Few-Shot Named Entity Recognition via Meta-Learning

Jing Li, Billy Chiu, Shanshan Feng and Hao Wang

Abstract—Few-shot learning under the N-way K-shot setting (i.e., K annotated samples for each of N classes) has been widely studied in relation extraction (e.g., FewRel) and image classification (e.g., Mini-ImageNet). Named entity recognition (NER) is typically framed as a sequence labeling problem where the entity classes are inherently entangled together because the entity number and classes in a sentence are not known in advance, leaving the N-way K-shot NER problem so far unexplored. In this paper, we first formally define a more suitable N-way K-shot setting for NER. Then we propose FEWNER, a novel meta-learning approach for few-shot NER. FEWNER separates the entire network into a task-independent part and a task-specific part. During training in FEWNER, the task-independent part is meta-learned across multiple tasks and the task-specific part is learned for each individual task in a low-dimensional space. At test time, FEWNER keeps the task-independent part fixed and adapts to a new task via gradient descent by updating only the task-specific part, resulting in it being less prone to overfitting and more computationally efficient. Compared with pre-trained language models (e.g., BERT and ELMo) which obtain the transferability in an implicit manner (i.e., relying on large-scale corpora), FEWNER explicitly optimizes the capability of "learning to adapt quickly" through meta-learning. The results demonstrate that FEWNER achieves state-of-the-art performance against nine baseline methods by significant margins on three adaptation experiments (i.e., intra-domain cross-type, cross-domain intra-type and cross-domain cross-type).

Index Terms—Natural Language Processing, Sequence Labeling, Few-Shot Learning, Meta-Learning

1 INTRODUCTION

NAMED entity recognition (NER) is a fundamental task in natural language processing, aiming at jointly resolving the boundaries and the categorical label of a named entity in text [1], [2]. NER not only acts as a standalone tool for information extraction (IE), but also plays an essential role in a variety of downstream applications, such as information retrieval [3], question answering [4], etc.

A significant amount of work [5]–[11] has been devoted to developing end-to-end neural-based models for NER, but these require large quantities of annotated corpora. An effective solution to reduce the data requirement is transfer learning [12], which makes use of abundant data in a source task to improve performance in a low-resource target task. Recently, several studies have contributed effort to leveraging deep transfer learning for NER, and can be categorized along two lines: homogeneous label-set and heterogeneous label-set approaches. The homogeneous label-set approaches [13]–[15] assume that the source and target tasks have the same entity labels, which limits their applications. The heterogeneous label-set approaches [16]–[19] commonly partition an NER model into shared and private parts to address the label-discrepancy problem, resulting in the need of training the private part from scratch in target tasks. This paper addresses the following question, which has not yet been explored: *For NER adaptation, can the shared knowledge be learned across homogeneous and heterogeneous label-sets?*

An effective solution to address the above adaptation problem is few-shot learning under the N-way K-shot setting (i.e., K annotated samples for each of N classes, and a fixed label space: N classes) which has been widely used in relation classification (e.g., FewRel [20]) and image classification (e.g., Mini-ImageNet [21]–[23]). However, NER is typically framed as a sequence labeling problem whose goal is to assign a class to each word in a sentence. Sequence labeling is drastically different from the conventional classification task, where each member is independently classified into a category without taking sequence dependency into account. Moreover, a training sample of NER (e.g., a sentence) may have multiple entities whose number and classes are not known in advance. Therefore, the entity classes are inherently entangled together in a sentence. This raises a natural question: *can the standard N-way K-shot setting in few-shot image classification be adjusted so that it is more suitable for sequence labeling?*

In the N-way K-shot setting, it is still possible to transfer knowledge among different tasks because named entities often share lexical and context features. As humans, we have a remarkable ability to quickly grasp new concepts from a very small number of examples or a limited amount of experience, leveraging prior knowledge and context. In short, we *learn how to learn* much faster and more efficiently across various tasks. Meta-learning [24], [25] was proposed to mimic the human ability of quickly learning new skills with few examples. Recently, meta-learning has received resurgence in the context of few-shot learning [21], [26], [27]. Unfortunately, most meta-learning methods can easily overfit since the entire network is updated on just few samples at test time [28], [29].

In this paper, we first formally define the N-way K-shot setting in few-shot NER. Inspired by fast context

• *J. Li, B. Chiu, S. Feng and H. Wang are with the Inception Institute of Artificial Intelligence, Abu Dhabi 54115, UAE. E-mail: jingli.phd@hotmail.com; hon.chiu@inceptioniai.org; victor_fengss@foxmail.com; wanghao.paper@gmail.com. (Corresponding author: Shanshan Feng.)*

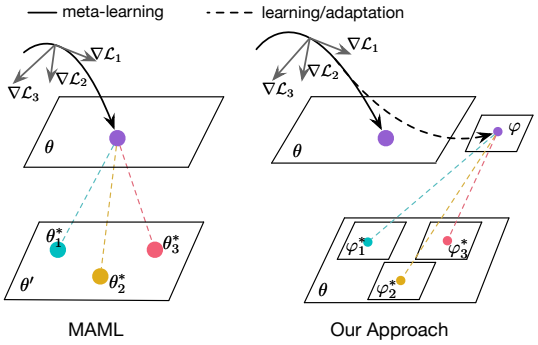

Fig. 1. Illustration of FEWNER. Our approach is less prone to overfitting and more computationally efficient because it adapts to new tasks on a low-dimensional space φ.

networks [28] in computer vision, we propose FEWNER, a novel meta-learning based approach for few-shot NER. FEWNER separates the entire network into a task-independent part (i.e., θ in Figure 1) and a task-specific part (i.e., φ in Figure 1). During training, θ is meta-learned across multiple tasks and φ is learned for each individual task in a low-dimensional space. At test time, FEWNER keeps θ fixed and adapts to a new task via gradient descent by updating only the task-specific part φ, while model agnostic meta learning (MAML) [25] performs adaptations by updating the entire network, as shown in Figure 1. Intuitively, FEWNER has two inherent advantages: 1) it is less prone to overfitting; 2) it is computationally efficient because it does not need the second order gradient computation with respect to θ, but only φ. On the other hand, pre-trained language models (e.g., BERT and GPT3) have showed powerful transfer performance in many NLP tasks. However, the transferability is obtained in an implicit manner where they rely heavily on large-scale corpora. Instead, FEWNER explicitly optimizes the capability of "learning to adapt quickly" through meta-learning. In summary, the main contributions of this work are four-fold:

- To the best of our knowledge, we are the first to investigate the problem of few-shot NER under N-way K-shot settings in a meta-learning manner.
- We propose FEWNER, a novel meta-learning approach for few-shot NER. FEWNER can quickly solve a new task by updating only a small set of parameters in a low-dimensional space with few gradient update steps, resulting in it being less prone to overfitting and more computationally efficient.
- We extensively evaluate FEWNER on three experiments. FEWNER achieves state-of-the-art performance by significant margins, outperforming recent baseline methods.
- We conduct experiments to further analyze the parameter settings and architectural choices. We also present a study for qualitative analysis.

2 BACKGROUND

2.1 Named Entity Recognition

Named entity recognition (NER) is usually framed as a sequence labeling problem. There are three common paradigms for NER [1]: *knowledge-based unsupervised* systems, *feature-based supervised* systems and *neural-based* systems. *Knowledge-based unsupervised* systems rely on lexical knowledge, including domain-specific gazetteers [30], and shallow syntactic knowledge [31]. *Feature-based supervised* systems cast NER as a multi-class classification or sequence labeling task. Feature engineering is critical in these systems. For example, Ji et al. [32] designed 19 local features and 5 global features for location recognition in Tweets. Based on manually crafted features, many algorithms have been applied in supervised NER, e.g., the Support Vector Machine (SVM) [33], Hidden Markov Model (HMM) [34] and Conditional Random Field (CRF) [32].

Recently, several neural architectures have been widely applied in NER because neural-based systems have the advantage of inferring latent features and learning sequence labels in an end-to-end fashion. The use of neural models for NER was pioneered in [8], where an architecture based on temporal convolutional neural networks (CNNs) over a word sequence was proposed. Since then, there has been a growing body of work on neural-based NER. Existing neural-based systems can be unified into a framework with three components: an input representation, context encoder and tag decoder. Commonly used input representations include word-level and character-level representations [5], [6], [35]. Widely used context encoder architectures include CNNs [8], recurrent neural networks (RNNs) [7], recursive neural networks [36] and deep transformers [37]. At the top of the context encoder, a CRF layer [38], a pointer network [15], [39], or an RNN layer [40] is employed to make sequence label predictions.

In addition, transfer learning aims to perform a machine learning task in a target domain by taking advantage of knowledge learned from a source domain [12]. Several studies [18], [41]–[44] have already contributed effort to leveraging deep transfer learning for NER. Yang et al. [45] first investigated the transferability of different layers of representations. Pius and Mark [46] extended Yang's approach to allow joint training on the informal corpus and incorporate sentence-level feature representations. Jia et al. [17] utilized the cross-domain language model as a bridge across domains to design a novel parameter generation network. Zhou et al. [19] proposed two adversarial transfer network to explore effective feature fusion between high and low resource domains. Different from these parameter-sharing architectures, some methods [47], [48] apply transfer learning in NER by first training a model on a source task and then using it on the target task for fine-tuning. Recently, Li et al. [15] proposed an adversarial approach to transfer knowledge between two domains, rather than few-shot NER. Specially, Hou et al. [44] just investigated the K-shot setting for sequence labeling, leaving N-way unexplored. To the best of our knowledge, there is no work addressing the N-way K-shot scenario for few-shot NER.

