Sequence Labeling with Meta-Learning

Jing Li, Peng Han, Xiangnan Ren, Jilin Hu, Lisi Chen and Shuo Shang

Abstract—Recent neural architectures in sequence labeling 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. To make a sequence labeling system more broadly useful, it is crucial to reduce its training data requirements and transfer knowledge to other domains. In this paper, we investigate the problem of domain adaptation for sequence labeling under homogeneous and heterogeneous settings. We propose METASEQ, a novel meta-learning approach for domain adaptation in sequence labeling. Specifically, METASEQ incorporates meta-learning and adversarial training strategies to encourage robust, general and transferable representations for sequence labeling. The key advantage of METASEQ is that it is capable of adapting to new unseen domains with a small amount of annotated data from those domains. We extensively evaluate METASEQ on named entity recognition, part-of-speech tagging and slot filling under homogeneous and heterogeneous settings. The experimental results show that METASEQ achieves state-of-the-art performance against eight baselines. Impressively, METASEQ surpasses the in-domain performance using only 16.17% and 7% of target domain data on average for homogeneous settings, and 34.76%, 24%, 22.5% of target domain data on average for heterogeneous settings.

Index Terms—Natural Language Processing, Sequence Labeling, Domain Adaptation, Meta-Learning

1 INTRODUCTION

SEQUENCE Labeling is a fundamental task in natural language processing (NLP), aiming at assigning a categorical class or label to each member of a sequence of observed values [1]–[3]. Some typical sequence labeling tasks include named entity recognition(NER) [2], part-of-speech (POS) tagging [4], slot filling [5], etc. Sequence labeling 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 [6], automatic text summarization [7], etc. Recently, a significant amount of work [8]–[14] has been devoted to developing end-to-end neural-based sequence labeling models. 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 sequence labeling, the source and target domains have the same label space. The model trained on source domains can be directly transferred to target domains [15]. For *heterogeneous DA* in sequence labeling, 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 sequence labeling by learning correlations between label sets [16], [17] and fine-tuning the source model with target domain data [18], [19]. However, most works often require a large amount of annotated target domain data to achieve accurate domain adaptation. To make an sequence labeling 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 in NLP because semantic tags (e.g., named entities in NER) often share lexical and context features (e.g., common vocabularies, similar word semantics and similar sentence syntaxes) [20]. 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 [21]. In short, we *learn how to learn* much faster and more efficiently across various tasks. Meta-learning [22], [23] 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 [24]–[26]. Inspired by the essence of meta-learning [27], [28], 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 our idea of meta-learning in sequence labeling.

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 designed to aggregate the meta-knowledge from multiple source domains so that it has maximal performance on a new domain with a small

This paper is an extended version of the conference paper [1].

- *J. Li, L. Chen and S. Shang are with University of Electronic Science and Technology of China, Chengdu 611731, China. E-mail: jingli.phd@hotmail.com, chenlisi.cs@gmail.com, jedi.shang@gmail.com. (Corresponding authors: Peng Han and Shuo Shang)*
- *P. Han is with King Abdullah University of Science and Technology, Saudi Arabia. E-mail: peng.han@kaust.edu.sa.*
- *X. Ren is with Inception Institute of Artificial Intelligence, Abu Dhabi 54115, UAE. E-mail: renxiangnan90@gmail.com.*
- *J. Hu is with Aalborg University. E-mail: hujilin@cs.aau.dk.*

Manuscript received xx, 2020; revised xx , 2020.

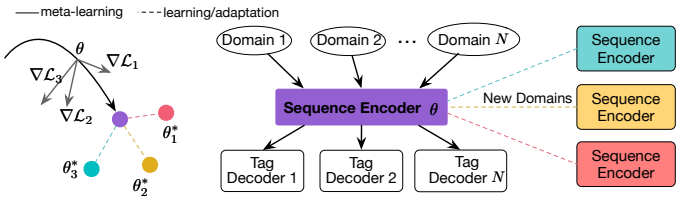


Fig. 1. Illustration of the idea of meta-learning in sequence labeling.

amount of data. Most existing meta-learning approaches are designed for classification under the few-shot setting (i.e., N-way K-shot [29]). 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 METASEQ, a novel meta-learning approach for sequence labeling. First, METASEQ explicitly simulates the training-to-testing domain shift by splitting source domains into meta-training and meta-validation sets. METASEQ is trained in two alternating phases. In the meta-training phase, METASEQ 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 METASEQ, a novel meta-learning approach for sequence labeling. METASEQ 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 METASEQ on NER, POS tasks under homogeneous domain adaptation settings. The results show that METASEQ achieves state-of-the-art performance against eight baselines. Impressively, METASEQ surpasses the in-domain performance using only 16.17% and 7% of target domain data on average, for NER and POS respectively.
- We extensively evaluate METASEQ on NER, POS and slot filling tasks under heterogeneous domain adaptation settings. The results show that METASEQ achieves state-of-the-art performance against baselines. METASEQ surpasses the in-domain performance using only 34.76%, 24%, and 22.5% of target domain data on average, for NER, POS and slot filling respectively.
- We conduct experiments to further analyze the parameter settings and architectural choices. We also present a study for qualitative analysis.

2 BACKGROUND AND RELATED WORK

2.1 Sequence Labeling

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. In the context of NER, 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].

There are three common paradigms for sequence labeling [2]: *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 *Feature-based supervised* systems cast NER as a multi-class classification or sequence labeling task. Recently, several neural architectures [8], [9], [11], [11], [31]–[36] 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. 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 [37]. Several studies [19], [38]–[51] have already contributed effort to leveraging deep transfer learning for NER. Most recent transfer approaches fall into the *parameter-sharing* method [42]. Commonly, different neural models [43]–[49] share certain parts of model parameters between the source domain and target domain. Yang et al. [50] proposed three different parameter-sharing models to investigate the transferability of different layers of 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. [18] 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 [19], [39].

In this paper, we extend our previous MetaNER [1] approach for generic sequence labeling. 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.

2.2 Meta-Learning

Meta-learning (a.k.a. learning to learn) [22], [52] 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 focus on few-shot learning and can be broadly categorized as metric-based methods [29], [53], memory-based methods [54], [55], and optimization-based methods [23], [25]. Some studies [24], [28], [56], [57] have applied meta-learning strategies for image classification in few-shot learning.

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

Here, we introduce an optimization-based method, Model-Agnostic Meta-Learning (MAML) [23], 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 training set and a validation set. In the N-way K-shot [29] 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:

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

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:

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

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:

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

where β is the learning rate of meta optimization.

There have been a few attempts devoted to the application of meta-learning in NLP over the last two years. Gu et al. [58] 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. [59] proposed a method for query generation based on MAML [23], by reducing a regular supervised learning problem to the few-shot meta-learning scenario. Qian and Zhou [27] 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. [60] 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.

