Structured Refinement for Sequential Labeling

Yiran Wang[1]*, **Hiroyuki Shindo**[2], **Yuji Matsumoto**[3], **Taro Watanabe**[2]
[1]National Institute of Information and Communications Technology (NICT), Kyoto, Japan
[2]Nara Institute of Science and Technology (NAIST), Nara, Japan
[3]RIKEN Center for Advanced Intelligence Project (AIP), Tokyo, Japan
yiran.wang@nict.go.jp, shindo@is.naist.jp,
yuji.matsumoto@riken.jp, taro@is.naist.jp

Abstract

Filtering target-irrelevant information through hierarchically refining hidden states has been demonstrated to be effective for obtaining informative representations. However, previous work simply relies on locally normalized attention without considering possible labels at other time steps, the capacity for modeling long-term dependency relations is thus limited. In this paper, we propose to extend previous work with globally normalized attention, e.g., structured attention, to leverage structural information for more effective representation refinement. We also propose two implementation tricks to accelerate CRF computation and an initialization trick for Chinese character embeddings to further improve performance. We provide extensive experimental results on various datasets to show the effectiveness and efficiency of our proposed method.

1 Introduction

Sequential labeling tasks, e.g., named entity recognition (NER) and part-of-speech (POS) tagging, play an important role in natural language processing. Figure 1 shows two examples of sequential labeling tasks. Early studies focused on introducing rich features to improve performance. For example, to handle out-of-vocabulary words by introducing morphological features, Lample et al. (2016) and Ma and Hovy (2016) leveraged character-level features, whereas Heinzerling and Strube (2019) exploited subword-level features. Moreover, introducing long-term dependency features is also found to be beneficial for sequential labeling. Jie and Lu (2019) attempted to explicitly exploit dependency relations with additional annotations, while Zhang et al. (2018) and Chen et al. (2019) endeavored to learn these relations implicitly with more complex encoders.

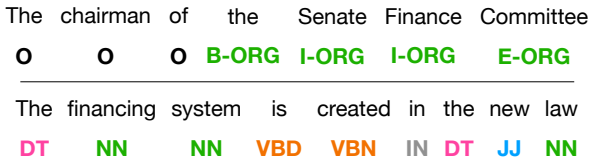

Figure 1: Examples of NER (top) and POS tagging (bottom). For NER, "the Senate Finance Committee" is a named entity of type ORG (organization). The prefixes `S-`, `I-`, or `E-` indicate this word is located at the beginning, intermediate, or ending of the current named entity, while `O` signifies this word is outside any named entity. In the case of POS tagging, each tag is a part-of-speech category. For instance, `NN` represents a singular noun and `VBN` is the past participle of a verb.

However, as Tishby and Zaslavsky (2015) pointed out, features are not created equal, only the target-relevant features are profitable for improving model performance. Recently, Cui and Zhang (2019) proposed a hierarchically-refined label attention network (LAN), which explicitly leverages label embeddings and captures long-term label dependency relations through multiple refinements layers.

Individually picking up the most likely label at each time step is undoubtedly critical, however, considering the entire historical progress is also indispensable. We find that the locally normalized attention, which Cui and Zhang (2019) used to leverage information from label embeddings, can eventually hurt performance. Since it only considers the current time step but ignores labels at other time steps, thus we presume its ability to capture long-term dependency relations is limited.

On the other hand, Kim et al. (2017) incorporated neural networks with probabilistic graphical models to obtain structural distributions as an alternative to conventional attention mechanisms. Their method relies on attending to cliques of linear-chain conditional random fields (CRF). These in-

*This work was done when the first author was at NAIST.

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 1873–1884
August 1–6, 2021.

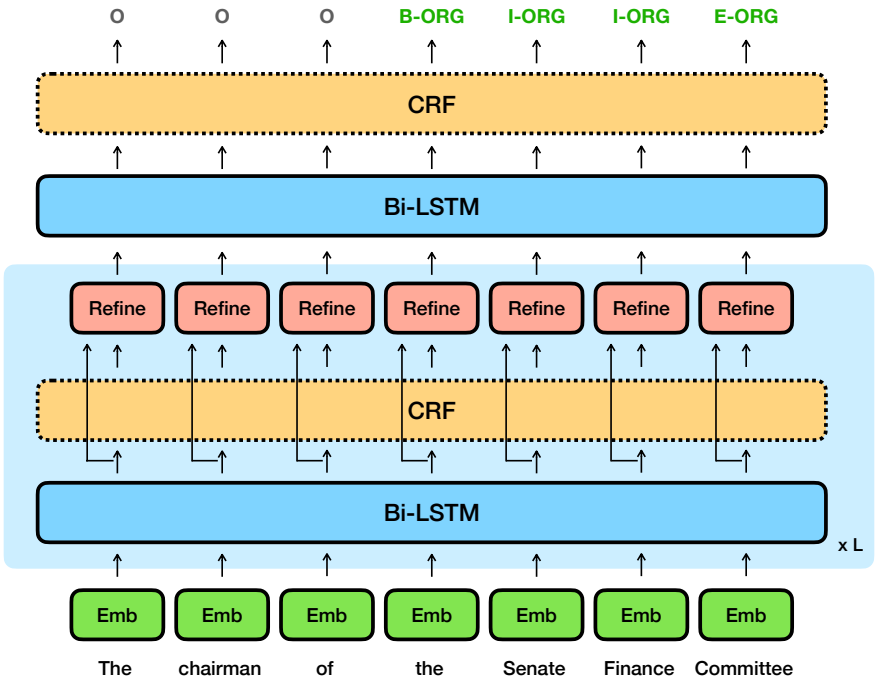

Figure 2: The architecture of the proposed model. The dotted lines mean these components are shared across layers.

ferred inner structures, i.e., represented as the marginal probabilities, are not the targets of their tasks but only serve as the latent variables, thus they do not impose direct supervision on these attention weights. In contrast, since we consider to repeatedly refine these inferred structures to obtain the final outputs, we compute structural attention over these target labels instead, without introducing unobserved variables.

