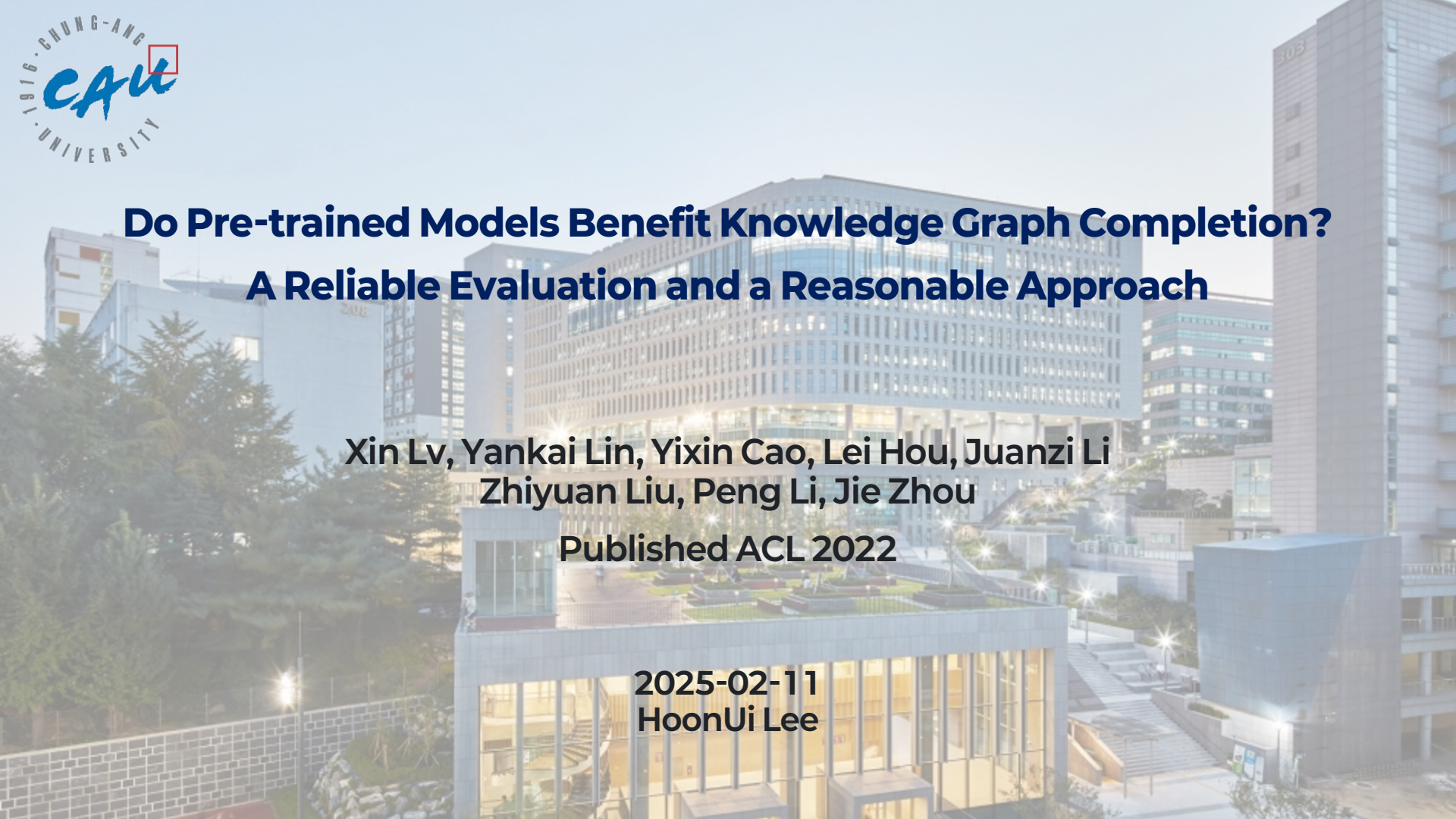


Do Pre-trained Models Benefit Knowledge Graph Completion? A Reliable Evaluation and a Reasonable Approach

