Natural Language Processing with Deep Learning
CS224N/Ling284

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

The students opened their **books** *.*

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

went ***store***

I [MASK] to the [MASK].

- 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 __Apple__.
- London Jazz Festival is located in __London__.
- Dani Alves plays with __Santos__.
- Carl III used to communicate in __German__.
- Ravens can __fly__.

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.

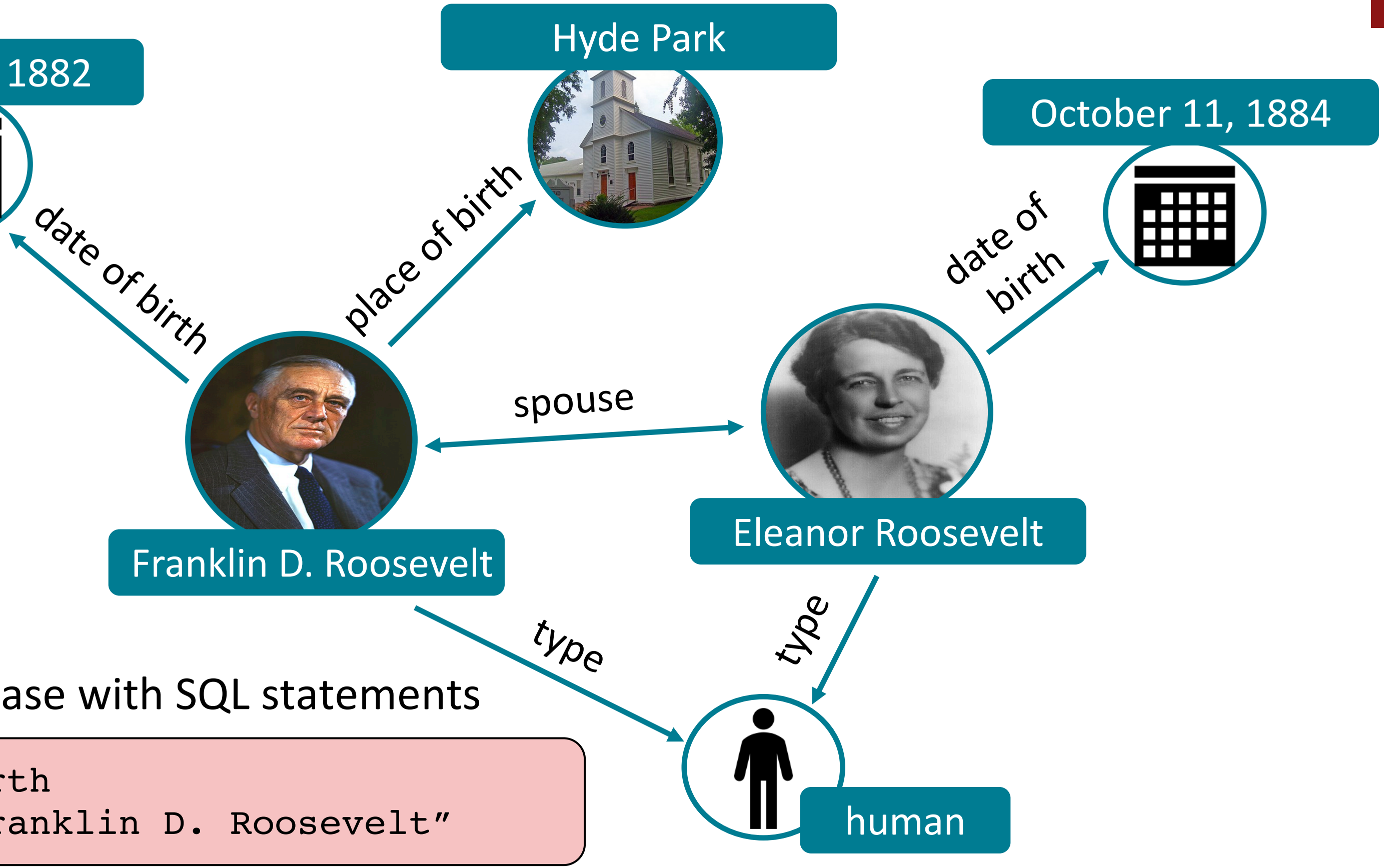
The importance of knowledge-aware language models