2.2 Meta-Learning

Meta-learning (a.k.a. learning to learn) [24], [49] aims to learn a general model that can quickly adapt to a new task given very few training samples, without needing to be retrained from scratch. Most recent approaches to meta-

learning focus on few-shot learning and can be broadly categorized as metric-based methods [50], [51], memory-based methods [52], [53], and optimization-based methods [25], [26]. Here, we introduce an optimization-based method, Model-Agnostic Meta-Learning (MAML) [25], in detail.

Formally, a model is represented by a function f_θ with parameters θ. MAML first forms a set of training tasks $\mathcal{T} = \{\mathcal{T}_1, ..., \mathcal{T}_i, ...\}$, where each task consists of a support set $\mathcal{D}_{spt}$ and a query set $\mathcal{D}_{qry}$ ($\mathcal{D}_{spt} \cap \mathcal{D}_{qry} = \emptyset$). In the N-way K-shot [50] setting (i.e., the training instances are sampled with K labeled examples from each of N classes), the model changes parameters θ to θ_i' on $\mathcal{D}_{spt}$ by gradient descent:

$$\theta_i' \leftarrow \theta - \alpha \nabla \mathcal{L}_{\mathcal{T}_i}^{spt}(f_\theta) \tag{1}$$

where α is a universal learning rate, and $\mathcal{L}_{\mathcal{T}_i}^{spt}$ is the task-related training loss. Model parameters θ are trained to optimize the performance of $f_{\theta_i'}$ on the unseen validation examples from $\mathcal{T}_i$ across tasks. This leads to the MAML meta-objective:

$$\min_\theta \sum_{\mathcal{T}_i} \mathcal{L}_{\mathcal{T}_i}^{qry}(f_{\theta_i'}) = \min_\theta \sum_{\mathcal{T}_i} \mathcal{L}_{\mathcal{T}_i}^{qry}(f_{\theta - \alpha \nabla \mathcal{L}_{\mathcal{T}_i}^{spt}(f_\theta)}) \tag{2}$$

The goal of MAML is to optimize the model parameters θ to quickly adapt to new tasks over a few gradient steps, with few training examples from the unseen tasks. The model parameter θ is updated by gradient descent:

$$\theta \leftarrow \beta \nabla_\theta \sum_{\mathcal{T}_i} \mathcal{L}_{\mathcal{T}_i}^{qry}(f_{\theta_i'}) \tag{3}$$

where β is the learning rate of meta optimization. Note that the objective of MAML is designed for few-shot classification under the problem setting of N-way K-shot. For sequence labeling, a training example may have multiple entities whose number and classes in a sentence are not known in advance. Thus, we need to reformulate the N-way K-shot setup for the sequence labeling problem.

Some studies [21]–[23], [54] have applied meta-learning strategies for image classification in few-shot learning. However, meta-learning for natural language processing is less common than for computer vision. There have been a few attempts devoted to the application of meta-learning in NLP over the last two years. Gu et al. [55] first explored meta-learning in neural machine translation. They framed the low-resource translation as a meta-learning problem which learns to adapt to low-resource languages based on multilingual high-resource language tasks. Huang et al. [56] proposed a method for natural language to structured query generation based on MAML [25], by reducing a regular supervised learning problem to the few-shot meta-learning scenario. Qian and Zhou [57] proposed DAML, which is based on meta-learning, to combine multiple dialog tasks during training, in order to learn general and transferable information that is applicable to new domains. Lin et al. [58] proposed casting personalized dialog learning as a meta-learning problem, which allows the model to generate personalized responses by efficiently leveraging only a few dialog samples instead of human-designed persona descriptions. Especially, Wu et al. [43] extended MAML to the cross-lingual NER task with minimal resources. However, Wu's approach does not work on N-way K-shot settings, requiring a certain percentage of labeled training data (i.e., 5%) for fine-tuning in low-resource settings. In addition, our approach differs from these strategies in that FEWNER performs adaptations in a low-dimensional space instead of the entire network.

3 FEWNER: FEW-SHOT NAMED ENTITY RECOGNITION VIA META-LEARNING

In this section, we first define the problem of meta-learning for NER, especially under the N-way K-shot setting. Then we present a layer-by-layer description of FEWNER.

3.1 Problem Statement: N-Way K-Shot in NER

In few-shot learning, we aim to obtain a model $f : \boldsymbol{s} \mapsto \hat{\boldsymbol{y}}$ that maps a sentence $\boldsymbol{s} = \{w_1, ..., w_l, ...w_L\}$ in which each word has a true label $\boldsymbol{y} = \{y_1, ..., y_l, ...y_L\} \in \mathcal{Y}$ to predictions $\hat{\boldsymbol{y}} \in \mathcal{Y}$ with few training samples. A task $\mathcal{T}_i$ is a batch of sentences, which consists of a support set $\mathcal{D}_{\mathcal{T}_i}^{spt}$ and a query set $\mathcal{D}_{\mathcal{T}_i}^{qry}$ ($\mathcal{D}_{spt} \cap \mathcal{D}_{qry} = \emptyset$). In the training phase, the true labels of $\mathcal{D}_{\mathcal{T}_i}^{spt}$ and $\mathcal{D}_{\mathcal{T}_i}^{qry}$ are both available in source tasks. In the testing phase, an unseen target task $\mathcal{T}_j$ only contains a few labeled samples $\mathcal{D}_{\mathcal{T}_j}^{spt}$. The ultimate goal is to make predictions for $\mathcal{D}_{\mathcal{T}_j}^{qry}$, given the $\mathcal{D}_{\mathcal{T}_j}^{spt}$.

The N-way K-shot setting has been widely adopted [20], [25], [54] in recent research on few-shot learning, where $\mathcal{D}_{\mathcal{T}_i}^{spt}$ and $\mathcal{D}_{\mathcal{T}_j}^{spt}$ usually both include K samples (K-shot) for each of N classes (N-way). Constructing a task is easier in conventional classification problems (i.e., where each instance has a unique class) than our sequence labeling problem. This is because a training example (e.g., a sentence) in sequence labeling may have multiple entities whose number and classes are not known in advance. That is, the classes are inherently entangled together in sequence labeling.

Similar to the dependency transfer approach [44], we utilize a *greedy-including* approach to construct N-way K-shot tasks for sequence labeling: (1) we randomly pick up a sentence and then greedily extend the set $\mathcal{D}_{\mathcal{T}_i}^{spt}$. (2) a sentence is included if it can bring a gain for "way" or "shot". Taking the 5-way 1-shot configuration as an example, $\mathcal{D}_{\mathcal{T}_i}^{spt} = \emptyset$ is initialized an empty set at first. Given a sentence, "*[Jordan]$_{PER}$ is a [NBA]$_{ORG}$ player.*", it is picked. Given a new sentence, "*[The Chicago Bulls]$_{ORG}$ selected [Jordan]$_{PER}$*", it is not picked because there is no gain for "way" or "shot". Given other sampled sentence, "*[Jordan]$_{PER}$ was seen gambling in [Atlantic City]$_{LOC}$*", it is picked because there is a gain for "way" (i.e., new class LOC). (3) repeat until the number of classes and shots reaches N and K, respectively. Finally, at least one class will appear less than K times in $\mathcal{D}_{\mathcal{T}_i}^{spt}$ if a sentence is removed from it. $\mathcal{D}_{\mathcal{T}_i}^{qry}$ is constructed with the same N classes of $\mathcal{D}_{\mathcal{T}_i}^{spt}$ and the rest of the dataset after constructing $\mathcal{D}_{\mathcal{T}_i}^{spt}$.

3.2 The FEWNER Approach

3.2.1 *Overview of* FEWNER

Figure 2 shows an overview of our proposed FEWNER, which consists of a training phase and a testing phase. The key idea of this paper is to learn a model on a variety of tasks, such that it can quickly solve new unseen tasks with