In this paper, we propose a novel structured refinement mechanism by combining representation refinement and structured attention. Following and extending Cui and Zhang (2019), we hierarchically refine hidden representations with global normalized structured attention, i.e., the marginal probability of CRF. Besides, to impose direct supervision on the target structures, we share the label embeddings and the transition matrix of CRF across layers. Our method can be considered as iteratively re-constructing hidden representations with only label embeddings, and thus it is capable of filtering target-irrelevant information out.

Besides, we propose a character embedding initialization trick to enhance performance on Chinese datasets and two CRF implementation tricks to accelerate computation.

Our contributions are considered as four-folds, (a) we propose a novel structured refinement network by combing representation refinement and structured attention for sequential labeling tasks, (b) we propose an initialization trick for Chinese character embeddings, (c) we propose two implementation tricks to accelerate CRF training and decoding, (d) and we prove the effectiveness and efficiency of our model through extensive experiments for NER and POS tagging on various datasets.

2 Baseline

Formally speaking, given a token sequence $\{x_t\}_{t=1}^n$, the aim of sequential labeling tasks is to find the most probable label sequence $\{y_t\}_{t=1}^n$.

2.1 Label Attention Network

Label attention network (Cui and Zhang, 2019) consists of an embedding layer followed by several encoding and refinement layers alternatively. The decoding layer is a bidirectional LSTM followed by a refinement layer.

Embedding Layer Cui and Zhang (2019) employed the concatenation of word and character-based word representations as the token representations $\boldsymbol{x}_t = [\boldsymbol{w}_t, \boldsymbol{c}_t]$, they convert words to word embeddings $\boldsymbol{w}_t \in \mathbb{R}^{d_w}$ and use a character-level bidirectional LSTM to build character-based word embeddings $\boldsymbol{c}_t \in \mathbb{R}^{d_c}$.

Encoding Layer They utilized an independent bidirectional LSTM for each layer l as follows,

$$\{\boldsymbol{h}_t^{(l)}\}_{t=1}^n = \mathrm{LSTM}^{(l)}\left(\{\tilde{\boldsymbol{h}}_t^{(l-1)}\}_{t=1}^n\right) \quad (1)$$

where $\tilde{\boldsymbol{h}}_t^{(l-1)}$ is the refined representation from the last refinement layer. Specially, hidden vector $\tilde{\boldsymbol{h}}_t^{(0)}$ is the token representation $\boldsymbol{x}_t$. After this, they employ a refinement layer, which is called "label-attention inference sublayer" in the original paper, to refine hidden states. They make use of attention mechanism (Vaswani et al., 2017) to produce the attention matrix $\alpha_{t,j}^{(l)}$ as in Equation 2, and further calculate the label-aware hidden states $\hat{\boldsymbol{h}}_t^{(l)} \in \mathbb{R}^{d_h}$, which jointly encode information from the token representation subspace and the label representation subspace.

$$\alpha_{t,j}^{(l)} = \underset{j\in\{1,\ldots,m\}}{\mathrm{softmax}}\left(\frac{(\mathbf{Q}^{(l)}\boldsymbol{h}_t^{(l)})^\top(\mathbf{K}^{(l)}\boldsymbol{v}_{y_j})}{\sqrt{d_h}}\right) \quad (2)$$

$$\hat{\boldsymbol{h}}_t^{(l)} = \sum_{j=1}^{m}\alpha_{t,j}^{(l)}\cdot(\mathbf{V}^{(l)}\boldsymbol{v}_{y_j}) \quad (3)$$

Where $\mathbf{Q}^{(l)}, \mathbf{K}^{(l)}, \mathbf{V}^{(l)} \in \mathbb{R}^{d_h\times d_h}$ are all parameters, and $\boldsymbol{v}_{y_j} \in \mathbb{R}^{d_h}$ is the embedding of label $y_j \in \mathcal{Y}$. In practice, they use multiple heads to capture representations from multiple aspects in parallel. After that, they concatenate the hidden state $\boldsymbol{h}_t^{(l)}$ and the label-aware hidden state $\hat{\boldsymbol{h}}_t^{(l)}$ as the refined representation $\tilde{\boldsymbol{h}}_t^{(l)} \in \mathbb{R}^{2d_h}$, and feed it into the next encoding layer.

$$\tilde{\boldsymbol{h}}_t^{(l)} = [\boldsymbol{h}_t^{(l)}, \hat{\boldsymbol{h}}_t^{(l)}] \quad (4)$$

Decoding Layer Similar to the encoding layer, the decoding layer contains a bidirectional LSTM and a refinement layer, but at this layer, the attention matrix $\alpha_{t,j}^{(L+1)}$ only servers as the label probability distribution to predict the most probable label sequence.

