Generate it.
A Trivial Method for End-to-End Relation Extraction

Shanshan Liu[1] Tatsuya Ishigaki[2] Yui Uehara[2] Hiroya Takamura[2] Yuji Matsumoto[1]

[1]Center for Advanced Intelligence Project, RIKEN
[2]Artificial Intelligence Research Center, AIST
{shanshan.liu yuji.matsumoto}@riken.jp
{ishigaki.tatsuya, yui.uehara, takamura.hiroya}@aist.go.jp

Abstract

End-to-end Relation Extraction (RE) is a fundamental problem of information extraction, which includes two tasks: identifying named entities from text and classifying relations between entities. In this work, we propose a simple but effective method to extract entities and relations from text jointly by designing the target output of a BART-based generative model for Named Entity Recognition (NER) without changing its architecture. Compared to existing methods on ChEMU, our method performs better on RE and produces comparable results on NER. Our experimental results also demonstrate that the generative model designed for a single task is capable of joint learning.

1 Introduction

To obtain necessary information from natural language text, it is common to perform Named Entity Recognition (NER) and Relation Extraction (RE) on the same text. This task of identifying both entities and the relations between entities is called End-to-End RE.

In recent times, joint models have gained popularity as a way to utilize the interaction between entities and relations. A joint model is often more complicated than a pipeline model. The joint approach can be implemented using higher-level data structures, such as transforming tasks into table-filling tasks [1] or graph-based methods [2]. Another option is to share parameters [3] and representations [4] or to add cross-task constraint [5].

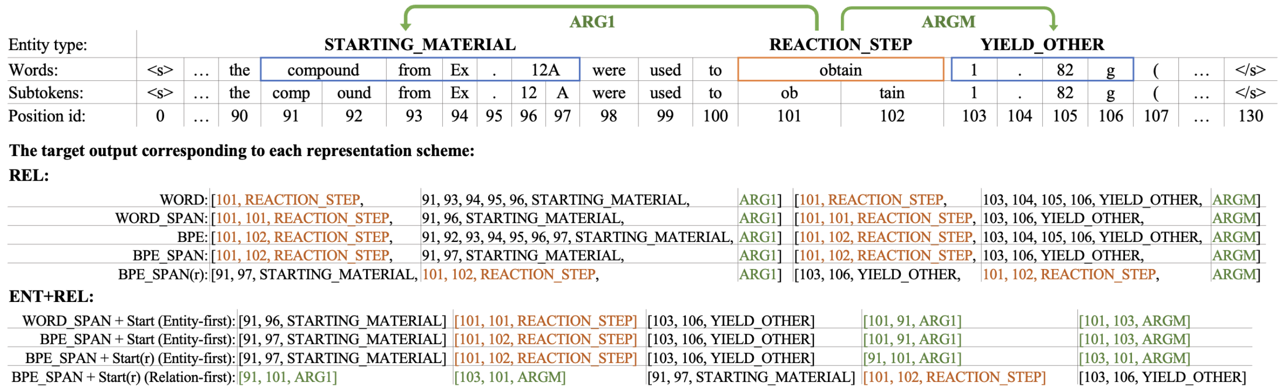
The brilliance of the joint model does not mean that the traditional pipeline method is obsolete. A pipeline frame of end-to-end RE usually consists of two models for two different tasks. [6] proposed a pipeline method PURE, which outperforms most of joint models by adding special tokens to introduce entity position and type information. PL-Marker [7] is also a pipeline extractor that advanced in three datasets, provides a novel span representation approach to consider the dependencies between the spans.

Models built for end-to end RE task are often complex and difficult to be implemented for more practical tasks. Most well-performing models treat the task as a span-based problem, so that their computation complexity is high. Even though the length of input text does not exceed the limit of the pre-trained language model (PLM), if there are far more entities and relations in the input than those at the sentence-level, the model may have difficulty handling new datasets due to its complexity.

Inspired by a recent generative framework BARTNER [8] developed for the NER task, we convert the end-to-end RE task into a generation problem. Formulating a span-based problem to a generation problem can effectively reduce the computational complexity - there is no need to enumerate all possible entity spans, and to pair entities one by one to extract the relation triple. BARTNER exploits *BART-large* as encoder-decoder and achieves state-of-the-art (SOTA) performance on multiple NER benchmarks. Since *BART-large* is a PLM with a large set of trainable parameters, we make a wild guess that BARTNER can serve as a simple and efficient model for end-to-end RE if it is trained with appropriate target outputs, without any change in its architecture.

