Enhanced Character Embedding for Chinese Short Text Entity Linking

Li Yang, Shijia E and Yang Xiang

li.yang@tongji.edu.cn

College of Electronics and Information Engineering, Tongji University

Introduction

Entity Linking (EL), which aims to recognize potentially ambiguous mentions of entities in a text and link them to a target knowledge base. It typically includes two subtasks: **Entity Recognition** (ER) and **Entity Disambiguation** (ED).

“比特币吸粉无数,但央行的心另有所属|界面新闻 • jmedia”

mention	offset	kb_id
比特币	0	278410
央行	9	199602
界面新闻	18	215472

Fig. 1 Problem Description

Entity linking has been studied for many years and has achieved great advancement with neural network. While most of the works are designed for English corpus, especially for long texts, the CCKS2019 Task 2 focuses on **Chinese short texts** instead. It is a more challenging task due to the lack of **explicit word delimiters** and **rich context**. To address these challenges, we propose an **enhanced character embedding** based neural approach, which explicitly encodes **mention dictionary** and **mention position** information into ER and ED model respectively.

Model Architecture

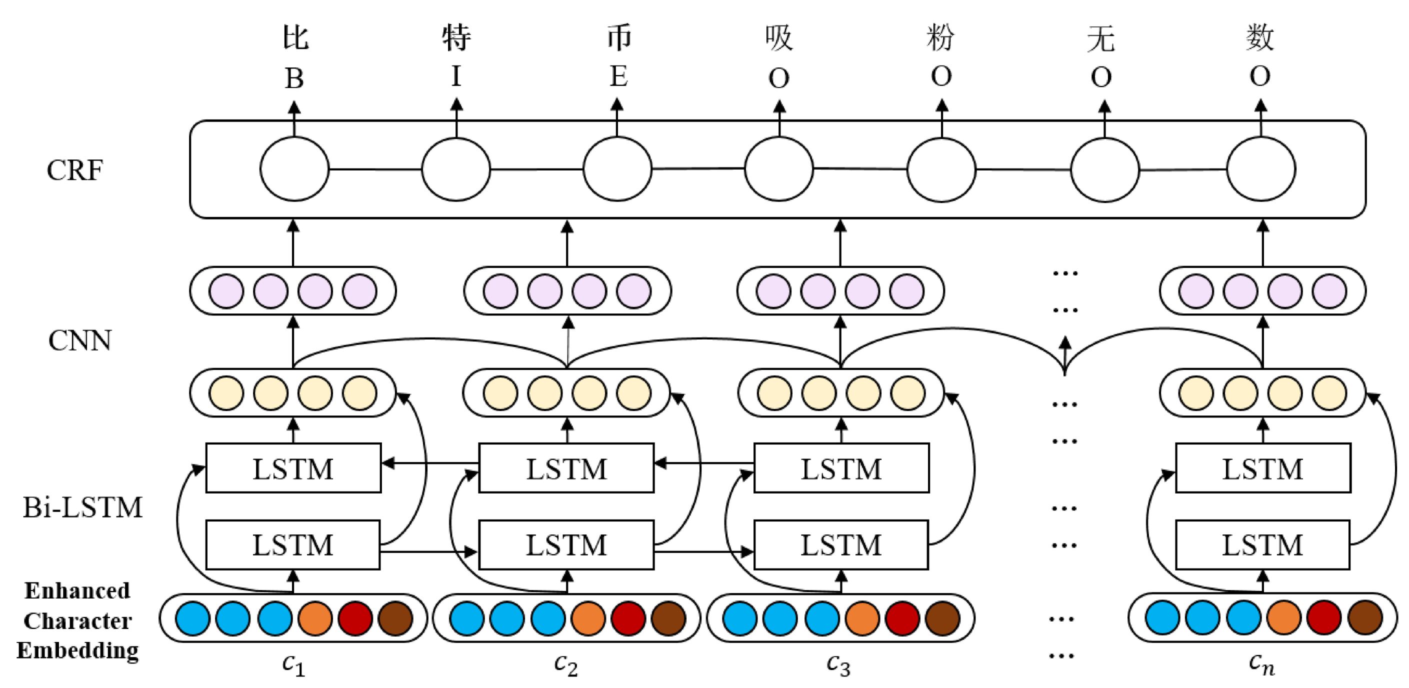

Fig. 2 Overall framework of ER model

Entity Recognition Model. We use **character-based BiLSTM-CNN-CRF** as the main network structure. On the embedding layer, apart from the **vanilla character embedding** pre-trained with word2vec, we design **7 extra feature embeddings** that capture **mention dictionary** and **rich contextual information** to enhance vanilla character embedding for identifying mention boundaries effectively:

① Word Embedding: We use word2vec to train on word sequences cut by jieba, with mention dictionary as its default dictionary, and add the same word embedding to each character in the word.

