Few-Shot Named Entity Recognition: An Empirical Baseline Study

Jiaxin Huang[1 *], **Chunyuan Li**[2 †], **Krishan Subudhi**[2], **Damien Jose**[2],
Shobana Balakrishnan[2], **Weizhu Chen**[2], **Baolin Peng**[2], **Jianfeng Gao**[2], **Jiawei Han**[1]
[1]University of Illinois Urbana-Champaign [2]Microsoft
{jiaxinh3, hanj}@illinois.edu
{chunyl,krkusuk,dajose,shobanab,wzchen,bapeng,jfgao}@microsoft.com
Website: https://few-shot-ner-benchmark.github.io/index.html

Abstract

This paper presents an empirical study to efficiently build named entity recognition (NER) systems when a small amount of in-domain labeled data is available. Based upon recent Transformer-based self-supervised pre-trained language models (PLMs), we investigate three orthogonal schemes to improve model generalization ability in few-shot settings: (1) meta-learning to construct prototypes for different entity types, (2) task-specific supervised pre-training on noisy web data to extract entity-related representations and (3) self-training to leverage unlabeled in-domain data. On 10 public NER datasets, we perform extensive empirical comparisons over the proposed schemes and their combinations with various proportions of labeled data, our experiments show that (*i*) in the few-shot learning setting, the proposed NER schemes significantly improve or outperform the commonly used baseline, a PLM-based linear classifier fine-tuned using domain labels. (*ii*) We create new state-of-the-art results on both few-shot and training-free settings compared with existing methods.

1 Introduction

Named Entity Recognition (NER) involves processing unstructured text, locating and classifying named entities (certain occurrences of words or expressions) into particular categories of pre-defined entity types, such as persons, organizations, locations, medical codes, dates and quantities. NER serves as an important first component for tasks such as information extraction (Ritter et al., 2012), information retrieval (Guo et al., 2009), question answering (Mollá et al., 2006), task-oriented dialogues (Peng et al., 2020a; Gao et al., 2019) and other language understanding applications (Nadeau and Sekine, 2007; Shaalan, 2014). Deep learning

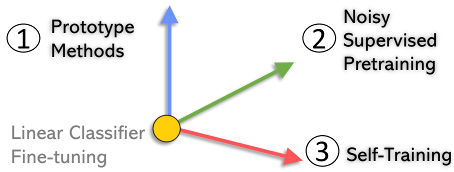


Figure 1: An overview of methods studied in our paper. Linear classifier fine-tuning is a default baseline that updates an NER model from pre-trained Roberta/BERT. We study three orthogonal strategies to improve NER models in the limited labeled data settings.

has shown remarkable success in NER in recent years, especially with self-supervised pre-trained language models (PLMs) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019c). State-of-the-art (SoTA) NER models are often initialized with PLM weights, fine-tuned with standard supervised learning. One classic approach is to add a linear classifier on top of the representations provided by PLMs, and fine-tune the entire model using a cross-entropy objective on domain-specific labels (Devlin et al., 2019). Despite its simplicity, the approach generally results in good performance on benchmarks and serves as a strong baseline in this study.

Unfortunately, even with these PLMs, building NER systems still remains a labor-intensive, time-consuming task. It requires rich domain knowledge and expert experience to annotate a large corpus of in-domain labeled tokens to teach the models to achieve a reasonable accuracy. However, this is in contrast to the real-world application scenarios, where only very small amounts of labeled data are available for new domains, such as medical (Hofer et al., 2018) domain. The cost of building NER systems at scale with rich annotations (*i.e.,* hundreds of different enterprise use-cases/domains) can be prohibitively expensive. This draws attentions to a challenging but practical research problem: few-shot NER.

To deal with the challenge of few-shot learning,

[*] Work performed during an internship at Microsoft
[†] Corresponding author

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 10408–10423
November 7–11, 2021.

we focus on improving the generalization ability of PLMs for NER from three complementary directions, shown in Figure 1. Instead of limiting ourselves in making use of limited in-domain labeled tokens with the classic approach, (i) we create prototypes as the representations for different entity types, and assign labels via the nearest neighbor criterion; (ii) we continuously pre-train PLMs using web data with noisy labels that is available in much larger quantities to improve NER accuracy and robustness; (iii) we tag the in-domain unlabeled data with soft labels via self-training (Xie et al., 2020), and perform semi-supervised learning in conjunction with the limited labeled data.

Our contributions include: (i) We present the first systematic study for few-shot NER, a problem that is little explored in the literature. Three distinctive schemes and their combinations are investigated. (ii) We perform comprehensive comparisons of these schemes on 10 public NER datasets from different domains. (iii) Compared with existing methods on few-shot and training-free NER settings, the proposed schemes achieve SoTA performance despite their simplicity. To shed light on future research on few-shot NER, our study suggests that: (i) Noisy supervised pre-training can significantly improve NER accuracy, and we will release our pre-trained checkpoints. (ii) Self-training consistently improves few-shot learning when the ratio of data amounts between unlabeled and labeled data is high. (iii) The performance of prototype learning varies on different datasets. It is useful when the number of labeled examples is small, or when new entity types are given in the training-free settings.

2 Background on Few-shot NER

Few-shot NER is a sequence labeling task, where the input is a text sequence (*e.g.,* sentence) of length T, $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T]$, and the output is a corresponding length-T labeling sequence $\mathbf{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_T]$, where $\boldsymbol{y} \in \mathcal{Y}$ is a one-hot vector indicating the entity type of each token from a pre-defined discrete label space. The training dataset for NER often consists of pair-wise data $\mathcal{D}^{\mathrm{L}} = \{(\mathbf{X}_n, \mathbf{Y}_n)\}_{n=1}^{N}$, where N is the number of training examples. Traditional NER systems are trained in the standard supervised learning paradigms, which usually requires a large number of pairwise examples, *i.e.,* N is large. In real-world applications, the more favorable scenarios are that only a small number of labeled examples are given for each entity type (N is small), because expanding labeled data increases annotation cost and decreases customer engagement. This yields a challenging task *few-shot NER*.

Linear Classifier Fine-tuning. Following the recent self-supervised PLMs (Devlin et al., 2019; Liu et al., 2019c), a typical method for NER is to utilize a Transformer-based backbone network to extract the contextualized representation of each token $\boldsymbol{z} = f_{\boldsymbol{\theta}_0}(\boldsymbol{x})$. A linear classifier (*i.e.,* a linear layer with parameter $\boldsymbol{\theta}_1 = \{\mathbf{W}, \boldsymbol{b}\}$ followed by a Softmax layer) is applied to project the representation $\boldsymbol{z}$ into the label space $f_{\boldsymbol{\theta}_1}(\boldsymbol{z}) = \mathrm{Softmax}(\mathbf{W}\boldsymbol{z}+\boldsymbol{b})$. In another word, the end-to-end learning objective for linear classifier based NER can be obtained via a function composition $\boldsymbol{y} = f_{\boldsymbol{\theta}_1} \circ f_{\boldsymbol{\theta}_0}(\boldsymbol{x})$, with trainable parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta}_0, \boldsymbol{\theta}_1\}$. The pipeline is shown in Figure 2(a). The model is optimized by minimizing the cross-entropy:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{(\mathbf{X},\mathbf{Y})\in\mathcal{D}^{\mathrm{L}}} \sum_{i=1}^{T} \mathrm{KL}(\boldsymbol{y}_i || q(\boldsymbol{y}_i|\boldsymbol{x}_i)), \quad (1)$$

