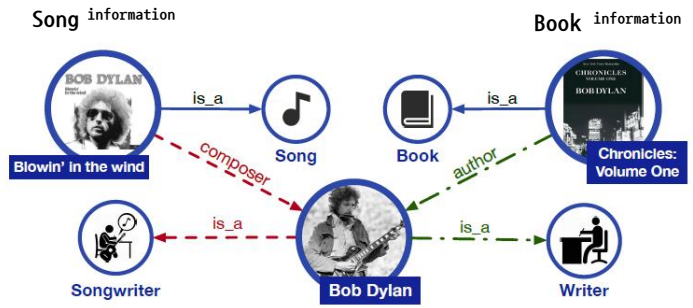
ERNIE,
Knowledge Enhanced Contextual Word Representations,
and LUKE

Sunkyung Park

ERNIE 4 Jun 2019

Enhanced Language Representation w/ Informative Entities

- **Limitations of Existing Research**
  - pre-trained LM neglects knowledge information for language understanding
  - **(not included)** Entity Typing Assigning types to mentions in docs
  - **(not included)** Relation Classification Predicts a relation between two entities in a sentence



Bob Dylan wrote ***Blowin' in the Wind*** in 1962, and wrote ***Chronicles: Volume One*** in 2004.

Existing pre-trained LM : UNK wrote UNK in UNK. → syntactically ambiguous

Rich Knowledge Information: Language Understanding → entity typing & relation classification

- **Challenges** knowledge into language representation models
  - **Structured Knowledge Encoding**
    - How to extract and encode related informative facts in KGs
  - **Heterogeneous Information Fusion**
    - How to **design** a special **pre-training objective** to fuse lexical, syntactic and knowledge information

ERNIE 4 Jun 2019

Enhanced Language Representation w/ Informative Entities

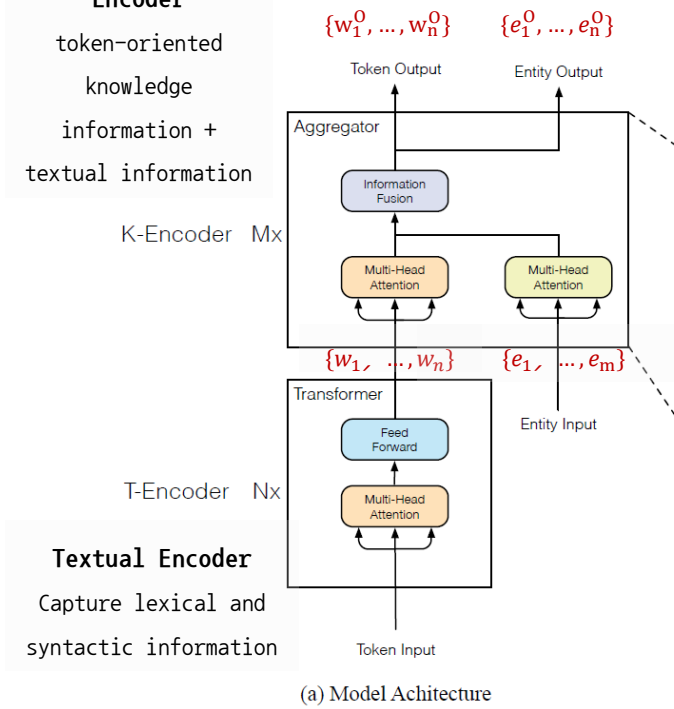
- **Proposed Model** ERNIE
  - Recognize **mentions** in text → Align mentions to **entities** in KGs
  - **Informative entity embeddings**
    - Encode **graph structure** of KGs **w/ knowledge embedding**
  - Integrates **entity representations** in the knowledge module into layers of **the semantic module**
  - **New objectives**
    - Randomly **mask** some of **the named entity alignments** in the input
    - Ask the model to select appropriate entities from KGs
    - Require models to aggregate **context** and **knowledge facts**
    - Predict **tokens** and **entities**

ERNIE 4 Jun 2019

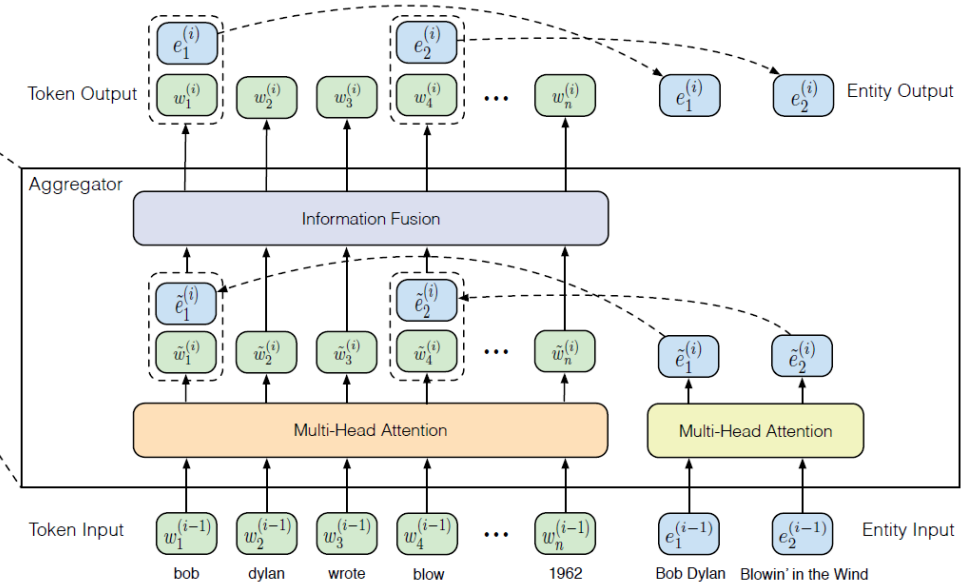
Enhanced Language Representation w/ Informative Entities

Knowledgeable Encoder

token-oriented knowledge information + textual information



(a) Model Achitecture



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

(b) Aggregator

Heterogeneous Information

Textual Encoder

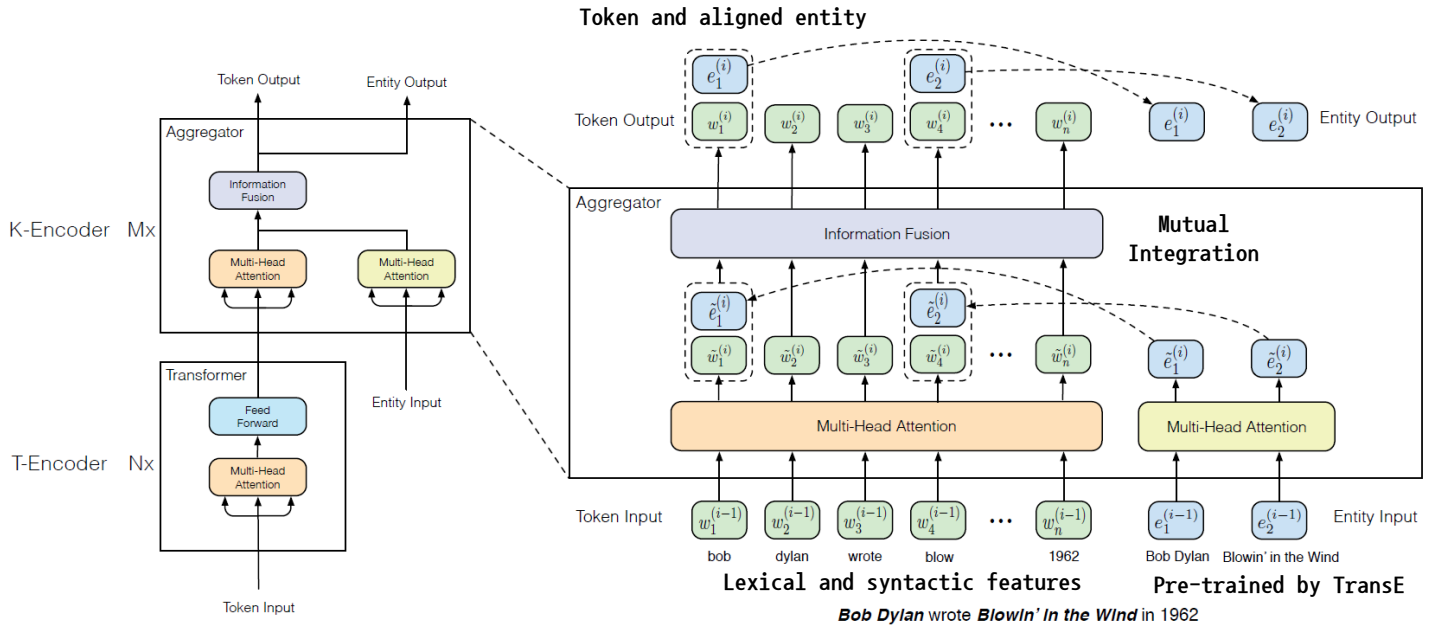
Capture lexical and syntactic information

1. **T-Encoder does**
   - Sum token · segment · positional embedding
   - Compute $\{w_1, \ldots, w_n\}$ lexical and syntactic features
2. **Both $\{w_1, \ldots, w_n\}$ and $\{e_1, \ldots, e_m\}$ pre-trained by TransE fed into K-Encoder**
3. **Output Embeddings ($\{w_1^o, \ldots, w_n^o\}, \{e_1^o, \ldots, e_n^o\}$) will be used for specific tasks**

ERNIE 4 Jun 2019

Enhanced Language Representation w/ Informative Entities

- Knowledgeable Encoder > Information fusion layer



(a) Model Achitecture (b) Aggregator

W/ corresponding entities

Token Entity

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

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

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

W/O corresponding entities

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

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

ERNIE 4 Jun 2019

Enhanced Language Representation w/ Informative Entities

- Pre-training for Injecting Knowledge
  - New pre-training task
    - Randomly masks some **token-entity alignments**
    - Requires the system to **predict all corresponding entities** based on aligned tokens

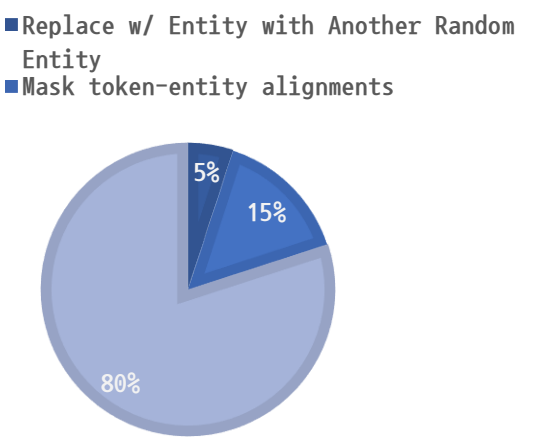
    } dEA denoising entity auto-encoder

$$p(e_j|w_i) = \frac{\exp(\text{linear}(\boldsymbol{w}_i^o) \cdot \boldsymbol{e}_j)}{\sum_{k=1}^{m} \exp(\text{linear}(\boldsymbol{w}_i^o) \cdot \boldsymbol{e}_k)},$$

