Hybrid Multi-stage Decoding for Few-shot NER with Entity-aware Contrastive Learning

Anonymous ACL submission

Abstract

Few-shot named entity recognition can identify new types of named entities based on a few labeled examples. Previous methods employing token-level or span-level metric learning suffer from the computational burden and a large number of negative sample spans. In this paper, we propose the Hybrid **M**ulti-**s**tage Decoding for **F**ew-shot **NER** with Entity-aware Contrastive Learning (**MsFNER**). Specifically, we first detect named entities without type and then classify entity types by the entity classification module. We divide **MsFNER** into 3 stages: training stage, finetuning stage, and inference stage. In the training stage, we train the entity-span detection model and the entity classification model separately on the source domain, where we create a contrastive learning module to enhance entity representations for entity classification. During finetuning, we finetune the model on the support dataset of target domain. In the inference stage, we replace the contrastive learning module with a *KNN* module and the final entity type inference is jointly determined by *KNN* and entity classification module. We conduct experiments on the open FewNERD dataset and the results demonstrate the advance of **MsFNER**.

1 Introduction

Named Entity Recognition (NER) is a fundamental task in Natural Language Processing (NLP), involving the identification and categorization of text spans into predefined classes such as people, organizations, and locations (Yadav and Bethard, 2018; Li et al., 2022). Although traditional deep neural network architectures have achieved success in the fully supervised named entity recognition (NER) task with sufficient training data (Lample et al., 2016; Ma and Hovy, 2016), they are difficult to adapt to the dynamic nature of real-world applications, which require the model to quickly adjust itself to new data or environmental changes. In this case, researchers have proposed the few-shot named entity recognition (Few-shot NER) to explore entity recognition with limited labeled data (Fritzler et al., 2019). Few-shot NER enables existing models to quickly transfer learned knowledge and adapt to new domains or entity classes.

Specifically, the Few-shot NER model is first trained on the source domain dataset $D_{source} = (S_{source}, Q_{source})$ and then the trained model is transferred to new target domain dataset $D_{target} = (S_{target}, Q_{target})$ to infer for Q_{target} (Snell et al., 2017). We can consider a piece of data in D_{source} or D_{target} as a Few-shot NER task and formalize it as $T = (S, Q)$, where S represents the support dataset comprising N entity types (N-way), and each type is exemplified by K annotated examples (K-shot). The Q is the query dataset with the same entity types as the support dataset. Few-shot paradigm can offer a flexible and cost-effective solution to the adaptability challenge, making it a focal point of research to enhance the performance of NER systems in scenarios with limited labeled data or emerging entity types.

Currently, there are two mainstream researches for Few-shot NER: token-level (Fritzler et al., 2019; Hou et al., 2020; Yang and Katiyar, 2020; Das et al., 2022) and span-level metric learning methods (Wang et al., 2022a; Yu et al., 2021; Ma et al., 2022). In token-level methods, each token is assigned an entity label based on its distances from the prototypes of entity classes or the support tokens. However, these approaches often have high computational costs and fail to maintain the semantic integrity of entity tokens, leading to increased susceptibility to interference from non-entity markers. On the other hand, although span-level methods can mitigate the partial issues associated with token-level approaches by evaluating entities as spans, all possible spans are enumerated would result in an N-square complexity and an increase in noise from a large number of negative samples.

Considering the challenges, we hope to solve the

following problems: 1) To improve the Few-shot NER identification efficiency, how can we encourage the semantic divergence between entities and non-entities to determine effective entity spans? 2) To improve the entity span classification, how can we control and coordinate the semantic distance of different entity types to make the semantic representations of entities within the same types be proximate while those in disparate types be distant?