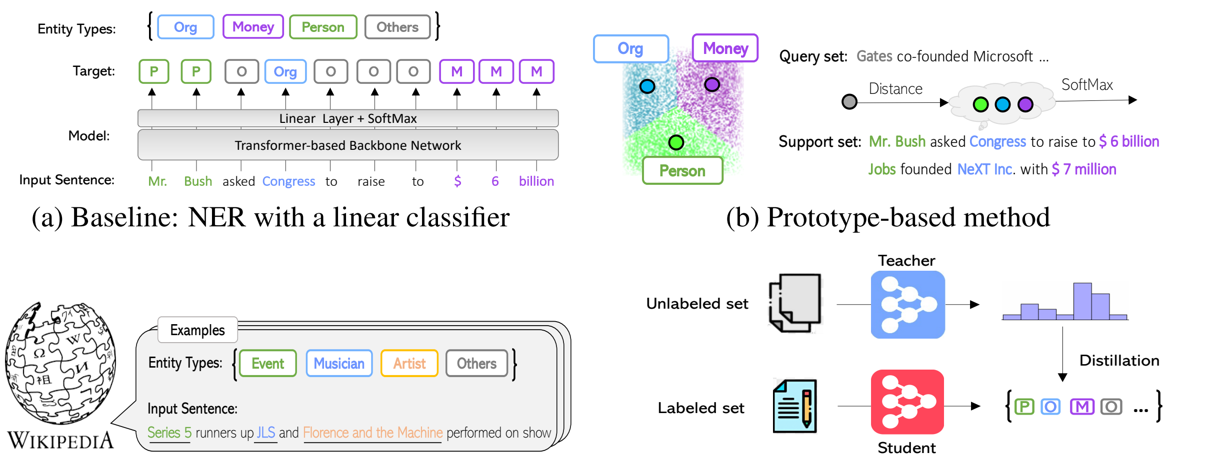
where the KL divergence between two distributions is $\mathrm{KL}(p||q) = \mathbb{E}_p \log(p/q)$, and the prediction probability vector for each token is

$$q(\boldsymbol{y}|\boldsymbol{x}) = \mathrm{Softmax}(\mathbf{W} \cdot f_{\boldsymbol{\theta}_0}(\boldsymbol{x}) + \boldsymbol{b}) \quad (2)$$

In practice, $\boldsymbol{\theta}_1 = \{\mathbf{W}, \boldsymbol{b}\}$ is always updated, while $\boldsymbol{\theta}_0$ can be either frozen (Liu et al., 2019a,b; Jie and Lu, 2019) or updated (Devlin et al., 2019; Yang and Katiyar, 2020).

3 Methods

When only a small number of labeled tokens are available, it renders difficulties for the classical supervised fine-tuning approach: the model tends to over-fit the training examples and shows poor generalization performance on the testing set (Fritzler et al., 2019). In this paper, we provide a comprehensive study specifically for limited NER data settings, and explore three orthogonal directions shown in Figure 1: (i) How to adapt meta-learning such as prototype-based methods for few-shot NER? (ii) How to leverage freely-available web data as noisy supervised pre-training data? (iii) How to leverage unlabeled in-domain sentences in a semi-supervised manner? Note that these three directions are complementary to each other and can be used jointly to further extrapolate the methodology space in Figure 1.



(a) Baseline: NER with a linear classifier

(b) Prototype-based method

(c) Noisy supervised pre-training

(d) Self-training

Figure 2: Illustration of different methods for few-shot NER. In this example, each token in the input sentence is categorized into one of the four entity types. (a) A typical NER system, where a linear classifier is built on top of unsupervised pre-trained Transformer-based networks such as BERT/Roberta. (b) A prototype set is constructed via averaging features of all tokens belonging to a given entity type in the support set (*e.g.,* the prototype for `Person` is an average of three tokens: *Mr.*, *Bush* and *Jobs*). For a token in the query set, its distances from different prototypes are computed, and the model is trained to maximize the likelihood to assign the query token to its target prototype. (c) The Wikipedia dataset is employed for supervised pre-training, whose entity types are related but different (*e.g.,* `Musician` and `Artist` are more fine-grained types of `Person` in the downstream task). The associated types on each token can be noisy. (d) Self-training: An NER system (teacher model) trained on a small labeled dataset is used to predict soft labels for sentences in a large unlabeled dataset. The joint of the predicted dataset and original dataset is used to train a student model.

3.1 Prototype-based Methods

Meta-learning (Ravi and Larochelle, 2017) has shown promising results for few-shot image classification (Tian et al., 2020) and sentence classification (Yu et al., 2018; Geng et al., 2019). It is natural to adapt this idea to few-shot NER. The core idea is to use episodic classification paradigm to simulate few-shot settings during model training. Specifically in each episode, M entity types (usually $M < |\mathcal{Y}|$) are randomly sampled from $\mathcal{D}^{\mathrm{L}}$, containing a *support* set $\mathcal{S} = \{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1}^{M \times K}$ (K sentences per type) and a *query* set $\mathcal{Q} = \{(\hat{\mathbf{X}}_i, \hat{\mathbf{Y}}_i)\}_{i=1}^{M \times K'}$ (K' sentences per type).

We build our method based on prototypical network (Snell et al., 2017), which introduces the notion of *prototypes*, representing entity types as vectors in the same representation space of individual tokens. To construct the prototype for the m-th entity type $\boldsymbol{c}_m$, the average of representations is computed for all tokens belonging to this type in the support set $\mathcal{S}$:

$$\boldsymbol{c}_m = \frac{1}{|\mathcal{S}_m|} \sum_{\boldsymbol{x} \in \mathcal{S}_m} f_{\boldsymbol{\theta}_0}(\boldsymbol{x}), \tag{3}$$

where $\mathcal{S}_m$ is the token set of the m-th type in $\mathcal{S}$, and $f_{\boldsymbol{\theta}_0}$ is defined in (2). For an input token $\boldsymbol{x} \in \mathcal{Q}$ from the query set, its prediction distribution is computed by a softmax function of the distance between $\boldsymbol{x}$ and all the entity prototypes. For example, the prediction probability for the m-th prototype is:

$$q(\boldsymbol{y} = \mathbb{I}_m | \boldsymbol{x}) = \frac{\exp\left(-d(f_{\boldsymbol{\theta}_0}(\boldsymbol{x}), \boldsymbol{c}_m)\right)}{\sum_{m'} \exp\left(-d(f_{\boldsymbol{\theta}_0}(\boldsymbol{x}), \boldsymbol{c}_{m'})\right)} \tag{4}$$

where $\mathbb{I}_m$ is the one-hot vector with 1 for m-th coordinate and 0 elsewhere, and $d(f_{\boldsymbol{\theta}_0}(\boldsymbol{x}), \boldsymbol{c}_m) = \|f_{\boldsymbol{\theta}_0}(\boldsymbol{x}) - \boldsymbol{c}_m\|_2$ is used in our implementation. We provide a simple example to illustrate the prototype method in Figure 2(b). At each training iteration, a new episode is sampled, and the model parameter $\boldsymbol{\theta}_0$ is updated via plugging (4) into (1). In the evaluation phase, the label of a new token $\boldsymbol{x}$ is assigned using the nearest neighbor criterion $\arg\min_m d(f_{\boldsymbol{\theta}_0}(\boldsymbol{x}), \boldsymbol{c}_m)$.

