K-BERT: Enabling Language Representation with Knowledge Graph

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang.

Outline

- Motivation
- Challenges
- Proposed Method
- Experiments
- Results Analysis
- Questions

Motivation

- Pre-trained language representation models, such as BERT, capture a general language representation from large-scale corpora, but lack domain-specific knowledge.
- Propose a **knowledge-enabled language representation model (K-BERT)** with knowledge graphs (KGs). It makes the model as domain expert.

Challenges

- Heterogeneous Embedding Space (HES)

  the embedding vectors of words in text and entities in KG are obtained in separate ways, making their vector-space inconsistent

- Knowledge Noise (KN):

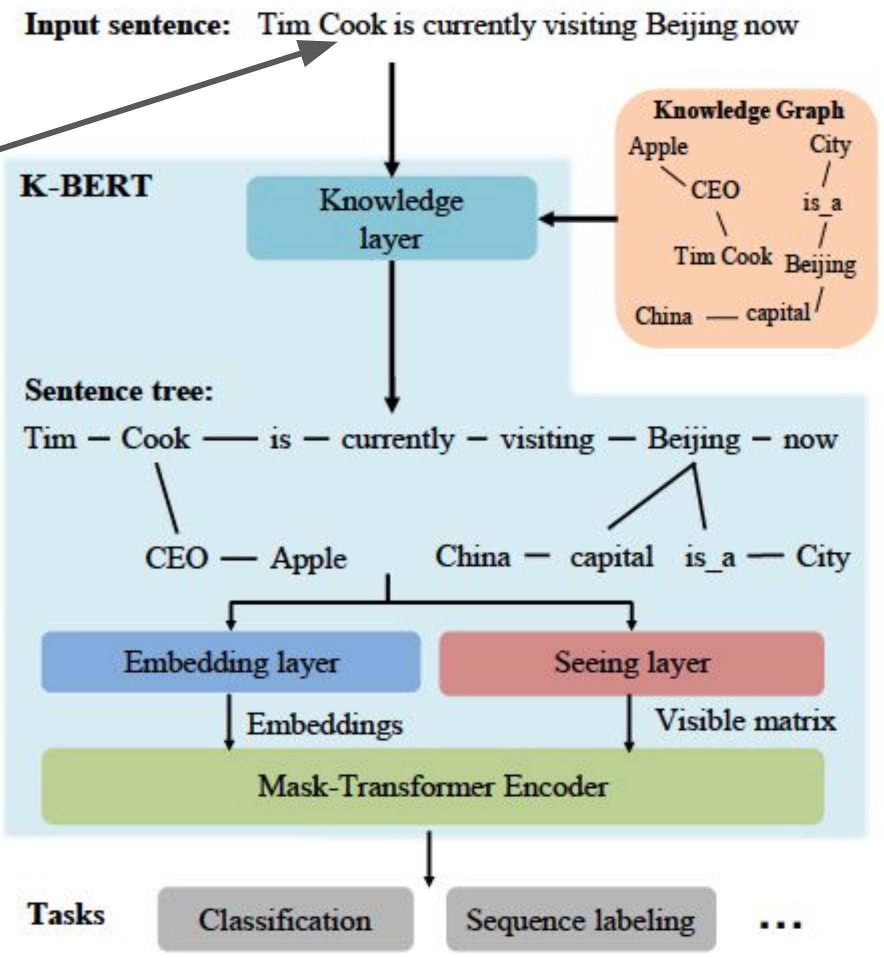
  Too much knowledge incorporation may divert the sentence from its correct meaning.

Method: Overview

Notation:

$$s = \{w_0, w_1, w_2, ..., w_n\}$$

vocabulary $\mathbb{V}$ $w_i \in \mathbb{V}$



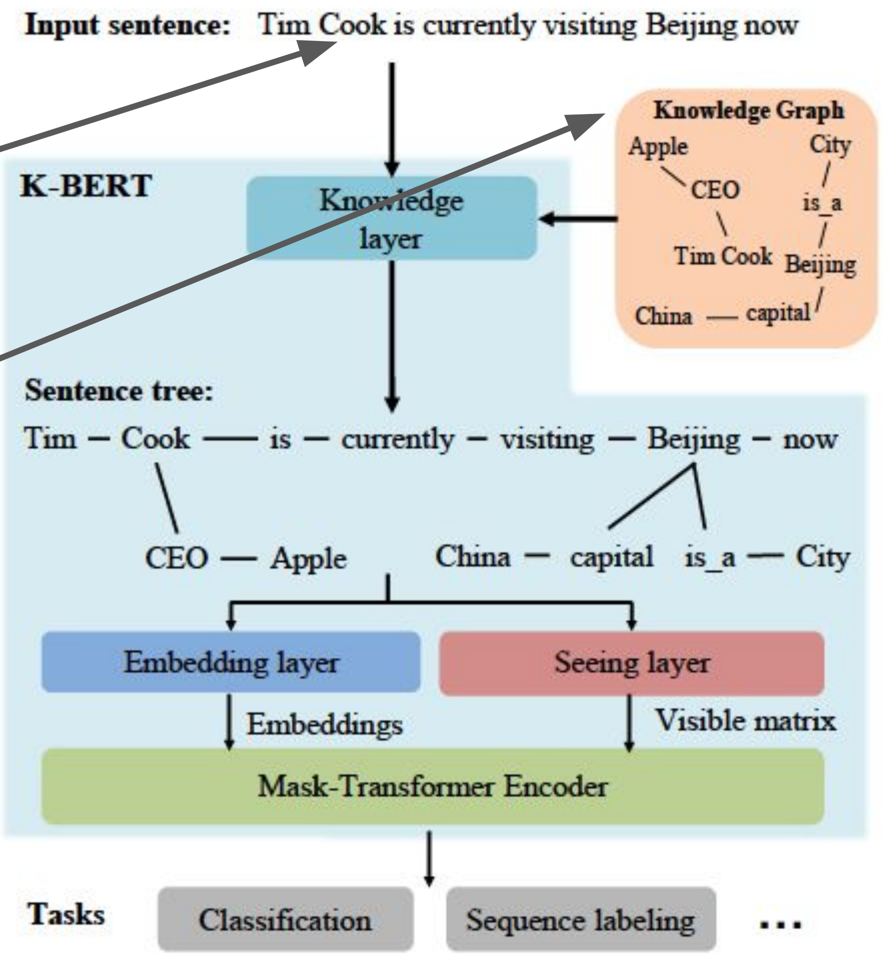
Method: Overview

Notation:

