An Investigation of Potential Function Designs for Neural CRF

Zechuan Hu[◇], **Yong Jiang**[†*], **Nguyen Bach**[†], **Tao Wang**[†], **Fei Huang**[†], **Kewei Tu**[◇*]
[◇]School of Information Science and Technology, ShanghaiTech University
Shanghai Engineering Research Center of Intelligent Vision and Imaging
Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences
University of Chinese Academy of Sciences
[†]DAMO Academy, Alibaba Group
{huzch,tukw}@shanghaitech.edu.cn
{yongjiang.jy,nguyen.bach,leeo.wangt,f.huang}@alibaba-inc.com

Abstract

The neural linear-chain CRF model is one of the most widely-used approach to sequence labeling. In this paper, we investigate a series of increasingly expressive potential functions for neural CRF models, which not only integrate the emission and transition functions, but also explicitly take the representations of the contextual words as input. Our extensive experiments show that the decomposed quadrilinear potential function based on the vector representations of two neighboring labels and two neighboring words consistently achieves the best performance.

1 Introduction

Sequence labeling is the task of labeling each token of a sequence. It is an important task in natural language processing and has a lot of applications such as Part-of-Speech Tagging (POS) (DeRose, 1988; Toutanova et al., 2003; Xin et al., 2018), Named Entity Recognition (NER) (Ritter et al., 2011; Akbik et al., 2019), Chunking (Tjong Kim Sang and Buchholz, 2000; Suzuki et al., 2006).

The neural CRF model is one of the most widely-used approaches to sequence labeling and can achieve superior performance on many tasks (Collobert et al., 2011; Chen et al., 2015; Ling et al., 2015; Ma and Hovy, 2016; Lample et al., 2016a). It often employs an encoder such as a BiLSTM to compute the contextual vector representation of each word in the input sequence. The potential function at each position of the input sequence in a neural CRF is typically decomposed into an emission function (of the current label and the vector representation of the current word) and a transition function (of the previous and current labels) (Liu et al., 2018; Yang et al., 2018).

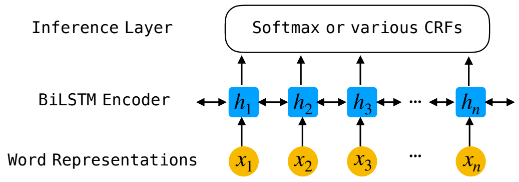

Figure 1: Neural architecture for sequence labeling

In this paper, we design a series of increasingly expressive potential functions for neural CRF models. First, we compute the transition function from label embeddings (Ma et al., 2016; Nam et al., 2016; Cui and Zhang, 2019) instead of label identities. Second, we use a single potential function over the current word and the previous and current labels, instead of decomposing it into the emission and transition functions, leading to more expressiveness. We also employ tensor decomposition in order to keep the potential function tractable. Thirdly, we take the representations of additional neighboring words as input to the potential function, instead of solely relying on the BiLSTM to capture contextual information.

To empirically evaluate different approaches, we conduct experiments on four well-known sequence labeling tasks: NER, Chunking, coarse- and fine-grained POS tagging. We find that it is beneficial for the potential function to take representations of neighboring words as input, and a quadrilinear potential function with a decomposed tensor parameter leads to the best overall performance.

Our work is related to Reimers and Gurevych (2017); Yang et al. (2018), which also compared different network architectures and configurations and conducted empirical analysis on different sequence labeling tasks. However, our focus is on the potential function design of neural CRF models, which has not been sufficiently studied before.

[*] Kewei Tu and Yong Jiang are the corresponding authors.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 2600–2609
November 16 - 20, 2020.

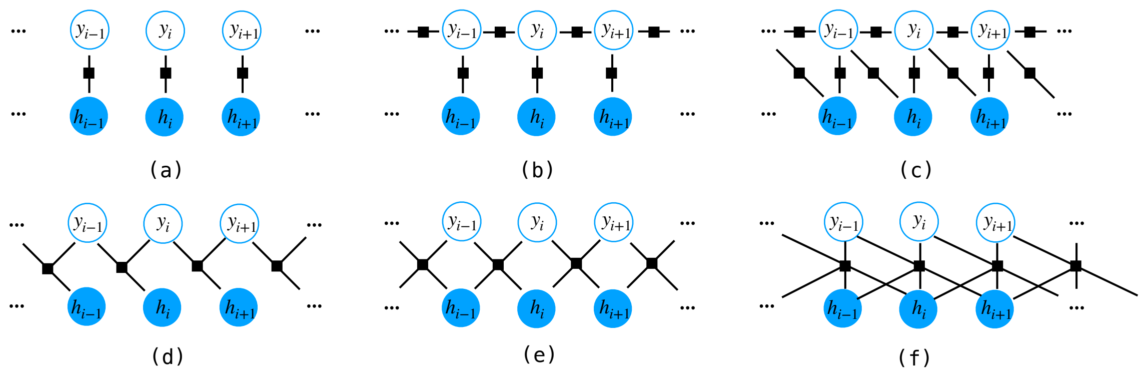

Figure 2: Factor graphs of different models. The solid circles and hollow circles indicate random variables of word encodings and labels respectively. The black squares represent factors.

2 Models

Our overall neural network architecture for sequence labeling is shown in Figure 1. It contains three parts: a word representation layer, a bi-directional LSTM (BiLSTM) encoder, and an inference layer. The BiLSTM encoder produces a sequence of output vectors $\mathbf{h}_1, \mathbf{h}_2, ...\mathbf{h}_M \in \mathbb{R}^{D_h}$, which are utilized by the inference layer to predict the label sequence. The inference layer typically defines a potential function $s(\mathbf{x}, \mathbf{y}, i)$ for each position i of the input sequence $\mathbf{x}$ and label sequence $\mathbf{y}$ and computes the conditional probability of the label sequence given the input sequence as follows:

$$P(\mathbf{y}|\mathbf{x}) = \frac{\exp(\sum_{i=1}^{M} s(\mathbf{x}, \mathbf{y}, i))}{\sum_{\mathbf{y}'} \exp(\sum_{i=1}^{M} s(\mathbf{x}, \mathbf{y}', i))}$$

where M is the length of the sequence.

