MANNER: A Variational Memory-Augmented Model for Cross Domain Few-Shot Named Entity Recognition

Jinyuan Fang[1,*], **Xiaobin Wang**[2], **Zaiqiao Meng**[1], **Pengjun Xie**[2], **Fei Huang**[2], **Yong Jiang**[2,†]
[1] School of Computing Science, University of Glasgow
[2] DAMO Academy, Alibaba Group
fangjy6@gmail.com, xuanjie.wxb@alibaba-inc.com,
zaiqiao.meng@glasgow.ac.uk, yongjiang.jy@alibaba-inc.com

Abstract

This paper focuses on the task of cross domain few-shot named entity recognition (NER), which aims to adapt the knowledge learned from source domain to recognize named entities in target domain with only a few labeled examples. To address this challenging task, we propose MANNER, a variational memory-augmented few-shot NER model. Specifically, MANNER uses a memory module to store information from the source domain and then retrieve relevant information from the memory to augment few-shot tasks in the target domain. In order to effectively utilize the information from memory, MANNER uses optimal transport to retrieve and process information from memory, which can explicitly adapt the retrieved information from source domain to target domain and improve the performance in the cross domain few-shot setting. We conduct experiments on both English and Chinese cross domain few-shot NER datasets, and the experimental results demonstrate that MANNER can achieve superior performance [1].

1 Introduction

Named Entity Recognition (NER) is a fundamental NLP task that aims at classifying mention spans into entity types. Previous works mainly study the NER task in a supervised setting (Chiu and Nichols, 2016; Devlin et al., 2019; Yamada et al., 2020). However, supervised learning requires large-scale annotated datasets, which can be difficult to obtain in some scenarios (e.g., annotating biomedical named entities always requires domain expertise (Ogren et al., 2008)). In this paper, we focus on a more practical and challenging setting in real-world applications, namely *cross domain few-shot NER* — given a source domain with sufficient labeled data and a target domain with a few labeled data, the goal is to correctly recognize named entities in the target domain (Hou et al., 2020). This is achieved by adapting the knowledge learned from the source domain to the target domain based on few-shot examples available in the target domain. Figure 1 provides an illustration of the cross domain few-shot NER task.

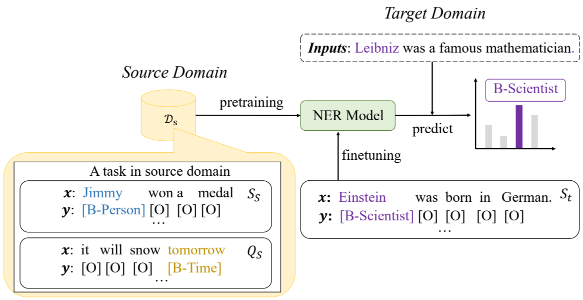

Figure 1: Illustration of the cross domain few-shot NER task, where the NER model is first pretrained on a set of tasks (each task has a support set, e.g., $\mathcal{S}_s$ and a query set, e.g., $\mathcal{Q}_s$) in source domain and then adapted to a few-shot task in target domain with a few labeled data.

Recent work demonstrated that learning prototype representations for each label class could be effective to address few-shot tasks (Snell et al., 2017), and this idea has also been applied to few-shot NER tasks (Fritzler et al., 2019; Huang et al., 2021; Ma et al., 2022). Specifically, when dealing with a few-shot task in the target domain, these models learn prototypes for each entity type based on a few labeled data available in the support set and then assign labels to tokens in the query set by measuring their distances to the prototypes. However, since there are only a few labeled examples for each entity type in the support set, the prototypes obtained from the support set only may not be accurate and representative, leading to the suboptimal and unstable performance of prototype-based few-shot NER models (Huang et al., 2020).

To this end, we propose a variational **M**emory-**A**ugme**N**ted cross domain few-shot **NER** model, abbreviated as **MANNER**. It introduces an external

*Work done when interning at Alibaba DAMO Academy.
†Yong Jiang is the corresponding author.
[1] Our code is publicly available at: https://github.com/Alibaba-NLP/MANNER

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 4261–4276
July 9-14, 2023 ©2023 Association for Computational Linguistics

memory module that utilizes information from the source domain to augment the support set in the target domain, so as to learn more accurate prototypes. The basic idea of introducing the memory module is that the entity type information from the source domain can provide additional background knowledge for learning prototypes in the target domain (Zhen et al., 2020). For example, in Figure 1, the information of the entity type "Person" in the source domain can provide guidance for recognizing the entity type "Scientist" in the target domain. Specifically, MANNER stores token representations of entity types from the source domain in a memory module. For each entity type in the target domain, MANNER first retrieves the most similar entity types from the memory and then leverages the retrieved information to learn prototype for the entity type.

One critical issue when using the memory module is how to utilize the information from the memory to augment few-shot tasks in the target domain. Recent research indicates that the performance of memory-augmented methods which directly use neural networks to fuse information from the memory and the task (He et al., 2020; Zhen et al., 2020), is suboptimal when dealing with cross domain tasks (Du et al., 2022), such as our cross domain few-shot NER task. This is because the knowledge (i.e., entity type information) of the source domain stored in the memory can be inconsistent with that of the target domain. Therefore, in cross domain few-shot NER tasks where the entity types of the source domain and target domain are disjoint, directly utilizing the information retrieved from the memory may be suboptimal. Actually, we empirically found that this could degrade the model performance (see § 4.2). To address this problem, we take inspiration from domain adaption and leverage optimal transport (Villani, 2009) to retrieve and process information from the memory. One benefit of using optimal transport is that we can adapt the retrieved information from the source domain to the target domain via the optimal transport plan. This adaption process helps alleviate the inconsistency problem between the two domains.

Our contributions can be summarized as follows: (1) We propose MANNER, a novel cross domain few-shot NER model, which uses a memory module to utilize the information from the source domain to augment few-shot NER tasks in the target domain. (2) We leverage optimal transport to retrieve and process information from the memory, which is conducive to improve the performance of MANNER in the cross domain setting. (3) Experimental results on English and Chinese cross domain few-shot NER datasets demonstrate that MANNER can achieve superior performance compared with existing few-shot NER models.

2 Preliminaries

In this section, we formalize the cross domain few-shot NER task, and provide a brief introduction to optimal transport, which serves as the foundation of our model.

Task Formulation. NER is a sequence labeling task, where each token in the sequence is assigned a label representing an entity class or "O" (not an entity). In this paper, we focus on a practical setting of NER, namely *corss domain few-shot NER* (Hou et al., 2020; Yang and Katiyar, 2020), where the NER model is first pretrained on data-sufficient source domain(s) $\mathcal{D}_s$ and then tranferred to target domain(s) $\mathcal{D}_t$ with only a few labeled examples.

Formally, we denote a sentence and its labels as $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$ and $\boldsymbol{y} = \{y_1, y_2, \ldots, y_n\}$, respectively. Following previous works (Hou et al., 2020; Ma et al., 2022), we adopt the episode learning paradigm in this paper, where we first pretrain the model on a set of tasks $\mathcal{D}_s = \{(\mathcal{S}_s, \mathcal{Q}_s)\}$ from the source domain and then adapt the model to another set of tasks $\mathcal{D}_t = \{(\mathcal{S}_t, \mathcal{Q}_t)\}$ from the target domain. Each task consists of a *support set* $\mathcal{S} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{N \times K}$ for task adaption, and a *query set* $\mathcal{Q} = \{(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})\}_{j=1}^{N \times K'}$ for evaluation, where N denotes the number of entity types in a task, K and K' denote the number of few-shot samples that belong to each entity type in the support set and the query set, respectively. Given a task in the target domain, the goal of our model is to predict the labels of sentences in the query set after adapting the model to the task with its support set (i.e., finetuning the model with the support set). Figure 1 provides an illustration of the cross domain few-shot NER task.

