




Decomposed Meta-Learning for Few-Shot Sequence Labeling

Tingting Ma , Qianhui Wu , Huiqiang Jiang, Jieru Lin, Börje F. Karlsson, Tiejun Zhao , and Chin-Yew Lin

Abstract—Few-shot sequence labeling is a general problem formulation for many natural language understanding tasks in data-scarcity scenarios, which require models to generalize to new types via only a few labeled examples. Recent advances mostly adopt metric-based meta-learning and thus face the challenges of modeling the miscellaneous `Other` prototype and the inability to generalize to classes with large domain gaps. To overcome these challenges, we propose a decomposed meta-learning framework for few-shot sequence labeling that breaks down the task into few-shot mention detection and few-shot type classification, and sequentially tackles them via meta-learning. Specifically, we employ model-agnostic meta-learning (MAML) to prompt the mention detection model to learn boundary knowledge shared across types. With the detected mention spans, we further leverage the MAML-enhanced span-level prototypical network for few-shot type classification. In this way, the decomposition framework bypasses the requirement of modeling the miscellaneous `Other` prototype. Meanwhile, the adoption of the MAML algorithm enables us to explore the knowledge contained in support examples more efficiently, so that our model can quickly adapt to new types using only a few labeled examples. Under our framework, we explore a basic implementation that uses two separate models for the two subtasks. We further propose a joint model to reduce model size and inference time, making our framework more applicable for scenarios with limited resources. Extensive experiments on nine benchmark datasets, including named entity recognition, slot tagging, event detection, and part-of-speech tagging, show that the proposed approach achieves start-of-the-art performance across various few-shot sequence labeling tasks.

Index Terms—few-shot sequence labeling, task decomposition, meta-learning.

I. INTRODUCTION

SQUENCE labeling is the task that assigns a label to each element within a sequence. It has a wide range of applications in NLP, such as named entity recognition (NER) [1], slot tagging [2], event detection [3], and part-of-speech (POS) tagging [4]. However, most state-of-the-art systems for sequence labeling are based on deep neural architectures, and thus require large amounts of quality annotated data for training, which is costly and time-consuming to obtain. Moreover,

Manuscript received November 4, 2022. (Corresponding author: Tiejun Zhao.)

Tingting Ma, Jieru Lin, and Tiejun Zhao are with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China (e-mail: hittingtingma@gmail.com; v-jierulin@microsoft.com; tjzhao@hit.edu.cn).

Qianhui Wu, Huiqiang Jiang, Börje F. Karlsson, and Chin-Yew Lin are with Microsoft Research Asia, Beijing 100080, China (e-mail: {qianhuiwu, hjjiang, borje.karlsson, cy}@microsoft.com).

This work was conducted during Tingting Ma and Jieru Lin's internship at Microsoft Research Asia.

sequence labeling systems deployed in real-life applications are generally expected to rapidly adapt to emerging types for which there is limited labeled data available. Therefore, few-shot sequence labeling, which attempts to leverage labeled data from rich-resource types to recognize the new types with only a few labeled examples (*a.k.a.* support set), has attracted more and more attention in recent years [5]–[7].

One of the most effective solutions for few-shot sequence labeling has been metric-based meta-learning [5]–[9]. These methods learn a similarity metric function at either the token-level or the span-level, and assign the label of a query instance according to its similarities to support instances or class representations (*i.e.*, class prototypes) obtained from the support set. However, despite considerable progress in these metric-learning based approaches, there still exist several limitations:

i) The learned `Other` prototype(s) from few-shot examples cannot well represent all non-target instances (*e.g.*, tokens or spans that do not belong to any interested semantic types). Unlike the target instances which correspond to specific semantic meanings such as `PERSON` or `LOCATION`, the interpretation of non-target instances with `Other` labels is very disparate including function words, punctuation marks, and other non-target verbs or nouns. Therefore, it is unreasonable to represent all these non-target instances with only one [6] or multiple [8], [10] `Other` prototype(s) and cluster all representations of these instances around them for classification, which may even impair the embedding space of target types.

ii) The label dependency among tokens is difficult to model and transfer across tasks. Existing methods typically learn a conditional random field (CRF) [11] layer to model the label dependency between tokens [6], [12]. However, this practice has to estimate the label transition matrix from very limited examples under the few-shot setting, which leads to a biased transition matrix due to data sparsity [9]. Furthermore, knowledge of label dependency is hard to transfer across tasks due to the differences in label spaces.

iii) Adversity exists in large domain gap scenarios due to insufficient exploration of support examples. Current metric-learning based methods directly transfer the learned metric function to target tasks without any further adaptation [6], [8], [12]. Here, we contemplate that additionally exploring the information contained in target-task support examples can help bridge the domain gap and thus benefit the few-shot transfer, especially in distant cases.

In this paper, we propose a decomposed meta-learning framework to address the aforementioned limitations: we

distill the problem of few-shot sequence labeling into two subtasks, *i.e.*, few-shot mention detection and few-shot type classification, and sequentially tackle them via meta-learning.

To avoid the difficulties of modeling miscellaneous Other prototype(s) from sparse data, we propose to decompose the few-shot sequence labeling task. Specifically, the mention detector aims to detect mention spans that belong to target types with specific meanings, and the type classifier is developed based on the span-level prototypical network which assigns a type label for each detected span obtained from the mention detector. The decomposition framework allows the type classifier to only deal with spans corresponding to target interested semantic types and perform span-level metric learning without modeling miscellaneous Other prototypes. It is worth noting that the abstract tags (*e.g.*, BIO) are shared across different tasks at the mention detection step, although the target types are different across tasks. Therefore, **our decomposition framework can promote the transfer of knowledge about label dependencies among tokens by sharing the boundary knowledge across tasks through the abstract tag.**

Moreover, and key in our approach, we seek to **narrow the domain gap in both subtasks by thoroughly exploring the knowledge contained in support examples** via model-agnostic meta-learning (MAML) [13]. More specifically, at the mention detection step, we train the mention detector with the MAML algorithm for a good parameter initialization that models the shared boundary knowledge across different types. The target-type-specific boundary knowledge is further captured via a few gradient updates over support examples. In this way, our mention detector is expected to better generalize to examples of target types. At the type classification step, we propose MAML-SpanProto, a MAML-enhanced span-level prototypical network that takes advantage of the matching information in a few labeled examples to refine the learned type embedding space, and hence facilitates fast adaptation to target types.

In addition to the basic version using two separate models for the subtasks to implement our framework, we also propose a joint version that uses a shared BERT model, which can further reduce latency and is more suitable for resource-constrained scenarios, such as deployment on devices with limited capacity. A straightforward way to implement the joint version is to use a fully-shared BERT encoder, but it may fail to model task-specific knowledge, resulting in a significant drop in performance. To address this issue, we plug two groups of task-specific adapters into the upper layers of the shared BERT model, assuming that the lower layers are more task-agnostic [14] and can be fully shared between the two subtasks. In this way, the joint version can achieve comparable performance with the separate one while saving model size and computation cost.

We evaluate our approach on nine datasets of four classic sequence labeling tasks (*i.e.*, named entity recognition, slot tagging, event detection, and POS tagging) under multiple in-domain and cross-domain few-shot settings. Experimental results show that our approach significantly outperforms prior SOTA methods, especially in challenging settings where pre-

vious methods struggle. Comprehensive analyses demonstrate the effectiveness of our decomposition framework, the meta-learning enhancement, and its robustness under different tagging schemes.

This paper presents a substantially refined and improved version of our previous work [15], which focused on few-shot NER task. In this article, we make the following additional contributions:

- We generalize the few-shot task scope from named entity recognition to the broader field of sequence labeling, which further includes slot tagging, event detection, and part-of-speech tagging.
- We propose a joint model that further enhances the applicability of our framework in resource-constrained scenarios and reveals more insights into our method, showing the importance of task-specific knowledge and the performance gain of our approach without relying on more parameters through additional analysis and results.
- We include fresh start-of-the-art baselines and new experimental results on various benchmarks (*i.e.*, FewNERD, SNIPS, FewEvent, WSJ, and Twitter), demonstrating the effectiveness of our approach on various few-shot sequence labeling tasks.
- We provide additional ablations and analysis in Section V to discuss the working mechanism of our model, the robustness of our approach, and the factors that influence our model's performance.

Our code is publicly available at the GitHub.¹

II. TASK FORMULATION

Let $\mathbf{x} = (x_1, \dots, x_L)$ denote an input sequence with L tokens where x_l denotes the l -th token. The task of sequence labeling is to extract a set of $\mathbf{y} = \{(x_{[i,j]}, t)\}$, where each $x_{[i,j]} = (x_i, \dots, x_j)$, $i \leq j$, denotes a mention span (*i.e.*, entity in named entity recognition, slot in slot filling, event trigger in event extraction, or a token in POS tagging), $t \in \mathcal{T}$ denotes the type label associated with $x_{[i,j]}$, and \mathcal{T} is a pre-defined type set. Tokens that do not belong to any mention span are tagged with the label Other and $\text{Other} \notin \mathcal{T}$.

In this paper, we follow the typical setting, *i.e.*, N -way K -shot setting of few-shot sequence labeling as in [6], [7], [16]. In the meta-training phase, we manipulate training episodes $\mathcal{E}_{src} = \{(\mathcal{S}_{src}, \mathcal{Q}_{src}, \mathcal{T}_{src})\}$ constructed from source domain (training) data \mathcal{D}_{src} , where $\mathcal{S}_{src} = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N \times K}$ denotes the support set, $\mathcal{Q}_{src} = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N \times K}$ denotes the query set, \mathcal{T}_{src} denotes the type set, $|\mathcal{T}_{src}| = N$, and there are K instances for each type (so-called N -way K -shot). In the meta-testing phase, the model meta-trained on training episodes \mathcal{E}_{src} is expected to leverage the support set $\mathcal{S}_{tgt} = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N \times K}$ of a test episode $\mathcal{E}_{tgt} = \{(\mathcal{S}_{tgt}, \mathcal{Q}_{tgt}, \mathcal{T}_{tgt})\}$ built from target-domain data \mathcal{D}_{tgt} to make predictions on the query examples $\mathcal{Q}_{tgt} = \{\mathbf{x}\}$. \mathcal{T}_{tgt} denotes the type set of a test episode with a cardinality of N . It usually contains new types different from types in \mathcal{D}_{src} .