Note that the objective of MAML is designed for few-shot classification under the problem setting of N-way K-shot. In this paper, we seek a more suitable meta-learning optimization objective for sequence labeling scenarios, which would be more suitable for encouraging robust representations, rather than for N-way K-shot few-shot learning. To the best of our knowledge, we are the first to attempt adopting meta-learning in sequence labeling.

3 METASEQ: SEQUENCE LABELING WITH META-LEARNING

3.1 Problem Statement

Let $\boldsymbol{\mathcal{D}_s} = \{\mathcal{D}_1, ..., \mathcal{D}_n, ..., \mathcal{D}_N\}$ 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 $\boldsymbol{\mathcal{D}_t} = \{\mathcal{D}_1, ..., \mathcal{D}_k, ..., \mathcal{D}_K\}$, which are unseen in $\boldsymbol{\mathcal{D}_s}$. Likewise, $\mathcal{D}_k$ is the k-th target domain containing annotated data $(\mathcal{X}_k, \mathcal{Y}_k)$. Taking NER as an example, 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* sequence labeling, all the source domains and the target domains share the same label space. For *heterogeneous* sequence labeling, the domains can have different, and even completely, disjoint label spaces.

To make domain adaptation possible, 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 $\boldsymbol{\mathcal{D}_s}$. Given a new unseen domain from $\boldsymbol{\mathcal{D}}_{new}$ (which can be either homogeneous or heterogeneous), the new learning task 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 $\boldsymbol{\mathcal{D}_s}$, resulting in more robust, general and transferable representations, which can be broadly adapted to achieve optimum performance in $\boldsymbol{\mathcal{D}}_{new}$ with as little as possible.

3.2 The METASEQ Approach

3.2.1 *Overview of* METASEQ

Figure 2 shows an overview of our proposed METASEQ, 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 $\boldsymbol{\mathcal{D}_s} = \mathcal{D}_{tr} \cup \mathcal{D}_{val}$ and $\mathcal{D}_{tr} \cap \mathcal{D}_{val} = \emptyset$. A meta-training task $\mathcal{T}_i$ is 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.

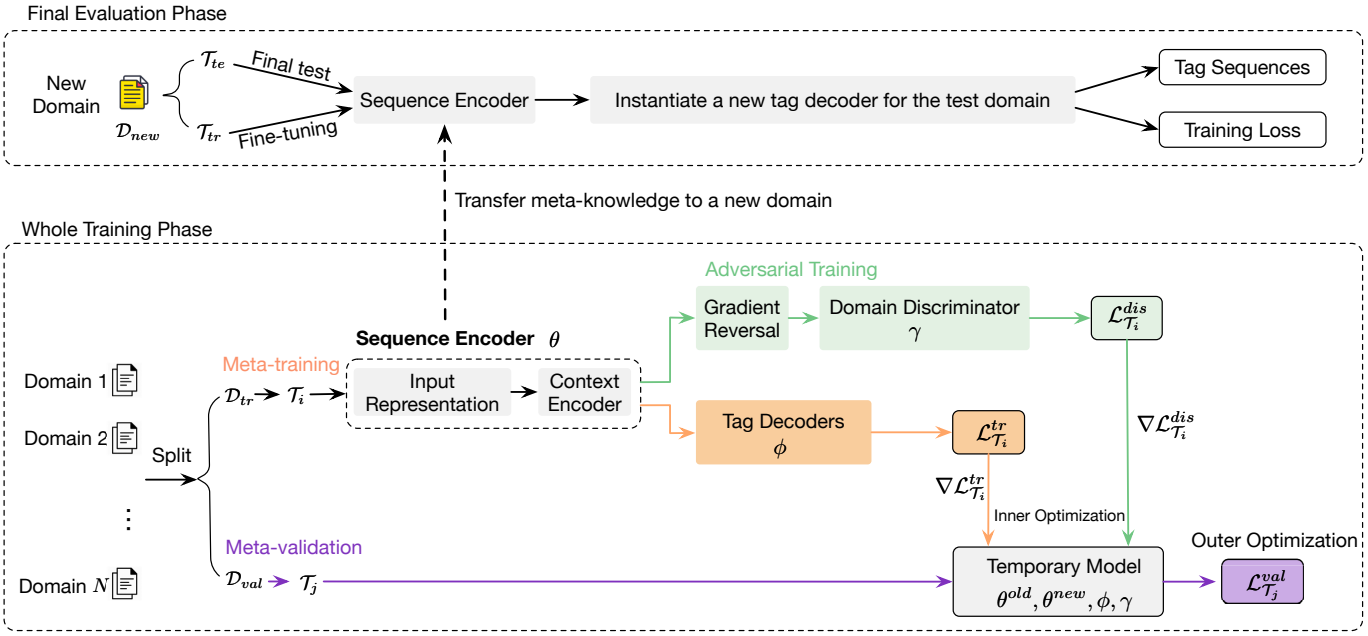


Fig. 2. An overview of our proposed METASEQ (best viewed in color). METASEQ explicitly simulates domain shift ($\mathcal{D}_{tr} \rightarrow \mathcal{D}_{val}$) during the training process. In the meta-training phase, a temporary model $(\theta^{old}, \theta^{new}, \phi, \gamma)$ is learned from $\mathcal{D}_{tr}$. 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 $\mathcal{D}_{new}$.

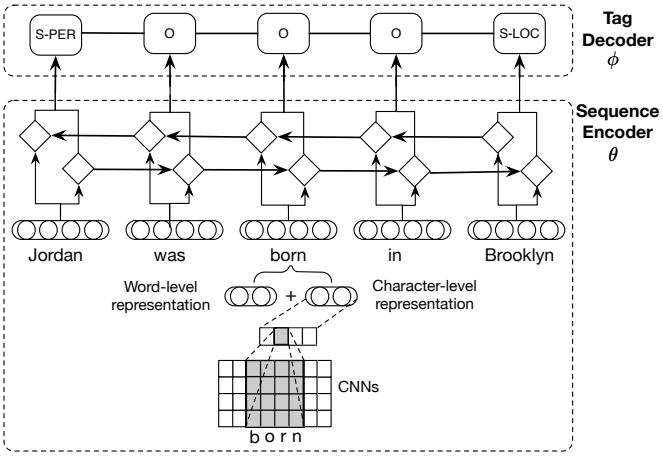


Fig. 3. CNN-BiGRU-CRF for sequence labeling. We decompose it into two modules: a sequence encoder θ and a tag decoder ϕ.

3.2.2 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 $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_L)$. We use a BiGRU to encode the sequence context. Specifically, GRU activations at time step l are computed as follows:

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

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

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

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

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.