Optimal Transport. Cross domain few-shot NER task can be considered as a domain adaption task. Optimal transport (OT) is a widely used method to solve the domain adaption tasks in the field of computer vision (Courty et al., 2017a,b; Damodaran et al., 2018; Fatras et al., 2021). Specifically, OT is a metric that measures the distance

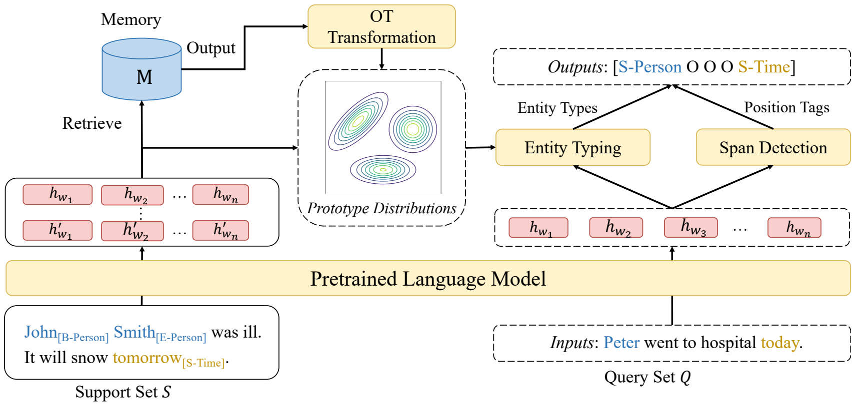

Figure 2: Overall framework of the proposed MANNER. For a few-shot task, MANNER first uses representations in the support set and the memory to infer prototype distributions, which are then leveraged in the entity typing module to predict the entity types of a sentence in the query set. MANNER also use a span detection module to predict the position tags of the query sentence. The predicted entity types and position tags are combined to obtain the final label.

between two probability distributions. In this paper, we focus on the discrete OT for two discrete empirical distributions, i.e., $\boldsymbol{\nu}_s$ and $\boldsymbol{\nu}_t$:

$$\mathcal{W}(\boldsymbol{\nu}_s, \boldsymbol{\nu}_t) = \min_{\mathbf{T} \in \Sigma(\boldsymbol{\nu}_s, \boldsymbol{\nu}_t)} \langle \mathbf{C}, \mathbf{T} \rangle , \quad (1)$$

where $\Sigma(\boldsymbol{\nu}_s, \boldsymbol{\nu}_t) = \{\mathbf{T} \in \mathbb{R}_+^{n\times m} : \mathbf{T}\mathbf{1}_m = \boldsymbol{\nu}_s, \mathbf{T}^\top \mathbf{1}_n = \boldsymbol{\nu}_t\}$ is a set of joint probabilities, $\mathbf{1}_m$ and $\mathbf{1}_n$ denote m-dimensional and n-dimensional vectors of ones respectively, $\langle \cdot, \cdot \rangle$ is the Frobenius dot product, and $\mathbf{C} = [c_{ij}] \in \mathbb{R}_+^{n\times m}$ is a cost matrix with each element representing the distance between the i-th data point of $\boldsymbol{\nu}_s$ and the j-th one of $\boldsymbol{\nu}_t$. The optimal solution of $\mathbf{T}$ is called *optimal transport plan*, denoted as $\mathbf{T}^*$, which can be efficiently obtained through the Sinkhorn algorithm (Cuturi, 2013) by solving an entropy regularized version of Equation (1) (see Appendix A).

3 Methodology

The overall framework of our MANNER is shown in Figure 2. In this section, we first introduce the details of MANNER in §3.1, and then introduce the pretraining of MANNER on the source domain in §3.2. We finally introduce how to adapt the model to the target domain in §3.3.

3.1 The MANNER Model

Following previous prototype-based NER models (Fritzler et al., 2019; Wang et al., 2021c), we learn a prototype for each entity type, which is the mean of representations of tokens that belong to this type in the support set. However, compared with vanilla prototype-based methods which model prototypes as deterministic vectors, we employ a probabilistic framework by modeling prototypes as stochastic variables, which is conducive to learn more informative prototypes and improve the robustness of few-shot models by capturing the uncertainties of prototypes (Allen et al., 2019; Zhen et al., 2020).

Moreover, following previous two-stage few-shot NER models (Wang et al., 2021b; Ma et al., 2022), we decompose the label prediction of NER into two sub-tasks: span detection which aims to predict the *position tags* of tokens, such as "B" and "I", and entity typing which aims to predict the *entity types* of tokens. Accordingly, for each sentence, we additionally introduce two types of labels, namely position tags $\boldsymbol{a} = \{a_1, a_2, \ldots, a_n\}$ and entity types $\boldsymbol{e} = \{e_1, e_2, \ldots, e_n\}$. We adopt the BIOES tagging scheme in this paper. Therefore, for a few-shot task $\tau = \{\mathcal{S}, \mathcal{Q}\}$[2], the position tags are chosen from $\{O, B, I, E, S\}$, while the entity types are chosen from the entity set $\mathcal{E}$ in the task, such as {Person, Location, ... }. We define the joint probability distribution of our model as:

$$\begin{aligned} & p_\theta(\boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e}, \mathbf{Z} \mid \boldsymbol{x}, \mathcal{S}, \mathbf{M}) \\ & \quad = p_\theta(\boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e}, \mid \boldsymbol{x}, \mathbf{Z})\, p_\theta(\mathbf{Z} \mid \mathcal{S}, \mathbf{M})\,, \quad (2) \\ & p_\theta(\boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e}, \mid \boldsymbol{x}, \mathbf{Z}) \\ & \quad = p_\theta(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{e})\, p_\theta(\boldsymbol{a} \mid \boldsymbol{x})\, p_\theta(\boldsymbol{e} \mid \boldsymbol{x}, \mathbf{Z}) \quad (3) \end{aligned}$$

where $\mathbf{Z} \in \mathbb{R}^{|\mathcal{E}|\times D}$ denotes prototypes of all entity types in the task, which are obtained from the support set $\mathcal{S}$ and a memory module $\mathbf{M}$ (detailed below), i.e., $p_\theta(\mathbf{Z} \mid \mathcal{S}, \mathbf{M})$. These prototypes are

[2]As the model applies to both source and target domain, we drop subscripts s and t in this section for clarity.

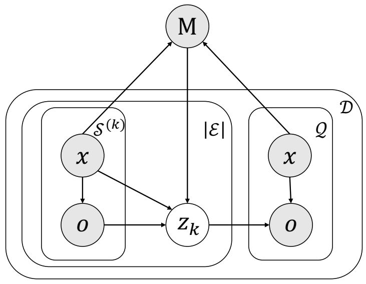

Figure 3: Probabilistic graphical model of our MANNER, where $\mathbf{M}$ is the external memory module, $\boldsymbol{o} = \{\boldsymbol{e}, \boldsymbol{a}, \boldsymbol{y}\}$ denotes a set of labels, $\mathcal{S}^{(k)}$ denotes samples of entity type k in the support set, $\mathcal{Q}$ is the query set, and $|\mathcal{E}|$ is the number of entity types in the task.

then used to predict the entity types of tokens, i.e., $p_\theta(\boldsymbol{e} \mid \boldsymbol{x}, \mathbf{Z})$, which is further combined with the predicted position tags, i.e., $p_\theta(\boldsymbol{a} \mid \boldsymbol{x})$, to obtain the distribution over labels, i.e., $p_\theta(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{e})$. Figure 3 illustrates the probabilistic graphical model of MANNER. In what follows, we will introduce the details of the joint probability distribution.

Memory-Augmented Prototypes $p_\theta(\mathbf{Z} \mid \mathcal{S}, \mathrm{M})$: Since there are only a few labeled data in the support set, the prototypes that are obtained from the support set may not be accurate and representative. Therefore, we leverage an external memory module to store entity type information from the source domain to augment the support sets of few-shot tasks in the target domain. Specifically, we denote the memory as $\mathbf{M}$, which contains key-value pairs that correspond to different entity types in the source domain. The keys are different entity types and the values are representations of tokens that belong to the corresponding entity types. For efficient retrieval from memory, we limit the number of token representations of each entity type to be m, which is referred to as the memory size.

