Entity Position Matters - Towards Improving a Generative End-to-End Relation Extraction Model

Shanshan Liu Yuji Matsumoto

Center for Advanced Intelligence Project, RIKEN
{shanshan.liu yuji.matsumoto}@riken.jp

Abstract

In order to improve the performance of the generative end-to-end relation extraction method for the material synthesis procedure extraction task, we added direct entity position information to the existing generative model during decoding, and explored whether the utilization of position information is effective in the situation after the output space changes. Experimental results show that when the output space is equipped with the absolute positions of the input tokens, the improvement is limited with the pretrained language model (PLM) that uses absolute position embeddings, while the PLM using relative position embeddings can be significantly improved.

1 Introduction

With the rise of generative methods, end-to-end relation extraction (RE) has become a topic of increasing interest in recent years [1, 2, 3, 4]. The task requires identifying entities and capturing relations between pairs of entities. Most generative methods do not provide the location of entities in the text, and often treat mentions that appear at different locations in the text but have the same content equally [1, 2, 3].

In some applications, the position of an entity in the original text is very critical and cannot be neglected. Position information can help us distinguish between two mentions with the same content but in different places. It can also inform us about the role of entities in more complex structured information. For example, in the material synthesis procedure extraction task, two ”heating” operations that occur at different places in the procedure should have distinct effects. Their duration and temperature can be quite different. It would be confusing if the model only outputs text triplets (head entity, tail entity, relation type) without knowing the exact positions of the entities, such as (heating, 2h, condition_of) and (heating, 1h, condition_of).

One generative model that meets the need for entity locations for such tasks is the BartNER model [5]. This model generates entity information by setting the output spaces to the types of the entity and the positions of input tokens. With the practical output formats for relation-level information, the model is capable to do end-to-end RE [6].

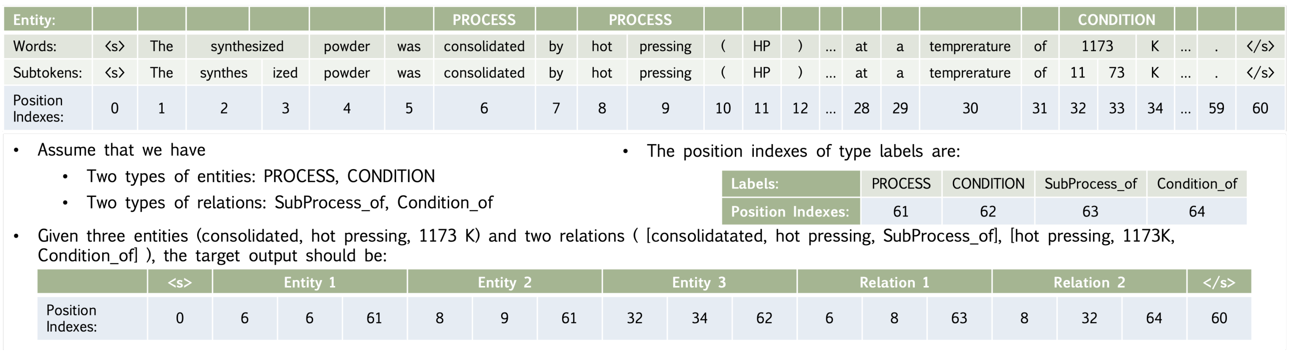
Generative pretrained language models (PLMs) like BART [7] or T5 [8], have taken into account the position information of input text. Thus, the generative model gives acceptable performance even if the target output changes from tokens (i.e. predicting the most appropriate token from the whole vocabulary) to position indexes (i.e. predicting the position of the most suitable token in the original text).

We are interested in the following question: *Does the change of the output space put forward higher requirements for the learning of position information by the generative model?* In order to explore this problem, based on BartNER, we added a feature of decoded entity offsets before doing classification to enhance position information. We conducted experiments using models with BART and T5 as PLMs.

Experimental results showed that the added feature only slightly improved the model using BART, but significantly improved the model using T5. On tasks that require absolute position information, the added feature has a positive impact on models that utilize relative position embeddings. The small effect on BART indicates that BART has mastered absolute position well, and strengthening position information would not be a helpful way to boost it.

