MetaNER: Named Entity Recognition with Meta-Learning

Jing Li[#], Shuo Shang[★,*] and Ling Shao[#]
[★]University of Electronic Science and Technology of China, Chengdu, China
[#]Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates
jingli.phd@hotmail.com; jedi.shang@gmail.com; ling.shao@inceptioniai.org

ABSTRACT

Recent neural architectures in named entity recognition (NER) have yielded state-of-the-art performance on single domain data such as newswires. However, they still suffer from (i) requiring massive amounts of training data to avoid overfitting; (ii) huge performance degradation when there is a domain shift in the data distribution between training and testing. In this paper, we investigate the problem of domain adaptation for NER under homogeneous and heterogeneous settings. We propose MetaNER, a novel meta-learning approach for domain adaptation in NER. Specifically, MetaNER incorporates meta-learning and adversarial training strategies to encourage robust, general and transferable representations for sequence labeling. The key advantage of MetaNER is that it is capable of adapting to new unseen domains with a small amount of annotated data from those domains. We extensively evaluate MetaNER on multiple datasets under homogeneous and heterogeneous settings. The experimental results show that MetaNER achieves state-of-the-art performance against eight baselines. Impressively, MetaNER surpasses the in-domain performance using only 16.17% and 34.76% of target domain data on average for homogeneous and heterogeneous settings, respectively.

CCS CONCEPTS

• **Information systems → Information extraction**; • **Computing methodologies → Transfer learning**.

KEYWORDS

Named entity recognition, domain adaptation, meta-learning

ACM Reference Format:
Jing Li[#], Shuo Shang[★,*] and Ling Shao[#]. 2020. MetaNER: Named Entity Recognition with Meta-Learning. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3366423.3380127

1 INTRODUCTION

Named entity recognition (NER) is a fundamental task in natural language processing (NLP), aiming at jointly resolving the boundaries and type of a named entity in text [26, 37]. 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 [15], automatic text summarization [38], question answering [24], machine translation [1], knowledge base construction [6], etc.

NER is typically framed as a sequence labeling problem whose goal is to assign a label to each word in a sentence. Recently, a significant amount of work [3, 12, 19, 23, 34, 35, 42] has been devoted to developing end-to-end neural-based sequence labeling models for NER. Despite their general success, they still suffer from (1) requiring a large amount of training data to avoid overfitting; (2) huge performance degradation when these is a domain shift in the data distribution between training and testing.

Domain adaptation (DA) has been studied as an effective solution to address the above data insufficiency and domain shift issues. For *homogeneous DA* in NER, the source and target domains have the same label space. The model trained on source domains can be directly transferred to target domains [40]. For *heterogeneous DA* in NER, it is more difficult to directly transfer models from source domains due to the label set discrepancy among different domains. Several studies have tackled heterogeneous DA in NER by learning correlations between label sets [22, 44] and fine-tuning the source model with target domain data [25, 32]. However, most works often require a large amount of annotated target domain data to achieve accurate domain adaptation. To make an NER system more broadly useful, it is crucial to reduce its training data requirement. This raises a natural question: *if we have sufficient annotated training data in multiple source domains, can we distill the knowledge and transfer it to help train models in a new target domain with few annotations from this new domain*?

Despite the difficulties arising from label discrepancy, it is possible to transfer knowledge between domains because named entities often share lexical and context features (*e.g.,* common vocabularies, similar word semantics and similar sentence syntaxes) [4]. As humans, we are able to quickly learn new things from a small number of examples or a limited amount of experience by leveraging prior knowledge [48]. In short, we *learn how to learn* much faster and more efficiently across various tasks. Meta-learning [8, 50] was proposed to mimic the human ability of acquiring multiple tasks simultaneously with minimum information. Recently, meta-learning has received resurgence in the context of few-shot learning [9, 63, 64]. Inspired by the essence of meta-learning [43], our key idea is to leverage the abundant data available in multiple resource domains to find a robust and general initialization that could be adapted to new unknown domains or novel entity categories with a small amount of new data. Figure 1 illustrates the idea of meta-learning in NER.

In this paper, we decompose any sequence labeling model into "sequence encoder + tag decoder". In such a way, different tag decoders are instantiated for heterogeneous domain adaptation. However, the sequence encoder is shared across all domains and is

[*] Corresponding author.

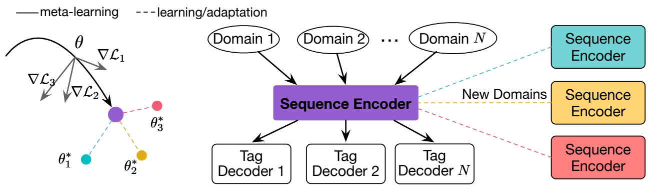


Figure 1: Illustration of the idea of meta-learning in NER.

designed to aggregate the meta-knowledge from multiple source domains so that it has maximal performance on a new domain with a small amount of data. Unfortunately, most existing meta-learning approaches are designed for classification under the few-shot setting (*i.e.,* N-way K-shot [57]) which is inapplicable for the sequence labeling problem where one sentence may have multiple training instances (*i.e.,* entities). Our research tries to seek a new meta-learning strategy which would be more suitable for encouraging robust and general representations in sequence labeling, rather than for few-shot learning.

More specifically, we propose MetaNER, a novel meta-learning approach for NER. First, inspired by the recent feature-critic network [31], MetaNER explicitly simulates the training-to-testing domain shift by splitting source domains into meta-training and meta-validation sets. MetaNER is trained in two alternating phases. In the meta-training phase, MetaNER minimizes the loss over all meta-training sets, resulting in a temporary model. In the meta-validation phase, the temporary model is evaluated on the meta-validation sets to minimize the domain divergence, enabling meta-knowledge transfer across different domains. Second, an adversarial network is used to improve model generalization. All together, these deliver a robust, general and transferable sequence encoder for both homogeneous and heterogeneous DA problems. In summary, the main contributions of this work are five-fold:

- To the best of our knowledge, we are the first to investigate the problem of transferring meta-knowledge learned from multiple source domains for sequence labeling in a meta-learning manner.
- We propose MetaNER, a novel meta-learning approach for NER. MetaNER incorporates meta-learning and adversarial training strategies to encourage robust, general and transferable representations which can be effectively adapted to new domains with a small amount of training data.
- We extensively evaluate MetaNER on six domains under homogeneous domain adaptation settings. The results show that MetaNER achieves state-of-the-art performance against eight baselines. Impressively, MetaNER surpasses the in-domain performance using only 16.17% of target domain data on average.
- We extensively evaluate MetaNER on nine domains under heterogeneous domain adaptation settings. The results show that MetaNER achieves state-of-the-art performance against baselines. MetaNER surpasses the in-domain performance using only 34.76% of target domain data on average.
- We conduct experiments to further analyze the parameter settings and architectural choices. We also present a study for qualitative analysis.

2 RELATED WORK

In this section, we review related work in three parts: named entity recognition, transfer learning in NER and meta-learning.

2.1 Named Entity Recognition

There are three common paradigms for NER [26]: *knowledge-based unsupervised* systems, *feature-based supervised* systems and *neural-based* systems. *Knowledge-based unsupervised* systems rely on lexical knowledge, including domain-specific gazetteers [6], and shallow syntactic knowledge [65]. These systems work very well when there is exhaustive lexicon. Due to domain- and language-specific rules and incomplete dictionaries, high precision and low recall are often observed from such systems. *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.* [20] designed 19 local features and 5 global features for location recognition in Tweets. Settles *et al.* [51] designed rich orthographic features and semantic features in biomedical named entity recognition. Based on manually crafted features, many algorithms have been applied in supervised NER, *e.g.,* the Support Vector Machine (SVM) [30], Hidden Markov Model (HMM) [36] and Conditional Random Field (CRF) [20].

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 [3], 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 [35, 42, 55]. Widely used context encoder architectures include CNNs [3], recurrent neural networks (RNNs) [19], recursive neural networks [29] and deep transformers [5]. At the top of the context encoder, a CRF layer [67], a pointer network [27, 28], or an RNN layer [53] is employed to make sequence label predictions.

2.2 Transfer Learning in NER

Transfer learning aims to perform a machine learning task on a target domain by taking advantage of knowledge learned from a source domain [41]. Several studies have already contributed effort to leveraging deep transfer learning for NER, and can be categorized along two lines: *multi-task learning based* approaches and *parameter-sharing* approaches. Multi-task learning based approaches leverage related tasks to improve the performance of all tasks at the same time. Wang *et al.* [59] collectively used the training data of different types of entities and improved the performance on each of them. Lin *et al.* [33] proposed a multi-lingual multi-task model to jointly solve multiple tasks in different languages for sequence labeling.