In order to adapt the retrieved information (i.e., token representations) from the source domain to the target domain, we leverage optimal transport to retrieve and process information from the memory. Specifically, for an entity type k in a few-shot task τ, we first retrieve its most similar entity types k^* in the memory based on the OT distance:

$$\begin{aligned} k^* &= \arg\min_{k' \in \mathcal{E}} \mathcal{W}(\mathbf{M}_{k'}, \mathbf{H}_k) \\ &= \arg\min_{k' \in \mathcal{E}} \min_{\mathbf{T} \in \Sigma(\frac{1}{m}\mathbf{1}_m, \frac{1}{n_k}\mathbf{1}_{n_k})} \langle \mathbf{C}, \mathbf{T} \rangle , \end{aligned} \tag{4}$$

where $\mathbf{H}_k = f_\theta(\mathcal{S}^{(k)})$, $\mathcal{S}^{(k)} = \{x_{k,1}, \ldots, x_{k,n_k}\}$, is the contextualized representations of tokens that belong to entity type k in the support set, f_θ is a token encoder such as BERT (Devlin et al., 2019), $\mathbf{M}_{k'}$ denotes the token representations of entity type k' stored in the memory, and $\mathbf{C}$ is a cost matrix with each element computed as: $c(\mathbf{M}_{k',i}, \mathbf{H}_{k,j}) = ||\mathbf{M}_{k',i} - \mathbf{H}_{k,j}||_2^2$. We denote the retrieved information for entity type k as $\mathbf{M}_{k^*}$, and the optimal transport plan between token representations $\mathbf{H}_k$ and $\mathbf{M}_{k^*}$ as $\mathbf{T}_k^*$, which is obtained through the Sinkhorn algorithm (Cuturi, 2013) in this paper.

We next follow previous works (Courty et al., 2017a,b) to adapt the retrieved information from the source domain, i.e., $\mathbf{M}_{k_*}$, to the domain of task τ through the following barycentric mapping:

$$\hat{\boldsymbol{h}}_i = \arg\min_{\boldsymbol{h} \in \mathbb{R}^D} \sum_j \mathbf{T}_k^*(i,j) \cdot c(\boldsymbol{h}, \mathbf{H}_{k,j}), \tag{5}$$

for all $i = 1, \ldots, m$, where $\hat{\boldsymbol{h}}_i$ denotes the projected representation of the i-th item in $\mathbf{M}_{k^*}$, and $\mathbf{T}_k^*(i,j)$ represents an element of the optimal transport plan $\mathbf{T}_k^*$. It has been shown that when the cost function is squared Euclidean norm, the solution to above barycenter mapping corresponds to a weighted average of $\mathbf{H}_k$ (Courty et al., 2017b), which is given by:

$$\hat{\mathbf{H}}_k = \mathrm{diag}(\mathbf{T}_k^* \mathbf{1}_{n_k})^{-1} \, \mathbf{T}_k^* \mathbf{H}_k, \tag{6}$$

where $\mathrm{diag}(\cdot)$ is a diagonal matrix.

After obtaining the adapted memory, i.e., $\hat{\mathbf{H}}_k$, we combine it with token representations in the support set to get the prototype distributions:

$$\begin{aligned} p_\theta(\mathbf{Z} \mid \mathcal{S}, \mathbf{M}) &= \prod_{k \in \mathcal{E}} p_\theta\left(\boldsymbol{z}_k \mid \mathcal{S}^{(k)}, \mathbf{M}_{k_*}\right) \\ &= \prod_{k \in \mathcal{E}} \mathcal{N}\left(\boldsymbol{z}_k \mid g_\theta(\hat{\mathbf{H}}_k, \mathbf{H}_k), \sigma_1^2 \mathbf{I}\right), \end{aligned} \tag{7}$$

where we model the distributions over prototypes as Gaussian distributions, whose mean is obtained through a mean function g_θ and the covariance is given by $\sigma_1^2\mathbf{I}$. We define the mean function as:

$$\begin{aligned} g_\theta(\hat{\mathbf{H}}_k, \mathbf{H}_k) =& \gamma \cdot \mathrm{Neural}([\hat{\boldsymbol{r}}_k, \boldsymbol{r}_k]) \\ &+ (1-\gamma) \cdot \boldsymbol{r}_k , \end{aligned} \tag{8}$$

where $\hat{\boldsymbol{r}}_k = \frac{1}{m}\sum_i \hat{\mathbf{H}}_{k,i}$ and $\boldsymbol{r}_k = \frac{1}{n_k}\sum_j \mathbf{H}_{k,j}$ are the mean of token representations in the memory and the support set respectively, $[\cdot,\cdot]$ is the concatenation operation, and $\mathrm{Neural}(\cdot)$ is a feed-forward neural network with Relu activation function. We introduce a hyperparameter γ to interpolate the information from the memory and the support set.

Span Detection $p_\theta(\boldsymbol{a} \mid \boldsymbol{x})$: We formulate span detection as a sequence labeling task, i.e., predicting the position tags of tokens. Note that we use an encoder function f_θ, e.g., BERT, to obtain the contextualized representations of tokens when computing prototypes. Based on these token representations, we use a linear classifier to compute the probability distributions of position tags. Specifically, for a sentence $\boldsymbol{x}$, its distribution of position tags is:

$$p_\theta(\boldsymbol{a} \mid \boldsymbol{x}) = \text{Softmax}(f_\theta(\boldsymbol{x})\mathbf{W} + \boldsymbol{b}), \quad (9)$$

where $\mathbf{W} \in \mathbb{R}^{D\times 5}$ and $\boldsymbol{b}$ are model parameters.

Entity Typing $p_\theta(\boldsymbol{e} \mid \boldsymbol{x}, \mathbf{Z})$: We follow the principle of prototypical networks (Snell et al., 2017) to compute the probability distributions of entity types. Specifically, for a sentence $\boldsymbol{x}$, we compute its distribution of entity types as:

$$p_\theta(\boldsymbol{e} \mid \boldsymbol{x}, \mathbf{Z}) = \text{Softmax}(f_\theta(\boldsymbol{x})\mathbf{Z}^\top), \quad (10)$$

where $\mathbf{Z}$ is the sampled prototypes from prototype distribution defined in Equation (7).

Label Prediction $p_\theta(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{e})$: Finally, we combine the results of span detection and entity typing to get the label distributions of a sentence $\boldsymbol{x}$, i.e., $p_\theta(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{e}) = \prod_{i=1}^n p_\theta(\boldsymbol{y}_i \mid \boldsymbol{a}_i, \boldsymbol{e}_i)$, where the predicted label distribution of each token is:

$$p_\theta(\boldsymbol{y}_i \mid \boldsymbol{a}_i, \boldsymbol{e}_i) \propto p_\theta(\boldsymbol{a}_i \mid \boldsymbol{x}) \cdot p_\theta(\boldsymbol{e}_i \mid \boldsymbol{x}, \mathbf{Z}). \quad (11)$$

For example, for a token x_i with label "B-Person", the probability of the token being classified as "B-Person" is proportional to the product of $p_\theta(\boldsymbol{a}_i = \text{B} \mid \boldsymbol{x})$ and $p_\theta(\boldsymbol{e}_i = \text{Person} \mid \boldsymbol{x}, \mathbf{Z})$.

3.2 Learning in Source Domain

We next introduce how to learn our model on source domain. The goal of learning is to maximize the likelihood of observations in source domain[3]:

$$p_\theta(\mathcal{D}_s \mid \mathbf{M}) = \int p_\theta(\mathcal{D}_s \mid \mathbf{Z}_s) p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathbf{M}) \mathrm{d}\mathbf{Z}_s$$

where $\mathcal{D}_s = \{\mathcal{S}_s, \mathcal{Q}_s\} = \{(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e})\}$ represents the set of observed variables in both support set and query set. The learning of such probabilistic models requires inferring the posterior distributions of stochastic variables, i.e., inferring the distribution of prototypes after seeing both the support set and the query set in our case. However, exact inference is intractable due to the non-Gaussian likelihood function in our model. Therefore, we resort to variational inference to approximate the posteriors and learn the model (Kingma and Welling, 2014).