¹<https://github.com/microsoft/vert-papers/tree/master/papers/Decomposed-MetaSL>.

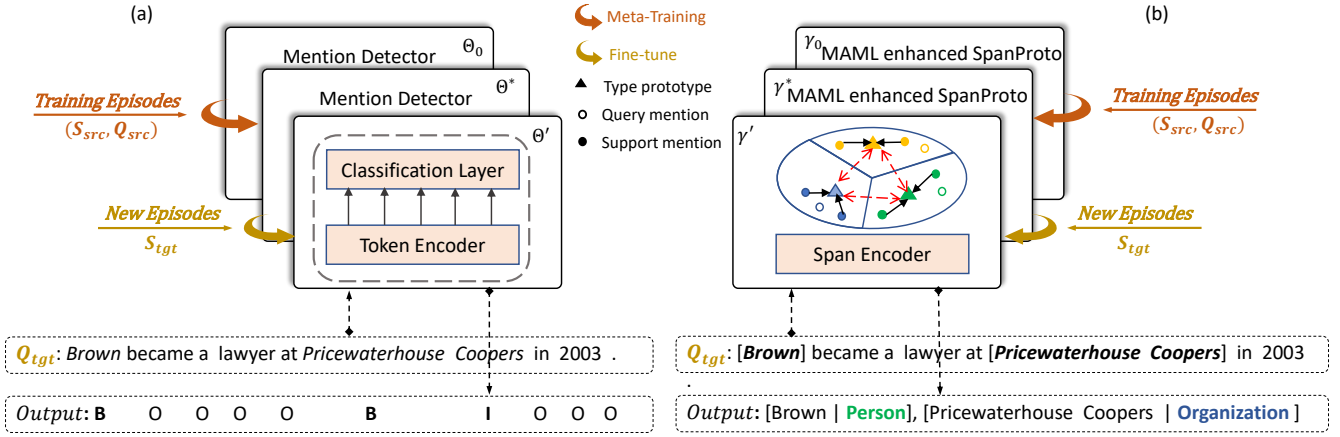


Fig. 1. Illustration of our decomposed meta-learning framework. Mention Detection identifies mention spans of interest via a sequence labeling model. Type Classification categorizes the mentions into predefined types with span-level ProtoNet. Both models are enhanced with MAML and adapted to new tasks via gradient updates on S_{tgt} .

III. METHODOLOGY

Figure 1 depicts the overall framework of our approach, which consists of two stages: i) mention detection, and ii) type classification.

A. Mention Detection

Mention detection aims to detect all mention spans in a query sentence belonging to a pre-defined target type set \mathcal{T} based on a support set \mathcal{S} . Note that our mention detector is type-agnostic, *i.e.*, we only detect the mention boundaries (*i.e.*, spans) regardless of their specific types. In this way, we expect to learn a model that captures common characteristics and patterns shared across different types and episodes, and hence benefits the few-shot transfer. Furthermore, we exploit model-agnostic meta-learning [13] (MAML) to fully explore information in support examples, so that the discrepancy between source episodes and target episodes could be mitigated. More specifically, MAML could prompt the mention detector to learn a good parameter initialization for adaptation, based on which the model can rapidly adapt to new episodes by a few gradient updates using only a few support examples without overfitting.

1) *Base Mention Detector*: In this paper, we employ a classic sequence labeling model as the base mention detector. The label set \mathcal{Y}^m can be any tagging scheme, such as BIO² or BIOES³.

a) *Token encoder*: Given an input sequence $\mathbf{x} = (x_1, \dots, x_L)$, the token representation h_i of x_i is produced by a contextualized encoder f_θ :

$$[h_1, \dots, h_L] = f_\theta(\mathbf{x}), \quad (1)$$

where θ denotes the parameters of the encoder to learn.

b) *Classification layer*: For each token x_i , its representation h_i is fed into a linear classification layer with the *softmax* function for prediction. The probability distribution over \mathcal{Y}^m can be formulated as:

$$p(x_i) = \text{softmax}(Wh_i + b), \quad (2)$$

where $p(x_i) \in \mathbb{R}^{|\mathcal{Y}^m|}$, W and b are learnable parameters.

c) *Training loss*: Here we train the base mention detector by minimizing the averaged cross-entropy loss over all tokens. To make the model focus more on the hard tokens, *i.e.*, tokens with higher losses, we additionally adopt a maximum loss term as in the work [17]:

$$\mathcal{L}_m(\Theta) = \frac{1}{L} \sum_{i=1}^L \text{CrossEntropy}(y_i^m, p(x_i)) + \lambda \max_{i \in \{1, 2, \dots, L\}} \text{CrossEntropy}(y_i^m, p(x_i)), \quad (3)$$

where $\Theta = \{\theta, W, b\}$, $\mathbf{y}^m = (y_1^m, \dots, y_L^m)$ denotes the label sequence for the mention detection model *w.r.t.* \mathbf{x} , $y_i^m \in \mathcal{Y}^m$, and $\lambda \geq 0$ is a weighting factor.

d) *Inference*: we use the Viterbi decoding algorithm [18] to decode the label sequence during inference. For simplicity, we do not learn the label transition matrix but simply forbid the invalid label transitions such as an Outside-Inside transition, and set an equal transition probability for all valid label transitions.

2) *MAML Enhanced Mention Detector*: The goal of MAML is to train an initialization of parameters Θ so that the meta-trained model could quickly adapt to a new episode ($S_{tgt}, Q_{tgt}, \mathcal{T}_{tgt}$) via only a few gradient updates on S_{tgt} .

Particularly, in the meta-training phase, given a sampled training episode ($S_{src}^{(i)}, Q_{src}^{(i)}, \mathcal{T}_{src}^{(i)}$), the model first performs gradient updates on $S_{src}^{(i)}$, called *inner-update*:

$$\Theta'_i = \mathbf{U}_n(\mathcal{L}_m^{S_{src}^{(i)}}(\Theta), \alpha), \quad (4)$$

where \mathbf{U}_n denotes n -step gradient descent updates via $\mathcal{L}_m^{S_{src}^{(i)}}(\Theta)$, *i.e.*, $\mathcal{L}_m(\Theta)$ calculated on $S_{src}^{(i)}$, with an inner-loop learning rate α .

²{Begin, Inside, Outside}

³{Begin, Inside, Outside, End, Single}

Then, Θ' from inner-update is evaluated on the query set $\mathcal{Q}_{src}^{(i)}$ and the model Θ is finally optimized over multiple episodes with the *meta-objective*:

$$\min_{\Theta} \sum_i \mathcal{L}_m^{\mathcal{Q}_{src}^{(i)}}(\Theta'_i). \quad (5)$$

Since Eq. (5) involves the computationally expensive second-order derivative, we simply use its first-order approximation as [13] to perform the *meta-update*:

$$\Theta \leftarrow \Theta - \beta \sum_i \nabla_{\Theta'_i} \mathcal{L}_m^{\mathcal{Q}_{src}^{(i)}}(\Theta'_i), \quad (6)$$

where β is the learning rate.

In the meta-testing phase, we adapt the meta-trained model to a new episode $(\mathcal{S}_{tgt}, \mathcal{Q}_{tgt}, \mathcal{T}_{tgt})$ by first performing Eq. (4) on \mathcal{S}_{tgt} , then evaluating on \mathcal{Q}_{tgt} .

B. Type Classification

The type classification model is expected to assign a type $t \in \mathcal{T}$ for each span predicted by the mention detection model. Here we take ProtoNet [19] as our base model. But differently from how it is employed in previous work [6], [8], [12], our ProtoNet is **span-level** metric learning based; **without modeling the miscellaneous Other prototype**. To further facilitate more effective adaptation to new episodes for the learned span-level ProtoNet, we propose a MAML enhanced Span-level ProtoNet to make the learned embedding space more distinguishable by better leveraging the information in support examples.

1) Span-Level ProtoNet:

a) *Mention representation*: For an input sequence $\mathbf{x} = (x_1, \dots, x_L)$, we first compute contextualized representations for each token:

$$[h_1, \dots, h_L] = f_{\gamma}(\mathbf{x}), \quad (7)$$

where γ denotes the learnable parameters of the encoder.

For a mention $x_{[i,j]}$ that starts at index i and ends at index j , its representation $m_{[i,j]}$ is calculated as:

$$m_{[i,j]} = \frac{1}{j-i+1} \sum_{k=i}^j h_k. \quad (8)$$

b) *Prototype representation*: Given a support set \mathcal{S} of an episode $(\mathcal{S}, \mathcal{Q}, \mathcal{T})$, we compute the prototype representation of each type $t_k \in \mathcal{T}$ by averaging the representations of all mentions belong to the type t_k in \mathcal{S} :

$$c_k = \frac{1}{|\mathcal{M}_{\mathcal{S}}(t_k)|} \sum_{x_{[i,j]} \in \mathcal{M}_{\mathcal{S}}(t_k)} m_{[i,j]}, \quad (9)$$

where $\mathcal{M}_{\mathcal{S}}(t_k)$ denotes all mentions in \mathcal{S} corresponding to the type t_k .

c) *Inference*: Given a mention $x_{[i,j]}$, we predict its type distribution by computing the distances between the mention representation and each prototype representation c_k :

$$p(t = t_k | x_{[i,j]}) = \frac{e^{-d(m_{[i,j]}, c_k)}}{\sum_{t_{k'} \in \mathcal{T}} e^{-d(m_{[i,j]}, c_{k'})}}, \quad (10)$$

where $d(\cdot, \cdot)$ denotes the distance function.