3.2 Noisy Supervised Pre-training

Generic representations via self-supervised pre-trained language models (Devlin et al., 2019; Liu et al., 2019c) have benefited a wide range of NLP applications. These models are pre-trained with the task of randomly masked token prediction on massive corpora, and are agnostic to the downstream tasks. In other words, PLMs treat each token equally, which is not aligned with the goal of NER: identifying named entities as emphasized tokens and assigning labels to them. For example, for a sentence " *Mr. Bush asked Congress to raise to \$ 6 billion* ", PLMs treat *to* and *Congress* equally, while NER aims to highlight entities like *Congress* and downplay their collocated non-entity words like *to*.

This intuition inspires us to endow the backbone network with an ability to upweight the representations of entities for NER. Hence, we propose to employ the large-scale noisy web data `WiFiNE` (Ghaddar and Langlais, 2018) for noisy supervised pre-training (NSP). The authors automatically annotated the 2013 English Wikipedia dump by querying anchored strings as well as the coreference mentions in each wiki page to the Freebase. The `WiFiNE` dataset is of 6.8GB and contains 113 entity types along with over 50 million sentences. Though introducing inevitable noises (*e.g.,* a random subset of 1000 mentions are manually evaluated and the accuracy of automatic annotations reaches 77% as reported in the paper, due to the error of identifying coreferences), this automatic annotation procedure is highly scalable and affordable. The label set of `WiFiNE` covers a wide range of fine-grained entity types, which are often related but different from entity types in the downstream datasets. For example in Figure 2(c), the entity types `Musician` and `Artist` in Wikipedia are more fine-grained than `Person` in a typical NER dataset. The proposed NSP learns representations to distinguish entities from others. This particularly favors the few-shot settings, preventing over-fitting via the prior knowledge of extracting entities from various contexts in pre-training.

Two pre-training objectives are considered in NSP, respectively: the first one is to use the linear classifier in (2), the other is a prototype-based objective in (4). For the linear classifier, we found that the batch size of 1024 and learning rate of $1e^{-4}$ works best, and for the prototype-based approach, we use the episodic training paradigm with $M = 5$ and set learning rate to be $5e^{-5}$. For both objectives, we train the whole corpus for 1 epoch and apply the Adam Optimizer (Kingma and Ba, 2015) with a linearly decaying schedule with warmup at 0.1. We empirically compare both objectives in experiments, and found that the linear classifier in (2) improves pre-training more significantly.

3.3 Self-training

Though manually labeling entities is expensive, it is easy to collect large amounts of unlabeled data in the target domain. Hence, it becomes desired to improve the model performance by effectively leveraging unlabeled data $\mathcal{D}^{\mathtt{U}}$ with limited labeled data $\mathcal{D}^{\mathtt{L}}$. We resort to the recent self-training scheme (Xie et al., 2020) for semi-supervised learning. The algorithm operates as follows:

1. Learn teacher model $\boldsymbol{\theta}^{\mathtt{tea}}$ via cross-entropy using (1) with labeled tokens $\mathcal{D}^{\mathtt{L}}$.
2. Generate soft labels using a teacher model on unlabeled tokens:
$$\tilde{\boldsymbol{y}}_i = f_{\boldsymbol{\theta}^{\mathtt{tea}}}(\tilde{\boldsymbol{x}}_i), \forall \tilde{\boldsymbol{x}}_i \in \mathcal{D}^{\mathtt{U}} \quad (5)$$
3. Learn a student model $\boldsymbol{\theta}^{\mathtt{stu}}$ via cross-entropy using (1) on labeled and unlabeled tokens:
$$\begin{aligned}\mathcal{L}_{\text{ST}} =& \frac{1}{|\mathcal{D}^{\mathtt{L}}|}\sum_{\boldsymbol{x}_i \in \mathcal{D}^{\mathtt{L}}} \mathcal{L}(f_{\boldsymbol{\theta}^{\mathtt{stu}}}(\boldsymbol{x}_i), \boldsymbol{y}_i) \\ &+ \frac{\lambda_{\mathtt{U}}}{|\mathcal{D}^{\mathtt{U}}|}\sum_{\tilde{\boldsymbol{x}}_i \in \mathcal{D}^{\mathtt{U}}} \mathcal{L}(f_{\boldsymbol{\theta}^{\mathtt{stu}}}(\tilde{\boldsymbol{x}}_i), \tilde{\boldsymbol{y}}_i) \quad (6)\end{aligned}$$
where $\lambda_{\mathtt{U}}$ is the weighting hyper-parameter.

A visual illustration for self-training procedure shown in Figure 2(d). It is optional to iterate from Step 1 to Step 3 multiple times, by initializing $\boldsymbol{\theta}^{\mathtt{tea}}$ in Step 1 with newly learned $\boldsymbol{\theta}^{\mathtt{stu}}$ in Step 3. We only perform self-training once for simplicity, which has already shown excellent performance.

4 Experiments

4.1 Settings

Methods. Throughout our experiments, the pre-trained base RoBERTa model is employed as the backbone network. We investigate the following 6 schemes for the comparative study: (*i*) **LC** is the *linear classifier* fine-tuning method in Section 2, *i.e.,* adding a linear classifier on the backbone, and directly fine-tuning on entire model on the target dataset; (*ii*) **P** indicates the *prototype-based method* in Section 3.1; (*iii*) **NSP** refers to the *noisy supervised pre-training* in Section 3.2; Depending on the pre-training objective, we have **LC+NSP** and **P+NSP**. (*iv*) **ST** is the *self-training* approach in Section 3.3, it is combined with *linear classifier* fine-tuning, denoted as **LC+ST**; (*v*) **LC+NSP+ST**.

We evaluate our methods on 10 public benchmark datasets, covering a wide range of domains.

Datasets	CoNLL	Onto	WikiGold	WNUT	Movie	Restaurant	SNIPS	ATIS	Multiwoz	I2B2
Domain	News	General	General	Social Media	Review	Review	Dialogue	Dialogue	Dialogue	Medical
#Train	14.0k	60.0k	1.0k	3.4k	7.8k	7.7k	13.6k	5.0k	20.3k	56.2k
#Test	3.5k	8.3k	339	1.3k	2.0k	1.5k	697	893	2.8k	51.7k
#Entity Types	4	18	4	6	12	8	53	79	14	23

Table 1: Statistics on the 10 public datasets studied in our NER benchmark.

The statistics of these datasets are summarized in Table 1, and detailed descriptions are provided in Appendix. For each dataset, we conduct three sets of experiments using various proportions of the training data: 5-shot, 10% and 100%. More experimental settings such as the hyper-parameters and evaluation details are shown in Appendix.

4.2 Comprehensive Comparison Results

To gain thorough insights and benchmark few-shot NER, we first perform an extensive comparative study on 6 methods across 10 datasets. The results are shown in Table 2. We can draw the following major conclusions: (*i*) By comparing columns ① and ② (or comparing ③ and ④), it clearly shows that noisy supervised pre-training provides better results in most datasets, especially in the 5-shot setting, which demonstrates that **NSP** endows the model an ability to extract better NER-related features. (*ii*) The comparison between columns ① and ③ provides a head-to-head comparison between linear classifier and prototype-based methods: while the prototype-based method demonstrates better performance than **LC** on CoNLL, WikiGold, WNUT17 and Multiwoz in the 5-shot learning setting, it falls behind **LC** on other datasets and in average statistics. It shows that the prototype-based method only yields better results when there is very limited labeled data: the size of both entity types and examples are small. (*iii*) When comparing columns ⑤ with ① (or comparing columns ⑥ and ②), we observe that using self-training consistently works better than directly fine-tuning with labeled data only, suggesting that **ST** is a useful technique to leverage in-domain unlabeled data if allowed. (*iv*) Column ⑥ shows the highest F1-score in most cases, demonstrating the three proposed schemes in this paper are complementary to each other, and can be combined to yield best results in practice.