Given $\{w_1, \ldots, w_n\}^{\text{token sequence}}$ and $\{e_1, \ldots, e_m\}^{\text{corresponding entity sequence}}$,

This is an aligned entity distribution for token w_i

- Operations for dEA



+

- Tokens in Text
  - MLM
  - NSP

ERNIE 4 Jun 2019

Enhanced Language Representation w/ Informative Entities

- Fine-tuning for Specific Tasks

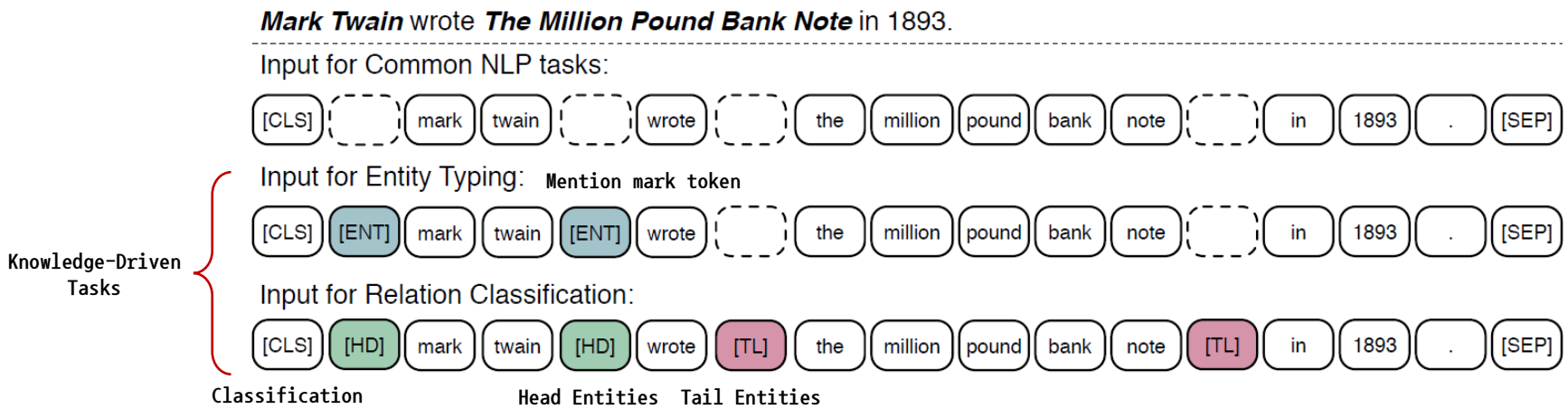


Figure 3: Modifying the input sequence for the specific tasks. To align tokens among different types of input, we use dotted rectangles as placeholder. The colorful rectangles present the specific mark tokens.

- Task 1. Relation Classification
  - Modifies the input token sequence by adding two marktokens to highlight entity mention
- Task 2. Entity Typing
  - Modified input sequence guides ERNIE to combine both context information and entity-mention information

ERNIE 4 Jun 2019

Enhanced Language Representation w/ Informative Entities

Pre-training Dataset

- Before pre-training ERNIE, **Adopts** the knowledge embeddings by TransE
- Multi-Tasks: **NSP**, **MLM**, and **dEA**
- Use Eng Wikipedia and Align text to Wiki-data Convert corpus into **the formatted data**

(Downstream)Entity Typing

- (Task) Given an entity mention and its context, Label the entity mention w/ corresponding semantic types
- (Evaluate) Fine-tunes ERNIE on 2 datasets

- FIGER
  - Train set : Labeled
  - Test set : Annotated by Human
- Open Entity
  - Completely manually-annotated

Dataset	Train	Develop	Test	Type
FIGER	2,000,000	10,000	563	113
Open Entity	2,000	2,000	2,000	6

Table 1: The statistics of the entity typing datasets FIGER and Open Entity.

ACC: Prediction = Human annotations

SOTA on FIGER
2 Bi-LSTMs

Model	Acc.	Macro	Micro
NFGEC (Attentive)	54.53	74.76	71.58
NFGEC (LSTM)	55.60	75.15	71.73
BERT	52.04	75.16	71.63
ERNIE	**57.19**	**76.51**	**73.39**

Precision Recall

1 Bi-LSTM

Model	P	R	F1
NFGEC (LSTM)	68.80	53.30	60.10
UFET	77.40	60.60	68.00
BERT	76.37	70.96	73.56
ERNIE	**78.42**	**72.90**	**75.56**

Table 3: Results of various models on Open Entity (%).

(Compare) ERNIE vs. BERT

(Acc) External Knowledge regularizes ERNIE to fitting the noisy labels

(Compare) BERT & ERNIE vs. Others

Pre-training models **make full use of** manually-annotated data for entity-typing

(Compare) ERNIE vs. BERT

informative entities help predict the labels precisely

ERNIE 4 Jun 2019

Enhanced Language Representation w/ Informative Entities

(Downstream)Relation Classification

- (Task) Determine the correct relation between two entities Head and Tail entities in a sentence
- (Evaluate) Fine-tunes ERNIE on 2 datasets

- **FewRel**
  - **W/O any null instances**
- **TACRED**
  - **W/ 80% null instances**

Dataset	Train	Develop	Test	Relation
FewRel	8,000	16,000	16,000	80
TACRED	68,124	22,631	15,509	42

Table 4: The statistics of the relation classification datasets FewRel and TACRED.

Graph Conv

Model	FewRel P	FewRel R	FewRel F1	TACRED P	TACRED R	TACRED F1
CNN	69.51	69.64	69.35	70.30	54.20	61.20
PA-LSTM	-	-	-	65.70	64.50	65.10
C-GCN	-	-	-	69.90	63.30	66.40
BERT	85.05	85.11	84.89	67.23	64.81	66.00
ERNIE	88.49	88.44	**88.32**	69.97	66.08	**67.97**

Table 5: Results of various models on FewRel and TACRED (%).

FewRel

(Feature) Training data w/ not enough instances

(Compare) CNN vs. BERT & ERNIE

(F1) Pre-training model outperforms

(Compare) BERT vs. ERNIE

(F1) Fusing external knowledge effective

TACRED

(Compare) C-GCN vs. BERT

(F1) C-GCN utilizes **entity mask strategy**

(Compare) BERT vs. ERNIE

(R, F1) effectiveness of the knowledgeable module

Knowledge Enhanced Contextual Word Representations 31 Oct 2019

- **Limitations of Existing Research**
  - Do not contain **any explicit ground** to real word entities → difficult to recover **factual knowledge**
  - Integrate external knowledge into task-specific models

- **(Good Stuff) KB** knowledge bases
  - Rich source
  - Complementary information to that in raw text
  - Encode factual knowledge impossible in infrequent mentions and long-range dependencies

Knowledge Enhanced Contextual Word Representations 31 Oct 2019

- **Proposed Model** KAR
  - Method to insert **multiple KB**s into a large pretrained model w/ KAR Knowledge Attention and Recontextualization
  - Idea in **KAR**
    - Model explicitly **entity spans** in the input text
    - Use **entity linker** to retrieve relevant entity embeddings from a KB
  - Recontextualizes **entity-span representations** w/ **word-to-entity attention → word representation & all entity spans in the context**
  - **KAR** is **inserted between 2 Layers** in the middle of a pre-trained model.
  - Learns **the entity linkers** w/ self-supervision on unlabeled data

- **Benefits**
  - Leaves Original model **unchanged** → Swap out BERT for KnowBert in any downstream tasks.
  - KAR is **Lightweight**
  - **Easy to incorporate** additional KBs by inserting them at other locations

Knowledge Enhanced Contextual Word Representations 31 Oct 2019

KnowBert

- (Definition) incorporates **knowledge bases** into pretrained **BERT** using the **KAR** Knowledge Attention and Recontextualization component
- **Knowledge Bases**
  - w/ **fixed** collection of K entity nodes → computes entity embeddings $e_k \in \mathbb{R}^E$
  - **entity embeddings** includes
    - KBs w/ a **(subj, rel, obj) graph structure**
    - KBs w/ only entity metadata **w/o graph**
    - KBs w/ **a graph** and **entity metadata**
  - **Assumption**
    - The entities are not typed
    - KB accompanied by an **entity candidate selector**
  - **Entity Candidate Selector**
    - **(input)** some text
    - **(output)** list of C potential **entity links** → passed to a context dependent **entity linker** to **discriminate** the candidates

start / end indices of mention span candidate entities in KG

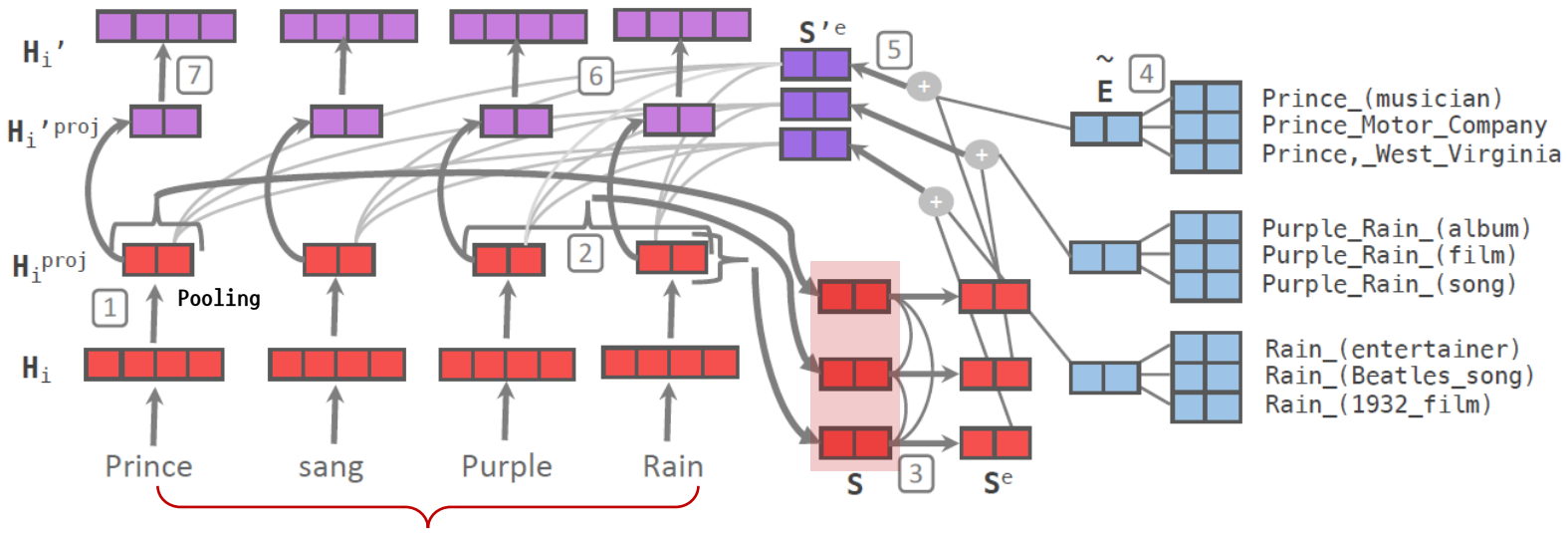
$$\mathcal{C} = \{\langle(\text{start}_m, \text{end}_m), (e_{m,1}, \ldots, e_{m,M_m})\rangle \mid m \in 1 \ldots C, e_k \in 1 \ldots K\}.$$