$s = \{w_0, w_1, w_2, ..., w_n\}$

vocabulary $\mathbb{V}$ $w_i \in \mathbb{V}$

KG, denoted as $\mathbb{K}$,



Method: Overview

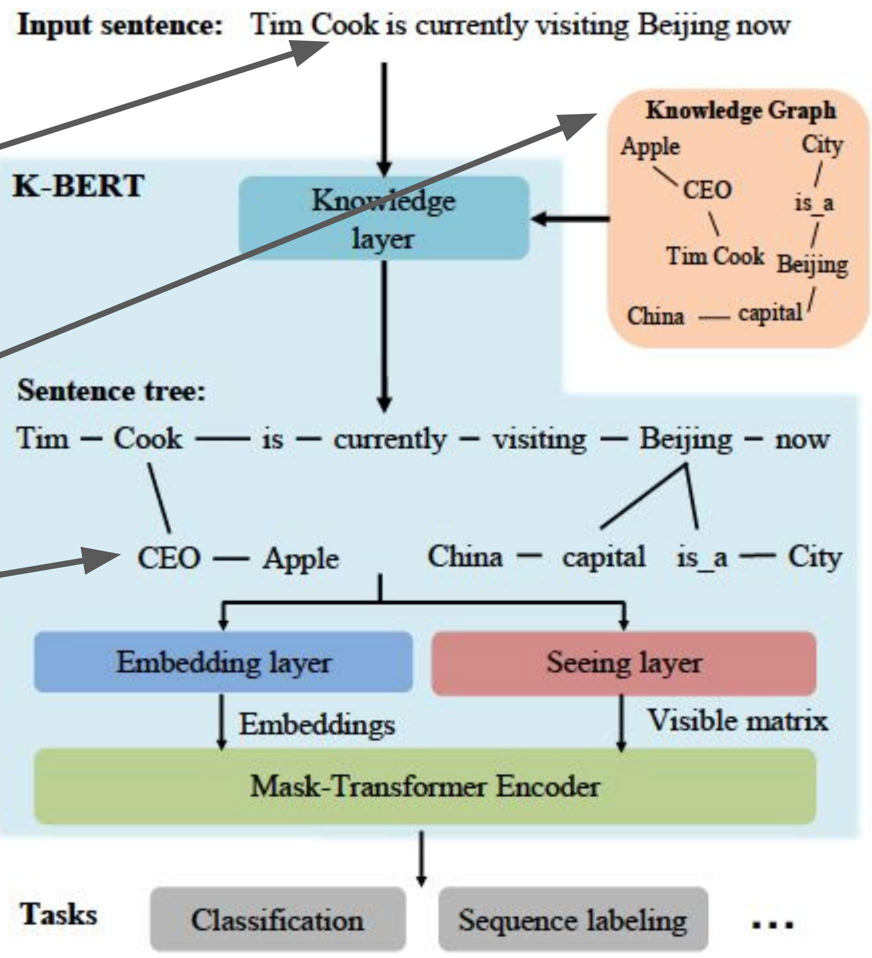
Notation:

$$s = \{w_0, w_1, w_2, ..., w_n\}$$

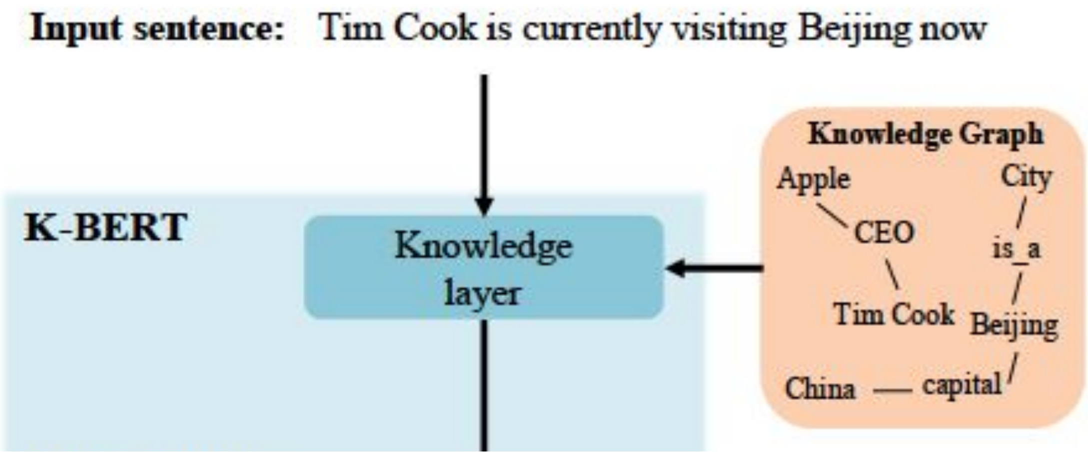
vocabulary $\mathbb{V}$ $w_i \in \mathbb{V}$

KG, denoted as $\mathbb{K}$,

a collection of triples $\varepsilon = (w_i, r_j, w_k)$, where w_i and w_k are the name of entities, and $r_j \in \mathbb{V}$ is the relation between them. All the triples are in KG, i.e., $\varepsilon \in \mathbb{K}$.



Knowledge layer



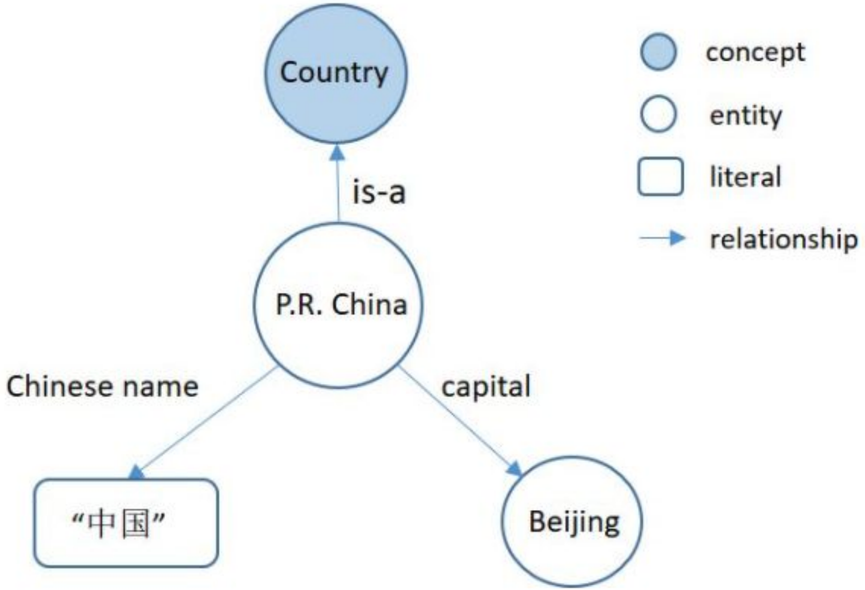
Knowledge layer

The knowledge layer (KL) is used for sentence knowledge injection and sentence tree conversion. Specifically, given an input sentence $s = \{w_0, w_1, w_2, ..., w_n\}$ and a KG $\mathbb{K}$, KL outputs a sentence tree $t =$ $\boxed{\{w_0, w_1, ..., w_i\{(r_{i0}, w_{i0}), ..., (r_{ik}, w_{ik})\}, ..., w_n\}}$.