In Figure 3, we show the learning curve of average F1-score on 5-shot CONLL-2003 testing dataset over 10 repeated experiments. The checkpoint via NSP provides a better initialization than Roberta, as NSP exhibits improvement over their

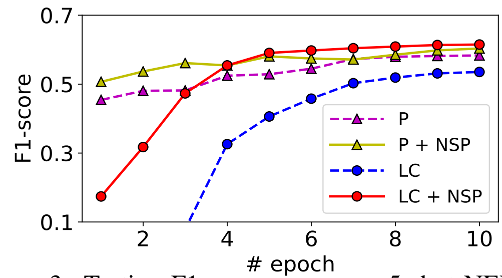


Figure 3: Testing F1-score curves on 5-shot NER on CONLL-2003 dataset.

counterpart methods at the beginning of learning, and eventually leads to higher F1-score.

4.3 Comparison with SoTA Methods

Competitive methods. The current SoTA on few-shot NER includes: (*i*) *StructShot* (Yang and Katiyar, 2020), which extends the nearest neighbor classification with a decoding process using abstract tag transition distribution. Both the model and the transition distribution are trained from the source dataset OntoNotes. (*ii*) *L-TapNet+CDT* (Hou et al., 2020) is a slot tagging method which constructs an embedding projection space using label name semantics to well separate different classes. It also includes a collapsed dependency transfer mechanism to transfer label dependency information from source domains to target domains. (*iii*) *SimBERT* is a simple baseline reported in (Yang and Katiyar, 2020; Hou et al., 2020); it utilizes a nearest neighbor classifier based on the contextualized representation output by the pre-trained BERT, without fine-tuning on few-shot examples. The results reported in the StructShot paper use IO schema instead of BIO schema, thus we report our performance on both for completeness.

For fair comparison, following (Yang and Katiyar, 2020), we also continuously pre-train our model on OntoNotes after the noisy supervised pre-training stage. For each 5-shot learning task, we repeat the experiments by re-sampling few-shot examples for 10 times. The results are reported in Table 3. We observe that our proposed methods consistently outperform the StructShot model across all three datasets, even by simply pre-training the

Datasets	Settings	① LC	② LC + NSP	③ P	④ P + NSP	⑤ LC + ST	⑥ LC + NSP + ST
CoNLL	5-shot	0.535	0.614	0.584	0.609	0.567	**0.654**
	10%	0.855	0.891	0.878	0.888	0.878	**0.895**
	100%	0.919	**0.920**	0.911	0.915	-	-
Onto	5-shot	0.577	0.688	0.533	0.570	0.605	**0.711**
	10%	0.861	**0.869**	0.854	0.846	0.867	0.867
	100%	0.892	**0.899**	0.886	0.883	-	-
WikiGold	5-shot	0.470	0.640	0.511	0.604	0.481	**0.684**
	10%	0.665	0.747	0.692	0.701	0.695	**0.759**
	100%	0.807	**0.839**	0.801	0.827	-	-
WNUT17	5-shot	0.257	0.342	0.295	0.359	0.300	**0.376**
	10%	0.483	0.492	0.485	0.478	0.490	**0.505**
	100%	0.489	0.520	0.552	**0.560**	-	-
MIT Movie	5-shot	0.513	0.531	0.380	0.438	0.541	**0.559**
	10%	0.651	0.657	0.563	0.583	0.659	**0.666**
	100%	**0.693**	0.692	0.632	0.641	-	-
MIT Restaurant	5-shot	0.487	0.491	0.441	0.484	0.503	**0.513**
	10%	0.745	0.734	0.713	0.721	**0.750**	0.741
	100%	0.790	**0.793**	0.787	0.791	-	-
SNIPS	5-shot	0.792	0.824	0.750	0.773	0.796	**0.830**
	10%	0.945	**0.950**	0.879	0.896	0.946	0.942
	100%	0.970	**0.972**	0.923	0.956	-	-
ATIS	5-shot	**0.908**	**0.908**	0.842	0.896	0.904	0.905
	10%	0.883	0.898	0.785	0.896	0.898	**0.903**
	100%	0.953	**0.956**	0.929	0.943	-	-
Multiwoz	5-shot	0.123	0.198	0.219	**0.451**	0.200	0.225
	10%	0.826	0.830	0.787	0.805	0.835	**0.841**
	100%	0.880	**0.885**	0.837	0.845	-	-
I2B2	5-shot	0.360	0.385	0.320	0.366	0.365	**0.393**
	10%	0.855	0.869	0.703	0.762	0.865	**0.871**
	100%	0.932	**0.935**	0.895	0.906	-	-
Average	5-shot	0.502	0.562	0.488	0.555	0.526	**0.585**
	10%	0.777	0.794	0.734	0.758	0.788	**0.799**
	100%	0.833	**0.841**	0.815	0.827	-	-

Table 2: F1-score on benchmark datasets with various sizes of training data. LC is *linear classifier* fine-tuning method, P is *prototype-based training* using a nearest neighbor objective, NSP is *noising supervised pre-training* and ST is *self-training*. The best results are in **bold**.

Schema	Methods	CoNLL	I2B2	WNUT	Average
IO	SimBERT †	0.286±0.025	0.091±0.007	0.077±0.022	0.151
	L-TapNet+CDT †	0.671±0.016	0.101±0.009	0.238±0.039	0.336
	StructShot †	0.752±0.023	0.318±0.018	0.272±0.067	0.447
	P + NSP	0.757±0.021	0.322±0.033	0.442±0.024	0.507
	LC + NSP	0.771±0.035	0.371±0.035	0.417±0.022	**0.520**
	LC + NSP + ST	0.779±0.040	0.376±0.028	0.419±0.028	**0.525**
BIO	P + NSP	0.756±0.017	0.334±0.024	0.424±0.012	0.505
	LC + NSP	0.712±0.048	0.364±0.032	0.403±0.029	0.493
	LC + NSP + ST	0.722±0.011	0.369±0.021	0.409±0.013	0.500

Table 3: Comparison of F1-score with SoTA on 5-shot NER tasks. Results of both BIO and IO schemas are reported for fair comparison. The best results are in **bold**. † indicates results from (Yang and Katiyar, 2020).

model on large-scale noisily tagged datasets like Wikipedia. Our best model outperforms the previous SoTA by 8% F1-score, which demonstrates that using large amounts of unlabeled in-domain corpus is promising for enhancing the few-shot NER performance.

