LEARNING COMMON DEPENDENCY STRUCTURE FOR UNSUPERVISED CROSS-DOMAIN NER

Luchen Liu[†‡], Xixun Lin[†], Peng Zhang[†], Lei Zhang[†], Bin Wang[‡]*

[†]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[†]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[‡]Xiaomi AI Lab, Beijing, China

ABSTRACT

Unsupervised cross-domain NER task aims to solve the issues when data in a new domain are fully-unlabeled. It leverages labeled data from source domain to predict entities in unlabeled target domain. Since training models on large domain corpus is time-consuming, in this paper, we consider an alternative way by introducing syntactic dependency structure. Such information is more accessible and can be shared between sentences from different domains. We propose a novel framework with dependency-aware GNN (DGNN) to learn these common structures from source domain and adapt them to target domain, alleviating the data scarcity issue and bridging the domain gap. Experimental results show that our method outperforms state-of-the-art methods.

Index Terms— Named entity recognition, unsupervised cross-domain, graph neural networks, dependency structure

1. INTRODUCTION

Named entity recognition (NER) is a fundamental technology of knowledge extraction [1]. It aims at extracting named-entities mentioned in such unstructured texts and classifying them into pre-defined categories. Normally, we train NER model and extract entities from texts in a certain domain. However, not all domains have sufficient labeled training data and such approach will be unavailable. So cross-domain NER [2], which aims to adapt the source knowledge to target domains lacking labeled data, has received much attention. Compared with traditional NER task, it is more challenging due to the difficulty of modeling different text genres across domains. Most cross-domain methods typically follow a supervised learning manner [3] in which the labels of training data for both source and target domains are required. However, for such low-resource target domain, collecting enough annotated data is very costly.

To sidestep the above dilemma [4], unsupervised cross-domain NER [5, 6] is a new task following the unsupervised domain adaption [7] paradigm. It only requires labeled NER data of source domain, so we can just train models on these data and directly adapt them to the unlabeled target domain for test. The core problem in unsupervised cross-domain NER is the domain gap, i.e., sentences across domains often have different vocabularies or idiomatic expressions. To bridge the domain gap between source and target domains, external information is necessary. Some promising methods [5, 6] use external unlabeled data corpora on both of the source and target domains to train language models and achieve state-of-the-art results. Unfortunately, such large-scale domain corpora are not always available and training a good language model is a non-trivial task. Others [8, 4] conduct unsupervised cross-domain NER with much less external information, but their performances are not satisfying empirically.

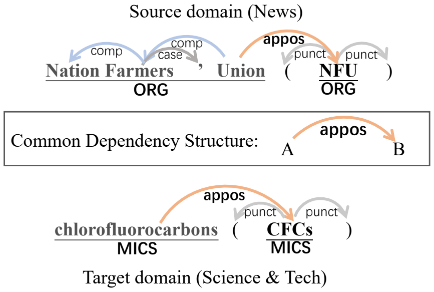
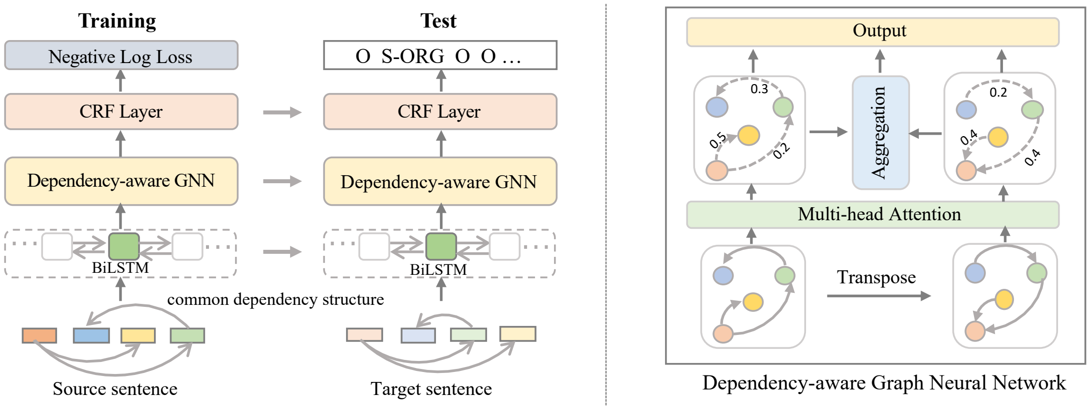

Source domain (News)
comp
comp
case
appos
punct
punct
Nation Farmers ' Union (NFU)
ORG
ORG
Common Dependency Structure:
appos
A
B
appos
punct
punct
chlorofluorocarbons (CFCs)
MICS
MICS
Target domain (Science & Tech)

Fig. 1: A concrete example to illustrate the common dependency structure of the two sentences from the source domain and target domain respectively. The common dependency structure `A-appos-B` helps the model recognize the entity better on the low-resource target domain.