$$p(\boldsymbol{y} \mid \boldsymbol{h}) = \prod_{t=1}^{n}\alpha_{t,y_t}^{(L+1)} \quad (5)$$

3 Proposed Method

3.1 Structured Refinement

A notable highlight of the model of Cui and Zhang (2019) is that it is not equipped with the commonly used CRFs (Lample et al., 2016; Ma and Hovy, 2016), however, it still can achieve remarkable performance. And just because of abandoning the computationally expensive CRFs, their model obtains a significant acceleration on both training and decoding stages. However, we find that the time step independent attention, i.e., the softmax operation in Equation 2, only considers these labels at the current time step and ignores all the possible label combinations at other time steps, thus the performance is eventually degraded since the ability of capturing long-term dependency relation is local and limited. We thus bring CRF back and use the marginal probability to construct refined representations. We claim replacing the attention matrix $\alpha_{t,j}^{(l)}$ with the globally normalized marginal probability can capture long-term dependency relations more effectively.

The potential function of CRF is defined as,

$$\phi(y_{t-1}, y_t, \boldsymbol{h}_t^{(l)}) = A_{y_{t-1},y_t} + \boldsymbol{h}_t^{(l)\top}\boldsymbol{v}_{y_t} \quad (6)$$

where $\mathbf{A} \in \mathbb{R}^{|\mathcal{Y}|\times|\mathcal{Y}|}$ is the transition matrix, A_{y_{t-1},y_t} denotes the transition score from label y_{t-1} to label y_t, and $\boldsymbol{v}_{y_t}$ is the embedding of label y_t. The conditional probability of a specified label sequence $\boldsymbol{y}$ can be described as

$$p(\boldsymbol{y} \mid \boldsymbol{h}^{(l)}) = \frac{1}{Z(\boldsymbol{h}^{(l)})}\exp\sum_{t=1}^{n}\phi(y_{t-1}, y_t, \boldsymbol{h}_t^{(l)}) \quad (7)$$

$$Z(\boldsymbol{h}^{(l)}) = \sum_{\boldsymbol{y}'\in\mathcal{Y}^n}\exp\sum_{t=1}^{n}\phi(y'_{t-1}, y'_t, \boldsymbol{h}_t^{(l)}) \quad (8)$$

where $Z(\boldsymbol{h}^{(l)})$ is the global normalization term, commonly known as the partition function. Furthermore, the marginal probability is defined as follow.

$$\mu_t(y_j, \boldsymbol{h}^{(l)}) = \sum_{\boldsymbol{y}':Y'_t=y_j} p(\boldsymbol{y}' \mid \boldsymbol{h}^{(l)}) \quad (9)$$

Marginal probability stands for the sum of the probabilities of all possible label sequences that emit label y_j at time step t. Calculating marginal probability requires enumerating all possible structures, and it thus can be called globally normalized probability or structured attention.

We replace the locally normalized attention $\alpha_{t,j}^{(l)}$ in Equation 3 with our globally normalized one, i.e., $\mu_t(y_j, \boldsymbol{h}^{(l)})$. Furthermore, we employ residual connection (He et al., 2016) and layer normalization (Ba et al., 2016), instead of concatenation, to

construct the refined representation $\tilde{\boldsymbol{h}}_t^{(l)} \in \mathbb{R}^{d_h}$,

$$\hat{\boldsymbol{h}}_t^{(l)} = \sum_{j=1}^{m} \mu_t\left(y_j, \boldsymbol{h}^{(l)}\right) \cdot \left(\mathbf{V}^{(l)} \boldsymbol{v}_{y_j}\right) \quad (10)$$

$$\tilde{\boldsymbol{h}}_t^{(l)} = \text{LayerNorm}\left(\boldsymbol{h}_t^{(l)} + \max\left(\mathbf{0}, \hat{\boldsymbol{h}}_t^{(l)}\right)\right) \quad (11)$$

where $\mathbf{V}^{(l)} \in \mathbb{R}^{d_h \times d_h}$ is a matrix parameter. The obtained refined representation $\tilde{\boldsymbol{h}}_t^{(l)}$ is then fed into the next layer.

3.2 Computing Marginal probability

Conventional method to compute the marginal probability $\mu_t\left(y_j, \boldsymbol{h}^{(l)}\right)$ requires running the forward-backward algorithm. Fortunately, as Eisner (2016) indicates, merely computing the log-partition function, $\log Z\left(\boldsymbol{h}^{(l)}\right)$, and differentiating it with an automatic differentiation library yields equivalent marginal probability efficiently. Thus, we use the `torch.autograd.grad` function of `PyTorch` to compute the marginal probability as follow.

$$\mu_t\left(y_j, \boldsymbol{h}^{(l)}\right) = \frac{\partial \log Z\left(\boldsymbol{h}^{(l)}\right)}{\partial\left(\boldsymbol{h}_t^{(l)\top} \boldsymbol{y}_j\right)} \quad (12)$$

3.3 Training and Decoding

We train our model by maximizing the log-likelihood with the back-propagation algorithm. The objective function is defined as follow,

$$\mathcal{L} = -\log\ p\left(\boldsymbol{y} \mid \boldsymbol{h}^{(L+1)}\right) \quad (13)$$

