Improving Low-Resource Named Entity Recognition using Joint Sentence and Token Labeling

Canasai Kruengkrai Thien Hai Nguyen Sharifah Mahani Aljunied Lidong Bing
DAMO Academy, Alibaba Group
canasai@gmail.com,
{thienhai.nguyen,mahani.aljunied,l.bing}@alibaba-inc.com

Abstract

Exploiting sentence-level labels, which are easy to obtain, is one of the plausible methods to improve low-resource named entity recognition (NER), where token-level labels are costly to annotate. Current models for jointly learning sentence and token labeling are limited to binary classification. We present a joint model that supports multi-class classification and introduce a simple variant of self-attention that allows the model to learn scaling factors. Our model produces 3.78%, 4.20%, 2.08% improvements in F1 over the BiLSTM-CRF baseline on e-commerce product titles in three different low-resource languages: Vietnamese, Thai, and Indonesian, respectively.

1 Introduction

Neural named entity recognition (NER) has become a mainstream approach due to its superior performance (Huang et al., 2015; Lample et al., 2016; Ma and Hovy, 2016; Chiu and Nichols, 2016; Akbik et al., 2018). However, neural NER typically requires a large amount of manually labeled training data, which are not always available in low-resource languages. Training neural NER with limited labeled data can be very challenging. In this paper, we consider bridging multi-task learning (MTL) (Caruana, 1993; Ruder, 2017) and pre-training (Peters et al., 2018; Devlin et al., 2019) to leverage training signals of an auxiliary task that has a sufficiently large number of labeled data.

Researchers have investigated a wide variety of auxiliary tasks and resources to boost the performance of neural NER, e.g., training coarse-grained NER (Aguilar et al., 2017), fine-tuning bilingual word embeddings (Wang et al., 2017), applying language models (Rei, 2017), integrating part-of-speech (POS) tagging (Lin et al., 2018), using cross-lingual knowledge (Feng et al., 2018), and learning paraphrases (Watanabe et al., 2019).

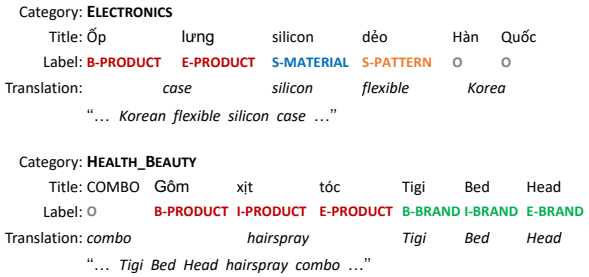

Figure 1: Examples of product titles with NER annotation in Vietnamese. Product categories are provided by sellers and can be used as sentence-level labels.

While most of the previous studies have exploited token-level information from auxiliary tasks, a few of them have tried to use sentence-level information (Rei and Søgaard, 2018; Devlin et al., 2019). Our work is closely related to the joint labeling framework in Rei and Søgaard (2019). However, they only focused on binary classification, while we attempt to handle multi-class classification on both sentence and token levels.

In this work, we focus on improving low-resource NER by exploiting large data, only having sentence-level labels. Figure 1 shows examples of product titles on an e-commerce website in Vietnamese. While the product titles with NER annotation done by our annotators are limited, those with product categories (e.g., ELECTRONICS) labeled by sellers are abundant, which can be used to train a sentence-level classifier.[1] A key challenge is to pass useful training signals from the sentence-level classification to the token-level NER.

Our contributions are as follows. We present the joint sentence and token labeling framework that enables multi-class classification equipped with a pre-training strategy (§2.1). We show that the current attention mechanisms can produce suboptimal

[1]The sellers are required to assign a category when uploading the product, but such input could be noisy as well.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 5898–5905

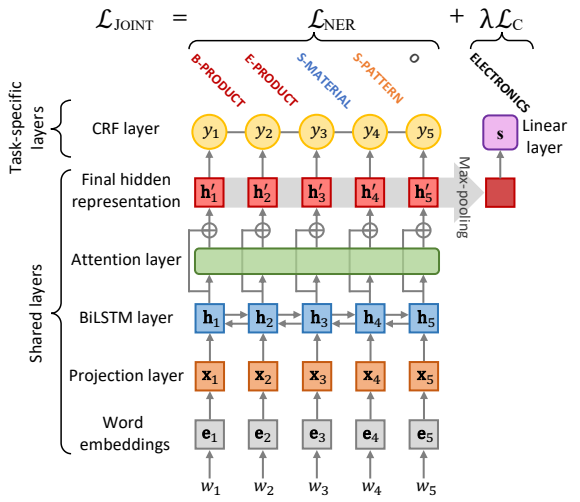

Figure 2: Architecture of our joint sentence and token labeling model. The attention layer is optional, which can be skipped or replaced with the desired approach.

results and propose a simple approach that allows the model to learn scaling factors to obtain a proper attention distribution (§2.2). Results on product title texts indicate that the proposed method is effective for low-resource NER across three different languages: Vietnamese, Thai, and Indonesian.