- 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

WIKIDATA

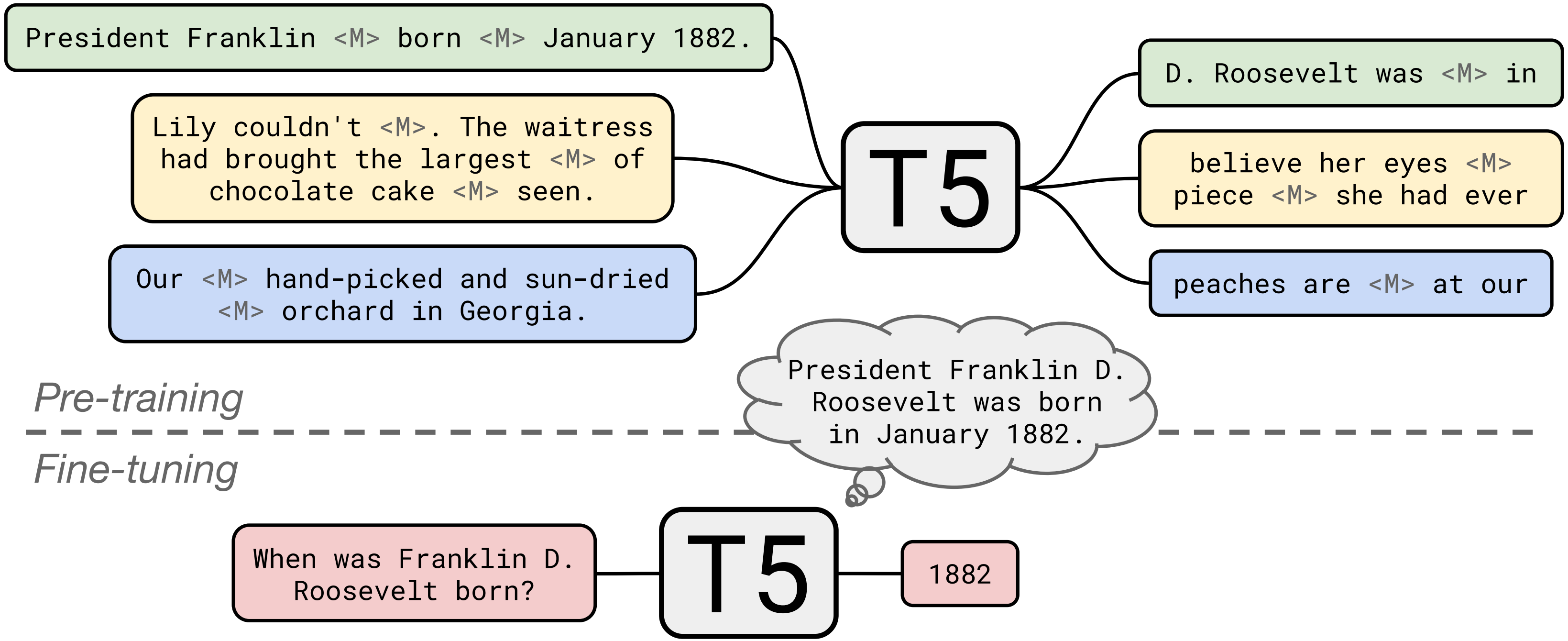


- Query knowledge base with SQL statements

```
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.



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

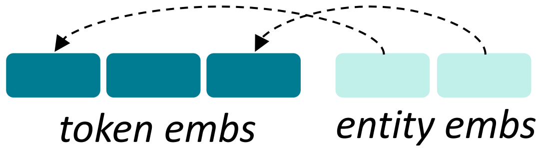
- LMs support more flexible natural language queries
  - Example: *What does the final F in the song U.F.O.F. stand for?*
    - Traditional KB wouldn't have a field for "final F"; LM *may* learn this

- 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

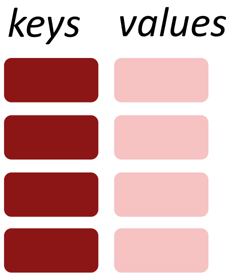
Section 2:
Techniques to add knowledge to LMs

Techniques to add knowledge to LMs



Add pretrained entity embeddings

- ERNIE
- KnowBERT



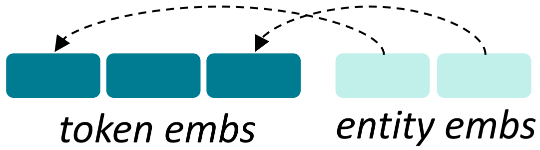
Use an external memory

- KGLM
- kNN-LM

Modify the training data

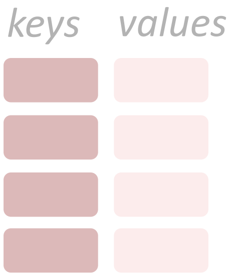
- WKLM
- ERNIE (another!), salient span masking

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 ^entity 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!

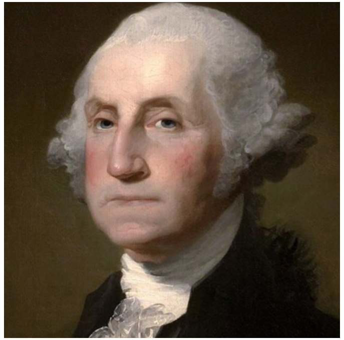
Aside: What is entity linking?

- Link mentions in text to entities in a knowledge base

mention *mention*

Washington was the first president of the United States.

candidate

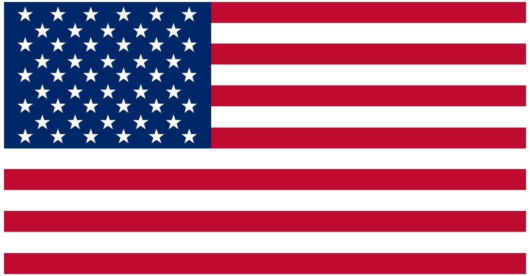


Q23 (Wikidata)

candidate

Q1223 (Wikidata)

candidate



Q30 (Wikidata)

- 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!