    - **Over-generate** potential candidates and add a **NULL entity** to each candidate list → allows linker to **discriminate** between actual links and false positive candidates

KAR Knowledge Attention and Recontextualization component

- Mention-span representations

$$\mathbf{H}_i^{\text{proj}} = \mathbf{H}_i \mathbf{W}_1^{\text{proj}} + \mathbf{b}_1^{\text{proj}}.$$



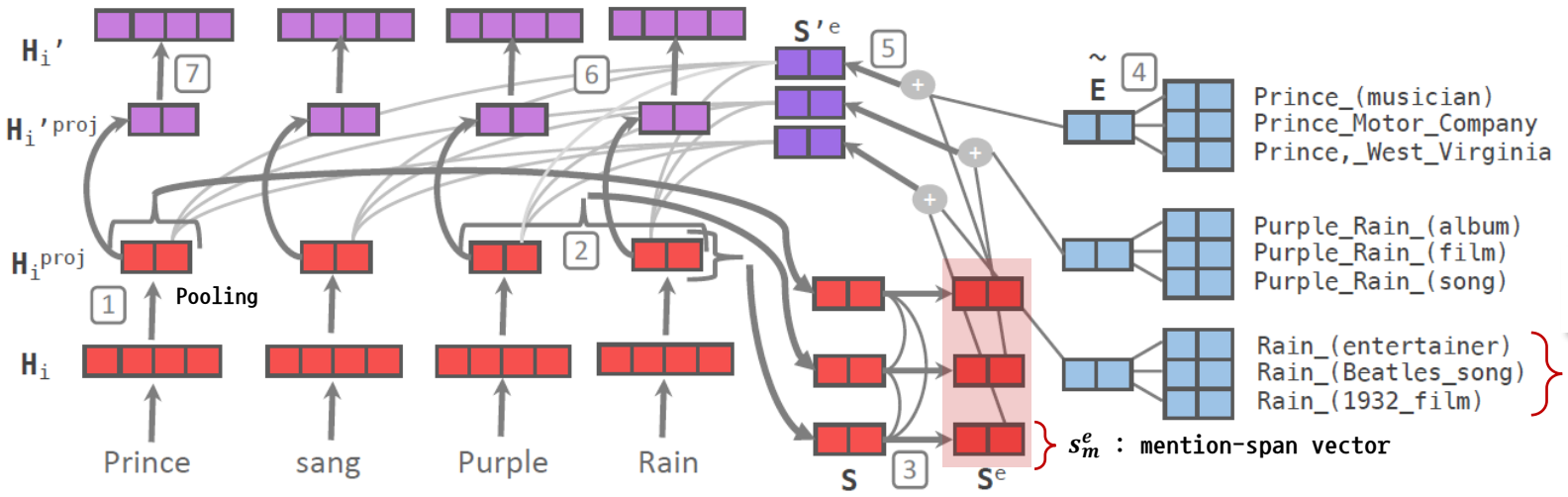
C mention-span representations

- (candidate mention-span) $s_m \in \mathbb{R}^E$
- $S \in \mathbb{R}^{C \times E}$

Knowledge Enhanced Contextual Word Representations 31 Oct 2019

KAR Knowledge Attention and Recontextualization component

- Entity-linker
  - **(Responsibility)** Performs entity disambiguation for each potential mention from candidates



- M_m Scores
- p_{mk} : prior probabilities

$$\psi_{mk} = \text{MLP}(p_{mk}, \mathbf{s}_m^e \cdot \mathbf{e}_{mk}),$$

M_m Candidate entities w/ embeddings e_{mk}

s_m^e : mention-span vector

$$\mathbf{S}^e = \text{TransformerBlock}(\mathbf{S}).$$

Mention-span self attention

KnowBert resolves which of several overlapping candidate mentions should be linked

If supervision is available…

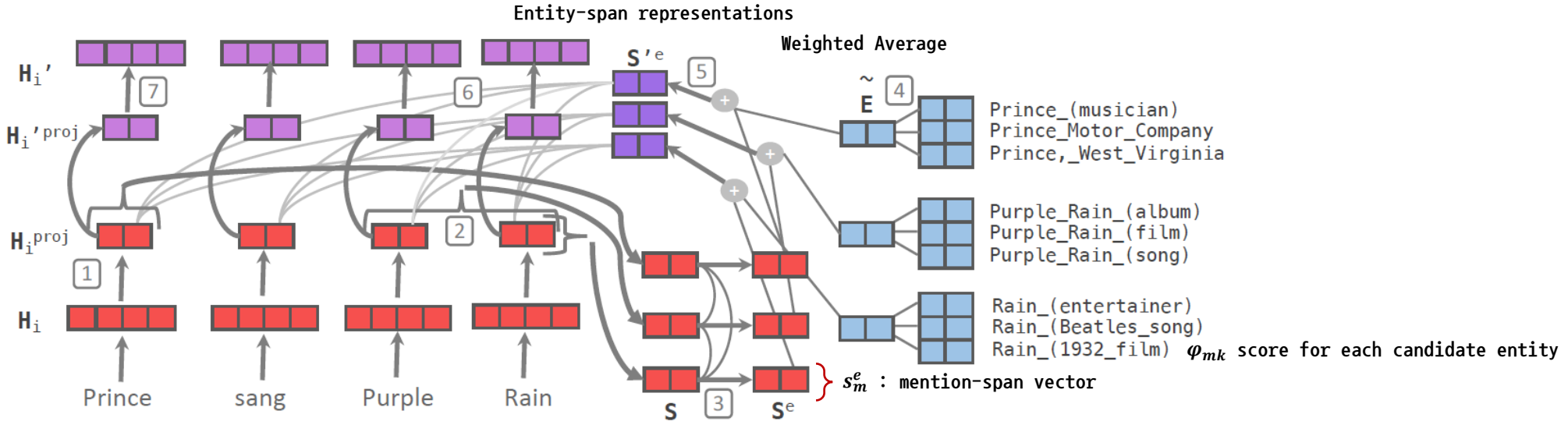
$$\mathcal{L}_{\text{EL}} = -\sum_m \log \left(\frac{\exp(\psi_{mg})}{\sum_k \exp(\psi_{mk})} \right)$$

- e_{mg} gold entity
- Loss depending on the KB

Knowledge Enhanced Contextual Word Representations 31 Oct 2019

KAR Knowledge Attention and Recontextualization component

- **Knowledge enhanced entity-span representations**
  - Injects **the KB entity** info into the mention-span representations to form entity-span representations



$$\tilde{\psi}_{mk} = \begin{cases} \dfrac{\exp(\psi_{mk})}{\sum_{\psi_{mk} \geq \delta} \exp(\psi_{mk})}, & \psi_{mk} \geq \delta \\ 0, & \psi_{mk} < \delta. \end{cases}$$

- Disregard all candidates below a threshold

$$\tilde{\mathbf{e}}_m = \sum_k \tilde{\psi}_{mk} \mathbf{e}_{mk}.$$

- Weighted entity embedding
- $\tilde{e}_m$ < below δ → NULL embedding

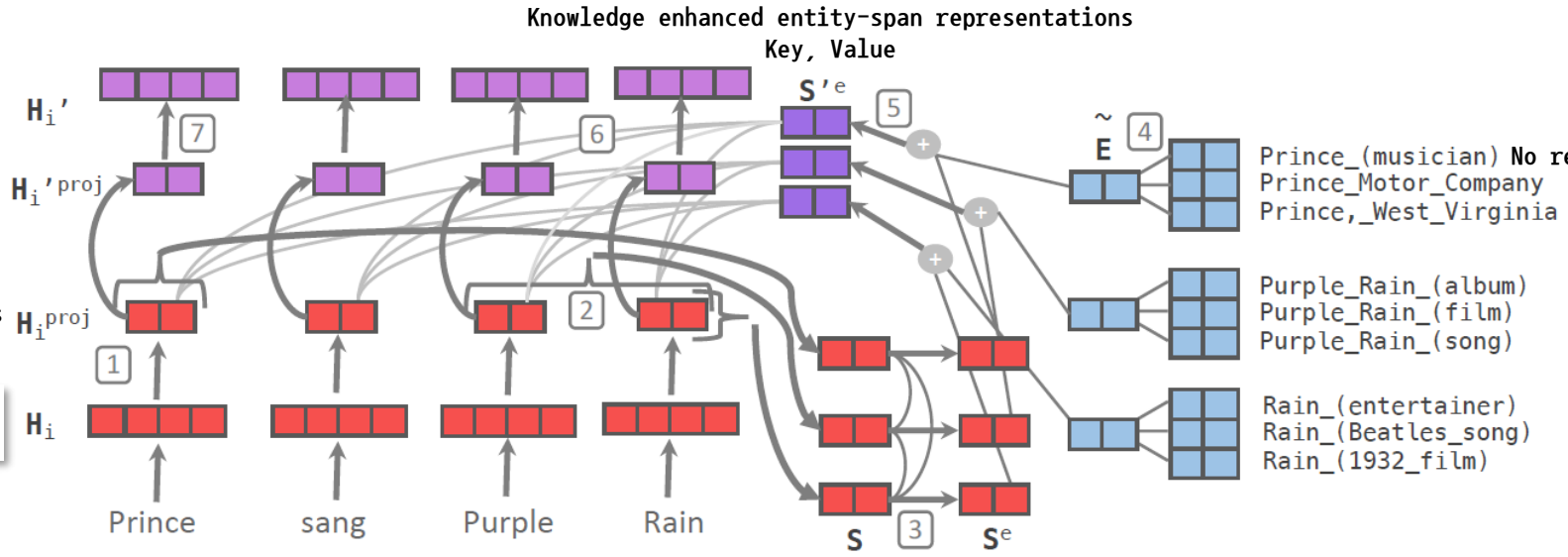
$$\mathbf{s}_m'^e = \mathbf{s}_m^e + \tilde{\mathbf{e}}_m,$$

- Entity-span representations updated
- $S'^e \in \mathbb{R}^{C \times E}$

Knowledge Enhanced Contextual Word Representations 31 Oct 2019

KAR Knowledge Attention and Recontextualization component

- **Recontextualization**
  - Uses entity-span representations to recontextualize the word piece representations



No restriction on the entity embeddings

$$\mathbf{H}_i^{\prime\text{proj}} = \text{MLP}(\text{MultiHeadAttn}(\mathbf{H}_i^{\text{proj}}, \mathbf{S}'^e, \mathbf{S}'^e)).$$

- Word-to-entity span attention
- Propagates entity information over long contexts

$$\mathbf{H}'_i = \mathbf{H}_i^{\prime\text{proj}} \mathbf{W}_2^{\text{proj}} + \mathbf{b}_2^{\text{proj}} + \mathbf{H}_i$$

- Projected back to the BERT dimension
- Alignment of BERT and entity vectors
  - Initialize W_2^{proj} as the inverse of W_1^{proj}

Experiment

- **Setup**
  - **BERT** $_{BASE}$
    - **(+)** biasing dataset to shorter sequences of 128 words pieces
  - **KnowBert-Wiki**
    - Insert the KB $^{Knowledge\ Base}$ between layers 10 and 11 of 12 layers BERT
  - **KnowBert-WordNet**
    - Synset metadata, Lemma metadata, and the relational graph
    - Insert the KB $^{Knowledge\ Base}$ between layers 10 and 11 of 12 layers BERT
  - **KnowBert-W** ^{Wiki} **+ W** ^{WordNet}
    - Insert the Wikipedia KB $^{Knowledge\ Base}$ between layers 10 and 11, with WordNet between 11 and 12.