2 Proposed method

Figure 2 shows the architecture of our joint sentence and token labeling model. Our model is based on hard parameter sharing (Ruder, 2017) in which the hidden layers are shared between two tasks. The task-specific layers include a conditional random field (CRF) layer for NER and a linear layer for sentence classification.[2]

Unlike the standard MTL, which trains multiple tasks at once and expects the model to perform well on all tasks (Hashimoto et al., 2017; Rei and Søgaard, 2019), the goal of our work is to improve the performance of the main task (NER) using the auxiliary task (sentence classification) for creating pre-trained representations and as a regularizer.

2.1 Joint learning framework for multi-class classification

Shared layers Let $w_1, \ldots, w_T$ be an input token sequence, where w_t denotes the t-th token in the sequence. We represent each w_t using a pre-trained word embedding $\mathbf{e}_t \in \mathbb{R}^{d_e}$, where d_e is the dimensionality of word embeddings. We do not fine-tune word embeddings but project them into a new space using $\mathbf{x}_t = \mathbf{W}_1\mathbf{e}_t$, where $\mathbf{W}_1 \in \mathbb{R}^{d_e \times d_e}$ is a trainable weight matrix. We then feed the projected embedding sequence $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_T] \in \mathbb{R}^{T \times d_e}$ to a bidirectional long short-term memory (BiLSTM) layer to obtain a forward hidden state sequence $\overrightarrow{\mathbf{H}} = [\overrightarrow{\mathbf{h}}_1, \ldots, \overrightarrow{\mathbf{h}}_T] \in \mathbb{R}^{T \times \frac{d_h}{2}}$ and a backward hidden state sequence $\overleftarrow{\mathbf{H}} = [\overleftarrow{\mathbf{h}}_1, \ldots, \overleftarrow{\mathbf{h}}_T] \in \mathbb{R}^{T \times \frac{d_h}{2}}$, where d_h is the number of hidden units.

We concatenate the hidden states of both directions to obtain the final hidden representation $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_T] \in \mathbb{R}^{T \times d_h}$, where $\mathbf{h}_t = \text{concat}(\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t) \in \mathbb{R}^{d_h}$. We can either use $\mathbf{H}$ for both the sentence classification and NER tasks directly or apply an attention mechanism on it to help the model focus on particular tokens (detailed in §2.2).

Sentence classification We create a fixed size vector by applying max-pooling (Collobert et al., 2011; Conneau et al., 2017) over $\mathbf{H}$, which encourages the model to capture the most useful local features encoded in the hidden states. We feed the fixed size global feature vector to a linear layer to obtain the unnormalized predicted scores for each class. Let K be the number of target classes, s_k be the k-th normalized predicted score after applying a softmax function, and $\mathbf{t} \in \mathbb{R}^K$ be the one-hot encoded true label. To train the sentence classification model, we minimize the multi-class cross-entropy loss:

$$\mathcal{L}_{\text{C}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} t_k^{(i)} \log(s_k^{(i)}), \qquad (1)$$

where i denotes the sentence index, and N is the number of training examples.

We not only train the sentence classification and NER models jointly but also pre-train the sentence classification model using a sufficiently large number of training examples with sentence-level labels only. We expect that pre-trained hidden representations would help the model generalize better on our main task, as described below.

NER Following Huang et al. (2015); Lample et al. (2016), we feed $\mathbf{H}$ to a CRF layer to obtain the probability of a label sequence $\mathbf{y}$. To train the NER model, we minimize the negative log-likelihood of the correct label sequences over the training set:

$$\mathcal{L}_{\text{NER}} = -\frac{1}{N}\sum_{i=1}^{N} \log p(\mathbf{y}^{(i)}|\mathbf{H}^{(i)}). \qquad (2)$$

[2] We use the term "sentence" to conform with the literature, although our data are not always complete sentences.