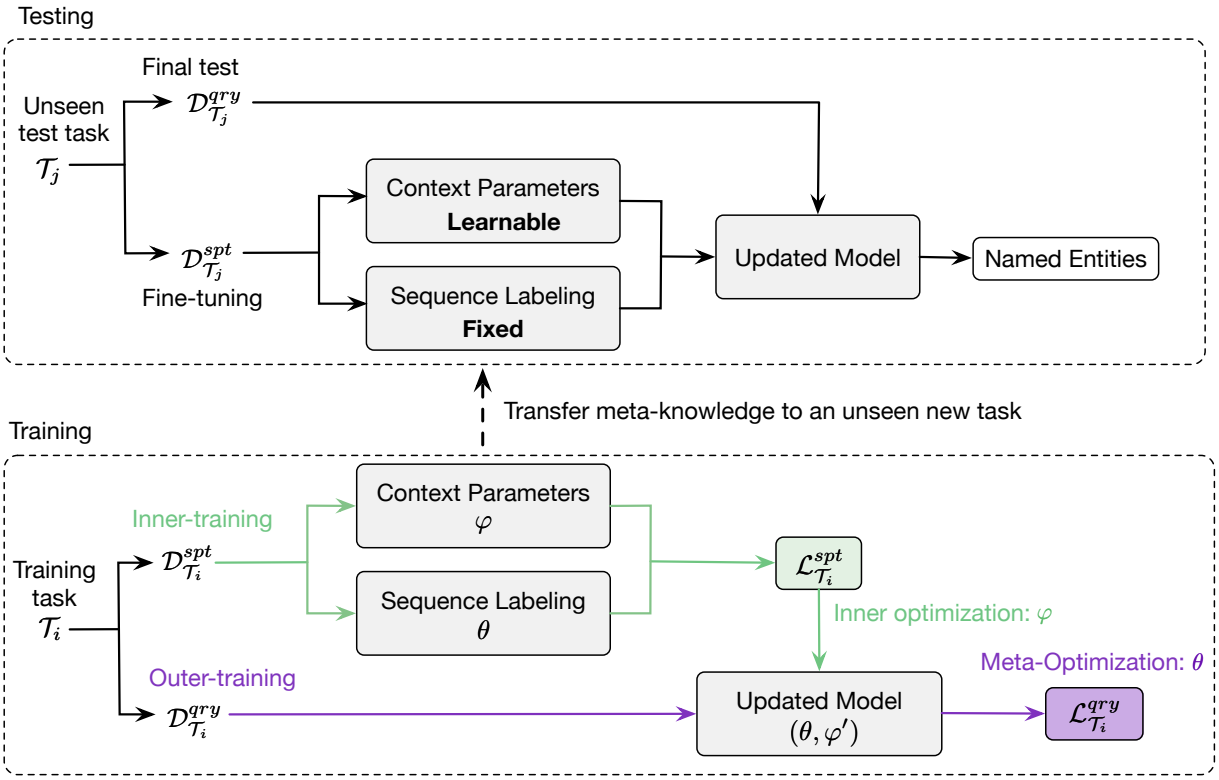

Fig. 2. An overview of FEWNER, which consists of the shared task-independent parameters θ and learned task-specific context parameters φ. During inner-training, φ is initialized to $\mathbf{0}$ and is learned on $\mathcal{D}^{spt}_{\mathcal{T}_i}$ from scratch for each task. During outer-training, θ is learned on $\mathcal{D}^{qry}_{\mathcal{T}_i}$ across multiple tasks. In the test phase, θ is fixed and the context parameters φ are adapted to new tasks with few samples $\mathcal{D}^{spt}_{\mathcal{T}_j}$, resulting in FEWNER being less prone to overfitting and more computationally efficient.

only a small number of training samples. More importantly, FEWNER adapts to new tasks via gradient descent by updating only a small set of learned parameters (i.e., task-specific context parameters φ) at test time, instead of the entire network θ. FEWNER has two inherent advantages: (1) it adapts to new tasks through a low-dimensional space φ and is less prone to overfitting, compared to some existing methods [25], [26] that make adaptation updates on θ with few training samples. (2) it does not need the second order gradient computation with respect to θ, but only φ. In addition, the N-way K-shot mechanism allows FEWNER to learn meta-knowledge and label dependencies from the learning experience across many different tasks that share a same label space, i.e., N ways. In summary, the context parameters φ (initialized to $\mathbf{0}$ in each task loop) are learned from each individual task; the sequence labeling parameters θ are learned across multiple different tasks.

3.2.2 The Backbone of Sequence Labeling

Based on the summarization in the survey of NER [1], CRF is powerful to capture label transition dependencies when adopting static embeddings such as Word2vec and GloVe. CNN-BiGRU is much computationally cheaper than Transformers based encoders (e.g., GPT3 has 175 billion parameters). Transformers fail on NER task if they are not pre-trained and when the training data is limited [59], [60]. The performance of CNN-BiGRU is comparative with, even better than Transformers-based models on small corpora when training from scratch. Essentially, our approach is

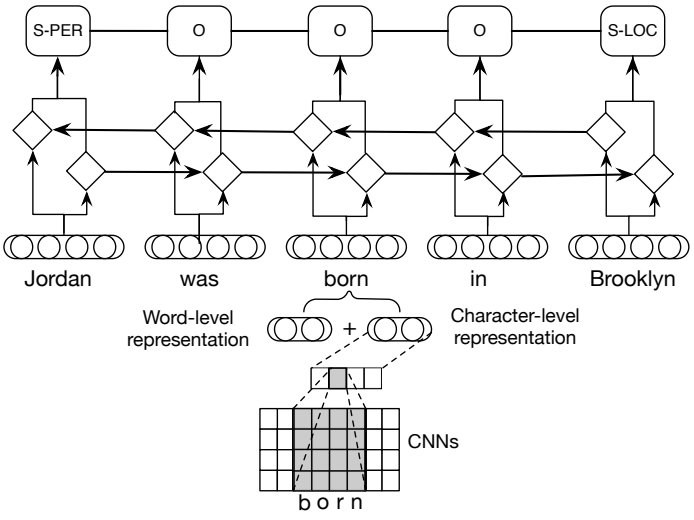

Fig. 3. CNN-BiGRU-CRF for sequence labeling (i.e., θ).

model-agnostic. Therefore, we adopt CNN-BiGRU-CRF as the backbone of sequence labeling based on the consideration of the size of experimental corpora and the computational complexity. Note that we use θ to denote all parameters in CNN-BiGRU-CRF. As shown in Figure 3, the backbone utilizes a Convolutional Neural Network (CNN) to extract character-level representations, a bidirectional Gated Recurrent Unit (BiGRU) to encode sequence context

and a Conditional Random Field (CRF) layer to produce the tag sequence.

Given an input sentence $\boldsymbol{W} = (W_1, W_2, \ldots, W_L)$ of length L, let W_l denote its l-th word. After the input representation layer, the input sequence can be represented as $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_L)$. Then a BiGRU is used to encode the sequence context and yields hidden states in $\boldsymbol{h} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_L\} \in \mathbb{R}^{L \times 2H}$, where H is the hidden size of the GRU layer. For the tag decoder, we rely on a CRF that can model the label sequence jointly instead of decoding each label independently. Formally, consider $\boldsymbol{h} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_L\}$ as the input; $\boldsymbol{y} = \{y_1, y_2, ..., y_L\}$ is the corresponding label sequence. $\mathcal{Y}(\boldsymbol{h})$ denotes the set of possible label sequences for $\boldsymbol{h}$. The probabilistic model for the sequence CRF defines a series of probabilities $p(\boldsymbol{y}|\boldsymbol{h}; \boldsymbol{W}, \boldsymbol{b})$ over all possible label sequences $\boldsymbol{y}$, given $\boldsymbol{h}$, by:

$$p(\boldsymbol{y}|\boldsymbol{h}; \boldsymbol{W}, \boldsymbol{b}) = \frac{\prod_{i=1}^{L} \psi_i(y_{i-1}, y_i, \boldsymbol{h})}{\sum_{\boldsymbol{y'} \in \mathcal{Y}(\boldsymbol{h})} \prod_{i=1}^{L} \psi_i(y'_{i-1}, y'_i, \boldsymbol{h})} \tag{4}$$

where $\psi_i(y', y, \boldsymbol{h}) = \exp(\boldsymbol{W}_{y',y}^T \boldsymbol{h}) + \boldsymbol{b}_{y',y}$, $\boldsymbol{W}_{y',y}^T$ and $\boldsymbol{b}_{y',y}$ are the weights and bias corresponding to label pair (y', y), respectively.

3.2.3 Meta-Learning Strategy

The meta-learning strategy consists of two core phases: an inner-training phase and an outer-training phase, as shown in Figure 2.

Inner-Training. For each task $\mathcal{T}_i \in \boldsymbol{T}$ in the inner training, we first initialize φ to $\mathbf{0}$ before the forward pass. Then the task-specific context parameters φ are adapted by gradient descent on $\mathcal{D}_{\mathcal{T}_i}^{spt}$ at the inner step k:

$$\varphi_k = \varphi_{k-1} - \alpha \nabla_{\varphi_{k-1}} \mathcal{L}_{\mathcal{T}_i}^{spt}(\theta, \varphi_{k-1}) \tag{5}$$

where α is the learning rate of the inner optimization and $\mathcal{L}_{\mathcal{T}_i}^{spt} = -\sum p(\boldsymbol{y}|\boldsymbol{h})$. At each inner step, the gradients are calculated with respect to the parameters from the previous step (i.e., $\nabla_{\varphi_{k-1}}$). Note that the task-independent parameters θ are not changed in the inner loop and the updated parameters φ_k are also a function of θ since the gradients flow through the model θ during backpropagation.

Outer-Training. After inner-training, we have already obtained an updated model (θ, φ_k). This updated model will be tested on new samples (i.e., $\mathcal{D}_{\mathcal{T}_i}^{qry}$) from $\mathcal{T}_i$. The intuition is that θ is improved by considering how the test error on new data changes with respect to the parameters. It mimics the process of the temporary model (i.e., (θ, φ_k)) being adapted to unseen data. More specifically, the task-independent parameters θ are updated by gradient descent:

$$\theta \leftarrow \theta - \beta \nabla_\theta \frac{1}{|\boldsymbol{T}|} \sum_{\mathcal{T}_i \in \boldsymbol{T}} \mathcal{L}_{\mathcal{T}_i}^{qry}(\theta, \varphi_k) \tag{6}$$

where β is the meta-learning rate and $|\boldsymbol{T}|$ is the batch size. Note that Equation (6) is computed by differentiating the loss $\mathcal{L}_{\mathcal{T}_i}^{qry}$ with respect to the parameters θ. Unlike the common gradient, the update mechanism of Equations (6) involves a gradient (i.e., ∇_θ) through a gradient (i.e., $\nabla_{\varphi_{k-1}}$). This process requires second order optimization partial derivatives due to the dependency on Equation 5.

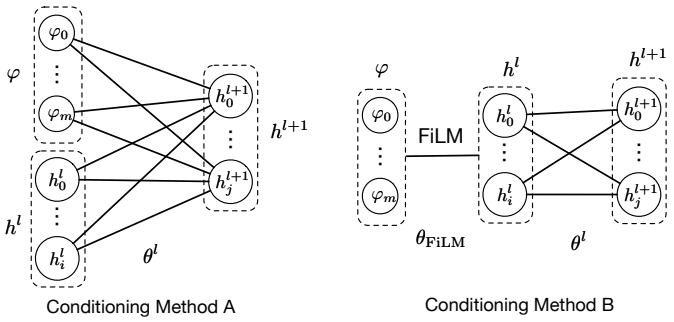

Fig. 4. Conditioning context parameters φ on backbone θ. A network layer h^l is augmented with task-specific parameters φ.