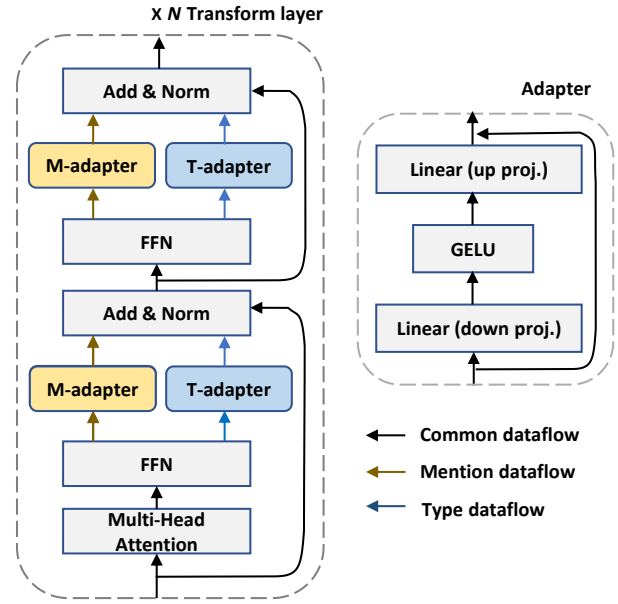


Fig. 2. Architecture of transformer layers with task-specific adapters.

d) *Meta-training*: Given an episode $(\mathcal{S}_{src}, \mathcal{Q}_{src}, \mathcal{T}_{src})$ sampled from \mathcal{D}_{train} , we first compute a prototype representation c_k for each $t_k \in \mathcal{T}_{src}$ on the support set \mathcal{S}_{src} via Eq. (9). Then update the model by minimizing the cross-entropy loss $\mathcal{L}_t(\gamma)$ on the query set \mathcal{Q}_{src} :

$$\mathcal{L}_t^{\mathcal{Q}_{src}}(\gamma) = \sum_{(x_{[i,j]}, t_k) \in \mathcal{Q}_{src}} -\log p(t = t_k | x_{[i,j]}). \quad (11)$$

e) *Meta-testing*: Given a new episode $(\mathcal{S}_{tgt}, \mathcal{Q}_{tgt}, \mathcal{T}_{tgt})$, we utilize the support set \mathcal{S}_{tgt} to obtain prototype representations for all types in \mathcal{T}_{tgt} , as in meta-training. For each example, in \mathcal{Q}_{tgt} , we first derive a set of mentions $\{x_{[i,j]}\}$ via the mention detection model, then assign the type label with the highest probability in Eq. (10) for each detected mention.

2) *MAML Enhanced SpanProto*: We further apply MAML to enhance the span-level ProtoNet. In the meta-training phase, given a sampled training episode $(\mathcal{S}_{src}, \mathcal{Q}_{src}, \mathcal{T}_{src})$, we first perform the *inner-update* on the support set $\mathcal{S}_{src}^{(i)}$ by minimizing $\mathcal{L}_t^{\mathcal{S}_{src}^{(i)}}(\gamma)$:

$$\gamma'_i = \mathbf{U}_n(\mathcal{L}_t^{\mathcal{S}_{src}^{(i)}}(\gamma), \alpha). \quad (12)$$

We recompute prototype representations with γ' and then evaluate γ' on the query set $\mathcal{Q}_{src}^{(i)}$ for *meta-update*. By aggregating multiple training episodes and employing first-order approximation, *meta-update* can be formulated as:

$$\gamma \leftarrow \gamma - \beta \sum_i \nabla_{\gamma'_i} \mathcal{L}_t^{\mathcal{Q}_{src}^{(i)}}(\gamma'_i), \quad (13)$$

where β is the learning rate.

For the meta-testing phase, we first update the meta-trained model on the support set \mathcal{S}_{tgt} with Eq. (12) when given a test episode $(\mathcal{S}_{tgt}, \mathcal{Q}_{tgt}, \mathcal{T}_{tgt})$, then perform evaluation on the query set \mathcal{Q}_{tgt} .

C. Joint Model vs. Separate Model

We explore two model implementations within the proposed framework: i) A separate model that uses two different BERT encoders for the two stages and thus captures sub-task-specific knowledge in each encoder; and ii) A joint model that shares the same BERT model for the two stages so that the model parameters and computation cost can be largely reduced. For the latter, to further maintain the task-specific knowledge, we propose to insert two groups of adapters [20] into the upper layers of the shared encoder for the two subtasks, respectively. As shown in Fig. 2, we adopt an M-adapter for the mention detection step and a T-adapter for the type classification step. Both adapters use the same structure that consists of two feed-forward layers for down-projection and up-projection, respectively, and takes a nonlinear function in between. They also contain a skip connection. Assume that the adapters are added to the transformer layers from the L-th layer to the uppermost layer, the hidden representations obtained from the L-th layer can be shared for the two subtasks and only need to be computed once. Assume the shared BERT model, the M-adapter, and the T-adapter are parameterized by ϕ_0 , ϕ_m , and ϕ_t , the joint optimization loss function is:

$$\mathcal{L}(\{\phi_0, \phi_m, \phi_t\}) = \mathcal{L}_m(\{\phi_0, \phi_m\}) + \mathcal{L}_t(\{\phi_0, \phi_t\}). \quad (14)$$

IV. EXPERIMENTS

A. Evaluation Tasks

We aim to tackle few-shot sequence labeling tasks that have the following characteristics: (1) they involve a large number of types, such as diverse entity or event types; (2) they require models to generalize to new types with few examples in practice. Following previous work [8], [21], we choose named entity recognition and slot filling as the representative few-shot sequence labeling tasks, and additionally include event detection task for more comprehensive evaluation. We also include POS tagging task which may have different tagsets in different scenarios. POS tagging is a special case where the mentions are predefined as single tokens in the sentence and there is no `Other` label that varies meaning across tasks. For this task, we apply our framework by omitting the mention detection stage and use the MAML-enhanced ProtoNet to fully exploit the information in support set to mitigate the large domain gap issue.

B. Experimental Setup

1) *Datasets*: We conduct experiments on nine public datasets involving the four sequence labeling tasks mentioned above. Table I summarizes the dataset statistics.

Few-NERD: a large-scale human-annotated few-shot NER dataset constructed from Wikipedia [16]. Following the work [16], we consider two evaluation settings: i) *inter*, where distinct fine-grained types in the train/dev/test splits may share coarse-grained types; and ii) *intra*, where the train/dev/test splits are built from different coarse-grained types.

Cross-Dataset: we combine four NER datasets from different domains: CoNLL-2003 [22] (news), GUM [23] (Wiki), WNUT-2017 [24] (social media), and Ontonotes [25] (mixed).

Following the previous work [6], episodes used for meta-training and meta-testing are constructed from the different datasets, *i.e.*, different domains.

SNIPS: a public benchmark dataset for slot filling [26]. It contains user utterances with slot tag annotations from seven domains: weather (We), music (Mu), playList (Pl), book (Bo), search screen (Se), restaurant (Re), and creative work (Cr).

FewEvent: a large-scale event detection dataset [7] constructed from ACE-2005⁴, TAC-KBP-2017 Event Track Data⁵, and some external resources such as Wikipedia.

WSJ: a collection of articles from Wall Street Journal (WSJ) with POS tagging annotations from the English Penn Treebank (PTB) corpus⁶ [27]. The PTB-style tagset uses 45 POS tags, while the Universal POS tagset [28], [29] has 12 POS tags.

Twitter: a human-annotated POS tagged corpus of tweets [30]. It uses a POS tagset of 25 tags.

TABLE I
STATISTICS OF EVALUATION DATASETS.

Dataset	Domain	# Sentences	# Types
Few-NERD [16]	Wikipedia	188.2k	66
CoNLL03 [22]	News	20.7k	4
GUM [23]	Wiki	3.5k	11
WNUT17 [24]	Social	5.6k	6
Ontonotes [25]	Mixed	159.6k	18
SNIPS [26]	7 domains	14.4k	39
FewEvent [7]	Mixed	70.8k	100
WSJ [27]	News	43.9k	45/12
Twitter [30]	Social	1.8k	25

2) *Episode Construction*: Following [6], [12], [16], here we evaluate our approach under two few-shot settings.

N-way K-shot: For Few-NERD and FewEvent, we split the dataset into train/dev/test without type overlap. Specifically, in FewEvent, there are N types in each episode, *i.e.*, N-way, and the support set contains K annotated mentions for each type, *i.e.*, K-shot [12]. In our experiments, we use the same data splits as in [12] for a fair comparison. In Few-NERD, it is a little different in that each type has $K \sim 2K$ annotated entities in the support set as one sentence may contain several entities [16]. For fairer comparison, we directly use the episodes released by the work [16] for training and evaluation.⁷

Cross Domain K-shot: For Cross-Dataset and SNIPS, we perform the cross-domain evaluation as [6]. When choosing one domain for evaluation, all other domains except the domain for validation are used for training. For POS tagging, we evaluate it in a coarse-to-fine setting that transfers from universal POS tagset to Twitter tagset or PTB tagset. According to [6], the support set of one K-shot episode from the domain D_j must satisfy the following constraints: each label in the domain D_j has at least K labeled examples, and dropping any sentence from the support set will lead to at least

⁴<http://projects ldc.upenn.edu/ace/>

⁵<https://tac.nist.gov/2017/KBP/Event/index.html>

⁶<https://catalog ldc.upenn.edu/LDC99T42>

⁷<https://cloud.tsinghua.edu.cn/f/56fb277d3fd2437a8ee3/?dl=1>, which is different from data used in our ACL paper.

TABLE II

F1 SCORES WITH STANDARD DEVIATIONS ON FEW-NERD. THE BEST RESULTS ARE IN **BOLD** AND THE SECOND HIGHEST RESULTS ARE UNDERLINED.

