cloze statements from KG triples and question-answer pairs
- Compare LMs to supervised relation extraction (RE) and question answering systems
- **Goal:** evaluate knowledge present in existing pretrained LMs (this means they may have different pretraining corpora!)

**Mean precision at one (P@1)**

<table>
<thead>
<tr>
<th>Corpus</th>
<th>DrQA</th>
<th>RE baseline</th>
<th>fairseq-fconv</th>
<th>Transformer-XL</th>
<th>ELMo</th>
<th>ELMo (5.5B)</th>
<th>BERT-base</th>
<th>BERT-large</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google-RE</td>
<td>-</td>
<td>7.6</td>
<td>2.6</td>
<td>1.6</td>
<td>2.0</td>
<td>3.0</td>
<td>9.8</td>
<td>10.5</td>
</tr>
<tr>
<td>T-REx</td>
<td>-</td>
<td>33.8</td>
<td>8.9</td>
<td>18.3</td>
<td>4.7</td>
<td>7.1</td>
<td>31.1</td>
<td>32.2</td>
</tr>
<tr>
<td>ConceptNet</td>
<td>-</td>
<td>-</td>
<td>3.6</td>
<td>5.7</td>
<td>6.1</td>
<td>6.2</td>
<td>15.6</td>
<td>19.2</td>
</tr>
<tr>
<td>SQuAD</td>
<td>37.5</td>
<td>-</td>
<td>3.6</td>
<td>3.9</td>
<td>1.6</td>
<td>4.3</td>
<td>14.1</td>
<td>17.4</td>
</tr>
</tbody>
</table>

BERT struggles on N-to-M relations

LMs are NOT finetuned!
LLanguage Model Analysis (LAMA) Probe [Petroni et al., EMNLP 2019]

You can try out examples to assess knowledge in popular LMs: https://github.com/facebookresearch/LAMA

The cat is on the [MASK].

[1] Example courtesy of the authors at link above.
LLanguage Model Analysis (LAMA) Probe [Petroni et al., EMNLP 2019]

- Limitations of the LAMA probe:
  - Hard to understand *why* models perform well when they do
    - BERT-large may be memorizing co-occurrence patterns rather than “understanding” the cloze statement
    - LM could just be identifying similarities between the surface forms of the subject and object (e.g., Pope Clement VII has the position of pope)
  - LMs are sensitive to the phrasing of the statement
    - LAMA has only one manually defined template for each relation
    - This means probe results are a *lower bound* on knowledge encoded in the LM
A More Challenging Probe: LAMA-UnHelpful Names (LAMA-UHN)  
[Poerner et al., EMNLP 2020]

- Key idea: Remove the examples from LAMA that can be answered without relational knowledge

- Observation: BERT may rely on surface forms of entities to make predictions
  - String match between subject and object
  - “Revealing” person name
    - Name can be a (possibly incorrect) prior for native language, place of birth, nationality, etc.

- BERT’s score on LAMA drops ~8% with LAMA-UHN
  - Knowledge-enhanced model E-BERT score drops only <1%

<table>
<thead>
<tr>
<th>Person Name</th>
<th>BERT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jean Marais</td>
<td>French</td>
</tr>
<tr>
<td>Daniel Ceccaldi</td>
<td>Italian</td>
</tr>
<tr>
<td>Orane Demazis</td>
<td>Albanian</td>
</tr>
<tr>
<td>Sylvia Lopez</td>
<td>Spanish</td>
</tr>
<tr>
<td>Annick Alane</td>
<td>English</td>
</tr>
</tbody>
</table>
Developing better prompts to query knowledge in LMs

[Jiang et al., TACL 2020]

• LMs may know the fact, but fail on completion tasks like LAMA due to the query itself
  • Pretraining may be on different contexts and sentence structures than the query
    Example: “The birth place of Barack Obama is Honolulu, Hawaii” (pretraining corpus) versus “Barack Obama was born in _____” (query)

• Generate more LAMA prompts by mining templates from Wikipedia\(^1\) and generating paraphrased prompts by using back-translation

• Ensemble prompts to increase diversity of contexts that fact can be seen in

[1] One mining approach uses dependency parsing to build the template!
Developing better prompts to query knowledge in LMs

[Jiang et al., TACL 2020]

• Performance on LAMA for BERT-large increases 7% when using top-performing query for each relation. Ensembling leads to another 4% gain.

• Small changes in the query lead to large gains.
  • LMs are extremely sensitive to the query!