3.2.3 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:

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

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

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

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:

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

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

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

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

Note that the $-\lambda$ factor in Equation (14) is very important because the stochastic gradient descent would directly minimize the domain prediction loss without such factor, resulting in discriminative features across domains only. Following [61], we add a special gradient reversal layer (GRL) below the shared layer to address the minimax optimization problem. During the forward propagation, GRL acts as an identity transform (i.e., multiplies it by 1). During the backpropagation, the GRL takes the gradient from the subsequent level and changes its sign, i.e., $\lambda\mathcal{L}_{\mathcal{T}_j}^{adv}(\theta,\varphi)$ is is effectively replaced with $-\lambda\mathcal{L}_{\mathcal{T}_j}^{adv}(\theta,\varphi)$. Formally, we define the GRL as a function $R_\lambda(\boldsymbol{x})$ by two two equations describing the forward- and backpropagation behaviours:

$$R_\lambda(\boldsymbol{x}) = \boldsymbol{I} \tag{17}$$

$$\frac{dR_\lambda(\boldsymbol{x})}{d\boldsymbol{x}} = -\lambda\boldsymbol{I} \tag{18}$$

where $\boldsymbol{I}$ is an identity matrix. This adversarial training strategy will lead to the emergence of features that are domain-invariant and discriminative at the same time.

3.2.4 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 [28]:

$$\theta_i^{old} = \theta_{i-1} - \alpha\nabla_{\theta_{i-1}}\mathcal{L}_{\mathcal{T}_i}^{tr}(\theta_{i-1},\phi_{i-1}) \tag{19}$$

$$\theta_i^{new} = \theta_i^{old} - \alpha\nabla_{\theta_{i-1}}\lambda\mathcal{L}_{\mathcal{T}_i}^{dis}(\theta_{i-1},\gamma_{i-1}) \tag{20}$$

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 $\mathcal{L}_{\mathcal{T}_i}^{dis}$ 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, METASEQ 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

$$\mathcal{L}_{\mathcal{T}_j}^{val}(\theta_i^{old},\theta_i^{new},\phi_0,\gamma_0) = \mathcal{L}_{\mathcal{T}_j}^{tr}(\theta_i^{old},\phi_0) + \mathcal{L}_{\mathcal{T}_j}^{dis}(\theta_i^{new},\gamma_0) \tag{21}$$

Equation (21) can make the value range and gradient more stable [28]. The base model is updated by gradient descent:

$$\theta_0 \leftarrow \theta_0 - \beta\nabla_{\theta_0}\sum_{\mathcal{T}_i}(\mathcal{L}_{\mathcal{T}_i}^{tr}(\theta,\phi) - \lambda\mathcal{L}_{\mathcal{T}_i}^{dis}(\theta,\gamma)) \tag{22}$$

$$\phi_0 \leftarrow \phi_0 - \beta\nabla_{\phi_0}\sum_{\mathcal{T}_j}\mathcal{L}_{\mathcal{T}_j}^{val} \tag{23}$$

$$\gamma_0 \leftarrow \gamma_0 - \beta\nabla_{\gamma_0}\sum_{\mathcal{T}_j}\mathcal{L}_{\mathcal{T}_j}^{val} \tag{24}$$

where β is the meta-learning rate. Note that Equations (23) and (24) are computed by differentiating the loss $\mathcal{L}_{\mathcal{T}_i}^{val}$ with respect to the parameters ϕ_0,φ_0. Unlike the common gradient, the update mechanism of Equations (23) and (24) 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.

Algorithm 1: Training and Testing METASEQ

Training `Procedure()`
  Input: $\mathcal{D} = \{\mathcal{D}_1, ..., \mathcal{D}_N\}$, and α, β
  Initialize θ, ϕ, γ;
  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 θ_{i-1}; // $\mathcal{T}_i, \nabla_{\theta_{i-1}}$
        $\theta^{old}_i = \theta_{i-1} - \alpha \nabla_{\theta_{i-1}} \mathcal{L}^{tr}_{\mathcal{T}_i}(\theta_{i-1}, \phi_{i-1})$;
        $\theta^{new}_i = \theta^{old}_i - \alpha \nabla_{\theta_{i-1}} \lambda \mathcal{L}^{dis}_{\mathcal{T}_i}(\theta_{i-1}, \phi_{i-1})$;
      Meta-validation:
      Compute meta-validation loss on $\mathcal{T}_j$:
      $\mathcal{L}^{val}_{\mathcal{T}_j}(\theta^{old}_i, \theta^{new}_i, \phi_0, \gamma_0)$;
    Meta-optimization:
    Perform gradient step w.r.t Φ:
    $\theta_0 \leftarrow \theta_0 - \beta \nabla_{\theta_0} \sum_{\mathcal{T}_i} \mathcal{L}^{adv}_{\mathcal{T}_i}$; // $\mathcal{T}_i, \nabla_{\theta_0}$
    $\phi_0 \leftarrow \phi_0 - \beta \nabla_{\phi_0} \sum_{\mathcal{T}_j} \mathcal{L}^{val}_{\mathcal{T}_j}$; // $\mathcal{T}_j, \nabla_{\phi_0}$
    $\gamma_0 \leftarrow \gamma_0 - \beta \nabla_{\gamma_0} \sum_{\mathcal{T}_j} \mathcal{L}^{val}_{\mathcal{T}_j}$; // $\mathcal{T}_j, \nabla_{\gamma_0}$
  return $\theta_{\text{Meta}}, \phi_{\text{Meta}}, \gamma_{\text{Meta}}$
Testing `Procedure()`
  Input: Training set S_{tr} and test set S_{te} of an unseen domain $\mathcal{D}_{new}$
  Initialize θ from θ_{Meta};
  Instantiate a new tag decoder ϕ;
  while *available training data* **do**
    Sample a task $\mathcal{T}_{tr}$ from S_{tr};
    Update $\theta \leftarrow \theta - \beta \nabla_{\theta} \sum_{\mathcal{T}_{tr}} \mathcal{L}^{tr}_{\mathcal{T}_{tr}}$
    Update $\phi \leftarrow \phi - \beta \nabla_{\phi} \sum_{\mathcal{T}_{tr}} \mathcal{L}^{tr}_{\mathcal{T}_{tr}}$
    return Optimal θ^* and ϕ^*
  $Precison, Recall, F1 = f_{\mathcal{T}_{te}}(\theta^*, \phi^*)$
  return Sequence labeling performance on S_{te}

3.2.5 Algorithm Flow

The pseudocodes for training and testing METASEQ are given in Algorithm 1. At each iteration during training, 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, METASEQ takes a gradient step to get new adaptation parameters, and obtains the new meta-validation loss. In the outer loop, METASEQ uses the validation on $\mathcal{D}_{val}$ to differentiate through the inner loop and update the parameters of the base model: $\theta_0, \phi_0, \varphi_0$. During testing, METASEQ initialize the sequence encoder with θ_{Meta} and instantiate a new tag decoder ϕ.