1. Knowledge query (K-Query)
2. Knowledge injection (K-Inject).

KG is Directed, out-going neighbours.

Example from CN-BDpedia



https://www.mdpi.com/2071-1050/10/9/3245

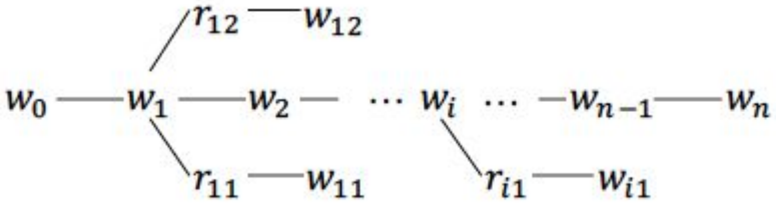
Knowledge layer

- Knowledge query (K-Query)

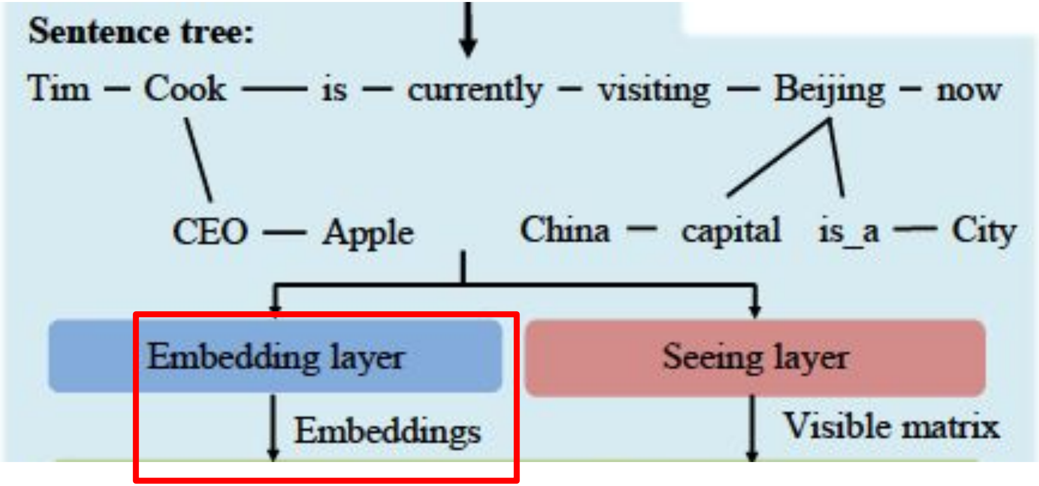
$$E = K\_Query(s, \mathbb{K}), \qquad E = \{(w_i, r_{i0}, w_{i0}), ..., (w_i, r_{ik}, w_{ik})\}$$

- Knowledge injection (K-Inject): a sentence tree can have multiple branches, but its depth is fixed to 1.