Most recent transfer approaches fall into the *parameter-sharing* category [17]. Commonly, different neural models [7, 13, 16, 21, 47, 61, 66] share certain parts of model parameters between the source domain and target domain. Yang *et al.* [62] proposed three different parameter-sharing models to investigate the transferability of different layers of representations. Pius and Mark [58] extended Yang's approach to allow joint training on an informal corpus, incorporating sentence-level feature representations. In addition, a fine-tuning strategy is usually used in parameter-sharing approaches. Some studies first train a model on source domains and then use the learned parameters to initialize a model on target domains. For example, Lee *et al.* [25] trained a neural model on a large dataset (MIMIC) and then fine-tuned it on smaller datasets (i2b2 2014). Other examples along this line can be found in [2, 32].

Our approach differs from these existing solutions in that (1) it aggregates meta-knowledge from multiple resource domains rather than a single one to increase the transferability; (2) it learns robust and general sequence representations for handling both homogeneous and heterogeneous adaptations.

2.3 Meta-Learning

Meta-learning (a.k.a. learning to learn) [50, 56] 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 [54, 57], memory-based methods [46, 49], and optimization-based methods [8, 64]. Some studies [9, 31, 45, 60] have applied meta-learning strategies for image classification in few-shot learning.

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.* [14] 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.* [18] proposed a method for query generation based on MAML [8], by reducing a regular supervised learning problem to the few-shot meta-learning scenario. Qian and Zhou [43] 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.* [39] 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.

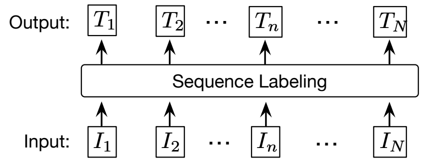
Different from the above studies on classification under the few-shot setting, our study focuses on sequence labeling in NER, where one sentence may have multiple training instances (*i.e.,* entities). In this paper, we are seeking a new meta-learning strategy which would be more suitable for encouraging robust representations in sequence labeling, rather than for few-shot learning. To the best of our knowledge, we are the first to attempt adopting meta-learning in sequence labeling.

3 PRELIMINARIES

In this section, we first introduce the sequence labeling problem and distinguish it from classification. Then, we briefly describe the Model-Agnostic Meta-Learning model and point out the difference from our scenario.

3.1 Sequence Labeling Problem

Named entity recognition is usually framed as a sequence labeling problem, which is illustrated, with the input and output, by the following figure:



Formally, given an input sequence $I = \{I_1, I_2, ..., I_n, ..., I_N\}$, the sequence labeling problem involves the algorithmic assignment of a categorical label to each member of I. This process produces an output sequence $T = \{T_1, T_2, ..., T_n, ..., T_N\}$. Sequence labeling is different from the conventional classification task, where each member is independently classified into a category without taking sequence dependency into account. How to model the sequence context and dependency has been a hot spot in the field of sequence labeling. For the NER task, the named entities can be obtained by extracting patterns from the produced output tags. For example, given the input sequence (*i.e.,* sentence) "*Michael Jordan was born in New York City*", we can get two entities (*i.e.,* Person: *Michael Jordan* and Location: *New York City*) from the corresponding tag sequence "*B-Person, E-Person, O, O, O, B-Location, I-Location, E-Location*"[1].

3.2 Model-Agnostic Meta-Learning

Model-Agnostic Meta-Learning (MAML) [8] provides a general approach to adapting parameters across different domains. MAML solves few-shot learning problems by learning a good parameter initialization. At test time, such an initialization can be fine-tuned with a few gradient steps using a limited amount of training examples from target domains.

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 training set and a validation set. In the N-way K-shot [57] classification, the training instances are sampled with K labeled examples from each of N classes, the model changes parameters θ to θ'_i by gradient descent:

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

where α is a universal learning rate, and $\mathcal{L}_{\mathcal{T}_i}$ 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:

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

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:

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

where β is the learning rate of meta optimization.

[1]BIOES stands for Begin, Inside, Outside, End, Single.

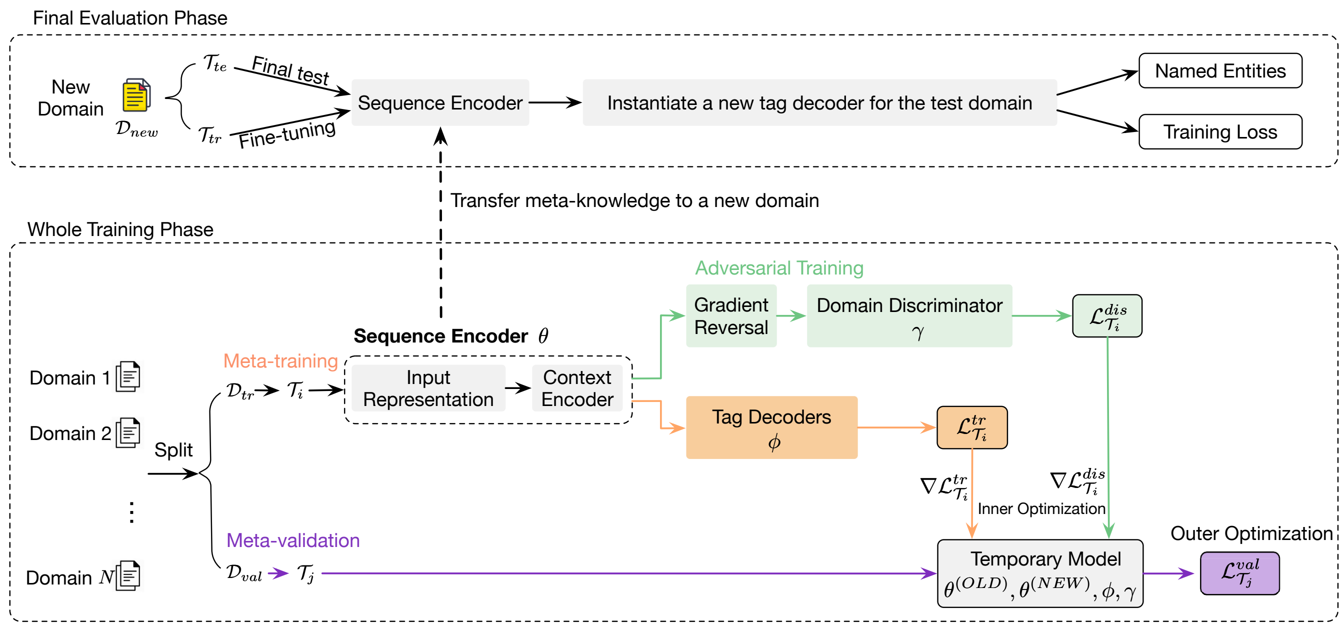


Figure 2: An overview of our proposed MetaNER (best viewed in color). MetaNER explicitly simulates domain shift ($\mathcal{D}_{tr} \rightarrow \mathcal{D}_{val}$) during the training process. In the meta-training phase, a temporary model $(\theta^{(OLD)}, \theta^{(NEW)}, \phi, \gamma)$ is learned from $\mathcal{D}_{tr}$. In the meta-validation phase, the base model is updated by gradient descent with respect to the parameters (θ, ϕ, γ) on $\mathcal{D}_{val}$. In the final evaluation phase, the learned sequence encoder is fine-tuned on $\mathcal{T}_{tr}$ and tested on $\mathcal{T}_{te}$ from a unseen domain $\mathcal{D}_{new}$.

Note that the objective of MAML is designed for few-shot classification under the problem setting of N-way K-shot. As discussed in Section 3.1, sequence labeling is commonly not a classification problem. This is because a training example (*i.e.,* a sentence) in sequence labeling may have multiple entities whose number and classes are not known in advance. Therefore, the N-way K-shot setup is inapplicable for the sequence labeling problem. On the other hand, the training set and validation set (also known as query set) in MAML are both from the same task in a single domain. In this paper, we seek a more suitable optimization objective for sequence labeling scenarios. More specifically, we use meta-learning to encourage robust and general representations for adapting to new domains with a small amount of annotated training data.

4 METANER: NER WITH META-LEARNING

In this section, we first define the problem of meta-learning for named entity recognition. Then we present a layer-by-layer description of MetaNER.