The simplest inference layer assumes independence between labels. It applies a linear transformation to $\mathbf{h}_i$ followed by a **Softmax** function to predict the distribution of label y_i at each position i (Figure 2(a)). In many scenarios, however, it makes sense to model dependency between neighboring labels, which leads to linear-chain CRF models.

Vanilla CRF In most previous work of neural CRFs, the potential function is decomposed to an emission function and a transition function (Figure 2(b)), and the transition function is represented by a table ϕ maintaining the transition scores between labels.

$$s(\mathbf{x}, \mathbf{y}, i) = \mathbf{v}_{y_{i-1}}^T \phi \mathbf{v}_{y_i} + \mathbf{h}_i^T \mathbf{W}_h \mathbf{v}_{y_i}$$

where $\mathbf{v}_{y_i}$ is a one-hot vector for label y_i and $\mathbf{W}_h \in \mathbb{R}^{D_h \times D_t}$ is a weight matrix.

TwoBilinear Instead of one-hot vectors, we may use dense vectors to represent labels, which has the benefit of encoding similarities between labels. Accordingly, the emission and transition functions are modeled by two bilinear functions.

$$s(\mathbf{x}, \mathbf{y}, i) = \mathbf{t}_{y_{i-1}}^T \mathbf{W}_t \mathbf{t}_{y_i} + \mathbf{h}_i^T \mathbf{W}_h \mathbf{t}_{y_i}$$

where $\mathbf{W}_t \in \mathbb{R}^{D_t \times D_t}$ is a weight matrix, and $\mathbf{t}_{y_i} \in \mathbb{R}^{D_t}$ is the embedding of label y_i. The factor graph remains the same as vanilla CRF (Figure 2(b)).

ThreeBilinear Figure 2(c) depicts the structure of ThreeBilinear. Compared with TwoBilinear, ThreeBilinear has an extra emission function between the current word representation and previous label.

$$\begin{aligned} s(\mathbf{x}, \mathbf{y}, i) =& \mathbf{t}_{y_{i-1}}^T \mathbf{W}_t \mathbf{t}_{y_i} + \mathbf{h}_i^T \mathbf{W}_{h_1} \mathbf{t}_{y_i} \\ &+ \mathbf{h}_i^T \mathbf{W}_{h_2} \mathbf{t}_{y_{i-1}} \end{aligned}$$

Trilinear Instead of three bilinear functions, we may use a trilinear function to model the correlation between $\mathbf{h}_i$, $\mathbf{t}_{y_i}$ and $\mathbf{t}_{y_{i-1}}$. It has strictly more representational power than the sum of three bilinear functions.

$$s(\mathbf{x}, \mathbf{y}, i) = \mathbf{h}_i^T \mathbf{U} \mathbf{t}_{y_{i-1}} \mathbf{t}_{y_i}$$

where $\mathbf{U} \in \mathbb{R}^{D_h \times D_t \times D_t}$ is an order-3 weight tensor. Figure 2(d) presents the structure of Trilinear.

D-Trilinear Despite the increased representational power of Trilinear, its space and time complexity becomes cubic. To reduce the computational complexity without too much compromise of the representational power, we assume that $\mathbf{U}$ has rank D_r and can be decomposed into the product of three matrices $\mathbf{U}_{t_1}$, $\mathbf{U}_{t_2} \in \mathbb{R}^{D_t \times D_r}$ and

$\mathbf{U_h} \in \mathbb{R}^{D_h \times D_r}$. Then the trilinear function can be rewritten as,

$$s(\mathbf{x}, \mathbf{y}, i) = \sum_{j=1}^{D_r} (\mathbf{g}_1 \circ \mathbf{g}_2 \circ \mathbf{g}_3)_j$$

$$\mathbf{g}_1 = \mathbf{t}_{y_{i-1}}^T \mathbf{U}_{t_1}; \quad \mathbf{g}_2 = \mathbf{t}_{y_i}^T \mathbf{U}_{t_2}; \quad \mathbf{g}_3 = \mathbf{h}_i^T \mathbf{U}_h$$

where $\circ$ denotes element-wise product. We call the resulting model D-Trilinear. The factor graph of D-Trilinear is the same as Trilinear (Figure 2(d)).

D-Quadrilinear We may take the representation of the previous word as an additional input and use a quadrilinear function in the potential function.

$$s(\mathbf{x}, \mathbf{y}, i) = \mathbf{h}_{i-1}^T \mathbf{h}_i^T \mathbf{U} \mathbf{t}_{y_{i-1}} \mathbf{t}_{y_i}$$

where $\mathbf{U}$ is an order-4 weight tensor. However, the computational complexity of this function becomes quartic. Hence we again decompose the tensor into the product of four matrices and rewrite the potential function as follows.

$$s(\mathbf{x}, \mathbf{y}, i) = \sum_{j=1}^{D_r} (\mathbf{g}_1 \circ \mathbf{g}_2 \circ \mathbf{g}_3 \circ \mathbf{g}_4)_j$$

$$\mathbf{g}_1 = \mathbf{t}_{y_{i-1}}^T \mathbf{U}_{t_1}; \quad \mathbf{g}_2 = \mathbf{t}_{y_i}^T \mathbf{U}_{t_2}$$

$$\mathbf{g}_3 = \mathbf{h}_{i-1}^T \mathbf{U}_{h_1}; \quad \mathbf{g}_4 = \mathbf{h}_i^T \mathbf{U}_{h_2}$$

We call the resulting model D-Quadrilinear and its factor graph is shown in Figure 2(e).