Joint labeling objective Combining Eqs. (1) and (2), we obtain:

$$\mathcal{L}_{\text{JOINT}} = \mathcal{L}_{\text{NER}} + \lambda \mathcal{L}_{\text{C}}, \tag{3}$$

where λ is the balancing parameter. The $\mathcal{L}_{\text{C}}$ acts as a regularization term, which helps in reducing the risk of overfitting on our main task.

2.2 Revisiting attention mechanisms

We first consider a soft-attention mechanism (Shen and Lee, 2016), which is used in Rei and Søgaard (2018, 2019). This method is computationally efficient because the attention distribution $\mathbf{a} \in \mathbb{R}^T$ over tokens in a sentence is computed from the final hidden representation without considering relationships between hidden states. Specifically, the new final representation $\mathbf{H}' \in \mathbb{R}^{T \times d_h}$ can be derived as follows:

$$\begin{aligned} \mathbf{H}' &= \mathbf{H} + \mathbf{H} \otimes \mathbf{a}, \\ \mathbf{a} &= \frac{\tilde{\mathbf{a}}}{\sum_{j=1}^{T} \tilde{a}_j}, \\ \tilde{\mathbf{a}} &= \sigma(\mathbf{w}_2 \mathbf{g} + b_2), \\ \mathbf{g} &= \tanh(\mathbf{W}_3 \mathbf{H}^\top + \mathbf{b}_3), \end{aligned} \tag{4}$$

where $\mathbf{w}_2 \in \mathbb{R}^{d_h}, b_2 \in \mathbb{R}, \mathbf{W}_3 \in \mathbb{R}^{d_h \times d_h}, \mathbf{b}_3 \in \mathbb{R}^{d_h}$ are trainable parameters, and $\otimes$ denotes the column-wise matrix-vector multiplication. We use a residual connection (He et al., 2016) between the input hidden representation and the attention output as shown in Figure 2. $\mathbf{H}'$ can be fed to NER and sentence classification.

We further explore attention mechanisms that take into account the relationships between hidden states. In particular, we apply the multi-head self-attention mechanism in Transformer (Vaswani et al., 2017), which has shown promising results in many applications (Radford et al., 2018; Devlin et al., 2019). We replace Eq. (4) with:

$$\begin{aligned} \mathbf{H}' &= \mathbf{H} + \text{concat}(\text{head}_1, \ldots, \text{head}_n)\mathbf{W}^O, \\ \text{head}_j &= \text{attention}(\mathbf{Q}_j, \mathbf{K}_j, \mathbf{V}_j), \\ \mathbf{Q}_j, \mathbf{K}_j, \mathbf{V}_j &= \mathbf{H}\mathbf{W}_j^Q, \mathbf{H}\mathbf{W}_j^K, \mathbf{H}\mathbf{W}_j^V, \end{aligned} \tag{5}$$

where $\mathbf{W}_j^Q, \mathbf{W}_j^K, \mathbf{W}_j^V \in \mathbb{R}^{d_h \times \frac{d_h}{n}}; \mathbf{W}^O \in \mathbb{R}^{d_h \times d_h}$ are trainable parameters, and n is the number of parallel heads. The attention function can be computed by:

$$\text{attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\alpha})\mathbf{V}. \tag{6}$$

We drop the head index j for simplicity and introduce the scaling factor $\alpha \in \mathbb{R}$. When setting $\alpha = \sqrt{d_h/n}$, Eq. (6) falls back to the standard scaled dot-product attention in Transformer. Yan et al. (2019) observed that the scaled dot-product attention produces poor results for NER and proposed the un-scaled dot-product attention, where $\alpha = 1$.

In this work, we consider α as the softmax temperature (Hinton et al., 2015) that allows adjusting the probability distribution of a softmax output. Using a higher temperature yields a softer attention distribution. However, a sharper attention distribution might be more suitable for NER because only a few tokens in the sentence are named entities. Instead of setting α to 1 or $\sqrt{d_h/n}$, we propose to learn the scaling factors $\boldsymbol{\delta} \in \mathbb{R}^T$ for each token. We modify Eq. (6) with:

$$\begin{aligned} \text{attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\boldsymbol{\delta}})\mathbf{V}, \\ \boldsymbol{\delta} &= \min(\text{ReLU}(\mathbf{w}_4 \mathbf{H}^\top + b_4), \sqrt{d_h/n}) + 1, \end{aligned} \tag{7}$$

where $\mathbf{w}_4 \in \mathbb{R}^{d_h}, b_4 \in \mathbb{R}$ are the trainable parameters. Since the ReLU activation function produces output values in the range $[0, \infty)$, the t-th element of $\boldsymbol{\delta}$ is bounded in the range $[1, 1 + \sqrt{d_h/n}]$. This allows the model to dynamically adapt $\boldsymbol{\delta}$ without increasing much computational cost.