$$\textit{George Washington} = \begin{pmatrix} 0.111 \\ -0.345 \\ 0.876 \\ -0.201 \end{pmatrix}$$

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.

$$\boldsymbol{h}_j = F\left(\boldsymbol{W}_t \boldsymbol{w}_j + \boldsymbol{W}_e \boldsymbol{e}_k + b\right)$$

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

- $\boldsymbol{w}_j$ is the embedding of word j in a sequence of words
- $\boldsymbol{e}_k$ is the corresponding entity embedding

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

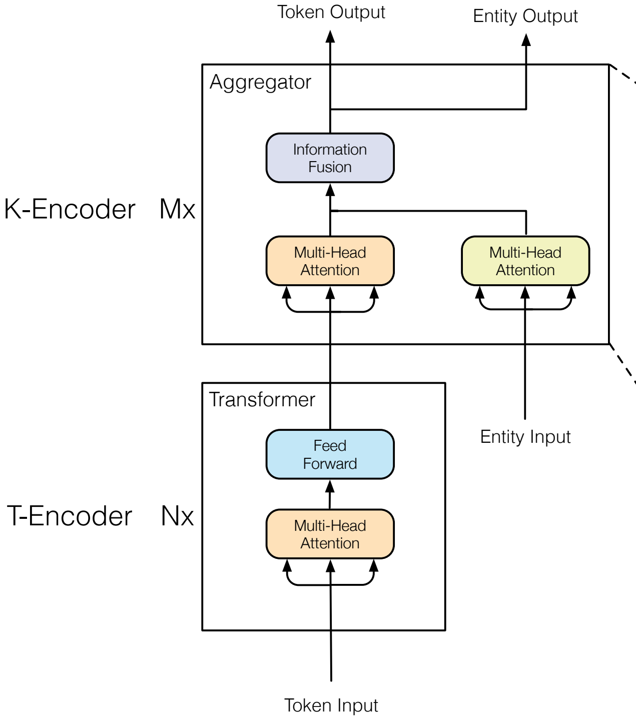
- Text encoder: multi-layer bidirectional Transformer encoder over the words in the sentence
- Knowledge encoder: stacked blocks composed of:
  - Two multi-headed attentions (MHAs) over entity embeddings and token embeddings
  - A fusion layer to combine the output of the MHAs

$$\boldsymbol{h}_j = \sigma\left(\widetilde{\boldsymbol{W}}_t^{(i)}\widetilde{\boldsymbol{w}}_j^{(i)} + \widetilde{\boldsymbol{W}}_e^{(i)}\widetilde{\boldsymbol{e}}_k^{(i)} + \widetilde{\boldsymbol{b}}^{(i)}\right)$$

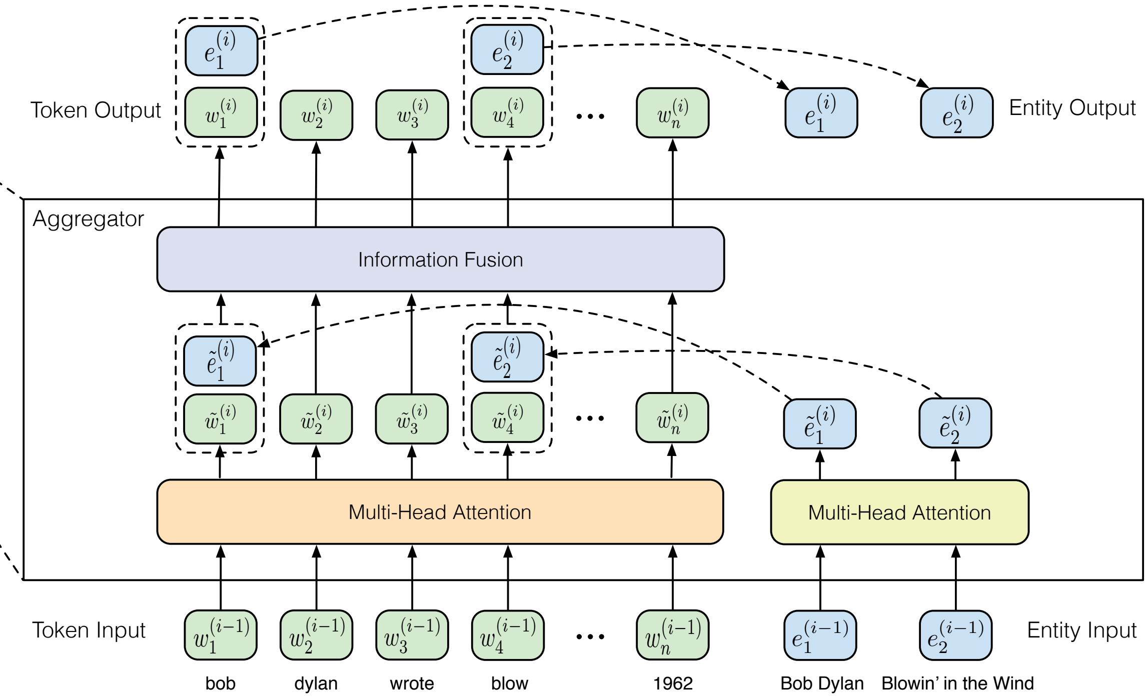
$$\boldsymbol{w}_j^{(i)} = \sigma\left(\boldsymbol{W}_t^{(i)}\boldsymbol{h}_j + \boldsymbol{b}_t^{(i)}\right)$$

$$\boldsymbol{e}_k^{(i)} = \sigma\left(\boldsymbol{W}_e^{(i)}\boldsymbol{h}_j + \boldsymbol{b}_e^{(i)}\right)$$

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



(a) Model Achitecture



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