D-Pentalinear Following the same idea, we extend D-quadrilinear to D-Pentalinear by taking the representation of the next word as an additional input. Figure 2(f) shows the structure of D-Pentalinear.

$$s(\mathbf{x}, \mathbf{y}, i) = \sum_{j=1}^{D_r} (\mathbf{g}_1 \circ \mathbf{g}_2 \circ \mathbf{g}_3 \circ \mathbf{g}_4 \circ \mathbf{g}_5)_j$$

$$\mathbf{g}_1 = \mathbf{t}_{y_{i-1}}^T \mathbf{U}_{t_1}; \quad \mathbf{g}_2 = \mathbf{t}_{y_i}^T \mathbf{U}_{t_2}$$

$$\mathbf{g}_3 = \mathbf{h}_{i-1}^T \mathbf{U}_{h_1}; \quad \mathbf{g}_4 = \mathbf{h}_i^T \mathbf{U}_{h_2}; \quad \mathbf{g}_5 = \mathbf{h}_{i+1}^T \mathbf{U}_{h_3}$$

3 Experiments

We compare neural Softmax and the seven variants of neural CRFs on four sequence labeling tasks: NER, Chunking, coarse- and fine-grained POS tagging. For NER, we use the datasets from CoNLL 2002 and 2003 shared tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003). For Chunking, we use the English and German datasets of the CoNLL 2003 shared task (Tjong Kim Sang and De Meulder, 2003) and the Vietnamese dataset (Pham et al., 2017). For the two POS tagging tasks, we select 8 languages from Universal Dependencies (UD **v2.4**) treebanks (Nivre et al., 2019).

We conduct our experiments with pretrained word embeddings, character embeddings, and BERT embeddings (Devlin et al., 2019a). For NER and Chunking, we use the BIOES scheme for its better performance than the BIO scheme (Ratinov and Roth, 2009; Dai et al., 2015; Yang et al., 2018). We use F1-score as the evaluation metric for both NER and Chunking. We run each model for 5 times with different random seeds for each experiment and report the average score and the standard derivation. More details can be found in supplementary material.

3.1 Results

We show the detailed results on NER and Chunking with BERT embeddings in Table 1 and the averaged results on all the tasks in Table 2 (the complete results can be found in the supplementary materials). We make the following observations. **Firstly**, D-Quadrilinear has the best overall performance in all the tasks. Its advantage over D-Trilinear is somewhat surprising because the BiLSTM output $\mathbf{h}_i$ in D-Trilinear already contains information of both the current word and the previous word. We speculate that: 1) information of the previous word is useful in evaluating the local potential in sequence labeling (as shown by traditional feature-based approaches); and 2) information of the previous word is obfuscated in $\mathbf{h}_i$ and hence directly inputting $\mathbf{h}_{i-1}$ into the potential function helps. **Secondly**, D-Quadrilinear greatly outperforms BiLSTM-LAN (Cui and Zhang, 2019), one of the state-of-the-art sequence labeling approaches which employs a hierarchically-refined label attention network. **Thirdly**, D-Trilinear clearly outperforms both ThreeBilinear and Trilinear. This suggests that tensor decomposition could be a viable way to both regularize multilinear potential functions and reduce their computational complexity.

3.2 Analysis

Small training data We train four of our models on randomly selected 10% or 30% of the training data on the NER and Chunking tasks. We run each experiment for 5 times. Figure 3 shows the average difference in F1-scores between each model and

		NER					CHUNKING			
		English	German	Dutch	Spanish	Avg.	English	German	Vietnamese	Avg.
BERT EMBEDDING	SOFTMAX	90.42±0.16	81.91±0.15	89.02±0.31	85.86±0.34	86.80±0.24	90.72±0.11	93.48±0.07	74.13±0.27	86.11±0.15
	VANILLA CRF	91.33±0.18	83.56±0.18	90.03±0.18	87.32±0.38	88.06±0.23	91.05±0.12	93.65±0.08	76.07±0.08	86.92±0.09
	TWOBILINEAR	91.23±0.07	83.21±0.35	90.02±0.26	87.40±0.24	87.96±0.23	91.16±0.04	93.60±0.10	76.10±0.23	86.95±0.13
	THREEBILINEAR	91.19±0.24	83.35±0.19	90.06±0.45	87.38±0.18	87.99±0.26	91.13±0.14	93.52±0.14	75.98±0.23	86.87±0.17
	TRILINEAR	91.24±0.11	83.11±0.27	90.53±0.41	87.38±0.26	88.07±0.26	91.11±0.04	93.68±0.09	75.64±0.25	86.81±0.13
	D-TRILINEAR	91.28±0.16	83.25±0.36	90.52±0.25	87.68±0.13	88.18±0.22	91.32±0.08	93.79±0.10	76.18±0.13	87.10±0.10
	D-QUADRILINEAR	91.46±0.07	83.61±0.22	**90.76±0.13**	**87.71±0.29**	**88.38±0.18**	**91.51±0.11**	94.08±0.08	**76.29±0.36**	**87.29±0.18**
	D-PENTALINEAR	**91.47±0.20**	**83.63±0.26**	90.50±0.27	87.69±0.20	88.33±0.23	91.45±0.08	**94.23±0.06**	76.01±0.20	87.23±0.11

Table 1: Results on NER and Chunking with BERT embeddings.