3 Experiments

3.1 Datasets

The data used in our experiments are product titles obtained from major e-commerce websites in Southeast Asian countries during May-June, 2019. They cover three languages, including Vietnamese (VI), Thai (TH), and Indonesian (ID). A product title is a brief, information-rich description (less than 200 characters) written by the sellers. We hired annotators and linguists for each language to annotate the product titles based on our definitions and annotation guidelines.

After the annotation process, we obtained 2,000 product titles per language labeled with 6 product attribute NER tags, including PRODUCT, BRAND, CONSUMER_GROUP, MATERIAL, PATTERN, and

COLOR. For each language, we split the data into 1,000/500/500 – training/development/test sets.[3] The statistics of NER tags can be found in Table 3 (see Appendix A).

For some NER tags, especially PRODUCT, the number of tags is much larger than the number of examples used. One reason is that the sellers writing a product title tend to include multiple different expressions referring to the same entity (near-synonyms), with the likely intention of acquiring more hits from potential customers. Using English to illustrate: "*Genuine Leather Sling Bag Crossbody Bag Messenger bag for Men Women Office Laptop*", the underlined elements are 3 PRODUCT and 2 CONSUMER_GROUP entities.

The other reason is that in one product title, it is common to find repeated identical expressions in the same language, as well as the same entity words appearing in English. Using a VI example to illustrate: "*T-Shirt - Áo thun in phản quang - Ao thun Nam - Ao thun nữ - Áo thun phong cách Nam Nữ*", the underlined elements refer to the same product (*t-shirt*), appearing multiple times in VI and in English.

3.2 Training details

We implement our model on top of the Flair framework (Akbik et al., 2019), which has recently achieved state-of-the-art results in various sequence labeling tasks. Following Lample et al. (2016), we use the IOBES tagging scheme. We use the pre-trained word embeddings of fastText[4] (Bojanowski et al., 2016) with $d_e = 300$ dimensions for each language and a single-layer BiLSTM with $d_h = 512$ hidden units. We apply a locked dropout (Merity et al., 2018) with the probability of 0.5 before and after the BiLSTM layer and to the attention output before the residual connection. For the multi-head self-attention layer, we adapt the implementation of "The Annotated Transformer" (Rush, 2018)[5] and use its default hyperparameters.

We train all models using Adam (Kingma and Ba, 2015) with the batch size of 32, the learning rate of 1e-3, and the gradient clipping of 5. We initialize all model parameters by sampling from $\mathcal{U}(-0.1, 0.1)$. We set λ in Eq. (3) to 1. We use the same parameter setting for all languages. We apply early stopping in which the learning rate decays by 0.5 if the F1 score on the NER development set does not improve 3 times. We train until the learning rate drops below 1e-5, or the training epochs reach 100.

3.3 Pre-trained classification models

We collect unannotated product titles for each language and group them into 6 main categories, including FASHION, HEALTH_BEAUTY, ELECTRONICS, HOME_FURNITURE, MOTORS, and OTHER. Since the number of product titles is different from one language to another, we can create 360k/30k, 1.2M/60k, 864k/60k – training/development sets for VI, TH, and ID, respectively. Since product titles are not segmented in TH, we segment them using a character cluster-based method simplified from the hybrid model of Kruengkrai et al. (2009). We implement our word segmenter based on CRFsuite (Okazaki, 2007) and train the model using the BEST corpus (Kosawat et al., 2009).

We pre-train the classification models for each language. Since our batch size is relatively small compared to the training data size, we find it suffices to train for 2 epochs. The F1 scores on the development sets are 90.08%, 89.79%, and 91.91% for VI, TH, and ID, respectively. The pre-trained model parameters are used to initialize the projection and BiLSTM layers.

3.4 Main results

We run each experiment 10 times using different random seeds and report the average F1 score. All experiments are run on NVIDIA Tesla P100 GPUs. Table 1 shows the results of various models on the test sets. The **Joint** models consistently show improvements over the **NER-only** models, while the **Joint + Pre-trained** models further boost the F1 scores. These results suggest that the proposed framework is effective for all three languages. The **Joint + Pre-trained** model with the **Self + Learned** attention mechanism achieves the best F1 scores at 62.16%, 61.54%, and 76.10% (i.e., 3.78%, 4.20%, and 2.08% improvements over the **NER-only** baselines) for VI, TH, and ID, respectively.