Recent works [9, 10, 11] show that the dependency structure of sentences is important auxiliary information. The dependency structure, usually formulated as graph or tree structure, shows the directed links and syntactic relations between words in a sentences, which can be accessible easily by a dependency parser. A lot of works try to convert the dependency structures of sentences into trees or graphs, modelling the relation between words via tree-based Recurrent Neural Networks (RNN) [12] or Graph Neural Networks (GNN) [13] as extra features. But the importance of dependency structures

[*]Corresponding author (pengzhang@iie.ac.cn).

978-1-6654-0540-9/22/$31.00 ©2022 IEEE



(a) (b)

Fig. 2: A simple illustration of the proposed framework (a) with our Dependency-aware GNN (b). With the goal of learning common dependency structure across domains, we train our NER model with DGNN on the source domain and apply it to the target domain.

for domain adaptation is less explored. In fact, the common dependency structures facilitated from sentences across domains are critical to our task. As shown in Fig.1, the phrase pairs, *Nation Farmers' Union - NFU* in source domain and *chlorofluorocarbons - CFCs* in target domain share the same dependency structure A−appos−B (B serves to define A, i.e., A and B have the same entity type). Actually, the word "NFU" in source domain and "CFCs" in target domain can be easily recognized due to the capital letter pattern learned previous methods. But "chlorofluorocarbons" in the target domain is a long-tailed word without special pattern. It is difficult to be inferred by previous methods if the corpus information is not enough. Thus, if our model can learn the common dependency structure from source domain, the long-tailed word "chlorofluorocarbons" can be correctly labeled. It is significant to make the model capable to learn such common dependency structures from the source domain data and exploit them for better domain adaptation.

In this paper, we propose a novel framework to improve the performance of unsupervised cross-domain NER. Specifically, we convert sentences with dependency relations into graphs for both domains. Compared with other external information like massive raw data, these dependency graphs are easier to obtain. To the best of our knowledge, we are the first to exploit the common dependency structures across domains and apply them to unsupervised cross-domain NER. Building upon the dependency structures, we develop a Dependency-aware Graph Neural Network (DGNN) to model the relational structure of each dependency graph. Our DGNN generates feature sequences into a latent common space by aggregating neighbors over dependency graphs of words. Using this common space, our model can learn the shared dependency structure just from source domain and adapt it to the target domain naturally. Moreover, different from traditional Graph Convolutional Networks (GCN) [14], our DGNN incorporates directions of dependency graph and applies attention mechanism to reduce the dependency parsing errors. Experimental results show our method outperforms state-of-the-art methods for unsupervised cross-domain NER task.

2. PROPOSED METHOD

Fig.2 illustrates our framework with Dependency-aware GNN. The architecture of our model is composed of three main components: (1) a BiLSTM encoder to initialize node representation for GNN, (2) a Dependency-aware GNN to learn the dependency structure, and (3) the CRF layer for NER tagging. We train our model on the source domain and apply it to the target domain via parameter transferring.

2.1. Input Representation

The input graph of GNN can be built upon the dependency tree of the input sentence, where an edge is added to every two words, if there exists a dependency relation between them. Therefore, given an input sentence $\mathcal{S} = \{w_i\}_{i=1}^{N}$, we first get its dependency tree $\mathcal{T} = \{(\mathbf{head}(w_i), r_i)\}_{i=1}^{N}$ by dependency parsing, where $\mathbf{head}(w_i)$ is the parent word of w_i on the dependency tree and r_i is the dependency relation between w_i and its parent word[1]. According to $\mathcal{S}$ and $\mathcal{T}$, we convert each sentence into the dependency graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is a set of N words in a sentence, and $\mathcal{E} = \{a_1, a_2, \ldots, a_M\}$ is a set of M dependency edges between words. We convert each node v_i into a vector $\mathbf{v}_i$ via concatenating predefined representations: word embedding, contextualized embedding, character embedding and dependency relation embedding as $\mathbf{v}_i$. The node vectors $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N\}$ are then fed into a BiLSTM to generate initial input of GNN as $\{\mathbf{x}_1^0, \mathbf{x}_2^0, \ldots, \mathbf{x}_N^0\}$.

[1] The root word of the dependency tree does not have parent word but has the *root* dependency relation.