4.1 Problem Statement

Let $\mathcal{D}_s = \{\mathcal{D}_1, ..., \mathcal{D}_n, ..., \mathcal{D}_N\}$ be N source domains in the training phase, where $\mathcal{D}_n$ is the n-th source domain containing annotated data $(\mathcal{X}_n, \mathcal{Y}_n)$. Meanwhile, there are K target domains $\mathcal{D}_t = \{\mathcal{D}_1, ..., \mathcal{D}_k, ..., \mathcal{D}_K\}$, which are unseen in $\mathcal{D}_s$. Likewise, $\mathcal{D}_k$ is the k-th target domain containing annotated data $(\mathcal{X}_k, \mathcal{Y}_k)$. For the NER task, the input space $\mathcal{X}_n$ is raw text (*i.e.,* sentences) and the label space $\mathcal{Y}_n$ is the corresponding tag sequence that indicates the start and end positions of a named entity with the BIOES schema. For *homogeneous* NER, all the source domains and the target domains share the same label space. For *heterogeneous* NER, the domains can have different, and even completely, disjoint label spaces.

To make domain adaptation possible in NER, we generally decompose any sequence labeling model into two unified modules: a *sequence encoder* (learnable parameters θ) and a *tag decoder* (learnable parameters ϕ). Our ultimate goal is to learn a meta-knowledge learner for the sequence encoder by leveraging sufficient source data $\mathcal{D}_s$. Given a new unseen domain from $\mathcal{D}_{new}$ (which can be either homogeneous or heterogeneous), the new learning task of NER can be solved by fine-tuning the learned sequence encoder (domain-invariant parameters) and a new tag decoder (domain-specific parameters) with only a small number of training samples. The meta-knowledge learner of the sequence encoder should aggregate the knowledge learned from multiple domains in $\mathcal{D}_s$, resulting in more robust, general and transferable representations, which can be broadly adapted to achieve optimum performance in $\mathcal{D}_{new}$ with as little as possible.

4.2 The MetaNER Approach

4.2.1 Overview of MetaNER. Figure 2 shows an overview of our proposed MetaNER, which consists of a training phase and an evaluation phase. In the training phase, we adopt a meta-learning strategy to distill meta-knowledge from a number of source domains. During each iteration, we randomly split all source domains into a meta-training set $\mathcal{D}_{tr}$ and a meta-validation set $\mathcal{D}_{val}$, where $\mathcal{D}_s = \mathcal{D}_{tr} \cup \mathcal{D}_{val}$ and $\mathcal{D}_{tr} \cap \mathcal{D}_{val} = \emptyset$. A meta-training task $\mathcal{T}_i$ is

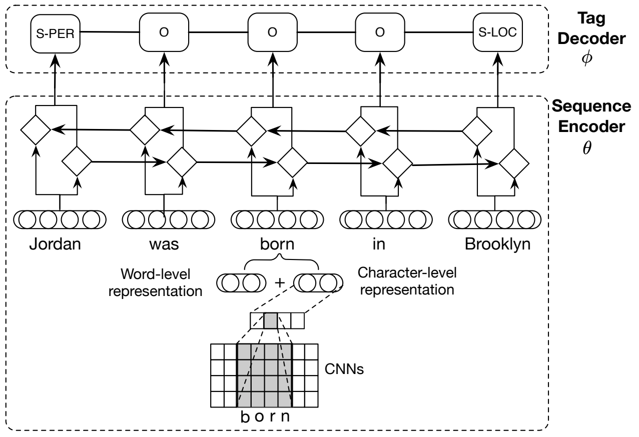


Figure 3: CNN-BiGRU-CRF for sequence labeling. We decompose it into two modules: a sequence encoder θ and a tag decoder ϕ.

sampled from $\mathcal{D}_{tr}$ and is composed of n instances from a particular domain. Likewise, a meta-validation task $\mathcal{T}_j$ is sampled from $\mathcal{D}_{val}$. The validation errors on $\mathcal{D}_{val}$ should be considered to improve the transferability of the model. In short, the meta-learning strategy aims to encourage the model to learn good parameters that can be adapted to a new domains with as little data as possible. We also adopt an adversarial training strategy to improve model generalization. The adversarial network ensures that the intermediate representations from the sequence encoder can mislead the domain discriminator and correctly guide the tag decoder prediction, while the domain discriminator tries its best to correctly determine the domain class of each training instance.

In the final evaluation phase, the meta-knowledge learned by the sequence encoder can be applied to new domains. Given a new domain $\mathcal{D}_{new} = \{\mathcal{T}_{tr}, \mathcal{T}_{te}\}$, the learned sequence encoder and a new tag decoder are fine-tuned on $\mathcal{T}_{tr}$ and finally tested on $\mathcal{T}_{te}$. Next, we briefly introduce the sequence labeling model (*i.e.,* "sequence encoder + tag decoder"). Then, we describe the adversarial training strategies and meta-learning strategy in detail.

4.2.2 *Sequence Labeling Model.*

Figure 3 shows the architecture of CNN-BiGRU-CRF, which uses 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) to produce the tag sequence. Specifically, we decompose the CNN-BiGRU-CRF model into two modules: a *sequence encoder* with parameters θ and a *tag decoder* with parameters ϕ.

The sequence encoder consists of two layers: the input representation and context encoder. The input representation in our study consists of character-level and word-level representations. Given an input sentence $\boldsymbol{W} = (W_1, W_2, \ldots, W_L)$ of length L, $\boldsymbol{W} \in \mathcal{D}$, let W_l denote its l-th word. The character-level representation (extracted by the CNN) and word-level embedding (*e.g.,* pretrained embedding) for W_l are concatenated as its final representation, $\boldsymbol{x}_l \in \mathbb{R}^D$, where D represents the dimension of $\boldsymbol{x}_l$. Note that hand-crafted features can be easily integrated into this architecture. However, we do not use any hand-crafted features in this study.

After the input representation layer, the input sequence can be represented as $\mathrm{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_L)$. We use a BiGRU to encode the sequence context. Specifically, GRU activations at time step l are computed as follows:

$$\boldsymbol{z}_l = \sigma(\boldsymbol{U}_z \boldsymbol{x}_l + \boldsymbol{R}_z \boldsymbol{h}_{l-1} + \boldsymbol{b}_z) \tag{4}$$

$$\boldsymbol{r}_l = \sigma(\boldsymbol{U}_r \boldsymbol{x}_l + \boldsymbol{R}_r \boldsymbol{h}_{l-1} + \boldsymbol{b}_r) \tag{5}$$

$$\boldsymbol{n}_l = \tanh(\boldsymbol{U}_h \boldsymbol{x}_l + \boldsymbol{R}_h(\boldsymbol{r}_l \odot \boldsymbol{h}_{l-1}) + \boldsymbol{b}_h) \tag{6}$$

$$\boldsymbol{h}_l = \boldsymbol{z}_l \odot \boldsymbol{h}_{l-1} + (1 - \boldsymbol{z}_l) \odot \boldsymbol{n}_l \tag{7}$$

where $\sigma(\cdot)$ is the sigmoid function, $\tanh(\cdot)$ is the hyperbolic tangent function, $\odot$ is an element-wise multiplication, $\boldsymbol{z}_l$ is the update gate vector, $\boldsymbol{r}_l$ is the reset gate vector, $\boldsymbol{n}_l$ is the new gate vector, and $\boldsymbol{h}_l$ is the hidden state at time step l. $\boldsymbol{U}$, $\boldsymbol{R}$, $\boldsymbol{b}$ are encoder parameters that need to be learned. Each hidden state of the BiGRU is formalized as: $\boldsymbol{h}_l = \overrightarrow{\boldsymbol{h}}_l \oplus \overleftarrow{\boldsymbol{h}}_l$, where $\oplus$ indicates a concatenation operation, and $\overrightarrow{\boldsymbol{h}}_l$ and $\overleftarrow{\boldsymbol{h}}_l$ are the hidden states of the forward (left-to-right) and backward (right-to-left) GRUs, respectively. Assuming the size of the GRU layer is H, the encoder yields hidden states in $\boldsymbol{h} \in \mathbb{R}^{L \times 2H}$.

For the tag decoder, we use 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:

$$p(\boldsymbol{y}|\boldsymbol{h}; \boldsymbol{W}, \boldsymbol{b}) = \frac{\prod_{i=1}^{L} \psi_i(y_{i-1}, y_i, \boldsymbol{h})}{\sum_{\boldsymbol{y}' \in \mathcal{Y}(\boldsymbol{h})} \prod_{i=1}^{L} \psi_i(y'_{i-1}, y'_i, \boldsymbol{h})} \tag{8}$$

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.