We apply the Viterbi algorithm (Forney, 1973) to efficiently search for the most probable label sequences on the decoding stage.

$$\hat{\boldsymbol{y}} = \underset{\boldsymbol{y}' \in \mathcal{Y}^n}{\arg\max}\, p\left(\boldsymbol{y}' \mid \boldsymbol{h}^{(L+1)}\right) \quad (14)$$

3.4 Complexity and Implementation Tricks

One concern regarding our proposed method is its computational complexity, as it requires to compute not only the partition function but also the marginal probability. Calculating the partition function, as in Equation 8, is the well-known bottleneck of CRF computation. And this is commonly achieved through reducing potential matrices by applying matrix multiplications. Similar to Rush (2020), we make use of the associative property of matrix multiplication to accelerate computation. The product of multiplying matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ is equivalent to the product of $\mathbf{AB}$ and $\mathbf{CD}$. Leveraging the power of GPU to compute $\mathbf{AB}$ and $\mathbf{CD}$ in parallel, and recursively applying this trick, we can reduce the time complexity of obtaining the partition function from $\mathcal{O}\left(\sum_{i=1}^{|\mathcal{B}|} |\boldsymbol{x}|_i\right)$ to $\mathcal{O}\left(\sum_{i=1}^{|\mathcal{B}|} \log |\boldsymbol{x}|_i\right)$, where $|\boldsymbol{x}|_i$ is the length of i-th sentence in batch $\mathcal{B}$. Moreover, instead of padding the sequence length $|\boldsymbol{x}_i|$ out to the nearest power of two as Rush (2020) does, we pre-compile argument indices of the matrix multiplication to handle the variant sentence length issue in a batch. Our method can effectively avoid out-of-memory error since we don't waste memory for paddings. This pre-compiling trick can further reduce the time complexity to $\mathcal{O}\left(\max_i \log |\boldsymbol{x}|_i\right)$. We release our CRF implementation with these two tricks as an independent library[1] for future study and use.

3.5 Character Embeddings Initialization

We describe a trick for Chinese character embeddings initialization. The most striking difference between Chinese and English is that the minimal semantic units, i.e., sememes, of Chinese are characters instead of words or subwords. The character vocabulary size of Chinese, e.g., around 2,000 on the OntoNote 5.0 dataset, is markedly larger than English, e.g., around 100 on the OntoNotes 5.0 English dataset. Existing models (Zhang and Yang, 2018; Li et al., 2020a) generally focused on introducing additional pre-trained character embeddings on the top of lexicon embeddings, and attempted to selectively leverage information from both of them according to the different word segmentation schemes. However, we notice that most of these characters already exist in the word vocabulary as single-character words, thus we employ a randomly initialized orthogonal matrix[2] to project the pre-trained word embeddings into the same dimension as the character embeddings, and use these projected embeddings for initialization.

4 Experiments

4.1 Datasets

We conduct experiments on the CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) and the OntoNotes 5.0 (Weischedel et al., 2013) datasets for English NER, and on the OntoNotes 5.0 and the OntoNotes 4.0 datasets for Chinese NER experiments. We also conduct experiments on the Wall

[1] https://github.com/speedcell4/torchlatent

[2] `torch.nn.init.orthogonal_`

Task	Dataset	Language	Sentences	$\lvert\mathcal{Y}\rvert$
NER	OntoNotes 5.0	English	59,924 / 8,528 / 8,262	73
	CoNLL 2003	English	14,987 / 3,466 / 3,684	17
	OntoNotes 5.0	Chinese	36,487 / 6,083 / 4,472	73
	OntoNotes 4.0	Chinese	15,724 / 4,301 / 4,346	17
POS	WSJ	English	38,219 / 5,527 / 5,462	45
	UD 2.2	English	12,544 / 2,003 / 2,078	50

Table 1: Dataset statistics, where the "Sentences" column displays the number of sentences in train/dev/test split respectively, the $|\mathcal{Y}|$ column displays the number of target label types. For NER datasets, we count types with the IOBES labeling scheme.

Street Journal (WSJ) dataset (Marcus et al., 1993) and the Universal Dependencies (UD) v2.2 English dataset for POS tagging experiments.

The only data pre-processing that we have performed is replacing digital tokens with a special token. And we convert labels to the IOBES labeling scheme (Ramshaw and Marcus, 1995; Ratinov and Roth, 2009) on NER datasets. The dataset statistics are provided in Table 1.

4.2 Hyper-parameter Settings

Following Cui and Zhang (2019) and Jie and Lu (2019), 100-dimensional Glove (Pennington et al., 2014) word embeddings are utilized for all the English experiments, and 300-dimensional FastText (Mikolov et al., 2018) word embeddings are employed for Chinese experiments. The dimension of character embeddings is 30, and the hidden states dimension d_c of the character bidirectional LSTM is 100, i.e., 50 in each direction. We apply dropout (Srivastava et al., 2014) on token representations with a rate of 0.5.

For encoding and refinement layers, the dimension of the hidden state d_h of bidirectional LSTMs is 600, i.e., 300 in each direction. We apply dropout on hidden states $\boldsymbol{h}_t^{(l)}$ with a rate of 0.5 before feeding into refinement layers. The number of refinement layers L is just 1.