Experiment

- **Downstream Tasks**
  - **(evaluate)** Validates that the knowledge addition improves performance on tasks

	System	LM	P	R	F_1
	Zhang et al. (2018)	—	69.9	63.3	66.4
	Alt et al. (2019)	GPT	70.1	65.0	67.4
	Shi and Lin (2019)	$BERT_{BASE}$	73.3	63.1	67.8
ERNIE	Zhang et al. (2019)	$BERT_{BASE}$	70.0	66.1	68.0
	Soares et al. (2019)	$BERT_{LARGE}$	—	—	70.1
Matching in the blanks	Soares et al. (2019)	$BERT_{LARGE}$†	—	—	**71.5**
	KnowBert-W+W	$BERT_{BASE}$	71.6	71.4	**71.5**

Table 4: Single model test set results on the TACRED relationship extraction dataset. † with MTB pretraining.

System	Accuracy
ELMo†	57.7
$BERT_{BASE}$†	65.4
$BERT_{LARGE}$†	65.5
$BERT_{LARGE}$††	69.5
KnowBert-W+W	**70.9**

Table 6: Test set results for the WiC dataset (v1.0).
†Pilehvar and Camacho-Collados (2019)
††Wang et al. (2019a)

- **Relation extraction**
  - **Settings**
    - (dataset) TACRED and SemEval 2010 Task 8
    - (given) a sentence w/ marked a Sub and Obj → (pred) which realtation
  - **Model**
    - [E1][/E1] Loc of Sub, [E2] [/E2] Loc of Obj
    - Concatenate the contextual representations of them to predict relation
  - **Results**
    - Better than ERNIE, $BERT_{LARGE}$
    - Match the MTB relation specific pretraining
- **Words in Context**
  - **Definition**
    - **2 sentences** containing a word with the same lemma → determine if they are **the same sense or not** → Test **contextual word representations**
  - **Settings**
    - [CLS] fine tuning Sentence 1 [SEP] Sentence 2
  - **Results**
    - Achieves **SOTA** in Accuracy

Experiment

- Downstream Tasks

System	P	R	F_1
UFET	68.8	53.3	60.1
$BERT_{BASE}$	76.4	71.0	73.6
ERNIE	78.4	72.9	75.6
KnowBert-W+W	78.6	73.7	**76.1**

Table 7: Test set results for entity typing using the nine general types from (Choi et al., 2018).

- **Entity-Typing**
  - **Settings**
    - (dataset) Choi et al.(2018)
  - **Model**
    - Evaluation protocol w/ 9 general entity types
    - [E][/E] mark the loc of a target span
  - **Results**
    - Shows improvements F1 over ERNIE and $BERT_{BASE}$

LUKE: Deep Contextualized Entity Representations w/ Entity-aware Self-attention 2 Oct 2020

- **Limitations of Existing Research**

  - Requires entity linking to represent entities in a text, and can**not represent** entities **not in the KB**
  - **(Alternative)** CWRs provides word representations trained w/ **unsupervised pretraining task**

  - **CWR** Contextualized word representation
    - Many studies solved entity-related tasks w/ **contextualized representations** of entities **computed on CWR**
    - CWR Transformer based **not suited** to representing **entities**
      - (Why) CWRs do not output **the span-level representations** of entities
      - (Why) CWRs are **difficult** to perform **reasoning** between entities multiple tokens
        - (In contrary) Possible to capture the complex relationships between words

        "The Lord of the [MASK]"

      - (e.g) **Word-based** pretraining task of CWRs
        - Predicting a masked word given other words in the entity
        - **Not suitable for** learning the representations of entities Book, Movie

LUKE: Deep Contextualized Entity Representations w/ Entity-aware Self-attention [2 Oct 2020]

- **Proposed Model** [LUKE]
  - New pretrained **contextualized representations of words and entities**
  - **Difference between LUKE and CWR**
    - **Treats** not only words but also **entities** as **independent tokens**
    - Computes **intermediate** and **output representations** for **all tokens** w/ transformer
  - **Task**
    - Randomly **masking entities** by [MASK] entities
    - Trains model by predicting **the originals of the masked entities.**
  - **Contribution: Model**
    - Base pre-trained model : **RoBERTa**
    - (+)conduct pretraining of the model by optimizing **MLM** and **proposed task**
    - (downstream task) the resulting model **computes representations of arbitrary entities** in the text using [MASK] entities
  - **Contribution: entity-aware self-attention**
    - **Extends** the transformer using entity-aware self-attention mechanism.
    - **Determine easily** the types of tokens.

LUKE: Deep Contextualized Entity Representations w/ Entity-aware Self-attention 2 Oct 2020

- **Input Representation of a token**

  - **Token Embedding**
    - (Word token embedding) $A \in \mathbb{R}^{V_w \times D}$
    - (Entity token embedding) $B \in \mathbb{R}^{V_e \times H}$ and $U \in \mathbb{R}^{H \times D}$
  - **Position Embedding** i-th pos in seq
    - (Word) $C_i \in \mathbb{R}^D$
    - (Entity) $D_i \in \mathbb{R}^D$
      - Multiple words

      → avg the embeddings of the corresponding positions
  - **Entity Type Embedding**
    - Single vector $e \in \mathbb{R}^D$

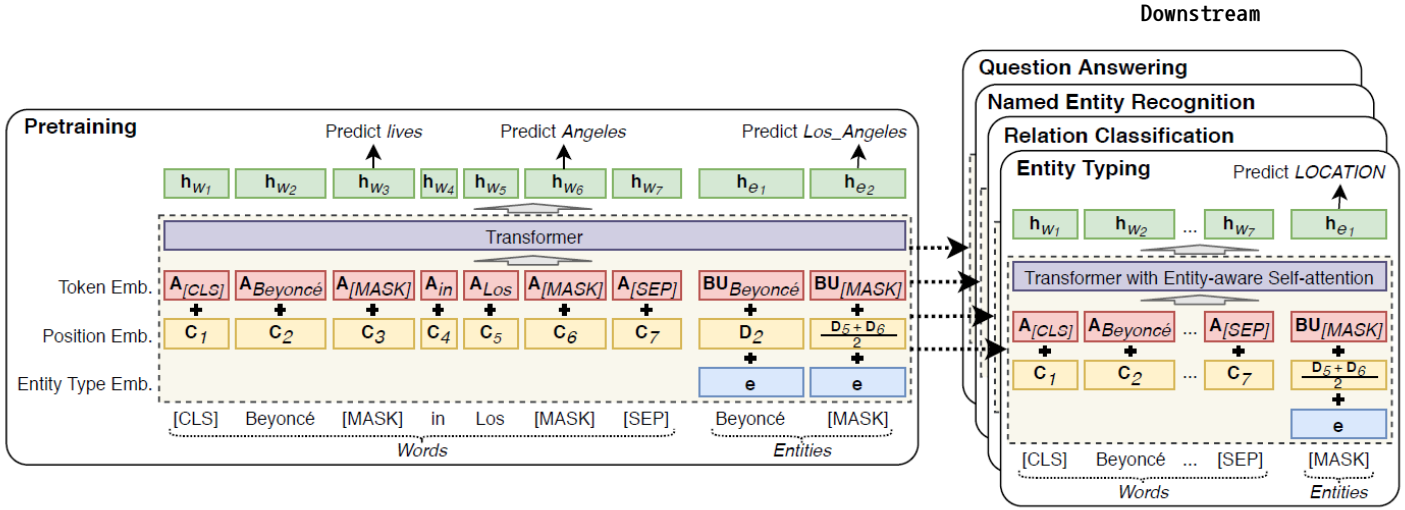


Figure 1: Architecture of LUKE using the input sentence "*Beyoncé lives in Los Angeles.*" LUKE outputs contextualized representation for each word and entity in the text. The model is trained to predict randomly masked words (e.g., *lives* and *Angeles* in the figure) and entities (e.g., *Los_Angeles* in the figure). Downstream tasks are solved using its output representations with linear classifiers.

LUKE: Deep Contextualized Entity Representations w/ Entity-aware Self-attention 2 Oct 2020

- **Entity-aware Self-attention**
  - Relates token each other based on **the attention score** between each pair of tokens

A seq of input or output vector

- $x_1, \ldots, x_k$ *where* $x_i \in \mathbb{R}^D$
- $y_1, \ldots, y_k$ *where* $y_i \in \mathbb{R}^L$
- $\boldsymbol{k = m + n}$ **(word or entity)**

$$\mathbf{y}_i = \sum_{j=1}^{k} \alpha_{ij} \mathbf{V} \mathbf{x}_j$$

$$e_{ij} = \frac{\mathbf{K}\mathbf{x}_j^\top \mathbf{Q}\mathbf{x}_i}{\sqrt{L}}$$

$$\alpha_{ij} = \mathrm{softmax}(e_{ij})$$

How?

$$e_{ij} = \begin{cases} \mathbf{K}\mathbf{x}_j^\top \mathbf{Q}\mathbf{x}_i, & \text{if both } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are words} \\ \mathbf{K}\mathbf{x}_j^\top \mathbf{Q}_{w2e}\mathbf{x}_i, & \text{if } \mathbf{x}_i \text{ is word and } \mathbf{x}_j \text{ is entity} \\ \mathbf{K}\mathbf{x}_j^\top \mathbf{Q}_{e2w}\mathbf{x}_i, & \text{if } \mathbf{x}_i \text{ is entity and } \mathbf{x}_j \text{ is word} \\ \mathbf{K}\mathbf{x}_j^\top \mathbf{Q}_{e2e}\mathbf{x}_i, & \text{if both } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are entities} \end{cases}$$

- **Assumption**
  - Beneficial to use the information of **target token types** when computing e_{ij}
- **Entity-aware query mechanism**
  - Uses **a different query matrix** for each pair of x_i and x_j

- **Different Query Matrices**
  - $Q_{w2e}, Q_{e2w}, Q_{e2e} \in \mathbb{R}^{L \times D}$

LUKE: Deep Contextualized Entity Representations w/ Entity-aware Self-attention 2 Oct 2020

- **Pretraining Task**
  - **Conventional MLM**
  - **An extension of the MLM** to learn entity representations
    - **Settings**
      - **(entity annotation)** Hyperlinks in Wikipedia
      - **(entity annotated corpus)** Retrieved from Wikipedia
    - **Operations**
      - Randomly mask by replacing entities w/ [MASK]
      - The original entity to a masked one is predicted by the softmax function **over all entities in voca.**

$$\hat{\mathbf{y}} = \mathrm{softmax}(\mathbf{B}\mathbf{T}\mathbf{m} + \mathbf{b}_o)$$
$$\mathbf{m} = \mathrm{layer\_norm}\big(\mathrm{gelu}(\mathbf{W}_h\mathbf{h}_e + \mathbf{b}_h)\big)$$

B : all entities
T : representation corresponding to the masked entity

  - **Final Loss**
    - **MLM loss + Cross entropy loss on MLM extension**

LUKE: Deep Contextualized Entity Representations w/ Entity-aware Self-attention 2 Oct 2020

- **Experiments**

Name	Prec.	Rec.	F1
BERT (Zhang et al., 2019)	67.2	64.8	66.0
C-GCN (Zhang et al., 2018b)	69.9	63.3	66.4
ERNIE (Zhang et al., 2019)	70.0	66.1	68.0
SpanBERT (Joshi et al., 2020)	70.8	70.9	70.8
MTB (Baldini Soares et al., 2019)	-	-	71.5
KnowBERT (Peters et al., 2019)	**71.6**	71.4	71.5
KEPLER (Wang et al., 2019b)	70.4	73.0	71.7
K-Adapter (Wang et al., 2020)	68.9	**75.4**	72.0
RoBERTa (Wang et al., 2020)	70.2	72.4	71.3
LUKE	70.4	75.1	**72.7**

Table 2: Results of relation classification on the TACRED dataset.