4 EXPERIMENTS

4.1 Experimental Setups

4.1.1 Baseline Methods

We evaluate METASEQ 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 [18].
- **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 [50].
- **DANN** - This is an unsupervised domain adaptation approach with adversarial training [61]. 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 [62]. It is first trained on source domains and then retrained on target domains for token-level classification.
- **METASEQ-Zero** - This is trained on source domains with the Algorithm 1, without fine-tuning.

TABLE 1
Statistics of the ACE2005 dataset used in homogeneous NER.

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

4.1.2 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 METASEQ is implemented with the PyTorch framework and evaluated on NVIDIA Tesla V100 GPUs. Note that METASEQ 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.

TABLE 2
Performance (F1 Scores, %) of homogeneous domain adaptation for NER. The best performance is in boldface and the second best is underlined.

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

4.2 Homogeneous Domain Adaptation for NER

4.2.1 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 [28], [63], 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.

4.2.2 Experimental Results

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

First, METASEQ outperforms all baseline methods in terms of F1 scores. More specifically, our model outperforms In-Domain, D-Shift, AGG, FineTuning, MultiTask, DANN, WPZ and METASEQ-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 METASEQ explicitly simulates domain shift during training via meta-learning, which is helpful for adapting to a novel target domain.

Second, D-Shift and METASEQ-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 (METASEQ-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, METASEQ 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 METASEQ with respect to the data ratios of the

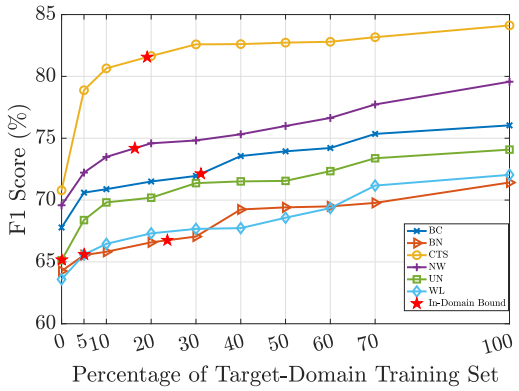


Fig. 4. F1 score (METASEQ) w.r.t. the percentage of the target domain training set for different target domains on homogeneous NER. METASEQ surpasses the In-Domain performance using only 16.17% of training data of target domains, on average.

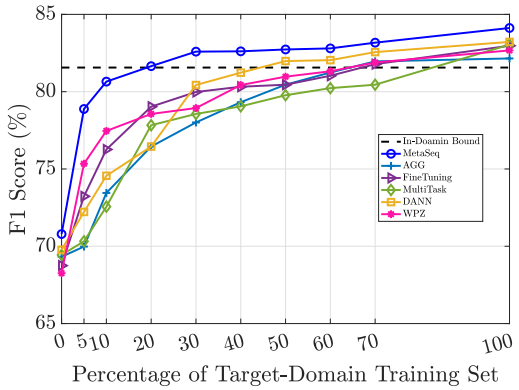


Fig. 5. F1 score w.r.t. the percentage of the target domain training set for different methods on the CTS target domain (homogeneous NER).

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

TABLE 3
Statistics of the datasets used in heterogeneous NER.

Datasets	#Types	#Sentences			#Mentions
		Train	Dev	Test	
Source Domains					
CoNLL03	4	14041	3250	3453	34841
OntoNotes5.0	18	59917	8525	8262	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

target domain training data across different target domains. Impressively, METASEQ 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 baselines perform on par with the In-Domain model (In-Domain bound) using 67% of training data on CTS. Compared with these baselines, METASEQ 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.

4.3 Heterogeneous Domain Adaptation for NER

4.3.1 Datasets and Setups

For the heterogeneous domain adaptation, we use four datasets as source domains and five datasets as target domains. Table 3 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 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 METASEQ-Zero cannot be used in heterogeneous settings because source and target domains have different label spaces.

4.3.2 Experimental Results

Table 4 reports the results of different methods under heterogeneous settings. We make the following observations:

First, METASEQ 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.

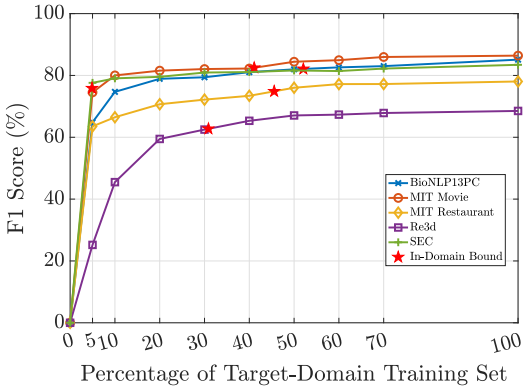


Fig. 6. F1 score (METASEQ) w.r.t. the percentage of the target domain training set for different target domains (heterogeneous NER). METASEQ surpasses the In-Domain performance using only 34.76% of training data of target domains, on average.

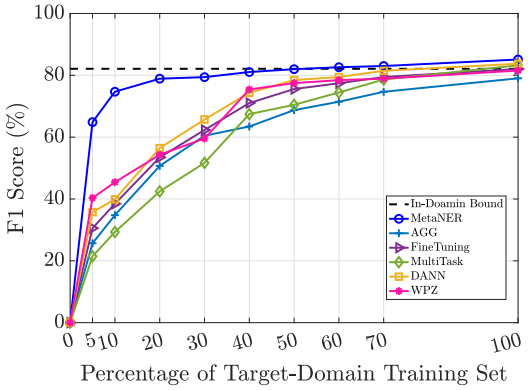


Fig. 7. F1 score w.r.t. the percentage of the target domain training set for different methods on the BioNLP13PC (heterogeneous NER).

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 [28]. 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 METASEQ with respect to the data ratios of the target domain training sets across different target domains. Different from the previous homogeneous adaptation cases, METASEQ needs more training data from target domains to outperform the In-Domain method. More specifically, METASEQ surpasses the performance of In-Domain methods using only 52%, 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

TABLE 4
Performance of the heterogeneous domain adaptation for NER. The best performance is in boldface and the second best is underlined.

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

TABLE 5
Statistics of datasets used in homogeneous POS.

Domains	#Types	#Sentences		
		Train	Dev	Test
adventure	81	3696	462	463
belles_lettres	81	5705	713	714
learned	81	6140	768	768
lore	81	3866	483	484
fiction	81	3370	421	422
news	81	3656	457	458
romance	81	3535	442	442

for different methods on the BioNLP13PC target domain. Compared with the baseline methods, METASEQ 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.

4.4 Homogeneous Domain Adaptation for POS

4.4.1 Datasets and Setups

In this experiment, we use 7 domains of Brown corpus: adventure, belles_lettres, learned, lore, fiction, news and romance. Table 5 summarizes the statistics of these 7 datasets. Note that the corpus originally contains 86 POS tags. In the homogeneous setting, we use a intersection set of these 7 domains, resulting in a set of 81 tags.