4.4 Training-free Method Comparison

Some real-world applications require immediate inference on unseen entity types. For example, novel entity types with a few examples are frequently given in an online fashion, but updating model weights $\boldsymbol{\theta}$ frequently is prohibitive. One may store some token examples as supports and utilize them for nearest neighbor classification. The setting is referred to as *training-free* in (Wiseman and Stratos, 2019; Ziyadi et al., 2020), as the models identify new entities in a completely unseen target domain using only a few supporting examples in this new domain, without updating network parameters $\boldsymbol{\theta}$ in

Datasets	Methods	Number of support examples per entity type					
		10	*20*	*50*	*100*	*200*	*500*
ATIS	Neigh.Tag.†	0.067$\pm$0.008	0.088$\pm$0.007	0.111$\pm$0.007	0.143$\pm$0.006	0.221$\pm$0.006	0.339$\pm$0.006
	Example†	0.174$\pm$0.011	0.198$\pm$0.012	0.222$\pm$0.011	0.268$\pm$0.027	0.345$\pm$0.022	0.401$\pm$0.010
	Prototype	0.381$\pm$0.021	0.391$\pm$0.022	0.376$\pm$0.008	0.379$\pm$0.005	0.377$\pm$0.006	0.376$\pm$0.003
	Prototype + NSP	0.684$\pm$0.013	0.712$\pm$0.014	0.716$\pm$0.013	0.705$\pm$0.010	0.705$\pm$0.006	0.708$\pm$0.002
	Multi-Prototype	0.396$\pm$0.015	0.415$\pm$0.016	0.419$\pm$0.012	0.420$\pm$0.008	0.422$\pm$0.006	0.424$\pm$0.005
	Multi-Prototype + NSP	**0.712**$\pm$0.014	**0.748**$\pm$0.011	**0.760**$\pm$0.008	**0.742**$\pm$0.005	**0.743**$\pm$0.003	**0.746**$\pm$0.002
MIT.Restaurant	Neigh.Tag.†	0.042$\pm$0.018	0.038$\pm$0.008	0.037$\pm$0.007	0.046$\pm$0.008	0.055$\pm$0.011	0.081$\pm$0.006
	Example.†	0.276$\pm$0.018	0.295$\pm$0.010	0.312$\pm$0.007	0.337$\pm$0.005	0.345$\pm$0.004	0.346$\pm$0.000
	Prototype	0.330$\pm$0.013	0.332$\pm$0.013	0.332$\pm$0.010	0.329$\pm$0.003	0.329$\pm$0.004	0.331$\pm$0.003
	Prototype + NSP	0.455$\pm$0.016	0.455$\pm$0.012	0.455$\pm$0.013	0.438$\pm$0.013	0.437$\pm$0.008	0.438$\pm$0.006
	Multi-Prototype	0.345$\pm$0.012	0.360$\pm$0.015	0.371$\pm$0.012	0.376$\pm$0.009	0.385$\pm$0.005	0.386$\pm$0.004
	Multi-Prototype + NSP	**0.461**$\pm$0.019	**0.482**$\pm$0.011	**0.496**$\pm$0.008	**0.496**$\pm$0.011	**0.500**$\pm$0.005	**0.501**$\pm$0.003
MIT Movie	Neigh.Tag.†	0.031$\pm$0.020	0.045$\pm$0.019	0.041$\pm$0.011	0.053$\pm$0.009	0.054$\pm$0.007	0.086$\pm$0.008
	Example.†	**0.401**$\pm$0.011	**0.395**$\pm$0.007	**0.402**$\pm$0.007	**0.400**$\pm$0.004	**0.400**$\pm$0.005	**0.395**$\pm$0.007
	Prototype	0.175$\pm$0.007	0.168$\pm$0.006	0.170$\pm$0.004	0.174$\pm$0.003	0.173$\pm$0.002	0.173$\pm$0.002
	Prototype + NSP	0.303$\pm$0.011	0.293$\pm$0.007	0.285$\pm$0.006	0.284$\pm$0.002	0.282$\pm$0.002	0.280$\pm$0.002
	Multi-Prototype	0.197$\pm$0.007	0.207$\pm$0.005	0.219$\pm$0.004	0.227$\pm$0.002	0.229$\pm$0.003	0.230$\pm$0.002
	Multi-Prototype + NSP	0.364$\pm$0.020	0.368$\pm$0.011	0.380$\pm$0.006	0.382$\pm$0.003	0.354$\pm$0.003	0.383$\pm$0.002

Table 4: F1-score on training-free settings, *i.e.,* predicting novel entity types using nearest neighbor methods. The best results are in **bold**. † indicates results from (Ziyadi et al., 2020; Wiseman and Stratos, 2019).

that target domain. Our prototype-based method is able to perform such immediate inference. Two recent studies on training-free NER are: (i) *Neighbor-tagging* (Wiseman and Stratos, 2019) which copies token-level labels from weighted nearest neighbors; (ii) *Example-based NER* (Ziyadi et al., 2020) which is the SoTA on training-free NER, identifying the starting and ending tokens of unseen entity types.

We observed that our basic prototype-based method, under the training-free setting, does not gain from more given examples. We hypothesize that this is because tokens belonging to the same entity type are not necessarily close to each other, and are often separated in the representation space. Though it is hard to find one single centroid for all tokens in the same type, we assume that there exist local clusters of tokens belonging to the same type. To resolve such issue, we follow (Deng et al., 2020) and extend our method to a version called *Multi-Prototype*, by creating $K/5$ prototypes for each type given K examples per type. (*e.g.,* 2 prototypes per class are used for the 10-shot setting). The prediction score for a testing token belonging to a type is computed via averaging the prediction probabilities from all prototypes of the same type.

We compare with previous methods in Table 4 and observe that multi-prototype methods not only benefit from more support examples, but also surpass neighbor tagging methods and example-based NER by a large margin on two out of three datasets. For the MIT Movie dataset, one entity type can span a large chunk with multiple consecutive words in a sentence, which favors the span-based method like (Ziyadi et al., 2020). For example, the underlined part in the sentence "*what movie does the quote i dont think we are in kansas anymore come from*" is annotated as entity type `Quote`. The proposed methods in this paper can be combined with the span-based approach to specifically tackle this problem, and we leave it as future work. Further, if slightly fine-tuning is allowed, we see that the prototype-based method achieves 0.438 with 5-shot learning in Table 2, better than 0.395 achieved by example-based NER given 500 examples.

5 Related Work

General NER. NER is a long standing problem in NLP. Deep learning has significantly improved the recognition accuracy. Early efforts include exploring various neural architectures (Lample et al., 2016) such as Bidirectional LSTMs (Chiu and Nichols, 2016) and adding CRFs to capture structures (Ma and Hovy, 2016). Early studies have noticed the importance of reducing the annotation labor, where semi-supervised learning is employed, such as clustering (Lin and Wu, 2009), and combining supervised objective with unsupervised word representations (Turian et al., 2010). PLMs have recently revolutionized NER, where large-scale Transformer-based architectures (Peters et al., 2018; Devlin et al., 2019) are used as backbone network to extract informative representations. Contextualized string embedding (Akbik et al., 2018) is proposed to capture subword structures and polysemous words in different usage. Masked words and entities are jointly trained for prediction in (Yamada et al., 2020) with entity-aware self-attention.

These methods are designed for standard supervised learning, and have a limited generalization ability in few-shot settings, as empirically shown in (Fritzler et al., 2019).

Prototype-based methods recently become popular few-shot learning approaches in machine learning community. It was firstly studied in the context of image classification (Vinyals et al., 2016; Sung et al., 2018; Zhao et al., 2020), and has recently been adapted to different NLP tasks such as text classification (Wang et al., 2018; Geng et al., 2019; Bansal et al., 2020), machine translation (Gu et al., 2018) and relation classification (Han et al., 2018). The work closest related to ours is (Fritzler et al., 2019) which explores prototypical network on few-shot NER, but only utilizes RNNs as the backbone model and does not leverage the power of large-scale Transformer-based architectures for word representations. Our work is similar to (Ziyadi et al., 2020; Wiseman and Stratos, 2019) in that all of them utilize the nearest neighbor criterion to assign the entity type, but differs in that (Ziyadi et al., 2020; Wiseman and Stratos, 2019) consider every individual token instance for nearest neighbor comparison, while ours considers prototypes for comparison. Hence, our method is much more scalable when the number of given examples increases.