$$t = K\_Inject(s, E).$$



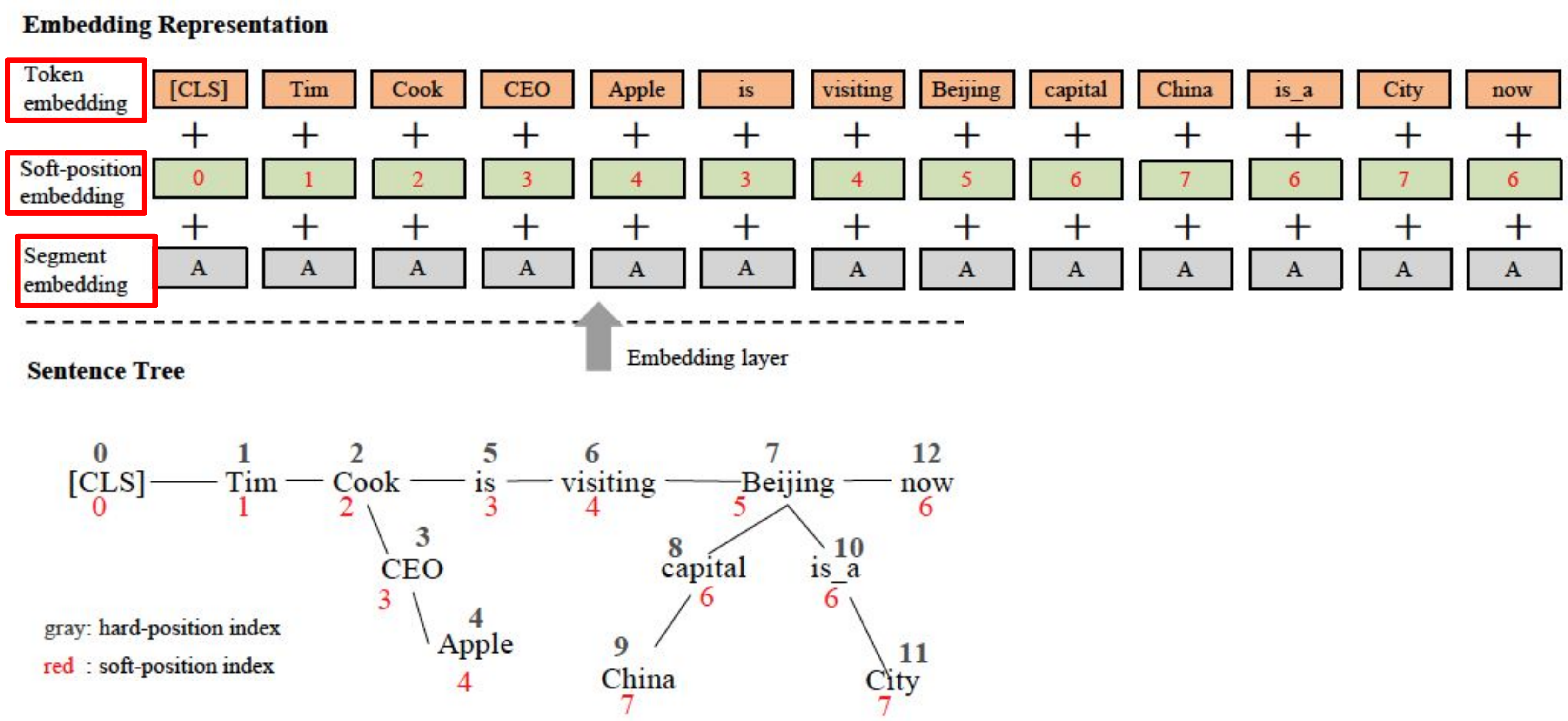
Embedding layer



Embedding layer

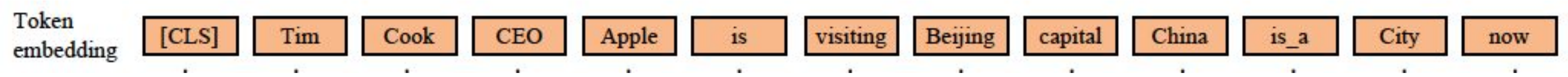
The function of the Embedding Layer (EL) is to convert the sentence tree into an embedding representation that can be fed into the Mask-Transformer.

1. Token embedding
2. Soft-position embedding
3. Segment embedding


Embedding Representation
Token embedding
[CLS]
Tim
Cook
CEO
Apple
is
visiting
Beijing
capital
China
is_a
City
now
Soft-position embedding
0
1
2
3
4
3
4
5
6
7
6
7
6
Segment embedding
A
Embedding layer
Sentence Tree
0 [CLS] 0
1 Tim 1
2 Cook 2
5 is 3
6 visiting 4
7 Beijing 5
12 now 6
3 CEO 3
4 Apple 4
8 capital 6
9 China 7
10 is_a 6
11 City 7
gray: hard-position index
red : soft-position index

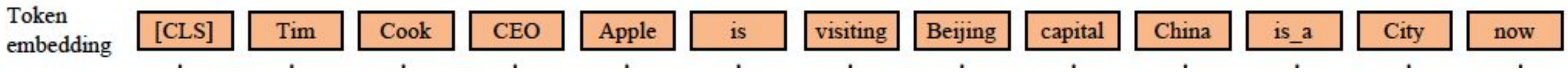
Embedding layer

Token Embedding: the tokens in the sentence tree are flattened into a sequence of token embedding by their hard-position index

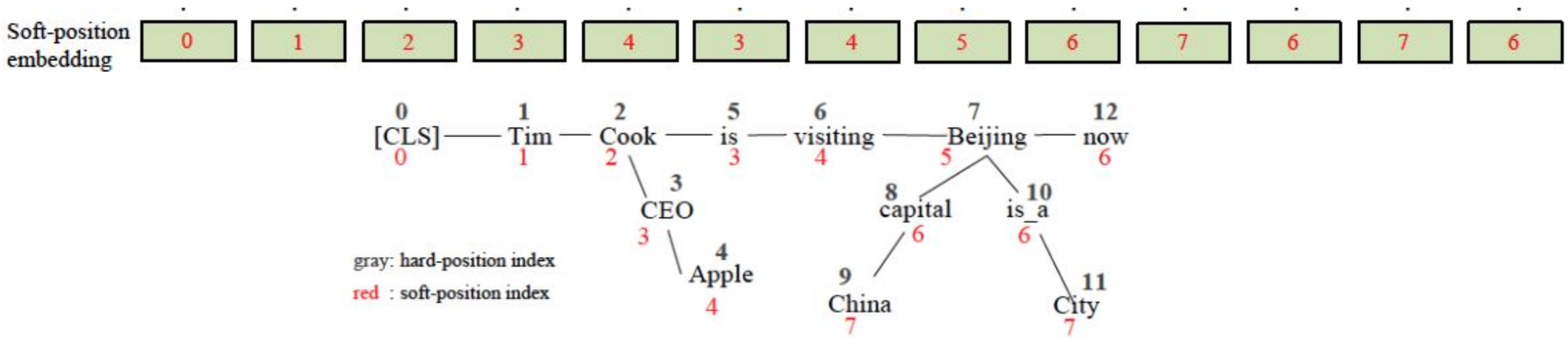


Embedding layer

Token Embedding: the tokens in the sentence tree are flattened into a sequence of token embedding by their hard-position index.

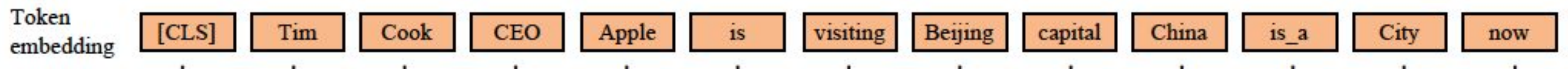


Soft-position embedding: Make the sentence reable and keep the correct structural information.

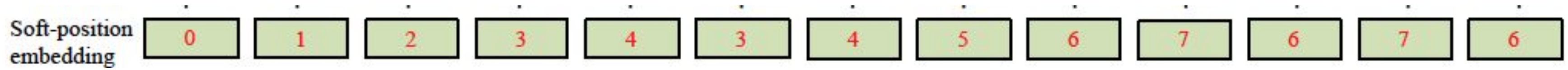


Embedding layer

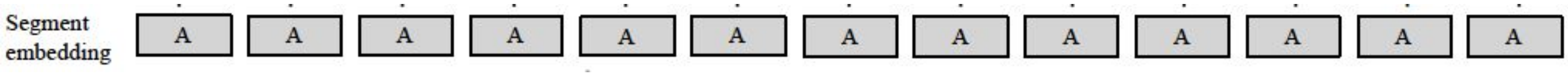
Token Embedding: the tokens in the sentence tree are flattened into a sequence of token embedding by their hard-position index.