4.2.3 *Adversarial Training Strategy.*

Recall that $\boldsymbol{h} \in \mathbb{R}^{2H}$ is the hidden state of the last step in the context encoder. We apply a Multi-Layer Perceptron (MLP) as a domain discriminator to predict domain labels y_d:

$$\boldsymbol{\omega} = \mathrm{softmax}(\tanh(\boldsymbol{h} \cdot P + p)) \tag{9}$$

$$\boldsymbol{c} = \boldsymbol{h}\boldsymbol{\omega} \tag{10}$$

$$p(y_d|\boldsymbol{c}) = \mathrm{MLP}(\boldsymbol{c}) \tag{11}$$

The tag prediction loss and the domain discriminator prediction loss are calculated over the meta-training samples in task $\mathcal{T}_i$ from $\mathcal{D}_{tr}$. These two losses can be written as

$$\mathcal{L}_{\mathcal{T}_i}^{tr}(\theta, \phi) = \sum_{\mathcal{T}_i} -\log p(\boldsymbol{y}|\boldsymbol{h}; \theta, \phi) \tag{12}$$

$$\mathcal{L}_{\mathcal{T}_i}^{dis}(\theta, \gamma) = \sum_{\mathcal{T}_i} -\log p(y_d|\boldsymbol{c}; \theta, \gamma) \tag{13}$$

where θ are the learnable parameters of the sequence encoder, ϕ are the parameters of the tag decoder, and γ are the parameters of the discriminator. At learning time, in order to encourage domain-invariant features, we seek the parameters θ that *maximize* the loss of the domain discriminator (by making the two feature distributions as indistinguishable as possible), while simultaneously seeking the parameters θ and γ that *minimize* the loss of the domain discriminator. In addition, we seek the parameters ϕ that *minimize* the loss of the tag decoder. Thus, the optimization problem involves a minimization with respect to some parameters and a maximization with respect to others. Based on this idea, we define the adversarial objective as:

$$\mathcal{L}^{adv}_{\mathcal{T}_i}(\theta,\phi,\gamma) = \mathcal{L}^{tr}_{\mathcal{T}_i}(\theta,\phi) - \lambda\mathcal{L}^{dis}_{\mathcal{T}_i}(\theta,\gamma) \tag{14}$$

The parameter λ controls the trade-off between the two objectives. Then, we deliver a saddle point of $\mathcal{L}^{adv}_{\mathcal{T}_i}(\theta,\phi,\gamma)$ as

$$(\hat{\theta},\hat{\phi}) = \arg\min_{\theta,\phi}\mathcal{L}^{adv}_{\mathcal{T}_i}(\theta,\phi,\hat{\gamma}) \tag{15}$$

$$\hat{\gamma} = \arg\max_{\gamma}\mathcal{L}^{adv}_{\mathcal{T}_i}(\hat{\theta},\hat{\phi},\gamma) \tag{16}$$

Following [11], we add a special gradient reversal layer below the shared layer to address the minimax optimization problem.

4.2.4 Meta-Learning Strategy. The meta-learning strategy consists of two core phase: a meta-training phase and a meta-validation phase, as shown in Figure 2.

Meta-Training (Inner Loop). In the meta-training phase, our approach tries to learn adaptation parameters from the meta-training domains $\mathcal{D}_{tr}$, resulting in a temporary model. The parameters of the temporary model are adapted by gradient descent in a similar manner to the feature-critic networks [31]:

$$\theta_i^{(old)} = \theta_{i-1} - \alpha\nabla_{\theta_{i-1}}\mathcal{L}^{tr}_{\mathcal{T}_i}(\theta_{i-1},\phi_{i-1}) \tag{17}$$

$$\theta_i^{(new)} = \theta_i^{(old)} - \alpha\nabla_{\theta_{i-1}}\lambda\mathcal{L}^{dis}_{\mathcal{T}_i}(\theta_{i-1},\gamma_{i-1}) \tag{18}$$

where i is the adaptation step in the inner loop, and α is the learning rate of the inner optimization. At each adaptation step, the gradients are calculated with respect to the parameters from the previous step (*i.e.,* $\nabla_{\theta_{j-1}}$). Note that $\mathcal{L}^{dis}_{\mathcal{T}_i}$ is already operated with a gradient reversal layer. The base model parameters $\theta_0,\phi_0,\varphi_0$ should not be changed in the inner loop (*i.e.,* when updating the temporary model).

Meta-Validation (Outer Loop). After meta-training, MetaNER has already learned a temporary model $(\theta_i^{(old)},\theta_i^{(new)},\phi_0,\gamma_0)$ in the meta-training domains $\mathcal{D}_{tr}$. The meta-validation phase tries to minimize the distribution divergence between the source domains $\mathcal{D}_{tr}$ and simulated target domains $\mathcal{D}_{val}$ using the learned temporary model. It mimics the process of the temporary model being adapted to unseen domains. More specifically, the outer meta-validation loss is computed on the task $\mathcal{T}_j$ from the meta-validation domains $\mathcal{D}_{val}$ by

$$\mathcal{L}^{val}_{\mathcal{T}_j}(\theta_i^{(old)},\theta_i^{(new)},\phi_0,\gamma_0) = \mathcal{L}^{tr}_{\mathcal{T}_j}(\theta_i^{(old)},\phi_0) + \mathcal{L}^{dis}_{\mathcal{T}_j}(\theta_i^{(new)},\gamma_0) \tag{19}$$

Algorithm 1: Training MetaNER

```
Input: $\mathcal{D} = \{\mathcal{D}_1, ..., \mathcal{D}_N\}$, and $\alpha$, $\beta$
Output: Model $\Phi$
Initialize $\theta$, $\phi$, $\gamma$;
while not converge do
    Randomly split $\mathcal{D} = \mathcal{D}_{tr} \cup \mathcal{D}_{val}$ and $\mathcal{D}_{tr} \cap \mathcal{D}_{val} = \emptyset$;
    Let $\Phi = \{\theta_0, \phi_0, \gamma_0\}$
    for j in meta batch do                                  // Outer loop
        Sample a task $\mathcal{T}_j$ fom $\mathcal{D}_{val}$;
        Meta-training:
        for i in adaptation steps do                        // Inner loop
            Sample a task $\mathcal{T}_i$ from $\mathcal{D}_{tr}$;
            Compute meta-training loss $\mathcal{L}^{tr}_{\mathcal{T}_i}$ ;
            Compute domain loss $\mathcal{L}^{dis}_{\mathcal{T}_i}$;
            Compute adapted parameters with gradient descent
             for $\theta_{i-1}$;                            // $\mathcal{T}_i$, $\nabla_{\theta_{i-1}}$
            $\theta_i^{(old)} = \theta_{i-1} - \alpha\nabla_{\theta_{i-1}}\mathcal{L}^{tr}_{\mathcal{T}_i}(\theta_{i-1}, \phi_{i-1})$;
            $\theta_i^{(new)} = \theta_i^{(old)} - \alpha\nabla_{\theta_{i-1}}\lambda\mathcal{L}^{dis}_{\mathcal{T}_i}(\theta_{i-1}, \phi_{i-1})$;
        Meta-validation:
        Compute meta-validation loss on $\mathcal{T}_i$:
        $\mathcal{L}^{val}_{\mathcal{T}_j}(\theta_i^{(old)}, \theta_i^{(new)}, \phi_0, \gamma_0)$;
    Meta-optimization:
    Perform gradient step w.r.t $\Phi$:
    $\theta_0 \leftarrow \theta_0 - \beta\nabla_{\theta_0}\sum_{\mathcal{T}_i}\mathcal{L}^{adv}_{\mathcal{T}_i}$ ;        // $\mathcal{T}_i$, $\nabla_{\theta_0}$
    $\phi_0 \leftarrow \phi_0 - \beta\nabla_{\phi_0}\sum_{\mathcal{T}_j}\mathcal{L}^{val}_{\mathcal{T}_j}$ ;        // $\mathcal{T}_j$, $\nabla_{\phi_0}$
    $\gamma_0 \leftarrow \gamma_0 - \beta\nabla_{\gamma_0}\sum_{\mathcal{T}_j}\mathcal{L}^{val}_{\mathcal{T}_j}$ ;        // $\mathcal{T}_j$, $\nabla_{\gamma_0}$
```

Equation (19) can make the value range and gradient more stable [31]. The base model is updated by gradient descent:

$$\theta_0 \leftarrow \theta_0 - \beta\nabla_{\theta_0}\sum_{\mathcal{T}_i}(\mathcal{L}^{tr}_{\mathcal{T}_i}(\theta,\phi) - \lambda\mathcal{L}^{dis}_{\mathcal{T}_i}(\theta,\gamma)) \tag{20}$$

$$\phi_0 \leftarrow \phi_0 - \beta\nabla_{\phi_0}\sum_{\mathcal{T}_j}\mathcal{L}^{val}_{\mathcal{T}_j} \tag{21}$$

$$\gamma_0 \leftarrow \gamma_0 - \beta\nabla_{\gamma_0}\sum_{\mathcal{T}_j}\mathcal{L}^{val}_{\mathcal{T}_j} \tag{22}$$

where β is the meta-learning rate. Note that Equations (21) and (22) are computed by differentiating the loss $\mathcal{L}^{val}_{\mathcal{T}_i}$ with respect to the parameters ϕ_0,φ_0. Unlike the common gradient, the update mechanism of Equations (21) and (22) involves a gradient (w.r.t. the parameters of the base model) through a gradient (w.r.t. the parameters of the temporary model). This process requires second order optimization partial derivatives.