Models	Intra					Inter				
	1~2-shot		5~10-shot		Avg.	1~2-shot		5~10-shot		Avg.
	5 way	10 way	5 way	10 way		5 way	10 way	5 way	10 way	
ProtoNet [†]	20.76±0.84	15.04±0.44	42.54±0.94	35.40±0.13	28.44	38.83±1.49	32.45±0.79	58.79±0.44	52.92±0.37	45.75
NNShot [32] [†]	25.78±0.91	18.27±0.41	36.18±0.79	27.38±0.53	26.90	47.24±1.00	38.87±0.21	55.64±0.63	49.57±2.73	47.83
StructShot [32] [†]	30.21±0.90	21.03±1.13	38.00±1.29	26.42±0.60	28.92	51.88±0.69	43.34±0.10	57.32±0.63	49.57±3.08	50.53
CONTAINER [33]	40.43	33.84	53.70	47.49	43.87	55.95	48.35	61.83	57.12	55.81
ESD [8]	36.08±1.6	30.00±0.70	52.14±1.5	42.15±2.6	40.09	59.29±1.25	52.16±0.79	69.06±0.80	64.00±0.43	61.13
Ours-joint (BIO)	46.19±0.69	39.59±0.42	61.63±0.80	55.21±0.43	50.66	59.47±0.55	53.65±0.59	70.55±0.16	67.15±0.27	62.71
Ours (BIO)	48.62±0.13	42.25±0.32	63.07±0.53	56.49±0.15	52.61	61.12±0.58	54.89±0.29	69.61±0.15	66.04±0.08	62.92
Ours-joint (BIOES)	47.87±0.32	40.84±0.19	62.87±0.37	56.49±0.33	52.02	<u>61.34±0.82</u>	<u>54.96±0.48</u>	71.26±0.15	67.85±0.18	<u>63.85</u>
Ours (BIOES)	49.90±0.33	43.03±0.29	64.36±0.20	57.58±0.26	53.72	62.09±0.93	55.61±0.32	<u>70.79±0.09</u>	<u>67.44±0.19</u>	63.99

[†] denotes the results reported in [16].

one label to appear less than K times. For Cross-Dataset and SNIPS, we evaluated under 1-shot and 5-shot settings, and use sampled episodes provided by [6] for a fair comparison. For POS tagging, we only evaluate under the 1-shot setting since there exist tags that occur less than 5 times in the Twitter dataset.

3) *Evaluation Metric*: For Few-NERD and FewEvent, we report the micro-F1 over all meta-testing episodes, as in [12] and [16]. For Cross-Dataset and SNIPS, we record the micro-F1 of each meta-testing episode and report the average micro-F1 of all episodes as [31]. For POS tagging, we report the accuracy scores. All experimental results are reported as means and standard deviations from 5 runs with different random seeds.

4) *Implementation Details*: We implement our approach based on *HuggingFace* Transformers.⁸ For all experiments, we utilize the `BERT-base-uncased` model [34] for contextualized token representations as [6], [12], [16]. We adopt the most commonly used BIO tagging scheme as previous work [6], [12]. Besides, we also report the results with BIOES since it can better model the boundary information. We use AdamW [35] with a learning rate of $5e-5$ for outer-loop optimization. For all experiments, the learning rate for the inner-loop is set to $2e-5$, the dropout rate is set to 0.5, and the max-loss weight λ is set to 2. The model is meta-trained on 1,000 meta batches and each batch contains 16 episodes. The hidden size of both adapters in the joint model is set to 64. We add task-specific adapters from the 8th transformer layer to the uppermost layer. For mention detection, we fine-tune the meta-trained detector for 3 steps in the Few-NERD inter 1~2 setting and 30 steps in the other settings. For type classification, we fine-tune the meta-trained ProtoNet for 3 steps in in-domain settings (*i.e.*, experiments on Few-NERD and FewEvent) and 20 steps in cross-domain settings.⁹ Following the previous works [6], [7], [16], we perform validation on the dev set and select the checkpoint with the best validation performance.

⁸<https://github.com/huggingface/transformers>⁹The number of steps was set empirically per dataset and their settings on the dev set. Compared with our prior work [15], we omit the post-processing step with a confidence threshold in the mention detection step for higher robustness.

C. Evaluation on the NER Task

1) *Baselines*: We compare our method with the following representative few-shot NER baselines.

Token-level fine-tuning based models: **TransferBERT** [6], which pre-trains a BERT-based sequence tagger on training data and then fine-tunes the model on the few-shot examples for the new task.

Token-level metric learning models: **ProtoNet** [16] is a token-level prototypical network model. **L-TapNet+CDT** [6] uses label names and a collapsed dependency transfer mechanism to enhance the basic TapNet [36] model. **NNshot** [32] is a simple token-level nearest neighbor classifier, which learns a token embedding model via pretraining on training data. **Structshot** [32] enhances NNShot by applying a Viterbi decoder on abstract tags. **CONTAINER** [33] uses contrastive learning to minimize the inter-token distance.

Span-level metric learning models: **ESD** [8] formulates the task into a span-level matching problem and exploits span representations and prototype representations to solve the task.

2) *Performance Comparison*: Table II shows the results on Few-NERD. We can see that our method outperforms all baselines in all settings. More interestingly, our approach shows significant superiority under the more challenging *intra* setting, for example, improving about 14.3 F1 points over the previous state-of-the-art ESD in the 10-way 5~10-shot setting. Table III demonstrates the results of both cross-domain 1-shot and 5-shot settings. It shows that, when facing a large domain gap, our approach achieves substantial improvement compared with the previous SOTA method L-TapNet+CDT - an 18.1 F1 point improvement on Mixed domain under the 1-shot setting. The results on Few-NERD and Cross-Dataset also convincingly validate the effectiveness of our method in the few-shot NER task, especially when dealing with the huge transfer gaps.

D. Evaluation on the Slot Filling Task

1) *Baselines*: Besides the fine-tuning baseline **TransferBERT** and the metric-learning baselines either at token-level (*i.e.*, **ProtoNet**, **L-TapNet+CDT**) or at span-level (*i.e.*, **ESD**), we include the following baselines for few-shot slot tagging:

TABLE III
F1 SCORES WITH STANDARD DEVIATIONS ON CROSS-DATASET.

Models	1-shot					5-shot				
	News	Wiki	Social	Mixed	Avg.	News	Wiki	Social	Mixed	Avg.
TransferBERT [‡]	4.75±1.42	0.57±0.32	2.71±0.72	3.46±0.54	2.87	15.36±2.81	3.62±0.57	11.08±0.57	35.49±7.60	16.39
ProtoNet [‡]	32.49±2.01	3.89±0.24	10.68±1.40	6.67±0.46	13.43	50.06±1.57	9.54±0.44	17.26±2.65	13.59±1.61	22.61
L-TapNet+CDT [6] [‡]	44.30±3.15	12.04±0.65	20.80±1.06	15.17±1.25	23.08	45.35±2.67	11.65±2.34	23.30±2.80	20.95±2.81	25.31
Ours-joint (BIO)	48.96±1.07	13.14±0.15	30.87±0.43	31.58±2.52	31.14	63.68±0.99	22.20±1.30	35.82±1.31	42.78±1.63	41.12
Ours	51.97±1.21	13.77±1.18	30.24±0.36	33.26±1.07	<u>32.31</u>	<u>63.58±1.38</u>	23.70±1.74	31.17±1.07	36.72±1.29	38.79
Ours-joint (BIOES)	<u>50.77±1.63</u>	<u>14.60±0.58</u>	<u>30.67±0.91</u>	34.45±1.00	32.62	63.14±0.60	<u>24.79±2.55</u>	<u>34.58±1.11</u>	44.27±0.93	41.70
Ours (BIOES)	49.17±1.01	15.03±1.39	28.75±0.40	<u>33.43±1.10</u>	31.59	61.92±1.21	26.94±1.82	28.64±0.55	40.63±1.12	39.53

[‡] denotes the results reported in [6].

TABLE IV
F1 SCORES WITH STANDARD DEVIATIONS ON SNIPS ON 1-SHOT SETTING AND 5-SHOT SETTING.

Models	We	Mu	Pl	Bo	Se	Re	Cr	Avg.	
	1-shot	TransferBERT [‡]	55.82±2.75	38.01±1.74	45.65±2.02	31.63±5.32	21.96±3.98	41.79±3.81	38.53±7.42
ProtoNet [‡]		46.72±1.03	40.07±0.48	50.78±2.09	68.73±1.87	60.81±1.70	55.58±3.56	67.67±1.16	55.77
L-TapNet+CDT [6]		71.53±4.04	60.56±0.77	66.27±2.71	84.54±1.08	76.27±1.72	70.79±1.60	62.89±1.88	70.41
BERT-MRC [37]		-	-	-	-	-	-	-	69.3
ESD [8]		78.25±1.50	54.74±1.02	71.15±1.55	71.45±1.38	67.85±0.75	71.52±0.98	78.14±1.46	70.44
Ours-joint (BIO)		78.25±0.91	63.79±1.40	70.75±0.68	82.85±0.69	71.44±1.55	74.75±0.73	70.00±3.14	73.12
Ours (BIO)		<u>78.57±0.51</u>	66.20±0.85	69.89±0.91	82.77±0.46	74.25±0.94	75.79±0.31	70.47±1.19	74.00
Ours-joint (BIOES)		78.42±0.67	64.23±1.06	<u>71.71±1.05</u>	83.09±0.93	73.90±1.60	76.90±0.37	<u>75.61±0.92</u>	<u>74.84</u>
Ours (BIOES)	78.70±0.61	<u>65.54±0.57</u>	72.56±1.04	<u>83.81±0.18</u>	<u>75.42±0.58</u>	<u>76.62±0.32</u>	71.55±2.02	74.89	
5-shot	TransferBERT [‡]	59.41±0.30	42.00±2.83	46.07±4.32	20.73±3.36	28.20±0.29	67.75±1.28	58.61±3.67	46.11
	ProtoNet [‡]	67.82±4.11	55.99±2.24	46.02±3.19	72.17±1.75	73.59±1.60	60.18±6.96	66.89±2.88	63.24
	L-TapNet+CDT [6]	71.64±3.62	67.16±2.97	75.88±1.51	84.38±2.81	82.58±2.12	70.05±1.61	73.41±2.61	75.01
	Retriever [9]	82.95	61.74	71.75	81.65	73.10	79.54	51.35	71.72
	BERT-MRC [37]	89.39	75.11	77.18	84.16	73.53	82.29	72.51	79.17
	ConVEx [38]	71.5	77.6	79.0	84.5	84.0	73.8	67.4	76.8
	GPT-Inverse [31]	70.63	71.97	78.73	87.34	81.95	72.07	72.44	76.73
	ESD [8]	84.50±1.06	66.61±2.00	79.69±1.35	82.57±1.37	82.22±0.81	80.44±0.88	81.31±1.84	79.59
	Ours-joint (BIO)	89.45±0.34	77.58±0.85	82.66±0.84	<u>87.39±0.47</u>	85.17±1.29	83.68±0.70	77.05±0.50	83.28
	Ours (BIO)	89.44±0.31	79.04±0.66	80.82±0.65	87.19±0.57	86.63±0.79	84.98±0.32	75.50±0.98	83.37
	Ours-joint (BIOES)	89.29±0.51	76.91±0.89	84.13±0.73	86.93±0.62	85.87±0.51	83.98±0.56	<u>78.91±0.63</u>	<u>83.72</u>
Ours (BIOES)	89.83±0.71	<u>78.48±0.40</u>	<u>83.92±0.81</u>	88.18±0.43	87.89±0.34	85.33±0.40	78.17±0.19	84.54	