3.2.4 Conditioning on Backbone Parameters

For each task $\mathcal{T}_i$, the context parameters φ are learned from scratch. Following [25], [28], we set the initial context parameters to a vector filled with zeros before inner updates, i.e., $\varphi = \mathbf{0} = [0, ..., 0]^\top$. However, φ is independent of the network input, so we need to know where and how to condition the backbone network θ. For a layer θ_l, we can simply concatenate φ to the input of this layer h_l so that the concatenation can modulate the rest of the network to solve new tasks:

$$h_j^{l+1} = g(\sum \theta_{i,j} \cdot h_i^l + \sum \theta_{m,j} \cdot \varphi_m + b) \tag{7}$$

where $\theta_{i,j}$ are the weights associated with layer input h_i^l, and $\theta_{m,j}$ are the weights associated with the context parameter φ_m. Another conditioning method is based on *feature-wise linear modulation* FiLM [61], which performs an affine transformation on the feature maps:

$$FiLM(h_i) = \gamma h_i + \eta \tag{8}$$

$$[\gamma, \eta] = \sum \theta_{FiLM} \cdot \varphi_m + b \tag{9}$$

Note that φ is updated in the inner loop. $\theta_{i,j}$, $\theta_{m,j}$ and θ_{FiLM} are updated in the outer loop. These two conditioning methods are illustrated in Figure 4. In our implementation, we condition φ on the output of BiGRU $\boldsymbol{h}$ using FiLM. The intuition is that the adapted hidden states are more beneficial for capturing task-specific label dependencies in the subsequent CRF layer.

3.2.5 Algorithm Flow

Algorithm 1 summarizes the procedures for training and adapting FEWNER. In the training procedure, the task-specific parameters φ are initialized to $\mathbf{0}$ and updated by the loss $\mathcal{L}_{\mathcal{T}_i}^{spt}$ for each task $\mathcal{T}_i$. The task-independent parameters θ are updated by the loss $\mathcal{L}_{\mathcal{T}_i}^{qry}$ across different tasks, and φ is not updated in the outer optimization. In the adapting procedure, we first initialize the sequence labeling model using θ_{Meta}, which was already learned in the previous training procedure. The main difference between these two procedures is that θ_{Meta} is fixed and φ is learned from the current hold-out task $\mathcal{T}_j$ in the adapting procedure.

4 EXPERIMENTS

In this section, we first detail our experimental settings. Then, we present our experimental results on three adaptation scenarios: intra-domain cross-type, cross-domain intra-type, and cross-domain cross-type adaptations.

Algorithm 1: Training and Adapting FEWNER

Training `Procedure()`
  Set step size α, β; Initialize θ;
  while *not converge* **do**
    Sample a batch of tasks $\boldsymbol{T} = \{\mathcal{T}_i\}_{i=1}^{\text{MetaBatch}}$;
    for $\mathcal{T}_i$ *in MetaBatch* **do** `// Outer loop`
      Set $\varphi = \mathbf{0} = [0, ..., 0]^\top$;
      for k *in InnerSteps* **do** `// Inner gradient steps`
        $\varphi_k = \varphi_{k-1} - \alpha \nabla_{\varphi_{k-1}} \mathcal{L}_{\mathcal{T}_i}^{spt}(\theta, \varphi_{k-1})$
    $\theta \leftarrow \theta - \beta \nabla_\theta \frac{1}{|\boldsymbol{T}|} \sum_{\mathcal{T}_i \in \boldsymbol{T}} \mathcal{L}_{\mathcal{T}_i}^{qry}(\theta, \varphi_k)$; `// Meta-update`
  return θ_{Meta}
Adapting `Procedure()`
  Set step size α; Initialize θ using θ_{Meta};
  Sample a hold-out task $\mathcal{T}_j$;
  Set $\varphi = \mathbf{0} = [0, ..., 0]^\top$;
  for k *in InnerSteps* **do** `// Inner gradient steps`
    $\varphi_k = \varphi_{k-1} - \alpha \nabla_{\varphi_{k-1}} \mathcal{L}_{\mathcal{T}_j}^{spt}(\theta_{\text{Meta}}, \varphi_{k-1})$
  Evaluate $\mathcal{D}_{\mathcal{T}_j}^{qry}$ using the model $(\theta_{\text{Meta}}, \varphi_k)$; `//` θ_{Meta} `is fixed,` φ_k `is learned from the current task` $\mathcal{T}_j$
  return NER Performance

TABLE 1
Statistics of datasets used in our experiments.

Dataset	Genre	#Types	#Sentences	#Mentions
NNE	Newswire	114	39932	185925
FG-NER	Newswire	200	3941	7384
GENIA	Medical	36	18546	76625
ACE2005	Various	54	17399	48397
OntoNotes	Various	18	42224	104248
BioNLP13CG	Medical	16	5939	21315

4.1 Experimental Settings

4.1.1 Datasets and Evaluation Metrics

Unlike image classification [51] and relation extraction [20], there is no existing few-shot learning dataset for NER. In our three groups of experiments, we collect a total of seven datasets, as summarized in Table 1. NNE, FG-NER and GENIA are used in the intra-domain cross-type adaptation. ACE 2005 consists of 7 domains and is used in the cross-domain intra-type adaptation. GENIA, BioNLP13CG, OntoNotes and FG-NER are used in the cross-domain cross-type adaptation. Details of data splits will be introduced in sections 4.2.1, 4.3.1 and 4.4.1.

In the N-way K-shot setting, each hold-out task $\mathcal{T}_j$ consists of a support set $\mathcal{D}_{\mathcal{T}_j}^{spt}$ and a query set $\mathcal{D}_{\mathcal{T}_j}^{qry}$. We consider the samples in $\mathcal{D}_{\mathcal{T}_j}^{qry}$ as a testing episode. For each episode, g is the total number of annotated entities in ground truth; r is the total number of entities detected by a model; c is the total number of correctly detected entities by the model. F1 score for each episode is calculated as $F1 = \frac{2 \times \text{Precison} \times \text{Recall}}{\text{Precison} + \text{Recall}} = \frac{2c}{g+r}$. We take the average of the F1 scores (95% confidence intervals) over all the episodes as the evaluation metric, i.e., mean $\pm$ 1.96 $\times$ standard deviation/$\sqrt{\text{sample size}}$.

4.1.2 Baseline Methods

We evaluate FEWNER against the following competitors:

- **FineTune** - This method takes CNN-BiGRU-CRF (see Section 3.2.2) as the sequence labeling model. It is trained on the support sets of training tasks. Then, it is fine-tuned on the support sets of test tasks and finally evaluated on the query sets of test tasks.
- **ProtoNet** - This method regards sequence labeling as classification for each single token [62]. Prototypical Network [51] is a few-shot classification model that learns a metric space in which classification can be performed by computing the distances to prototype representations of each class.
- **MAML** - This is a Model-Agnostic Meta-Learning method [25], which does not partition network parameters into task-specific and task-independent parameters. It requires the entire network to be updated in inner loops and at test time.
- **SNAIL** - This is a meta-learning model that utilizes a combination of temporal convolutions and causal attention: the former to aggregate information from past experience and the latter to pinpoint specific pieces of information [29].
- **Pre-trained Language Models** - These models employ a language model during training. They are very efficient in predicting the next word or masked words in a sequence. We build up a **CRF** layer on the top of **GPT2** [63], **Flair** [64], **ELMo** [6], **BERT** [37] and **XLNet** [65]. Note that we utilize the Flair[1] tool to produce contextualized embeddings. However, the Flair framework does not allow further fine-tune language models during downstream training. This may be a threat to validity. These stacked models are first trained on the support sets of training tasks. Then, they are fine-tuned (i.e., only CRF can be fine-tuned) on the support sets of test tasks and finally evaluated on the query sets of test tasks.

4.1.3 Implementation Details

For FineTune, ProtoNet, MAML, SNAIl and FEWNER, we use GloVe 300-dimensional pre-trained word embeddings released by Stanford, which are fine-tuned during training. The dimension of the character-level representation is 100 and the CNN filters are $[2, 3, 4]$. The total number of CNN filters is 150. Note that the GloVe embeddings are uncased and the character-level representations are cased. The hyper-parameter selection is based on experiments with a grid search strategy. We present the optimal parameters as follows. The bidirectional GRU has a depth of 1 and hidden size of 128. In our approach, the inner learning rate α is 0.1 and meta-learning rate β is 0.0008. The size of inner gradient steps is 2 for training and 8 for testing. The size of a meta batch for outer loops is 8. We use a dropout of 0.3 after the convolutional or recurrent layers and a fixed L2 regularization of 10^{-7}. The decay rate is 0.9 for each 5000 tasks and the gradient clip is 5.0. Our proposed FEWNER is implemented with the PyTorch framework and evaluated on NVIDIA Tesla V100 GPUs. Note that FEWNER requires second order optimization partial derivatives.

1. https://github.com/flairNLP/flair

TABLE 2
Intra-domain cross-type adaptation performance (average F1 score with 95% confidence intervals on a set of 1000 randomly constructed tasks) on the test sets of NNE, FG-NER and GENIA.