(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

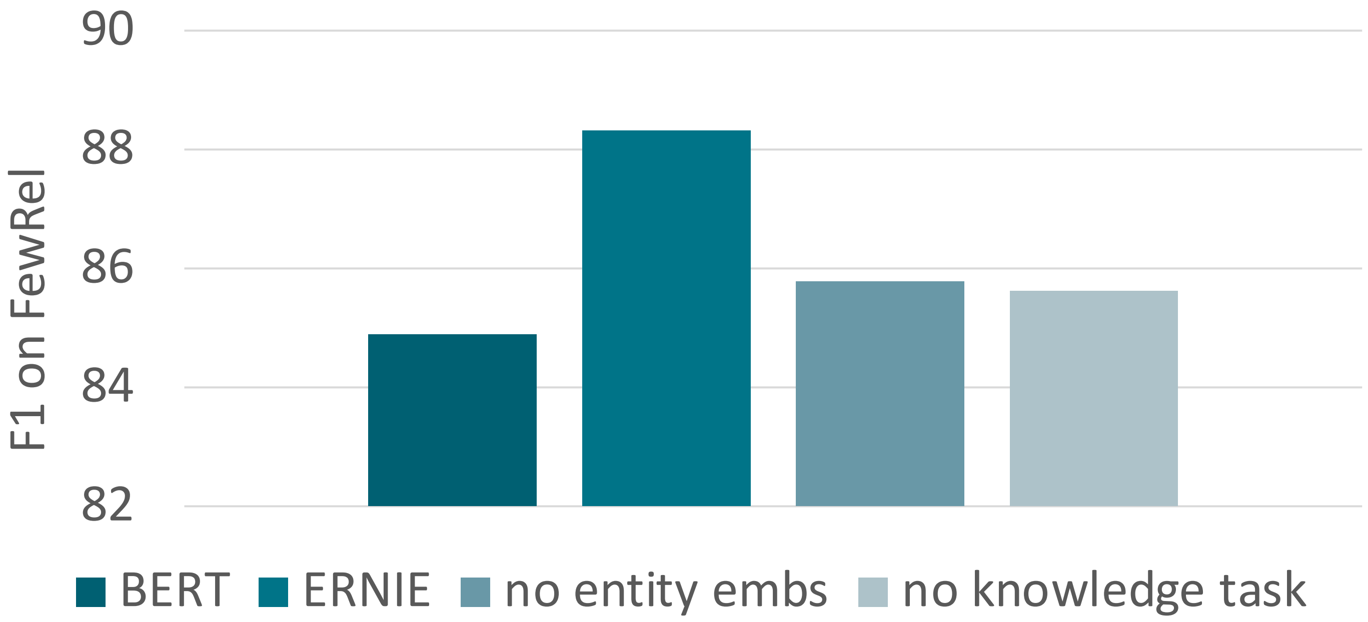
$$p(e_j \mid w_i) = \frac{\exp(\boldsymbol{W}\boldsymbol{w}_i \cdot \boldsymbol{e}_j)}{\sum_{k=1}^{m} \exp(\boldsymbol{W}\boldsymbol{w}_i \cdot \boldsymbol{e}_k)}$$

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

[1] dEA named for denoising entity autoencoder from Vincent et al., ICML 2008.

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.



ERNIE: 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]

[1] Check out Poerner et al., EMNLP 2020 for a method to avoid more LM pretraining.

Jointly learn to link entities with KnowBERT [Peters et al., EMNLP 2019]

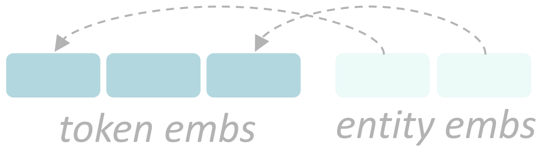
- Key idea: pretrain an integrated entity linker (EL) as an extension to BERT

$$\mathcal{L}_{KnowBERT} = \mathcal{L}_{NSP} + \mathcal{L}_{MLM} + \mathcal{L}_{EL}$$

Predict over set of hard candidates (not just those in sentence)

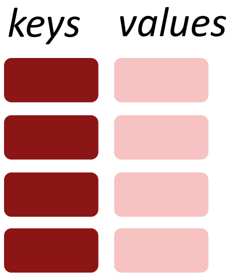
- On downstream tasks, EL predicts entities so entity annotations aren't required
- Learning EL may better encode knowledge - shows performance gains over ERNIE on downstream tasks
- Like ERNIE, KnowBERT uses a fusion layer to combine entity and context information and adds a knowledge pretraining task

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

$$\mathrm{P}\left(x^{(t+1)} \middle| x^{(t)}, \ldots, x^{(1)}\right)$$, where $x^{(1)}, \ldots, x^{(t)}$ is a sequence of words