[‡] denotes the results reported in [8]. “-” denotes no result reported in the original paper.

Span-level retriever model: **Retriever** [9] trains a span-level retriever and fine-tunes it on labeled examples for a new task. A decoding algorithm is further proposed to handle the overlapped span labels.

Span extraction based models: These methods generally train a span extraction model on source domains and then fine-tune it on the labeled data of target task. **BERT-MRC** [37] trains a BERT-based machine reading comprehension model. **ConVEx** [38] pre-trains the slot labeler as pairwise cloze task on Reddit data. **GPT-Inverse** [31] utilizes GPT2 [39] and constructs prompts by slot names to generate slot values.

2) *Performance Comparison*: Table IV summarizes the results for 1-shot and 5-shot slot tagging on SNIPS. It shows that our approach achieves the highest averaged F1 among all baseline methods under both 1-shot and 5-shot settings. Compared with ConVEx, which utilizes large-scale Reddit data for pretraining, our method shows better performance on each domain in 5-shot setting without additional pretraining. Moreover, our method surpasses all span-level one-stage methods (*i.e.*, ESD, BERT-MRC, Retriever, GPT-Inverse), well demonstrating the effectiveness and necessity of the decom-

position framework and the adoption of meta-learning.

E. Evaluation on the Event Detection Task

1) *Baselines*: In addition to **L-TapNet+CDT** and **ProtoNet**, we compare our method with the following baselines for evaluation in few-shot event detection:

Token-level fine-tuning based model: Cong *et al.* [12] pre-train and fine-tune a standard event detection model **PLMEE** [40] as a baseline for few-shot event detection.

Two-stage models: **LoLoss** [12], [41] and **DMBPN** [7] sequentially address event trigger identification and few-shot event type classification. Both of them use a BERT-based sequence tagger for event trigger identification. For few-shot event type classification, LoLoss employs the model proposed by Lai *et al.* [41] which enhances the prototypical network with matching information in the support set. DMBPN [7] designs a memory-based prototypical network. For LoLoss-Separate, the two sub-tasks use two different BERT encoders. For LoLoss-MultiTask, the two sub-tasks share one BERT encoder.

Token-level metric-learning models: **PA-CRF** [12] enhances the token-level ProtoNet by leveraging label prototypes to estimate the label transition matrix. **HCL-TAT** [42] improves token-level ProtoNet by applying support-support contrastive learning and prototype-query contrastive learning to learn a more discriminative representation space, and adopts a similarity threshold to deal with the label assignment of the `Other` type.

TABLE V
F1 SCORES WITH STANDARD DEVIATIONS ON FEWEVENT.

Models	5-shot		10-shot		Avg.
	5 way	10 way	5 way	10 way	
PLMEE [‡]	4.43±0.19	2.52±0.28	4.69±0.85	2.76±0.55	3.60
LoLoss-Separate [‡]	30.14±0.30	29.33±0.40	30.91±0.29	30.08±0.39	30.12
LoLoss-MultiTask [‡]	31.51±1.56	30.46±1.38	31.70±1.21	30.32±0.89	31.00
DMBPN [7] [‡]	37.51±2.60	34.21±1.45	38.14±2.32	35.31±1.69	36.29
ProtoNet [‡]	58.82±0.88	55.04±1.62	61.01±0.23	58.78±0.88	58.41
L-TapNet+CDT [6] [‡]	59.30±0.23	56.41±1.09	62.77±0.12	59.44±1.83	59.48
PA-CRF [12] [‡]	62.25±1.42	58.48±0.68	64.45±0.49	61.64±0.81	61.71
HCL-TAT [42]	66.96±0.70	64.19±0.96	68.80±0.85	66.00±0.81	66.49
Ours-joint (BIO)	70.97±0.84	67.06±2.16	74.39±1.03	71.17±0.55	70.90
Ours (BIO)	69.92±1.14	65.26±1.18	74.54±0.23	68.26±0.97	69.50
Ours-joint (BIOES)	72.20±1.03	67.64±1.41	75.00±1.07	72.06±1.14	71.73
Ours (BIOES)	70.09±0.57	65.16±0.86	74.09±0.58	69.12±0.49	69.62

[‡] denotes the results reported in [12].

2) *Performance Comparison*: Table V reports the results of different approaches on FewEvent. We can see that our method achieves the best performance over all settings, which well verifies our system's generalization ability on the few-shot event detection task. Contrary to the observations in [12] and [42] that the unified one-stage paradigm works better than the two-stage paradigm due to error propagation in few-shot event detection, Table V reveals that our two-stage method developed based on task decomposition and meta-learning obtains superior performance to previous unified/one-stage methods.

F. Evaluation on POS Tagging Task

1) *Baselines*: Here we compare our method with three commonly used baselines for few-shot classification: a fine-tuning baseline **TransferBERT** and two metric-learning baselines (**ProtoNet** and **NNShot**).

2) *Performance Comparison*: Table VI shows the effectiveness of our framework on few-shot POS tagging tasks. Our method surpasses the best results of baselines by 1.10 and 2.48 on WSJ and Twitter, respectively. This well demonstrates the importance of exploiting meta-learning algorithm to fully utilize the support examples, especially in the large domain gap scenario, *i.e.*, transferring from News to Twitter domain.

V. ANALYSIS

A. Ablation Study of Main Strategies

Here we introduce four variants of our method to gain a deeper understanding of the strategies employed in our approach: a) *Ours w/o MAML mention*, which removes the MAML algorithm used in mention detection and directly adapt the pre-trained mention detector. b) *Ours w/o MAML type*, which removes MAML algorithm used in type classification

TABLE VI
ACCURACY SCORES WITH STANDARD DEVIATIONS ON POS TAGGING.

Models	1-shot	
	WSJ	Twitter
TransferBERT	66.22±1.11	68.44±0.37
ProtoNet	90.68±0.39	77.42±0.33
NNShot	89.75±0.46	78.53±0.19
Ours	91.78±0.21	81.01±0.15

and becomes the basic SpanProto in the second stage. c) *Ours w/o MAML* removes MAML used in both mention detection and type classification. d) *Ours w/o Task Decomposition*, which eliminates the mention detection stage and thus degrades into a MAML enhanced token-level ProtoNet with modeling of the `Other` prototype.

Table VII shows the ablation results of Few-NERD 5-way 1~2-shot.

TABLE VII
ABLATION RESULTS (F1) OF FEW-NERD 5-WAY 1~2-SHOT.

	<i>inter</i>	<i>intra</i>
Ours	61.12	48.62
a) Ours w/o MAML mention	55.58	34.72
b) Ours w/o MAML type	60.86	47.32
c) Ours w/o MAML	55.22	33.88
d) Ours w/o Task Decomposition	44.63	28.60

Comparing *Ours* with *Ours w/o MAML*, *Ours w/o MAML type*, and *Ours w/o MAML mention*, we observe that removing MAML from our method at any stage will lead to a significant performance drop, especially for the more challenging *intra* setting with a larger domain gap. This suggests the importance of our MAML enhancement in mitigating the large domain gap. We can observe that removing MAML from the mention detection model leads to a larger drop than *Ours w/o MAML type*, indicating that the task discrepancy for mention detection is larger than for type classification. Table VII also indicates that *Ours* outperforms *Ours w/o Task Decomposition* with a huge margin, *i.e.*, 16.49 F1 on *inter* and 20.02 F1 on *intra*, fully demonstrating the necessity of the task decomposition framework in our method.

B. Analysis on Mention Detection

In this subsection, we dive into the mention detection model and introduce the following baselines for detailed analysis:

a) *Ours w/o MAML-training*, which replaces the meta-training procedure for the mention detection with a conventional supervised training but keeps the fine-tuning on support examples for the new episodes, and thus degenerates into the typical pretraining-finetuning paradigm. b) *Ours w/o MAML*, which cuts out the whole MAML procedure (episodic training and fine-tuning) from our method. In other words, we directly apply a pre-trained mention detector to a new task. c) *ProtoNet*, which utilizes the token-level similarities between support tokens and query tokens to do classification.

TABLE VIII
CASE STUDY OF MENTION DETECTION ON FEW-NERD INTRA. THE PREDICTED MENTION SPANS OF DIFFERENT METHODS ARE SHOWN.