Methods	NNE: 5-way		FG-NER: 5-way		GENIA: 5-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Dynamic Token Representation: Contextualized Language Model Embeddings + CRF						
GPT2	$14.36 \pm 0.59\%$	$15.51 \pm 0.60\%$	$13.96 \pm 0.65\%$	$14.21 \pm 0.85\%$	$13.75 \pm 0.78\%$	$14.45 \pm 0.79\%$
Flair	$15.26 \pm 0.48\%$	$16.32 \pm 0.46\%$	$15.85 \pm 0.63\%$	$16.87 \pm 0.81\%$	$9.77 \pm 0.43\%$	$11.44 \pm 0.46\%$
ELMo	$15.85 \pm 0.54\%$	$16.33 \pm 0.58\%$	$18.74 \pm 0.73\%$	$18.90 \pm 0.91\%$	$15.21 \pm 0.44\%$	$19.18 \pm 0.64\%$
BERT	$16.61 \pm 0.56\%$	$17.16 \pm 0.59\%$	$16.56 \pm 0.64\%$	$19.67 \pm 0.83\%$	$12.02 \pm 0.55\%$	$14.93 \pm 0.53\%$
XLNet	$16.34 \pm 0.61\%$	$17.23 \pm 0.58\%$	$16.83 \pm 0.67\%$	$19.01 \pm 0.85\%$	$11.98 \pm 0.44\%$	$12.03 \pm 0.52\%$
Static Token Representation: GloVe + CNN						
FineTune	$18.24 \pm 0.50\%$	$18.34 \pm 0.52\%$	$17.85 \pm 0.69\%$	$20.69 \pm 0.87\%$	$6.67 \pm 0.32\%$	$7.21 \pm 0.34\%$
ProtoNet	$19.45 \pm 0.75\%$	$21.44 \pm 0.65\%$	$22.78 \pm 0.85\%$	$25.67 \pm 0.81\%$	$12.34 \pm 0.47\%$	$15.03 \pm 0.50\%$
MAML	$19.98 \pm 0.83\%$	$22.56 \pm 0.73\%$	$24.09 \pm 0.79\%$	$26.82 \pm 0.74\%$	$13.73 \pm 0.59\%$	$16.46 \pm 0.49\%$
SNAIL	$20.17 \pm 0.78\%$	$24.48 \pm 0.82\%$	$25.68 \pm 0.76\%$	$29.89 \pm 0.94\%$	$15.66 \pm 0.52\%$	$20.74 \pm 0.68\%$
FewNER (ours)	$\mathbf{23.74 \pm 0.65\%}$	$\mathbf{29.50 \pm 0.68\%}$	$\mathbf{30.54 \pm 0.85\%}$	$\mathbf{40.16 \pm 1.24\%}$	$\mathbf{23.24 \pm 0.73\%}$	$\mathbf{29.19 \pm 0.64\%}$

4.2 Intra-Domain Cross-Type Adaptation

4.2.1 Setups

In this experiment, we investigate the intra-domain cross-type adaptation, which aims to identify novel entity types with few training samples for a specific domain. We verify the effectiveness of different methods on three datasets: NNE (Newswire domain), FG-NER (Newswire domain) and GENIA (Medical domain). Because we aim to identify novel entity types, we split each dataset into non-overlapping partitions, i.e., the entities used for testing do not appear during training. In the end, the training, validation and testing sets contain $52, 10, 15$ types in NNE, $163, 15, 20$ types in FG-NER and $18, 8, 10$ types in GENIA. We report the average F1 scores with 95% confidence intervals on a random set of 1000 tasks from the test sets. For fair comparison, different methods are evaluated on the same randomly sampled 1000 tasks because we fix the random seed in the evaluation phase.

4.2.2 Experimental Results

Table 2 reports the experimental results of intra-domain cross-type adaptation. We make the following observations:

First, our approach FEWNER achieves state-of-the-art performance by significant margins, outperforming two groups of baseline methods (i.e., dynamic token representation and static token representation). More specifically, our model outperforms the best results of dynamic-representation-based methods by relative F1 improvements of 42.93%, 62.91% and 52.79% in the 1-shot setting, and 71.21%, 104.17% and 52.19% in the 5-shot setting, for NNE, FG-NER and GENIA datasets, respectively. Our model outperforms the best result of static-representation-based methods by relative F1 improvements of 17.70%, 18.89% and 48.40% in the 1-shot setting, and 20.51%, 34.36% and 40.74% in the 5-shot setting, for the NNE, FG-NER and GENIA datasets, respectively. We attribute this to the fact that FEWNER is effective in adapting to a new task by learning its task-specific context information (i.e., φ).

Second, the performance in the 5-shot setting is better than the 1-shot setting. This is reasonable because there are more training samples in the 5-shot setting. However, the improvement is only slight. Compared with relation extraction [20] and image classification [52], our empirical results show that few-shot learning under the N-way K-shot setting in sequence labeling remains challenging and there is still much room for improvement.

Third, the NNE and FG-NER datasets are from newswire text and GENIA is from medical text. We observe that few-shot learning in domain-specific NER (e.g., the medical domain) is more difficult than in general domains (e.g., the newswire domain). In particular, the FineTune method (without using any adaptation strategy) yields the worst performance (i.e., $6.67 \pm 0.32\%$ and $7.21 \pm 0.34\%$) on the GENIA dataset. Notably, our approach (i.e., $23.24 \pm 0.73\%$ and $29.19 \pm 0.64\%$) significantly outperforms this baseline method.

Fourth, the methods with adaptation strategies (i.e., ProtoNe, MAML and SNAIL) generally outperform the contextualized language models (i.e., GPT2, Flair, ELMo, BER and XLNet) with a CRF. Although the pe-trained language models are successful in many downstream tasks, such as text classification, they fail in few-shot NER. In contrast, our approach works on a lower-dimensional space. That is, the sequence labeling network (i.e., θ) is fixed during adaptation and the task-specific context parameters (i.e., low-dimensional φ) are dynamically learned for each individual task to modulate θ.

4.3 Cross-Domain Intra-Type Adaptation

4.3.1 Setups

In this experiment, we investigate the problem of cross-domain intra-type adaptation, which aims to identify the same entity types (intra-type) across different domains (cross-domain). We use the Automatic Content Extraction 2005 (ACE2005) dataset[2], which consists of six domains: Broadcast Conversations (BC), Broadcast News (BN), Conversational Telephone Speech (CTS), Newswire (NW), Usenet (UN), and Weblog (WL). In each domain, there are 7 coarse-grained entity types and 54 fine-grained subtypes annotated in the corpus. We use the fine-grained annotations in this experiment, although some studies [66], [67] report high performance scores on the coarse-grained types. On the other hand, ACE2005 is annotated with nested named entities. For example, the sentence "*Orders went out today to*

2. https://catalog.ldc.upenn.edu/LDC2006T06

TABLE 3
Cross-domain intra-type adaptation performance (average F1 score with 95% confidence intervals on a set of 1000 randomly constructed tasks) on the test sets of UN, CTS and WL domains.

Methods	BC → UN: 5-way 1-shot	BC → UN: 5-way 5-shot	BN → CTS: 5-way 1-shot	BN → CTS: 5-way 5-shot	NW → WL: 5-way 1-shot	NW → WL: 5-way 5-shot
Dynamic Token Representation: Contextualized Language Model Embeddings + CRF						
GPT2	$16.53 \pm 0.73\%$	$17.08 \pm 0.71\%$	$31.12 \pm 0.77\%$	$32.69 \pm 0.79\%$	$14.96 \pm 0.52\%$	$15.51 \pm 0.58\%$
Flair	$14.12 \pm 0.50\%$	$14.96 \pm 0.56\%$	$34.79 \pm 0.81\%$	$37.03 \pm 0.87\%$	$15.10 \pm 0.61\%$	$15.74 \pm 0.63\%$
ELMo	$17.05 \pm 0.61\%$	$17.61 \pm 0.66\%$	$37.10 \pm 0.91\%$	$38.52 \pm 0.95\%$	$16.88 \pm 0.54\%$	$17.77 \pm 0.59\%$
BERT	$17.57 \pm 0.62\%$	$18.20 \pm 0.68\%$	$34.37 \pm 0.85\%$	$36.28 \pm 0.90\%$	$15.28 \pm 0.58\%$	$16.29 \pm 0.57\%$
XLNet	$16.12 \pm 0.69\%$	$17.94 \pm 0.72\%$	$29.32 \pm 0.73\%$	$34.31 \pm 0.86\%$	$16.81 \pm 0.44\%$	$17.56 \pm 0.51\%$
Static Token Representation: GloVe + CNN						
FineTune	$16.60 \pm 0.83\%$	$17.49 \pm 0.84\%$	$24.19 \pm 0.52\%$	$24.37 \pm 0.54\%$	$17.28 \pm 0.75\%$	$17.48 \pm 0.75\%$
ProtoNet	$17.46 \pm 0.71\%$	$17.98 \pm 0.67\%$	$28.38 \pm 0.75\%$	$30.55 \pm 0.71\%$	$19.39 \pm 0.59\%$	$20.46 \pm 0.64\%$
MAML	$17.93 \pm 0.68\%$	$18.68 \pm 0.59\%$	$30.57 \pm 0.68\%$	$31.78 \pm 0.83\%$	$22.87 \pm 0.68\%$	$27.83 \pm 0.59\%$
SNAIL	$18.45 \pm 0.83\%$	$20.43 \pm 0.74\%$	$36.19 \pm 0.81\%$	$37.61 \pm 0.68\%$	$25.38 \pm 0.63\%$	$29.92 \pm 0.75\%$
FewNER (ours)	$\mathbf{21.65 \pm 0.61\%}$	$\mathbf{25.87 \pm 0.57\%}$	$\mathbf{39.66 \pm 0.75\%}$	$\mathbf{45.65 \pm 0.66\%}$	$\mathbf{31.93 \pm 0.77\%}$	$\mathbf{38.66 \pm 0.73\%}$

deploy 17,000 U.S. Army soldiers in the Persian Gulf region" is originally annotated as [17,000 U.S. Army soldiers]$_{\text{PER:Group}}$, [U.S.]$_{\text{GPE:Nation}}$, [U.S. Army]$_{\text{ORG:Government}}$, [the Persian Gulf region]$_{\text{LOC:Region\_International}}$, [Persian Gulf]$_{\text{LOC:water-Body}}$. We only keep the innermost entities for all nested entities. That is, the above example is preprocessed as [U.S.]$_{\text{GPE-Nation}}$ and [Persian Gulf]$_{\text{LOC:Water-Body}}$ in our experiments. We design three cross-domain adaptations in total: BC → UN, BN → CTS and NW → WL. The data split ratio is $8/1/1$ for training, validation and testing. Note that the entity types for testing have already appeared during training, but the domains for the two are different.