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

$$\mathrm{P}\left(x^{(t+1)}, \mathcal{E}^{(t+1)} \middle| x^{(t)}, \ldots, x^{(1)}, \mathcal{E}^{(t)}, \ldots, \mathcal{E}^{(1)}\right)$$

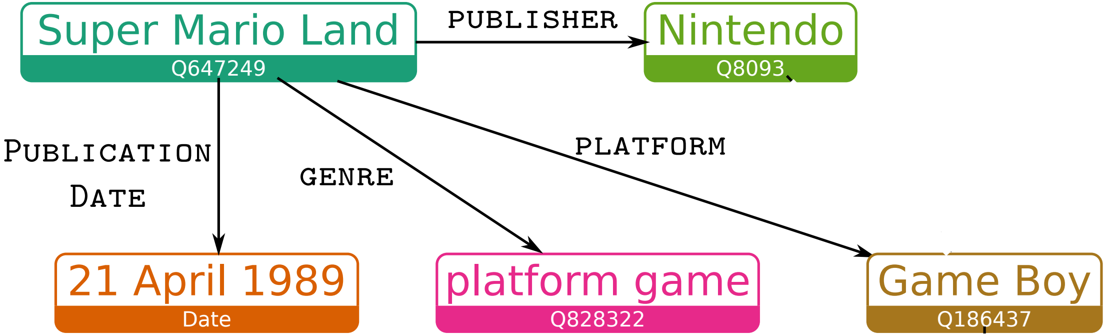
where $\mathcal{E}^{(t)}$ is the set of KG entities mentioned at timestep t

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

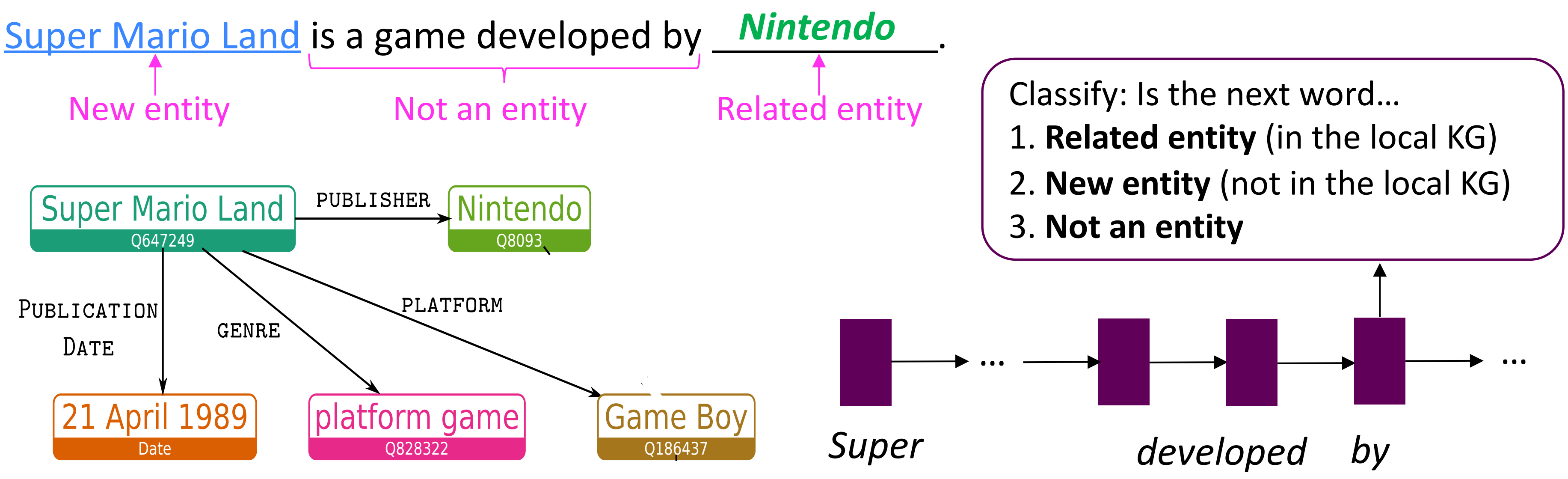
Super Mario Land is a game developed by ___*Nintendo*___.

Assumes entities are known during training!



- **When** should the LM use the local KG to predict the next word?

KGLM [Logan et al., ACL 2019]



- 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?

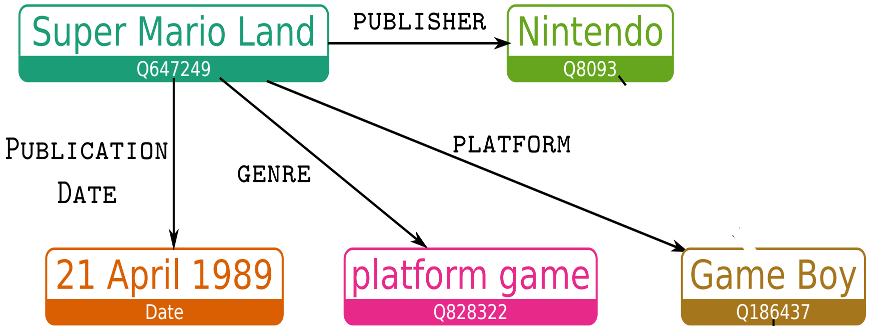
KGLM [Logan et al., ACL 2019]

Super Mario Land is a game developed by Nintendo.

New entity | Not an entity | Related entity

Related entity (in the local KG)

KG triple = (parent entity, relation, tail entity)



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).

KGLM [Logan et al., ACL 2019]

$\underbrace{\text{Super Mario Land}}_{\text{New entity}}$ $\underbrace{\text{is a game developed by}}_{\text{Not an entity}}$ $\underbrace{\text{Nintendo}}_{\text{Related entity}}$.

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
  - $P(p_t) = \text{softmax}(\boldsymbol{v_p} \cdot \boldsymbol{h_t})$, where p_t is the "parent" entity, $\boldsymbol{v_p}$ is the corresponding entity embedding, and $\boldsymbol{h_t}$ 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)