We optimize our model by applying stochastic gradient descent (SGD) with decaying learning rate $\eta_\tau = \eta_0/(1 + 0.075 \cdot \tau)$, where τ is the index of the current epoch, and the initial learning rate η_0 for Chinese experiments without contextual word representations is 0.05, and for all the other experiments we use 0.1. The weight decay rate is 10^{-8}, the momentum is 0.15, the batch size is 10, the number of epochs is 100, and gradients exceed 5 will be clipped.

In addition, since the pre-trained contextualized word embeddings technique is widely accepted as a new fundamental utility of natural language processing, we also conduct experiments with ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019). In these settings, tokens are represented as $\boldsymbol{x}_t = [\boldsymbol{w}_t, \boldsymbol{c}_t, \boldsymbol{e}_t]$, where $\boldsymbol{e}_t$ is the contextual word representation.

ELMo vectors are obtained by averaging output vectors over all layers of ELMo. For English experiments, we use the `original` checkpoint, and use the checkpoints provided by Che et al. (2018) for Chinese experiments.

BERT representations are the averages all BERT subword embeddings in the last four layers. Following Li et al. (2020b) and Li et al. (2020a), we utilize `bert-large-cased` and `hfl/chinese-bert-wwm` checkpoints for English and Chinese experiments respectively.

4.3 Evaluation

NER experiments are evaluated by using F_1 scores, and POS tagging experiments are evaluated with accuracy scores. All of our experiments were run 4 times with different random seeds, and the averaged scores are reported in the following tables.

Our models[3] are implemented with deep learning framework `PyTorch` (Paszke et al., 2019) and we ran experiments on `GeForce GTX 1080Ti` with 11 GB memory.

4.4 Experimental Results

4.4.1 Named Entity Recognition

Table 2 compares the performance of our proposed method and baselines on the OntoNotes 5.0 English dataset. Our model significantly outperforms Cui and Zhang (2019) and Jie and Lu (2019) by 0.49 and 0.13 F_1 scores respectively. These results demonstrate that our model can filter irrelevant information more effectively than Cui and Zhang (2019). Notably, the model of Jie and Lu (2019) relies on external dependency annotations, whereas our model requires no external knowledge[4]. In the case of employing ELMo, our model outperforms Jie and Lu (2019) by 0.11 F_1 score.

On the CoNLL 2003 English dataset, our model performs worse than these baseline models, but, with ELMo, it outperforms Jie and Lu (2019) and

[3] https://github.com/speedcell4/refiner

[4] In this paper, we use "external knowledge" to denote any additional resources other than word embeddings and contextual word representations.

Model	EK	P	R	F_1
Chiu and Nichols (2016)	✓	86.04	86.53	86.28
Strubell et al. (2017)	-	-	-	86.84
Li et al. (2017)	✓	88.00	86.50	87.21
Ghaddar and Langlais (2018)	-	-	-	87.95
Fisher and Vlachos (2019)	-	-	-	87.59
Cui and Zhang (2019)	-	-	-	88.16
Yan et al. (2019)	-	-	-	88.43
Jie and Lu (2019)	✓	88.53	88.50	88.52
Our Method	-	88.71	88.60	**88.65**
Yan et al. (2019) [E]	-	-	-	89.78
Jie and Lu (2019) [E]	✓	89.59	90.17	89.88
Our Method [E]	-	89.51	90.48	**89.99**
Devlin et al. (2019) [B]	-	90.01	88.35	89.16
Fisher and Vlachos (2019) [B]	-	-	-	89.71
Li et al. (2020b) [B]	-	92.98	89.95	91.11
Yu et al. (2020) [B]	-	91.1	91.5	**91.3**
Our Method [B]	-	90.00	91.17	90.93

Table 2: Experimental results on the OntoNotes 5.0 English dataset. Checkmark ✓ in the "EK" column indicates that external knowledge is utilized in that model. [E] and [B] stands for ELMo and BERT respectively. **Bold** and underlined numbers indicate the best and the second-best results respectively.

Model	EK	P	R	F_1
Huang et al. (2015)	✓	-	-	88.83
Lample et al. (2016)	-	-	-	90.94
Ma and Hovy (2016)	-	-	-	91.21
Zhang et al. (2018)	-	-	-	91.57
Chiu and Nichols (2016)	✓	-	-	91.62
Liu et al. (2019a)	-	-	-	91.80
Yan et al. (2019)	-	-	-	91.33
Liu et al. (2019b)	✓	-	-	**91.96**
Our Method	-	90.70	90.81	90.76
Jie and Lu (2019) [E]	✓	-	-	92.40
Yan et al. (2019) [E]	-	-	-	92.62
Our Method [E]	-	92.60	93.19	**92.89**
Devlin et al. (2019) [B]	-	-	-	92.8
Li et al. (2020b) [B]	-	92.33	94.61	93.04
Yu et al. (2020) [B]	-	93.7	93.3	**93.5**
Our Method [B]	-	92.66	92.98	93.23

Table 3: Experimental results on the CoNLL 2003 English dataset.

Yan et al. (2019) by 0.49 and 0.27 F_1 score. Our hypothesis is that the CoNLL 2003 dataset contains much fewer examples and entity categories, thus the label dependency relations are not as important as on the OntoNotes 5.0 English dataset, thus our method could bring about limited improvement.