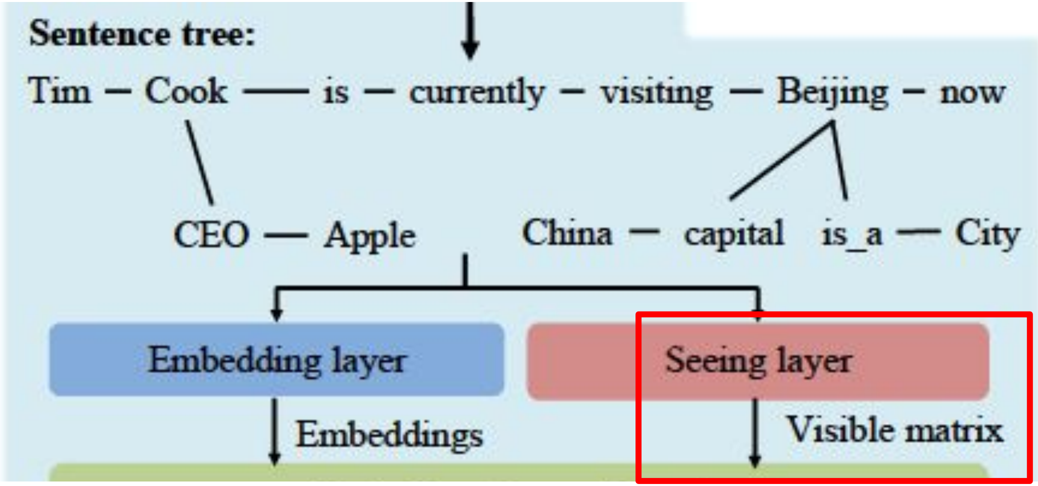
Soft-position embedding: Make the sentence reable and keep the correct structural information.



Segment embedding: to identify different sentences when multiple sentences are included.

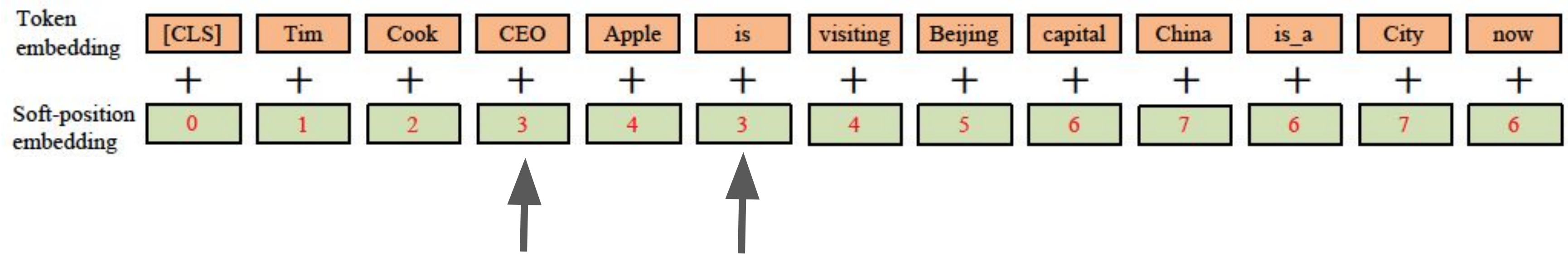


Seeing layer



Seeing layer

Problem: Have same soft-position index, but there is no connection between them.



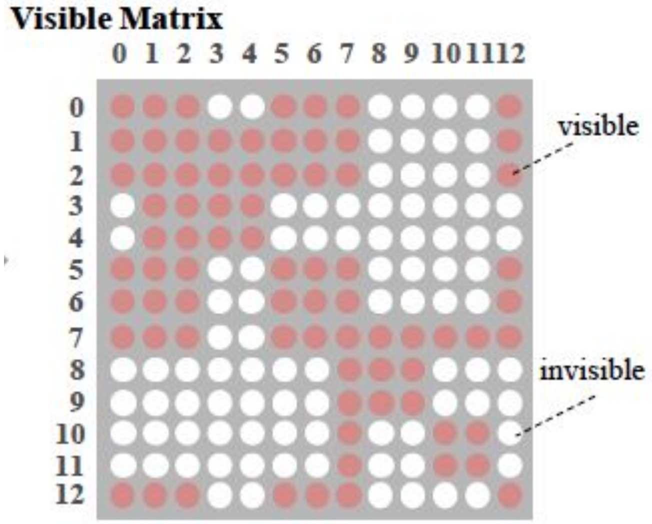
Seeing layer

Problem: Knowledge Noise (KN): Too much knowledge incorporation may divert the sentence from its correct meaning.

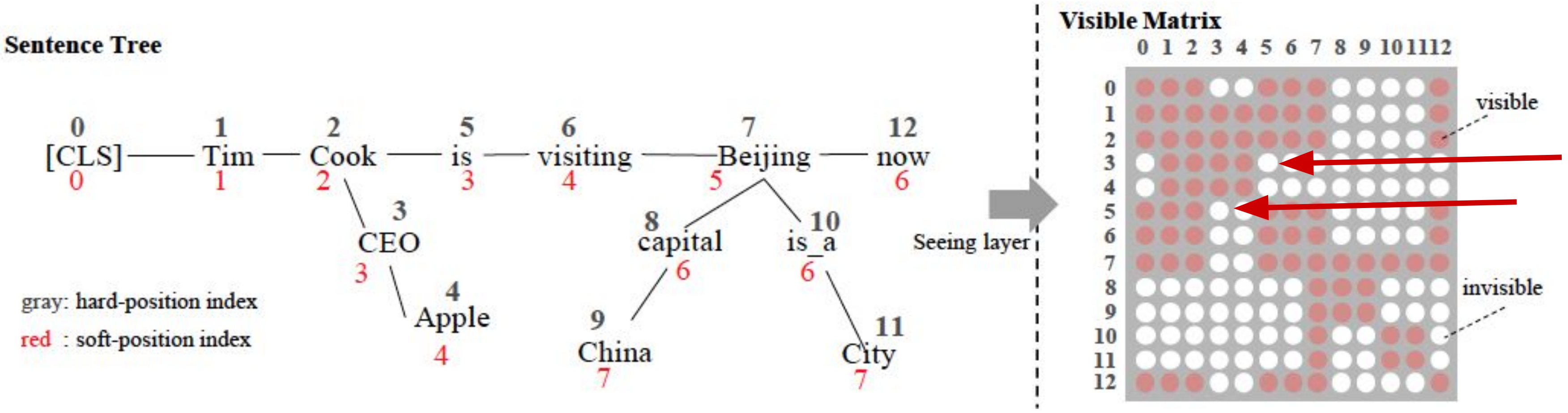
Solution: Visible matrix ***M*:**

$$M_{ij} = \begin{cases} 0 & w_i \ominus w_j \\ -\infty & w_i \oslash w_j \end{cases}$$

where, $w_i \ominus w_j$ indicates that w_i and w_j are in the same branch, while $w_i \oslash w_j$ are not. i and j are the hard-position index.