4.3.2 Experimental Results

Table 3 reports the experimental results of cross-domain intra-type adaptation. We make the following observations:

First, FEWNER achieves state-of-the-art performance by significant margins, outperforming all baseline methods. More specifically, our model outperforms the second best method (i.e., SNAIL) by relative F1 improvements of 17.34%, 9.59% and 25.81% in the 1-shot setting, and 26.63%, 21.38% and 29.21% in the 5-shot setting, for BC → UN, BN → CTS and NW → WL adaptations, respectively. Similar observations in Section 4.2.2 hold for the trend in dynamic-representation-based methods and static-representation-based methods.

Second, although the entity types for testing are seen during training, the cross-domain intra-type adaptation under the N-way K-shot setting is still a challenging problem. Because the usage of entities varies from domain to domain, a fast context adaptation strategy is required for cross-domain NER. Compared with these baseline methods, FEWNER is more effective in learning task-specific context parameters, resulting in a better performance.

Third, for the three adaptation experiments we designed, BC → UN yields the worst performance (i.e., $21.65 \pm 0.61\%$ and $25.87 \pm 0.57\%$) and BN → CTS yields the best performance (i.e., $39.66 \pm 0.75\%$ and $45.65 \pm 0.66\%$). This is because the domain difference between UN (Usenet) and BC (Broadcast Conversations) is greater than the difference between BN (Broadcast News) and CTS (Conversational Telephone Speech).

4.4 Cross-Domain Cross-Type Adaptation

4.4.1 Setups

In this experiment, we investigate a more difficult scenario: cross-domain cross-type adaptation, which means that a model is trained on a source domain $\mathcal{A}$ and then adapted to an unseen target domain $\mathcal{B}$, which also has a different entity type space from $\mathcal{A}$. In total, we design three adaptations: GENIA (medical domain with 36 types) → BioNLP13CG (medical domain with 16 types), OntoNotes (various domains with 18 types) → BioNLP13CG (medical domain with 16 types), OntoNotes (various domains with 18 types) → FG-NER (newswire domain with 200 types). Training sets are taken from the source domains. Note that although GENIA and BioNLP13CG are both from medical text, we consider them as different domains because the annotation and entity types are different. We hold out 20% of target domain data as the validation set, and test models on the remaining 80% of target domain data (i.e., test set). We report the average F1 scores with 95% confidence intervals on a random set of 1000 tasks from the test sets.

4.4.2 Experimental Results

Table 4 reports the experimental results of cross-domain cross-type adaptation. We make the following observations:

First, FEWNER achieves state-of-the-art performance by significant margins, outperforming all baseline methods. More specifically, our model outperforms the second best method (i.e., SNAIL) by relative F1 improvements of 35.06%, 32.36% and 37.95% in the 1-shot setting, and 43.95%, 35.85% and 33.95% in the 5-shot setting, for GENIA → BioNLP13CG, OntoNotes → BioNLP13CG and OntoNotes → FG-NER adaptations, respectively.

Second, the performance in the 5-shot setting is slightly better than the performance in the 1-shot setting. The relatively poor performance indicates the difficulty in cross-domain cross-type adaptation for NER.

Third, the OntoNotes → BioNLP13CG adaptation yields the worst performance (i.e., $13.09 \pm 0.63\%$ and $15.46 \pm 0.62\%$) and the OntoNotes → FG-NER adaption yields the best performance (i.e., $28.06 \pm 1.12\%$ and $32.87 \pm 1.41\%$). Compared to the OntoNotes → BioNLP13CG adaptation, the GENIA → BioNLP13CG delivers a better performance. This

TABLE 4
Cross-domain cross-type adaptation performance (average F1 score with 95% confidence intervals on a set of 1000 randomly constructed tasks) on BioNLP13CG, BioNLP13CG and FG-NER.

Methods	GENIA → BioNLP13CG: 5-way 1-shot	GENIA → BioNLP13CG: 5-way 5-shot	OntoNotes → BioNLP13CG: 5-way 1-shot	OntoNotes → BioNLP13CG: 5-way 5-shot	OntoNotes → FG-NER: 5-way 1-shot	OntoNotes → FG-NER: 5-way 5-shot
Dynamic Token Representation: Contextualized Language Model Embeddings + CRF						
GPT2	$10.31 \pm 0.41\%$	$12.17 \pm 0.49\%$	$9.68 \pm 0.41\%$	$10.23 \pm 0.42\%$	$14.67 \pm 0.73\%$	$14.51 \pm 0.94\%$
Flair	$10.53 \pm 0.33\%$	$12.49 \pm 0.45\%$	$8.37 \pm 0.31\%$	$9.15 \pm 0.33\%$	$13.44 \pm 0.76\%$	$15.18 \pm 0.87\%$
ELMo	$10.39 \pm 0.41\%$	$11.45 \pm 0.42\%$	$10.76 \pm 0.55\%$	$11.85 \pm 0.59\%$	$15.15 \pm 0.77\%$	$16.08 \pm 0.97\%$
BERT	$13.36 \pm 0.53\%$	$15.15 \pm 0.61\%$	$9.15 \pm 0.29\%$	$9.98 \pm 0.31\%$	$14.14 \pm 0.71\%$	$15.86 \pm 0.89\%$
XLNet	$9.15 \pm 0.32\%$	$10.59 \pm 0.37\%$	$7.30 \pm 0.34\%$	$7.72 \pm 0.34\%$	$14.13 \pm 0.72\%$	$15.97 \pm 0.88\%$
Static Token Representation: GloVe + CNN						
FineTune	$13.86 \pm 0.64\%$	$13.96 \pm 0.65\%$	$6.16 \pm 0.35\%$	$6.53 \pm 0.38\%$	$13.70 \pm 0.85\%$	$14.81 \pm 0.93\%$
ProtoNet	$14.05 \pm 0.57\%$	$15.38 \pm 0.49\%$	$8.34 \pm 0.47\%$	$8.93 \pm 0.43\%$	$15.45 \pm 0.74\%$	$16.78 \pm 0.83\%$
MAML	$14.98 \pm 0.63\%$	$17.34 \pm 0.53\%$	$9.22 \pm 0.38\%$	$10.57 \pm 0.34\%$	$16.82 \pm 0.74\%$	$18.34 \pm 0.92\%$
SNAIL	$16.63 \pm 0.59\%$	$19.41 \pm 0.63\%$	$9.89 \pm 0.33\%$	$11.38 \pm 0.56\%$	$20.34 \pm 0.76\%$	$24.54 \pm 0.89\%$
FewNER (ours)	$\mathbf{22.46 \pm 0.61\%}$	$\mathbf{27.94 \pm 0.52\%}$	$\mathbf{13.09 \pm 0.63\%}$	$\mathbf{15.46 \pm 0.62\%}$	$\mathbf{28.06 \pm 1.12\%}$	$\mathbf{32.87 \pm 1.41\%}$

TABLE 5
Ablation study on intra-domain cross-type adaptation with the NNE data. ↑ and ↓ indicate absolute F1 changes (improvement and degradation).

	1-shot	5-shot
FewNER (ours)	$23.74 \pm 0.65\%$	$29.50 \pm 0.68\%$
Conditioning method A	$-2.34\% \downarrow$	$-3.43\% \downarrow$
Remove character CNN	$-15.56\% \downarrow$	$-18.73\% \downarrow$
Inner gradient steps: 4	$+0.35\% \uparrow$	$+0.79\% \uparrow$
Inner gradient steps: 6	$+0.78\% \uparrow$	$+0.95\% \uparrow$
Inner gradient steps: 8	$+1.02\% \uparrow$	$+1.47\% \uparrow$
Dimensions of φ: 128	$-2.45\% \downarrow$	$-3.74\% \downarrow$
Dimensions of φ: 512	$-4.32\% \downarrow$	$-3.68\% \downarrow$
Training "way": 3	$+0.46\% \uparrow$	$+0.93\% \uparrow$
Training "way": 10	$-1.24\% \downarrow$	$-1.89\% \downarrow$
Training "way": 15	$-2.31\% \downarrow$	$-3.25\% \downarrow$

is reasonable because GENIA and BioNLP13CG are both from the medical domain.

4.5 Further Analysis

4.5.1 Ablation Study

Table 5 reports an ablation analysis on intra-domain cross-type adaptation with the NNE dataset. FewNER is our proposed approach with conditioning method B, 2 inner gradient steps, 256 dimension of φ and 5 ways. There are five groups of variants: adopting the conditioning method A described in Section 3.2.4; removing the character-level CNN layer; adopting 4, 6 and 8 inner gradient steps during training; adopting 128 and 512 dimensions of φ; adopting 3, 10 and 15 ways during training.