<table>
<thead>
<tr>
<th>ID</th>
<th>Modifications</th>
<th>Acc. Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>P413</td>
<td>$x$ plays in $\rightarrow$ at $y$ position</td>
<td>+23.2</td>
</tr>
<tr>
<td>P495</td>
<td>$x$ was created $\rightarrow$ made in $y$</td>
<td>+10.8</td>
</tr>
<tr>
<td>P495</td>
<td>$x$ was $\rightarrow$ is created in $y$</td>
<td>+10.0</td>
</tr>
<tr>
<td>P361</td>
<td>$x$ is a part of $y$</td>
<td>+2.7</td>
</tr>
<tr>
<td>P413</td>
<td>$x$ plays $in$ $y$ position</td>
<td>+2.2</td>
</tr>
</tbody>
</table>
Knowledge-driven downstream tasks

- Measures how well the knowledge-enhanced LM transfers its knowledge to downstream tasks
- Unlike probes, this evaluation usually requires finetuning the LM on downstream tasks, like evaluating BERT on GLUE tasks

- Common tasks:
  - Relation extraction
    - Example: [Bill Gates] was born in [Seattle]; label: city of birth
  - Entity typing
    - Example: [Alice] robbed the bank; label: criminal
  - Question answering
    - Example: “What kind of forest is the Amazon?”; label: “moist broadleaf forest”
Relation extraction performance on TACRED

• Knowledge-enhanced systems (ERNIE, Matching the Blanks, KnowBERT) improve over previously state-of-the-art models for relation extraction

<table>
<thead>
<tr>
<th>Model</th>
<th>LM</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-GCN</td>
<td>-</td>
<td>69.9</td>
<td>63.3</td>
<td>66.4</td>
</tr>
<tr>
<td>BERT-LSTM-base</td>
<td>BERT-Base</td>
<td>73.3</td>
<td>63.1</td>
<td>67.8</td>
</tr>
<tr>
<td>ERNIE (Zhang et al.)</td>
<td>BERT-Base</td>
<td>70.0</td>
<td>66.1</td>
<td>68.0</td>
</tr>
<tr>
<td>Matching the Blanks (MTB)</td>
<td>BERT-Large</td>
<td>_</td>
<td>_</td>
<td>71.5</td>
</tr>
<tr>
<td>KnowBert-W+W</td>
<td>BERT-Base</td>
<td>71.6</td>
<td>71.4</td>
<td>71.5</td>
</tr>
</tbody>
</table>

Peters et al., EMNLP 2019
Entity typing performance on Open Entity

- Knowledge-enhanced LMs (ERNIE, KnowBERT) improve over prior LSTM and BERT-Base models on entity typing
- Impressively, NFGEC and UFET were designed for entity typing

<table>
<thead>
<tr>
<th>Model</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFGEC (LSTM)</td>
<td>68.8</td>
<td>53.3</td>
<td>60.1</td>
</tr>
<tr>
<td>UFET (LSTM)</td>
<td>77.4</td>
<td>60.6</td>
<td>68.0</td>
</tr>
<tr>
<td>BERT-Base</td>
<td>76.4</td>
<td>71.0</td>
<td>73.6</td>
</tr>
<tr>
<td>ERNIE (Zhang et al.)</td>
<td>78.4</td>
<td>72.9</td>
<td>75.6</td>
</tr>
<tr>
<td>KnowBert-W+W</td>
<td>78.6</td>
<td>73.7</td>
<td>76.1</td>
</tr>
</tbody>
</table>

Zhang et al., ACL 2019 & Peters et al., EMNLP 2019
Recap: Evaluating knowledge in LMs

• Probes
  • Evaluate the knowledge already present in models without more training
  • Challenging to construct benchmarks that require factual knowledge
  • Challenge to construct the queries used in the probe

• Downstream tasks
  • Evaluate the usefulness of the knowledge-enhanced representation in applications
  • Often requires finetuning the LM further on the downstream task
  • Less direct way to evaluate the knowledge in the LM
Other exciting progress & what’s next?

• Retrieval-augmented language models
  • REALM, Guu et al., ICML 2020

• Modifying knowledge in language models
  • Modifying Memories in Transformer Models, Zhu et al., arXiv 2020

• More multitask pre-training for language models
  • KEPLER, Wang et al., TACL 2020

• More efficient knowledge systems
  • NeurIPS Efficient QA challenge

• Better knowledge benchmarks
  • KILT, Petroni et al., arXiv 2020
Good luck with your projects!