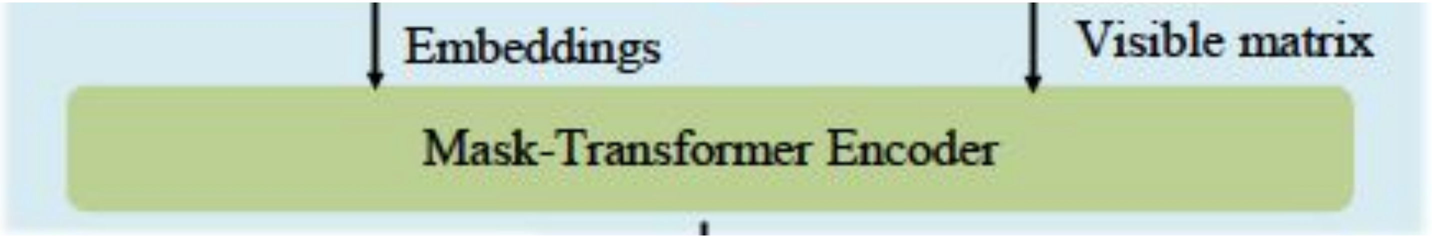


Seeing layer



“CEO” and “is” cannot see each other.

Mask-Transformer



Mask-Transformer

Modify BERT to Mask-Transformer which can limit the self-attention region according to M.

$$Q^{i+1}, K^{i+1}, V^{i+1} = h^i W_q, h^i W_k, h^i W_v,$$

$$S^{i+1} = softmax(\frac{Q^{i+1} {K^{i+1}}^\top + M}{\sqrt{d_k}}),$$

$$h^{i+1} = S^{i+1} V^{i+1},$$

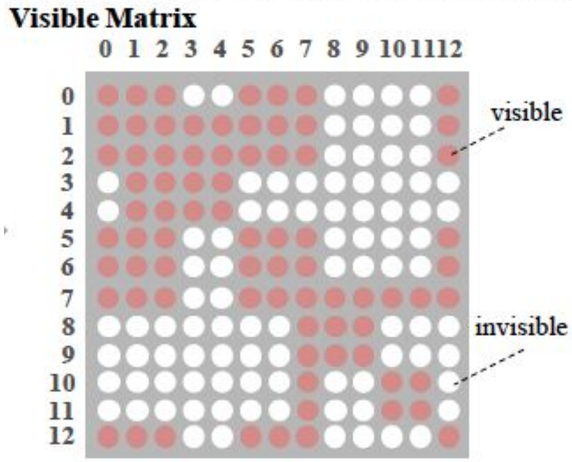
$$M_{ij} = \begin{cases} 0 & w_i \ominus w_j \\ -\infty & w_i \oslash w_j \end{cases}$$

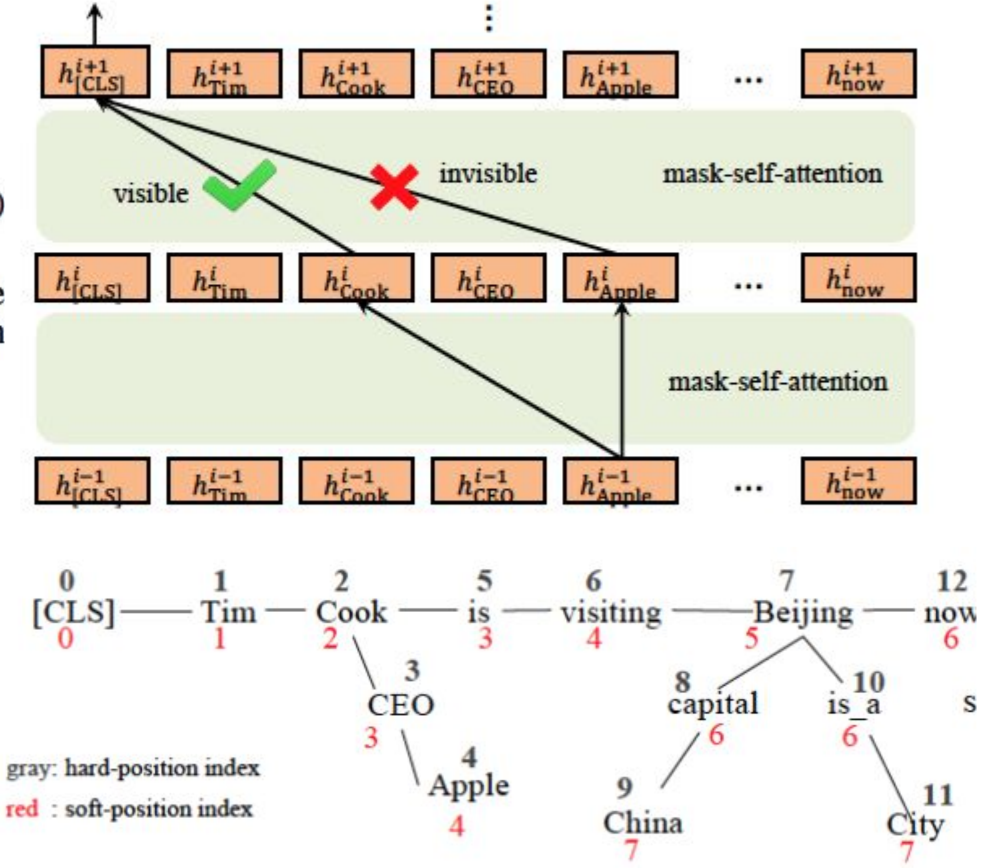
where W_q, W_k and W_v are trainable model parameters. h^i is the hidden state of the i-th mask-self-attention blocks. d_k is the scaling factor[1]. M is the visible matrix calculated by the seeing layer. Intuitively, if w_k is invisible to w_j, the M_{jk} will mask the attention score S_{jk}^{i+1} to 0, which means w_k make no contribution to the hidden state of w_j.