In this paper, we propose a hybrid multi-stage decoding approach for few-shot NER with contrastive learning (i.e., **MsFNER**). Specifically, **MsFNER** can be divided into three stages: training stage, finetuning stage, and inference stage. The training stage consists of two objectives: training the best entity span detection model and the best entity classification model separately based on the supervised dataset of source domain. In this stage, we train entity detection model by employing the Conditional Random Field (*CRF*) layer, and train entity classification model through the entity-aware contrastive learning and a softmax layer. In the finetuning stage, we finetune the trained entity detection model and entity classification model on the support dataset of target domain. In the inference stage, we deploy the *KNN* and two finetuned models on the query dataset of target domain to predict entity spans and their types, and there are four phases for prediction. In the first phase, we build a *key-value* datastore $\mathcal{D}_{knn}$ with the support dataset S_{target}, where *key* is the entity representation and *value* is the corresponding label. Subsequently, we apply the entity detection model to get entity spans. With the entity detection model, we could reduce the impact of negative samples and reduce computational complexity. After that, we pass the representations of detected entity spans into the entity classification model to produce their predicted type distribution p_{soft} while we also feed the representations into *KNN* to obtain the predicted results p_{knn} based on the $\mathcal{D}_{knn}$. In the final phase, we combine p_{soft} and p_{knn} to obtain final types for prediction entities.

We summarize our contributions as follows:

- We propose a hybrid multi-stage decoding for few-shot NER with entity-aware contrastive learning, in which we first detect entity spans to improve efficiency and then employ contrastive learning or *KNN* to augment entity classification performance.
- Experimental results show that our model achieves new SOTA performance compared with previous works. We also evaluate the few-shot NER task on LLM ChatGPT and the experimental results show that our model outperforms ChatGPT in terms of performance and efficiency.

2 Methodology

In this section, we will introduce the details of **MsFNER**. The following content is arranged according to the three stages of the model.

2.1 Training Stage

In this stage, we introduce the separate process of training and finding the optimal entity span detection model and entity classification model.

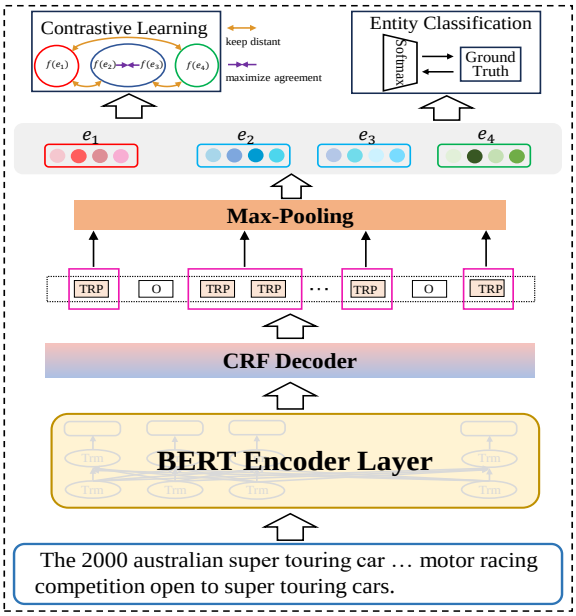

Figure 1: The schematic diagram of the training stage.

2.1.1 Entity Span Detection (ESD)

In this module, we regard the entity span detection as a sequence labeling task with the *BIOES* tagging scheme. The tag set L = {*B, I, O, E, S*} means that we only care about the boundaries of the entities without entity types.

For a given sentence x with n tokens x=$(x_1, x_2, ..., x_n)$ in D_{source}, we first encode it using the pre-trained language model BERT. After that, we can obtain the contextualized representations $h = (h_1, h_2, ..., h_n)$ for all tokens.

The sequence token representations are then sent into a *CRF* to detect entity spans. Finally, the training loss in D_{source} can be depicted as:

$$L_{ESD} = -\sum logP(y|x) \quad (1)$$

$$P(y|x) = \frac{\prod_{i=1}^{|x|} \varphi_i(y_{i-1}, y_i, x)}{\sum_{y'} \prod_{i=1}^{|x|} \varphi_i(y'_{i-1}, y'_i, x)} \quad (2)$$

where $\varphi_i(y_{i-1}, y_i, x)$ and $\varphi_i(y'_{i-1}, y'_i, x)$ are potential functions.