② Charcater Position Feature: With segmentation results from jieba, we can also provide boundary information by adopting BMES tagging scheme to represent character positions in the word.

③ Position-aware Charcater Embedding: We combine character sequences with the character position label sequences descried above, which are again trained using word2vec to get position-aware character embeddings.

④ Max Match Feature: We use BiDirectional Maximum Matching algorithm with mention dictionary to segment sentences as well. We then use BMEO tagging scheme to indicate if the character is in a matched mention and its position.

⑤ N-gram Match Feature: Based on the pre-defined n-gram feature templates, we use a multi-hot vector to represent text segments that contain the character and its surroundings are mentions or not.

⑥ Bigram Embedding: We also use word2vec to train on bigram sequences to get bigram embeddings, which have been shown useful for word segmentation.

⑦ BERT Embedding: We try 3 pre-trained language models: BERT, ERNIE and BERT_wwm to generate enriched contextual representation for each character.

Entity Disambiguation Model. We use **BiLSTM-CNN** as the main network structure to generate representations for mention and candidate entity. We then use **cosine function** to calculate semantic similarity score for each (mention, entity) pair. The highest scored candidate entity will be chosen as the matching one.

① Mention Representation: To take **mentions' positions** into consideration, we first calculate each character's relative distance from the mention and convert it to a **position embedding**, which will be combined with vanilla character embedding as input. Besides, we represent mention only using **hidden states of the mention part**. Specifically, we concatenate the first and the last hidden state as well as the results of max-pooling and self-attention over hidden states of mention part.

② Entity Representation: Apart from using **max-pooling** over hidden states, we also use attention mechanism, by using mention representation to attend to hidden states, to produce entity representation. We explore 3 ways to calculate attentive weight: **additive mention**, **multiplicative attention** and **scaled-dot attention**.