Our method achieves competitive results on ChEMU [9] compared to existing powerful methods. In summary, the main contributions of this work are listed as follows:

- We adopt a generative framework for NER to end-to-



Entity type:				STARTING_MATERIAL										REACTION_STEP		YIELD_OTHER						
Words:	<s>	…	the	compound		from	Ex	.	12A		were	used	to	obtain		1	.	82	g	(	…	</s>
Subtokens:	<s>	…	the	comp	ound	from	Ex	.	12	A	were	used	to	ob	tain	1	.	82	g	(	…	</s>
Position id:	0	…	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	…	130

The target output corresponding to each representation scheme:

REL:

WORD:	[101, REACTION_STEP,	91, 93, 94, 95, 96, STARTING_MATERIAL,	ARG1]	[101, REACTION_STEP,	103, 104, 105, 106, YIELD_OTHER,	ARGM]
WORD_SPAN:	[101, 101, REACTION_STEP,	91, 96, STARTING_MATERIAL,	ARG1]	[101, 101, REACTION_STEP,	103, 106, YIELD_OTHER,	ARGM]
BPE:	[101, 102, REACTION_STEP,	91, 92, 93, 94, 95, 96, 97, STARTING_MATERIAL,	ARG1]	[101, 102, REACTION_STEP,	103, 104, 105, 106, YIELD_OTHER,	ARGM]
BPE_SPAN:	[101, 102, REACTION_STEP,	91, 97, STARTING_MATERIAL,	ARG1]	[101, 102, REACTION_STEP,	103, 106, YIELD_OTHER,	ARGM]
BPE_SPAN(r):	[91, 97, STARTING_MATERIAL,	101, 102, REACTION_STEP,	ARG1]	[103, 106, YIELD_OTHER,	101, 102, REACTION_STEP,	ARGM]

ENT+REL:

WORD_SPAN + Start (Entity-first):	[91, 96, STARTING_MATERIAL]	[101, 101, REACTION_STEP]	[103, 106, YIELD_OTHER]	[101, 91, ARG1]	[101, 103, ARGM]
BPE_SPAN + Start (Entity-first):	[91, 97, STARTING_MATERIAL]	[101, 102, REACTION_STEP]	[103, 106, YIELD_OTHER]	[101, 91, ARG1]	[101, 103, ARGM]
BPE_SPAN + Start(r) (Entity-first):	[91, 97, STARTING_MATERIAL]	[101, 102, REACTION_STEP]	[103, 106, YIELD_OTHER]	[91, 101, ARG1]	[103, 101, ARGM]
BPE_SPAN + Start(r) (Relation-first):	[91, 101, ARG1]	[103, 101, ARGM]	[91, 97, STARTING_MATERIAL]	[101, 102, REACTION_STEP]	[103, 106, YIELD_OTHER]

Figure 1 Examples of target output under different representation schemes. With three entities and two relations in the input, the target output consists of two relation spans in REL scheme while five spans (three for entities and two for relations) in ENT+REL scheme. "[...]" indicates the content in one span. Relation spans in ENT+REL are highlighted by green. Details of all representations are in 2.2.

end RE by providing proper representations, without any change in model architecture. This proves that generative models can jointly learning for NER and RE given proper target outputs.

- Under the representations we designed, we achieve comparable or better performance than the SOTA methods for ChEMU.
- Our method is more efficient and flexible in prediction than others - It is no need to enumerate all possible entity spans and relation triples, or limit the length of the span.

2 Method

2.1 Task definition

A golden sample in the ChEMU dataset has the following components: All words in a chemical reaction snippet is concatenated as input. Each mention in the input is an entity. Each entity may or may not be related to other entities in the same snippet.

The target output of each sample is a sequence of spans according to our defined representation. Regardless of the kind of representation we use, the target output should contain the boundary and type information for each entity, as well as the information to indicate two entities and the type for each relation.

2.2 Representation

This section explains how the entities and relations are represented in our generative model. Please refer to the examples in Fig.1.