4.2.5 Algorithm Flow. The pseudocode for training MetaNER is given in Algorithm 1. At each iteration, we randomly split $\mathcal{D}$ into $\mathcal{D}_{tr}$ and $\mathcal{D}_{val}$ for the inner loop and outer loop, respectively. In the inner loop, MetaNER takes a gradient step to get new adaptation parameters, and obtains the new meta-validation loss. In the outer loop, MetaNER uses the validation on $\mathcal{D}_{val}$ to differentiate through the inner loop and update the parameters of the base model: $\theta_0,\phi_0,\varphi_0$. The pseudocode for adapting MetaNER is given in Algorithm 2.

Algorithm 2: Adapting MetaNER

```
Input: Training set $S_{tr}$ and test set $S_{te}$ of an unseen domain $\mathcal{D}_{new}$
Output: Performance on $S_{te}$
/* Fine-tuning the sequence labeling model */
1 Initialize $\theta$ from Algorithm 1;
2 Instantiate a new tag decoder $\phi$;
3 while available training data do
4     Sample a task $\mathcal{T}_{tr}$ from $S_{tr}$;
5     Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_{tr}} \mathcal{L}^{tr}_{\mathcal{T}_{tr}}$
6     Update $\phi \leftarrow \phi - \beta \nabla_\phi \sum_{\mathcal{T}_{tr}} \mathcal{L}^{tr}_{\mathcal{T}_{tr}}$
7     return Optimal $\theta^*$ and $\phi^*$
/* Final test on $S_{te}$ */
8 $Precison, Recall, F1 = f_{\mathcal{T}_{te}}(\theta^*, \phi^*)$
```

5 EXPERIMENTS

In this section, we first detail the experimental setups. Then, we present our experimental results on both homogeneous domain adaptation and heterogeneous domain adaption.

5.1 Experimental Setups

5.1.1 Baseline Methods. We evaluate MetaNER against the following competitors:

- **In-Domain** - This is trained on the training set of a target domain, and tested on the test set of that target domain using the CNN-BiGRU-CRF model. Therefore, its performance can be regarded as the upper bound of transferring tasks without using any additional resources.
- **D-Shift** - This is trained on the combination of training sets from all source domains. Then, the evaluation is conducted on the test set of a target domain using the direct domain shift strategy.
- **AGG** - This simply aggregates the training sets across all source and target domains. Note that no domain adaptation technique is used during training and testing.
- **FineTuning** - This is first trained on the training sets of the source domains, and then retrained on the training set of a target domain [25].
- **MultiTask** - This models different domains as different tasks, which are jointly trained on the training sets of the source and target domains in a multi-task learning manner [62].
- **DANN** - This is an unsupervised domain adaptation approach with adversarial training [11]. This model is further fine-tuned using the training set of the target domain.
- **WPZ** - This is a few-shot learning model that regards the sequence labeling problem as classification of each single token [10]. It is first trained on source domains and then retrained on target domains for token-level classification.
- MetaNER-**Zero** - This is trained on source domains the Algorithm 1 proposed in Section 4.2.5, without fine-tuning.

5.1.2 Implementation Details. For all neural network models, 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, 5]. The total number of CNN filters is 100. The bidirectional GRU has a depth of 1 and hidden size of 128. The inner learning rate α is 0.0001 and meta-learning rate β is 0.001. We use a dropout of 0.5 after the convolutional or recurrent layers and a fixed L2 regularization of 10^{-6}. The decay rate is 0.09 and the gradient clip is 5.0. Our proposed MetaNER is implemented with the PyTorch framework and evaluated on NVIDIA Tesla V100 GPUs. Note that MetaNER requires second order optimization partial derivatives. Unfortunately, the double backward for `_cudnn_rnn_backward` has not been implemented in PyTorch so far. Thus, we use the first order derivatives in meta-learning.

Domains	#Types	#Sentences			#Mentions
		Train	Dev	Test	
BC	7	2381	298	298	7291
BN	7	3427	428	429	8606
CTS	7	2731	342	342	8047
NW	7	1858	232	233	7969
UN	7	1790	224	224	5177
WL	7	1730	216	217	5141

Table 1: Statistics of the ACE2005 dataset used in homogeneous domain adaptation for NER.

5.2 Homogeneous Domain Adaptation

5.2.1 Datasets and Setups. In this experiment, 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). The six domains have homogeneous entity types: Person, Organization, Location, Geo-Political Entity, Facility, Vehicle and Weapon. ACE2005 is annotated with nested named entities. For example, the sentence "*Orders went out today to deploy 17,000 U.S. Army soldiers in the Persian Gulf region*" is originally annotated as [17,000 U.S. Army soldiers]$_{PER}$, [U.S.]$_{GPE}$, [U.S. Army]$_{ORG}$, [the Persian Gulf region]$_{LOC}$, [Persian Gulf]$_{LOC}$. We only keep the innermost entities for nested entities. That is, this example sentence is preprocessed as [U.S.]$_{GPE}$ and [Persian Gulf]$_{LOC}$ in our experiments. Table 1 reports the statistics of the six domains of ACE2005.

Following the setting in [31, 52], we adopt the *leave-one-out* evaluation protocol by picking one domain to hold out as the target domain $\mathcal{D}_{new}$ for the final evaluation. For each iteration in the training phase, four source domains are randomly chosen as the meta-training domains $\mathcal{D}_{tr}$, and the remaining domain as the meta-validation domain $\mathcal{D}_{val}$. We measure the performance of all models based on the popular and widely adopted standard metric used in NER: micro-F1 score.

5.2.2 Experimental Results. Table 2 reports the results of different methods under the leave-one-out settings. We make the following observations:

First, MetaNER outperforms all baseline methods in terms of F1 scores. More specifically, our model outperforms In-Domain,

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

Target Domains	In-Domain	D-Shift	AGG	FineTuning	MultiTask	DANN	WPZ	MetaNER-Zero	MetaNER
BC	72.13	64.43	73.22	73.94	74.65	<u>74.92</u>	73.83	67.78	**76.04***
BN	66.75	62.78	67.11	68.13	<u>69.02</u>	68.35	67.61	64.26	**71.42***
CTS	81.56	68.43	82.15	82.97	83.04	<u>83.22</u>	82.67	70.79	**84.12***
NW	74.19	66.93	75.28	76.82	<u>77.21</u>	75.96	76.44	69.58	**79.57***
UN	65.17	64.18	68.20	69.73	<u>72.73</u>	71.34	70.04	65.13	**74.08***
WL	65.60	61.32	67.21	68.56	<u>70.32</u>	69.41	68.27	63.59	**72.04***
Average	70.90	64.67	72.19	73.35	<u>74.49</u>	73.86	73.14	66.85	**76.21***
Improvement	↑7.49%	↑17.84%	↑5.57%	↑3.90%	↑2.31%	↑3.18%	↑4.20%	↑14.00%	-

Table 2: Performance (F1 Scores, %) of homogeneous domain adaptation for NER. The best performance is in boldface and the second best is underlined. Significant improvements over the baselines are marked with * (p-value < 0.05).