Training-Support S_{src}	It was added to the <i>New South Wales State Heritage Register</i> on 20 June 2008.
New-Support S_{tgt}	It operates the <i>Keio Plaza Hotel</i> properties in Tokyo...
New-Query Q_{tgt}	<i>Langham Hotel</i> was listed on the Queensland Heritage Register on 21 October...
Ours	<i>Langham Hotel</i>
Ours w/o MAML-training	<i>Langham Hotel</i> , <i>Queensland Heritage Register</i>
Ours w/o MAML	<i>Queensland Heritage Register</i>
ProtoNet	<i>Langham(I-hospital)</i> , <i>Hotel(I-hotel)</i> , <i>Queensland(I-eventprotest)</i> , <i>Heritage(I-eventother)</i> , <i>Register(I-eventprotest)</i>

The corrected mentions of new types are in **bold**.
Golden mentions of original types that appear in the meta-training data D_{src} are underlined.

TABLE IX
F1 SCORE OF MENTION DETECTION ON FEW-NERD (5-WAY 1~2-SHOT).

	<i>inter</i>	<i>intra</i>
Ours	69.51	68.33
a) Ours w/o MAML-training	68.39	68.07
b) Ours w/o MAML	63.49	48.59
c) ProtoNet	46.91	37.49

We ignore the types of its output entities and only evaluate its performance for mention detection.

Table IX shows that all variants of our mention detector yield better performance than the one-stage *ProtoNet*. We conjecture that this is because token representations produced by *ProtoNet* are sub-optimal for mention detection. As illustrated by the case study in Table VIII, *ProtoNet* struggles to model the token-level label dependency and ignores the noun phrase characteristic compared with other methods. In contrast, our mention detector and its variants are designed to focus on this subtask and can fully explore the shared boundary knowledge across different tasks regardless of their types, thus achieving more satisfying performance.

Table IX also indicates that both *Ours* and *Ours w/o MAML-training* outperform *Ours w/o MAML*, which emphasizes the importance of updating the model with support examples to reduce the train-test discrepancy and thus benefit few-shot transfer. We can use Table VIII for exemplification. Given a query sentence from a new task, *Ours w/o MAML* only predicts a false positive mention “Queensland Heritage Register” while missing the golden one “Langham Hotel” for the new target type. Note that “Queensland Heritage Register” is a mention belonging to the original types in training corpus D_{src} , implying that the pre-trained mention detector performs well on the seen types, but struggles to detect mentions of new types. Table VIII also shows that both *Ours* and *Ours w/o MAML-training* can successfully detect “Langham Hotel”. However, *Ours w/o MAML-training* still outputs “Queensland Heritage Register”, while our method can produce more accurate predictions. This shows that though fine-tuning can benefit pre-trained models on new mentions to some extent, it may present too much bias to the training data, which well demonstrates the necessity and effectiveness of our MAML enhancement to help the model find a better parameter initialization and quickly adapt to new tasks.

C. Analysis on Type Classification

Here we investigate the performance of different models on type classification **given a golden mention**. We design the following variants for a more thorough understanding of our type classification model: a) *Ours w/o MAML*, which removes the MAML enhancement from *Ours* and thus becomes the span-level *ProtoNet*. b) *Ours w/ token-level*, which differs from *Ours* only in the metric granularity: employs MAML to enhance *ProtoNet* at token-level instead of span-level. Assuming the target types are $\{\text{PER}, \text{LOC}\}$, the label space corresponding to this setting will be $\{\text{B-PER}, \text{I-PER}, \text{B-LOC}, \text{I-LOC}\}$. c) *Ours w/ token-level w/ Other*, which additionally models an *Other* prototype during the meta-training stage of the type classification model, *i.e.*, clustering all tokens that do not belong to any types around the *Other* prototype. Therefore, the corresponding label space will become $\{\text{B-PER}, \text{I-PER}, \text{B-LOC}, \text{I-LOC}, \text{Other}\}$.

Note that here we focus on the subtask of type classification and suppose access to golden mentions. For span-level methods, *i.e.*, *Ours* and *Ours w/o MAML*, we consider assigning type labels for each given golden mention. For token-level methods, *i.e.*, *Ours w/ token-level* and *Ours w/ token-level w/ Other*, we constrain the label decoding by employing the Viterbi algorithm to prohibit invalid label transitions, so that the predicted labels can be consistent with the golden boundaries.

Table X highlights the performance contributions of each strategy in our approach. Based on that, we can draw some in-depth observations as follows.

i) *Ours* achieves higher results than *Ours w/o MAML*, which implies that MAML can leverage the information in support examples to derive a more distinguishable type embedding space. Comparing Fig 3(a) with Fig 3(b) supports the same conclusion.

ii) *Ours* outperforms *Ours w/ token-level* by a large margin of 6.14 accuracy points in *inter* and 9.98 accuracy points in *intra*, indicating that span-level representation is much more effective than just token-level representation for type classification. Fig. 3 depicts the t-SNE [43] visualization of span-/token-level type embeddings sampled from different type classification models. Fig. 3(c) and Fig. 3(d) show that representations produced by token-level models (*i.e.*, *Ours w/ token-level* and *Ours w/ token-level w/ Other*) are less separable than those derived from span-level models (*i.e.*, *Ours* and *Ours w/o MAML*). We theorize that token-level repre-

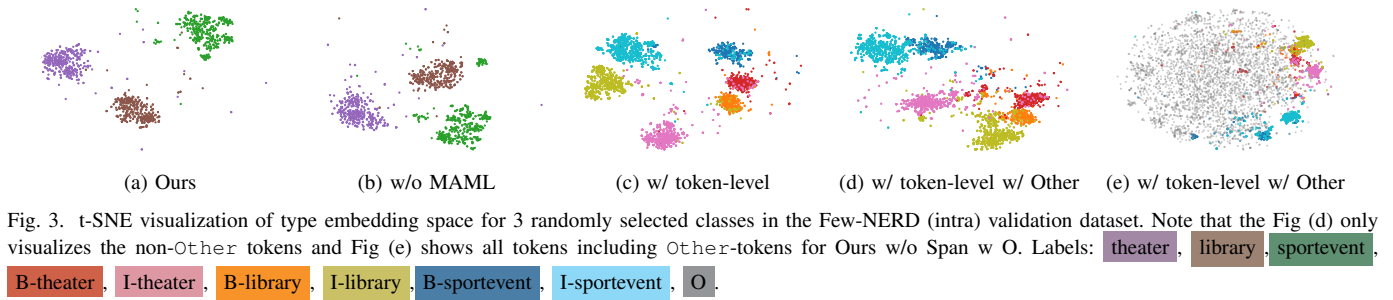


Fig. 3. t-SNE visualization of type embedding space for 3 randomly selected classes in the Few-NERD (intra) validation dataset. Note that the Fig (d) only visualizes the non-Other tokens and Fig (e) shows all tokens including Other-tokens for Ours w/o Span w O. Labels: theater, library, sportevent,

sentations have to model the boundary information and the type information simultaneously, which may result in tokens of different type labels but the same boundary label located close to each other (e.g., B-theater and B-library). By comparison, the span-level representations are developed to convey the complete semantic meaning for each mention and thus can predict its label with more comprehensive information than single tokens.

iii) *Ours w/ token-level* outperforms *Ours w/ token-level w/ Other*, which demonstrates that clustering all non-type tokens into a single *Other* prototype will hurt the performance of type classification. As shown in Fig. 3(e), non-type tokens do not have a unified semantic meaning and hence are spread across the embedding space. Comparing Fig 3(d) to Fig 3(c), we can see that enforcing the clustering of *Other* tokens in embedding space will destroy the representation distribution corresponding to target types. This observation clearly verifies the benefit of solving the miscellaneous *Other* issue via task decomposition.

TABLE X
ACCURACY OF TYPE CLASSIFICATION ON FEW-NERD (5-WAY 1~2-SHOT).

	metric	w/Other	MAML	inter	intra
Ours	span	no	yes	87.96	70.18
a) Ours w/o MAML	span	no	no	87.43	68.14
b) Ours w/ token-level	token	no	yes	81.82	60.20
c) Ours w/ token-level w/ Other	token	yes	yes	72.79	55.40

D. Analysis on Tagging Scheme

Table XI summarizes the results of our approach and token-level ProtoNet with different tagging schemes. We can observe that our method is more robust to the choice of tagging scheme compared to ProtoNet. Additionally, Table II~V show that our method can achieve better performance with the BIOES tagging scheme in general. We speculate that the BIOES scheme can provide more fine-grained boundary information for our model. However, token-level ProtoNet faces the difficulty of representing class prototypes with sparse and few examples especially when the label space is larger, e.g., using BIOES.

E. Efficiency of Joint Model

To evaluate the model efficiency of the joint model, we test the inference time of *Ours* and *Ours-joint* using the same

TABLE XI
F1 SCORES UNDER DIFFERENT TAGGING SCHEMES.

	scheme	inter	intra
Ours	BIOES	62.09	49.90
	BIO	61.12	48.62
	IO	60.56	48.34
ProtoNet [‡]	BIOES	10.55	6.83
	BIO	14.78	8.26
	IO	37.19	20.78

[‡] indicates our re-implementation.

hardware environment. Table XII shows the inference time and the number of model parameters.

TABLE XII
EVALUATION ON SNIPS-WE DOMAIN (1-SHOT). INFERENCE TIME: SECONDS /PER SENTENCE ON 4-CORE CPUs.

	#Params	Inference time	F1
Ours	220M	0.069	78.57
Ours-joint	111M	0.046	78.25

Comparing *Ours* with *Ours-Joint*, we can see that the joint model reduces the number of parameters to near half and speeds up the inference time up to 33.3% since the representation from the shared bottom layers can be reused without re-computation. This makes our method easily adopt to resource-constrained environments that have limited storage space and computation resource. Besides, the joint model achieves comparable performance on most datasets as shown in Table II~V. This also well demonstrates that the performance gain of our method comes from the proposed framework, not from more parameters.

F. Analysis on Task-specific Adapters

To explore how much the models for the two sub-tasks can share their representations, we train different models with task-specific adapters attached from different layers of the transformer to the uppermost layer. We also add a baseline that removes the task-specific adapter modules from *Ours-joint* completely, denoted as “Shared”. Figure 4 shows the results.