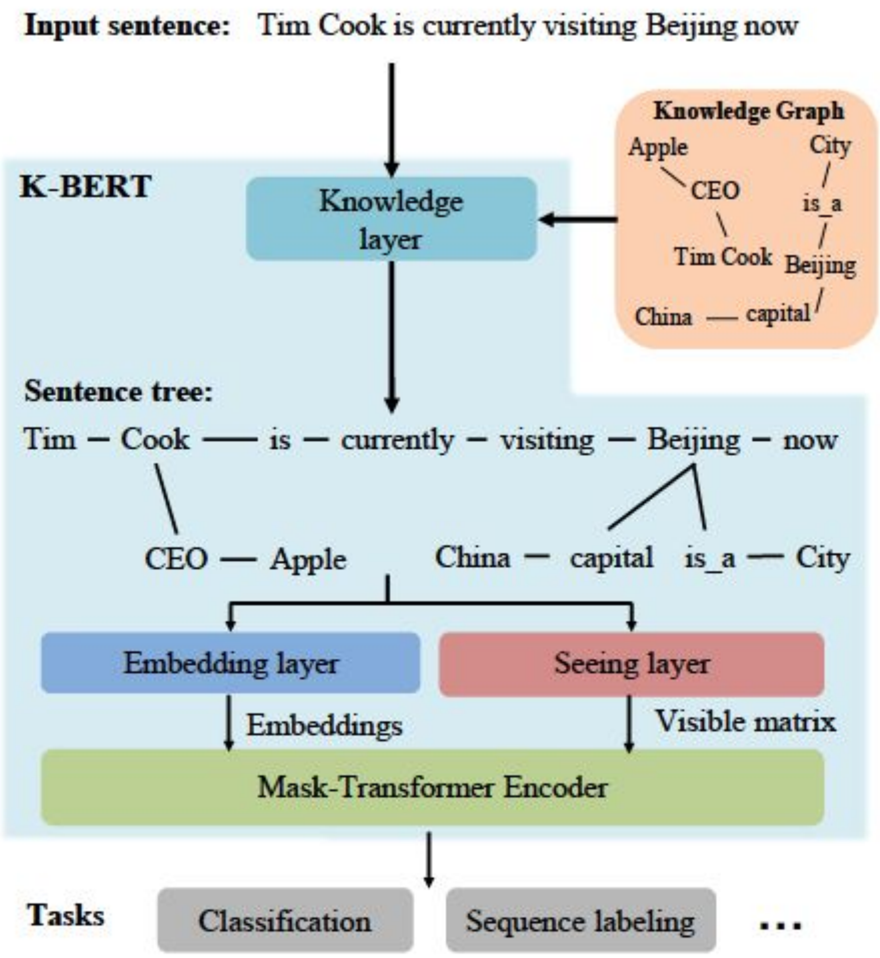
Mask-Transformer

$$M_{ij} = \begin{cases} 0 & w_i \ominus w_j \\ -\infty & w_i \oslash w_j \end{cases} \quad (3)$$

where, $w_i \ominus w_j$ indicates that w_i and w_j are in the same branch, while $w_i \oslash w_j$ are not. i and j are the hard-position index.






Input sentence: Tim Cook is currently visiting Beijing now
K-BERT
Knowledge layer
Knowledge Graph
Apple
CEO
Tim Cook
City
is_a
Beijing
China
capital
Sentence tree:
Tim — Cook — is — currently — visiting — Beijing — now
CEO — Apple
China — capital
is_a — City
Embedding layer
Seeing layer
Embeddings
Visible matrix
Mask-Transformer Encoder
Tasks
Classification
Sequence labeling
...

Experiment

- Pre-training corpora:

  WikiZh and WebtextZh

- Knowledge graph:

  CN-DBpedia, HowNet and MedicalKG

- Baseline:
  - Google BERT (I think it should be mBERT): pretraining on WikiZh.
  - Our BERT: pretraining on WikiZh and WebtextZh.
  - Architecture: L = 12, A = 12 and H = 768.

Fine-tuning and evaluation

- Open-domain tasks
  - Book review: positive v.s. Negative
  - Chnsenticorp hotel review: positive v.s. Negative
  - Shopping review: positive v.s. Negative
  - Weibo: positive v.s. Negative
  - XNLI: Cross-lingual Natural Language Inference corpus
  - LCQMC: Chinese question matching corpus.
  - NLPCC-DBQA: a task to predict answers to each question from the given document;
  - MSRA-NER: recognize the entity names in the text, including person names, place names, organization names, etc.

Results

Table 1: Results of various models on sentence classification tasks on open-domain tasks (*Acc. %*)

Models\Datasets	Book_review		Chnsenticorp		Shopping		Weibo		XNLI		LCQMC	
	Dev	*Test*	*Dev*	*Test*	*Dev*	*Test*	*Dev*	*Test*	*Dev*	*Test*	*Dev*	*Test*
Pre-trainied on WikiZh by Google.												
Google BERT	88.3	**87.5**	93.3	94.3	96.7	96.3	98.2	98.3	76.0	75.4	88.4	86.2
K-BERT (HowNet)	88.6	87.2	**94.6**	**95.6**	**97.1**	**97.0**	98.3	98.3	**76.8**	**76.1**	**88.9**	86.9
K-BERT (CN-DBpedia)	**88.6**	87.3	93.9	95.3	96.6	96.5	98.3	98.3	76.5	76.0	88.6	**87.0**
Pre-trained on WikiZh and WebtextZh by us.												
Our BERT	**88.6**	87.9	94.8	95.7	96.9	**97.1**	98.2	98.2	77.0	76.3	89.0	86.7
K-BERT (HowNet)	88.5	87.4	**95.4**	95.6	96.9	96.9	98.3	**98.4**	**77.2**	**77.0**	**89.2**	**87.1**
K-BERT (CN-DBpedia)	88.8	87.9	95.0	**95.8**	**97.1**	97.0	98.3	98.3	76.2	75.9	89.0	86.9

Results

Table 2: Results of various models on NLPCC-DBQA (MRR %) and MSRA-NER ($F1$ %).

Models\Datasets	NLPCC-DBQA		MSRA-NER	
	Dev	$Test$	Dev	$Test$
Pre-trained on WikiZh by Google.				
Google BERT	93.4	93.3	94.5	93.6
K-BERT (HowNet)	93.2	93.1	95.8	94.5
K-BERT (CN-DBpedia)	**94.5**	**94.3**	**96.6**	**95.7**
Pre-trained on WikiZh and WebtextZh by us.				
Our BERT	93.3	93.6	95.7	94.6
K-BERT (HowNet)	93.2	93.1	96.3	95.6
K-BERT (CN-DBpedia)	**93.6**	**94.2**	**96.4**	**95.6**