KGLM [Logan et al., ACL 2019]

Super Mario Land is a game developed by Nintendo.

New entity — Not an entity — Related entity

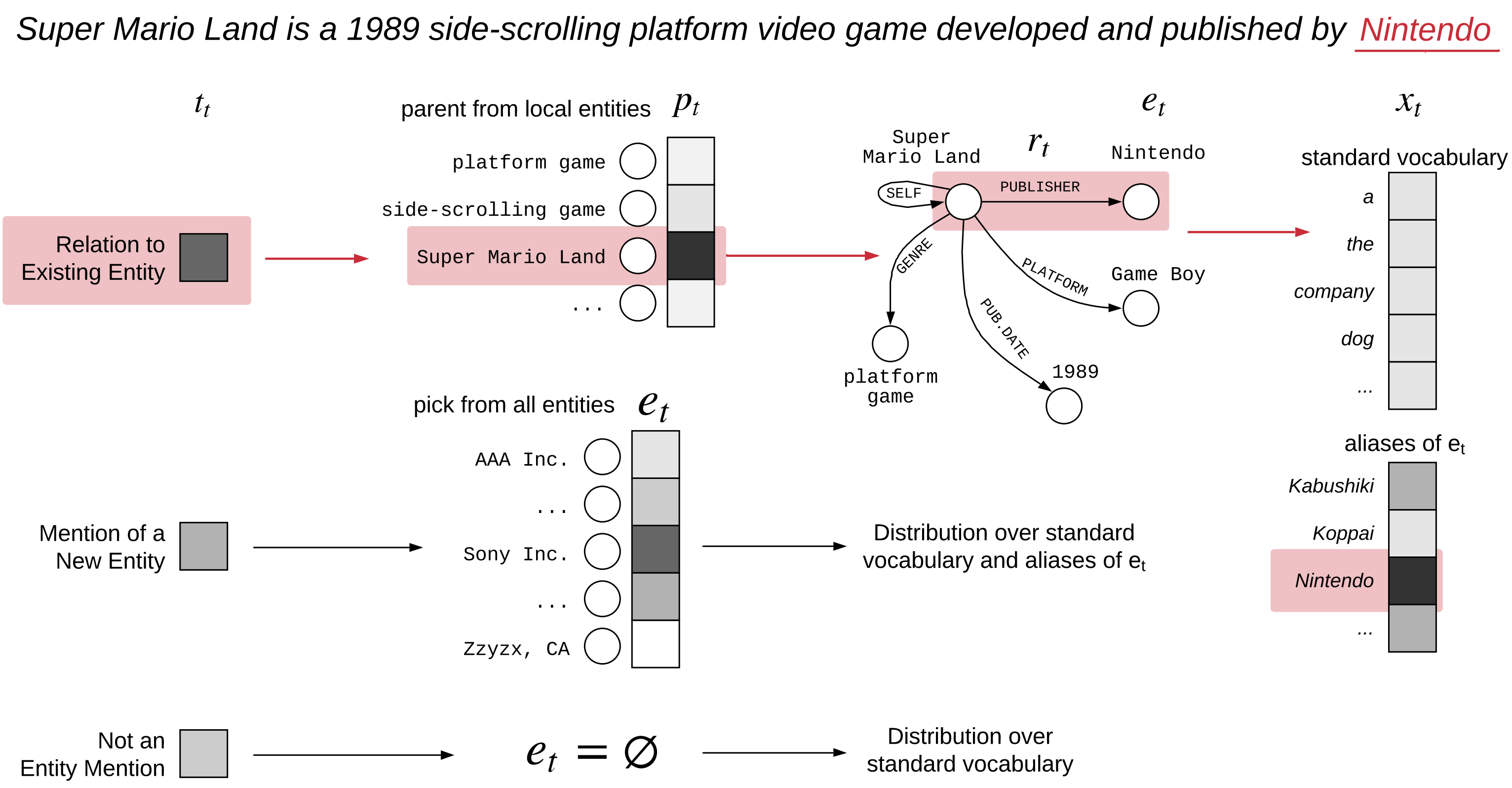
New entity (not in the local KG)

- 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

KGLM [Logan et al., ACL 2019]



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:	**Most likely next word:**
(Barack Obama, *birthDate*, 1961-08-04)	"August", "4", "1961"
(Barack Obama, *birthDate*, 2013-03-21)	"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

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

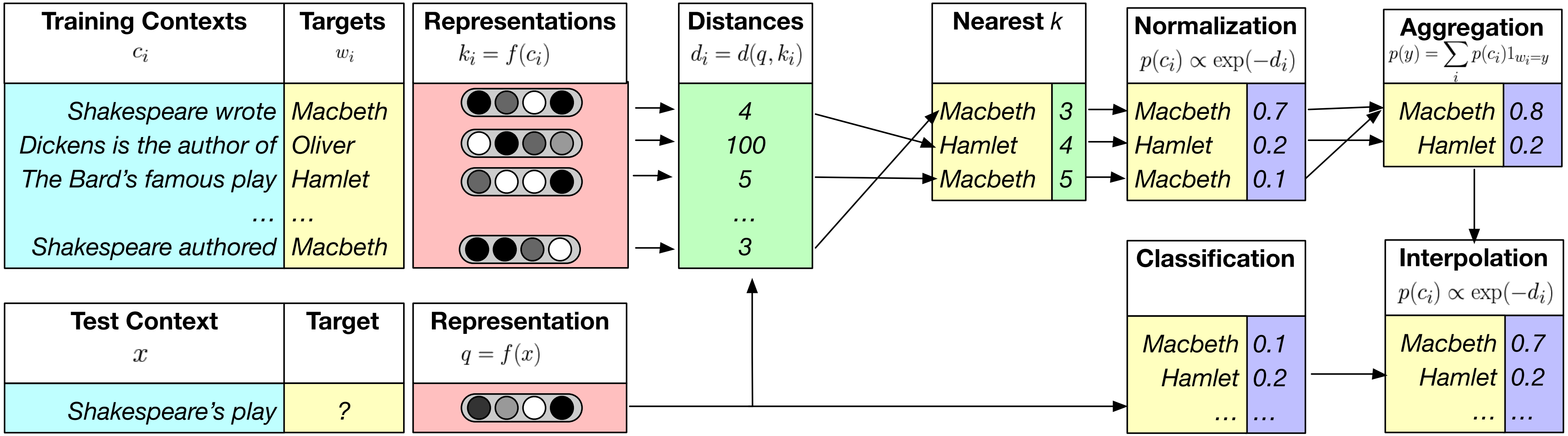
[1] λ is a tuned hyperparameter

More recent takes: **Nearest Neighbor Language Models (kNN-LM)**

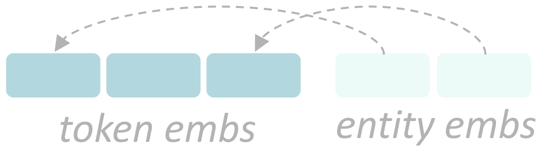
[Khandelwal et al., ICLR 2020]

Example: Shakespeare's play ______ ....

Task: Predict the next word with kNN-LM

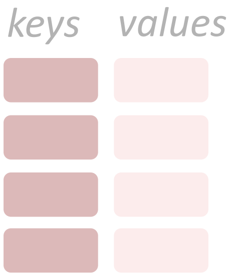


Techniques to add knowledge to LMs



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

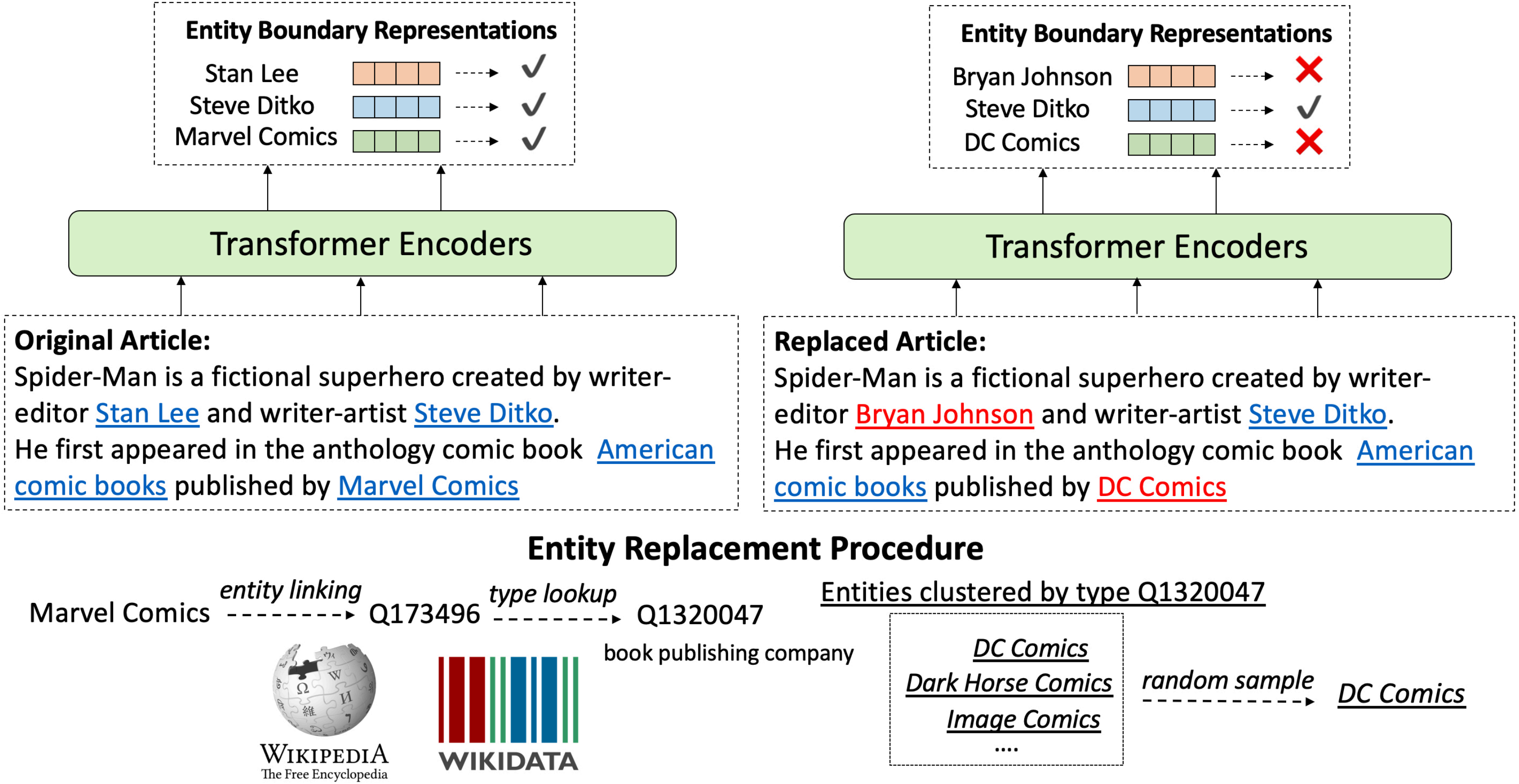
Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model (WKLM) [Xiong et al., ICLR 2020]

- 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.

WKLM [Xiong et al., ICLR 2020]



WKLM [Xiong et al., ICLR 2020]

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

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

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