A similar phenomenon can be noticed on the OntoNotes 4.0 Chinese dataset, as in Table 4, our model is inferior to Li et al. (2020a), but on the contextual word representations experiment setting, our model significantly outperforms them by 1.41 F_1 score with BERT. Moreover, on the OntoNotes 5.0 Chinese dataset, our model constantly outperforms the best previous work (Jie and Lu, 2019) by 0.65 F_1 score without utilizing external knowledge.

Besides, we can notice initializing character embeddings with our trick remarkably improves model performance by 0.76 F_1 score on the OntoNotes 4.0 Chinese dataset, even this improvement reduces to only 0.00 and 0.20 F_1 scores on ELMo and BERT experiments. We hypothesize that contextual word representation already provides rich enough morphological information, thus careful character embeddings initialization can only bring little benefit. On the OntoNotes 5.0 Chi-

Model	EK	P	R	F_1
Zhang and Yang (2018)	✓	76.35	71.56	73.88
Mengge et al. (2019)	✓	76.78	72.54	74.60
Gui et al. (2019a)	✓	76.40	72.60	74.45
Gui et al. (2019b)	✓	76.13	73.68	74.89
Yan et al. (2019)	✓	-	-	72.43
Li et al. (2020a)	✓	-	-	**76.45**
Our Method	-	75.28	72.39	73.80
Our Method (init)	-	75.49	73.69	74.56
Our Method [E]	-	80.21	78.50	79.34
Our Method (init) [E]	-	79.75	78.94	**79.34**
Devlin et al. (2019) [B]	-	78.01	80.35	79.16
Zhang and Yang (2018) [B]	✓	79.79	79.41	79.60
Gui et al. (2019a) [B]	✓	79.41	80.32	79.86
Mengge et al. (2019) [B]	✓	79.62	81.82	80.60
Li et al. (2020a) [B]	✓	-	-	81.82
Li et al. (2020b) [B]	-	82.98	81.25	82.11
Our Method [B]	-	81.80	84.31	83.03
Our Method (init) [B]	-	81.73	84.79	**83.23**

Table 4: Experimental results on the OntoNotes 4.0 Chinese dataset. "init" stands for utilizing projected FastText embeddings to initialize the character embeddings.

Model	EK	P	R	F_1
Pradhan et al. (2013)	-	78.20	66.45	71.85
Zhang and Yang (2018)	✓	76.34	77.01	76.67
Jie and Lu (2019)	✓	77.40	77.41	77.40
Our Method	-	77.09	77.50	77.29
Our Method (init)	-	77.99	78.11	**78.05**
Jie and Lu (2019) [E]	✓	78.86	81.00	79.92
Our Method [E]	-	79.75	79.83	79.78
Our Method (init) [E]	-	79.49	80.32	**79.92**
Our Method [B]	-	79.61	82.47	81.01
Our Method (init) [B]	-	79.66	82.45	**81.03**

Table 5: Experimental results on the OntoNotes 5.0 Chinese dataset.

Model	EK	WSJ	UD v2.2
Huang et al. (2015)	✓	97.55	-
Ma and Hovy (2016)	-	97.55	-
Zhang et al. (2018)	-	97.55	-
Yasunaga et al. (2018)	-	97.58	95.41
Xin et al. (2018)	-	97.58	-
Cui and Zhang (2019)	-	97.58	95.59
Our Method	-	**97.64**	**95.62**
Our Method [E]	-	**97.83**	**96.86**
Our Method [B]	-	**97.85**	**97.15**

Table 6: Experimental results on the WSJ and the UD v2.2 datasets.

$\{\boldsymbol{y}_j\}_{j=1}^m$	$\lvert\mathcal{Y}\rvert$	$\mathbf{A}$	$\lvert\theta\rvert$	$\mathbf{F}_1$
separated	10	separated	8,212,505	88.51
separated	20	separated	8,218,825	88.55
separated	50	separated	8,238,985	88.62
separated	73	separated	8,255,660	88.48
separated	100	separated	8,276,585	88.58
shared	73	separated	8,211,860	88.41
shared	73	shared	8,206,385	**88.65**

Table 7: Influence of weight tying, where $\{\boldsymbol{y}_j\}_{j=1}^m$ stands for whether share label embeddings across layers, $|\mathcal{Y}|$ denotes the number of labels, $\mathbf{A}$ is the CRF transition matrix in Equation 6, and $|\theta|$ is the number of parameters.

nese dataset, the performance improvements are 0.76, 0.16, and 0.02 F_1 scores, respectively.

4.4.2 Part-of-speech Tagging

Table 6 shows the experimental results on the Wall Street Journal English and the Universal Dependencies v2.2 English dataset respectively. Although Cui and Zhang (2019) claimed that the simple Markov label transition model of CRF can barely bring information gain over bidirectional LSTM, we observe 0.06 and 0.03 gain in accuracy scores. Besides, our model achieves 97.83 and 96.86 accuracy scores with ELMo, and further improves the performance to 97.85 and 97.15 accuracy scores with BERT.

4.5 Discussion

Influence of Weight Tying The major difference between our method and Kim et al. (2017) is that we use only observed labels, while they employ unobserved labels as latent variables. In the actual implementation, this difference is reflected in whether to share label embeddings and the transition matrix of CRF across layers. Intuitively, completely relying on unobserved variables would implicitly performing clustering on latent representation space, and it might introduce noise. Besides, the state transitions in a different layer may obey different dynamics. Thus sharing the transition matrix across layers might have an impact on performance.