Xin Lv, Yankai Lin, Yixin Cao, Lei Hou, Juanzi Li
Zhiyuan Liu, Peng Li, Jie Zhou

Published ACL 2022

2025-02-11
HoonUi Lee

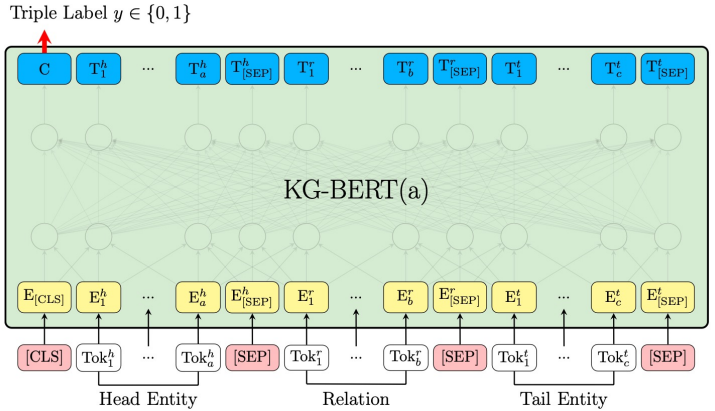
Contents

Previous Work

- KG-BERT

❖ Introduced PLM into KGC

- ❑ Make input by splicing entities and relation
  - with their description
- ❑ Score calculation using the [CLS] token's vector
  - [CLS] is used as the aggregate sequence representation
- ❑ Performance lagging behind embedding-based models



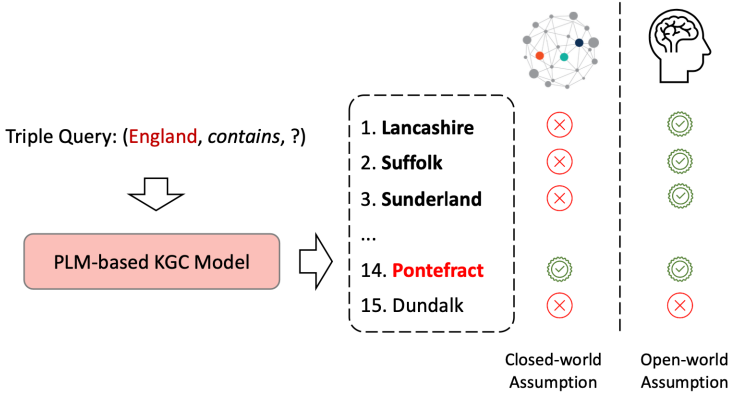
Model	Wiki27K				FB15K-237-N			
	MRR	@1	@3	@10	MRR	@1	@3	@10
TuckER (Balažević et al., 2019)	24.6	18.3	26.5	38.2	31.2	22.8	**34.6**	48.6
RotatE (Sun et al., 2019)	21.6	12.3	25.6	39.4	27.9	17.7	32.0	48.1
KG-BERT (Yao et al., 2019)	19.2	11.9	21.9	35.2	20.3	13.9	20.1	40.3

Previous Work

- 2 main reasons

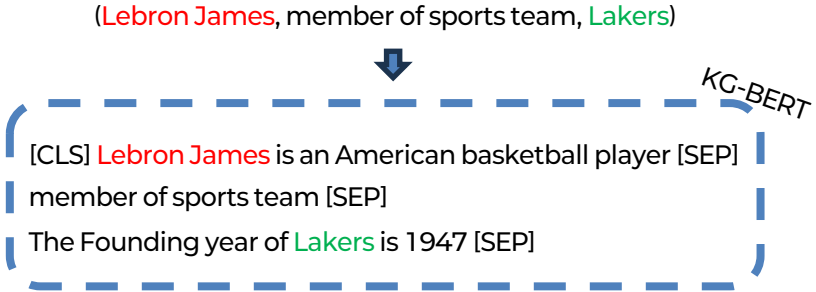
❖ Inaccurate evaluation setting

- ❑ closed-world assumption VS open-world assumption
  - ‣ PLM brings in much **unseen knowledge**
  - ‣ but it would be **wrong** in CWA