	NER		CHUNKING		FINE-GRAINED POS			COARSE-GRAINED POS		
	WORD	CHAR	WORD	CHAR	WORD	CHAR	BERT	WORD	CHAR	BERT
BILSTM-LAN	77.70±0.39	82.42±0.55	85.59±0.12	86.12±0.12	94.45±0.14	95.41±0.13	—	94.75±0.10	95.68±0.08	—
SOFTMAX	78.22±0.32	82.14±0.26	84.99±0.14	85.49±0.07	94.91±0.08	95.72±0.07	95.83±0.07	94.47±0.09	95.58±0.08	96.18±0.08
VANILLA CRF	79.46±0.57	83.59±0.66	85.86±0.11	86.39±0.08	94.89±0.08	95.70±0.11	95.81±0.09	94.53±0.10	95.60±0.10	96.23±0.09
TWOBILINEAR	79.16±0.42	83.36±0.42	85.57±0.19	85.94±0.15	94.81±0.11	95.64±0.10	95.79±0.09	94.48±0.08	95.58±0.11	96.18±0.09
THREEBILINEAR	78.66±0.94	83.53±0.28	85.51±0.23	85.95±0.21	94.87±0.09	95.66±0.09	95.74±0.11	94.49±0.09	95.54±0.09	96.14±0.08
TRILINEAR	79.24±0.35	83.50±0.38	85.57±0.28	86.08±0.31	94.94±0.13	95.71±0.11	95.67±0.11	94.61±0.11	95.63±0.12	96.17±0.14
D-TRILINEAR	79.41±0.24	83.75±0.39	85.83±0.13	86.42±0.14	95.07±0.10	95.75±0.08	95.74±0.11	94.70±0.11	95.69±0.08	96.25±0.08
D-QUADRILINEAR	**80.09±0.35**	**84.20±0.39**	**86.58±0.14**	87.07±0.10	**95.19±0.08**	**95.88±0.08**	**95.90±0.09**	**94.91±0.10**	**95.82±0.10**	**96.32±0.07**
D-PENTALINEAR	79.52±0.28	84.01±0.42	86.53±0.15	**87.11±0.20**	95.07±0.19	95.82±0.11	95.85±0.08	94.80±0.15	95.79±0.13	96.31±0.11

Table 2: Results averaged over all the languages for each task. We also show the results of BiLSTM-LAN (Cui and Zhang, 2019), one of the current state-of-the-art sequence labeling approaches, for reference. We do not report the results of BiLSTM-LAN with BERT embedding because BERT is not available in the BiLSTM-LAN code.

		NER	CHUNKING
LAYERS=2	VANILLA CRF	79.86±0.47	85.84±0.19
	D-TRILINEAR	80.21±0.34	85.86±0.19
	D-QUADRILINEAR	**80.36±0.34**	**86.32±0.14**
LAYERS=3	VANILLA CRF	78.72±0.66	85.73±0.15
	D-TRILINEAR	79.84±0.62	85.65±0.20
	D-QUADRILINEAR	**79.97±0.31**	**85.88±0.15**

Table 3: Average results with more BiLSTM layers.

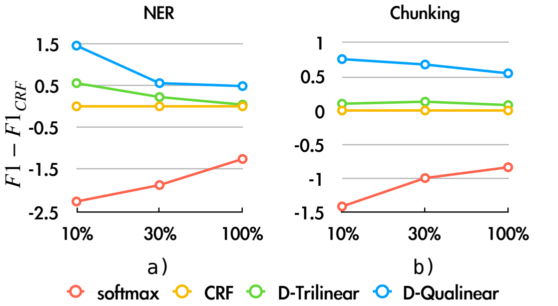

Figure 3: The average differences in F1-scores compared with Vanilla CRF with different training data sizes.

Vanilla CRF. It can be seen that with small data, the advantages of D-Trilinear and D-Quadrilinear over Vanilla CRF and Softmax become even larger.

Multi-layers LSTM As discussed in section 3.1, D-Quadrilinear outperforms D-Trilinear probably because $\mathbf{h}_i$, the BiLSTM output at position i, does not contain sufficient information of the previous word. Here we study whether increasing the number of BiLSTM layers would inject more information into $\mathbf{h}_i$ and hence reduce the performance gap between the two models. Table 3 shows the results on the NER and Chunking tasks with word embedding. D-Quadrilinear still outperforms D-Trilinear, but by comparing Table 3 with Table 2, we see that their difference indeed becomes smaller with more BiLSTM layers. Another observation is that more BiLSTM layers often lead to lower scores. This is consistent with previous findings (Cui and Zhang, 2019) and is probably caused by overfitting.

Speed We test the training and inference speed of our models. Our decomposed multilinear approaches are only a few percent slower than Vanilla CRF during training and as fast as Vanilla CRF during inference, which suggests their practical usefulness. The details can be found in the supplementary material.

4 Conclusion

In this paper, we investigate several potential functions for neural CRF models. The proposed potential functions not only integrate the emission and transition functions, but also take into consideration representations of additional neighboring words. Our experiments show that D-Quadrilinear achieves the best overall performance. Our proposed approaches are simple and effective and could facilitate future research in neural sequence labeling.

Acknowledgement

This work was supported by Alibaba Group through Alibaba Innovative Research Program. This work was also supported by the National Natural Science Foundation of China (61976139).

References

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. 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 724–728, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015. Long short-term memory neural networks for Chinese word segmentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1197–1206, Lisbon, Portugal. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*.

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 4106–4119, Hong Kong, China. Association for Computational Linguistics.

Hong-Jie Dai, Po-Ting Lai, Yung-Chun Chang, and Richard Tzong-Han Tsai. 2015. Enhancing of chemical compound and drug name recognition using representative tag scheme and fine-grained tokenization. *Journal of cheminformatics*, 7(1):S14.

Steven J. DeRose. 1988. Grammatical category disambiguation by statistical optimization. *Computational Linguistics*, 14(1).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. 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.

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

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. *ArXiv*, abs/1802.06893.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016a. 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.

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

Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramón Fermandez, Silvio Amir, Luís Marujo, and Tiago Luís. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal. Association for Computational Linguistics.