We conducted experiments on the OntoNotes 5.0 English dataset to compare the performance of all the above-mentioned settings, as reported in Table 7. Notably, our model, with both the label embeddings $\{\boldsymbol{v}_{y_j}\}_{j=1}^m$ and the transition matrix $\mathbf{A}$ shared, surpasses all separated models. These results support our claim that tying the weights of embeddings and the label transition matrix can indeed leverage annotation information and thus is better than completely relying on unobserved variables. Besides, we did not notice significant performance changes when varying the number of labels $|\mathcal{Y}|$. Furthermore, the number of parameters of our shared model is, in fact, the smallest one, even compared to LAN (about 10.0 million parameters).

Influence of the Connection Mechanism A minor difference between our method and Cui and Zhang (2019) is that we utilize residual connection (He et al., 2016) and layer normalization (Xu et al., 2019), as in Equation 11, while Cui and Zhang (2019) only use concatenation, as in Equation 4. Table 8 shows the comparison on the OntoNotes 5.0 English dataset, measuring the influence of these two connection mechanisms. We find that the residual connection works better than the concatenation connection, that might because the residual connection can make training more smoothly by preventing the chaotic loss surface (Li et al., 2018).

Connection	$\mathbf{F}_1$
Concatenation	88.54
Residual	**88.65**

Table 8: Influence of the connection mechanism.

Influence of Parameter Size As in Table 9, we did not observe performance increase along with the increasing of the number of refinement layers. Therefore, we claim that one refinement layer is enough for our model, while Cui and Zhang (2019) needs two refinement layers. Our hypothesis is that the long-term dependency modeling capacity of the

L	d_h	$\|\theta\|$	$\mathbf{F}_1$
1	400	6,227,985	88.28
	600	8,206,385	**88.65**
	800	10,904,785	88.64
2	400	7,352,385	88.51
	600	10,732,985	88.36
	800	15,393,585	88.49

Table 9: Influence of parameter size, where L is the number of refinement layers, d_h is the hidden state dimension of the bidirectional LSTM, and $|\theta|$ represents the number of parameters.

first-order CRF is relatively limited, and we remain the use of the higher-order CRF as future work.

Training and Decoding Speeds We report the training and decoding speeds on the OntoNotes 5.0 English dataset. We demonstrate the efficiency of our CRF implementation tricks by comparing it with a widely used library, `pytorch-crf`[5]. According to Table 10, our CRF implementation tricks remarkably accelerate both training and decoding. In particular, with our CRF implementation, our computation extensive model even achieves a greater training speed than BiLSTM-CRF with `pytorch-crf`. Therefore, we claim that the efficiency of our model is acceptable.

Model	CRF	Training	Decoding
BiLSTM-CRF	`pytorch-crf`	82.25	480.08
	ours	**205.96**	**850.08**
Our Model	`pytorch-crf`	45.06	219.48
	ours	**93.04**	**296.91**

Table 10: Training and decoding speeds on the OntoNotes 5.0 English dataset. The "training" and "decoding" columns indicate the numbers of sentences our model can process per second on average.

5 Related Work

Early-stage research of NER and POS tagging focused on introducing rich features, for example, Yang et al. (2016) conducted experiments on the influence of discrete manual features, Chiu and Nichols (2016); Ma and Hovy (2016) introduced morphological features by employing a convolution network to encode character-level features, while Lample et al. (2016) chose bidirectional LSTM.

[5] https://github.com/kmkurn/pytorch-crf

Some other research aimed at leveraging syntactic information, Li et al. (2017) and Jie and Lu (2019) proposed to run external parsers first and directly encode this syntactic information. Other work attempted to infer dependency relations among words implicitly, such that, Strubell et al. (2017) introduced iterative dilated convolution networks as an alternative to BiLSTM, and Zhang et al. (2018) and Liu et al. (2019b) designed encoders which maintain and update global representations along with local token representations.

Recently, Li et al. (2020b) unified flat and nested NER by formulating them as a machine reading comprehension task. Yu et al. (2020) proposed to enumerate all possible spans and to utilize a biaffine classifier to assign category labels to them.

Besides, the widespread use of contextual word representations, e.g., ELMo (Peters et al., 2018), Flair (Akbik et al., 2018), and BERT (Devlin et al., 2019), greatly improves the performance of NER models and they are accepted as new fundamental techniques of natural language processing.

Intuitively speaking, the refinement mechanism provides the models with additional chances to revise previous decisions. In existing work, this method was successfully applied to various tasks, e.g., text classification (Yu et al., 2017), sequential labeling (Cui and Zhang, 2019; Lyu et al., 2019), machine translation (Lee et al., 2018), and question answering (Nema et al., 2019). Our work is not the first attempt of introducing refinement mechanism to sequential labeling tasks. Cui and Zhang (2019) relied on locally normalized attention to softly refine hidden representations layer by layer, while Liu et al. (2019a) chose to discretely filter out target-irrelevant semantic aspects and thus could be considered as a hard refinement mechanism.

6 Conclusion

Motivated by the structured attention, we enhanced the previous refinement mechanism by replacing the locally normalized attention with our globally normalized attention. Experimental results on various tasks and datasets demonstrate that structured refinement is capable of filtering out target-irrelevant information through capturing long-term dependency relations. Besides, we remarkably accelerated training and decoding through two implementation tricks for CRF, and obtained further model performance improvement with an initialization trick for Chinese character embeddings. We

remain to employ the higher-order CRF as future work.