❖ Inappropriate utilization of PLMs

- ❑ simply splice the labels of the entities and relations
  - ‣ incoherent sentences
  - ‣ make gaps with the pre-trained task



Proposed Method

- **New evaluation setting**
  - Based on Open-World Assumption (OWA)

- **Convert each triple into natural prompt sentences**
  - Manually define the prompt template for each relation type
  - Insert support prompt at the end of the triple prompt
    - definition and attributes

Type	Template
Definition	"[***Entity***]: [***Definition Text***]."
Attribute	"The [***Attribute***] of [***Entity***] is [***Value***]."

PKGC

- Framework

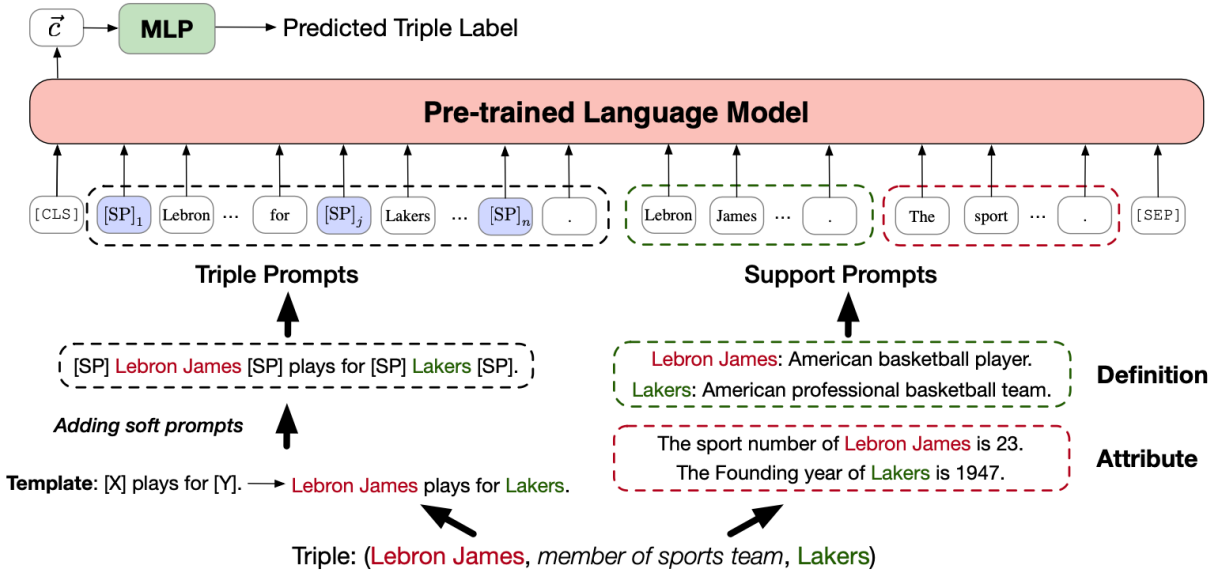


Figure 2: Illustration of our PKGC model for triple classification. Triples are transformed into triple prompts (left part) and support prompts (right part) to do the classification using PLMs.

PKGC

- Framework

❖ Novel PLM-based KGC model

- ❑ transforms given triple into two prompts
  - ▸ triple prompts P^T
  - ▸ support prompts P^S
- ❑ The final input texts T to the PLM can be defined as T
  - ▸ [CLS] P^T P^S [SEP]
- ❑ [CLS] is used to predict the label of the given triple

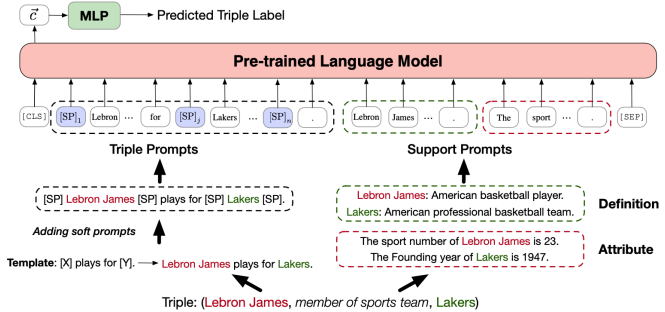


Figure 2: Illustration of our PKGC model for triple classification. Triples are transformed into triple prompts (left part) and support prompts (right part) to do the classification using PLMs.