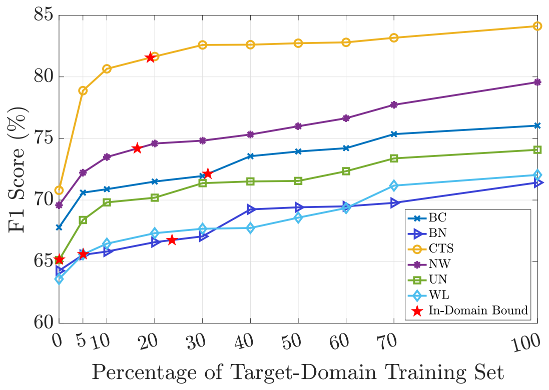


Figure 4: F1 score (MetaNER) w.r.t. the percentage of the target domain training set for different target domains. MetaNER **surpasses the In-Domain performance using only 16.17% of training data of target domains, on average.**

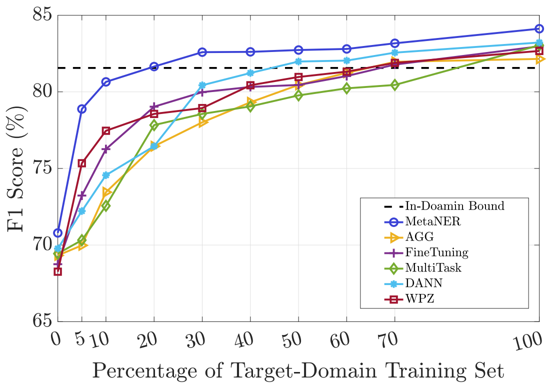


Figure 5: F1 score w.r.t. the percentage of the target domain training set for different methods on the CTS target domain.

D-Shift, AGG, FineTuning, MultiTask, DANN, WPZ and MetaNER-Zero by relative F1 improvements of 7.49%, 17.84%, 5.57%, 3.90%, 2.31%, 3.18%, 4.20% and 14.00%, respectively. We attribute this to the fact that MetaNER explicitly simulates domain shift during training via meta-learning, which is helpful for adapting to a novel target domain.

Second, D-Shift and MetaNER-Zero both do not use the target domain data. Both methods suffer from performance degradation when adapting to specific domains. However, the zero-shot version of our approach (MetaNER-Zero) still outperforms the direct domain shift method (D-Shift).

Third, AGG consistently outperforms In-Domain (which is the upper bound performance using in-domain training sets) on all target domains. This is because the six domains have homogeneous entity labels from the same dataset (ACE2005). Therefore, the differences among these six domains in terms of statistical distributions are relatively small. As such, aggregating the training sets across all source and target domains can slightly enhance the performance of the In-Domain method. Notably, MetaNER significantly outperforms the In-Domain method by an average improvement of 7.49%.

Although the above experiments gain significant improvements when the full training sets of target domains are available, we are more interested in the low-resource scenarios. We employ the same setup as previously and vary the data ratio of the target training set as 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, and 1. Figure 4 illustrates the F1 scores of MetaNER with respect to the data ratios of the target domain training data across different target domains. Impressively, MetaNER surpasses the performance of In-Domain methods using only 31%, 23%, 19%, 16%, 2% and 6% of the target domain training sets for BC, BN, CTS, NW, UN, and WL, respectively.

Figure 5 shows the F1 scores with respect to the data ratios of the target domain training sets for different methods on the CTS target domain. On average, the baseline methods perform on par with the In-Domain model (In-Domain bound) using 67% of training data on CTS. Compared with these baseline methods, MetaNER surpasses the In-Domain bound using only 19% of the training data. The same observations hold for the BC, BN, NW, UN and WL domains.

5.3 Heterogeneous Domain Adaptation

5.3.1 Datasets and Setups. For the heterogeneous domain adaptation, we use four datasets as source domains and five datasets as target domains. Table 4 summarizes the statistics of these nine datasets. CoNLL03, OntoNotes5.0, WikiGold and WNUT17 are from the domains of newswires, various, social media and Wikipedia, respectively. BioNLP13PC, MIT Movie, MIT Restaurant, Re3d and SEC are from the domains of medical, movie, restaurant, defense and finance, respectively.

For each iteration in the training phase, three source domains are randomly chosen as the meta-training domains, and the remaining

Target Domains	In-Domain	D-Shift	AGG	FineTuning	MultiTask	DANN	WPZ	MetaNER-Zero	MetaNER
BioNLP13PC	82.11	-	79.03	82.05	83.23	<u>83.75</u>	81.54	-	**85.11***
MIT Movie	82.48	-	83.18	84.27	<u>85.03</u>	83.95	82.79	-	**86.41***
MIT Restaurant	74.86	-	73.12	75.39	<u>76.21</u>	75.97	74.84	-	**78.05***
Re3d	62.74	-	26.04	63.21	64.52	<u>65.78</u>	63.89	-	**68.52***
SEC	75.86	-	70.00	76.38	78.82	<u>81.32</u>	77.53	-	**83.44***
Average	75.61	-	66.27	76.26	77.56	<u>78.15</u>	76.11	-	**80.30***
Improvement	↑6.20%	-	↑21.17%	↑5.29%	↑3.53%	↑2.75%	↑5.51%	-	-

Table 3: Performance (F1 Scores, %) of the heterogeneous domain adaptation for NER. The best performance is in boldface and the second best is underlined. Significant improvements over the baselines are marked with * (p-value < 0.05).

Datasets	#Types	#Sentences			#Mentions
		Train	Dev	Test	
Source Domains					
CoNLL03	4	14041	3250	3453	34841
OntoNotes5.0	18	59917	1009	1287	104248
WikiGold	4	143342	1500	1696	300069
WNUT17	6	3394	1009	1287	3850
Target Domains					
BioNLP13PC	4	2498	856	1694	15885
MIT Movie	12	8797	978	2443	26634
MIT Restaurant	8	6894	766	1521	18514
Re3d	10	687	77	199	3388
SEC	4	1047	117	303	1479

Table 4: Statistics of the datasets used in heterogeneous domain adaptation for NER.

one as the meta-validation domain. In the final evaluation phase, we fine-tune the sequence encoder learned from source domains and instantiate a new tag decoder for each target domain. Note that the methods of D-Shift and MetaNER-Zero cannot be used in heterogeneous settings because source and target domains have different label spaces.

5.3.2 *Experimental Results.* Table 3 reports the results of different methods under heterogeneous settings. We make the following observations:

First, MetaNER outperforms all competitors in terms of F1 scores. More specifically, our model outperforms In-Domain, AGG, FineTuning, MultiTask, DANN and WPZ by relative F1 improvements of 6.20%, 21.17%, 5.29%, 3.53%, 2.75% and 5.51%, respectively.

Second, on average across all target domains, the performance of In-Domain is better than AGG, and all other baselines (*i.e.*, FineTuning, MultiTask, DANN and WPZ) are better than In-Domain.

Third, different from homogeneous adaptation, under heterogeneous settings, simply aggregating source and target domains can both boost and worsen the performance of the In-Domain method, which is also observed in [31]. For example, the performance of AGG is much worse than the In-Domain method on the Re3d domain, while sightly better on the MIT Movie domain. Overall, the performance of AGG is worse than In-Domain in most cases. This is because the larger difference between source and target domains makes heterogeneous adaptation more challenging.

We also investigate the performances in low-resource scenarios. We employ the same setup as previously and vary the data ratio of the target training sets as 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, and 1. Figure 6 illustrates the F1 scores of MetaNER with respect to the data ratios of the target domain training sets across different target domains. Different from the previous homogeneous adaptation cases, MetaNER needs more training data from target domains to outperform the In-Domain method. More specifically, MetaNER surpasses the performance of In-Domain methods using only 52%,

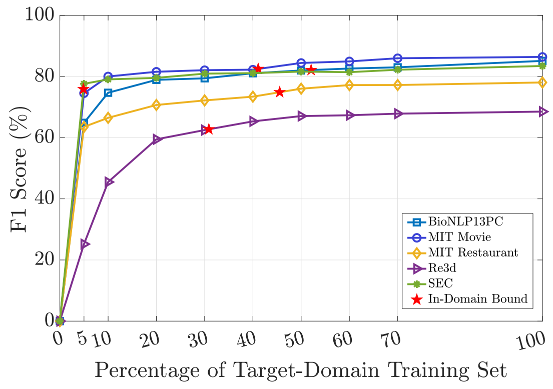


Figure 6: F1 score (MetaNER) w.r.t. the percentage of the target domain training set for different target domains. MetaNER surpasses the In-Domain performance using only 34.76% of training data of target domains, on average.

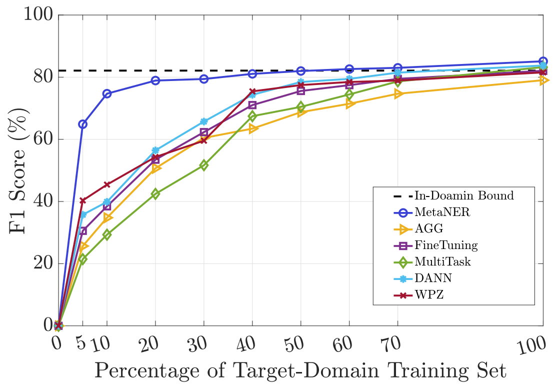
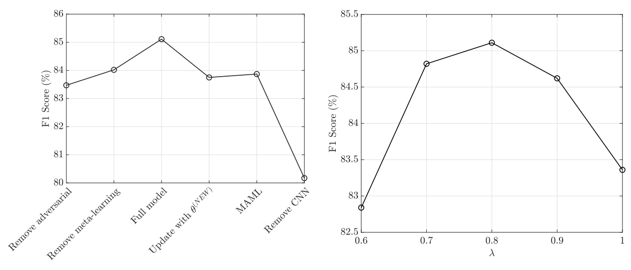


Figure 7: F1 score w.r.t. the percentage of the target domain training set for different methods on the BioNLP13PC target domain.