Our contributions:

- Increasing the impact of entity position information by adding position features to the BartNER model,



Entity:							PROCESS		PROCESS										CONDITION					
Words:	<s>	The	synthesized		powder	was	consolidated	by	hot	pressing	(	HP	)	...	at	a	temprerature	of	1173		K	...	.	</s>
Subtokens:	<s>	The	synthes	ized	powder	was	consolidated	by	hot	pressing	(	HP	)	...	at	a	temprerature	of	11	73	K	...	.	</s>
Position Indexes:	0	1	2	3	4	5	6	7	8	9	10	11	12	...	28	29	30	31	32	33	34	...	59	60

- Assume that we have
    - Two types of entities: PROCESS, CONDITION
    - Two types of relations: SubProcess_of, Condition_of
- The position indexes of type labels are:

Labels:	PROCESS	CONDITION	SubProcess_of	Condition_of
Position Indexes:	61	62	63	64

- Given three entities (consolidated, hot pressing, 1173 K) and two relations ([consolidatated, hot pressing, SubProcess_of], [hot pressing, 1173K, Condition_of]), the target output should be:

	<s>	Entity 1			Entity 2			Entity 3			Relation 1			Relation 2			</s>
Position Indexes:	0	6	6	61	8	9	61	32	34	62	6	8	63	8	32	64	60

Figure 1 Example of the target output under our definition.

improves the effect of end-to-end relation extraction.

- We found that pretrained language models using absolute position embeddings outperform those using relative position embeddings in the task of directly generating token indexes instead of token text.
- The pretrained language model using absolute position embeddings is recommended for tasks that require entity offset.

2 Methods

2.1 Task formulation

In this work, the end-to-end RE task can be formulated as generating a sequence of entity spans and relation spans given input text. First, we assign specific indexes to tokens in the input text, entity types, and relation types following BartNER's scheme [5]. Given an input text containing n tokens, the i_th token will be assigned with index i. If we have m types of entities and k types of relations, the range of indexes of the entity types is $(n, n+m]$, and the range of indexes of the relation types will be $(n+m, n+m+k]$. Then we let the model generate all entity spans represented by assigned indexes. The entity span is in the form of **[first subtoken, last subtoken, Entity type]**. After that, we let the model generates all relation spans in the form of **[first subtoken of Head entity, first subtoken of Tail entity, Relation type]**. The order of spans is determined by the appearance order of entities in the text when generating entities or by the appearance order of the head entities when generating relations (See Fig. 1).

2.2 BartNER

BartNER model uses Encoder-Decoder of the generative PLM as a basis to make the index probability distribution P_t at time t given input X, generated indexes $\hat{Y}_{<t} = [\hat{y}_1, ..., \hat{y}_{t-1}]$, and labels of entity and relation Z as follows:

$$H^e = Encoder(X) \tag{1}$$

$$E^e = TokenEmbed(X) \tag{2}$$

$$\hat{H}^e = MLP(H^e) \tag{3}$$

$$\bar{H}^e = \alpha * \hat{H}^e + (1-\alpha) * E^e \tag{4}$$

$$h_t^d = Decoder(H^e; \hat{Y}_{<t}) \tag{5}$$

$$Z^d = TokenEmbed(Z) \tag{6}$$

$$P_t = Softmax([\bar{H}^e \otimes h_t^d; Z^d \otimes h_t^d]) \tag{7}$$

where TokenEmbed is the token embeddings used in PLM; $\hat{Y}_{<t}$ should be indexes of input tokens or indexes of entity/relation types; α is a hyper-parameter set as 0.5 in this work; $[\cdot;\cdot]$ means concatenation of vectors.

2.3 Position Feature

According to the index scheme used in decoding, the original algorithm may be viewed as a generation process where the output space is changed to the absolute position of the concatenated text of the input tokens and all entity types. We intuitively consider that the embeddings of generated indexes can help introduce absolute position information that has not been complicatedly encoded by PLM. We therefore refer to this embeddings as a position feature. Given that only entity position information is generated in the end-to-end RE task, this feature actually only provides entity position information.