Primary baseline

Name	F1
LSTM-CRF (Lample et al., 2016)	91.0
ELMo (Peters et al., 2018)	92.2
BERT (Devlin et al., 2019)	92.8
Akbik et al. (2018)	93.1
Baevski et al. (2019)	93.5
RoBERTa	92.4
LUKE	**94.3**

Table 3: Results of named entity recognition on the CoNLL-2003 dataset.

- **Entity-Typing**
  - **Settings**
    - (dataset) Open Entity
    - **9** general entity types
    - micro-precision, recall, and F1
  - **Model**
    - Treat the task as **multi-label classification**
    - Train w/ **Binary-cross entropy** averaged over **all** entity types
  - **Results**
    - Achieves **SOTA** in F1 points

- **Relation Classification**
  - **Settings**
    - (dataset) TACRED
    - micro-precision, recall, and F1
  - **Model**
    - [HEAD], [TAIL]
    - Use a concatenated representations of the head and tail entities
    - Traub w/ **cross-entropy**
  - **Results**
    - Achieves **SOTA** in F1 points

ERNIE vs. KAR vs. LUKE

- **ERNIE** 4 Jun 2019
  - **(Purpose)** incorporate knowledge information into language representation models
  - **(Proposed)** For **better fusion** of heterogeneous info from both text and KGs
    - knowledgeable aggregator
    - Pre-training task dEA
  - (Experimental Result) **Better** abilities than BERT
  - (Future research) Injecting knowledge into **feature-based** pre-training models, introducing diverse **structured knowledge** into **different** language representation model

- **KAR** 31 Oct 2019
  - **(Purpose)** presents an efficient and general method to insert **prior knowledge** into a Neural Net.
  - **(Proposed)** incorporates KB into BERT using **Knowledge Attention** and **Recontextualization** component
  - (Experimental Result) **Improvements** for relationship extraction, entity typing and word sense disambiguation datasets
  - (Future research) **incorporates a diverse set** of domain specific **KBs**

- **LUKE** 2 Oct 2020
  - **(Limitation)** unsupervised pretraining task, **CWR**
  - **(Purpose) newly pretrained contextualized** representations of words and entities
  - **(Proposed) entity-aware self-attention mechanism**
  - (Experimental Result) **proves** its effectiveness on tasks
  - (Future research) **apply** LUKE to **domain-specific** biomedical, legal tasks

KiSTi 한국과학기술정보연구원 Korea Institute of Science and Technology Information www.kisti.re.kr

UST

NLP Seminar

KEPLER & GreaseLM

2022. 06. 03

KISTI - UST IKJE CHOI

CONTENTS

KEPLER

- Title : KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation

- Google Scholar

KEPLER: A **unified** model for knowledge embedding and pre-trained language representation

X Wang, T Gao, Z Zhu, Z Zhang, Z Liu, J Li… - Transactions of the …, 2021 - direct.mit.edu

… In this paper, we propose **KEPLER**, a simple but effective **unified** model for knowledge embedding and pre-trained language representation. We train **KEPLER** with both the KE and …

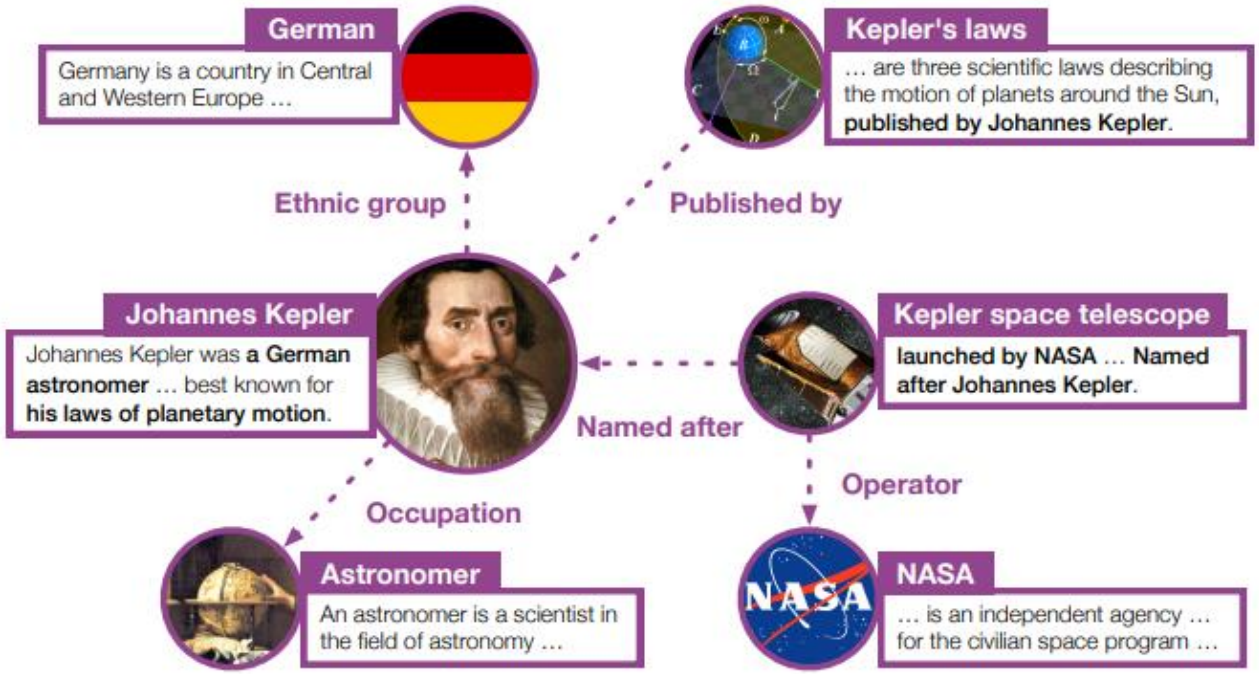
☆ Save Cite Cited by 160 Related articles All 12 versions

KEPLER

- KEPLER
  - ✓ Pre-trained language representation model :
    not good to capture factual knowledge from text
  - ✓ Knowledge graph :
    not good no abundant textual information.
  - ✓ Propose a unified model KEPLER which integrates factual knowledge into pre-trained language representation model by using Knowledge graph

KEPLER's Entity description

- KEPLER
  - ✓ Entity description is used to fill the gap between KE and PLM.
  - ✓ Using PLM encoder, we learn the model by encoding text and entity description into the unified semantic space and by jointly optimizing KE objective and MLM objective.



KEPLER's strengths

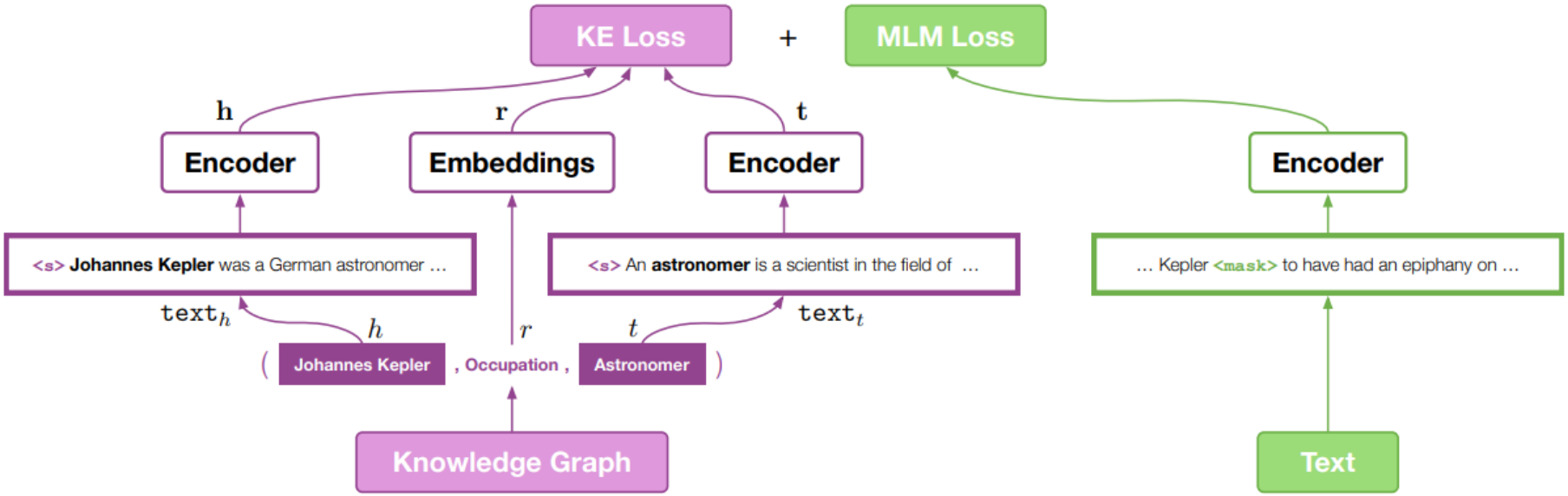
- As a PLM

  - ✓ KEPLER can integrate the actual knowledge into the language presentation using KE objective.
  - ✓ It uses MLM objective to embed a strong language understanding capability of PLM.
  - ✓ KE objective requires a model to encode an entity from an entity corresponding description, thus improving the possibility of extracting knowledge from text.
  - ✓ KEPLER does not modify the model structure but only adds training objectives, so it can be applied directly to NLP tasks like other PLMs

KEPLER's strengths

- As a KE model
  - ✓ KEPLER will be able to make good use of abundant information from the entity description using MLM objective.
  - ✓ KEPLER can also perform knowledge embedding in inductive setting.

