PLMs as Meta-function

Learning In-context Learning for Named Entity Recognition[CLL+23]

潘禹丞

2023年12月21日

Literature

Challenge of NER:

- Diversity of entity types
- Emergence of new entity types
- Lack of high-quality annotations

Few-shot learning techniques to address:

- Fine-tuning-based methods
- Metric-based methods
- In-context learning(this work)

Few-shot Learning Techniques

Fine-tuning-based methods:

- Adjust model weights using new instances
- Drawbacks:
  1. Expensive re-training
  2. New entity types cannot be addressed on-the-fly

Metric-based methods:

- Learning to compare query instances with support instances or prototypes
- Limitations:
  1. Matching architectures
  2. Sensitive to domain shift

This paper proposed an in-context learning-based NER approach

Notations

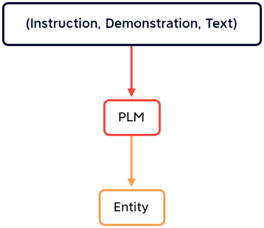
Prompt:

- I: Instruction, target entity types
  [*Target types: Disease, Virus*]
- D: Demonstration, examples in prompt
  [*Text: Cancer is a leading ...*
  Entities: Cancer is disease.
  Text: Rabies virus is estimated ...
  Entities: Rabies virus is virus.]
- T: Text, to be extracted
  [*SARS-CoV-2 is a strain of coronavirus that causes COVID-19.*]

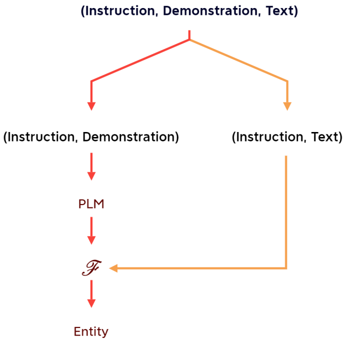
Response:

- E: Entities, Expected response.
  [*SARS-CoV-2 is virus. COVID-19 is disease.*]

Core Idea



(a) Overall



(b) This Paper

Core Idea

- Model pre-trained language models(PLMs) as a meta-function $\lambda_{I,D,T}.\mathcal{M}$
- Extractor function $\mathcal{F}$ extracts entities from text.
- When give PLMs a new prompt (I', D', T'),

$$\lambda.\mathcal{M}(I', D') \rightarrow \mathcal{F} \quad (1)$$

$$\mathcal{F}(T') \rightarrow E' \quad (2)$$

- Primary work: Use a meta-function pre-training algorithm to inject the in-context NER ability into PLMs.
- Source of Idea: What learning algorithm is in-context learning? Investigations with linear models[**?**]

1. Literature
2. Core Idea
3. **Pre-training**
   - Structure
   - Corpus Construction
   - Loss Function
   - Pre-training Procedure
4. Experiments and Results
5. Summary
6. Reference

Structure

- Input $X = [I; D; T]$
  - Instruction $I = [i_1, ..., i_n]$
  - Demonstrations $D = [d_1, ..., d_m]$
    - d_i="Text:{text}, Entities:{extractions}"
  - Text T
- Output $Y = [e_1, ..., e_n]$, where e_i is the i-th extracted entities.
- Architecture: encoder-decoder network(T5) [RSR+20]

Structure

Instruction

Target types: *disease; virus*

Demonstrations

Text: *Cancer is a leading cause of death worldwide.*
Entities: *Cancer is disease.*
Text: *Rabies virus is estimated to cause around 55,000 deaths per year.*
Entities: *Rabies virus is virus.*

Text

Text: *SARS-CoV-2 is a strain of coronavirus that causes COVID-19.*

Extractions

Entities: *SARS-CoV-2 is virus. COVID-19 is disease.*

Corpus Construction

Construct in-context NER corpus from traditional NER corpus

- $\mathcal{D}_{NER} = \{x'_1, ...\}$, where each x'_i is a (T, E) pair.
- $\Rightarrow \mathcal{D}_{in-context} = \{x_1, ...\}$, where each x_i is an in-context NER task (I, D, T, E)

Sampling Method:

- In-context Task Sampling
- Type Anonymization

In-context Task Sampling

1. I: Sample n target entity types (E in $\mathcal{D}_{NER}$)
   $I = [e_1, ..., e_n] \triangleq [i_1, ..., i_n]$
2. D: Sample k instances for each type ($n \times k$ in total)
   $D = [(t_{1,1}, e_1), ..., (t_{1,k}, e_1), ..., (t_{n,1}, e_n), ..., (t_{n,k}, e_n)]$
3. T, E:
   - Randomly sample an instance corresponding to a certain entity type in I
   - Or randomly sample an instance from instances of other entity types, to construct NIL instances. The proportion of NIL instances is hyperparameter γ.