We also adopt the leave-one-out evaluation protocol by picking one domain to hold out as the target domain for the final evaluation. For each iteration during training, six source domains are randomly chosen as the meta-training domains, and the remaining one as the meta-validation domain.

4.4.2 Experimental Results

Table 6 reports the results of different methods under the leave-one-out settings. We make the following observations:

First, METASEQ outperforms all baseline methods in terms of accuracy. More specifically, our model outperforms In-Domain, D-Shift, AGG, FineTuning, MultiTask, DANN, WPZ and METASEQ-Zero by relative accuracy improvements of 4.33%, 6.38%, 3.16%, 2.09%, 1.60%, 1.71%, 3.35% and 4.70%.

Second, the performance of In-Domain is slightly better than the one of D-Shift. This is because the domain differences among these 7 datasets are small. However, the zero-shot version of our approach (METASEQ-Zero) still outperforms the direct domain shift method (D-Shift), and is comparable with In-Domain.

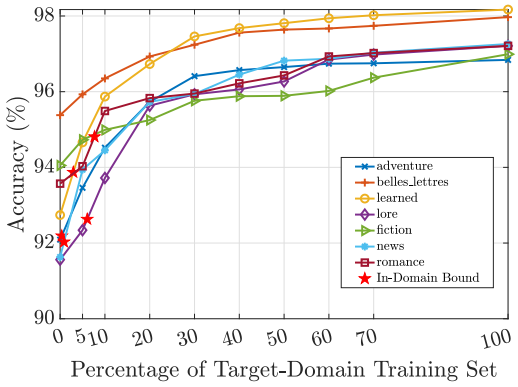


Fig. 8. Accuracy (METASEQ) w.r.t. the percentage of the target domain training set for different target domains on homogeneous POS.

Third, AGG and WPZ slightly outperform In-Domain in terms of the average accuracy scores. Aggregating the training sets across all source and target domains can slightly enhance the performance of the In-Domain method. In addition, FineTuning, MultiTask and DANN achieve comparable performance.

Furthermore, 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 8 illustrates the accuracy of METASEQ with respect to the data ratios of the target domain training data across different target domains. For belles_lettres and fiction domains, METASEQ-Zero outperform In-domain. METASEQ surpasses the performance of In-Domain methods using only 5%, 5%, 10%, 5%, 10% of the target domain training sets for adventure, learned, lore, news, and romance, respectively.

Figure 9 shows the accuracy scores with respect to the data ratios of the target domain training sets for different methods on the adventure target domain. Compared with these baseline methods, METASEQ surpasses the In-Domain bound using less training data (less than 5%).

4.5 Heterogeneous Domain Adaptation for POS

4.5.1 Datasets and Setups

For the heterogeneous POS, we use three datasets as source domains and six datasets as target domains. Table 7 summarizes the statistics of these ten datasets. WSJ, SWBD and ATIS are the Wall Street Journal, Switchboard and Air Travel Information System transcripts of Treebak3, respectively. HOB, REL and GOV are from the genres of 'hobbies',

TABLE 6
Performance (Accuracy, %) of homogeneous domain adaptation for POS. The best performance is in boldface and the second best is underlined.

Target Domains	In-Domain	D-Shift	AGG	FineTuning	MultiTask	DANN	WPZ	METASEQ-Zero	METASEQ
adventure	92.19	91.03	94.47	95.21	94.92	95.83	94.52	92.09	**96.84**
belles_lettres	94.20	93.11	95.31	95.98	96.34	94.87	93.41	95.38	**97.97**
learned	93.87	90.20	94.25	95.63	96.84	97.43	94.76	92.74	**98.17**
lore	92.63	91.26	93.55	94.37	95.72	94.28	94.09	91.56	**97.21**
fiction	93.57	93.38	94.73	95.62	95.02	96.02	94.85	94.05	**96.99**
news	92.03	90.33	92.98	94.52	96.38	95.34	93.72	91.63	**97.26**
romance	94.81	91.46	95.44	96.38	95.72	96.42	94.21	93.57	**97.21**
Average	93.32	91.54	94.39	95.39	95.85	95.74	94.23	93.00	**97.38**
Improvement	↑4.33%	↑6.38%	↑3.16%	↑2.09%	↑1.60%	↑1.71%	↑3.35%	↑4.70%	-

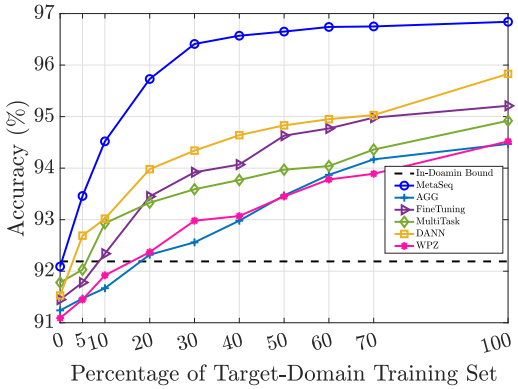


Fig. 9. Accuracy w.r.t. the percentage of the target domain training set for different methods on the 'adventure' domain (homogeneous POS).

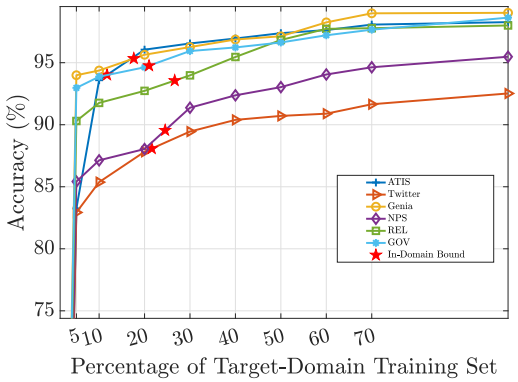


Fig. 10. Accuracy w.r.t. the percentage of the target domain training set for different target domains on heterogeneous POS.

TABLE 7
Statistics of the datasets used in heterogeneous POS.

Datasets	Domains	#Types	#Sentences		
			Train	Dev	Test
Source Domains					
WSJ	Newswires	45	41094	6265	7191
SWBD	Telephone speech	45	122990	15374	15374
HOB	Hobbies	86	3350	419	419
Target Domains					
ATIS	Air travel	32	452	57	57
Twitter	Social media	49	1121	140	141
Genia	Medical	47	16435	2054	2055
NPS	Online chat	46	8099	1012	1013
REL	Religion	85	1372	172	172
GOV	Governmet	83	2424	303	304

'religion' and 'government' of Brown corpus. Twitter is the dataset from social media. Genia is the primary collection of biomedical literature compiled and annotated within the scope of the GENIA project. NPS is gathered from various online chat services in accordance with their terms of service. For each iteration during training, two source domains are randomly chosen as the meta-training domains, and the remaining one as the meta-validation domain.