KEPLER framework



- KE Loss and MLM Loss are learned in different mini batches.

KEPLER

- Knowledge Embedding in KEPLER
  - ✓ Knowledge Embedding (KE) is an encoding of the entity and relation of a knowledge graph (KG) as distributed representation.
  - ✓ In KEPLER, Using corresponding text as the entity and encode it to vector. There are three kinds of corresponding text that can be used.

    (1) Entity description

    (2) Entity and relation description

    (3) Entity embeddings conditioned on relations

KEPLER

- Knowledge Embedding in KEPLER

$$\mathbf{h} = \mathrm{E}_{<\mathrm{s}>}(\mathtt{text}_h),$$
$$\mathbf{t} = \mathrm{E}_{<\mathrm{s}>}(\mathtt{text}_t),$$
$$\mathbf{r} = \mathbf{T}_r,$$

  - ✓ Three kinds of corresponding text

(1) Entity description

- Encodes the description text related to the head entity and uses it as an embedding of the head entity.

- Tail entity is also used by obtaining embeddings in the same way.

- <s> means that special token <s> at the beginning and end of the text.

KEPLER

- Knowledge Embedding in KEPLER

  - ✓ Three kinds of corresponding text

$$\mathbf{h} = \mathrm{E}_{<s>}(\mathrm{text}_h),$$
$$\mathbf{t} = \mathrm{E}_{<s>}(\mathrm{text}_t),$$
$$\hat{\mathbf{r}} = \mathrm{E}_{<s>}(\mathrm{text}_r),$$

(2) Entity and relation description

- The embeddings of the head entity and tail entity are obtained by using the same method as the previous entity description.

- The embeddings of the relation type are also obtained by encoding the relation corresponding text description.

- <s> means that special token <s> at the beginning and end of the text.

KEPLER

- Knowledge Embedding in KEPLER

$$\mathbf{h}_r = \mathrm{E}_{<s>}(\mathrm{text}_{h,r}),$$
$$\mathbf{t} = \mathrm{E}_{<s>}(\mathrm{text}_t),$$
$$\mathbf{r} = \mathbf{T}_r,$$

  - ✓ Three kinds of corresponding text

(3) Entity embeddings conditioned on relations

- It is a method derived from the intention that different relations will have different embeddings.

- Concatenate head entity and the relation entity, and uses is as an embedding of head entity.

KEPLER

- Loss function in KEPLER

$$\mathcal{L}_{\mathrm{KE}} = -\log \sigma(\gamma - d_r(\mathbf{h}, \mathbf{t})) - \sum_{i=1}^{n} \frac{1}{n} \log \sigma(d_r(\mathbf{h_i'}, \mathbf{t_i'}) - \gamma),$$

- Gamma means margin, and dr(h,t) means score function.

- The dr(h,t) used here is derived from the TransE model.

- The case of 'hi' and 'ti' means negative sample.

- Using negative sampling, related entities are learned close to each other and unrelated entities are learned far.

KEPLER

- Loss function in KEPLER

$$d_r(\mathbf{h}, \mathbf{t}) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_p,$$

- The dr(h,t) used here is derived from the TransE model.

- In the case of TransE, the entity is used as a vector, and the equation is designed that the sum of the head entity and the relation type is similar to the tail entity, resulting in the following score function. (h + r = t)

KEPLER

- Conclusion

  - ✓ In this paper, proposing a simple but effective unified model KEPLER for knowledge embedding and pre-trained language presentation.

  - ✓ KEPLER uses KE and MLM objective to learn to align factual knowledge and language repression in the same semantic space, showing good performance in many NLP and KE applications.

GreaseLM

- Title : GreaseLM: Graph REASoning Enhanced Language Models for Question Answering

- Google Scholar

GreaseLM: Graph REASoning Enhanced Language Models for Question Answering

X Zhang, A Bosselut, M Yasunaga, H Ren… - arXiv preprint arXiv …, 2022 - arxiv.org

… In this work, we propose **GREASELM**, a new model that fuses encoded representations …

-USMLE) domains demonstrate that **GREASELM** can more reliably answer questions that …

☆ Save 99 Cite Cited by 1 Related articles All 2 versions »

GreaseLM

- Main Task : Solving QA task
  - ✓ Existing Method : Knowledge Graph + Language Models fuse in a shallow.

    It means that they encode separately and fuse them at the output for a prediction.

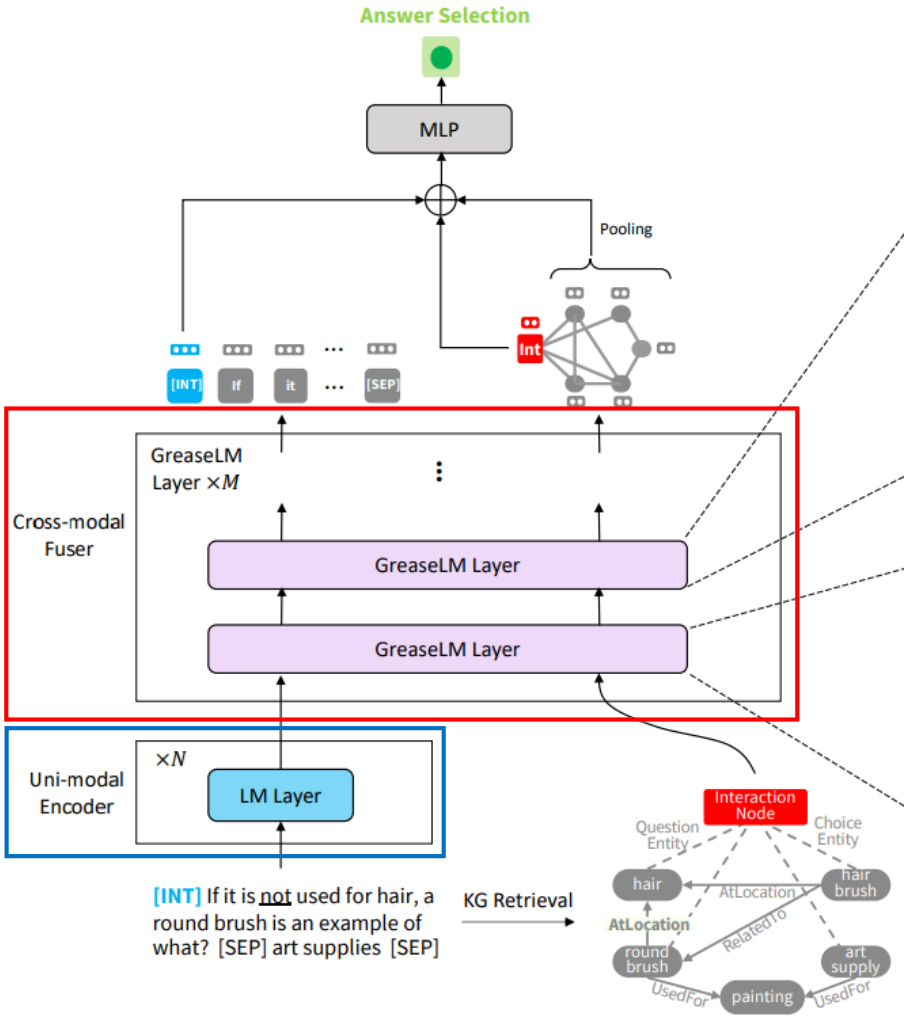
    -> so, they do not exchange useful information each other.

  - ✓ Proposed Model : GreaseLM

    Fuses pre-trained language models (PLMs) and Knowledge graphs(KGs) then encodes and operates.

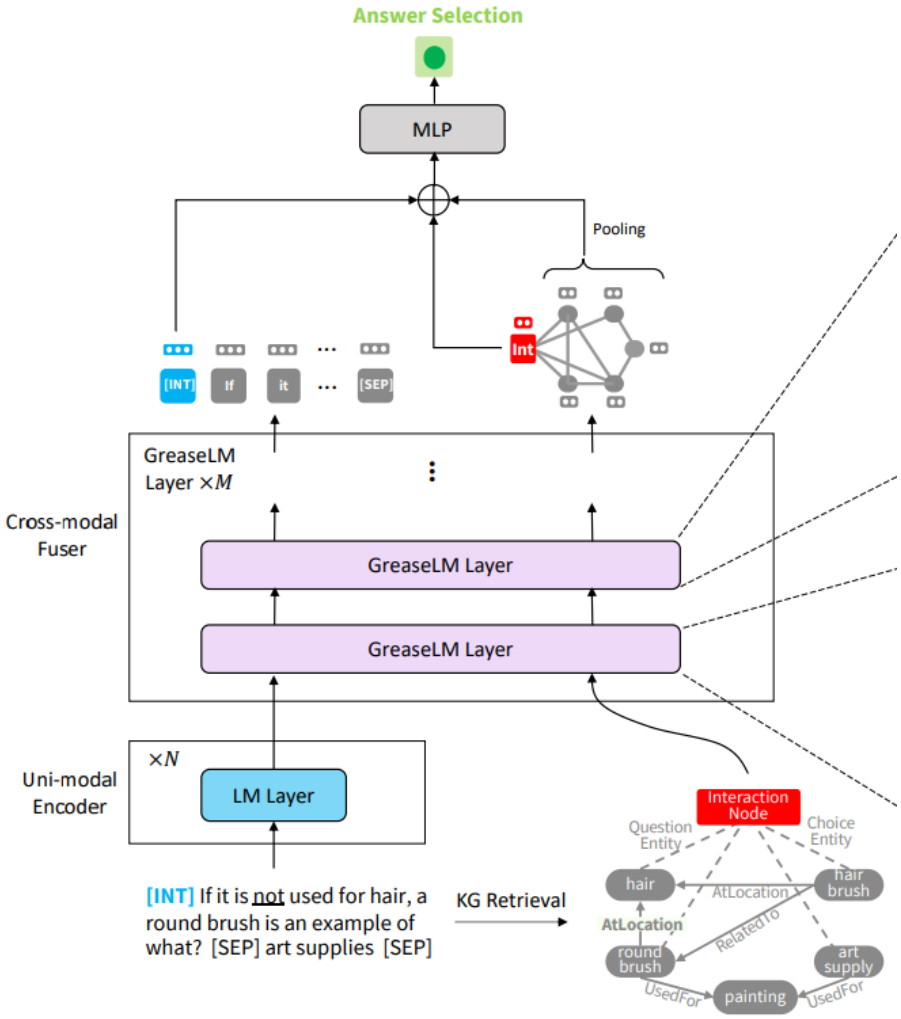
GreaseLM Architecture

- GreaseLM consists of two stacked components
  - ✓ Unimodal LM layers (N) : initial representation of input tokens
  - ✓ Cross-modal GreaseLM layers (M) : LM layer + GNN layer



GreaseLM Architecture

- Input Representation
  - ✓ Concatenate cotext paragraph c, question q, candidate answer a with SEP tokens
  - ✓ Use input sequence to retrieve subgraph of the KG
  - ✓ Special interaction token W : prepend to token sequence
  - ✓ Specoal interaction node E : connect to all linked nodes in V in G