2.2.1 Entity

We followed the three types of entity representations proposed in [8], namely *WORD*, *BPE*, *BPE_SPAN*, and added a new type of representation, *WORD_SPAN*.

- WORD The position indexes of each word in the entity are included in the entity span, concatenated with the entity type tag.
- BPE The position indexes of each subtoken in the entity are included in the entity span, concatenated with the entity type tag.
- WORD_SPAN The position indexes of the first subtokens of the first and last words in the entity are included in the entity span, concatenated with the entity type tag.
- BPE_SPAN The position indexes of the first and last subtokens in the entity are included in the entity span, concatenated with the entity type tag.

2.2.2 Relation (REL)

To make directly generation of relations possible, we tried a representation scheme as follows. The relation representation combines corresponding entity spans in [Head, Tail, Relation type] or [Tail, Head, Relation type] format. [Head, Tail, Relation type] are directly named after the entity representation type (e.g. *BPE*). When we used the latter format, we indicate it with (r) (e.g. *BPE(r)*).

2.2.3 Entity+Relation (ENT+REL)

In some end-to-end RE tasks, entities that do not appear in any relations may also need to be identified. To cope with this situation, we propose the representation scheme of Entity+Relation. If we do the entity-first generation, the model will generate all entity spans (in *WORD_SPAN* or *BPE_SPAN* that are consistent with 2.2.1). After that, the model generates all relation spans in following way.

- Start The position indexes of the first subtokens in the head and tail entities are included, concatenated with the relation type.
- Start(r) The position indexes of the first subtokens in the tail and head entities are included, concatenated with the relation type.

Alternatively, when we do the relation-first generation, the model generates all relation spans and then all entity spans ((e.g. *BPE_SPAN* + *Start(r) (Relation-first)*)).

2.3 Generative model

Following [8], we concatenate the representations of all spans in each text as the generated output. 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. We use the model developed by [8] but a rewritten version by us.

3 Experiment

3.1 Dataset

We experiment on the ChEMU dataset, which involves entity and event annotations of 1500 chemical reaction snippets. The event triggers are annotated as entities, and event arguments are treated as relations between an event trigger and an argument. Some entities may not be part of any events. The end-to-end RE on ChEMU is extracting 12 types of entities and 2 types of relations. We use the official data split that training:development:test = 7:1:2.

	DOC	ENT	REL
Train	900	23210	14310
Dev	225	5439	5448
Test	375	9435	5803
All	1500	38084	25561

Table 1 Statistics of the ChEMU dataset.

This dataset is chosen for two reasons: 1) Each sample contains rich entity and relation information, while most of entities are shown in annotated relations. So that the joint model requires equal effort to learn both entity and relation extraction information. 2) We hope to experiment on a practical data and help the development of automatic extraction of chemical domains, not only on sentence-level end-to-end task.

3.2 Baseline

We compare our method with the SOTA models on end-to-end RE task. DyGIE++ [4] is a general framework for several information extraction tasks including NER, RE, and EE. It is a span-based method using the dynamic span graph for better span representation. We implemented DyGIE++ in the default setting, with the PLM: RoBERTa. Since many samples in ChEMU would exceed the maximum input length of RoBERTa, we limited the task to the sentence-level when we apply DyGIE++. [1] [2]

The SOTA of end-to-end RE on ChEMU is a pipeline model provided by Melaxtech [10]. Melaxtech adopts BioBERT and BiLSTM-CRF for sentence-level NER while adds a linear classification layer on top of the BioBERT to predict the label of a candidate entity pair. The cross-sentence relations are extracted by post-processing not deep learning model. We directly report the values published on that paper.

3.3 Evaluation

The evaluation is similar to previous works. If the type and offsets of the entity match the gold entity, the prediction is correct; if the types and offsets of two entities and the relation type match the gold relation, the predicted relation is correct. All relations are directed. We report the precision (P), recall (R) and micro F1 score (F).

4 Results

The result of end-to-end RE (Tab.2) shows several representations we provided exceed the baselines, whether generating relation spans containing all entity information (Ours-REL) or generating entity and relation spans together (Ours-ENT+REL). *Ent-first* + *B_S* + *Start* achieves the new SOTA RE F1 of 92.61% [3].