Acknowledgements

This work was partly supported by JST CREST Grant Number JPMJCR1513. The authors would like to thank the anonymous reviewers for their instructive comments.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium. Association for Computational Linguistics.

Hui Chen, Zijia Lin, Guiguang Ding, Jianguang Lou, Yusen Zhang, and Borje Karlsson. 2019. GRN: Gated relation network to enhance convolutional neural network for named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6236–6243.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Leyang Cui and Yue Zhang. 2019. Hierarchically-refined label attention network for sequence labeling. 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)*, pages 4115–4128, Hong Kong, China. Association for Computational Linguistics.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jason Eisner. 2016. Inside-outside and forward-backward algorithms are just backprop (tutorial paper). In *Proceedings of the Workshop on Structured Prediction for NLP*, pages 1–17, Austin, TX. Association for Computational Linguistics.

Joseph Fisher and Andreas Vlachos. 2019. Merge and label: A novel neural network architecture for nested NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5840–5850, Florence, Italy. Association for Computational Linguistics.

G David Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Abbas Ghaddar and Phillippe Langlais. 2018. Robust lexical features for improved neural network named-entity recognition. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1896–1907, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Tao Gui, Ruotian Ma, Qi Zhang, Lujun Zhao, Yu-Gang Jiang, and Xuanjing Huang. 2019a. Cnn-based chinese ner with lexicon rethinking. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4982–4988. International Joint Conferences on Artificial Intelligence Organization.

Tao Gui, Yicheng Zou, Qi Zhang, Minlong Peng, Jinlan Fu, Zhongyu Wei, and Xuanjing Huang. 2019b. A lexicon-based graph neural network for Chinese NER. 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)*, pages 1040–1050, Hong Kong, China. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Benjamin Heinzerling and Michael Strube. 2019. Sequence tagging with contextual and non-contextual subword representations: A multilingual evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 273–291, Florence, Italy. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

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)*, pages 3862–3872, Hong Kong, China. Association for Computational Linguistics.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. *ArXiv*, abs/1702.00887.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the loss landscape of neural nets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 6389–6399. Curran Associates, Inc.

Peng-Hsuan Li, Ruo-Ping Dong, Yu-Siang Wang, Ju-Chieh Chou, and Wei-Yun Ma. 2017. Leveraging linguistic structures for named entity recognition with bidirectional recursive neural networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2664–2669, Copenhagen, Denmark. Association for Computational Linguistics.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020a. FLAT: Chinese NER using flat-lattice transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6836–6842, Online. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020b. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Kun Liu, Shen Li, Daqi Zheng, Zhengdong Lu, Sheng Gao, and Si Li. 2019a. A prism module for semantic disentanglement in name entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5358–5362, Florence, Italy. Association for Computational Linguistics.

Yijin 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 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2431–2441, Florence, Italy. Association for Computational Linguistics.

Chunchuan Lyu, Shay B. Cohen, and Ivan Titov. 2019. Semantic role labeling with iterative structure refinement. 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)*, pages 1071–1082, Hong Kong, China. Association for Computational Linguistics.

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 (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Xue Mengge, Yu Bowen, Liu Tingwen, Wang Bin, Meng Erli, and Li Quangang. 2019. Porous lattice-based transformer encoder for chinese ner.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Preksha Nema, Akash Kumar Mohankumar, Mitesh M. Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. 2019. Let's ask again: Refine network for automatic question generation. 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)*, pages 3314–3323, Hong Kong, China. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference*

on Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

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, Sofia, Bulgaria. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Alexander Rush. 2020. Torch-struct: Deep structured prediction library. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 335–342, Online. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2670–2680, Copenhagen, Denmark. Association for Computational Linguistics.

Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE.

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 2003*, pages 142–147.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

R Weischedel, M Palmer, M Marcus, E Hovy, S Pradhan, L Ramshaw, N Xue, A Taylor, J Kaufman, M Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. linguistic data consortium, philadelphia, pa(2013).

Yingwei Xin, Ethan Hart, Vibhuti Mahajan, and Jean-David Ruvini. 2018. Learning better internal structure of words for sequence labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2584–2593, Brussels, Belgium. Association for Computational Linguistics.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. 2019. Understanding and improving layer normalization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 4381–4391. Curran Associates, Inc.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. Tener: Adapting transformer encoder for named entity recognition.

Jie Yang, Zhiyang Teng, Meishan Zhang, and Yue Zhang. 2016. Combining discrete and neural features for sequence labeling. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 140–154. Springer.

Michihiro Yasunaga, Jungo Kasai, and Dragomir Radev. 2018. Robust multilingual part-of-speech tagging via adversarial training. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 976–986, New Orleans, Louisiana. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

Liang-Chih Yu, Jin Wang, K. Robert Lai, and Xuejie Zhang. 2017. Refining word embeddings for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 534–539, Copenhagen, Denmark. Association for Computational Linguistics.

Yue Zhang, Qi Liu, and Linfeng Song. 2018. Sentence-state LSTM for text representation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 317–327, Melbourne, Australia. Association for Computational Linguistics.

Yue Zhang and Jie Yang. 2018. Chinese NER using lattice LSTM. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1564, Melbourne, Australia. Association for Computational Linguistics.