Supervised pre-training. In computer vision, it is a *de facto* standard to transfer ImageNet-supervised pre-trained models to small image datasets to pursue high recognition accuracy (Yosinski et al., 2014). The recent work named big transfer (Kolesnikov et al., 2019) has achieved SoTA on various vision tasks via pre-training on billions of noisily labeled web images. To gain a stronger transfer learning ability, one may combine supervised and self-supervised methods (Li et al., 2020c,b). In NLP, supervised/grounded pre-training have been recently explored for natural language generation (NLG) (Keskar et al., 2019; Zellers et al., 2019; Peng et al., 2020b; Gao et al., 2020; Li et al., 2020a). They aim to endow GPT-2 (Radford et al.), an ability of enabling high-level semantic controlling in language generation, and are often pre-trained on massive corpus consisting of text sequences associated with prescribed codes such as text style, content description, and task-specific behavior. In contrast to NLG, to our best knowledge, large-scale supervised pre-training has been little studied for natural language understanding (NLU). There are early studies showing promising results by transferring from medium-sized datasets to small datasets in some NLU applications. For example, from MNLI to RTE for sentence classification (Phang et al., 2018; Clark et al., 2020; An et al., 2020), and from OntoNER to CoNLL for NER (Yang and Katiyar, 2020). Our work further increases the supervised pre-training at the scale of web data (Ghaddar and Langlais, 2018), 1000 orders of magnitude larger than (Yang and Katiyar, 2020), showing consistent improvements.

Self-training. Self-training (Scudder, 1965) is one of the earliest semi-supervised methods, and has recently achieved improved performance for tasks such as ImageNet classification (Xie et al., 2020), visual object detection (Zoph et al., 2020), neural machine translation (He et al., 2020) and sentence classification (Mukherjee and Awadallah, 2020; Du et al., 2020). It is shown via object detection tasks in (Zoph et al., 2020) that stronger data augmentation and more labeled data can diminish the value of pre-training, while self-training is always helpful in both low-data and high-data regimes. Our work presents the first study of self-training for NER, and we observe similar phenomenons: it *consistently* boosts few-shot learning performance across all 10 datasets.

6 Conclusion and Future Work

We have presented an empirical study on several directions in few-shot NER. Three foundational methods and their combinations are systematically investigated: prototype-based methods, noisy supervised pre-training and self-training. They are intensively compared on 10 public datasets under various settings. All of them improve the PLM's generalization ability when learning from a few labeled examples, among which supervised pre-training and self-training turn out to be particularly effective. Our proposed schemes achieve SoTA on both few-shot and training-free settings compared with recent studies. We will release our benchmarks and code, in hope of inspiring future few-shot NER research with more advanced methods to tackle this challenging and practical problem.

For future work, we believe our studies can be combined with other interesting explorations in distant supervised learning, such as augmentation-based methods (Dai and Adel, 2020) and methods dealing with noisy labels (Meng et al., 2021). It would also be promising for researchers to consider larger pre-trained language models to learn better entity representations.

7 Ethical Considerations

The dataset WiFiNE (Ghaddar and Langlais, 2018) used in our noisy supervised pre-training stage is a public dataset. It is consistent with the terms of use of any sources and the original authors' intellectual property and privacy rights. As a modified version of Wikipedia dataset, the collection procedure ensures no ethical concerns *e.g.,* toxic language and hate speech. The entity types in our pre-training and fine-tuning datasets are common objects observed in daily life, detailed in Appendix.

Acknowledgments

We thank anonymous reviewers for valuable and insightful feedback. The authors are grateful to Linjie Li for providing guidance on the dataset benchmark construction.

References

A. Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING*.

Bang An, Jie Lyu, Zhenyi Wang, Chunyuan Li, Changwei Hu, Fei Tan, Ruiyi Zhang, Yifan Hu, and Changyou Chen. 2020. Repulsive attention: Rethinking multi-head attention as bayesian inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and J. Curran. 2009. Named entity recognition in wikipedia. In *PWNLP@IJCNLP*.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR*.

A. Coucke, A. Saade, Adrien Ball, Théodore Bluche, A. Caulier, D. Leroy, Clément Doumouro, Thibault Gisselbrecht, F. Caltagirone, Thibaut Lavril, Maël Primet, and J. Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *ArXiv*, abs/1805.10190.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *COLING*.

Jiankang Deng, Jia Guo, Tongliang Liu, Mingming Gong, and Stefanos Zafeiriou. 2020. Sub-center arcface: Boosting face recognition by large-scale noisy web faces. In *European Conference on Computer Vision*.

Leon Derczynski, Eric Nichols, M. Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *NUT@EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL*.

Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Ves Stoyanov, and Alexis Conneau. 2020. Self-training improves pre-training for natural language understanding. *arXiv preprint arXiv:2010.02194*.

Alexander Fritzler, V. Logacheva, and M. Kretov. 2019. Few-shot classification in named entity recognition task. *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*.

Jianfeng Gao, Michel Galley, and Lihong Li. 2019. Neural approaches to conversational ai. *Foundations and Trends® in Information Retrieval*, 13(2-3):127–298.

Jianfeng Gao, Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Heung-Yeung Shum. 2020. Robust conversational ai with grounded text generation. *arXiv preprint arXiv:2009.03457*.

Ruiying Geng, Binhua Li, Yongbin Li, Xiao-Dan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. In *EMNLP/IJCNLP*.

Abbas Ghaddar and P. Langlais. 2018. Transforming wikipedia into a large-scale fine-grained entity type corpus. In *LREC*.

Jiatao Gu, Yong Wang, Yun Chen, Kyunghyun Cho, and V. Li. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

J. Guo, G. Xu, X. Cheng, and Hang Li. 2009. Named entity recognition in query. In *SIGIR*.

Dilek Z. Hakkani-Tür, G. Tür, A. Çelikyilmaz, Yun-Nung Chen, Jianfeng Gao, L. Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *INTERSPEECH*.

Xu Han, Hao Zhu, Pengfei Yu, Z. Wang, Y. Yao, Zhiyuan Liu, and M. Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *8th International Conference on Learning Representations, ICLR*.

M. Hofer, A. Kormilitzin, P. Goldberg, and Alejo J. Nevado-Holgado. 2018. Few-shot learning for named entity recognition in medical text. *ArXiv*, abs/1811.05468.

Yutai Hou, W. Che, Y. Lai, Zhihan Zhou, Yijia Liu, H. Liu, and T. Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *ACL*.

Zhanming Jie and Wei Lu. 2019. Dependency-guided lstm-crf for named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. 2019. Big transfer (bit): General visual representation learning. *arXiv preprint arXiv:1912.11370*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *NAACL*.

Chunyuan Li, Xiang Gao, Yuan Li, Xiujun Li, Baolin Peng, Yizhe Zhang, and Jianfeng Gao. 2020a. Optimus: Organizing sentences via pre-trained modeling of a latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chunyuan Li, Xiujun Li, Lei Zhang, Baolin Peng, Mingyuan Zhou, and Jianfeng Gao. 2020b. Self-supervised pre-training with hard examples improves visual representations. *arXiv preprint arXiv:2012.13493*.

Junnan Li, Caiming Xiong, and Steven CH Hoi. 2020c. Mopro: Webly supervised learning with momentum prototypes. *arXiv preprint arXiv:2009.07995*.

Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1030–1038.

J. Liu, Panupong Pasupat, D. Cyphers, and James R. Glass. 2013a. Asgard: A portable architecture for multilingual dialogue systems. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8386–8390.

J. Liu, Panupong Pasupat, Y. Wang, D. Cyphers, and James R. Glass. 2013b. Query understanding enhanced by hierarchical parsing structures. *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 72–77.