We can observe that the conditioning method B (i.e., FiLM in our implementation) yields a slightly better performance than method A (i.e., concatenation). Removing the character CNN layer results in a huge degradation performance (-15.56% and -18.73%). This clearly showcases that the character-level representations play a very important role in few-shot NER. This is attributed to the fact that words in named entities are prone to out-of-training-vocabulary (OOTV) tokens; however, the character-level representations can effectively handle OOTV tokens in NER adaptation. Increasing the inner gradient steps can slightly improve the performance, at the expense of more computations. The dimensions of the context parameters φ and the number of training "ways" are important considerations in order to achieve good results for few-shot NER.

4.5.2 Time Consuming Analysis

As show in Algorithm 1, FewNER consists fo two main phases: a training procedure and an adapting procedure. Now we empirically investigate the time consumption of these two phases on one single GPU (Tesla V100) for the intra-domain cross-type adaptation on NNE. In particular, the size of inner gradient steps is 2 for training and 8 for testing. The size of a meta batch for outer loops is 8. During the training phase, our experimental study shows that each inner loop (i.e., "k" at Line 7 in Algorithm 1) needs 0.04 second, and the total outer loops (i.e., all "$\mathcal{T}_i$" at Line 5 in Algorithm 1) need 2.19 seconds for the 5-way 1-shot configuration. For the 5-way 1-shot configuration, each inner loop takes 0.04 second and outer loops totally needs 3.44 seconds.

During the adapting phase, the learned meta-knowledge θ_{Meta} is fixed. Our proposed FewNER only needs to update the task-specific parameters φ from the current task $\mathcal{T}_j$. For each test task, each inner loop takes 0.04 second for both 5-way 1-shot and 5-way 5-shot configurations. Evaluating a task needs 0.36 second and 0.51 second for 5-way 1-shot and 5-way 5-shot configurations, respectively. Note that the time consumption is linearly proportional to the data size for both training and adapting phases. In summary, our approach FewNER has very high-computational efficiency because it updates only a small set of parameters in a low-dimensional space when adapting new tasks.

4.5.3 Qualitative Analysis

Table 6 shows 3 positive examples and 6 negative examples produced by FewNER under the 5-way 1-shot setting. For the example in NNE → NNE, FewNER successfully identifies the three entities which have novel types (i.e., unseen "ProductFood" and "Country"). For the example in BC → UN, FewNER successfully identifies the new instances ⟦Larry King⟧, ⟦last night⟧ and ⟦the BTK killer⟧, but the types of "Individual" and "Time" have already appeared in BC. For the example in GENIA → BioNLP13CG, FewNER successfully identifies two entities of "Genes" and one entity of "Cancer", while both types do not exist in GENIA.

TABLE 6
Positive and negative examples under the 5-way 1-shot setting. The results produced by FEWNER are marked with ⟦ ⟧. The green and red highlights indicate a correct and incorrect (missing) result, respectively.

Adaptation	5-way 1-shot NER Results by FEWNER	Correct
NNE → NNE	Some fruit visionaries say the ⟦Fuji⟧$_{\text{ProductFood}}$ could someday tumble the ⟦Red Delicious⟧$_{\text{ProductFood}}$ from the top of ⟦America⟧$_{\text{Country}}$'s apple heap .	✓
FG-NER → FG-NER	The manager of the " ⟦Oasis Coordinator Project⟧$_{\text{Product}}$ " is ⟦Enbridge⟧$_{\text{Government}}$.	✗
GENIA → GENIA	An overexpression of ⟦p50⟧$_{\text{ProteinSubunit}}$ has been described in follicular dendritic cells (⟦FDC⟧$_{\text{CellType}}$) .	✗
BC → UN	Watching ⟦Larry King⟧$_{\text{Individual}}$ ⟦last night⟧$_{\text{Time}}$ about ⟦the BTK killer ⟧$_{\text{Individual}}$	✓
BN → CTS	⟦Peterson Trial Scott Peterson⟧$_{\text{Individual}}$ has been found guilty of murdering ⟦his⟧$_{\text{Individual}}$ wife ⟦Laci⟧$_{\text{Individual}}$	✗
NW → WL	⟦The Virginia Economic Developers Association⟧$_{\text{Government}}$ has not taken a stance on the bill , but the organization 's leadership is expected to meet ⟦tomorrow⟧$_{\text{Time}}$ to discuss it .	✗
GENIA → BioNLP.	Expression of ⟦KISS1⟧$_{\text{Gene}}$ and ⟦MMP-9 ⟧$_{\text{Gene}}$ in ⟦non-small cell lung cancer⟧$_{\text{Cancer}}$	✓
Onto. → BioNLP.	the function of ⟦c-Ski⟧$_{\text{Gene}}$ in human diffuse - type gastric carcinoma ⟦OCUM - 2MLN cells⟧$_{\text{Cell}}$	✗
Onto. → FG-NER	Historians regard ⟦Portrait Of A Young Man⟧$_{\text{Picture}}$ as the most important painting missing since ⟦World War II⟧$_{\text{War}}$.	✗

For the negative example in FG-NER → FG-NER, FEWNER successfully identifies the mention of ⟦Enbridge⟧, but fails to identity its semantic type "Government". Typing is a challenging task because there are 200 types in FG-NER and the context of the entity is very short. For the example in GENIA → GENIA, FEWNER misses one entity ⟦follicular dendritic cells⟧$_{\text{CellType}}$. For the example in BN → CTS, FEWNER wrongly detects the boundaries of two entities, "⟦Peterson⟧$_{\text{Individual}}$ Trial ⟦Scott Peterson⟧$_{\text{Individual}}$". For the example in NW → WL, FEWNER misses the coreference, "⟦the organization⟧$_{\text{Government}}$". For the negative example in OntoNotes → BioNLP13CG, FEWNER misses the entity "⟦gastric carcinoma⟧$_{\text{Cancer}}$". For the negative example in OntoNotes → FG-NER, FEWNER misses the entity "⟦Historians⟧$_{\text{PositionVocation}}$". In summary, we observe that FEWNER misses some entities and wrongly detects the boundaries of entities. The different annotation criteria and writing formats in different domains are the key factors affecting few-shot transferability. The qualitative analysis indicates that few-shot learning in NER has important practical implications, e.g., transferring the 18 types in OntoNotes to 200 fine-grained types in FG-NER with few samples. We argue that such coarse-to-fine adaptation can benefit many downstream intelligence systems, for example, ⟦Portrait Of A Young Man⟧ is labeled as "Picture" rather than "Other", and ⟦World War II⟧ is labeled as "War" rather than "Miscellaneous".

5 Discussion and Conclusion

Difficulty and Challenge. Few-shot learning has been well studied in computer vision [21], [25], however, it has not been explored in sequence labeling. Our study is the first attempt of adopting few-shot learning in sequence labeling tasks. We formally define the N-Way K-Shot problem in NER (see Section 3.1). From our experimental results (see Sections 4.2.1, 4.3.1, 4.4.1), the performance of few-shot in NER is not comparable with the performance in text relation extraction [20] and image classification [51]. Our study reveals that few-shot learning in sequence labeling is a difficult NLP task. Existing powerful pre-trained language models (e.g., BERT and GPT2) cannot solve the problem well. The main reason lies in that sequence labeling is different from the conventional classification task, where each member is independently classified into a category without taking sequence dependency into account. Therefore, transferring sequence dependency in few-shot learning is more difficult to transferring the single token patterns in classification problem. We argue that few-shot learning in sequence labeling is far from being solved. We envision that more studies will be devoted to investigating few-shot learning in sequence labeling after our first attempt in this study. In future, we expect that more cut-edging technologies such as incremental learning and graph neural networks will be explored to push boundaries of this task.

Implication. We believe that named entity recognition is a fundamental problem in NLP. Although our experiments are conducted on NER, our approach can be easily extended to other sequence labeling tasks, such as part-of-speech tagging and slot filling. Few-shot learning in sequence labeling will benefit a wide variety of real world NLP applications. For example, it can save much human-annotation cost for many low-resource scenarios, such as domain-specific NER (e.g., medical and social media domains), resource scarce language tasks (e.g., Portuguese NER and machine translation). Moreover, few-shot learning can make sequence labeling systems work as human beings where we have a remarkable ability to quickly grasp new concepts from a very small number of examples or a limited amount of experience, leveraging prior knowledge and context. Our attempt in this study is the way to the artificial general intelligence (AGI), which effectively matches human intelligence. Likewise, AGI also needs to be able to transfer learning from one environment to another, use common sense, work collaboratively with other machine and human stakeholders, and attain consciousness.

Conclusion. In this paper, we are the first to investigate the N-way K-shot setting in few-shot NER. We proposed FEWNER, a novel meta-learning approach for few-shot NER, which can effectively adapt to new tasks by updating a small set of low-dimensional parameters, rather than the entire network at test time. FEWNER is simpler and more efficient than recent meta-learning approaches in adapting to unseen tasks with few gradient steps using little data. Finally, we conducted three adaptation experiments and the experimental results demonstrated the effectiveness of our proposed approach.

REFERENCES

[1] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Trans. Knowl. Data Eng.*, 2020.

[2] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[3] J. Guo, G. Xu, X. Cheng, and H. Li, "Named entity recognition in query," in *SIGIR*, 2009, pp. 267–274.

[4] C. Lee, Y. Hwang, and M. Jang, "Fine-grained named entity recognition and relation extraction for question answering," in *SIGIR*, 2007, pp. 799–800.

[5] X. Ma and E. H. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," in *ACL*, 2016, pp. 1064–1074.