PKGC

- Triple Prompts

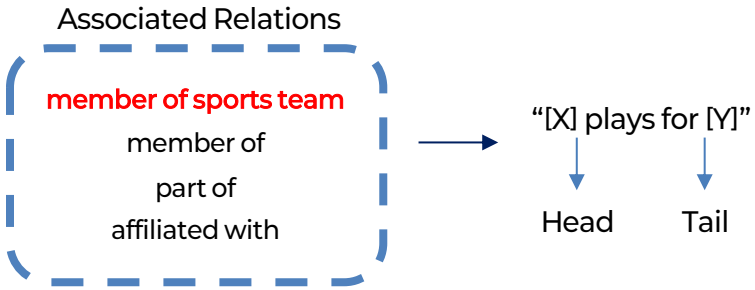
Template: [X] plays for [Y]. → Lebron James plays for Lakers.

⇧

Triple: (Lebron James, *member of sports team*, Lakers)

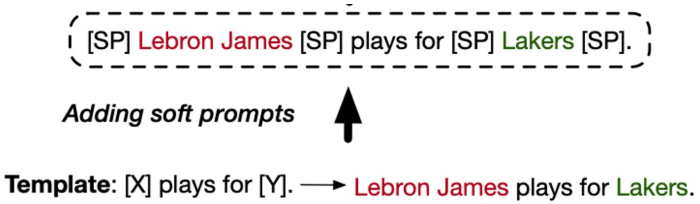
❖ Hard template for relation

- ❑ Human manually design a hard template for every relation $r \in R$
  - to represent the semantics of associated triples



PKGC

- Triple Prompts



❖ Add soft prompts

❑ vector lookup table $P \in \mathbb{R}^{|R| \times n \times d}$ for soft prompts

- n is the # of soft prompts contained in the triple prompts for one triple

❑ The template and entity label split the triple prompts into six positions

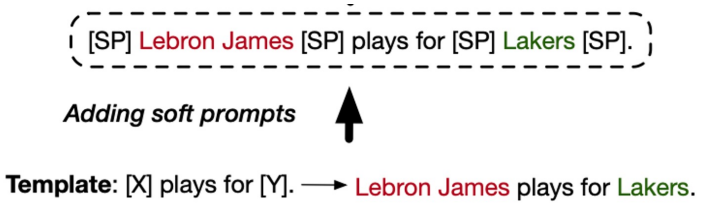
① ② ③ ④ ⑤ ⑥

__ **Template** __ ***Entity Label*** __ **Template** __ ***Entity Label*** __ **Template** __ .

- insert soft prompts in these positions
- the # of soft prompts at each position is $n_1, n_2, \cdots, n_6$, $n = \sum_{i=1}^{6} n_i$

PKGC

- Triple Prompts



❖ Using learnable vectors for relations

- ❑ k-th soft prompt $[SP]_k$ replaced with vector from P
  - a vector lookup table P

- ❑ $p_r^k = P_{[idx(r),k]} \in \mathbb{R}^{\boldsymbol{d}}$
  - $idx(r)$ is the ranking index of relation r
  - p_r^k is the k-th vector corresponding to the relation r in P

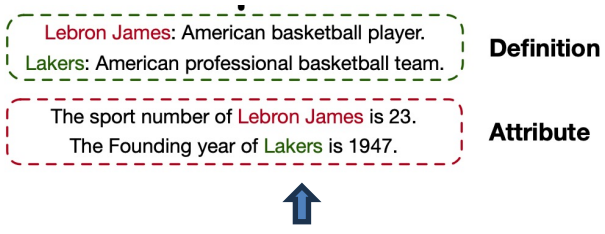
- ❑ Vector lookup table P will be updated in training
  - better represent the semantics for corresponding triple