In addition, we experiment using simple data augmentation. The "+10k" and "+50k" rows in Table 1 indicate the number of additional training examples automatically labeled using a dictionary created from the training set. We do not observe any improvement in both the development and test

[3] For TH, 941 training examples remain after removing annotation errors.

[4] https://fasttext.cc/docs/en/crawl-vectors.html

[5] https://nlp.seas.harvard.edu/2018/04/03/attention.html

Model	Attention	VI	TH	ID
NER-only (+10k)	–	53.47	52.47	74.22
NER-only (+50k)	–	51.12	50.35	71.60
NER-only	–	58.38	57.34	74.02
	Soft	58.18	57.49	74.20
	Self + Scaled	58.82	57.80	74.55
	Self + Un-scaled	59.68	58.53	75.24
	Self + Learned	60.18	58.63	74.83
Joint	–	59.47	58.81	74.67
	Soft	59.50	58.82	74.88
	Self + Scaled	59.34	58.46	75.03
	Self + Un-scaled	60.58	59.56	75.66
	Self + Learned	60.25	59.35	75.18
Joint + Pre-trained	–	61.26	60.27	75.86
	Soft	61.05	60.50	75.80
	Self + Scaled	61.80	61.32	75.90
	Self + Un-scaled	62.09	61.45	76.01
	Self + Learned	**62.16**	**61.54**	**76.10**

Table 1: F1 scores on the test sets. **NER-only** = baseline BiLSTM-CRF; **Joint** = joint labeling model; **Joint + Pre-trained** = **Joint** initialized with the pre-trained classification model; **Soft** = soft-attention (Shen and Lee, 2016; Rei and Søgaard, 2019); **Self** = multi-head self-attention described in §2.2, where **Scaled** = scaled dot-product (Vaswani et al., 2017), **Un-scaled** = un-scaled dot-product (Yan et al., 2019), and **Learned** = our learned scaling factors.

Model	VI	TH	ID
Joint + Pre-trained & Self + Learned	**62.16**	**61.54**	76.10
w/o residual connection	61.28	61.52	75.74
w/o locked dropout	61.87	61.08	**76.22**

Table 2: Model ablations for our best configuration, the **Joint + Pre-trained** model with the **Self + Learned** attention mechanism.

results and hence do not pursue this idea further with the attention mechanisms.

Table 2 shows the model ablations for our best configuration, the **Joint + Pre-trained** model with the **Self + Learned** attention mechanism. Feeding the attention output to the CRF layer without the residual connection leads to a consistent drop in the F1 scores, although it shows a less pronounced effect on TH. The results indicate that the residual connection is a useful component in our architecture. Adding the attention output to the hidden representation without applying the locked dropout (i.e., setting the dropout probability to 0) hurts the F1 scores on VI and TH but shows an improvement on ID, suggesting that fine-tuning the dropout rate could help boost the F1 scores.

3.5 Discussion

Our **Self + Learned** scaling approach shows the competitive results for the **NER-only** model and achieves the best results when training in tandem with the **Joint + Pre-trained** model. The **Soft** attention mechanism (Shen and Lee, 2016; Rei and Søgaard, 2019) shows slight or no improvements, suggesting that considering relationships between hidden states when computing the attention distribution is crucial for the NER task. The **Self + Un-scaled** approach (Yan et al., 2019) yields better F1 scores than the **Self + Scaled** approach (Vaswani et al., 2017) for all configurations, suggesting that a sharper attention distribution is helpful for the NER task.

Although VI, TH, and ID are used in Southeast Asia, they do not belong to the same language family and have different writing systems and scripts (i.e., VI = Austroasiatic; TH = Kra-Dai; ID = Austronesian). Handling these three languages without much engineering effort reflects the generalizability of our method. Furthermore, we examine whether our method still provides improvements, even if the NER training data size increases. We create an additional set of 2k labeled examples for VI and add them to the training set (3k in total). The baseline **NER-only** produces 66.81% F1, while **Joint + Pre-trained** with **Self + Learned** achieves 69.26% F1 (i.e., 2.45% improvement).

4 Conclusion

We have shown that the proposed joint sentence and token labeling model is remarkably effective for low-resource NER in three different languages: Vietnamese, Thai, and Indonesian. Our model supports multi-class classification where the sentence and token labels can be weakly related, which indicates the potential of our model for many other real-world applications. Using a larger amount of general domain texts to build pre-trained representations (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019; Clark et al., 2020) can complement with our model and is one of the directions that we plan to take in future work.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. Kruengkrai is grateful for support from National Institute of Informatics, Japan.