Tianyu Liu, J. Yao, and Chin-Yew Lin. 2019a. Towards improving neural named entity recognition with gazetteers. In *ACL*.

Yanjun Liu, Fandong Meng, Jinchao Zhang, Jinan Xu, Yufeng Chen, and Jie Zhou. 2019b. Gcdt: A global context enhanced deep transition architecture for sequence labeling. In *ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019c. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Xuan Wang, Yu Zhang, Heng Ji, and Jiawei Han. 2021. Distantly-supervised named entity recognition with noise-robust learning and language model augmented self-training. In *EMNLP*.

Diego Mollá, Menno Van Zaanen, Daniel Smith, et al. 2006. Named entity recognition for question answering.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Uncertainty-aware self-training for text classification with few labels. *arXiv preprint arXiv:2006.15315*.

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

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020a. Soloist: Few-shot task-oriented dialog with a single pretrained auto-regressive model. *arXiv preprint arXiv:2005.05298*.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020b. Few-shot natural language generation for task-oriented dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners.

Weischedel Ralph, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium*.

S. Ravi and H. Larochelle. 2017. Optimization as a model for few-shot learning. In *ICLR*.

Alan Ritter, Mausam, Oren Etzioni, and S. Clark. 2012. Open domain event extraction from twitter. In *KDD*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL*.

H Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*.

Khaled Shaalan. 2014. A survey of arabic named entity recognition and classification. *Computational Linguistics*.

J. Snell, Kevin Swersky, and R. Zemel. 2017. Prototypical networks for few-shot learning. In *NIPS*.

A. Stubbs and Özlem Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/uthealth corpus. *Journal of biomedical informatics*, 58 Suppl:S20–9.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Yonglong Tian, Yue Wang, Dilip Krishnan, J. Tenenbaum, and Phillip Isola. 2020. Rethinking few-shot image classification: a good embedding is all you need? *ArXiv*, abs/2003.11539.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Sam Wiseman and K. Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Qizhe Xie, E. Hovy, Minh-Thang Luong, and Quoc V. Le. 2020. Self-training with noisy student improves imagenet classification. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, H. Takeda, and Yuji Matsumoto. 2020. Luke: Deep contextualized entity representations with entity-aware self-attention. In *EMNLP*.

Y. 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)*.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, S. Chang, Saloni Potdar, Yu Cheng, G. Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. In *NAACL-HLT*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*.

Yang Zhao, Chunyuan Li, Ping Yu, and Changyou Chen. 2020. Remp: Rectified metric propagation for few-shot learning. *arXiv preprint arXiv:2012.00904*.

M. Ziyadi, Yuting Sun, A. Goswami, Jade Huang, and W. Chen. 2020. Example-based named entity recognition. *ArXiv*, abs/2008.10570.

Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin D Cubuk, and Quoc V Le. 2020. Rethinking pre-training and self-training. *arXiv preprint arXiv:2006.06882*.

A Experimental Settings

Datasets. We evaluate our methods on 10 public benchmark datasets, covering a wide range of domains: OntoNotes 5.0 (Ralph et al., 2013), WikiGold[1] (Balasuriya et al., 2009) on general domain, CoNLL 2003 (Sang and Meulder, 2003) on news domain, WNUT 2017 (Derczynski et al., 2017) on social domain, MIT Moive (Liu et al., 2013b) and MIT Restaurant[2] (Liu et al., 2013a) on review domain, SNIPS[3] (Coucke et al., 2018), ATIS[4] (Hakkani-Tür et al., 2016) and Multiwoz[5] (Budzianowski et al., 2018) on dialogue domain, and I2B2[6] (Stubbs and Uzuner, 2015) on medical domain. The detailed statistics of these datasets are summarized in Table 1.

For each dataset, we conduct three sets of experiments using various proportions of the training data: 5-shot, 10% and 100%. For 5-shot setting, we sample 5 sentences for each entity type in the training set and repeat each experiment for 10 times. For 10% setting, we down-sample 10 percent of the training set, and for 100% setting, we use the full training set as labeled data. We only study the self-training method in 5-shot and 10% settings, by using the rest of the training set as unlabeled in-domain corpus.

Hyper-parameters. We have described details for noisy supervised pre-training in Section 3.2. For training on target datasets, we set a fixed set of hyperparameters across all the datasets: For the linear classifier, we set batch size = 16 for 100% and 10% settings, batch size = 4 for 5-shot setting. For each episode in the prototype-based method, we set the number of sentences per entity type in support and query set (K, K') to be $(5, 15)$ for 100% and 10% settings, and $(2, 3)$ for 5-shot setting. For both training objectives, we set learning rate = $5e^{-5}$ for 100% and 10% settings, and learning rate = $1e^{-4}$ for 5-shot setting. For all training data sizes, we set training epoch = 10, and Adam optimizer (Kingma and Ba, 2015) is used with the same linear decaying schedule as the pre-training stage. For self-training, we set $\lambda_{\mathtt{U}} = 0.5$.

Evaluation. We follow the standard protocols for NER tasks to evaluate the performance on the test set (Sang and Meulder, 2003). Since RoBERTa tokenizes each word into subwords, we generate word-level predictions based on the first word piece of a word. Word-level predictions are then turned into entity-level predictions for evaluation when calculating the f1-score. Two tagging schemas are typically considered to encode chunks of tokens into entities: BIO schema marks the beginning token of an entity as `B-X` and the consecutive tokens as `I-X`, and other tokens are marked as `O`. IO schema uses `I-X` to mark all tokens inside an entity, thus is defective as there is no boundary tag between same type of entities. In our study, we use BIO schema by default, but also report the performance evaluated by IO schema for fair comparison with some previous studies.

B Dataset statistics

We show the entity types and their corresponding frequencies in pre-training dataset in Table 5 and downstream benchmark datasets in Table 6 and Table 7. We see that the entity types for pre-training and fine-tuning are semantically related, but different in granularity. For example, the location category in pre-training dataset contains fine-grained entity types like country, city, road, and bridge, while the Onto dataset for fine-tuning only gives a coarse-grained partition by geopolitical locations (countries, cities) and non geopolitical ones (highways, bridges). Further, for each categories of entity types, the pre-training dataset has a much higher frequency than fine-tuning dataset, allowing the model to learn heterogeneous contextual knowledge before deploying to a specific domain.

[1] https://github.com/juand-r/entity-recognition-datasets
[2] https://groups.csail.mit.edu/sls/downloads/
[3] https://github.com/sonos/nlu-benchmark/tree/master/2017-06-custom-intent-engines
[4] https://github.com/yvchen/JointSLU
[5] https://github.com/budzianowski/multiwoz
[6] https://portal.dbmi.hms.harvard.edu/projects/n2c2-2014/