	Targets	NER Results by MetaNER	Correct
Homogeneous	BC	Orders went out today to deploy 17,000 ⟦U.S. Army⟧$_{\text{Org}}$ soldiers in the ⟦Persian Gulf⟧$_{\text{Loc}}$.	✗
	BN	a ⟦major league baseball official⟧$_{\text{Org}}$ official is skilledded to conduct interviews in ⟦chicago⟧$_{\text{GPE}}$.	✗
	CTS	⟦Diane⟧$_{\text{Per}}$ Do y'all do cross country moves , or just local?	✗
	NW	⟦Protesters⟧$_{\text{Per}}$ also gathered in their thousands in ⟦Halifax⟧$_{\text{Gpe}}$, ⟦Calgary⟧$_{\text{Gpe}}$, ⟦Edmonton⟧$_{\text{Gpe}}$ and ⟦Vancouver⟧$_{\text{Gpe}}$.	✓
	UN	⟦Analysts⟧$_{\text{Per}}$ say both ⟦India⟧$_{\text{Gpe}}$ and ⟦Enron⟧$_{\text{Gpe}}$ have little to gain from a protracted legal battle.	✓
	WL	⟦Disgruntled soldiers⟧$_{\text{Per}}$ complained to ⟦Rumsfeld⟧$_{\text{Per}}$ about long deployments and a lack of ⟦armored vehicles⟧$_{\text{Veh}}$.	✓
Heterogeneous	BioNLP13PC	⟦TAK1⟧$_{\text{Ggr}}$ activated ⟦IKKalpha⟧$_{\text{Ggr}}$ and ⟦IKKbeta⟧$_{\text{Ggr}}$ in the presence of ⟦TAB1⟧$_{\text{Ggr}}$.	✓
	MIT Movie	list the ⟦five star⟧$_{\text{Ratings}}$ rated movies starring ⟦mel gibson⟧$_{\text{Actor}}$	✓
	MIT Restaurant	are there any places around here that has ⟦tomato sauce⟧$_{\text{Dish}}$ based dishes	✗
	Re3d	Responding to the report ⟦Gareth Bayley⟧$_{\text{Per}}$, the ⟦UK⟧$_{\text{Nat}}$ Special Representative for ⟦Syria⟧$_{\text{Nat}}$ said:	✗
	SEC	Attention: ⟦Frank Wouters⟧$_{\text{Per}}$, ⟦CEO Lender 138 Bartlett Street Marlboro⟧$_{\text{Per}}$, ⟦Massachusetts⟧$_{\text{Loc}}$.	✗

Table 5: Positive and negative examples for MetaNER **with 10% of the training data from target domains. The results produced by** MetaNER **are marked with ⟦ ⟧. The green and red highlights indicate a correct and incorrect (missed) result, respectively.**



(a) Study for architectural choices (b) Parameter study for λ

Figure 8: Impact of architectural choices and parameter λ.

41%, 45%, 31% and 5% of training data for BioNLP13PC, MIT Movie, MIT Restaurant, Re3d, and SEC, respectively. Figure 7 shows the F1 scores with respect to the data ratios of the target domain training sets for different methods on the BioNLP13PC target domain. Compared with the baseline methods, MetaNER quickly achieves the same performance as In-Domain (In-Domain bound) using less training data. The same observations hold for the MIT Movie, MIT Restaurant, Re3d, and SEC domains.

5.4 Further Analysis

In this section, we first study the impact of key hyperparameter settings and various architectural choices on model performance. Then we present a qualitative analysis.

5.4.1 Ablation Study. Table 8(a) reports an ablation analysis on the test set of BioNLP13PC. The full model is our proposed MetaNER. There are five variations: we remove the adversarial strategy, remove the meta-learning strategy, update the model using $\theta^{(new)}$ only, update the model using MAML [8], and remove CNNs. We observe that our update mechanism outperforms the MAML method. Meanwhile, the character-level representations play an important role in domain adaptation for NER. This ablation study clearly showcases the importance of each component of MetaNER.

Table 8(b) reports a study on parameter sensitivity for λ. Parameter λ is the trade-off between the tag decoder loss and domain discriminator loss during adversarial training. We observe that $\lambda = 0.8$ yields the best empirical performance. This empirical result demonstrates that we need to balance the two learning objectives for better transferability.

5.4.2 Qualitative Analysis. Because we are more interested in low-resource scenarios, we train MetaNER using only 10% of the training data for all target domains. Table 5 shows some positive and negative examples for homogeneous and heterogeneous domain adaptations. For the homogeneous domain adaptation, we only keep the innermost entities for all nested entities when preprocessing the ACE2005 dataset. This may lead to many short entities being present in the ground truth. For the negative example in BC, the ground truth entity is ⟦U.S. ⟧$_{\text{Gpe}}$, while our result is ⟦U.S. Army⟧$_{\text{Org}}$. For the negative example in BN, the ground truth entity is ⟦major league baseball⟧$_{\text{Org}}$, while MetaNER fails to detect the correct boundaries of this entity. For the example in CTS, MetaNER misses an entity ⟦y'all⟧$_{\text{Org}}$. From the negative examples in the heterogeneous domain adaptation, we also observe that MetaNER misses some entities and wrongly detects the boundaries of others. In summary, the different annotation criteria in different domains are the key factors affecting transferability. Although MetaNER is designed for domain adaptation, we do not claim that it can handle all cases in the real world where the natural language is complicated and noisy.

6 CONCLUSION

In this paper, we are the first to investigate meta-learning for sequence labeling. We proposed MetaNER, a novel meta-learning approach for both homogeneous and heterogeneous domain adaptations in NER. In particular, MetaNER can effectively learn a robust and general sequence encoder from multiple source domains. The key advantage of MetaNER is that it can accurately adapt to unseen domains with a small amount of data. We conducted extensive experiments on homogeneous and heterogeneous domain adaptations in NER. The experimental results demonstrate the effectiveness of our proposed approach. We also conducted experiments to analyze the parameter settings and architectural choices. Finally, a case study was presented for qualitative analysis.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No. 61932004). We also thank anonymous reviewers for their valuable suggestions.

REFERENCES

[1] Bogdan Babych and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the 7th International EAMT workshop on MT and other Language Technology Tools, Improving MT through other Language Technology Tools: Resources and Tools for Building MT*. 1–8.

[2] Genady Beryozkin, Yoel Drori, Oren Gilon, Tzvika Hartman, and Idan Szpektor. 2019. A Joint Named-Entity Recognizer for Heterogeneous Tag-sets Using a Tag Hierarchy. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*. 140–150.

[3] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* 12 (2011), 2493–2537.

[4] Wanyun Cui, Guangyu Zheng, Zhiqiang Shen, Sihang Jiang, and Wei Wang. 2019. Transfer Learning for Sequences via Learning to Collocate. In *Proceedings of the 7th International Conference on Learning Representations*.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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*. 4171–4186.

[6] Oren Etzioni, Michael J. Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the Web: An experimental study. *Artif. Intell.* 165, 1 (2005), 91–134.

[7] Xiaocheng Feng, Xiachong Feng, Bing Qin, Zhangyin Feng, and Ting Liu. 2018. Improving Low Resource Named Entity Recognition using Cross-lingual Knowledge Transfer. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. 4071–4077.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning*. 1126–1135.

[9] Chelsea Finn, Aravind Rajeswaran, Sham M. Kakade, and Sergey Levine. 2019. Online Meta-Learning. In *Proceedings of the 36th International Conference on Machine Learning*. 1920–1930.

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

[11] Yaroslav Ganin and Victor S. Lempitsky. 2015. Unsupervised Domain Adaptation by Backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning*. 1180–1189.

[12] Abbas Ghaddar and Philippe Langlais. 2018. Robust Lexical Features for Improved Neural Network Named-Entity Recognition. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1896–1907.

[13] John M. Giorgi and Gary D. Bader. 2018. Transfer learning for biomedical named entity recognition with neural networks. *Bioinformatics* 34, 23 (2018), 4087–4094.