PKGC

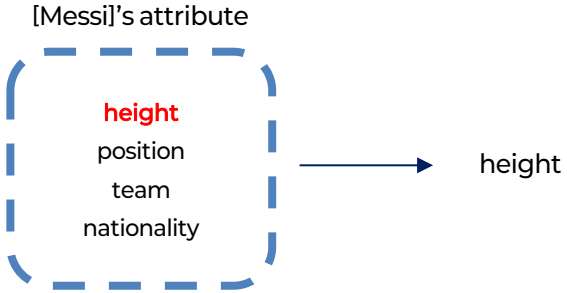
- Support Prompts

❖ Support information can help KGC

- ❑ Use **Definition** and **Attribute**
  - define templates to convert support information
- ❑ Use a random strategy to select attributes
  - to avoid too complex model
- ❑ Not require all support information to be present
  - only definition / only attribute
  - or another types of support information



Type	Template
Definition	"[***Entity***]: [***Definition Text***]."
Attribute	"The [***Attribute***] of [***Entity***] is [***Value***]."



PKGC

- Training

2 types of random triple

- Random negative triple $\mathcal{T}^-_{RAN}$
  - randomly replacing the head or tail entities
  - (h', r, t) / (h, r, t')
- KGE-based negative triple $\mathcal{T}^-_{KGE}$
  - also select (h', r, t) / (h, r, t')
  - but get high score in KGE model
  - difficult, hard negative for model

$$\frac{|\mathcal{T}^-_{\text{RAN}}|}{|\mathcal{T}^-_{\text{KGE}}|} = \frac{\alpha}{1-\alpha}$$

hyperparamter α

$$|\mathcal{T}| = K \cdot |\mathcal{T}^-|$$

hyperparamter K

Dataset	α	K	X	n	$n_1, n_2, n_3, n_4, n_5, n_6$
Wiki27K	0.5	30	30	6	1, 1, 1, 1, 1, 1
FB15K-237-N	0.5	30	30	6	1, 1, 1, 1, 1, 1
FB15K-237-NH	0.5	30	30	6	1, 1, 1, 1, 1, 1

Table 8: The best hyper-parameters on different datasets.

PKGC

- Training

❖ Classification scoring function

❑ $\mathbf{s}_\tau = \mathrm{Softmax}(\mathbf{W}\mathbf{c})$

- τ : given triple
- c : the output vector of [CLS], $\mathbf{c} \in \mathbb{R}^d$
- W : linear neural network, $\mathbf{W} \in \mathbb{R}^{2\times d}$

❖ Cross-entropy loss function

❑ $$\mathcal{L} = -\sum_{\tau \in \mathcal{T} \cup \mathcal{T}^-} (y_\tau \log(\mathbf{s}_\tau^1) + (1 - y_\tau)\frac{\log(\mathbf{s}_\tau^0)}{K})$$

- $y_\tau \in \{0, 1\}$, $\mathbf{s}_\tau^0, \mathbf{s}_\tau^1 \in [0, 1]$

Experiment

- **New metric for OWA**
  - **Hits@N, MRR**
    - for close-world assumption
  - **CR@1 (Correctness Rate @ 1)**
    - 1. sampling triple in test set
    - 2. fill the missing entity with the top-1 predicted entity
    - 3. human manually annotate the correct ratio of these triples
  - Useful for link prediction
    - not necessary for triple classification
    - 5% of false negative triple on average, while more than 30% for Hits@1 in link prediction