4.5.2 Experimental Results

Table 8 reports the results of different methods under heterogeneous POS settings. We make the following observations:

First, METASEQ outperforms all competitors in terms of accuracy scores. More specifically, our model outperforms In-Domain, AGG, Fine-Tuning, MultiTask, DANN and WPZ by relative F1 improvements of 4.79%, 11.56%, 3.73%, 2.22%, 1.64% and 3.09%, respectively.

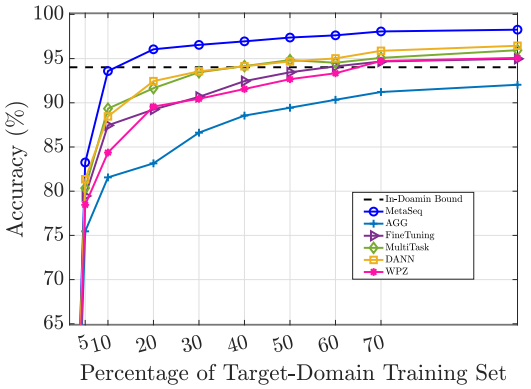


Fig. 11. Accuracy w.r.t. the percentage of the target domain training set for different methods on ATIS (heterogeneous POS).

Second, the performance of AGG is worse than In-Domain in all target domains. This empirical results show that naively aggregating source and target domains can worsen the performance of the In-Domain method for POS. This is because of the huge difference between source and target domain text genres, and the label space discrepancy.

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 10 illustrates the accuracy scores of METASEQ with respect to the data ratios of the target domain training sets across different target domains. More specifically, METASEQ surpasses the performance of In-Domain methods using only 10%, 30%, 20%, 30% and

TABLE 8
Performance (Accuracy, %) of heterogeneous domain adaptation for POS. The best performance is in boldface and the second best is underlined.

Target Domains	In-Domain	D-Shift	AGG	FineTuning	MultiTask	DANN	WPZ	METASEQ-Zero	METASEQ
ATIS	94.02	-	92.04	94.98	95.95	96.45	95.08	-	**98.27**
Twitter	88.08	-	84.82	89.45	91.34	90.45	89.32	-	**92.52**
Genia	95.34	-	93.73	96.33	97.44	98.03	96.98	-	**99.02**
NPS	89.55	-	85.73	90.77	92.45	93.42	91.34	-	**95.48**
REL	93.56	-	82.55	94.13	95.45	96.78	94.98	-	**98.00**
GOV	94.76	-	82.74	95.33	96.64	97.42	96.74	-	**98.63**
Average	92.55	-	86.94	93.49	94.88	95.43	94.07	-	**96.99**
Improvement	↑4.79%	-	↑11.56%	↑3.73%	↑2.22%	↑1.64%	↑3.09%	-	-

TABLE 9
Statistics of the datasets used in heterogeneous slot filling.

Domains	#Types	#Sentences		
		Train	Dev	Test
Source Domains				
AddToPlaylist	5	1747	195	100
GetWeather	9	1800	200	100
SearchScreeningEvent	7	1763	196	100
Target Domains				
BookRestaurant	14	1775	198	100
PlayMusic	9	1800	200	100
RateBook	7	1760	196	100
SearchCreativeWork	2	1758	196	100

30% of training data for ATIS, Twitter, Genia, NPS, REL and GOV. Figure 11 shows the accuracy scores with respect to the data ratios of the target domain training sets for different methods on the ATIS domain. Compared with the baseline methods, METASEQ quickly achieves the same performance as In-Domain (In-Domain bound) using less training data (around 10%).

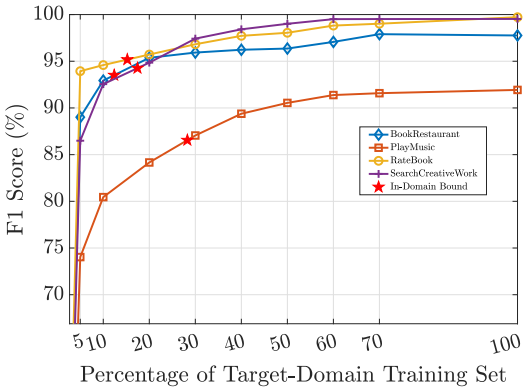


Fig. 12. F1 score w.r.t. the percentage of the target domain training set for different domains on heterogeneous slot filling.

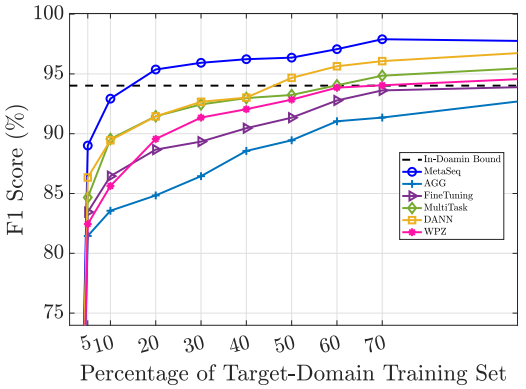


Fig. 13. F1 score w.r.t. the percentage of the target domain training set for different methods on BookRestaurant (heterogeneous slot filling).

4.6 Heterogeneous Domain Adaptation for Slot Filling

4.6.1 Datasets and Setups

For the heterogeneous domain adaptation for slot filling, we use the SNIPS corpus which consists of 7 domains. Table 9 summarizes the statistics of these 7 domains. In particular, AddToPlaylist, GetWeather and SearchScreeningEvent are considered as source domains. BookRestaurant, PlayMusic, RateBook and SearchCreativeWork are chosen as target domains. For each iteration during training, two source domains are randomly chosen as the meta-training domains, and the remaining one as the meta-validation domain.

4.6.2 Experimental Results

Table 10 reports the results of different methods under heterogeneous slot filling settings. We make the following observations:

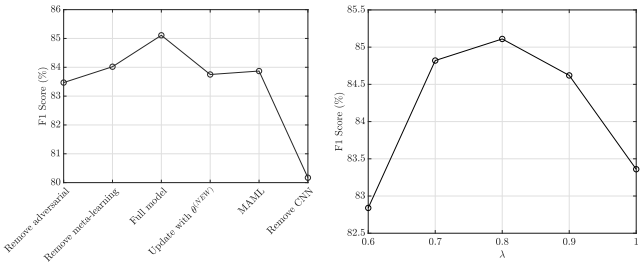
First, METASEQ outperforms all competitors in terms of F1 scores. More specifically, our model outperforms In-Domain, AGG, Fine-Tuning, MultiTask, DANN and WPZ by relative F1 improvements of 5.26%, 6.93%, 4.31%, 2.54%, 2.69% and 4.01%, respectively.