Liyuan Liu, Jingbo Shang, Xiang Ren, Frank Fangzheng Xu, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

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.

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics:*

Technical Papers, pages 171–180, Osaka, Japan. The COLING 2016 Organizing Committee.

Jinseok Nam, Eneldo Loza Mencía, and Johannes Fürnkranz. 2016. All-in text: Learning document, label, and word representations jointly. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Gabrielė Aleksandravičiūtė, Lene Antonsen, Katya Aplonova, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, Victoria Basmov, John Bauer, Sandra Bellato, Kepa Bengoetxea, Yevgeni Berzak, Irshad Ahmad Bhat, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Agnė Bielinskienė, Rogier Blokland, Victoria Bobicev, Loïc Boizou, Emanuel Borges Völker, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Kristina Brokaitė, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Flavio Massimiliano Cecchini, Giuseppe G. A. Celano, Slavomír Čéplö, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Carly Dickerson, Bamba Dione, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Hanne Eckhoff, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Tomaž Erjavec, Aline Etienne, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Kazunori Fujita, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Sebastian Garza, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta González Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Johannes Heinecke, Felix Hennig, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Takumi Ikeda, Radu Ion, Elena Irimia, Ọlájídé Ishola, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Andre Kaasen, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Boris Katz, Tolga Kayadelen, Jessica Kenney, Václava Kettnerová, Jesse Kirchner, Arne Köhn, Kamil Kopacewicz, Natalia Kotsyba, Jolanta Kovalevskaitė, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Lucia Lam, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phng Lê H`ông, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, Kyung-Tae Lim, Yuan Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Sarah McGuinness, Gustavo Mendonça, Niko Miekka, Margarita Misirpashayeva, Anna Missilä, Cătălin Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Keiko Sophie Mori, Tomohiko Morioka, Shinsuke Mori, Shigeki Moro, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Lng Nguy˜ên Thị, Huy`ên Nguy˜ên Thị Minh, Yoshihiro Nikaido, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Adédayọ Olúòkun, Mai Omura, Petya Osenova, Robert Östling, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Guilherme Paulino-Passos, Angelika Peljak-Łapińska, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Daria Petrova, Slav Petrov, Jussi Piitulainen, Tommi A Pirinen, Emily Pitler, Barbara Plank, Thierry Poibeau, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Michael Rießler, Erika Rimkutė, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Roșca, Olga Rudina, Jack Rueter, Shoval Sadde, Benoît Sagot, Shadi Saleh, Alessio Salomoni, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Dage Särg, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Hiroyuki Shirasu, Muh Shohibussirri, Dmitry Sichinava, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Carolyn Spadine, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Shingo Suzuki, Zsolt Szántó, Dima Taji, Yuta Takahashi, Fabio Tamburini, Takaaki Tanaka, Isabelle Tellier, Guillaume Thomas, Liisi Torga, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Eric Villemonte de la Clergerie, Veronika Vincze, Lars Wallin, Abigail Walsh, Jing Xian Wang, Jonathan North Washington, Maximilan Wendt, Seyi Williams, Mats Wirén, Christian Wittern, Tsegay Woldemariam, Tak-sum Wong, Alina Wróblewska, Mary Yako, Naoki Yamazaki, Chunxiao Yan, Koichi Yasuoka, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Daniel Zeman, Manying Zhang, and Hanzhi Zhu. 2019. Universal dependencies 2.4. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Jeffrey Pennington, Richard Socher, and Christopher

Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*.

Thai-Hoang Pham, Xuan-Khoai Pham, Tuan-Anh Nguyen, and Phuong Le-Hong. 2017. NNVLP: A neural network-based Vietnamese language processing toolkit. In *Proceedings of the IJCNLP 2017, System Demonstrations*, pages 37–40, Tapei, Taiwan. Association for Computational Linguistics.

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.

Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Jun Suzuki, Erik McDermott, and Hideki Isozaki. 2006. Training conditional random fields with multivariate evaluation measures. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 217–224, Sydney, Australia. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.

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.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 252–259.

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.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

A Appendices

A.1 Dataset Statistics

The statistics of the datasets used in our experiments are listed in table 4.

Task	D	#train	#dev	#test	#label
NER	en	14040	3250	3453	17
	de	12152	2867	3005	17
	nl	15796	2895	5196	17
	sp	8319	1914	1517	17
Chunking	en	14040	3250	3453	38
	de	12152	2867	3005	13
	vi	6284	786	785	37
POS	en	12543	2002	2077	50/17
	de	13814	799	977	52/17
	it	13121	564	482	39/17
	id	4477	559	557	81/16
	nl	12269	718	596	194/16
	hi	13304	1659	1684	31/16
	zh	3997	500	500	42/15
	ja	7125	511	550	37/16

Table 4: The statistics of different datasets for corresponding tasks. D: Datasets. The statistics of Coarse-grained POS is the same as Fine-grained POS except that the number of labels are not the same. The left of '/' indicates the number of labels of Fine-grained POS and the right of '/' indicates the number of labels of Coarse-grained POS.

A.2 Word representations

We have three different versions of word representations:

- **Word Embedding.** We use pretrained word embeddings such as GloVe (Pennington et al., 2014) and FastText (Grave et al., 2018).
- **Word Embedding and Character Embedding.** We use the same character LSTMs as in Lample et al. (2016b) and set the hidden

Hyperparameters	Setting
LSTM Hidden Size	512
Learning Rate	0.1
Char Embeddng Size	25
Char Hidden Size	50
Dropout Rate	0.5
L2 Regularization	1e-8
Batch Size	32
Maximal Epochs	300
Patience	10

Table 5: Other hyperparameters

size of the LSTM to 50. The final word representation is the concatenation of the output of the character LSTM and the pretrained word embedding.