The results show that adding adapters to the upper layers and sharing the lower encoder layers does not affect the performance much compared to adding adapters to every layer. However, adding adapters only to the last one or two

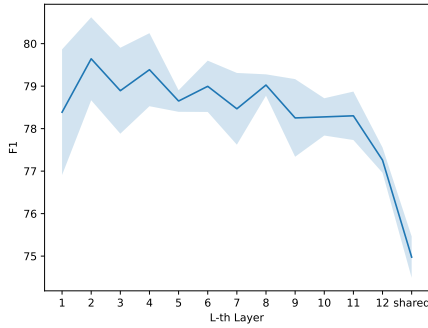


Fig. 4. F1 scores on SNIPS-We Domain by adding adapters from the L-th layer to the 12-th layer. “Shared” denotes a fully-shared encoder without adapters.

layers, or sharing all encoder layers, significantly reduces the performance. This demonstrates that the knowledge learned in upper layers is more task-specific and verifies the necessity to add task-specific adapters to model task-specific knowledge.

G. Influence Factors of Model Performance

1) *Difficulty of Few-shot Task*: We investigate the difficulty of few-shot tasks by analyzing the characteristics of target few-shot tasks ($\mathcal{S}_{tgt}, \mathcal{Q}_{tgt}$) via the metric QP-Dist. QP-Dist measures the average Euclidean distance between each query mention and its class prototype with pre-trained BERT model as the feature extractor. A larger QP-Dist means that the mentions with the same labels are more diverse in feature space and the class prototypes are less representative of them, making the few-shot task more difficult. In other words, as the task becomes more difficult, the QP-Dist tends to increase while the F1 score tends to decrease. Table XIII shows QP-Dist values along with F1-scores in different few-shot tasks. It suggests that few-shot NER is the most challenging task, while POS tagging is the easiest. We next delve into detailed characteristics of different tasks and datasets.

TABLE XIII
F1 SCORES OF OUR MODEL TRAINED ON FEW-SHOT EXAMPLES AND QUERY PROTOTYPE DISTANCE (QP-DIST) IN DIFFERENT FEW-SHOT SETTINGS.

Dataset	Setting	F1 score	QP-Dist
Few-NERD	5-way 1~2-shot (inter)	7.58	4011.4
Cross-Dataset	4-way 1-shot (News)	13.22	4619.6
SNIPS	5-way 1-shot (PI)	30.90	2595.2
FewEvent	5-way 5-shot	46.89	2771.3
Twitter	20-way 1-shot	61.29	167.4
WSJ	45-way 1-shot	74.84	147.5

- Tasks with more diverse sentence and mention expressions are more difficult than those with simpler ones. For example, NER datasets have more diverse entities and sentence expressions than SNIPS and FewEvent, thus have lower performance. In AddtoPlaylist (PI) domain of SNIPS, 89% of mentions of “playlist owner” slot type are “my” and 75.2% of sentences are in the format of “add ... to [playlist]_{slot}”. Similarly, event trigger words

in FewEvent are not as diverse as entities, with “arrest” accounting for 43.7% of mentions for the event type “Arrest-Jail”.

- Tasks that require fine-grained semantic understanding are more challenging. Among all the tasks, POS tagging is the easiest task since it does not involve mention detection and is less sensitive to contextual semantic than the others.
- Tasks that handle noisier informal text pose more challenges than those that use formal text, as shown by the lower performance of the few-shot POS tagging on the Twitter dataset than on the WSJ News dataset in Table XIII, despite the smaller classification space of the former.
- Tasks with less shots and larger classification ways are harder. For instance, Table IV shows that our model gains 11.13 F1 points from 1-shot to 5-shot on the We domain, while Table II shows that 10-way tasks are harder than 5-way tasks under the same transfer setting.

2) *Difficulty of Knowledge Transfer*: Here we discuss the knowledge transfer from meta-train tasks to meta-test tasks.

- The task similarity between meta-train and meta-test episodes affects the difficulty of knowledge transfer. Cross-Dataset is the most challenging setting due to the text domain shift and the changing meanings of the Other label. Unlike SNIPS, FewNERD, and FewEvent, where the meta-train and meta-test tasks come from the same dataset, Cross-Dataset transfers between different text domains from different datasets. Moreover, few-shot NER faces more challenges from the varying meanings of the Other label across tasks. The entity tokens in the meta-train tasks can be O tokens in the meta-test tasks, and vice versa. For example, in Cross-Dataset, both personal pronouns and person names are tagged as PERSON in the meta-train tasks, but only names are tagged as PERSON in the meta-test tasks. However, this phenomenon is less common in SNIPS and FewEvent, because one sentence usually expresses only one event in FewEvent and the short utterances in one intent domain (e.g., GetWeather) in SNIPS are less likely to mention a slot from another intent (e.g., PlayMusic).
- More meta-training episodes facilitate the knowledge transfer. As shown in Fig. 5, more meta-training episodes lead to the better performance on meta-testing tasks.

H. Error Analysis

The decomposition framework enables us to easily analyze the bottleneck of our system. Table XIV reports the results under the most challenging setting of each dataset. As shown in Table XIV, mention detection is still one of the main problems in the current system for all tasks. Besides, type classification remains challenging; especially in the NER task, which needs more fine-grained contextual information. In fact, within our decomposition framework, the boundary information provided by the mention detector allows exploring more powerful mention representations for more expressive span-level prototypes. We leave this for future work.

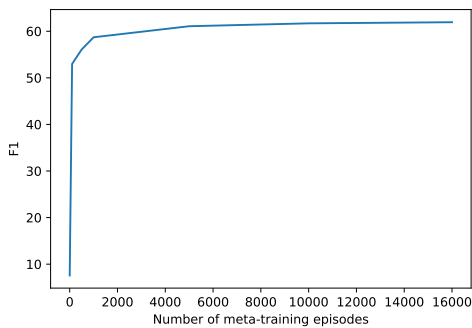


Fig. 5. F1 scores of our models trained on different numbers of episodes under Few-NERD inter 5-way 1shot setting.

TABLE XIV
EVALUATION OF MENTION DETECTION AND TYPE CLASSIFICATION APPLIED IN OUR METHOD.

Dataset	Setting	Mention F1	Type Accuracy
Few-NERD	10-way 1~2-shot (intra)	70.10	59.25
Cross-Dataset	1-shot (Wiki)	25.97	45.93
SNIPS	1-shot (Mu)	82.40	77.74
FewEvent	10-way 5-shot	67.00	96.14

VI. RELATED WORK

A. Few-Shot Sequence Labeling

Recently, there has been an increasing trend towards few-shot sequence labeling tasks [5]–[7], [12], [44]. Prior studies can be divided into two categories: one-stage and two-stage methods.

1) *One-stage Methods*: Most of them apply token-level or span-level metric learning based models. The token-wise sequence labeling formulation casts the problem into token-level classification. Fritzler *et al.* [5] directly apply prototypical networks [19] for few-shot NER. To capture the label dependencies among tokens, they further add a CRF layer trained in the target domains, which could be unreliable under few-shot settings because the transition matrix may overfit the tiny amount of examples. Hou *et al.* [6] propose a collapsed dependency transfer mechanism to only transfer the coarse-grained label transition probability of the CRF layer. Label semantics and pairwise embeddings are also used to boost performance. PA-CRF [12] learns to generate the Gaussian distribution of transition scores by utilizing the label prototypes to overcome the limited data issue when estimating the transition matrix in few-shot event detection. To mitigate the miscellaneous `Other` prototype issue, Yang and Katiyar [32] propose to use a simple nearest neighbor classifier with Viterbi decoding. Some other works [8], [10] try to better model the `Other` class by several prototypes. As for span-level metric learning methods, Yu *et al.* [9] explore a retrieval-based method to leverage the span-level similarity between support examples and query examples to solve the few-shot slot filling task. Wang *et al.* [8] propose to enhance the span representation and model the miscellaneous `Other` class via multiple prototypes. Besides the metric-learning based methods, some works explore a span extraction formulation for the few-shot sequence labeling task. These methods usually adopt

the pretrain-then-finetune paradigm. Ma *et al.* [37] formulate the task as a machine reading comprehension task and encode the slot names as questions to train a BERT-classifier for predicting the slot value span. ConVEx [38] pre-trains with pairwise cloze task on Reddit and fine-tunes on few-shot examples for each slot tag. Hou *et al.* [31] propose to use a generation model to generate the slot value given utterances and prompts constructed from slot names and values.

2) *Two-stage Methods*: While most previous works apply a unified framework to solve few-shot sequence labeling, a few works explore a two-stage paradigm for this task, *i.e.*, they perform mention detection and type classification separately. Deng *et al.* [7] mainly focus on using a dynamic memory network to improve the sentence embedding and prototype representation to enhance both the trigger identification and event type classification stages. They train a conventional trigger identifier with a dynamic-memory-based sentence encoding for trigger identification. Cong *et al.* [12] show that such conventional trigger identifier suffers from the trigger discrepancy problem and shows unsatisfactory results compared to unified metric-learning based methods. Our work differs from them in that we explore optimization based meta-learning algorithms at both stages for better leveraging information in support examples to deal with large domain gaps, and our framework is orthogonal to the their sentence encoding strategy. Wang *et al.* [45] first detect mentions with a conventional span classifier and then leverage type descriptions to achieve superior type classification performance in zero-/few-shot NER. Differently from previous works that adopt the two-stage strategy for either improving sentence encoding or leveraging type description on a single task, our task decomposition framework is developed to address the issue of miscellaneous `Other` prototype(s) and, at the same time, better models label dependencies via the shared boundary tags in mention detection. To the best of our knowledge, we are the first work to study and verify the effectiveness of the decomposition framework, further enhanced via meta-learning, for varied few-shot sequence labeling tasks - achieving SOTA performance in several few-shot sequence labeling benchmarks.