Specifically, we approximate posteriors of prototypes with the following variational distributions:

$$\begin{aligned} q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s) &= \prod_{k\in\mathcal{E}} q_\theta\left(\boldsymbol{z}_{s,k} \mid \mathcal{S}_s^{(k)}, \mathcal{Q}_s^{(k)}\right) \\ &= \prod_{k\in\mathcal{E}} \mathcal{N}\left(\boldsymbol{z}_{s,k} \mid g_\theta\left(f_\theta(\mathcal{Q}_s^{(k)}), f_\theta(\mathcal{S}_s^{(k)})\right), \sigma_2^2\mathbf{I}\right), \end{aligned}$$

where $\mathcal{Q}_s^{(k)}$ represents tokens that belong to entity type k in the query set, and $\sigma_2^2\mathbf{I}$ is the covariance. For parameter efficiency, we use the same inference network g_θ in Equation (7) to infer the posteriors of prototypes. However, the variational distributions are different from Equation (7), which can be regarded as prior distributions, in that the variational distributions are obtained based on both support and query sets while the prior distributions are obtained based on support set and the memory.

With the above variational distributions, we can derive the Evidence Lower BOund (ELBO) of log-likelihood function of our model as:

$$\begin{aligned} &\mathcal{L}_{\text{ELBO}} = \\ &- \mathrm{D}_{KL}\left[q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s) \,||\, p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathbf{M})\right] \\ &+ \sum_{\boldsymbol{o}\in\mathcal{D}_s} \mathbb{E}_{q_\theta(\mathbf{Z}_s|\mathcal{S}_s,\mathcal{Q}_s)}\left[\log p_\theta(\boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e}, \mid \boldsymbol{x}, \mathbf{Z}_s)\right] \\ &+ \text{const.}, \end{aligned} \quad (12)$$

where $\boldsymbol{o} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e})$ and $\mathrm{D}_{KL}[\cdot||\cdot]$ is the Kullback–Leibler (KL) divergence. Please refer to Appendix B for the detailed derivation of ELBO. We learn the model parameters θ by maximizing the ELBO defined in Equation (12), where KL divergence has a closed-form solution while the expectation term is approximated with Monte Carlo method by sampling from the variational distributions. At each training iteration, in order to remain the accuracy of the memory, we use the token representations, which is obtained through f_θ, in both the support and query sets to update the information in the memory. Specifically, for each entity type in a task, we randomly select m representations of tokens that belong to this entity type to update token representations stored in the memory. We leave the exploration of selecting representative token representations as our further research.

[3]We use a subscript s and t to denote variables in the source domain and the target domain respectively.

Models	1-shot				5-shot			
	Ontonotes	WNUT	GUM	CoNLL	Ontonotes	WNUT	GUM	CoNLL
TransferBERT†	3.46 ± 0.54	2.71 ± 0.72	0.57 ± 0.32	4.75 ± 1.42	35.49 ± 7.60	11.08 ± 0.57	3.62 ± 0.57	15.36 ± 2.81
SimBERT†	13.99 ± 0.00	5.18 ± 0.00	6.91 ± 0.00	19.22 ± 0.00	21.12 ± 0.00	8.20 ± 0.00	10.63 ± 0.00	32.01 ± 0.00
Matching Network†	15.06 ± 1.61	17.23 ± 2.75	4.73 ± 0.16	19.50 ± 0.35	8.08 ± 0.47	6.61 ± 1.75	5.58 ± 0.23	19.85 ± 0.74
ProtoBERT†	6.67 ± 0.46	10.68 ± 1.40	3.89 ± 0.24	32.49 ± 2.01	13.59 ± 1.61	17.26 ± 2.65	9.54 ± 0.44	50.06 ± 1.57
CONTaiNER	32.96 ± 0.91	16.45 ± 0.92	10.81 ± 0.45	34.09 ± 0.94	48.62 ± 0.64	27.50 ± 0.58	24.31 ± 0.66	58.63 ± 1.56
L-TapNet+CDT†	15.17 ± 1.25	20.80 ± 1.06	12.04 ± 0.65	44.30 ± 3.15	20.95 ± 2.81	23.30 ± 2.80	11.65 ± 2.34	45.35 ± 2.67
DecomposedMetaNER‡	34.13 ± 0.92	25.14 ± 0.24	17.54 ± 0.98	46.09 ± 0.44	45.55 ± 0.90	31.02 ± 0.91	31.36 ± 0.91	58.18 ± 0.87
MANNER	**43.61 ± 0.48**	**28.54 ± 0.69**	**23.17 ± 0.20**	**49.06 ± 1.37**	**58.37 ± 0.62**	**35.86 ± 1.42**	**40.86 ± 0.96**	**64.84 ± 0.51**

Table 1: Overall performance (F1 scores %) of MANNER and baselines on Cross-Dataset, where † and ‡ denote the results reported in (Hou et al., 2020) and (Ma et al., 2022), respectively.

Models	1-shot				5-shot			
	Address	Medical	Weibo	Cluener	Address	Medical	Weibo	Cluener
NNShot	35.87 ± 1.21	11.33 ± 0.87	27.22 ± 1.78	23.41 ± 0.91	44.45 ± 1.25	16.65 ± 0.59	34.80 ± 0.43	27.49 ± 0.89
StructShot	43.83 ± 0.93	14.45 ± 0.78	26.73 ± 1.81	26.20 ± 0.41	51.14 ± 1.38	23.43 ± 0.86	31.56 ± 2.22	31.67 ± 0.87
ProtoBERT	47.54 ± 1.73	18.12 ± 0.86	23.68 ± 0.79	19.01 ± 1.61	65.37 ± 0.28	38.60 ± 0.49	42.41 ± 1.78	37.20 ± 1.24
CONTaiNER	53.18 ± 1.95	18.11 ± 0.98	33.92 ± 2.18	23.83 ± 1.89	68.00 ± 1.17	34.00 ± 1.06	47.43 ± 1.49	39.59 ± 0.47
DecomposedMetaNER	55.38 ± 0.54	26.64 ± 0.76	34.92 ± 2.74	38.08 ± 1.35	60.83 ± 0.50	38.95 ± 4.74	41.02 ± 2.32	47.57 ± 0.95
MANNER	**68.47 ± 0.87**	**31.43 ± 0.60**	**42.64 ± 0.63**	**39.07 ± 1.01**	**78.58 ± 0.31**	**44.61 ± 0.47**	**53.36 ± 0.62**	**54.90 ± 0.53**

Table 2: Overall performance (F1 scores %) of MANNER and baselines on Chinese Cross-Dataset.

3.3 Adaption in Target Domain

Finally, we introduce how to adapt our model to the target domain. Similar to previous work (Ma et al., 2022), we finetune the model with few-shot examples in the target domain. However, since we do not have access to the query set in the target domain, we can not use the ELBO in Equation (12) to finetune our model. Therefore, we propose to adapt our model to the target domain by maximizing the likelihood function of the support set in the target domain. Formally, the objective function is:

$$\min_{\theta} \mathbf{E}_{p_\theta(\mathbf{Z}_t|\mathcal{S}_t,\mathbf{M})} \left[\log p_\theta(\mathcal{S}_t \mid \mathbf{Z}_t)\right] . \tag{13}$$

After adapting our model to the target domain, we make prediction for a sentence $\boldsymbol{x}$ in the query set with $p_\theta(\boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e} \mid \boldsymbol{x}, \tilde{\mathbf{Z}}_t)$, where $\tilde{\mathbf{Z}}_t$ represents the mean of prototype distributions. The pseudo code of the training and adaption process of our model is provided in Appendix C.

4 Experiments

4.1 Experimental Setups

Datasets. We conduct experiments on two groups of datasets: (1) **Cross-Dataset** (Hou et al., 2020): It is an English cross domain few-shot NER dataset constructed from four datasets: Ontonotes (Pradhan et al., 2013), WNUT-2017 (Derczynski et al., 2017), GUM (Zeldes, 2017), CoNLL-2003 (Sang and De Meulder, 2002). For fair comparison, we use the same sampled episodes and dataset splits as in (Hou et al., 2020), where two of the four datasets are used for training, one for validation and the other for test. For example, to evaluate the performance on Ontonotes, we take WNUT and GUM as the training sets and CoNLL as the validation set. (2) **Chinese Cross-Dataset**: We also construct a Chinese cross domain few-shot NER dataset using five publicly available datasets: CCKS[4], Address[5], Medical (Zhang et al., 2022), Weibo (Peng and Dredze, 2015) and Cluener (Xu et al., 2020). Following the settings in (Yang and Katiyar, 2020), we first train few-shot models on the training set of the CCKS dataset and then evaluate their performance on the other four datasets. For each test dataset, we sample K-shot data from their training set as the support set and use the whole test set as the query set to construct a test episode. We repeat the sampling process for five times and obtain five test episodes for each dataset. We compare the average performance on the five test episodes. More details about our datasets are provided in Appendix D.1.