Second, on average across all target domains, the performance of In-Domain is slightly better than AGG, and all other baselines (i.e., Fine- Tuning, MultiTask, DANN and WPZ) are better than In-Domain. Simply aggregating source and target domains hurts the performance of the In-Domain method,

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 12 illustrates the F1 scores of METASEQ with respect to the data ratios of the target domain training sets across different target domains. More specifically, METASEQ surpasses the performance of In-Domain methods using only 20%, 30%, 20%, and 20% of training data for BookRestaurant, PlayMusic, RateBook and SearchCreativeWork, respectively. Figure 7 shows the F1 scores with respect to the data ratios of the target domain training sets for different methods on the BookRestaurant domain. Compared with the baseline methods, METASEQ quickly achieves the same performance as In-Domain (In-Domain bound) using less training data (around 15%).

TABLE 10
Performance (F1 Scores, %) of heterogeneous slot filling. The best performance is in boldface and the second best is underlined.

Target Domains	In-Domain	D-Shift	AGG	FineTuning	MultiTask	DANN	WPZ	METASEQ-Zero	METASEQ
BookRestaurant	93.51	-	92.7	93.89	95.46	96.74	94.56	-	**97.76**
PlayMusic	86.55	-	83.22	87.88	89.35	88.25	87.47	-	**91.93**
RateBook	95.18	-	94.42	95.67	97.64	96.41	96.23	-	**99.72**
SearchCreativeWork	94.26	-	93.41	95.43	96.83	97.37	95.64	-	**99.53**
Average	92.38	-	90.94	93.21	94.82	94.69	93.48	-	**97.26**
Improvement	↑5.26%	-	↑6.93%	↑4.31%	↑2.54%	↑2.69%	↑4.02%	-	-



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

Fig. 14. Impact of architectural choices and parameter λ.

4.7 Further Analysis

4.7.1 Ablation Study

Table 14(a) reports an ablation analysis on the test set of BioNLP13PC. The full model is our proposed METASEQ. 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 [23], 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 METASEQ.

Table 14(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.

4.7.2 Qualitative Analysis

Because we are more interested in low-resource scenarios, we train METASEQ using only 10% of the training data for all target domains. Table 11 shows some positive and negative examples for homogeneous and heterogeneous domain adaptations.

For the homogeneous NER, we only keep the inner-most 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 $[\![\text{U.S. }]\!]_{\text{Gpe}}$, while our result is $[\![\text{U.S. Army}]\!]_{\text{Org}}$. For the negative example in BN, the ground truth entity is $[\![\text{major league baseball}]\!]_{\text{Org}}$, while METASEQ fails to detect the correct boundaries of this entity. For the example in CTS, METASEQ misses an entity $[\![\text{y'all}]\!]_{\text{Org}}$. From the negative examples in the heterogeneous NER, we also observe that METASEQ misses some entities and wrongly detects the boundaries of others. For POS, the ground truth tags are 'dried/VBD', 'Northumberland/NP', 'Guilford-Martin/NP', 'connecting/VBG', 'iguess/VBP', 'control/JJ' and 'depressed/JJ'. For slot filling, the ground truth tags are $[\![\text{tatar}]\!]_{\text{cuisine}}$, $[\![\text{Ready To Die}]\!]_{\text{album}}$. In summary, the different annotation criteria in different domains are the key factors affecting transferability. Although METASEQ 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.

5 CONCLUSION

In this paper, we proposed METASEQ, a novel meta-learning approach for both homogeneous and heterogeneous domain adaptations in sequence labeling. In particular, METASEQ can effectively learn a robust and general sequence encoder from multiple source domains. The key advantage of METASEQ is that it can accurately adapt to unseen domains with a small amount of data. We conducted extensive experiments on NER, POS and slot filling tasks under homogeneous and heterogeneous domain adaptation settings. 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.

ACKNOWLEDGMENT

This work was supported by the NSFC (U2001212, 62032001 and 61932004).

REFERENCES

[1] J. Li, S. Shang, and L. Shao, "Metaner: Named entity recognition with meta-learning," in *The Web Conference 2020 (WWW)*, 2020, pp. 429–440.
[2] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Trans. Knowl. Data Eng.*, 2020.
[3] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.
[4] B. Plank and Z. Agic, "Distant supervision from disparate sources for low-resource part-of-speech tagging," in *EMNLP*, E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds., 2018, pp. 614–620.
[5] H. E, P. Niu, Z. Chen, and M. Song, "A novel bi-directional interrelated model for joint intent detection and slot filling," in *ACL*, A. Korhonen, D. R. Traum, and L. Màrquez, Eds., 2019, pp. 5467–5471.
[6] J. Guo, G. Xu, X. Cheng, and H. Li, "Named entity recognition in query," in *SIGIR*, 2009, pp. 267–274.
[7] C. Nobata, S. Sekine, H. Isahara, and R. Grishman, "Summarization system integrated with named entity tagging and IE pattern discovery," in *LREC*, 2002.
[8] X. Ma and E. H. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," in *ACL*, 2016, pp. 1064–1074.

TABLE 11
Positive and negative examples for METASEQ with 10% of the training data from target domains. The results produced by METASEQ are marked with ⟦ ⟧ in NER and slot filling. The green and red highlights indicate a correct and incorrect (missed) result, respectively.

Tasks	Targets	Sequence Labeling Results by METASEQ	Correct
Homo-NER	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⟧$_{\text{Veh}}$.	✓
Heter-NER	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⟧$_{\text{Loc}}$.	✗
Homo-POS	adventure	The/AT burning/VBG air/NN dried/VBN his/PP$ sweat-soaked/JJ clothes/NNS in/IN salt-edged/JJ patches/NNS ./.	✗
	belles_lettres	And/CC Grey's/NP$ Northumberland/JJ background/NN was/BEDZ close/JJ to/IN Trevelyan's/NP$ own/JJ ./.	✗
	learned	Guilford-Martin/JJ personality/NN inventories/NNS ./.	✗
	lore	She/PPS is/BEZ even/RB prone/JJ to/TO regard/VB the/AT college/NN girl/NN as/CS immature/JJ	✓
	fiction	He/PPS would/MD take/VB his/PP$ leave/NN of/IN Michelangelo/NP by/IN announcing/VBG	✓
	news	It/PPS will/MD be/BE preceded/VBN by/IN luncheon/NN in/IN the/AT Teter/NP House/NN ./.	✓
	romance	Fights/NNS sprang/VBD up/RP and/CC were/BED quickly/RB squelched/VBN ./.	✓
Heter-POS	ATIS	What/WDT airline/NN provides/VBZ only/RB connecting/JJ flights/NNS between/IN Denver/NNP and/CC San/NNP Francisco/NNP	✗
	Twitter	friday/NNP night/NN missions/NNS iguess/NNS !/.	✗
	Genia	Compared/VBN to/TO the/DT control/NN group/NN ,/, fewer/JJR depressed/VBN subjects/NNS down-regulated/VBD Bmax/NN after/IN DEX/NN ./.	✗
	NPS	well/UH ya/PRP already/RB said/VBD ya/PRP saw/VBD me/PRP nakey/JJ ,/, so/UH can/MD we/PRP be/VB nakey/JJ together/RB ?/.	✓
	REL	You/PPSS can/MD do/DO likewise/RB though/CS Christ/NP is/BEZ not/* bodily/RB present/RB	✓
	GOV	There/EX will/MD of/IN course/NN be/BE times/NNS for/IN delay/NN and/CC inaction/NN ./.	✓
Heter-Slot	BookRestaurant	Book a reservation for ⟦five⟧$_{\text{party\_size\_number}}$ people for a ⟦tatar⟧$_{\text{served\_dish}}$ ⟦taverna⟧$_{\text{restaurant\_type}}$ in ⟦Sargents⟧$_{\text{country}}$	✗
	PlayMusic	Open ⟦itunes⟧$_{\text{service}}$ and play ⟦Ben Burnley⟧$_{\text{artist}}$ ⟦Ready To Die⟧$_{\text{track}}$	✗
	RateBook	rate ⟦this⟧$_{\text{object\_select}}$ ⟦album⟧$_{\text{object\_type}}$ ⟦four⟧$_{\text{rating\_value}}$ out of ⟦6⟧$_{\text{best\_rating}}$ ⟦stars⟧$_{\text{rating\_unit}}$	✓
	SearchScreeningEvent	I need ⟦animated movies⟧$_{\text{moive\_type}}$ ⟦in the area ⟧$_{\text{spatial\_relation}}$ for ⟦dinner⟧$_{\text{timeRange}}$ time	✓