- **BERT Embedding.** We use the respective BERT embedding from (Devlin et al., 2019b) for each language. If there is no pretrained BERT embedding for a language, we then use the multilingual BERT (M-BERT) instead. The word representation is from the last four layers of the BERT embedding.

We fine-tune the word embeddings and character embeddings during the training process. We don't fine-tune the BERT embeddings.

A.3 Hyperparameters setting

We tune the following hyperparameters in our experiments.

LSTM hidden size We test Softmax, Vanilla CRF, D-Trilinear and D-Quadlinear with LSTM hidden sizes of {200, 512} on the English and German datasets of each task and find that there is no significant difference between 200 and 512. Hence, we fix the LSTM hidden size to 512.

Learning Rate We tune it in the range of {0.03, 0.1, 0.3} on Softmax, Vanilla CRF, D-Trilinear and D-Quadlinear on the English and German datasets of each task. We find that the performance is always better when the learning rate is 0.1. So we fix the learning rate to 0.1.

Tag Embedding Dimension D_t We use tag embeddings in all the models except Softmax and Vanilla CRF. We search for the best dimension in {20, 50, 100, 200}.

	English	German	Dutch	Spanish	Avg.
VANILLA CRF	88.33	78.59	64.88	81.2	78.25
D-QUADRILINEAR	**89.49**	**79.93**	**67.23**	**81.6**	**79.56**

Table 7: Average results with transformer encoder.

Rank D_r In D-Trilinear, D-Quadlinear, and D-Pentalinear, D_r is a hyperparameter that controls the representational power of the multilinear functions. We select its value from {64, 128, 256, 384, 600}.

Other hyperparameter settings are list in table 5.

A.4 Additional Analysis

Multilinear vs. Concatenation Our best-performing models are based on multilinear functions with decomposed parameter tensors. An alternative to multilinear functions is to apply an MLP with nonlinear activations to the concatenated input vectors. We run the comparison on the NER task with word embeddings and tune the tag embedding size from {20, 50, 100, 200} and the hidden size of the MLP from {64, 128, 256, 384}. As shown in table 6, the two concatenation-based models underperform their decomposed multilinear counterparts, but they do outperform TwoBilinear and ThreeBilinear.

Transformer vs. BiLSTM As we discussed in section 3.1, information of the previous word may be obfuscated in $\mathbf{h}_i$. Transformer-like encoders which can model long-range context may alleviate the obfuscation. We use a 6-layers transformer encoder and run the comparison on vanilla CRF and D-Quadrilinear on NER tasks with word embeddings. As shown in table 7, with the transformer encoder, D-Quadrilinear outperforms the vanilla CRF by 1.31%. In comparison, with the BiLSTM encoder, D-Quadrilinear outperforms the vanilla CRF by 0.63%. So the advantage of our approach against the vanilla CRF becomes even larger when using the transformer encoder.

Speed We use a Nvidia Titan V GPU to test the training and inference speed of the 8 models on the NER English dataset. Figure 4 shows the training and inference time averaged over 10 epochs. Softmax is much faster than all the other approaches because it does not need to run Forward-Backward and Viberbi and can parallelize the predictions at all the positions of a sequence. Our decomposed multilinear approaches are not significantly slower than Vanilla CRF but generally have better perfor-

	English	German	Dutch	Spanish	Avg.
TWOBILINEAR	90.11±0.25	73.69±0.30	69.31±0.67	83.53±0.44	79.16±0.42
THREEBILINEAR	90.12±0.22	73.20±0.85	67.50±2.35	83.82±0.32	78.66±0.94
TRILINEAR	90.19±0.19	73.39±0.40	69.69±0.60	83.70±0.19	79.24±0.35
D-TRILINEAR	90.43±0.23	73.57±0.17	69.50±0.32	84.15±0.23	79.41±0.24
1WORD+2LABEL	89.91±0.04	74.37±0.12	68.96±0.83	83.68±0.22	79.23±0.30
D-QUADLINEAR	**90.44±0.07**	**75.05±0.35**	**70.49±0.68**	**84.41±0.29**	**80.10±0.35**
2WORD+2LABEL	90.27±0.09	74.19±0.32	70.34±0.06	83.81±0.37	79.65±0.21

Table 6: Comparison with concatenation-based potential functions (1WORD+2LABEL and 2WORD+2LABEL)

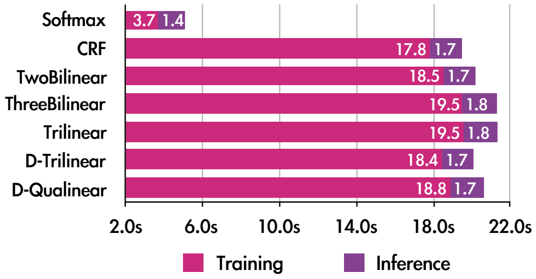

Figure 4: Training time and inference time averaged over 10 epochs.