[4] https://www.biendata.xyz/competition/ccks_2020_el/

[5] https://tianchi.aliyun.com/dataset/109339

Models	1-shot				5-shot			
	Ontonotes	WNUT	GUM	CONLL	Ontonotes	WNUT	GUM	CONLL
MANNER	**43.61** ± **0.48**	**28.54** ± **0.69**	**23.17** ± **0.20**	**49.06** ± **1.37**	**58.37** ± **0.62**	**35.86** ± **1.42**	**40.86** ± **0.96**	**64.84** ± **0.51**
w/o Memory	41.49 ± 1.02	26.15 ± 0.43	20.85 ± 0.50	48.58 ± 0.88	54.40 ± 1.13	33.84 ± 0.73	36.10 ± 1.01	63.58 ± 0.94
w/o OT	38.17 ± 1.13	25.27 ± 0.66	20.06 ± 0.77	47.78 ± 1.41	52.33 ± 0.91	33.13 ± 0.41	35.71 ± 1.02	62.21 ± 1.65
Deterministic	42.62 ± 1.31	28.21 ± 0.63	22.52 ± 0.32	48.50 ± 0.77	57.59 ± 1.03	35.79 ± 1.63	39.56 ± 0.40	64.51 ± 0.79

Table 3: Ablation study. F1 scores (%) on Cross-Dataset are reported.

Baselines. On Cross-Dataset, we take the following models as our baselines: DecomposedMetaNER (Ma et al., 2022), CONTaiNER (Das et al., 2022), L-TapNet+CDT (Hou et al., 2020) and those baselines used in (Hou et al., 2020), such as TransferBERT, SimBERT, Matching Network, and ProtoBERT (Fritzler et al., 2019). On Chinese Cross-Dataset, we compare against some strong few-shot NER models such as DecomposedMetaNER, CONTaiNER, ProtoBERT, NNShot and StructShot (Yang and Katiyar, 2020).

Evaluation. We employ the episode evaluation as in (Hou et al., 2020) where we calculate micro F1 score within each test episode and then average over all test episodes. We repeat each experiment for 5 times with different seeds and report average micro F1 scores with their standard deviations.

Settings. Following previous works (Hou et al., 2020; Ma et al., 2022), we use bert-base-uncased (Devlin et al., 2019) to obtain contextualized token representations for Cross-Dataset. Similarly, bert-base-chinese is utilized for Chinese Cross-Dataset. We instantiate the mean function of the inference network, i.e., g_θ, with a two-layered feed-forward neural network with the ReLU activation function and set the number of hidden units as 128. Moreover, to effectively optimize the ELBO, we follow previous works (Osawa et al., 2019; Zhang et al., 2021) to introduce an additional hyperparameter λ to down-weight the KL-divergence in the ELBO. Throughout the experiments, we set λ as $1e^{-3}$, and sample 5 times from the variational distributions to approximate the expectation term in the ELBO.

We set the maximum sequence length of the BERT models as 128 and the hyperparameter γ as 0.5. To optimize the parameters, we use AdamW (Loshchilov and Hutter, 2019) with a 1% linearly scheduled warmup as the optimizer and freeze the embedding layers of bert during optimization. Moreover, we perform grid search to select hyperparameters. Additional details about hyperparameter settings are in Appendix D.2.

4.2 Results and Analysis

Overall Performance. The performance of MANNER and baselines on Cross-Dataset and Chinese Cross-Dataset are reported in Table 1 and Table 2, respectively. The results show that MANNER performs better than all the baselines on F1 score in all settings and surpass the second best models by a large margin in most cases. Particularly, on Cross-Dataset, MANNER achieves an average performance improvement of 5.37% and 7.58% in 1-shot and 5-shot settings respectively compared with the best baselines. Similarly, on Chinese Cross-Dataset, the average performance improvement of MANNER is 6.65% (1-shot) and 7.38% (5-shot). The experimental results well demonstrate the effectiveness of MANNER in handling both English and Chinese few-shot NER tasks. Moreover, compared with DecomposedMetaNER, a strong baseline, MANNER achieves performance improvement up to 9.48% (Ontonotes 1-shot) on Cross-Dataset and 13.09% (Address 1-shot) on Chinese Cross-Dataset, which suggests that MANNER can achieve superior performance even with very few labeled data (e.g., 1-shot).

Ablation Studies. We conduct ablation studies to investigate the effect of different components, i.e., memory module, optimal transport and probabilistic framework, in our model. We introduce three variants of MANNER for the ablation study: (1) MANNER *w/o Memory*, where the memory module is removed and the prototype distributions are inferred from the support set only. Note that this variant does not use OT either as it is unnecessary to adapt the retrieved information from the memory to the target domain. (2) MANNER *w/o OT*, where we remove the OT module and use cosine similarity to retrieve the most similar entity type from memory. The retrieved information is directly used to infer prototype distributions without any processing. (3) *Deterministic*, where we remove probabilistic framework and model prototypes as deterministic vectors.

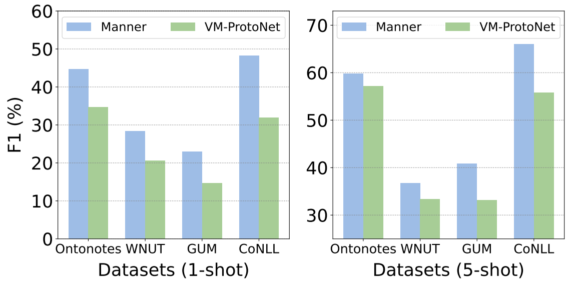

Figure 4: Overall performance (F1 scores %) of MANNER and VM-ProtoNet on Cross-Dataset.

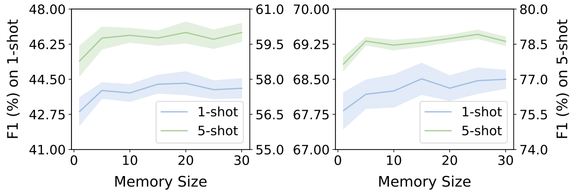

Figure 5: Effect of memory size on Ontonotes (left) and Address (right) datasets.

The results of ablation studies are reported in Table 3. It is shown that MANNER consistently outperforms MANNER *w/o Memory* in all settings, which indicates the effectiveness of our memory module in improving the performance. This is because with appropriate processing, e.g., OT in this paper, the information stored in memory can provide background knowledge for quickly and accurately learning new classes from a few examples and therefore brings performance improvement.

Table 3 also shows that MANNER outperforms MANNER *w/o OT*, which demonstrates the effectiveness of OT in MANNER. Moreover, we found that MANNER *w/o Memory* outperforms MANNER *w/o OT* which directly use the information from the memory without any processing. This is because in our cross domain few-shot setting, the information from the memory (source domain) is different from that of the test tasks (target domain), i.e., the entity types of two domain are disjoint, and therefore directly utilizing the information from the memory may introduce noises to the test tasks, leading to the performance degradation. The results demonstrate the necessity of leveraging OT to adapt information from the memory to current task to achieve satisfactory performance.

Table 3 shows that MANNER achieves better or comparative results compared with its deterministic counterpart, especially on the 1-shot settings. The improvement can be explained by the fact that MANNER introduces small noises to the prototypes by sampling from the prototype distributions to prevent the model from overfitting the few-shot data during the finetuning stage.

Effect of Decomposed Framework. It is worth noting that MANNER decomposes label prediction of NER into two-subtasks: span detection and entity typing. To investigate the effect of the decomposed framework, we additionally introduce a variant of MANNER: *VM-ProtoNet* where we only remove the decomposed framework and learn prototypes for each entity label, which is similar to ProtoBERT. Note that we also use memory and OT to augment few-shot NER tasks in VM-ProtoNet. We compare the performance of MANNER and VM-ProtoNet on Cross-Dataset, which is presented in Figure 4. The results show that MANNER surpasses VM-ProtoNet in all settings and achieves noticeable performance improvement in most cases, especially on the CoNLL dataset. The success of the decomposed framework maybe because it avoids learning prototype for non-entities (i.e., "*O*" class) which is noisy and meaningless. Overall, experimental results demonstrate the effectiveness of the decomposed framework in few-shot NER tasks, which is consistent with the results of previous works (Wang et al., 2021b; Ma et al., 2022).