2.1.2 Entity Classification

For an entity $e_k = (x_f, ..., x_{f+l})$, where f is the first token index and $f + l$ is the last token index in the sequence, we employ the max-pooling to get its representation when $l >= 1$:

$$\hat{e}_k = max(h_f, ..., h_{f+l}) \quad (3)$$

Contrastive learning could enhance the consistency between entities within the same types and widen the distance between entities belonging to different types. As e_k is the sample with certain type from the supervised D_{source}, we consider using the supervised contrastive learning to deepen the distinctiveness of entity types and augment representations of entities for entity classification improvement.

In details, given a batch with *N* entities and the anchor index $j \in \{1,2,...,N\}$, an entity-aware contrastive loss can be defined as follows:

$$L_{CL} = \sum_j -\frac{1}{|P(j)|} \sum_{p \in P(j)} log \frac{exp(sim(z^j, z^p)/\tau)}{\sum_{a \in A(j)} exp(sim(z^j, z^a)/\tau)} \quad (4)$$

where *P(j)* is the positive set whose entities are from the same type with e_j, z^j is the result of transforming $\hat{e}_j$ through a projection network MLP, *A(j)* is the set containing all *N* entities except for e_j, sim(·,·) denotes the *KL*-divergence for K-shot (K>1) but the squared euclidean distance for K-shot (K=1) referring to (Das et al., 2022), and τ is a temperature hyperparameter.

After pulling entities of the same type together and entities of different types farther apart in the semantic space with contrastive learning, we discard the projection network at classification time following the previous works of contrastive learning (Chen et al., 2020; Khosla et al., 2020; Liu et al., 2023). As a consequence, we put the entity representation $\hat{e}_k$ into the *softmax* function to compute the probability distribution of entity types (K shot) and also get entity classification loss with the cross entropy(CE) function:

$$p_{soft}(e_k) = softmax(\hat{e}_k) \quad (5)$$

$$L_{EC} = \frac{1}{N} \sum_k CE(y_k, p_{soft}(e_k)) \quad (6)$$

where $p_{soft}(e_k) \in \mathbb{R}^{|K|}$, y_k is the truth type of e_k.

2.2 Finetuning Stage

Model finetuning aims to enable the model to adapt to new knowledge types during the domain transfer process. We finetune the trained models for entity detection and entity classification on the support dataset of the target domain S_{target}, using the same method as training stage.

2.3 Inference Stage

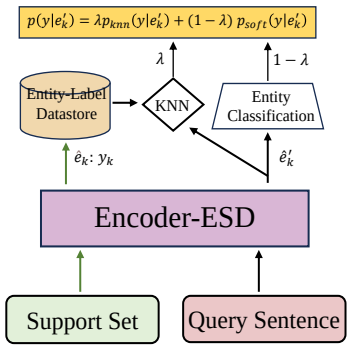

Figure 2: The schematic diagram of the inference stage for entity classification.

In the inference stage, we test the finetuned model on query dataset Q_{target}, and we will present this stage by four parts:

(1) We compute the representation $\hat{e}_k$ of each true entity in the support dataset S_{target} by Eq.3 and build a *key-value* datastore $\mathcal{D}_{knn}$ where *key* is the entity representation and *value* is entity type.

(2) We input the query sentence x' into the finetuned best model for entity span detection (ESD) to get all entities. Specifically, we apply the Viterbi algorithm (Forney, 1973) for decoding, and derive the entities from the maximum sequence probability of Eq.2:

$$y^* = argmax_{y \in \mathbb{Y}} P(y|x') \quad (7)$$

where $\mathbb{Y}$ is a set of all possible label sequences.

(3) Firstly, we compute the representation $\hat{e}'_k$ of each detected entity e'_k by Eq.3. Then, we input $\hat{e}'_k$ into two independent modules: on one hand, we feed the $\hat{e}'_k$ into *KNN* to obtain the predicted result p_{knn} based on the $\mathcal{D}_{knn}$ (Wang et al., 2022b), and on the other hand we pass the $\hat{e}'_k$ into the finetuned best model for entity classification to produce its predicted type distribution p_{soft} by Eq.5.