Type Anonymization

- Objective: Ensure the models rely on in-context demonstrations for entity knowledge, and avoid overfitting to entity type names
- Anonymize entity types by randomly substituting them with a set of type indicators {<type1>, . . ., <type99>}, rather than directly using the original type names such as Disease and Virus
- The substitute probability for each name is 80%.

Loss Function

Objective: Optimize the in-context learning ability and the extraction ability.

$$\mathcal{L} = \alpha \mathcal{L}_{meta-function} + \mathcal{L}_{extraction} \tag{3}$$

where α is the coefficient of meta-function loss, and defined as 0.05 in the source code, the formulas of $\mathcal{L}_{meta-function}$ and $\mathcal{L}_{extraction}$ will be discussed later.

Meta-function Pre-training

Optimize $\| \mathcal{F} - \mathcal{F}^* \|$ directly if $\mathcal{F}^*$ is known.
However, $\mathcal{F}^*$ is unknown, so we need to use surrogate fine-tuned extractor $\mathcal{F}'$ to approximate $\mathcal{F}^*$.

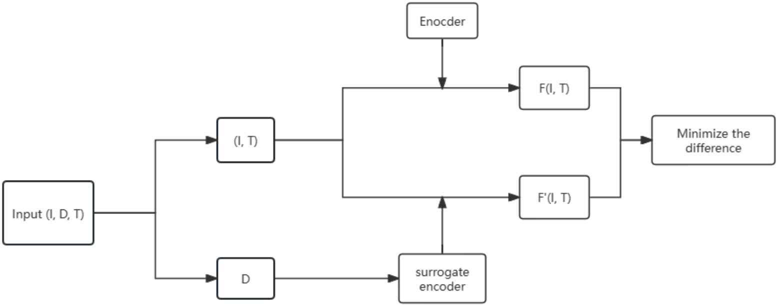


图 1: Meta-function Pre-training

Meta-function Pre-training

For each given task (I, D, T, E), the meta-function pre-training procedure needs the input $X = (I, D, T)$:

1. Separate the input into D and (I, T)
2. Use encoder to represent the meta-function $\mathcal{F}$
3. Use D to fine-tune the surrogate encoder with one-step gradient, then to encode (I, T) to get the features of extractor $\mathcal{F}'$: $F' = \mathcal{F}'(I, T) = Encoder'(I, T)$
4. Get the features of $\mathcal{F}$ by $F = \mathcal{F}(I, T) = Encoder(I, T)$
5. Train encoder with loss function

$$\mathcal{L}_{meta-function} = \frac{1}{n+k} \sum_{i=i}^{n+k} \parallel F_i - F_i^* \parallel_2$$

with gradient descent $\nabla\theta_{encoder} = \frac{\partial \mathcal{L}_{meta-function}}{\partial X}$ where $\mathcal{F}^*$ considered constant, n is the number of target types, k is the number of texts.

Extraction Function Pre-training

The decoder generates all extractions as a tokenized text sequence $Y = [y_1, \cdots, y_n]$.

- The sequence-to-sequence entity extractor directly models the generation probability
- Optimize the model parameters θ by minimizing the negative likelihood of in-context instances

$$\mathcal{L}_{extraction} = -\log \prod_{i=1}^{|Y|} P(y_i | y_{<i}, X, \theta) \tag{4}$$

and the extraction gradient is computed as

$$\nabla\theta = \frac{\partial \mathcal{L}_{extraction}}{\partial X} \tag{5}$$

Extraction Pre-training Tasks Construction

- Entity Extraction Task
  - In-context NER settings, input is (I, D, T), with type anonymization
  - Traditional NER settings, input is (I, T), without type anonymization
- Pseudo Extraction Language Modeling Task
  - Objective: Enlarge the training corpus with the text corpus for language modeling pre-training $\mathcal{D}_{text}$
  - Randomly sample unlabeled sentences from the text corpus
  - Automatically build pseudo extraction tasks according to both in-context $((I, D, T, E'))$ and traditional $((I, T, E'))$ NER settings, where E' is pseudo entities.

More about Pseudo Extraction

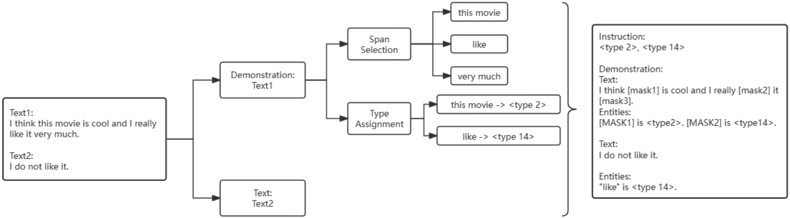


图 2: Pseudo Extraction Language Modeling Task

Corpus

- Build a large-scale distant NER dataset by aligning Wikipedia and Wikidata.
- Filter ambiguous and low-frequency types (occurrences $<$100k) to obtain higher-quality demonstrations
- Retain 2046 types and 55 million (T, E) pairs and use a 40/15 million split for training/validation
- Sample 5 million in-context tasks for training and 10k for validation