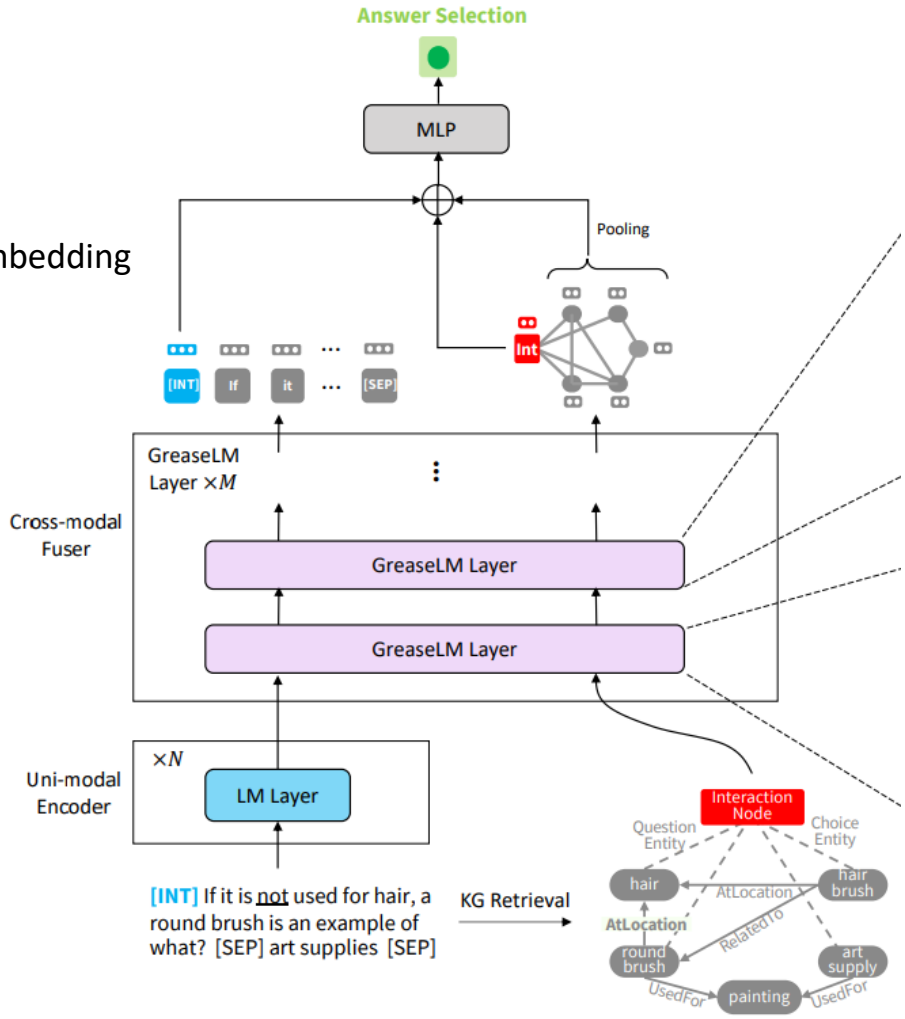


GreaseLM Architecture

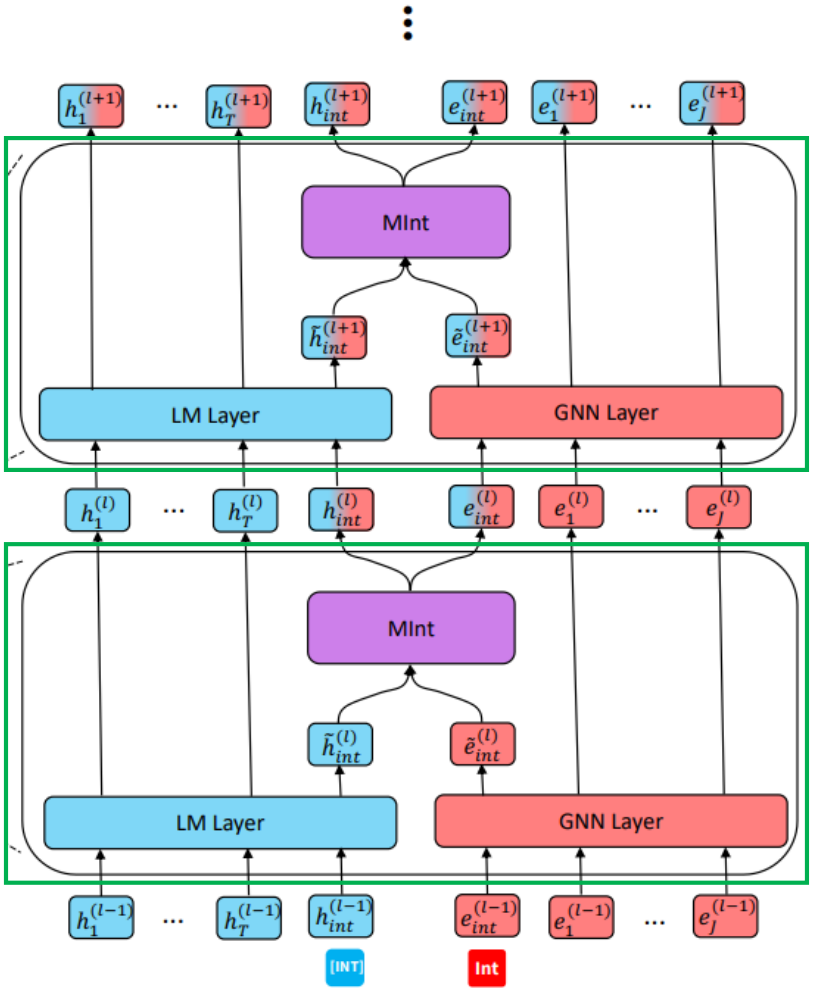
- Language Pre-encoding

$$\{\boldsymbol{h}_{int}^{(0)}, \boldsymbol{h}_1^{(0)}, \ldots, \boldsymbol{h}_T^{(0)}\} = \{w_{int}, w_1, \ldots, w_T\} + \text{segment + positional embedding}$$

$$\{\boldsymbol{h}_{int}^{(\ell)}, \boldsymbol{h}_1^{(\ell)}, \ldots, \boldsymbol{h}_T^{(\ell)}\} = \text{LM-Layer}(\{\boldsymbol{h}_{int}^{(\ell-1)}, \boldsymbol{h}_1^{(\ell-1)}, \ldots, \boldsymbol{h}_T^{(\ell-1)}\}) \quad \text{for } \ell = 1, \ldots, N$$

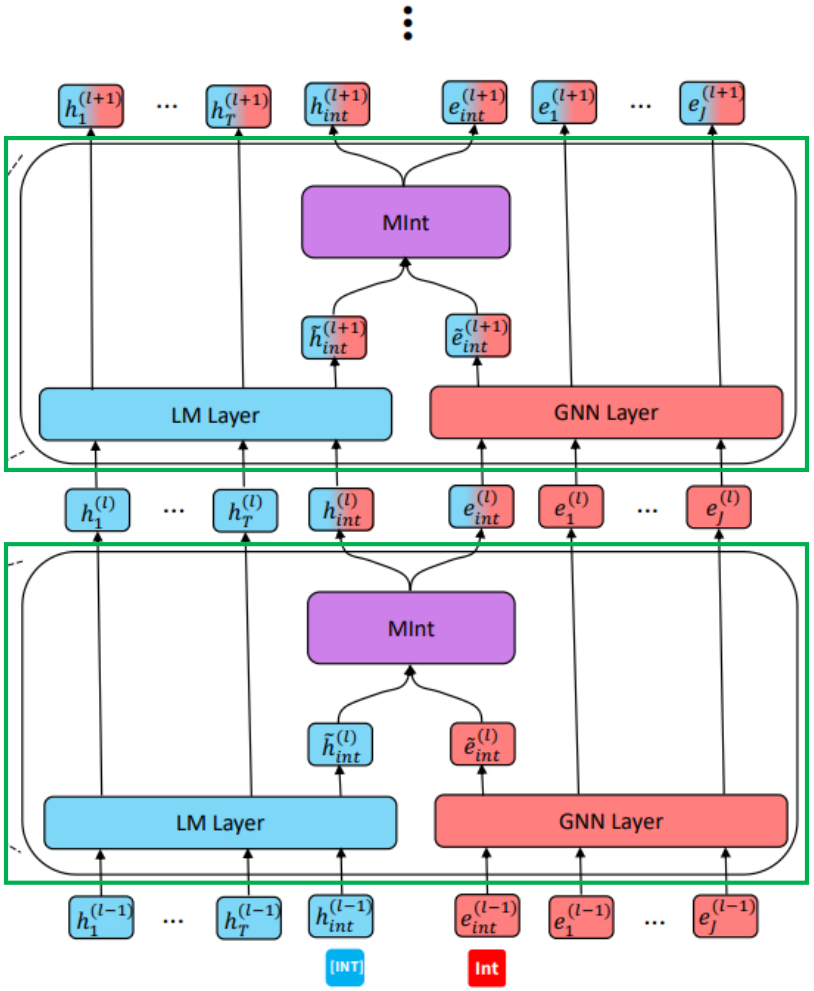


- What GreaseLM Layer does
  - ✓ Separately encode information from both modalities(LM and GNN layer)
  - ✓ Fuse both representations using the special interaction token and node
- Three Components of a GreaseLM layer
  - ✓ A transformer LM encoder block : continues to encode language context
  - ✓ A GNN layer : reason over KG
  - ✓ A modality interaction layer : two modalities fuse info



- Three Components of a GreaseLM layer
  - ✓ A transformer LM encoder block : continues to encode language context

$$\{\tilde{\boldsymbol{h}}_{int}^{(N+\ell)}, \tilde{\boldsymbol{h}}_1^{(N+\ell)}, \ldots, \tilde{\boldsymbol{h}}_T^{(N+\ell)}\} = \text{LM-Layer}(\{\boldsymbol{h}_{int}^{(N+\ell-1)}, \boldsymbol{h}_1^{(N+\ell-1)}, \ldots, \boldsymbol{h}_T^{(N+\ell-1)}\}) \quad \text{for } \ell = 1, \ldots, M$$



- Three Components of a GreaseLM layer
  - ✓ A GNN layer : reason over KG

(1) Compute initial node embedding
(2) Initial embedding of the interaction node is initialized randomly
(3) In each layer of GNN