[6] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *NAACL-HLT*, 2018, pp. 2227–2237.

[7] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *CoRR*, vol. abs/1508.01991, 2015.

[8] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.

[9] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *NAACL-HLT*, 2016, pp. 260–270.

[10] L. Liu, J. Shang, X. Ren, F. F. Xu, H. Gui, J. Peng, and J. Han, "Empower sequence labeling with task-aware neural language model," in *AAAI*, 2018, pp. 5253–5260.

[11] A. Ghaddar and P. Langlais, "Robust lexical features for improved neural network named-entity recognition," in *COLING*, 2018, pp. 1896–1907.

[12] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010.

[13] Y. Lin, S. Yang, V. Stoyanov, and H. Ji, "A multi-lingual multi-task architecture for low-resource sequence labeling," in *ACL*, 2018, pp. 799–809.

[14] H. D. III, "Frustratingly easy domain adaptation," in *ACL*, 2007, pp. 256–263.

[15] J. Li, D. Ye, and S. Shang, "Adversarial transfer for named entity boundary detection with pointer networks," in *IJCAI*, 2019, pp. 5053–5059.

[16] J. M. Giorgi and G. D. Bader, "Transfer learning for biomedical named entity recognition with neural networks," *Bioinformatics*, vol. 34, no. 23, pp. 4087–4094, 2018.

[17] C. Jia, L. Xiao, and Y. Zhang, "Cross-domain NER using cross-domain language modeling," in *ACL*, 2019, pp. 2464–2474.

[18] B. Y. Lin and W. Lu, "Neural adaptation layers for cross-domain named entity recognition," in *EMNLP*, 2018, pp. 2012–2022.

[19] J. T. Zhou, H. Zhang, D. Jin, H. Zhu, M. Fang, R. S. M. Goh, and K. Kwok, "Dual adversarial neural transfer for low-resource named entity recognition," in *ACL*, 2019, pp. 3461–3471.

[20] 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 *EMNLP*, 2018, pp. 4803–4809.

[21] C. Finn, A. Rajeswaran, S. M. Kakade, and S. Levine, "Online meta-learning," in *ICML*, 2019, pp. 1920–1930.

[22] A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine, "Meta-learning with implicit gradients," *CoRR*, vol. abs/1909.04630, 2019.

[23] Y. Wang, R. B. Girshick, M. Hebert, and B. Hariharan, "Low-shot learning from imaginary data," in *CVPR*, 2018, pp. 7278–7286.

[24] J. Schmidhuber, "On learning how to learn learning strategies," 1995.

[25] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017, pp. 1126–1135.

[26] J. Yoon, T. Kim, O. Dia, S. Kim, Y. Bengio, and S. Ahn, "Bayesian model-agnostic meta-learning," in *NIPS*, 2018, pp. 7343–7353.

[27] H. Yao, Y. Wei, J. Huang, and Z. Li, "Hierarchically structured meta-learning," in *ICML*, 2019, pp. 7045–7054.

[28] L. M. Zintgraf, K. Shiarlis, V. Kurin, K. Hofmann, and S. Whiteson, "Fast context adaptation via meta-learning," in *ICML*, 2019, pp. 7693–7702.

[29] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," in *ICLR*, 2018.

[30] O. Etzioni, M. J. Cafarella, D. Downey, A. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, "Unsupervised named-entity extraction from the web: An experimental study," *Artif. Intell.*, vol. 165, no. 1, pp. 91–134, 2005.

[31] S. Zhang and N. Elhadad, "Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts," *Journal of Biomedical Informatics*, vol. 46, no. 6, pp. 1088–1098, 2013.

[32] Z. Ji, A. Sun, G. Cong, and J. Han, "Joint recognition and linking of fine-grained locations from tweets," in *WWW*, 2016, pp. 1271–1281.

[33] Y. Li, K. Bontcheva, and H. Cunningham, "SVM based learning system for information extraction," in *First International Workshop on Deterministic and Statistical Methods in Machine Learning*, 2004, pp. 319–339.

[34] R. Malouf, "Markov models for language-independent named entity recognition," in *COLING*, 2002.

[35] E. Strubell, P. Verga, D. Belanger, and A. McCallum, "Fast and accurate entity recognition with iterated dilated convolutions," in *EMNLP*, 2017, pp. 2670–2680.

[36] P. Li, R. Dong, Y. Wang, J. Chou, and W. Ma, "Leveraging linguistic structures for named entity recognition with bidirectional recursive neural networks," in *EMNLP*, 2017, pp. 2664–2669.

[37] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT*, 2019, pp. 4171–4186.

[38] J. Zhuo, Y. Cao, J. Zhu, B. Zhang, and Z. Nie, "Segment-level sequence modeling using gated recursive semi-markov conditional random fields," in *ACL*, 2016, pp. 1413–1423.

[39] J. Li, A. Sun, and S. R. Joty, "Segbot: A generic neural text segmentation model with pointer network," in *IJCAI*, 2018, pp. 4166–4172.

[40] Y. Shen, H. Yun, Z. C. Lipton, Y. Kronrod, and A. Anandkumar, "Deep active learning for named entity recognition," in *ICLR*, 2018.

[41] X. Wang, Y. Zhang, X. Ren, Y. Zhang, M. Zitnik, J. Shang, C. Langlotz, and J. Han, "Cross-type biomedical named entity recognition with deep multi-task learning," *Bioinformatics*, vol. 35, no. 10, pp. 1745–1752, 2019.

[42] G. Beryozkin, Y. Drori, O. Gilon, T. Hartman, and I. Szpektor, "A joint named-entity recognizer for heterogeneous tag-sets using a tag hierarchy," in *ACL*, 2019, pp. 140–150.

[43] 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," *CoRR*, vol. abs/1911.06161, 2019.

[44] Y. Hou, Z. Zhou, Y. Liu, N. Wang, W. Che, H. Liu, and T. Liu, "Few-shot sequence labeling with label dependency transfer and pair-wise embedding," *arXiv preprint arXiv:1906.08711*, 2019.

[45] Z. Yang, R. Salakhutdinov, and W. W. Cohen, "Transfer learning for sequence tagging with hierarchical recurrent networks," in *ICLR*, 2017.

[46] P. von Däniken and M. Cieliebak, "Transfer learning and sentence level features for named entity recognition on tweets," in *W-NUT*, 2017, pp. 166–171.

[47] J. Y. Lee, F. Dernoncourt, and P. Szolovits, "Transfer learning for named-entity recognition with neural networks," in *LREC*, 2017, pp. 4470–4473.

[48] B. Y. Lin and W. Lu, "Neural adaptation layers for cross-domain named entity recognition," in *EMNLP*, 2018, pp. 2012–2022.

[49] S. Thrun and L. Y. Pratt, Eds., *Learning to Learn*. Springer, 1998.

[50] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *NIPS*, 2016, pp. 3630–3638.

[51] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *NIPS*, 2017, pp. 4077–4087.

[52] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *ICLR*, 2017.

[53] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. P. Lillicrap, "Meta-learning with memory-augmented neural networks," in *ICML*, 2016, pp. 1842–1850.

[54] Y. Li, Y. Yang, W. Zhou, and T. M. Hospedales, "Feature-critic networks for heterogeneous domain generalization," in *ICML*, 2019, pp. 3915–3924.

[55] J. Gu, Y. Wang, Y. Chen, V. O. K. Li, and K. Cho, "Meta-learning for low-resource neural machine translation," in *EMNLP*, 2018, pp. 3622–3631.

[56] P. Huang, C. Wang, R. Singh, W. Yih, and X. He, "Natural language to structured query generation via meta-learning," in *NAACL-HLT*, 2018, pp. 732–738.
[57] K. Qian and Z. Yu, "Domain adaptive dialog generation via meta learning," in *ACL*, 2019, pp. 2639–2649.
[58] A. Obamuyide and A. Vlachos, "Model-agnostic meta-learning for relation classification with limited supervision," in *ACL*, 2019, pp. 5873–5879.
[59] Q. Guo, X. Qiu, P. Liu, Y. Shao, X. Xue, and Z. Zhang, "Star-transformer," in *NAACL-HLT*, J. Burstein, C. Doran, and T. Solorio, Eds., 2019, pp. 1315–1325.
[60] H. Yan, B. Deng, X. Li, and X. Qiu, "TENER: adapting transformer encoder for named entity recognition," *CoRR*, vol. abs/1911.04474, 2019.
[61] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville, "Film: Visual reasoning with a general conditioning layer," in *AAAI*, 2018, pp. 3942–3951.
[62] A. Fritzler, V. Logacheva, and M. Kretov, "Few-shot classification in named entity recognition task," in *SAC*, 2019, pp. 993–1000.
[63] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, 2019.
[64] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *COLING*, 2018, pp. 1638–1649.
[65] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *CoRR*, vol. abs/1906.08237, 2019.
[66] M. Ju, M. Miwa, and S. Ananiadou, "A neural layered model for nested named entity recognition," in *NAACL-HLT*, 2018, pp. 1446–1459.
[67] J. Fisher and A. Vlachos, "Merge and label: A novel neural network architecture for nested NER," in *ACL*, 2019, pp. 5840–5850.

```
@article{li20fewshot,
author    = {Jing Li and Billy Chiu and Shanshan Feng and Hao Wang},
title     = {Few-Shot Named Entity Recognition via Meta-Learning},
journal   = {IEEE Transactions on Knowledge and Data Engineering (TKDE)},
volume   = {34},
number  = {9},
pages = {4245--4256},
year   = {2022},
url    = {https://doi.org/10.1109/TKDE.2020.3038670},
doi   = {10.1109/TKDE.2020.3038670},
}
```