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

Much worse without MLM

Much worse training for longer, compared to using the entity replacement loss

Learn inductive biases through masking

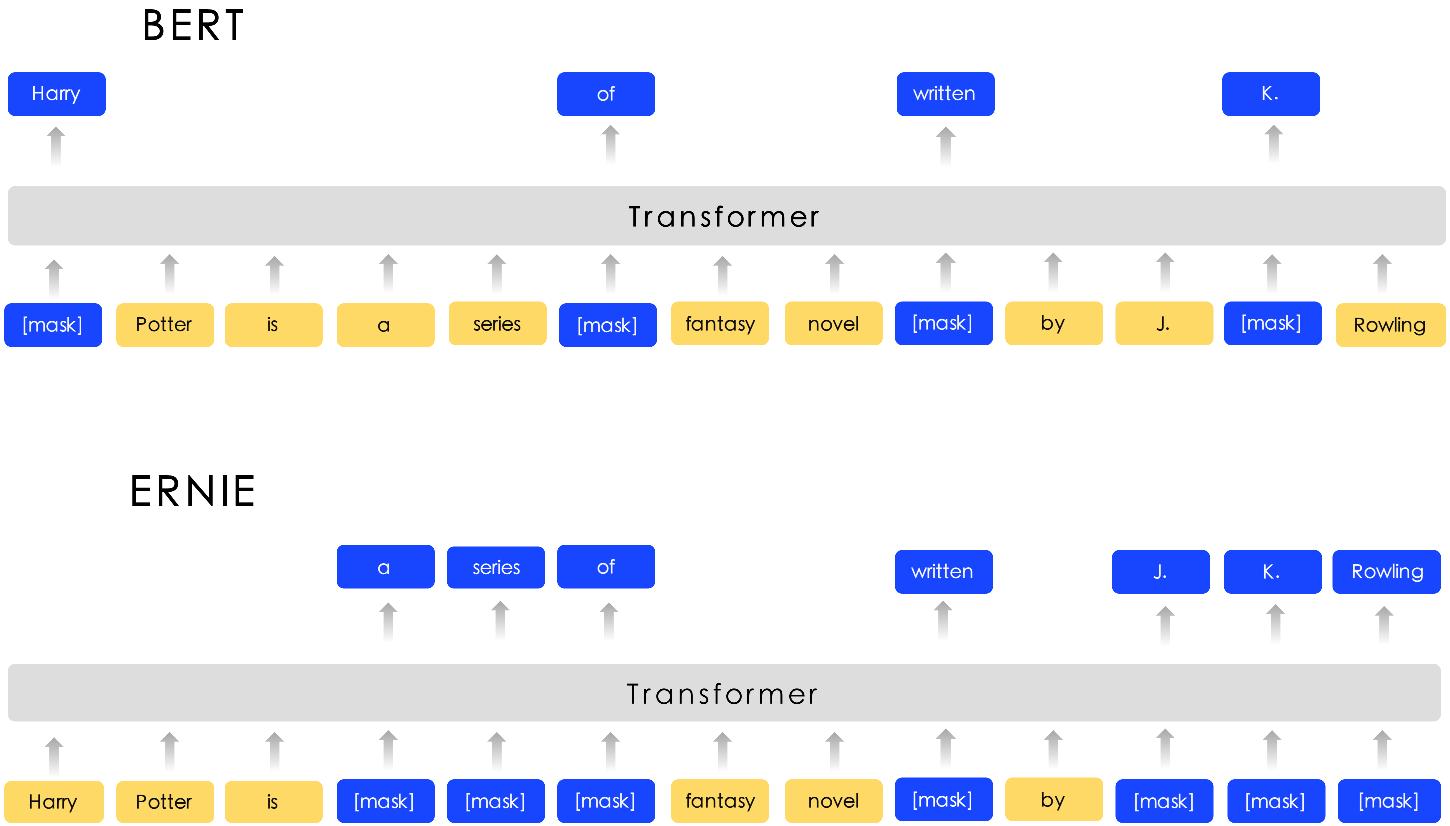
- Can we encourage the LM to learn factual knowledge by being clever about masking?
- Thread in several recent works:
  - ERNIE[1]: Enhanced Representation through Knowledge Integration, Sun et al., arXiv 2019
    - Shows improvements on downstream Chinese NLP tasks with phrase-level and entity-level 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[1]: Enhanced Representation through Knowledge Integration

[Sun et al., arXiv 2019]

[1] Yes, another ERNIE paper!

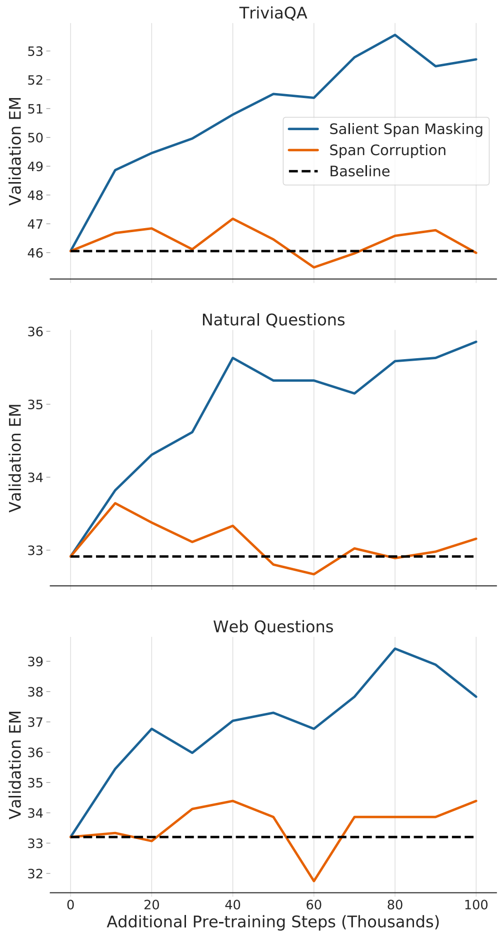


Salient span masking

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

REALM on Natural Questions

Masking technique	Exact Match	Retrieval Recall @5
Salient span masking	**38.2**	**38.5**
Random uniform masks	32.3	24.2
Random span masks	35.3	26.1



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

LAnguage 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].

LAnguage 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)

BERT struggles on N-to-M relations

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

LMs are NOT finetuned!

LAnguage Model Analysis (LAMA) Probe [Petroni et al.,

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

The cat is on the [MASK].