To incorporate this feature into the decoding process, we change the algorithm in the following way:

$$h_t^p = PositionFeature(\hat{y}_{t-1}) \tag{8}$$

$$\bar{h}_t^d = Linear([h_t^d : h_t^p]) \quad (9)$$

$$P_t = Softmax([\bar{H}^e \otimes \bar{h}_t^d; Z^d \otimes \bar{h}_t^d]) \quad (10)$$

where PositionFeature is the learnable embeddings to embed the $\hat{y}_{t-1}$ into a 32-dim vector.

3 Experiments

3.1 Dataset

To confirm the effectiveness of the additional position information on the traditional sentence-level task, we choose **CoNLL04**[9], a news domain dataset consists of four types of entities and five types of relations. Data split is conducted in accordance with previous works. The dataset statistics are shown in Table 1.

In order to provide a task in which the entities offsets must be extracted, we chose **Procedure dataset**[1], a dataset contains 576 thermoelectric material papers annotated by experts. All procedure information is structured into five types of entities and seven types of relations. The data are randomly split with a ratio of training:validation:test = 8:1:1. It should be noted that we take the sequence of sentences extracted from the full text of the chemical paper by a text block extractor as input. To adapt to practical applications, after the text block containing procedure information has been automatically extracted, an end-to-end relation extractor is used to obtain specific entity- and relation-level information.

3.2 Evaluation

Following other end-to-end RE tasks[2], we report precision (Pre), recall (Rec) and micro-F1 scores (F1). The prediction of an entity is correct only when the type and offset match the golden entity. If the type and offset of the two entities and the relation type match the golden relation, the prediction is correct. All relations are directed. We report the average score after 3 runs of each setting.

3.3 Experiment Setup

We use a rewritten version of the BartNER model developed by Yan's group [5] based on their public code[2]. On the Procedure dataset, we performed experiments utilizing BART-large (facebook/bart-large) [7] and T5-base (google/t5-v1_1-base) [8] as the PLMs in our method. As the model using T5 cannot converge to a small loss, we only estimated the effect of applying BART-large on the CoNLL04 dataset. We compare our method with the SOTA models on CoNLL04, the REBEL[2].

4 Results

The experiment results are shown in Table 2 and Table 3. Even if we only use a trivial approach to bring entity offset information into the generation process, the position feature still gives us a slight improvement. However, this improvement varies by PLMs and tasks. The gap (7.67% RE F1) between our method and the SOTA model on the CoNLL04 reminds us there is still a lot of room for improvement, but not from the absolute position aspect.

4.1 Entity Position Matters

The improvement after adding the decoded position information indicates that after the output space is changed to absolute positions, the requirements for the ability to utilize position information of the generative model are higher.

Taking advantage of the absolute position embeddings, the BART-based model is less affected and only slightly improved (0.27% RE F1 on Procedure dataset; 0.32% RE F1 on CoNLL04). This means that BART has learned absolute position information well, so simply providing similar information is somewhat redundant.

On procedure extraction, adding the position feature improves the Precision, but makes the Recall salient decrease(-2.52% NER Recall, -0.83% RE Recall). The count of predicted spans (both entities and relations) also decreases. We think this is related to the procedure extraction task itself. The text to be processed by RE is long (hundreds of tokens), the number of entities and the number of relations in a piece of text are large (including one or more synthesis procedures, and a synthesis procedure consists of several processes and related materials, operating conditions). When the model uses the added position feature to better fit the training data but the semantic representation ability does not improve simultaneously, it will tend to predict lesser spans while processing unseen data.

CoNLL04 has fewer data and most inputs are short. There are many difficult relation cases that require knowledge to make correct predictions. Therefore, the generalization ability in RE is not significantly affected, only

1) The annotation results will be made public.
2) https://github.com/yhcc/BARTNER

	Entity Types	Relation Types	Train Inst	Train Rel	Validation Inst	Validation Rel	Test Inst	Test Rel
Procedure	5	7	460	8994	59	1305	57	1217
CoNLL04	4	5	922	1290	231	343	288	422

Table 1 Dataset statistics.