Experiment

- Link prediction

❖ Result

	Model	Wiki27K					FB15K-237-N				
		MRR	@1	@3	@10	CR@1	MRR	@1	@3	@10	CR@1
KGE Models	TransE (Bordes et al., 2013)	15.5	3.2	22.8	37.8	16.0	25.5	15.2	30.1	45.9	42.0
	TransC (Lv et al., 2018)	17.5	12.4	21.5	33.9	20.0	23.3	12.9	29.8	39.5	44.0
	ConvE (Dettmers et al., 2018)	22.6	16.4	24.4	35.4	21.5	27.3	19.2	30.5	42.9	48.5
	WWV (Veira et al., 2019)	19.8	15.7	23.7	36.5	22.5	26.9	13.7	28.7	44.3	40.5
	TuckER (Balažević et al., 2019)	24.6	18.3	26.5	38.2	33.0	31.2	22.8	**34.6**	48.6	51.0
	RotatE (Sun et al., 2019)	21.6	12.3	25.6	39.4	30.5	27.9	17.7	32.0	48.1	53.0
PLM-based	KG-BERT (Yao et al., 2019)	19.2	11.9	21.9	35.2	35.5	20.3	13.9	20.1	40.3	47.5
	LP-RP-RR (Kim et al., 2020)	21.7	13.8	23.5	37.9	38.0	24.8	15.5	25.6	43.6	52.5
	PKGC	25.2	18.9	28.5	39.0	44.0	30.7	23.2	32.8	47.1	58.5
	PKGC w/ attribute	25.5	19.1	28.8	39.4	44.0	31.1	23.5	32.9	47.7	58.5
	PKGC w/ definition	**28.5**	**23.0**	**30.5**	**40.9**	**47.5**	**33.2**	**26.1**	**34.6**	**48.7**	**62.5**

Table 3: Link prediction results on two datasets. @X denotes Hits@X. CR@1 is the evaluation metric for OWA in Section 5.1. All metrics are multiplied by 100. The best score is in **bold**.

- ❑ Compare between CWA metrics and OWA metric (CR@1)
- ❑ KGE performs particularly poorly on Wiki27K
  - Wiki27K based on definition and attribute

Experiment

- Triple Classification

❖ Result

harder N-sample

	Model	Wiki27K		FB15K-237-N		FB15K-237-NH	
		Acc.	F1	Acc.	F1	Acc.	F1
KGE Models	TransE (Bordes et al., 2013)	65.5/64.2	72.3/71.5	66.2/64.0	71.1/70.4	50.3/49.5	66.2/62.1
	TransC (Lv et al., 2018)	68.7/68.4	71.5/71.2	66.4/64.6	71.3/70.8	51.2/50.4	67.7/64.0
	ConvE (Dettmers et al., 2018)	70.7/68.8	73.5/73.5	67.3/67.3	71.8/73.7	54.6/55.3	67.3/67.1
	WWV (Veira et al., 2019)	69.9/68.0	72.8/72.5	65.2/65.7	70.8/70.1	50.5/49.6	66.8/62.1
	TuckER (Balažević et al., 2019)	70.0/69.5	73.1/73.8	68.3/71.0	71.9/74.3	54.3/55.4	67.4/67.3
	RotatE (Sun et al., 2019)	72.3/64.0	75.1/71.3	67.9/63.2	72.3/69.9	51.7/51.9	66.8/64.8
PLM-based	KG-BERT (Yao et al., 2019)	83.7/82.4	84.3/83.1	71.8/72.7	72.8/73.6	56.4/57.6	63.3/63.6
	LP-RP-RR (Kim et al., 2020)	84.3/83.6	85.1/84.4	73.8/74.4	73.0/74.5	58.3/59.1	65.1/65.7
	PKGC	87.0/87.8	87.1/88.0	79.6/81.4	79.5/81.2	63.8/64.8	68.7/68.7
	PKGC w/ attribute	87.6/87.8	87.5/87.9	79.5/81.2	79.5/81.4	64.1/65.0	68.7/69.6
	PKGC w/ definition	**90.0/90.0**	**90.1/90.2**	**82.5/84.4**	**83.0/84.7**	**65.7/66.9**	**70.5/71.3**

Table 4: Triple classification results on three datasets. The values before and after the slash are the results under CWA and OWA, respectively. All metrics are multiplied by 100. The best score is in **bold**.