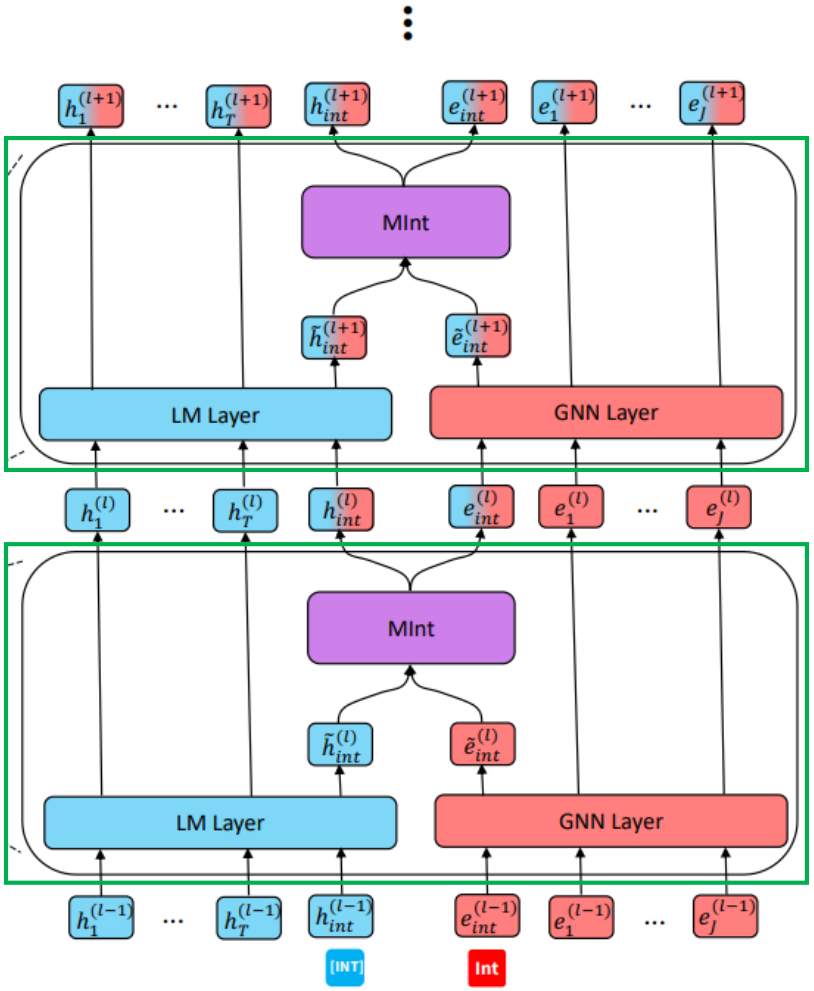
$$\{\tilde{\boldsymbol{e}}_{int}^{(\ell)}, \tilde{\boldsymbol{e}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{e}}_J^{(\ell)}\} = \text{GNN}(\{\boldsymbol{e}_{int}^{(\ell-1)}, \boldsymbol{e}_1^{(\ell-1)}, \ldots, \boldsymbol{e}_J^{(\ell-1)}\}) \quad \text{for } \ell = 1, \ldots, M$$

$$\tilde{\boldsymbol{e}}_j^{(\ell)} = f_n\left(\sum_{e_s \in \mathcal{N}_{e_j} \cup \{e_j\}} \alpha_{sj} \boldsymbol{m}_{sj}\right) + \boldsymbol{e}_j^{(\ell-1)}$$

$$\boldsymbol{r}_{sj} = f_r(\tilde{\boldsymbol{r}}_{sj}, \boldsymbol{u}_s, \boldsymbol{u}_j) \qquad \boldsymbol{m}_{sj} = f_m(\boldsymbol{e}_s^{(\ell-1)}, \boldsymbol{u}_s, \boldsymbol{r}_{sj})$$

$$\boldsymbol{q}_s = f_q(\boldsymbol{e}_s^{(\ell-1)}, \boldsymbol{u}_s) \qquad \boldsymbol{k}_j = f_k(\boldsymbol{e}_j^{(\ell-1)}, \boldsymbol{u}_j, \boldsymbol{r}_{sj})$$

$$\gamma_{sj} = \frac{\boldsymbol{q}_s^\top \boldsymbol{k}_j}{\sqrt{D}} \qquad \alpha_{sj} = \frac{\exp(\gamma_{sj})}{\sum_{e_s \in \mathcal{N}_{e_j} \cup \{e_j\}} \exp(\gamma_{sj})}$$



- Three Components of a GreaseLM layer

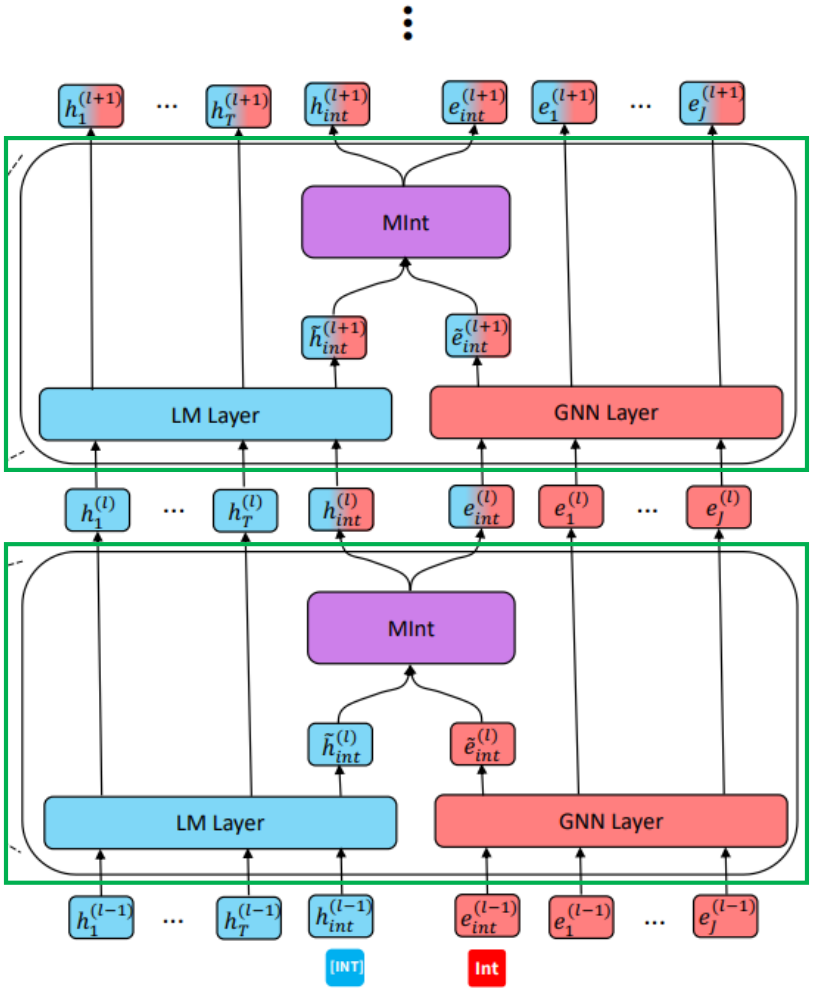
  - ✓ A modality interaction layer : two modalities fuse info

  (1) Concatenate pre-fused embeddings of the interaction token and interaction node

  (2) Pass joint representation through a mixing operation (MInt)

  (3) Split output post-fused embeddings into interaction token and interaction node

$$[\boldsymbol{h}_{int}^{(\ell)}; \boldsymbol{e}_{int}^{(\ell)}] = \text{MInt}([\tilde{\boldsymbol{h}}_{int}^{(\ell)}; \tilde{\boldsymbol{e}}_{int}^{(\ell)}])$$



GreaseLM Architecture

- Learning & Inference
  - ✓ Compute the probability of a being correct answer

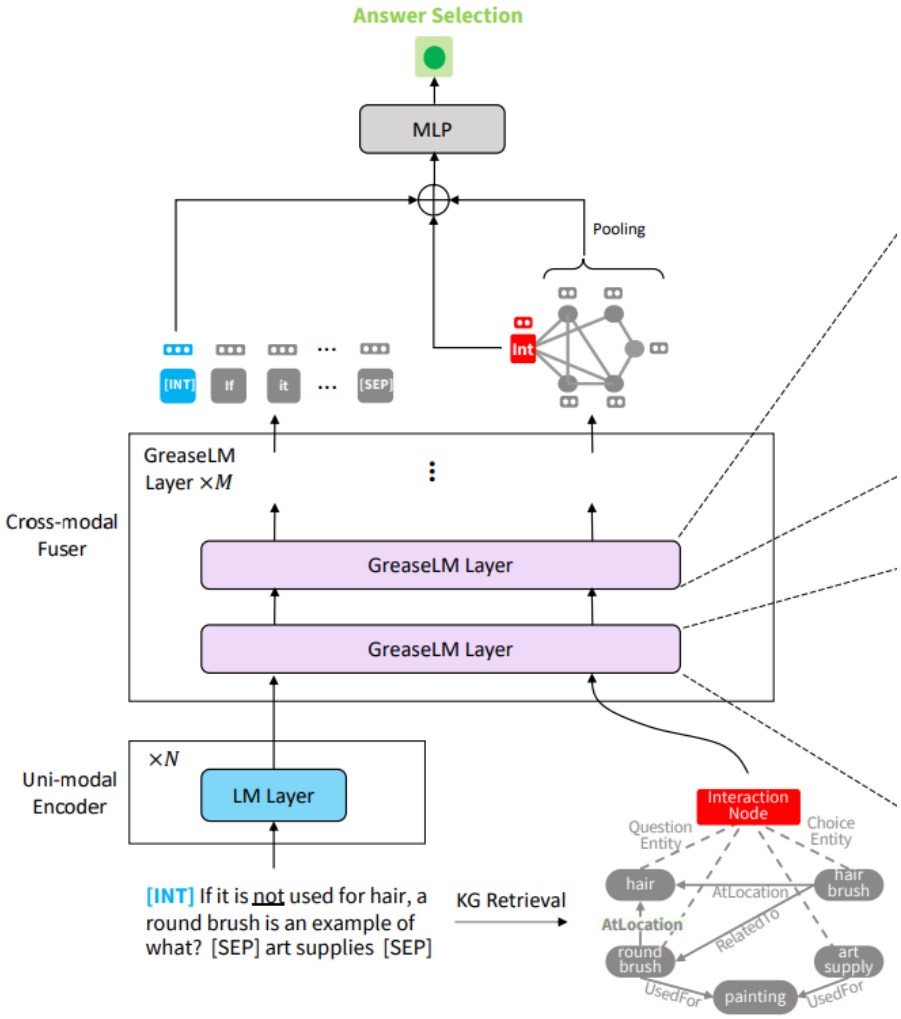
$$\exp(\text{MLP}(\boldsymbol{h}_{int}^{(N+M)},\ \boldsymbol{e}_{int}^{(M)},\ \boldsymbol{g}))$$

g : attention-based pooling of

$$\{\boldsymbol{e}_j^{(M)} \mid e_j \in \{e_1, \ldots, e_J\}\} \text{ using } \boldsymbol{h}_{int}^{(N+M)}$$

  - ✓ Loss function : Cross Entropy Loss
  - ✓ At inference :
    predict the most plausible answer as

$$\arg\max_{a \in \mathcal{A}}\ p(a \mid q, c).$$



GreaseLM

- Setup
  - ✓ CommonsenseQA, OpenBookQA, MedQA-USMLE
  - ✓ Knowledge graph : ConceptNet and self-constructed KG for MedQA-USMLE
  - ✓ LM : CommonsenseQA(RoBERTa-Large), OpenBookQA(AristoRoBERTa), MedQA-USMLE(SapBERT)

THANK YOU