[14] Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-Learning for Low-Resource Neural Machine Translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 3622–3631.

[15] Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 267–274.

[16] Hangfeng He and Xu Sun. 2017. A Unified Model for Cross-Domain and Semi-Supervised Named Entity Recognition in Chinese Social Media. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. 3216–3222.

[17] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzkebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*. 2790–2799.

[18] Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. 2018. Natural Language to Structured Query Generation via Meta-Learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 732–738.

[19] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *CoRR* abs/1508.01991 (2015).

[20] Zongcheng Ji, Aixin Sun, Gao Cong, and Jialong Han. 2016. Joint Recognition and Linking of Fine-Grained Locations from Tweets. In *Proceedings of the 25th International Conference on World Wide Web*. 1271–1281.

[21] Chen Jia, Liang Xiao, and Yue Zhang. 2019. Cross-Domain NER using Cross-Domain Language Modeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*. 2464–2474.

[22] Young-Bum Kim, Karl Stratos, Ruhi Sarikaya, and Minwoo Jeong. 2015. New Transfer Learning Techniques for Disparate Label Sets. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*. 473–482.

[23] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 260–270.

[24] Changki Lee, Yi-Gyu Hwang, and Myung-Gil Jang. 2007. Fine-grained named entity recognition and relation extraction for question answering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 799–800.

[25] Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2018. Transfer Learning for Named-Entity Recognition with Neural Networks. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*.

[26] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2018. A Survey on Deep Learning for Named Entity Recognition. *CoRR* abs/1812.09449 (2018).

[27] Jing Li, Aixin Sun, and Shafiq R. Joty. 2018. SegBot: A Generic Neural Text Segmentation Model with Pointer Network. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. 4166–4172.

[28] Jing Li, Deheng Ye, and Shuo Shang. 2019. Adversarial Transfer for Named Entity Boundary Detection with Pointer Networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. 5053–5059.

[29] Peng-Hsuan Li, Ruo-Ping Dong, Yu-Siang Wang, Ju-Chieh Chou, and Wei-Yun Ma. 2017. Leveraging Linguistic Structures for Named Entity Recognition with Bidirectional Recursive Neural Networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2664–2669.

[30] Yaoyong Li, Kalina Bontcheva, and Hamish Cunningham. 2004. SVM Based Learning System for Information Extraction. In *First International Workshop on Deterministic and Statistical Methods in Machine Learning*. 319–339.

[31] Yiying Li, Yongxin Yang, Wei Zhou, and Timothy M. Hospedales. 2019. Feature-Critic Networks for Heterogeneous Domain Generalization. In *Proceedings of the 36th International Conference on Machine Learning*. 3915–3924.

[32] Bill Yuchen Lin and Wei Lu. 2018. Neural Adaptation Layers for Cross-domain Named Entity Recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels*. 2012–2022.

[33] Ying Lin, Shengqi Yang, Veselin Stoyanov, and Heng Ji. 2018. A Multi-lingual Multi-task Architecture for Low-resource Sequence Labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 799–809.

[34] Liyuan Liu, Jingbo Shang, Xiang Ren, Frank Fangzheng Xu, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower Sequence Labeling with Task-Aware Neural Language Model. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. 5253–5260.

[35] Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

[36] Robert Malouf. 2002. Markov Models for Language-independent Named Entity Recognition. In *Proceedings of the 6th Conference on Natural Language Learning*.

[37] David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes* 30, 1 (2007), 3–26.

[38] Chikashi Nobata, Satoshi Sekine, Hitoshi Isahara, and Ralph Grishman. 2002. Summarization System Integrated with Named Entity Tagging and IE pattern Discovery. In *Proceedings of the Third International Conference on Language Resources and Evaluation*.

[39] Abiola Obamuyide and Andreas Vlachos. 2019. Model-Agnostic Meta-Learning for Relation Classification with Limited Supervision. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*. 5873–5879.

[40] Sinno Jialin Pan, Zhiqiang Toh, and Jian Su. 2013. Transfer joint embedding for cross-domain named entity recognition. *ACM Trans. Inf. Syst.* 31, 2 (2013), 7.

[41] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* 22, 10 (2010), 1345–1359.

[42] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2227–2237.

[43] Kun Qian and Zhou Yu. 2019. Domain Adaptive Dialog Generation via Meta Learning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*. 2639–2649.

[44] Lizhen Qu, Gabriela Ferraro, Liyuan Zhou, Weiwei Hou, and Timothy Baldwin. 2016. Named Entity Recognition for Novel Types by Transfer Learning. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 899–905.

[45] Aravind Rajeswaran, Chelsea Finn, Sham M. Kakade, and Sergey Levine. 2019. Meta-Learning with Implicit Gradients. *CoRR* abs/1909.04630 (2019).

[46] Sachin Ravi and Hugo Larochelle. 2017. Optimization as a Model for Few-Shot Learning. In *Proceedings of the 5th International Conference on Learning Representations*.

[47] Juan Diego Rodríguez, Adam Caldwell, and Alexander Liu. 2018. Transfer Learning for Entity Recognition of Novel Classes. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1974–1985.

[48] Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. 2019. Meta-Learning with Latent Embedding Optimization. In *Proceedings of the 7th International Conference on Learning Representations.*

[49] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy P. Lillicrap. 2016. Meta-Learning with Memory-Augmented Neural Networks. In *Proceedings of the 33nd International Conference on Machine Learning*. 1842–1850.

[50] Jürgen Schmidhuber. 1995. On learning how to learn learning strategies. (1995).

[51] Burr Settles. 2004. Biomedical Named Entity Recognition using Conditional Random Fields and Rich Feature Sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications.*

[52] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. 2018. Generalizing Across Domains via Cross-Gradient Training. In *Proceedings of the 6th International Conference on Learning Representations.*

[53] Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep Active Learning for Named Entity Recognition. In *Proceedings of the 6th International Conference on Learning Representations.*

[54] Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical Networks for Few-shot Learning. In *Proceedings of the Advances in Neural Information Processing Systems*. 4077–4087.

[55] Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and Accurate Entity Recognition with Iterated Dilated Convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2670–2680.

[56] Sebastian Thrun and Lorien Y. Pratt (Eds.). 1998. *Learning to Learn.* Springer.

[57] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems*. 3630–3638.

[58] Pius von Däniken and Mark Cieliebak. 2017. Transfer Learning and Sentence Level Features for Named Entity Recognition on Tweets. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*. 166–171.

[59] Xuan Wang, Yu Zhang, Xiang Ren, Yuhao Zhang, Marinka Zitnik, Jingbo Shang, Curtis Langlotz, and Jiawei Han. 2019. Cross-type biomedical named entity recognition with deep multi-task learning. *Bioinformatics* 35, 10 (2019), 1745–1752.

[60] Yu-Xiong Wang, Ross B. Girshick, Martial Hebert, and Bharath Hariharan. 2018. Low-Shot Learning From Imaginary Data. In *2018 IEEE Conference on Computer Vision and Pattern Recognition*. 7278–7286.

[61] Zhenghui Wang, Yanru Qu, Liheng Chen, Jian Shen, Weinan Zhang, Shaodian Zhang, Yimei Gao, Gen Gu, Ken Chen, and Yong Yu. 2018. Label-Aware Double Transfer Learning for Cross-Specialty Medical Named Entity Recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1–15.

[62] Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2017. Transfer Learning for Sequence Tagging with Hierarchical Recurrent Networks. In *roceedings of the 5th International Conference on Learning Representations.*

[63] Huaxiu Yao, Ying Wei, Junzhou Huang, and Zhenhui Li. 2019. Hierarchically Structured Meta-learning. In *Proceedings of the 36th International Conference on Machine Learning*. 7045–7054.

[64] Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. 2018. Bayesian Model-Agnostic Meta-Learning. In *Proceedings of the Advances in Neural Information Processing Systems*. 7343–7353.

[65] Shaodian Zhang and Noemie Elhadad. 2013. Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts. *Journal of Biomedical Informatics* 46, 6 (2013), 1088–1098.

[66] Joey Tianyi Zhou, Hao Zhang, Di Jin, Hongyuan Zhu, Meng Fang, Rick Siow Mong Goh, and Kenneth Kwok. 2019. Dual Adversarial Neural Transfer for Low-Resource Named Entity Recognition. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*. 3461–3471.

[67] Jingwei Zhuo, Yong Cao, Jun Zhu, Bo Zhang, and Zaiqing Nie. 2016. Segment-Level Sequence Modeling using Gated Recursive Semi-Markov Conditional Random Fields. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics.*