References

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. In *Proceedings of ACL Workshop on Noisy User-generated Text*, pages 148–153.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of NAACL (Demonstrations)*, pages 54–59.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of COLING*, pages 1638–1649.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv:1607.04606*.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of ICML*, pages 41–48.

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

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proceedings of ICLR*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, pages 2493–2537.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of EMNLP*, pages 670–680.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Xiaocheng Feng, Xiachong Feng, Bing Qin, Zhangyin Feng, and Ting Liu. 2018. Improving low resource named entity recognition using cross-lingual knowledge transfer. In *Proceedings of IJCAI*, pages 4071–4077.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of EMNLP*, pages 1923–1933.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of CVPR*, pages 770–778.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. In *Proceedings of NIPS Deep Learning and Representation Learning Workshop*.

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Krit Kosawat, Monthika Boriboon, Patcharika Chootrakool, Ananlada Chotimongkol, Supon Klaithin, Sarawoot Kongyoung, Kanyanut Kriengket, Sitthaa Phaholphinyo, Sumonmas Purodakananda, Tipraporn Thanakulwarapas, and Chai Wutiwiwatchai. 2009. Best 2009 : Thai word segmentation software contest. In *Proceedings of International Symposium on Natural Language Processing (SNLP)*, pages 83–88.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Kentaro Torisawa, Hitoshi Isahara, and Chuleerat Jaruskulchai. 2009. A word and character-cluster hybrid model for thai word segmentation. In *Proceedings of InterBEST*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL*, pages 260–270.

Ying Lin, Shengqi Yang, Veselin Stoyanov, and Heng Ji. 2018. A multi-lingual multi-task architecture for low-resource sequence labeling. In *Proceedings of ACL*, pages 799–809.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of ACL*, pages 1064–1074.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. Regularizing and optimizing LSTM language models. In *Proceedings of ICLR*.

Naoaki Okazaki. 2007. CRFsuite: a fast implementation of conditional random fields (CRFs). URL: http://www.chokkan.org/software/crfsuite.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL*, pages 2227–2237.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. URL: https://openai.com/blog/language-unsupervised.

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. In *Proceedings of ACL*, pages 2121–2130.

Marek Rei and Anders Søgaard. 2018. Zero-shot sequence labeling: Transferring knowledge from sentences to tokens. In *Proceedings of NAACL*, pages 293–302.

Marek Rei and Anders Søgaard. 2019. Jointly learning to label sentences and tokens. In *Proceedings of AAAI*, pages 6916–6923.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv:1706.05098*.

Alexander Rush. 2018. The annotated transformer. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 52–60.

Sheng-syun Shen and Hung-yi Lee. 2016. Neural attention models for sequence classification: Analysis and application to key term extraction and dialogue act detection. In *Proceedings of INTERSPEECH*, pages 2716–2720.

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 *Proceedings of NIPS*, pages 5998–6008.

Dingquan Wang, Nanyun Peng, and Kevin Duh. 2017. A multi-task learning approach to adapting bilingual word embeddings for cross-lingual named entity recognition. In *Proceedings of IJCNLP*, pages 383–388.

Taiki Watanabe, Akihiro Tamura, Takashi Ninomiya, Takuya Makino, and Tomoya Iwakura. 2019. Multi-task learning for chemical named entity recognition with chemical compound paraphrasing. In *Proceedings of EMNLP-IJCNLP*, pages 6243–6248.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. TENER: adapting transformer encoder for named entity recognition. *arXiv:1911.04474*.

A Statistics of NER tags

Table 3 shows the statistics of NER tags in the training, development, and test sets.

NER Type	VI			TH			ID		
	Train	Dev	Test	Train	Dev	Test	Train	Dev	Test
BRAND	358	160	170	725	408	387	490	215	229
COLOR	488	249	195	640	298	322	582	277	295
CONSUMER_GROUP	763	369	341	399	238	217	1910	1098	1026
MATERIAL	291	154	135	490	258	221	260	109	151
PATTERN	843	435	392	501	273	245	1021	537	493
PRODUCT	1982	964	963	2808	1473	1521	4786	2584	2557
TOTAL	4725	2331	2196	5563	2948	2913	9049	4820	4751

Table 3: Statistics of NER tags.