[1] This setting makes DyGIE++ inaccessible to cross-sentence relations, so 0.88% of the relations in the test set are not seen.

[2] Due to the limitation of PLM input length, 1.65% of the relations in test set are unreachable when we perform document-level task by our approach.

[3] Give a null hypothesis: Melaxtech has the same RE performance as Ours-*Ent-first* + *B_S* + *Start*,the p-value < .00001.

		ENT	REL		NER			RE	
Model	First	Rep	Rep	P	R	F	P	R	F
DyGIE++	x	x	x	94.86%	87.22%	90.88%	95.07%	86.49%	90.58%
Melaxtech	x	x	x	95.71%	95.70%	95.70%	92.01%	91.47%	91.74%
Ours-REL	x	x	W	94.64%	80.34%	86.91%	91.34%	83.78%	87.40%
	x	x	W (r)	94.17%	80.79%	86.97%	91.02%	84.58%	87.68%
	x	x	B	95.53%	81.02%	87.68%	86.45%	84.96%	85.69%
	x	x	B (r)	95.88%	81.79%	88.27%	87.26%	85.71%	86.48%
	x	x	W_S	94.80%	84.09%	89.13%	91.58%	89.56%	90.56%
	x	x	W_S (r)	94.83%	84.50%	89.37%	91.51%	89.92%	90.71%
	x	x	B_S	96.03%	85.21%	90.30%	93.20%	91.44%	92.31%
	x	x	B_S (r)	95.91%	84.96%	90.10%	93.23%	91.09%	92.15%
Ours-ENT+REL	Ent	W_S	Start	95.04%	93.27%	94.15%	95.52%	89.20%	92.25%
	Rel	W_S	Start	94.79%	93.68%	94.23%	93.15%	86.77%	89.85%
	Ent	B_S	Start	95.41%	94.43%	94.91%	94.98%	90.35%	92.61%
	Rel	B_S	Start	95.59%	94.55%	95.07%	93.71%	87.78%	90.65%
	Ent	W_S	Start (r)	94.45%	93.40%	93.92%	95.30%	88.76%	91.92%
	Rel	W_S	Start (r)	94.78%	93.84%	94.31%	93.21%	88.04%	90.55%
	Ent	B_S	Start (r)	95.39%	94.72%	95.05%	94.72%	89.99%	92.29%
	Rel	B_S	Start (r)	95.83%	94.75%	95.29%	93.19%	88.97%	91.03%

Table 2 Results for the ChEMU dataset. ”W”: *WORD*; ”B”: *BPE*; ”W_S”: *WORD_SPAN*; ”B_S”: *BPE_SPAN*. ”r” indicates that when generating a relation span, the information of the tail entity is generated before the head entity. Ours-REL means that the target output follows the setup in 2.2.2, and ”Ours-ENT+REL” generates the target output under the definition in 2.2.3.

4.1 Representation matters

We found performance differences among different combinations. There is a huge gap between span and non-span representations. The F1 scores of RE between *B(r)* and *B_S(r)* in Ours-REL differ by nearly 6 points. We speculate that representations in the form of spans are stronger because they are lesser affected by the length of the entities. There are many long chemical substance names that may be divided into dozens of subtokens in ChEMU, making the task faced by non-span representation more difficult. In the case of REL, losing a subtoken in the middle will cause a relation and an entity falsely predicted simultaneously. However, when entity generation is not that closely tied with relation generation (ENT+REL), both the performance on NER and RE get better. Another reason for the low NER recalls by Ours-REL is that not every entity is included in relations. In addition, reversing the generation order of head and tail entities, or reversing the generation order of entities and relations, also have clear effects.

4.2 Joint learning: DyGIE++ vs. Ours

All representations in ENT+REL outperform DyGIE++ with differences between 3 to 4 points of NER F1 scores. Under the default setting of DyGIE++, the loss weight of RE is 1.0 and the weight of NER is 0.2, which can guarantee the effect of RE, but the part of NER is far inferior to our method. Except for *Rel-first* + *W_S* + *Start* and *Rel-first* + *W_S* + *Start(r)*, other representations in ENT+REL outperform DyGIE++. The low recall of DyGIE++ is primarily responsible for this gap. The limitation on span length makes DyGIE++ incapable of predicting long entities in the ChEMU. Overall these results, demonstrate that our method is a stronger joint model than DyGIE++ on ChEMU[4].