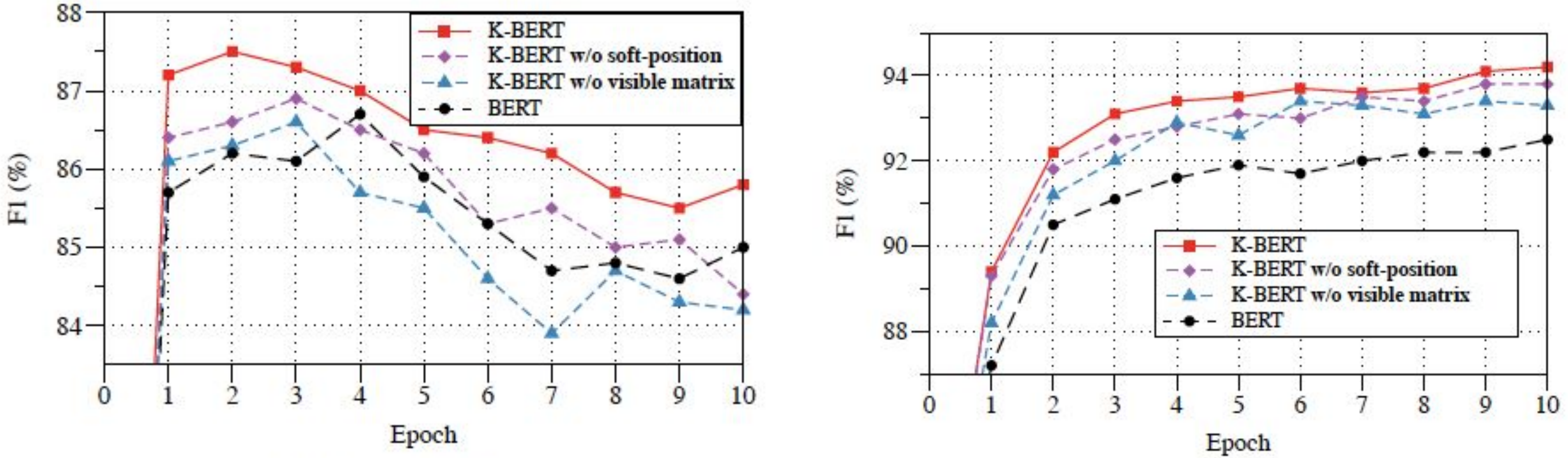
Fine-tuning and evaluation

- Specific-domain tasks
  - Domain Q&A: Finance Q&A and Law Q&A
  - Domain NER: Finance NER
  - Medicine NER

Table 3: Results of various models on specific-domain tasks (%).

Models\Datasets	Finance_Q&A			Law_Q&A			Finance_NER			Medicine_NER		
	$P.$	$R.$	$F1$	$P.$	$R.$	$F1$	$P.$	$R.$	$F1$	$P.$	$R.$	$F1$
Pre-trained on WikiZh by Google.												
Google BERT	81.9	86.0	83.9	83.1	90.1	86.4	84.8	87.4	86.1	91.9	93.1	92.5
K-BERT (HowNet)	83.3	84.4	83.9	83.7	91.2	87.3	86.3	89.0	**87.6**	93.2	93.3	93.3
K-BERT (CN-DBpedia)	81.5	88.6	**84.9**	82.1	93.8	**87.5**	86.1	88.7	87.4	93.9	93.8	93.8
K-BERT (MedicalKG)	-	-	-	-	-	-	-	-	-	94.0	94.4	**94.2**
Pre-trained on WikiZh and WebtextZh by us.												
Our BERT	82.1	86.5	84.2	83.2	91.7	87.2	84.9	87.4	86.1	91.8	93.5	92.7
K-BERT (HowNet)	82.8	85.8	84.3	83.0	92.4	87.5	86.3	88.5	87.3	93.5	93.8	93.7
K-BERT (CN-DBpedia)	81.9	87.1	**84.4**	83.1	92.6	**87.6**	86.3	88.6	**87.4**	93.9	94.3	94.1
K-BERT (MedicalKG)	-	-	-	-	-	-	-	-	-	94.1	94.3	**94.2**

Ablation studies



(a) Law_Q&A

(b) Medicine_NER

The soft-position and the visible matrix can make K-BERT more robust to KN interference and thus make more efficient use of knowledge.

Thanks

Question?

Tokenization of Chinese is a problem. Chinese text does not use white space to separate words. E.g.,:

诸如BERT之类的经过预训练的语言表示模型可以从大型语料库中获取通用的语言表示，但是缺少特定领域的知识。(First sentence of abstract)

The paper didn't provide precise details of their vocabulary. Google BERT uses byte-pair encoding vocabulary. Hence, the vocabulary is not purely character-level vocabulary. It is BPE.

E.g., vocabulary of mBERT

```
296
##nap
security
sunday
association
##ens
##700
##bra
```

```
##・
##･
##鰍
##一
##丁
##七
##万
##丈
##三
##上
```

https://huggingface.co/bert-base-multilingual-cased/blob/main/vocab.txt

More thinking

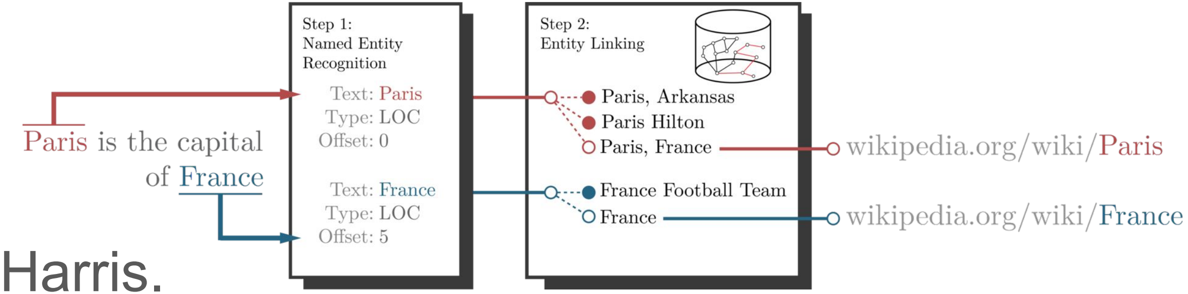
- The ambiguity of Chinese text segmentation.

我喜欢新西兰花	**Unsegmented Chinese sentence**
我 喜欢 新西兰 花	*I like New Zealand flowers*
我 喜欢 新 西兰花	*I like fresh broccoli*

- Disambiguate,Entity linking, and coreference:

Time Cook or cook,

Biden wins election 2020.
He is joining with Kamala Harris.



http://what-when-how.com/how-to-build-a-digital-library/word-segmentation-and-sorting-digital-library/
https://en.wikipedia.org/wiki/Entity_linking#/media/File:Entity_Linking_-_Example_of_pipeline.png