Effect of Memory Size. In MANNER, we limit the number of token representations of each entity type stored in the memory. We further conduct experiments to understand the effect of the memory size on the performance of MANNER. Specifically, we vary the memory size from 1 to 30 and report the performance of MANNER on Ontonotes and Address. The results in Figure 5 show that MANNER can achieve decent performance even with a low memory size and the performance converges with the increase of memory size. These findings suggest that MANNER is insensitive to memory size, which brings another benefit: it is sufficient for MANNER to achieve satisfactory performance by storing only a small number of token representations in the memory, which is efficient for both retrieving and processing information from the memory.

5 Related Work

Few-Shot NER. Recently, few-shot NER has received growing interest. Previous works mainly address few-shot NER with meta-learning methods (Fritzler et al., 2019; Wang et al., 2021c;

Huang et al., 2021; Tong et al., 2021; Ma et al., 2022). These methods build few-shot models either upon prototypical network (Snell et al., 2017), which learns prototypes for entity types (Fritzler et al., 2019; Huang et al., 2021; Tong et al., 2021; Wang et al., 2022b; Ji et al., 2022; Wang et al., 2022a), or MAML (Finn et al., 2017), which adapts the model parameters to few-shot tasks through inner-update on the support set (Li et al., 2022; Ma et al., 2022). Another line of work adopts the transfer learning paradigm, where they first learn a feature extractor on the source domain and then transfer the pretrained model to the target domain (Hou et al., 2020; Yang and Katiyar, 2020; Das et al., 2022). These methods make predictions through the nearest neighbor inference (Wiseman and Stratos, 2019). In addition, some recent works focus on the two-stage few-shot NER model (Ziyadi et al., 2020; Wang et al., 2021b; Ma et al., 2022), where they decompose the NER task into two-subtasks: span detection and entity typing. Moreover, prompt-based techniques (Cui et al., 2021; Ding et al., 2022; Chen et al., 2022) have also been proposed to address few-shot NER tasks. In contrast, MANNER stores the information from the source domain in the memory, which is then used to augment few-shot task in target domain.

Memory. Memory-augmented methods have been widely studied in the field of computer vision (Santoro et al., 2016; Bornschein et al., 2017; Ramalho and Garnelo, 2019; Munkhdalai et al., 2019; Zhen et al., 2020; Du et al., 2022). Particularly, Santoro et al. (2016) propose to augment neural network with Neural Turing Machine (Graves et al., 2014) for few-shot learning, which enables quickly encoding and retrieving new information. Ramalho and Garnelo (2019) further introduce a memory controller to select the minimum samples to be stored in the memory. Memory-augmented methods have also been successfully applied in NLP tasks, such as question answering (Das et al., 2017), text classification (Geng et al., 2020), text generation (He et al., 2020) and slot tagging (Wang et al., 2021a). Compared with above methods, our model utilizes optimal transport to adapt the retrieved memory to the target domain instead of using neural networks, which is more effective.

Optimal Transport. In domain adaption, optimal transport is a widely used method to transport data from the source domain to the target domain (Courty et al., 2017a,b; Damodaran et al., 2018; Fatras et al., 2021; Nguyen et al., 2021; Fatras et al., 2022). Theoretical guarantees have been provided in (Redko et al., 2017) to justify the use of OT in domain adaption. In Courty et al. (2017b), they propose to transport features from the source domain to the target domain through a barycentric mapping. However, they only consider transporting feature distributions. In contrast, some works propose to align the joint distributions of features and labels in source and target domains (Courty et al., 2017a; Damodaran et al., 2018).

6 Conclusion

This paper proposes MANNER to handle the cross domain few-shot NER task. MANNER uses a memory module to store information from the source domain, which is then leveraged to augment few-shot task in the target domain. To effectively utilize the information from the memory, MANNER uses optimal transport to retrieve and process information from the memory, which enables explicitly adapting the retrieved information to the target domain and improve the performance in the cross domain few-shot setting. Experimental results on both English and Chinese few-shot NER datasets show that MANNER can achieve superior performance over existing methods.

Limitations

One limitation of our work is that MANNER only *explicitly* utilizes the memory to enhance the performance of the entity typing module in target domain. However, we argue that the memory could also *implicitly* enhances the span detection module through the shared pretrained language model with entity typing module. We leave how to explicitly leverage memory to enhance both entity typing and span detection modules as future work.

Acknowledgements

We thank all the reviewers for their valuable feedback and constructive suggestions. We would also like to thank Zeqi Tan for helping constructing the Chinese cross domain few-shot NER dataset and Xuming Hu for helpful discussions on improving the quality of the paper.

References

Kelsey Allen, Evan Shelhamer, Hanul Shin, and Joshua Tenenbaum. 2019. Infinite mixture prototypes for few-shot learning. In *International Conference on Machine Learning*, pages 232–241. PMLR.

Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna, and Gabriel Peyré. 2015. Iterative bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing*, 37(2):A1111–A1138.

Jörg Bornschein, Andriy Mnih, Daniel Zoran, and Danilo Jimenez Rezende. 2017. Variational memory addressing in generative models. In *Advances in Neural Information Processing Systems*.

Yanru Chen, Yanan Zheng, and Zhilin Yang. 2022. Prompt-based metric learning for few-shot ner. *arXiv preprint arXiv:2211.04337*.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the association for computational linguistics*, 4:357–370.

Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. 2017a. Joint distribution optimal transportation for domain adaptation. *Advances in Neural Information Processing Systems*, 30.

Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. 2017b. Optimal transport for domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(9):1853–1865.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP*. Association for Computational Linguistics.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26.

Bharath Bhushan Damodaran, Benjamin Kellenberger, Rémi Flamary, Devis Tuia, and Nicolas Courty. 2018. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 447–463.

Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017. Question answering on knowledge bases and text using universal schema and memory networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 358–365.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca J. Passonneau, and Rui Zhang. 2022. Container: Few-shot named entity recognition via contrastive learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 6338–6353.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

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, NAACL-HLT*, pages 4171–4186.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Haitao Zheng, and Maosong Sun. 2022. Openprompt: An open-source framework for prompt-learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 105–113.

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

Ying-Jun Du, Xiantong Zhen, Ling Shao, and Cees G. M. Snoek. 2022. Hierarchical variational memory for few-shot learning across domains. In *International Conference on Learning Representations, ICLR*.

Kilian Fatras, Hiroki Naganuma, and Ioannis Mitliagkas. 2022. Optimal transport meets noisy label robust loss and mixup regularization for domain adaptation. *arXiv preprint arXiv:2206.11180*.

Kilian Fatras, Thibault Séjourné, Rémi Flamary, and Nicolas Courty. 2021. Unbalanced minibatch optimal transport; applications to domain adaptation. In *International Conference on Machine Learning*, pages 3186–3197. PMLR.

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, ICML*, pages 1126–1135.

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

Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2020. Dynamic memory induction networks for few-shot text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 1087–1094.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Junxian He, Taylor Berg-Kirkpatrick, and Graham Neubig. 2020. Learning sparse prototypes for text generation. In *Advances in Neural Information Processing Systems*.

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

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2020. Few-shot named entity recognition: A comprehensive study. *arXiv preprint arXiv:2012.14978*.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2021. Few-shot named entity recognition: An empirical baseline study. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 10408–10423.

Bin Ji, Shasha Li, Shaoduo Gan, Jie Yu, Jun Ma, Huijun Liu, and Jing Yang. 2022. Few-shot named entity recognition with entity-level prototypical network enhanced by dispersedly distributed prototypes. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING*, pages 1842–1854.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR*.

Jing Li, Billy Chiu, Shanshan Feng, and Hao Wang. 2022. Few-shot named entity recognition via meta-learning. *IEEE Trans. Knowl. Data Eng.*, 34(9):4245–4256.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations, ICLR*.

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

Tsendsuren Munkhdalai, Alessandro Sordoni, Tong Wang, and Adam Trischler. 2019. Metalearned neural memory. In *Advances in Neural Information Processing Systems*, pages 13310–13321.