(4) The final prediction type of the detected entity e'_k is jointly determined by *KNN* and the entity classification model:

$$p(y|e'_k) = \lambda p_{knn}(y|e'_k) + (1-\lambda) p_{soft}(y|e'_k) \quad (8)$$

where the λ is a hyper-parameter that makes a bal-

Models	FewNERD-INTRA				FewNERD-INTER			
	1-2 shot		5-10 shot		1-2 shot		5-10 shot	
	5 way	10 way	5 way	10 way	5 way	10 way	5 way	10 way
ProtoBERT†	23.45±0.92	19.76±0.59	41.93±0.55	34.61±0.59	44.44±0.11	39.09±0.87	58.80±1.42	53.97±0.38
NNShot†	31.01±1.21	21.88±0.23	35.74±2.36	27.67±1.06	54.29±0.40	46.98±1.96	50.56±3.33	50.00±0.36
StructShot†	35.92±0.69	25.38±0.84	38.83±1.72	26.39±2.59	57.33±0.53	49.46±0.53	57.16±2.09	49.39±1.77
CONTAINER†	40.436	33.84	53.70	47.49	55.95	48.35	61.83	57.12
ESD†	41.44±1.16	32.29±1.10	50.68±0.94	42.92±0.75	66.46±0.49	59.95±0.69	74.14±0.80	67.91±1.41
MAML-ProtoNet†	52.04±0.44	43.50±0.59	63.23±0.45	56.84±0.14	68.77±0.24	63.26±0.40	71.62±0.16	68.32±0.10
ChatGPT‡	**61.86**	**55.61**	64.17	54.79	61.05	57.94	64.34	58.58
MsFNER (Ours)	54.25±0.34	46.69±0.5	**66.57±0.29**	**58.70±1.39**	**72.91±0.34**	**66.34±0.65**	**78.41±0.30**	**72.06±0.11**
w/o cl*	—	—	65.72±0.33	57.33±0.08	—	—	77.84±0.42	71.23±0.47
w/o *KNN*	53.86±0.47	46.24±0.29	66.04±0.65	57.56±0.41	72.75±0.29	65.38±0.72	77.99±0.25	71.92±0.14

Table 1: F1 scores with standard deviations on FewNERD. †denotes the results reported in (Ma et al., 2022). ‡are the results we produce by LLM. * means the abbreviation of 'contrastive learning'. The best results are in bold.

ance between *KNN* and *softmax*.

3 Experiments

3.1 Experiments Setup

Datasets We conduct experiments on the FewNERD dataset (Ding et al., 2021). And more details about FewNERD are shown at A.1.1.

Parameter Settings The details are at A.1.2.

Baselines The baseline models include ProtoBERT (Fritzler et al., 2019), CONTAINER (Das et al., 2022), NNShot (Yang and Katiyar, 2020), StructShot (Yang and Katiyar, 2020), ESD (Wang et al., 2022a), MAML-ProtoNet (Ma et al., 2022), ChatGPT3.5 (chatgpt, 2023). And more details are at A.1.3 and A.2.

3.2 Main results

Table 1 illustrates the main results of different methods on FewNERD dataset. The results demonstrate that **MsFNER** significantly outperforms all the previous state-of-the-art approaches. **MsFNER** attains average F1 improvements of 2.65 and 4.44 on INTRA and INTER, respectively, compared to the previous best method, MAML-ProtoNet. In the 5-way 5-10 shot setting on the INTER dataset, we achieve the most substantial improvement over the MAML-ProtoNet method, reaching **7.79** points. Additionally, we perform the few-shot task employing ChatGPT, where **MsFNER** surpasses ChatGPT in F1 performance by an average score of **8.465** in the 5-10 shot setting. Furthermore, **MsFNER** surpasses ChatGPT by an average of **10.13** percentage points in the 1-2 shot setting in the INTER dataset. Overall, regardless of whether it is ChatGPT or small models, increasing the number of K-shot yields superior performance, while an increase in the number of N-way results in worse performance.