5 Conclusions

By proposing representations suitable for end-to-end RE, our approach allows the generative model designed for NER tasks to learn both entity- and relation-level information without increasing model complexity. This method provides SOTA results on the ChEMU dataset. Our work is still in progress - as we have only experimented on texts rich in entity and relation information. We will find out how well this method performs on more practical tasks, such as a task with higher proportion of cross-sentence relations, or event extraction. Also, we will keep exploring the potential of generative models in joint learning.

[4] Give a null hypothesis: DyGIE++ has the same RE performance as Ours-*Ent-first* + *B_S* + *Start*, p-value < .00001.

Acknowledgements

This paper is partly based on results obtained from a project, JPNP16010, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). Computational resource of AI Bridging Cloud Infrastructure (ABCI) provided by National Institute of Advanced Industrial Science and Technology (AIST) was used.

References

[1] Jue Wang and Wei Lu. Two are better than one: Joint entity and relation extraction with table-sequence encoders. **arXiv preprint arXiv:2010.03851**, 2020.

[2] Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. Joint type inference on entities and relations via graph convolutional networks. In **Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics**, pp. 1361–1370, 2019.

[3] Makoto Miwa and Mohit Bansal. End-to-end relation extraction using lstms on sequences and tree structures. **arXiv preprint arXiv:1601.00770**, 2016.

[4] David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. Entity, relation, and event extraction with contextualized span representations. **arXiv preprint arXiv:1909.03546**, 2019.

[5] Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. A joint neural model for information extraction with global features. In **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pp. 7999–8009, 2020.

[6] Zexuan Zhong and Danqi Chen. A frustratingly easy approach for joint entity and relation extraction. **CoRR**, Vol. abs/2010.12812, , 2020.

[7] Deming Ye, Yankai Lin, and Maosong Sun. Pack together: Entity and relation extraction with levitated marker. **arXiv preprint arXiv:2109.06067**, 2021.

[8] 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.

[9] Dat Quoc Nguyen, Zenan Zhai, Hiyori Yoshikawa, Biaoyan Fang, Christian Druckenbrodt, Camilo Thorne, Ralph Hoessel, Saber A Akhondi, Trevor Cohn, Timothy Baldwin, et al. Chemu: named entity recognition and event extraction of chemical reactions from patents. **Advances in Information Retrieval**, Vol. 12036, p. 572, 2020.

[10] JWYRZ Zhang and Yaoyun Zhang. Melaxtech: a report for clef 2020–chemu task of chemical reaction extraction from patent. **Work Notes CLEF. Published online.[Google Scholar]**, 2020.

[11] 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. pp. 7871–7880, Online, July 2020. Association for Computational Linguistics.

A Implementation details

We rewrote the BARTNER [8] using NumPy, PyTorch and Hugging Face's Transformers referring to Yan's public code. https://github.com/yhcc/BARTNER The codes of pre-processing and evaluation are written with NumPy and PyTorch.

The generative model is trained with max sequence length 1024 and batch size 1 for 30 epochs, without limitation on the length of each span. Only a single run for each representation.

No hyperparameter search and early stopping in our experiments. The details of hyperparameters are in Tab.3. We do not apply beam search during decoding.

All the entity and relation tags are set as special tokens and be initiated by prompt initiation. The parameter size of the model we implemented is 1633.624064M when we use *BART-large* [11] as the encoder-decoder. All experiments are conducted on are trained on a 20-core(CPU) machine with 1 GPU (NVIDIA A100 for NVLink 40GiB HBM2). The wallclock time of training one representation is approximately 15000s (30 epochs in total), maximum memory reaches 28.802G on the ChEMU dataset.

Hyper	Value
adam_epsilon	1e-06
batch_size	1
betas	[0.9, 0.999]
decoder_drop_rate	0.3
grad_clip	5.0
hidden_size	1024
learning_rate	1e-05
num_epochs	30
reset	5
vocab_size	50280
warmup_ratio	0.01
weight_decay	0.001

Table 3 Part of hyperparameters used in experiments.