Entity type name	# Entities	Entity type name	# Entities
person	10976096	person/author	8603619
person/athlete	5990955	person/actor	5646333
person/fictional character	4820634	person/musician	4414457
person/ethnicity	1603194	person/politician	4196151
person/artist	3971716	person/director	870744
person/monarch	847943	person/soldier	297018
person/coach	284695	person/religious leader	272999
person/engineer	202533	person/architect	192587
person/doctor	101930	person/terrorist	2759
location	11234535	location/city	13478825
location/country	10022782	location/province	2555375
location/island	741533	location/body of water	1372583
location/county	930301	location/road	740874
location/astral body	410792	location/mountain	409878
location/cemetery	155498	location/park	78388
location/railway	61438	location/bridge	39528
location/glacier	17158		
organization	5280100	organization/company	8070793
organization/sports team	3236586	organization/educational institution	2124661
organization/government	1146508	organization/military	1118635
organization/political party	1006768	organization/sports league	854429
organization/news agency	378262	organization/government agency	314572
organization/airline	170127	organization/terrorist organization	40272
organization/fraternity sorority	35299		
art	3420964	art/music	4480181
art/written work	3284486	art/film	2583704
art/play	214837	art/newspaper	17488
building	2038445	building/sports facility	289182
building/airport	235172	building/theater	134604
building/hospital	89793	building/restaurant	50499
building/hotel	41571	building/library	24556
building/power station	18211	building/dam	10634
computer/algorithm	1808698	computer/programming language	110646
event	2877275	event/military conflict	1199857
event/attack	453078	event/election	358101
event/sports event	268484	event/natural disaster	149982
event/protest	93586	event/terrorist attack	2244
livingthing	2736344	livingthing/animal	1117967
product	969818	product/ship	776310
product/game	770000	product/instrument	622648
product/train	558943	product/software	550727
product/car	347887	product/airplane	321361
product/weapon	320842	product/spacecraft	56071
product/computer	50812	product/mobile phone	14585
product/engine device	31956	product/camera	10198
education/educational degree	507088	education/department	126584
medicine/symptom	440572	medicine/medical treatment	360235
medicine/drug	158258		
finance/currency	140008	finance/stock exchange	14861
broadcast program	1944347	broadcast/tv channel	91262
time	31543479	title	5752995
language	1565042	broadcast network	997300
food	903886	disease	743015
body part	741448	religion	666482
god	578091	chemistry	542973
award	488515	internet website	259798
law	230483	transit	132448
biology	123600	metropolitan transit line	92136

Table 5: Entity type names and corresponding numbers on Wikipedia data used in supervised pre-training. For better visualization, we group entity label names belonging to the same root into the same blocks.

Dataset	Entity type name	# Entities	Entity type name	# Entities
CONLL-2003	location	7140	person	6600
	organization	6321	misc.	3438
Onto	person	15429	countries, cities, states	15405
	organization	12820	date	10922
	cardinal	7367	political groups	6870
	money	2434	percent	1763
	ordinal number	1640	time	1233
	work of art	974	buildings,highways,bridges	860
	event	748	quantity	657
	product	606	language	304
	law	282	other locations	12
Wikigold	location	628	organization	559
	person	538	misc.	365
WNUT17	person	660	location	548
	group	264	corporation	221
	product	142	creative work	140
MIT Movie	plot	6468	actor	5010
	genre	3384	year	2702
	character name	1025	director	1787
	opinion	810	origin	779
	relationship	580	award	309
	quotation	126	soundtrack	50
MIT Restaurant	location	3817	cuisine	2839
	amenity	2541	restaurant name	1901
	dish	1475	rating	1070
	hours	990	price	730
SNIPS	(AddToPlaylist) playlist	1869	(AddToPlaylist) playlist owner	1107
	(AddToPlaylist) music item	887	(AddToPlaylist) artist	738
	(AddToPlaylist) entity name	590	(BookRestaurant) restaurant type	1359
	(BookRestaurant) party size number	1022	(BookRestaurant) time range	674
	(BookRestaurant) state	519	(BookRestaurant) city	513
	(BookRestaurant) restaurant name	339	(BookRestaurant) country	356
	(BookRestaurant) spatial relation	324	(BookRestaurant) party size description	316
	(BookRestaurant) served dish	269	(BookRestaurant) cuisine	210
	(BookRestaurant) sort	203	(BookRestaurant) facility	159
	(BookRestaurant) poi	143	(GetWeather) time range	1047
	(GetWeather) city	851	(GetWeather) country	498
	(GetWeather) state	491	(GetWeather) condition temperature	476
	(GetWeather) condition description	454	(GetWeather) geographic poi	290
	(GetWeather) current location	271	(GetWeather) spatial relation	209
	(PlayMusic) artist	1169	(PlayMusic) music item	791
	(PlayMusic) service	756	(PlayMusic) year	630
	(PlayMusic) sort	346	(PlayMusic) track	211
	(PlayMusic) album	176	(PlayMusic) playlist	149
	(PlayMusic) genre	144	(RateBook) rating value	1924
	(RateBook) rating unit	1103	(RateBook) best rating	1033
	(RateBook) object name	979	(RateBook) object select	952
	(RateBook) object type	919	(RateBook) object part of series type	307
	(SearchCreativeWork) object name	1951	(SearchCreativeWork) object type	1462
	(SearchScreeningEvent) movie name	808	(SearchScreeningEvent) object type	692
	(SearchScreeningEvent) movie type	674	(SearchScreeningEvent) spatial relation	665
	(SearchScreeningEvent) location name	586	(SearchScreeningEvent) object location type	458
	(SearchScreeningEvent) time range	265		

Table 6: Entity type names and corresponding numbers on benchmark NER datasets CONLL-2003, Onto, WikiGold, WNUT17, MIT Movie, MIT Restaurant, and SNIPS.

Dataset	Entity type name	# Entities	Entity type name	# Entities
ATIS	(fromloc) city name	4326	(fromloc) airport name	89
	(fromloc) state code	46	(fromloc) state name	39
	(fromloc) airport code	15	(toloc) city name	4343
	(toloc) state code	86	(toloc) state name	77
	(toloc) airport name	39	(toloc) airport code	20
	(toloc) country name	3	(depart date) day name	889
	(depart date) day number	395	(depart date) month name	379
	(depart date) today relative	84	(depart date) date relative	82
	(depart date) year	25	(depart time) time	692
	(depart time) period of day	593	(depart time) time relative	323
	(depart time) period mod	44	(depart time) start time	25
	(depart time) end time	25	(stoploc) city name	239
	(arrive time) time relative	187	(arrive time) time	172
	(arrive time) period of day	64	(arrive time) start time	21
	(arrive time) end time	20	(arrive time) period mod	4
	(arrive date) day name	88	(arrive date) month name	47
	(arrive date) day number	47	(arrive date) date relative	11
	(arrive date) today relative	2	(return date) date relative	10
	(return date) month name	4	(return date) day number	4
	(return date) today relative	1	(return date) day name	1
	(stoploc) state code	5	(stoploc) airport name	1
	(return time) period of day	3	(return time) period mod	2
	airline name	701	round trip	348
	cost relative	344	flight mod	329
	city name	227	class type	217
	flight stop	168	airline code	136
	flight number	84	fare basis code	76
	flight time	71	meal description	57
	fare amount	53	transport type	48
	connect	40	flight days	39
	airport name	38	economy	36
	airline name	32	aircraft code	31
	mod	30	airport code	29
	restriction code	23	meal	17
	state code	8	meal code	6
	day name	5	period of day	5
	days code	3	time	2
	today relative	2	state name	2
	month name	2	day number	2
	time relative	1		
Multiwoz	restaurant food	4041	restaurant name	3054
	hotel name	2863	restaurant booktime	2361
	attraction name	1972	train leaveat	1617
	train arriveby	1518	taxi departure	995
	taxi destination	970	taxi leaveat	800
	taxi arriveby	439	hospital department	107
	bus destination	3	bus departure	2
I2B2	date	5195	doctor	1989
	patient	931	hospital	925
	medical record	408	city	258
	phone	233	state	221
	username	219	street	208
	id num	174	profession	150
	zip	139	organization	85
	country	53	age	8
	device	7	tax	5
	other locations	4	email	3
	url	2	bioid	1
	health plan	1		

Table 7: Entity type names and corresponding numbers on benchmark NER datasets ATIS, Multiwoz, and I2B2.