Settings

- Pre-training settings
  - Initial model: T5-v1.1-large
  - Pre-train 500k steps with learning rate=5e-5 and warm-up steps=10k
- Experiment settings
  - Few-shot settings: standard k-shot NER settings[HLS+21]
  - Evaluation settings: micro-F1, report the average performance by repeating each experiment 10 times
  - Test datasets: CoNLL03, WNUT17, NCBI-disease, SEC-filings

Main Results

Models	#Param	CoNLL03		WNUT17		NCBI-disease		SEC-filings		AVE
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	
Pre-trained Language Models										
T5v1.1-large	770M	38.61	44.90	25.52	26.32	26.02	37.63	41.89	53.44	36.79
GPT2-xl	1.5B	33.69	39.55	22.63	24.86	25.54	33.25	42.83	**57.05**	34.93
T5-xl	3B	38.99	45.74	26.39	26.31	23.10	36.78	30.58	42.22	33.76
GPT-J-6B	6B	46.14	50.10	31.41	30.93	35.82	40.98	40.12	39.61	39.39
T5-xxl	11B	40.97	46.14	24.76	25.27	12.19	26.34	32.65	42.44	31.35
OPT-13B	13B	46.65	51.71	27.74	28.36	23.73	34.00	41.60	43.10	37.11
GPT-Neox-20B	20B	52.68	58.12	**36.29**	35.68	35.42	42.85	45.07	45.17	43.91
OPT-30B	30B	42.86	44.77	25.85	27.44	22.31	32.76	40.83	46.52	35.42
OPT-66B	66B	43.83	53.89	30.77	32.00	25.87	34.58	39.15	47.01	38.39
Pre-trained NER Models										
ProtoNet	345M	30.04	60.26	9.74	23.03	24.73	42.32	16.79	23.67	28.82
NNShot	345M	41.92	58.39	15.76	21.78	31.59	33.14	30.19	37.86	33.83
StructShot	345M	42.34	58.44	15.78	22.05	19.87	31.48	30.40	38.44	32.35
CONTAINER	345M	45.43	61.69	15.64	20.37	23.24	27.02	34.07	40.44	33.49
MetaNER-base	220M	53.94	62.59	25.55	30.41	35.00	37.24	46.88	51.39	42.88
MetaNER	770M	**57.40**	**63.45**	31.59	**36.52**	**40.01**	**44.92**	**52.07**	54.87	**47.60**

Table 1: Micro-F1 scores of 1-shot and 5-shot in-context NER on test set. For a fair comparison, the results of each model are based on a single frozen model without fine-tuning and the pre-trained NER models are pre-trained using the same dataset as MetaNER.

Detailed Studies

	CoNLL03			NCBI-disease		
	P	R	F1	P	R	F1
MetaNER	73.59	57.19	**64.34**	**54.96**	**36.85**	**43.79**
w/o MF	68.97	57.62	62.77	38.27	35.26	36.28
w/o LM	70.86	**57.99**	63.77	37.54	34.82	35.67
w/o anonymization	**74.75**	52.86	61.93	47.47	35.30	40.48

Table 2: Ablation studies on dev set. The results are based on 5-shot setting.

(a) Ablation Studies

	CoNLL03		WNUT17	
	1shot	5shot	1shot	5shot
BERT-large (Devlin et al., 2019)	14.66	52.43	8.95	32.77
T5-v11-large (Raffel et al., 2020)	11.65	42.13	12.51	39.54
GPT-NEO-20B (Black et al., 2022)*	52.68	58.12	36.29	35.68
UIE-large (Lu et al., 2022b)	46.28	67.62	32.86	42.67
SDNet (Chen et al., 2022a)	/	71.40	/	44.10
CONTAINER-FT (Das et al., 2022)	48.56	66.45	19.46	24.95
MetaNER-ICL*	57.40	63.45	31.59	36.52
MetaNER-FT	**61.51**	**72.70**	**39.68**	**47.26**

Table 3: The experiments of fine-tuning based methods. * indicates in-context learning settings. CONTAINER is pre-trained using the same NER dataset as MetaNER. All the models are implemented by us except SDNet.

(b) ICL vs Fine-tuning

Summary

- PLMs can be considered as meta-functions to implicitly generate functions and implement learning algorithm
- We can use meta-function pre-training to insert in-context learning ability to PLMs, which considered only appearing on LLM before
- How to evaluate and improve the meta-functions need to be improved

[CLL+23] Jiawei Chen, Yaojie Lu, Hongyu Lin, Jie Lou, Wei Jia, Dai Dai, Hua Wu, Boxi Cao, Xianpei Han, and Le Sun.
Learning in-context learning for named entity recognition, 2023.

[HLS+21] Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han.
Few-shot named entity recognition: An empirical baseline study.
In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10408–10423, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[RSR+20] 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.
J. Mach. Learn. Res., 21(1), jan 2020.

Thanks!