```
bert:

| Top10 predictions
0       phone               -2.345
1       floor               -2.630
2       ground              -2.968
3       couch               -3.387
4       move                -3.649
5       roof                -3.651
6       way                 -3.718
7       run                 -3.757
8       bed                 -3.802
9       left                -3.965

-----------------------------------------------------------------------------------
index   token               log_prob    prediction          log_prob    rank@1000
-----------------------------------------------------------------------------------

1       The                 -5.547      .                   -0.607      14
2       cat                 -0.367      cat                 -0.367      0
3       is                  -0.019      is                  -0.019      0
4       on                  -0.001      on                  -0.001      0
5       the                 -0.002      the                 -0.002      0
6       [MASK]              -14.321     phone               -2.345      -1
7       .                   -0.002      .                   -0.002      0
```

[1] Example courtesy of the authors at link above.

LAnguage 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%

Native language of French-speaking actors according to BERT

Person Name	BERT
Jean Marais	French
Daniel Ceccaldi	Italian
Orane Demazis	Albanian
Sylvia Lopez	Spanish
Annick Alane	English

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!

ID	**Modifications**	**Acc. Gain**
P413	x plays in→at y position	+23.2
P495	x was created→made in y	+10.8
P495	x was→is created in y	+10.0
P361	x is a part of y	+2.7
P413	x plays ~~in~~ y position	+2.2

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

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

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

Model	Precision	Recall	F1
NFGEC (LSTM)	68.8	53.3	60.1
UFET (LSTM)	77.4	60.6	68.0
BERT-Base	76.4	71.0	73.6
ERNIE (Zhang et al.)	78.4	72.9	75.6
KnowBert-W+W	**78.6**	**73.7**	**76.1**

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!