Table 1: Performances comparison on the different target domains. The symbols † and ‡ denote that the numbers are retrieved from [4] and [5], respectively. *N/A* means no external resource is applied.

Dataset (*Source→Target*)	**External Resource**	**Models**	**Precision(%)**	**Recall(%)**	**F1-score(%)**
CoNLL03→SciTech	*N/A*	Transfer-Baseline	68.67	78.33	73.18
		MTL+MoEE [4]†	-	-	69.53
		EAAT [15]	68.96	79.16	73.71
	Descriptions / Corpus	Concept Tagger [16]†	-	-	67.14
		Robust Sequence [8]†	-	-	68.12
		Self-Train [17]‡	62.56	75.04	68.24
		DANN [18]‡	65.14	73.84	69.22
		Cross-Domain LM [5]‡	68.48	79.52	73.59
		Multi-Cell LSTM [6]	-	-	73.56
	Dependency Structure	DGLSTM-CRF [10]	67.95	79.04	73.07
		Tree-LSTM-CRF [12]	69.33	79.18	73.93
		Ours	**70.00**	**79.64**	**74.51**
CoNLL03→Twitter	*N/A*	Transfer-Baseline	38.46	61.74	47.39
		EAAT [15]	51.73	58.01	54.69
	Descriptions / Corpus	DANN [18]	45.59	58.70	51.33
		Cross-Domain LM [5]	59.41	44.53	50.91
	Dependency Structure	DGLSTM-CRF [10]	40.72	60.57	48.70
		Tree-LSTM-CRF [12]	39.80	63.62	48.97
		Ours	49.66	63.04	**55.56**

2.2. Dependency-aware GNN

We further propose a Dependency-aware GNN (DGNN) to capture the structural information of each dependency graph. Our DGNN considers bi-directional message passing among nodes with attention mechanism.

Directional Message Passing. Since the dependency graph is built from the tree structure, it is a uni-directional structure. However, there is no reason to assume that information flows only along the dependency arcs, i.e, from the parent word to the child word. Therefore, As [19] suggests, we allow node features to flow along both directions of edges. Here, we denote the transpose graph of $\mathcal{G}$ as $\mathcal{G}'(\mathcal{V}, \mathcal{E}')$, where $\mathcal{E}'$ is the set of corresponding inverse edges in $\mathcal{G}$. The transposed graph allows the information to flow in the opposite direction as well.

Multi-head Attention. Considering some attachment errors between nodes in dependency parsing, we use the multi-head attention to assign different weights to edges [13]. For each attention head, we calculate the attention score $\hat{\alpha}_{ij}$ between the node i and j:

$$\hat{\alpha}_{ij} = \frac{\exp(\cos(\mathbf{x}_i, \mathbf{x}_j))}{\sum_{k \in \mathcal{N}(i) \cup \{i\}} \exp\left(\cos\left(\mathbf{x}_i, \mathbf{x}_k\right)\right)}, \quad (1)$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are node features and $\mathcal{N}(i)$ denotes the set of neighbors of node i. Then we can obtain the final attention score α_{ij} via a linear transformation $\mathbf{W}_\alpha$ on n attention heads as

$$\alpha_{ij} = \left[\hat{\alpha}_{ij}^{(1)} \oplus \ldots \oplus \hat{\alpha}_{ij}^{(n)}\right] \mathbf{W}_\alpha, \quad (2)$$

where $\oplus$ is the concatenation operation.

Feature Aggregation. Our DGNN generates the node representation by aggregating neighbors over dependency graph recursively. We define the aggregation operation $Agg(\cdot)$ for each node on graph $\mathcal{G}$ as

$$Agg(\mathbf{x}_i, \mathcal{G}) = ReLU\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{ij} \mathbf{W} \mathbf{x}_j\right), \quad (3)$$

where $\mathbf{W}$ is a learnable matrix and $ReLU(\cdot)$ is the rectifier linear unit activation function. Combining with the node feature aggregated on the transpose graph $\mathcal{G}'$, the node feature $\mathbf{x}_i^k$ is expressed as

$$\mathbf{x}_i^k = MLP\left(Agg(\mathbf{x}_i^{k-1}, \mathcal{G}) + Agg(\mathbf{x}_i^{k-1}, \mathcal{G}')\right), \quad (4)$$

where $MLP(\cdot)$ is a multilayer perceptron (MLP) layer and k denotes the layer number and $\mathbf{x}_0$ is the initial input of GNN.