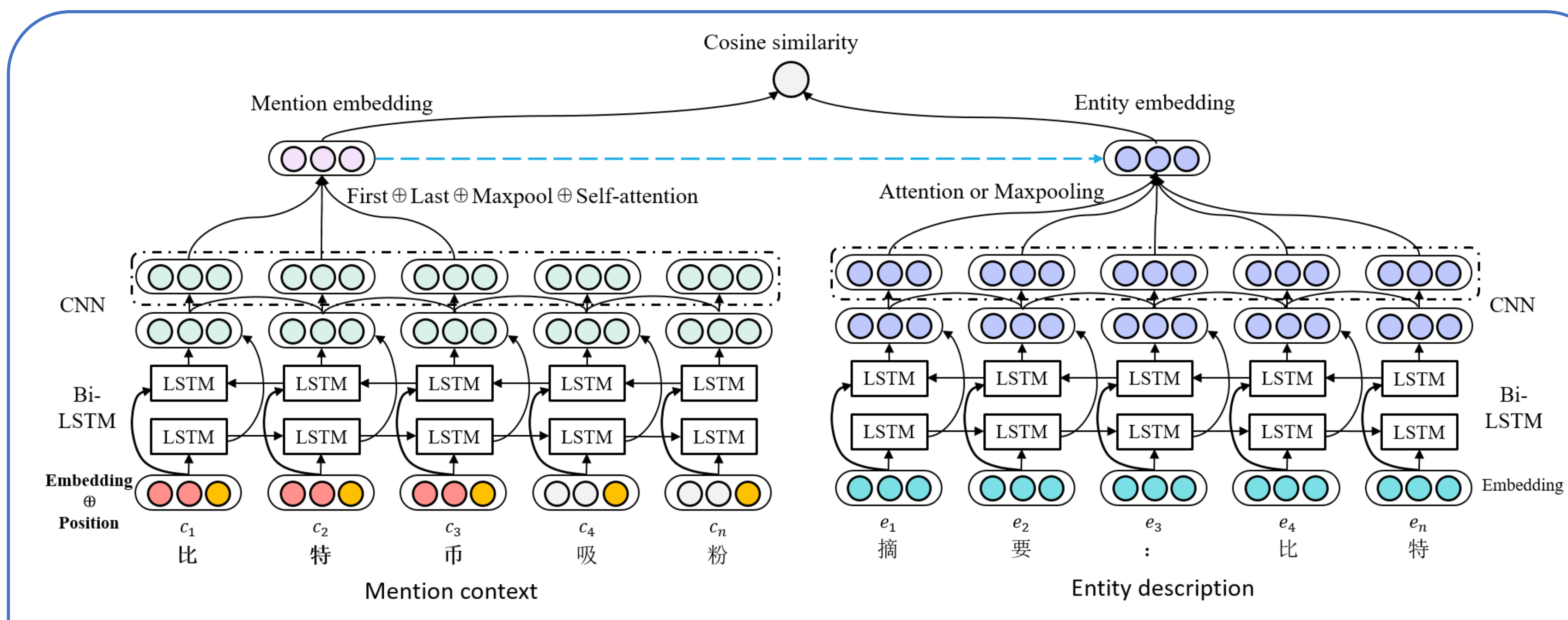

Fig. 3 Overall framework of ED model

Ensemble. We exploit 2 ensemble strategies to improve the final performance.

① Weights Averaging: During a single training process, we maintain a copy of weights of the model being trained, to keep track of the **averaged weights**.

② Outputs Averaging: We train various ER and ED models under different model configurations and then simply **average their outputs** to obtain final predictions.

Evaluation Results

Table 1. F1 scores of ER models on dev set.

Model	Dev F1	
		Weights ensemble
vanilla character	0.77316	0.77524
enhanced character (BERT)	0.83145	0.83323
enhanced character (ERNIE)	**0.83175**	**0.83355**
enhanced character (BERT-wwm)	0.83019	0.83126
ER model ensemble	-	**0.83836**

Table 2. Ablation study of enhanced character embeddings of ER model on dev set.

Model	Dev F1	
		Weights ensemble
enhanced character (ERNIE)	0.83175	0.83355
- word embedding	0.82878	0.83042
- character position feature	0.83080	0.83173
- position-aware character embedding	0.83159	0.83303
- n-gram match feature	0.83067	0.83185
- max match feature	0.83009	0.83116
- bigram embedding	0.83148	0.83282
- bert embedding	0.82969	0.83034

Results of ER Model. Enhanced character embedding does bring a significant performance gain. In ablation study, we observe performance degradation when eliminating any embedding. We train various models using different embeddings combination as input. We then combine them based on weights and output ensemble strategies and observe further performance boost.

Table 3. F1 scores of ED models on dev set.

Model			Dev F1	
Input	**Mention**	**Entity**		**Weights ensemble**
no pos	over all	maxpool	0.89582	0.90096
no pos	over 2 sides	maxpool	0.89414	0.89971
no pos	over subseq	maxpool	0.89902	0.90296
pos	over all	maxpool	0.89727	0.90169
pos	over 2 sides	maxpool	0.89682	0.90125
pos	**over subseq**	**maxpool**	**0.89925**	**0.90344**
pos	over subseq	add attend	0.89844	0.90249
pos	over subseq	mul attend	0.89891	0.90308
pos	over subseq	scaled-dot attend	0.89837	0.90278
ED model ensemble			-	**0.90965**

Results of ED model. For mention representation, our approach that incorporates mention position information achieves the best performance, indicating the utility of such information. For entity representation, “maxpool” strategy outperforms all the attention-based strategies. We then combine all the single models in the table above using weights and outputs ensemble strategies and obtain better results.

Results of EL model. Combining the ensemble ER and ED model is our proposed EL model for Chinese short text entity linking. Our solution achieves **a F1 score of 0.79266** on the final test set of CCKS2019 Task 2.

Enlightenments & Conclusions

We propose an enhanced character embedding based neural approach for Chinese short text entity linking, which explicitly encodes mention dictionary and mention position information.

😀 Our model achieves significant improvement over a collection of baselines on CCKS2019 Task 2, verifying the utility of external information.

☹ We haven't figure out a good way to apply BERT model to entity disambiguation due to the limit of GPU memory and time. We leave it to the future work.

Acknowledgement

This work was supported by 2019 Tencent Marketing Solution Rhino-Bird Focused Research Program.