3.3 Ablation Study

To investigate the contributions of different modules of **MsFNER**, we conduct the ablation study by removing each of them at a time to observe the performance of our model. The results are shown at the bottom of Table 1. Firstly, we remove the contrastive learning module. In the 1 shot experiment of this setting, *KNN* is unnecessary as there is only one sample in the support dataset. We can see that the removal of contrastive learning results in a decrease of **0.905** average score, which indicates the significant impact and necessity of contrastive learning for enhancing representation. Secondly, we remove the *KNN* module and reserve the contrastive learning module. The removal of *KNN* drops **0.523** average score. We can see that the impact of contrastive learning and *KNN* increases with the number of N-way and K-shot. Removing either of them will lead to a significant decrease in the performance of **MsFNER**.

3.4 Model Efficiency

To evaluate the efficiency of **MsFNER**, we compute the average inference time of **MsFNER** and ChatGPT in different settings. More details can be found at A.3.

4 Conclusion

In this paper, we propose the **MsFNER**, aiming to improve the performance and efficiency of few-shot NER. The **MsFNER** can be achieved by three stages: training, finetuning and inferring. We evaluate **MsFNER** on the open FewNERD dataset, and results show the advance of **MsFNER** compared with previous few-shot NER methods and LLM.

5 Limitations

- In practical, the effectiveness of Few-shot NER is seriously affected due to the data sparsity and data imbalance.
- The abundant external knowledge is very helpful for few-shot NER, which is worth exploring and studying.

6 Ethical Considerations

All the data we get is the open source, and there is no theft. We guarantee the originality, research ethics, societal impact and reproducibility of our work.

References

chatgpt. 2023. https://openai.com/blog/chatgpt.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca Passonneau, and Rui Zhang. 2022. CONTaiNER: Few-shot named entity recognition via contrastive learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6338–6353, Dublin, Ireland. Association for Computational Linguistics.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3198–3213, Online. Association for Computational Linguistics.

G.D. Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 993–1000.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1381–1393, Online. Association for Computational Linguistics.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2022. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

Peipei Liu, Xin Zheng, Hong Li, Jie Liu, Yimo Ren, Hongsong Zhu, and Limin Sun. 2023. Improving the modality representation with multi-view contrastive learning for multimodal sentiment analysis. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Tingting Ma, Huiqiang Jiang, Qianhui Wu, Tiejun Zhao, and Chin-Yew Lin. 2022. Decomposed meta-learning for few-shot named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1584–1596, Dublin, Ireland. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Peiyi Wang, Runxin Xu, Tianyu Liu, Qingyu Zhou, Yunbo Cao, Baobao Chang, and Zhifang Sui. 2022a. An enhanced span-based decomposition method for few-shot sequence labeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5012–5024, Seattle, United States. Association for Computational Linguistics.

Shuhe Wang, Xiaoya Li, Yuxian Meng, Tianwei Zhang, Rongbin Ouyang, Jiwei Li, and Guoyin Wang. 2022b. k nn-ner: Named entity recognition with nearest neighbor search. *arXiv preprint arXiv:2203.17103*.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th*

International Conference on Computational Linguistics, pages 2145–2158, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375, Online. Association for Computational Linguistics.

Dian Yu, Luheng He, Yuan Zhang, Xinya Du, Panupong Pasupat, and Qi Li. 2021. Few-shot intent classification and slot filling with retrieved examples. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 734–749, Online. Association for Computational Linguistics.

A Appendix

A.1 Experiments Setup

We conduct experiments on the Tesla V100 GPU.

A.1.1 Datasets

We conduct experiments on the FewNERD dataset (Ding et al., 2021) which is annotated with 8 coarse-grained and 66 fine-grained entity types. We evaluate our method on the two settings of FewNERD: 1) FewNERD-INTRA, which pertains to a scenario wherein entities within the training set (source domain), validation set, and test set (target domain) are exclusively associated with distinct coarse-grained types. 2) FewNERD-INTER, where only fine-grained entity types that do not intersect in different sets. Furthermore, we follow the settings of ESD (Wang et al., 2022a) and adopt N-way $K \sim 2K$-shot sampling method to construct tasks.