2.3. Tagging and Loss function

For the tagging layer, we use the standard Conditional Random Field (CRF) layer [20] on top of the GNN layer. CRF can use the state feature function and the state transfer function to consider the annotation information of adjacent words. Given the feature sequence $\mathbf{X} = \{\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k\}$ from the last layer of DGNN, the CRF layer predicts the probability of label sequence $\mathbf{y}$ as

$$P(\mathbf{y}|\mathbf{X}) = \frac{\exp(score(\mathbf{X}, \mathbf{y}))}{\sum_{\mathbf{y}'} \exp\left(score\left(\mathbf{X}, \mathbf{y}'\right)\right)}, \quad (5)$$

where the score is the sum of transitions from y_i to y_{i+1} as $A_{y_i, y_{i+1}}$ and emissions from another BiLSTM as $F_{\mathbf{X}, y_i}$:

$$score(\mathbf{X}, \mathbf{y}) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} F_{\mathbf{X}, y_i}. \quad (6)$$

Table 2: Ablation study of our model on SciTech. We also report the numbers of parameters of each compared models. Δ represents the reduction of F1-score compared with Ours.

Models	#Params	F1(%)	Δ(%)
BERT-BASE FINETUNE [26]	108M	74.23	
DGLSTM-CRF [10]	4.79M	73.07	
TREE-LSTM-CRF [12]	3.73M	73.89	
OURS	5.78M	**74.51**	
- w/o DGNN	4.05M	73.18	-1.33
- w/o multi-head attention	4.90M	74.03	-0.48
- w/o directional message passing	5.22M	73.88	-0.63
- w/o char embedding	5.71M	74.17	-0.34
- w/o relation embedding	5.58M	73.77	-0.74

When decoding, we search for the label sequence with the highest conditional probability. During training, we minimize the sentence-level negative log-likelihood [21] to obtain the model parameters and transition parameters.

3. EXPERIMENTS

3.1. Settings

Following [5, 15], we take the CoNLL03 dataset [22] as source domain, which is from news domain. And we take SciTech [5] and Twitter [23] dataset as our target domain data, which are related to the domain of science & technology and social media, respectively. These two datasets have comparable samples to the CoNLL03 test set. All of these datasets contain the same four types of entities. For all experiments, we use the CoNLL03 train/dev set for training/validation and test the model on SciTech and Twitter dataset [5, 6, 4]. We use the dependency parser in StanfordNLP Toolkit to generate the syntactic dependency annotations for each sentence. We choose 300-d pretrained FastText [24] for word embedding initialization and use pretrained ELMo [25] to extract word-level contextualized embeddings. We set the hidden size of DGNN and BiLSTM as 200. All models are trained for 50 epochs via SGD with 0.01 learning rate and 0.9 momentum.

3.2. Results

Performance comparison. We conduct experiments on the two benchmark datasets, compared with several advanced methods. The overall results are shown in Table 1. For each dataset, the baselines can be divided into three groups. The baselines in the first block are those without any external information. We also set a BiLSTM-CRF baseline w/o DGNN module termed as TRANSFER-BASELINE. The methods in the second block use high-resource knowledge like entity descriptions and domain corpus as external information. Those in the third block are methods using accessible dependency structures. On SciTech, among these methods with high-resource external information, CROSS-DOMAIN LM [5] and MULTI-CELL LSTM [6] have similar performances, and our method outperforms them. Specifically, compared with all state-of-the-art models, our model gives 74.51 F1 score, significantly better than the best result obtained by EAAT [15]. And among methods in the third block, our model is the best of all baselines using dependency structures. The results show the significant improvement of the DGNN. On the Twitter dataset, considering F1 score, OURS shows significant improvement compared with TRANSFER-BASELINE (by 8.71) and other methods using dependency structures (by 6-7). Our model also achieves the best performance.

Table 3: Results of F1 score with different contextualized embeddings on SciTech.

Features	Pre(%)	Rec(%)	F1(%)
Ours w/o Context Embedding	66.34	67.63	66.98
Ours w/ BERT Embedding	**70.71**	76.95	73.70
Ours w/ ELMo Embedding	70.00	**79.64**	**74.51**

Ablation Study. Table 2 shows the results on performance and number of model parameters. We first compare our methods with two dependency structure based baselines. They yield lower performances than OURS. We also report the F1 score of the BERT model, which is finetuned on the source domain and directly applied to the target domain. BERT has the competitive performance with our method, but a big number of parameters (around 20 times larger than ours). We can also see that all features contribute to the final performance. In our proposed DGNN, the attention mechanism and directional message passing brings the gain of 0.48 and 0.63 in F1. The character-level embedding and the dependency relation can also improve the performance.