Tuan Nguyen, Trung Le, He Zhao, Quan Hung Tran, Truyen Nguyen, and Dinh Phung. 2021. Most: Multi-source domain adaptation via optimal transport for student-teacher learning. In *Uncertainty in Artificial Intelligence*, pages 225–235. PMLR.

Philip Ogren, Guergana Savova, and Christopher Chute. 2008. Constructing evaluation corpora for automated clinical named entity recognition. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*.

Kazuki Osawa, Siddharth Swaroop, Mohammad Emtiyaz E Khan, Anirudh Jain, Runa Eschenhagen, Richard E Turner, and Rio Yokota. 2019. Practical deep learning with bayesian principles. In *Advances in neural information processing systems*, volume 32.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 548–554.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152.

Tiago Ramalho and Marta Garnelo. 2019. Adaptive posterior learning: few-shot learning with a surprise-based memory module. In *International Conference on Learning Representations, ICLR*.

Ievgen Redko, Amaury Habrard, and Marc Sebban. 2017. Theoretical analysis of domain adaptation with optimal transport. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 737–753. Springer.

Erik F Sang and Fien De Meulder. 2002. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Adam Santoro, Sergey Bartunov, Matthew M. 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, ICML*, pages 1842–1850.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Meihan Tong, Shuai Wang, Bin Xu, Yixin Cao, Minghui Liu, Lei Hou, and Juanzi Li. 2021. Learning from miscellaneous other-class words for few-shot named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP*.

Cédric Villani. 2009. *Optimal transport: old and new*, volume 338. Springer.

Hongru Wang, Zezhong Wang, Gabriel Pui Cheong Fung, and Kam-Fai Wong. 2021a. Mcml: A novel memory-based contrastive meta-learning method for few shot slot tagging. *arXiv preprint arXiv:2108.11635*.

Jianing Wang, Chengyu Wang, Chuanqi Tan, Minghui Qiu, Songfang Huang, Jun Huang, and Ming Gao. 2022a. Spanproto: A two-stage span-based prototypical network for few-shot named entity recognition. *arXiv preprint arXiv:2210.09049*.

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

Yaqing Wang, Haoda Chu, Chao Zhang, and Jing Gao. 2021b. Learning from language description: Low-shot named entity recognition via decomposed framework. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 1618–1630. Association for Computational Linguistics.

Yaqing Wang, Subhabrata Mukherjee, Haoda Chu, Yuancheng Tu, Ming Wu, Jing Gao, and Ahmed Hassan Awadallah. 2021c. Meta self-training for few-shot neural sequence labeling. In *The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1737–1747.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL*, pages 5363–5369.

Liang Xu, Qianqian Dong, Yixuan Liao, Cong Yu, Yin Tian, Weitang Liu, Lu Li, Caiquan Liu, Xuanwei Zhang, et al. 2020. Cluener2020: fine-grained named entity recognition dataset and benchmark for chinese. *arXiv preprint arXiv:2001.04351*.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. Luke: deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 6442–6454.

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

Amir Zeldes. 2017. The gum corpus: Creating multilayer resources in the classroom. *Language Resources and Evaluation*, 51(3):581–612.

Ningyu Zhang, Mosha Chen, Zhen Bi, Xiaozhuan Liang, Lei Li, Xin Shang, Kangping Yin, Chuanqi Tan, Jian Xu, Fei Huang, Luo Si, Yuan Ni, Guotong Xie, Zhifang Sui, Baobao Chang, Hui Zong, Zheng Yuan, Linfeng Li, Jun Yan, Hongying Zan, Kunli Zhang, Buzhou Tang, and Qingcai Chen. 2022. CBLUE: A chinese biomedical language understanding evaluation benchmark. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*, pages 7888–7915. Association for Computational Linguistics.

Qiang Zhang, Jinyuan Fang, Zaiqiao Meng, Shangsong Liang, and Emine Yilmaz. 2021. Variational continual bayesian meta-learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 24556–24568.

Xiantong Zhen, Yingjun Du, Huan Xiong, Qiang Qiu, Cees Snoek, and Ling Shao. 2020. Learning to learn variational semantic memory. *Advances in Neural Information Processing Systems*, 33:9122–9134.

Morteza Ziyadi, Yuting Sun, Abhishek Goswami, Jade Huang, and Weizhu Chen. 2020. Example-based named entity recognition. *arXiv preprint arXiv:2008.10570*.

A Sinkhorn Algorithm

Sinkhorn algorithm (Cuturi, 2013) is an efficient method to approximate the optimal transport (OT) distance. It aims to solve an entropy regularized optimal transport problem, which is defined as:

$$\mathcal{W}_\epsilon(\boldsymbol{\nu}_s, \boldsymbol{\nu}_t) = \min_{\mathbf{T} \in \Sigma(\boldsymbol{\nu}_s, \boldsymbol{\nu}_t)} \langle \mathbf{C}, \mathbf{T} \rangle + \epsilon h(\mathbf{T}) \,, \quad (14)$$

where $h(\mathbf{T}) = \sum_{i,j} \mathbf{T}_{ij} \log \mathbf{T}_{ij}$ denotes the entropy regularizer and ϵ is the regularization parameter. The optimization problem in Equation (14) can be efficiently solved through the following iterative Bregman projections (Benamou et al., 2015):

$$\boldsymbol{a}^{(l+1)} = \frac{\boldsymbol{\nu}_s}{\boldsymbol{G}\boldsymbol{b}^{(l)}} \,, \boldsymbol{b}^{(l+1)} = \frac{\boldsymbol{\nu}_t}{\boldsymbol{G}^\top \boldsymbol{a}^{(l+1)}} \,, \quad (15)$$

starting from $\boldsymbol{b}^0 = \frac{1}{m}\mathbf{1}_m$, where $\boldsymbol{G} = [\boldsymbol{G}_{ij}]$ and $\boldsymbol{G}_{ij} = e^{-\mathbf{C}_{ij}/\epsilon}$. After L iterations, the optimal transport plan $\mathbf{T}^*$ is calculated as $\mathbf{T}^*_{ij} = \boldsymbol{a}_i^L \boldsymbol{G}_{ij} \boldsymbol{b}_j^L$.

B Derivation of ELBO

Note that the joint probability distribution of our model on source domain is given by:

$$p_\theta(\mathcal{D}_s, \mathbf{Z}_s \mid \mathbf{M}) = p_\theta(\mathcal{D}_s \mid \mathbf{Z}_s) p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathbf{M}),$$

where $\mathcal{D}_s = \{\mathcal{S}_s, \mathcal{Q}_s\} = \{(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e})\}$ represents the set of observed variables. We further define a

Algorithm 1: Variational Memory-Augmented Few-Shot NER (MANNER).

Input : Tasks from source domain $\mathcal{D}_s$, few-shot tasks from target domain $\mathcal{D}_t$, training steps T, finetune steps J, training learning rate η, finetune learning rate ξ.

Initialize model parameters θ and memory $\mathbf{M}$;
`/* Part I: Training on source domain. */`
for $s = 1, \ldots T$ **do**
  Sample a batch of tasks $\mathcal{D}_{\text{batch}}$ from $\mathcal{D}_s$;
  for *each task* $\mathcal{T} = (\mathcal{S}_s, \mathcal{Q}_s) \in \mathcal{D}_{\text{batch}}$ **do**
    for *each entity type* k *in* $\mathcal{T}$ **do**
      Retrieve the most similar entity type k^* from memory based on Equation (4);
      Adapt the retrieved content $\mathbf{M}_{k*}$ to current task based on Equation (6);
    Calculate the prior distributions of prototypes, i.e., $p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathbf{M})$ based on Equation (7);
    Calculate the variational distributions of prototypes, i.e., $p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s)$;
    Sample prototypes $\mathbf{Z}_s$ from $p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s)$;
    Calculate the ELBO based on Equation (12);
    Accumulate gradients of model parameters θ which are obtained by maximizing the ELBO;
    Update memory with token representations in both support and query sets.
  Update model parameters θ with learning rate η;
`/* Part II: Finetuning on target domain. */`
for *each task* $\mathcal{T} = (\mathcal{S}_t, \mathcal{Q}_t) \in \mathcal{D}_t$ **do**
  Initialize model parameters $\theta' = \theta$;
  for s = *1, . . . , J* **do**
    for *each entity type* k *in* $\mathcal{T}$ **do**
      Retrieve the most similar entity type k^* from memory based on Equation (4);
      Adapt the retrieved content $\mathbf{M}_{k*}$ to current task based on Equation (6);
    Calculate the prior distributions of prototypes, i.e., $p_\theta(\mathbf{Z}_t \mid \mathcal{S}_t, \mathbf{M})$ based on Equation (7);
    Sample prototypes $\mathbf{Z}_t$ from $p_\theta(\mathbf{Z}_t \mid \mathcal{S}_t, \mathbf{M})$;
    Calculate the objective function based on Equation (13);
    Update model parameters θ' with learning rate ξ;