		NER					CHUNKING			
		English	German	Dutch	Spanish	Avg.	English	German	Vietnamese	Avg.
WORD EMBEDDING	BILSTM-LAN	89.46±0.24	72.48±0.35	66.02±0.70	82.83±0.26	77.70±0.39	91.46±0.11	93.16±0.10	72.16±0.14	85.59±0.12
	SOFTMAX	89.69±0.13	72.59±0.30	68.01±0.37	82.60±0.45	78.22±0.32	90.77±0.17	93.04±0.09	71.17±0.17	84.99±0.14
	VANILLA CRF	90.33±0.36	73.96±0.26	69.72±1.00	83.81±0.67	79.46±0.57	91.19±0.12	93.15±0.09	73.23±0.12	85.86±0.11
	TWOBILINEAR	90.11±0.25	73.69±0.30	69.31±0.67	83.53±0.44	79.16±0.42	91.45±0.09	92.98±0.09	72.28±0.39	85.57±0.19
	THREEBILINEAR	90.12±0.22	73.20±0.85	67.50±2.35	83.82±0.32	78.66±0.94	91.35±0.24	92.98±0.07	72.21±0.38	85.51±0.23
	TRILINEAR	90.19±0.19	73.39±0.40	69.69±0.60	83.70±0.19	79.24±0.35	91.44±0.19	93.00±0.08	72.28±0.56	85.57±0.28
	D-TRILINEAR	90.43±0.23	73.57±0.17	69.50±0.32	84.15±0.23	79.41±0.24	91.54±0.13	93.19±0.07	72.76±0.18	85.83±0.13
	D-QUADLINEAR	**90.44±0.07**	**75.05±0.35**	**70.49±0.68**	**84.41±0.29**	**80.09±0.35**	91.97±0.14	93.35±0.05	**74.42±0.24**	**86.58±0.14**
	D-PENTALINEAR	90.29±0.06	74.09±0.49	69.90±0.45	83.81±0.12	79.52±0.28	**91.98±0.09**	**93.43±0.07**	74.17±0.29	86.53±0.15
WORD & CHAR	BILSTM-LAN	90.71±0.20	77.18±0.28	77.83±0.90	83.97±0.84	82.42±0.55	91.84±0.10	94.20±0.09	72.33±0.16	86.12±0.12
	SOFTMAX	90.39±0.11	76.87±0.26	77.40±0.32	83.89±0.36	82.14±0.26	91.13±0.08	94.02±0.05	71.32±0.08	85.49±0.07
	VANILLA CRF	91.15±0.22	78.13±0.36	79.65±1.52	85.45±0.55	83.59±0.66	91.59±0.12	94.23±0.05	73.34±0.07	86.39±0.08
	TWOBILINEAR	90.98±0.10	77.84±0.41	79.12±0.85	85.48±0.30	83.36±0.42	91.78±0.11	93.99±0.07	72.07±0.26	85.94±0.15
	THREEBILINEAR	91.24±0.16	77.48±0.53	80.15±0.33	85.27±0.10	83.53±0.28	91.75±0.06	93.92±0.13	72.20±0.44	85.95±0.21
	TRILINEAR	91.30±0.11	77.41±0.25	79.69±0.81	85.60±0.32	83.50±0.38	91.70±0.24	94.14±0.11	72.42±0.59	86.08±0.31
	D-TRILINEAR	91.18±0.18	77.98±0.45	80.02±0.68	**85.83±0.25**	83.75±0.39	91.97±0.17	94.24±0.10	73.05±0.15	86.42±0.14
	D-QUADLINEAR	**91.34±0.12**	**78.89±0.29**	80.81±0.78	85.75±0.39	**84.20±0.39**	**92.36±0.06**	94.52±0.02	74.33±0.23	87.07±0.10
	D-PENTALINEAR	91.08±0.32	78.53±0.57	**80.99±0.55**	85.42±0.23	84.01±0.42	92.28±0.14	**94.58±0.04**	**74.48±0.41**	**87.11±0.20**

Table 8: Results on NER and Chunking tasks. BiLSTM-LAN (Cui and Zhang, 2019) is one of the current state-of-the-art sequence labeling approaches.

mance, which suggests their practical usefulness.

A.5 Complete Experimental Results

Table 8, 9, and 10 show the detailed results on the NER, Chunking and two POS tasks.

In addition, we show results of BiLSTM-LAN (Cui and Zhang, 2019), which is one of the state-of-the-art sequence labeling approaches. We run the released code of BiLSTM-LAN[1] on NER, Chunking and the two POS tagging tasks. We tune BiLSTM-LAN hyperparameters with the word-level hidden size of {100, 200, 400}, LSTM layer number of {1, 2, 3, 4}, learning rate of {0.003, 0.01, 0.03}, and decay rate of {0.03, 0.035, 0.04}. All the other hyperparameters follow their default settings. We do not report results of BiLSTM-LAN with BERT embedding because BERT is not available in the BiLSTM-LAN code.