B. Meta-Learning for Few-Shot Learning

Meta-learning has been a popular paradigm for few-shot learning problem in recent years [46]–[49]. Various meta-learning algorithms have been developed, which can be typically divided into three categories: the black-box adaption methods [50], optimization-based methods [13], and metric-learning based methods [19], [51]. There are also some works to combine methods of different categories [52]. In this paper, we explore the use of meta-learning algorithms for few-shot sequence labeling tasks. Particularly, we propose to use optimization-based method MAML [13] in the mention detection, and combine the metric-learning based method ProtoNet [19] and the optimization-based method MAML in the type classification stage.

VII. CONCLUSION

In this paper, we propose a decomposed meta-learning approach for few-shot sequence labeling. We tackle the sequence

labeling task by performing mention detection and type classification sequentially, which enables us avoid the drawback of modeling the miscellaneous other prototype of previous works and better transfer the shared boundary knowledge across tasks. Furthermore, we propose to use the meta-learning algorithm to enhance both stages for fully exploring the information in support examples, to help bridge large domain gaps. Specifically, in the mention detection stage, we use MAML to enhance the training of the mention detection model for better parameter initialization, which can help the model rapidly adapt to new tasks via only a few gradient updates. In the type classification stage, we employ MAML to prompt the span-level ProtoNet to learn a more distinguishable embedding space by leveraging information in support examples. We implement a separate and a joint version of our framework, with the joint version reducing the model size and speeding up 33% while maintaining comparable performance to the former one. Extensive experiments and analyses on nine benchmarks show that our approach substantially outperforms previous SOTA methods across different few-shot sequence labeling tasks.

REFERENCES

- [1] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "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*. Association for Computational Linguistics, 2016, pp. 260–270.
- [2] R. Gangadharaiah and B. Narayanaswamy, "Joint multiple intent detection and slot labeling for goal-oriented dialog," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Jun. 2019, pp. 564–569.
- [3] N. Ding, Z. Li, Z. Liu, H. Zheng, and Z. Lin, "Event detection with trigger-aware lattice neural network," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 347–356.
- [4] L. Cui, Y. Li, and Y. Zhang, "Label attention network for structured prediction," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 1235–1248, 2022.
- [5] A. Fritzier, V. Logacheva, and M. Kretov, "Few-shot classification in named entity recognition task," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019, pp. 993–1000.
- [6] Y. Hou, W. Che, Y. Lai, Z. Zhou, Y. Liu, H. Liu, and T. Liu, "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*, 2020, pp. 1381–1393.
- [7] S. Deng, N. Zhang, J. Kang, Y. Zhang, W. Zhang, and H. Chen, "Meta-learning with dynamic-memory-based prototypical network for few-shot event detection," in *Proceedings of the 13th International Conference on Web Search and Data Mining*, ser. WSDM '20, 2020, p. 151–159.
- [8] P. Wang, R. Xu, T. Liu, Q. Zhou, Y. Cao, B. Chang, and Z. Sui, "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*, 2022, pp. 5012–5024.
- [9] D. Yu, L. He, Y. Zhang, X. Du, P. Pasupat, and Q. Li, "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*, 2021, pp. 734–749.
- [10] M. Tong, S. Wang, B. Xu, Y. Cao, M. Liu, L. Hou, and J. Li, "Learning from miscellaneous other-class words for few-shot named entity recognition," 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)*, 2021, pp. 6236–6247.
- [11] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001, pp. 282–289.
- [12] X. Cong, S. Cui, B. Yu, T. Liu, W. Yubin, and B. Wang, "Few-Shot Event Detection with Prototypical Amortized Conditional Random Field," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 28–40.
- [13] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. PMLR, 2017, pp. 1126–1135.
- [14] A. Rogers, O. Kovaleva, and A. Rumshisky, "A Primer in BERTology: What We Know About How BERT Works," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 842–866, 01 2021. [Online]. Available: https://doi.org/10.1162/tacl_a_00349
- [15] T. Ma, H. Jiang, Q. Wu, T. Zhao, and C.-Y. Lin, "Decomposed meta-learning for few-shot named entity recognition," in *Findings of the Association for Computational Linguistics: ACL 2022*. Association for Computational Linguistics, 2022, pp. 1584–1596. [Online]. Available: <https://aclanthology.org/2022.findings-acl.124>
- [16] N. Ding, G. Xu, Y. Chen, X. Wang, X. Han, P. Xie, H. Zheng, and Z. Liu, "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*, 2021, pp. 3198–3213.
- [17] Q. Wu, Z. Lin, G. Wang, H. Chen, B. F. Karlsson, B. Huang, and C. Lin, "Enhanced meta-learning for cross-lingual named entity recognition with minimal resources," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*. AAAI Press, 2020, pp. 9274–9281.
- [18] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.
- [19] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4077–4087.
- [20] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 09–15 Jun 2019, pp. 2790–2799.
- [21] B. Athiwaratkun, C. Nogueira dos Santos, J. Krone, and B. Xiang, "Augmented natural language for generative sequence labeling," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 375–385.
- [22] E. F. Tjong Kim Sang, "Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition," in *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.
- [23] A. Zeldes, "The gum corpus: Creating multilayer resources in the classroom," *Language Resources and Evaluation*, vol. 51, no. 3, pp. 581–612, 2017.
- [24] L. Derczynski, E. Nichols, M. van Erp, and N. Limsopatham, "Results of the WNUT2017 shared task on novel and emerging entity recognition," in *Proceedings of the 3rd Workshop on Noisy User-generated Text*. Association for Computational Linguistics, 2017, pp. 140–147.
- [25] S. Pradhan, A. Moschitti, N. Xue, H. T. Ng, A. Björkelund, O. Uryupina, Y. Zhang, and Z. Zhong, "Towards robust linguistic analysis using OntoNotes," in *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Sofia, Bulgaria: Association for Computational Linguistics, 2013, pp. 143–152.
- [26] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril *et al.*, "Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces," *arXiv preprint arXiv:1805.10190*, 2018.
- [27] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993. [Online]. Available: <https://aclanthology.org/J93-2004>
- [28] S. Petrov, D. Das, and R. McDonald, "A universal part-of-speech tagset," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. European Language Resources Association (ELRA), May 2012, pp. 2089–2096.
- [29] J. Nivre, M.-C. de Marneffe, F. Ginter, Y. Goldberg, J. Hajič, C. D. Manning, R. McDonald, S. Petrov, S. Pyysalo, N. Silveira, R. Tsarfaty, and D. Zeman, "Universal Dependencies v1: A multilingual treebank collection," in *Proceedings of the Tenth International Conference on*

- Language Resources and Evaluation (LREC'16)*. European Language Resources Association (ELRA), 2016, pp. 1659–1666.
- [30] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith, "Part-of-speech tagging for Twitter: Annotation, features, and experiments," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2011, pp. 42–47.
- [31] Y. Hou, C. Chen, X. Luo, B. Li, and W. Che, "Inverse is better! fast and accurate prompt for few-shot slot tagging," in *Findings of the Association for Computational Linguistics: ACL 2022*, 2022, pp. 637–647.
- [32] Y. Yang and A. Katiyar, "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)*, 2020, pp. 6365–6375.
- [33] S. S. S. Das, A. Katiyar, R. Passonneau, and R. Zhang, "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)*, 2022, pp. 6338–6353.
- [34] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.
- [35] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR 2019*, 2019.
- [36] S. W. Yoon, J. Seo, and J. Moon, "Tapnet: Neural network augmented with task-adaptive projection for few-shot learning," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, vol. 97. PMLR, 2019, pp. 7115–7123.
- [37] J. Ma, Z. Yan, C. Li, and Y. Zhang, "Frustratingly simple few-shot slot tagging," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 1028–1033.
- [38] M. Henderson and I. Vulić, "ConVEx: Data-efficient and few-shot slot labeling," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Jun. 2021, pp. 3375–3389.
- [39] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
- [40] S. Yang, D. Feng, L. Qiao, Z. Kan, and D. Li, "Exploring pre-trained language models for event extraction and generation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5284–5294.
- [41] V. D. Lai, F. Deroncourt, and T. H. Nguyen, "Exploiting the matching information in the support set for few shot event classification," in *Advances in Knowledge Discovery and Data Mining: 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11–14, 2020, Proceedings, Part II*, 2020, p. 233–245.
- [42] R. Zhang, W. Wei, X.-L. Mao, R. Fang, and D. Chen, "Hcl-tat: A hybrid contrastive learning method for few-shot event detection with task-adaptive threshold," in *Findings of EMNLP*, 2022.
- [43] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.
- [44] J. Huang, C. Li, K. Subudhi, D. Jose, S. Balakrishnan, W. Chen, B. Peng, J. Gao, and J. Han, "Few-shot named entity recognition: An empirical baseline study," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 10 408–10 423.
- [45] Y. Wang, H. Chu, C. Zhang, and J. Gao, "Learning from language description: Low-shot named entity recognition via decomposed framework," in *Findings of EMNLP*, 2021.
- [46] X. Han, H. Zhu, P. Yu, Z. Wang, Y. Yao, Z. Liu, and M. Sun, "FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.
- [47] R. Geng, B. Li, Y. Li, X. Zhu, P. Jian, and J. Sun, "Induction networks for few-shot text classification," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3904–3913.
- [48] M. Chen, W. Zhang, W. Zhang, Q. Chen, and H. Chen, "Meta relational learning for few-shot link prediction in knowledge graphs," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.
- [49] J. Li, B. Chiu, S. Feng, and H. Wang, "Few-shot named entity recognition via meta-learning," *IEEE Transactions on Knowledge & Data Engineering*, 2020.
- [50] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. PMLR, vol. 48, 2016, pp. 1842–1850.
- [51] O. Vinyals, C. Blundell, T. Lillicrap, k. kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems*, vol. 29, 2016.
- [52] E. Triantafyllou, T. Zhu, V. Dumoulin, P. Lamblin, U. Evci, K. Xu, R. Goroshin, C. Gelada, K. Swersky, P. Manzagol, and H. Larochelle, "Meta-dataset: A dataset of datasets for learning to learn from few examples," in *ICLR 2020*, 2020.