PLM	With Position Feature	NER Pre.	NER Rec.	NER F1	RE Pre.	RE Rec.	RE F1
BART-large	Yes	80.82	82.02	81.41	**66.41**	64.33	**65.34**
BART-large	No	80.63	**84.54**	**82.52**	65.02	**65.16**	65.07
T5-base	Yes	**83.67**	79.43	81.49	65.57	57.17	61.08
T5-base	No	82.37	76.90	79.54	64.65	54.89	59.37

Table 2 Results on Procedure dataset.

PLM	With Position Feature	NER Pre.	NER Rec.	NER F1	RE Pre.	RE Rec.	RE F1
BART-large	Yes	**86.09**	84.06	**85.06**	70.92	58.81	64.30
BART-large	No	84.40	**84.65**	84.52	71.33	57.99	63.98
REBEL					**75.22**	**69.01**	**71.97**

Table 3 Results on CoNLL04 dataset.

the number of predictions in entity recognition is reduced, bringing lower NER Recall.

The absolute improvement of 1.71 points on RE F1 when using T5-base demonstrates the importance of position information.

4.2 Position embeddings Matters

From Table 2 we can see that BART-large (RE F1: 65.34%) outperforms T5-base (RE F1: 61.98%) a lot. That's quite interesting because both BART and T5 achieve promising results on so many tasks, there should not be that huge difference in their capability. In addition to the language model sizes (BART-large is twice the size of the T5-base), we speculate that this large discrepancy is caused by their different position embedding schemes.

Although both have their strengths and weaknesses in text-to-text tasks, relative position encoding appears to be a bit incompetent when it comes to tasks that output absolute positions. The lack of learning about the absolute position makes the T5-based model significantly improved on every indicator after corporated with the position feature.

4.3 Training Problem using T5-base

An interesting finding is that for large pretrained models with an encoder-decoder structure, BART can adapt to the output space we defined, but T5 cannot cope with changed settings. It is hard to get a relatively small loss value after we changed the PLM to T5-base. We tried several combinations of hyperparameters (changed the learning rate and increased the training epoch, etc.) to get the reported results on the Procedure dataset, and failed the training on CoNLL04 (NER F1: 28.69%; RE F1: 15.44%). Up to the present, we still have not found an appropriate training hyperparameter setting that allows the model to converge stably and effectively. The performance of T5, which we barely achieved, is lagging behind the BART results.

The training problem of the T5-based we met emphasizes that the information we hope the model to learn should compatible with the output space. Or we can design a better output space to fully exploit the potential of the PLMs.

5 Conclusion

In this work, we explore two generative PLMs when the output space is position indexes instead of text by introducing absolute position information. Results show that BART performs better due to its absolute position embeddings, and adding entity offset information does not have a significant impact. However, there is still a distance from the advanced model. T5, a model using relative position embeddings can benefit from added position feature, but there are still cases of low recall or difficult training. We recommend considering the task output space when picking a PLM, or transforming the output format to fit the model.

References

[1] Martin Josifoski, Nicola De Cao, Maxime Peyrard, and Robert West. Genie: generative information extraction. **arXiv preprint arXiv:2112.08340**, 2021.

[2] Pere-Lluís Huguet Cabot and Roberto Navigli. REBEL: Relation extraction by end-to-end language generation. In **Findings of the Association for Computational Linguistics: EMNLP 2021**, pp. 2370–2381, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[3] Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Song. DeepStruct: Pretraining of language models for structure prediction. In **Findings of the Association for Computational Linguistics: ACL 2022**, pp. 803–823, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[4] Tianyu Liu, Yuchen Jiang, Nicholas Monath, Ryan Cotterell, and Mrinmaya Sachan. Autoregressive structured prediction with language models, 2022.

[5] Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. A unified generative framework for various ner subtasks. **arXiv preprint arXiv:2106.01223**, 2021.

[6] Liu Shanshan, Ishigaki Tatsuya, Uehara Yui, Takamura Hiroya, and Matsumoto Yuji. Generate it. a simple method for end-to-end relation extraction. 2023.

[7] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. **CoRR**, Vol. abs/1910.13461, , 2019.

[8] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. **CoRR**, Vol. abs/1910.10683, , 2019.

[9] Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. In **Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004**, pp. 1–8, Boston, Massachusetts, USA, May 6 - May 7 2004. Association for Computational Linguistics.