[9] 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.

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

[11] 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.

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

[13] 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.

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

[15] S. J. Pan, Z. Toh, and J. Su, "Transfer joint embedding for cross-domain named entity recognition," *ACM Trans. Inf. Syst.*, vol. 31, no. 2, p. 7, 2013.

[16] L. Qu, G. Ferraro, L. Zhou, W. Hou, and T. Baldwin, "Named entity recognition for novel types by transfer learning," in *EMNLP*, 2016, pp. 899–905.

[17] Y. Kim, K. Stratos, R. Sarikaya, and M. Jeong, "New transfer learning techniques for disparate label sets," in *ACL*, 2015, pp. 473–482.

[18] J. Y. Lee, F. Dernoncourt, and P. Szolovits, "Transfer learning for named-entity recognition with neural networks," in *LREC*, 2018.

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

[20] W. Cui, G. Zheng, Z. Shen, S. Jiang, and W. Wang, "Transfer learning for sequences via learning to collocate," in *ICLR*, 2019.

[21] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell, "Meta-learning with latent embedding optimization," in *ICLR*, 2019.

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

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

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

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

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

[27] K. Qian and Z. Yu, "Domain adaptive dialog generation via meta learning," in *ACL*, 2019, pp. 2639–2649.

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

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

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

[32] 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.

[33] 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.

[34] J. Li, D. Ye, and S. Shang, "Adversarial transfer for named entity boundary detection with pointer networks," in *IJCAI*, 2019, pp. 5053–5059.
[35] J. Li, A. Sun, and S. R. Joty, "Segbot: A generic neural text segmentation model with pointer network," in *IJCAI*, 2018, pp. 4166–4172.
[36] N. Ding, G. Xu, Y. Chen, X. Wang, X. Han, P. Xie, H. Zheng, and Z. Liu, "Few-nerd: A few-shot named entity recognition dataset," in *ACL*, 2021.
[37] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010.
[38] 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.
[39] 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.
[40] 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.
[41] 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.
[42] N. Houlsby, A. Giurgiu, S. Jastrzkebski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP," in *ICML*, 2019, pp. 2790–2799.
[43] H. He and X. Sun, "A unified model for cross-domain and semi-supervised named entity recognition in chinese social media," in *AAAI*, 2017, pp. 3216–3222.
[44] J. D. Rodríguez, A. Caldwell, and A. Liu, "Transfer learning for entity recognition of novel classes," in *COLING*, 2018, pp. 1974–1985.
[45] X. Feng, X. Feng, B. Qin, Z. Feng, and T. Liu, "Improving low resource named entity recognition using cross-lingual knowledge transfer," in *IJCAI*, 2018, pp. 4071–4077.
[46] Z. Wang, Y. Qu, L. Chen, J. Shen, W. Zhang, S. Zhang, Y. Gao, G. Gu, K. Chen, and Y. Yu, "Label-aware double transfer learning for cross-specialty medical named entity recognition," in *NAACL-HLT*, 2018, pp. 1–15.
[47] 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.
[48] C. Jia, L. Xiao, and Y. Zhang, "Cross-domain NER using cross-domain language modeling," in *ACL*, 2019, pp. 2464–2474.
[49] 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.
[50] Z. Yang, R. Salakhutdinov, and W. W. Cohen, "Transfer learning for sequence tagging with hierarchical recurrent networks," in *ICLR*, 2017.
[51] P. von Däniken and M. Cieliebak, "Transfer learning and sentence level features for named entity recognition on tweets," in *WNUT*, 2017, pp. 166–171.
[52] S. Thrun and L. Y. Pratt, Eds., *Learning to Learn*. Springer, 1998.
[53] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *NIPS*, 2017, pp. 4077–4087.
[54] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *ICLR*, 2017.
[55] 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.
[56] A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine, "Meta-learning with implicit gradients," *CoRR*, vol. abs/1909.04630, 2019.
[57] Y. Wang, R. B. Girshick, M. Hebert, and B. Hariharan, "Low-shot learning from imaginary data," in *CVPR*, 2018, pp. 7278–7286.
[58] 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.
[59] 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.
[60] A. Obamuyide and A. Vlachos, "Model-agnostic meta-learning for relation classification with limited supervision," in *ACL*, 2019, pp. 5873–5879.
[61] Y. Ganin and V. S. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *ICML*, 2015, pp. 1180–1189.
[62] A. Fritzler, V. Logacheva, and M. Kretov, "Few-shot classification in named entity recognition task," in *SAC*, 2019, pp. 993–1000.
[63] S. Shankar, V. Piratla, S. Chakrabarti, S. Chaudhuri, P. Jyothi, and S. Sarawagi, "Generalizing across domains via cross-gradient training," in *ICLR*, 2018.

```
@article{jing21metaseq,
author    = {Jing Li and Peng Han and Xiangnan Ren and Jilin Hu and Lisi Chen and Shuo Shang},
title     = {Sequence Labeling with Meta-Learning},
journal   = {IEEE Transactions on Knowledge and Data Engineering (TKDE), Early Access},
year      = {2021},
url       = {https://doi.org/10.1109/TKDE.2021.3118469},
doi       = {10.1109/TKDE.2021.3118469},
}
```