Hyperparameters	1-shot				5-shot			
	Ontonotes	WNUT	GUM	CoNLL	Ontonotes	WNUT	GUM	CoNLL
batch size	16	16	1	1	16	1	1	1
training learning rate	1e-4	3e-5	1e-4	3e-5	1e-4	3e-5	1e-4	3e-5
finetune learning rate	1e-4	3e-5	1e-4	3e-5	1e-4	3e-5	1e-4	3e-5
training steps	500	500	1000	1000	500	1000	1000	1000
finetune steps	50	50	50	50	50	50	50	50

Table 4: Optimal hyperparameter settings on Cross-Dataset.

variational distribution $q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s)$ to approximate the posteriors of latent variables. Therefore, we can derive the ELBO as follows:

$$
\begin{aligned}
&\log p_\theta(\mathcal{D}_s \mid \mathbf{M}) \\
&= \log \int p_\theta(\mathcal{D}_s, \mathbf{Z}_s \mid \mathbf{M}) \frac{q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s)}{q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s)} \mathrm{d}\mathbf{Z}_s \\
&\geq \int q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s) \log \frac{p_\theta(\mathcal{D}_s, \mathbf{Z}_s \mid \mathbf{M})}{q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s)} \mathrm{d}\mathbf{Z}_s \\
&= \mathbb{E}_{q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s)} \left[\log p_\theta\left(\mathcal{D}_s \mid \mathbf{Z}_s\right)\right] \\
&\quad - \mathrm{D}_{KL}\left[q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s) \,||\, p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathbf{M})\right] \\
&\triangleq \mathcal{L}_{\text{ELBO}} ,
\end{aligned}
\tag{16}
$$

where the inequality is obtained via the Jensen's inequality. Since the likelihood function of our model is given by:

$$
p_\theta(\mathcal{D}_s \mid \mathbf{Z}_s) = \prod_{\boldsymbol{o} \in \mathcal{D}_s} p_\theta(\boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e}, \mid \boldsymbol{x}, \mathbf{Z}_s)\, p(\boldsymbol{x}).
$$

We can put this function into Equation (16) and further derive the ELBO as:

$$\begin{aligned}
&\mathcal{L}_{\text{ELBO}} = \\
&-\mathrm{D}_{KL}\left[q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s) \,||\, p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathbf{M})\right] \\
&+ \sum_{\boldsymbol{o}\in\mathcal{D}_s} \mathbb{E}_{q_\theta(\mathbf{Z}_s|\mathcal{S}_s,\mathcal{Q}_s)}\left[\log p_\theta(\boldsymbol{y}, \boldsymbol{a}, \boldsymbol{e}, \mid \boldsymbol{x}, \mathbf{Z}_s)\right] \\
&+ \text{const.}\,, \qquad (17)
\end{aligned}$$

where $\text{const.} = \sum_{\boldsymbol{o}\in\mathcal{D}_s} \log p(\boldsymbol{x})$ is a constant. The KL divergence in Equation (17) has a closed form solution, which is given by:

$$\begin{aligned}
&-\mathrm{D}_{KL}\left[q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s) \,||\, p_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathbf{M})\right] \\
&= -\frac{1}{2}\sum_{k\in\mathcal{E}} \frac{1}{\sigma_1^2}(\boldsymbol{\mu}_k - \boldsymbol{m}_k)^\top(\boldsymbol{\mu}_k - \boldsymbol{m}_k) \\
&\quad -\frac{|\mathcal{E}|}{2}\left((s-1)D - \log s\right)\,, \qquad (18)
\end{aligned}$$

where $\boldsymbol{\mu}_k$, $\boldsymbol{m}_k$ denote the mean of the prior and variational distributions of prototypes, respectively, $s = \sigma_2^2/\sigma_1^2$, and D is the dimension of prototypes. Moreover, we sample $\mathbf{Z}_s$ from variational distributions $q_\theta(\mathbf{Z}_s \mid \mathcal{S}_s, \mathcal{Q}_s)$ to approximate the expectation term in ELBO.

C Pseudo Code

The training and inference process of our model is provided in Algorithm 1. In the training process, we randomly sample a small batch of tasks $\mathcal{D}_{\text{batch}}$, accumulate the gradients of their objective function and then update the model parameters with the AdamW optimizer. In the inference process, for each task, we first initialize the model parameters θ' with the learned model parameters in the source domain, i.e., $\theta' = \theta$, and then finetune the parameters by maximizing the likelihood function in the support set for J steps.

D Experimental Details

D.1 Datasets

Table 5 shows the statistics of original datasets used to construct the experimental datasets and statistics of the constructed few-shot datasets.

Cross-Dataset is an English cross domain few-shot NER dataset, which is constructed to evaluate the performance of meta-learning based few-shot models. We use the public episodes[6] constructed by (Hou et al., 2020) in our experiments, where the training, validation and test episodes for each dataset are provided.

[6] https://github.com/AtmaHou/FewShotTagging.

	Dataset	# Sent	# label	Avg. $\|\mathcal{S}\|$	
				1-shot	5-shot
Cross-Dataset	Ontonotes	159,615	19	14.38	62.28
	WNUT	5,657	7	5.48	28.66
	GUM	3,493	12	6.50	27.81
	CoNLL	20,679	5	3.38	15.58
Chinese Cross-Dataset	CCKS	90,000	23	-	-
	Address	8,856	18	9.8	43.2
	Medical	15,000	6	4.6	17.2
	Weibo	1,890	4	3.4	11.6
	Cluener	10,748	10	9.0	30.4

Table 5: Statistics of datasets, where Avg. $|\mathcal{S}|$ denotes the average size of support in each dataset.

We additionally construct a Chinese cross domain few-shot NER dataset from five public Chinese NER datasets: CCKS, Address, Medical, Weibo and Cluener. We follow the experimental settings in (Yang and Katiyar, 2020), where they first train few-shot models on a source domain and then transfer the model to target domain with few-shot data. We take CCKS as source domain and the other four datasets as target domains. To construct few-shot data in target domains, we use the sampling method in (Ding et al., 2021) to sample K-shot data from the training set of each test dataset as support set and use the original test data as query set. We repeat the sampling process for five times to obtain accurate experimental results.

D.2 Hyperparameter Settings

We set the memory size m, i.e., number of token representations of each entity type in the memory, as 15. The hyperparameter γ and the standard deviation of prototype distributions is set to be 0.5 and e^{-10} respectively. The dropout rate and weight decay coefficient is set to be 0.1 and $1e-3$, respectively. On Cross-Dataset, we choose batch size from $\{1, 16, 32\}$, learning rate from {1e-5, 3e-5, 1e-4}, training steps from $\{300, 500, 1000\}$, and finetune steps from $\{30, 50\}$. We perform grid search to choose hyperparameters that have the best performance on the validation set. The optimal hyperparameter settings on Corss-Dataset are provided in Table 4. On Chinese Cross-Dataset, we set the batch size as 1, training steps as 1000, finetune steps as 50, training learning rate as 3e-5 and finetune learning rate as 3e-5 for all settings. During training, we evaluate our model on the validation set every 100 steps and select the checkpoint with best f1 scores on the validation set as the final model.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Please see the Limitations Section.

☒ A2. Did you discuss any potential risks of your work?
There is not obvious risk regarding our work.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Please see the Abstract and Section 1.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

Please see Section 4.

☒ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Our model is built based on the existing pretrained model, and the computational budget might vary depending on the used backbone model.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Please see Section 4.1 and Appendix D.2.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Please see Section 4.2.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Please see Appendix D.2.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.