4. CONCLUSION

In this paper, we exploit dependency graphs of sentences to solve the data scarcity issue in unsupervised cross-domain NER, since sentences in source domain and target domain share common dependency structures. Specifically, we develop the Dependency-aware GNN to model the characteristics of dependency graph. Then these structural information can be adapted to target domain. Extensive experiments demonstrate the better performance of our method in target domain even though there is no entity label.

5. ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China (Grant No. 61876223), Youth Innovation Promotion Association, Chinese Academy of Sciences (No.2020163), National Natural Science Foundation of China (Grant No. 61832004), Projects of International Cooperation and Exchanges NSFC (Grant No. 62061136006).

6. REFERENCES

[1] Xuezhe Ma and Eduard Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," in *Proceedings of ACL*, 2016, pp. 1064–1074.

[2] Bill Yuchen Lin and Wei Lu, "Neural adaptation layers for cross-domain named entity recognition," in *Proceedings of EMNLP*, 2018, pp. 2012–2022.

[3] Joey Tianyi Zhou, Hao Zhang, Di Jin, Hongyuan Zhu, Meng Fang, Rick Siow Mong Goh, and Kenneth Kwok, "Dual adversarial neural transfer for low-resource named entity recognition," in *Proceedings of ACL*, 2019, pp. 3461–3471.

[4] Zihan Liu, Genta Indra Winata, and Pascale Fung, "Zero-resource cross-domain named entity recognition," in *Proceedings of RepL4NLP@ACL*, 2020.

[5] Chen Jia, Xiaobo Liang, and Yue Zhang, "Cross-domain NER using cross-domain language modeling," in *Proceedings of ACL*, 2019, pp. 2464–2474.

[6] Chen Jia and Yue Zhang, "Multi-cell compositional LSTM for NER domain adaptation," in *Proceedings of ACL*, 2020, pp. 5906–5917.

[7] Yaroslav Ganin and Victor S. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proceedings of ICML*, 2015, vol. 37, pp. 1180–1189.

[8] Darsh Shah, Raghav Gupta, Amir Fayazi, and Dilek Hakkani-Tur, "Robust zero-shot cross-domain slot filling with example values," in *Proceedings of ACL*, 2019.

[9] Yuhao Zhang, Peng Qi, and Christopher D. Manning, "Graph convolution over pruned dependency trees improves relation extraction," in *Proceedings of EMNLP*, 2018, pp. 2205–2215.

[10] Zhanming Jie and Wei Lu, "Dependency-guided LSTM-CRF for named entity recognition," in *Proceedings of EMNLP-IJCNLP*, 2019, pp. 3862–3872.

[11] Luchen Liu, Xixun Lin, Peng Zhang, and Bin Wang, "Improving cross-domain slot filling with common syntactic structure," in *ICASSP*, 2021, pp. 7638–7642.

[12] Kai Sheng Tai, Richard Socher, and Christopher D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *Proceedings of ACL-IJCNLP*, 2015, pp. 1556–1566.

[13] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio, "Graph attention networks," in *ICLR*, 2018.

[14] Thomas N. Kipf and Max Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[15] Qi Peng, Changmeng Zheng, Yi Cai, Tao Wang, Haoran Xie, and Qing Li, "Unsupervised cross-domain named entity recognition using entity-aware adversarial training," *Neural Networks*, 2020.

[16] Ankur Bapna, Gökhan Tür, Dilek Hakkani-Tür, and Larry P. Heck, "Towards zero-shot frame semantic parsing for domain scaling," in *Proceedings of Interspeech*, 2017, pp. 2476–2480.

[17] Hal Daumé III, "Cross-task knowledge-constrained self training," in *Proceedings of EMNLP*, 2008.

[18] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, pp. 59:1–59:35, 2016.

[19] Diego Marcheggiani and Ivan Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," in *Proceedings of EMNLP*, 2017.

[20] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer, "Neural architectures for named entity recognition," in *Proceedings of NAACL*, 2016, pp. 260–270.

[21] Xuezhe Ma and Eduard H. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," in *Proceedings of ACL*, 2016.

[22] Erik F. Tjong Kim Sang and Fien De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," in *Proceedings of HLT-NAACL*, 2003, pp. 142–147.

[23] Qi Zhang, Jinlan Fu, Xiaoyu Liu, and Xuanjing Huang, "Adaptive co-attention network for named entity recognition in tweets," in *Proceedings of AAAI*, 2018, vol. 32.

[24] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov, "Enriching word vectors with subword information," *TACL*, vol. 5, pp. 135–146, 2017.

[25] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, "Deep contextualized word representations," in *Proceedings of NAACL-HLT*, 2018.

[26] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, 2019, pp. 4171–4186.