- ❑ Does not have a significant performance advantage under CWA
  - ▸ by using definite negative sample
- ❑ Best performance with definition
  - ▸ relatively lower performance with attribute in random choice

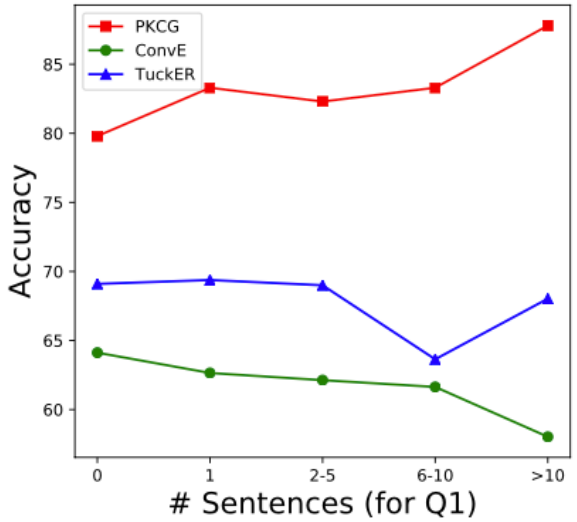
Experiment

- Analysis

❖ **Q1**

PLMs have seen many facts in the massive texts.

Is it because they remember these facts to help our model achieve better results?



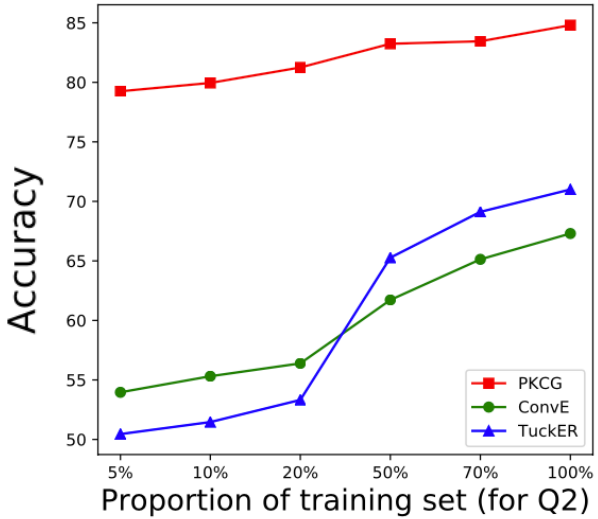
- Classification of triples that frequently appear in Wikipedia and those that do not
  - if h and t appear in sentence, consider it to imply the fact of (h, r, t)

- Inference performance improves when learning triple that frequently seen more in massive texts

Experiment

- Analysis

❖ **Q2** Can the introduced PLMs make our model less sensitive to the amount of training data?

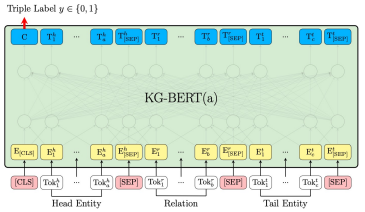


- PLM-based PKGC is insensitive to the amount of training data
  - PLM is pre-trained on a large amount of textual data
- While KGE-based model's performance decreases significantly

Conclusion

Previous work

- The existing method has the problem that the input sentences are incoherent
- A metric that does not consider OWA (Open World Assumption)



PKGC

- Design a hard template to properly transform triples into an appropriate format
- Add soft prompts and support prompts to input

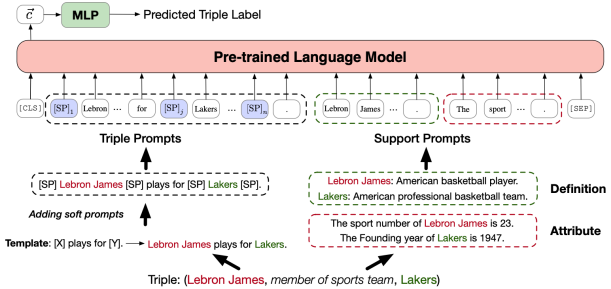


Figure 2: Illustration of our PKGC model for triple classification. Triples are transformed into triple prompts (left part) and support prompts (right part) to do the classification using PLMs.

Experiment

- Difference between the existing metric and CR@1
- Best performance using definition

Analysis

- Inference is performed using triples that frequently appear in the corpus
- Not sensitive to the number of training data