A.1.2 Parameter Settings

We use grid search for hyperparameter settings and we train our model for 1,000 steps and choose the best model on validation dataset for testing. We use the pretrained language model uncased BERT-base as our encoder. In the entity span detection module, we adopt the BIOES tags. We use the AdamW optimizer with a learning rate of 3e-5 and 0.01 linear warmup steps. The K in *KNN* is set to 10 and the max-loss coefficient λ of *KNN* is 0.1 when adding *KNN*. The batch size is set to 32, the max sequence length is set to 128 and we use a dropout probability of 0.2.

A.1.3 Baselines

We compare MsFNER with popular baselines as follows:

- **ProtoBERT** (Fritzler et al., 2019) employs the prediction of query labels by leveraging the similarity between the BERT hidden states derived from the support set and the query tokens.

- **CONTAINER** (Das et al., 2022) employs token-level contrastive learning to train BERT as the token embedding function. Subsequently, BERT undergoes finetuning on the support set, followed by the application of a nearest neighbor method during the inference phase.

- **NNShot** (Yang and Katiyar, 2020) bears resemblance to ProtoBERT in its methodology, as it conducts predictions utilizing nearest neighbor algorithms.

- **StructShot** (Yang and Katiyar, 2020) incorporates an augmented Viterbi decoder into the inference stage atop NNShot's framework.

- **ESD** (Wang et al., 2022a) formulates few-shot sequence labeling as a span-level matching problem and decomposes it into span-related procedures.

- **MAML-ProtoNet** (Ma et al., 2022) proposes a decomposed meta-learning approach for few-shot NER, which tackles few-shot span detection and entity typing using MAML to facilitate rapid adaptation to novel entity classes and proposes MAML-ProtoNet for improved representation of entity spans.

- **ChatGPT** (chatgpt, 2023) is an advanced neural language model developed by OpenAI, which is trained on diverse corpora of text and could generate human-like text.

A.2 ChatGPT Prompt Template

In this section, we introduce the ChatGPT few-shot NER prompt template that we use. The prompt is composed of 3 parts: task description, few-shot cases, and input queries.

In the task description, we initially elucidate the few-shot Named Entity Recognition (NER) task required from ChatGPT. Subsequently, we provide ChatGPT with explicit instructions for implementing this task, including delineation of the entity types involved. To facilitate easier interpretation of model output, we utilize a structured output JSON

Models	FewNERD-INTRA				FewNERD-INTER			
	1-2 shot		5-10 shot		1-2 shot		5-10 shot	
	5 way	10 way	5 way	10 way	5 way	10 way	5 way	10 way
MsFNER	0.3442	0.3808	0.5616	1.0546	0.3482	0.3806	0.6708	1.2266
ChatGPT	4.5581	5.4897	8.1842	9.3481	3.2633	8.0579	4.7755	5.3409

Table 2: The comparison of inference time between **MsFNER** and ChatGPT.

format, a domain in which ChatGPT exhibits proficiency. The template is as follows:

Your task is to perform fewshot Named Entity Recognition. You could perform the task by the following actions:

1. *Do named entity recognition task to recognize the entities. Entity type is defined as* `$ent_list`
2. *Check if the entity type is in the definition list.*
3. *Output a JSON object that contains the following keys: text, entity_list. Note that each item in entity_list is a dictionary with "entity" and "type" as keys.*

Here are the given few-shot cases: ```$case_input```
Output: $case_output
Recognize the entities in the text delimited by triple backticks.: ```$input_text```

In the template above, the $ent_list denotes the list of entity types corresponding to N-way. The $case_input comprises the sentences in the support set corresponding to K-shot, while the $case_output represents the labels in the support set. Since ChatGPT is better at structured output, we use JSON format for output, with each output format being: *{"entity": entity text ; "type": entity type}*. The output will be merged into a list. The $input_text denotes the query data. When we input this prompt to ChatGPT, it will return a list of relevant entity-type dictionaries.

A.3 Model Efficiency

To evaluate the efficiency of **MsFNER**, we compute the average inference time of **MsFNER** and ChatGPT in different settings. And the results are shown in Table 2, where we can see that **MsFNER** is on the millisecond level while the ChatGPT is on the second level. The time consumption is related to the amount of input data. In addition, the time consumption of ChatGPT will increase due to network impact.