[1] https://github.com/Nealcly/BiLSTM-LAN

		Chinese	Dutch	English	German	Hindi	Indonesian	Italian	Japanese	Avg.
WORD EMBEDDING	BILSTM-LAN	93.34±0.09	94.44±0.14	**95.09±0.08**	**94.18±0.12**	**97.02±0.10**	**91.76±0.35**	**97.62±0.20**	94.54±0.28	94.75±0.10
	SOFTMAX	93.13±0.05	94.11±0.04	94.66±0.09	93.43±0.09	96.72±0.06	91.00±0.06	97.34±0.07	95.39±0.26	94.47±0.09
	VANILLA CRF	93.10±0.08	94.14±0.14	94.70±0.04	93.41±0.13	96.74±0.05	91.23±0.05	97.41±0.05	95.50±0.24	94.53±0.10
	TWOBILINEAR	93.05±0.05	94.16±0.09	94.64±0.08	93.40±0.13	96.71±0.05	91.11±0.07	97.38±0.05	95.41±0.10	94.48±0.08
	THREEBILINEAR	93.06±0.07	94.21±0.11	94.66±0.05	93.43±0.11	96.72±0.05	91.07±0.09	97.33±0.10	95.41±0.16	94.49±0.09
	TRILINEAR	93.64±0.15	94.35±0.07	94.61±0.08	93.38±0.12	96.89±0.06	90.99±0.13	97.58±0.05	95.44±0.21	94.61±0.11
	D-TRILINEAR	93.71±0.12	94.35±0.07	94.77±0.10	93.52±0.20	96.93±0.04	91.27±0.11	97.59±0.04	95.46±0.18	94.70±0.11
	D-QUADLINEAR	**94.21±0.11**	94.59±0.08	94.85±0.09	93.84±0.13	**97.02±0.06**	91.36±0.06	**97.62±0.08**	**95.78±0.17**	**94.91±0.10**
	D-PENTALINEAR	93.91±0.21	**94.60±0.06**	94.78±0.12	93.54±0.18	97.01±0.13	91.46±0.12	97.51±0.13	95.60±0.23	94.80±0.15
WORD & CHAR	BILSTM-LAN	94.00±0.12	95.45±0.12	95.86±0.15	**94.72±0.06**	97.10±0.02	**93.80±0.03**	**98.14±0.09**	96.34±0.06	95.68±0.08
	SOFTMAX	93.67±0.07	95.28±0.12	95.92±0.02	94.28±0.11	96.96±0.10	93.48±0.06	97.88±0.04	97.14±0.14	95.58±0.08
	VANILLA CRF	93.55±0.16	95.37±0.14	96.01±0.04	94.28±0.13	96.96±0.05	93.51±0.05	97.94±0.05	97.21±0.17	95.60±0.10
	TWOBILINEAR	93.51±0.06	95.23±0.10	95.96±0.07	94.43±0.17	96.94±0.08	93.48±0.19	97.88±0.11	97.20±0.07	95.58±0.11
	THREEBILINEAR	93.50±0.10	95.25±0.10	95.92±0.09	94.47±0.10	96.89±0.04	93.35±0.15	97.84±0.04	97.08±0.11	95.54±0.09
	TRILINEAR	93.93±0.14	95.25±0.06	95.84±0.09	94.43±0.11	97.02±0.11	93.33±0.13	98.01±0.06	97.22±0.26	95.63±0.12
	D-TRILINEAR	94.03±0.12	95.30±0.10	96.04±0.05	94.34±0.06	97.06±0.02	93.45±0.12	98.09±0.03	97.21±0.17	95.69±0.08
	D-QUADLINEAR	**94.58±0.10**	**95.50±0.16**	96.01±0.13	94.51±0.09	97.19±0.06	93.53±0.13	98.02±0.04	**97.25±0.08**	**95.82±0.10**
	D-PENTALINEAR	94.50±0.21	95.45±0.13	**96.08±0.12**	94.29±0.18	**97.20±0.12**	93.60±0.12	97.98±0.05	97.23±0.08	95.79±0.13
BERT EMBEDDING	SOFTMAX	96.70±0.13	96.19±0.06	96.67±0.04	95.06±0.12	96.75±0.03	92.90±0.12	98.34±0.05	96.84±0.13	96.18±0.08
	VANILLA CRF	96.72±0.10	96.20±0.12	96.80±0.08	95.30±0.03	96.74±0.08	92.90±0.08	98.31±0.16	96.85±0.10	96.23±0.09
	TWOBILINEAR	96.54±0.09	96.02±0.19	96.78±0.09	95.33±0.05	96.71±0.04	92.88±0.10	98.36±0.06	96.86±0.08	96.18±0.09
	THREEBILINEAR	96.58±0.10	95.96±0.06	**96.85±0.06**	95.13±0.07	96.68±0.06	92.81±0.14	98.24±0.06	96.84±0.05	96.14±0.08
	TRILINEAR	96.71±0.03	96.01±0.29	96.84±0.42	95.26±0.09	96.68±0.03	92.70±0.10	98.25±0.11	96.75±0.06	96.17±0.14
	D-TRILINEAR	96.79±0.06	96.24±0.18	96.82±0.10	95.33±0.03	96.79±0.06	92.88±0.05	98.33±0.08	96.85±0.07	96.25±0.08
	D-QUADLINEAR	**96.86±0.06**	**96.25±0.09**	**96.85±0.06**	95.30±0.04	**96.87±0.03**	**92.98±0.10**	**98.37±0.07**	**97.05±0.11**	**96.32±0.07**
	D-PENTALINEAR	96.84±0.03	96.20±0.13	96.81±0.18	**95.50±0.11**	96.85±0.04	92.93±0.16	98.35±0.09	96.98±0.13	96.31±0.11

Table 9: Results on Coarse POS task. BiLSTM-LAN (Cui and Zhang, 2019) is one of the current state-of-the-art sequence labeling approaches.

		Chinese	Dutch	English	German	Hindi	Indonesian	Italian	Japanese	Avg.
WORD EMBEDDING	BILSTM-LAN	93.16±0.07	90.45±0.25	**94.60±0.12**	**96.41±0.04**	**96.54±0.02**	94.11±0.11	**97.62±0.05**	92.69±0.45	94.45±0.14
	SOFTMAX	93.33±0.09	92.17±0.11	94.40±0.05	96.22±0.03	96.25±0.08	94.66±0.03	97.26±0.10	95.02±0.14	94.91±0.08
	VANILLA CRF	93.22±0.11	92.13±0.09	94.41±0.09	96.22±0.06	96.39±0.09	94.60±0.05	97.27±0.06	94.86±0.12	94.89±0.08
	TWOBILINEAR	93.19±0.13	92.03±0.09	94.17±0.05	96.15±0.04	96.34±0.08	94.72±0.06	97.24±0.04	94.67±0.35	94.81±0.11
	THREEBILINEAR	93.29±0.09	92.12±0.06	94.12±0.10	96.20±0.05	96.39±0.06	94.82±0.12	97.30±0.06	94.73±0.14	94.87±0.09
	TRILINEAR	93.79±0.08	91.91±0.17	94.20±0.09	96.27±0.05	96.42±0.10	94.60±0.11	97.49±0.12	94.80±0.32	94.94±0.13
	D-TRILINEAR	93.78±0.02	92.21±0.28	94.29±0.09	96.27±0.08	96.46±0.05	94.77±0.09	97.50±0.05	**95.25±0.10**	95.07±0.10
	D-QUADLINEAR	**94.24±0.07**	**92.36±0.18**	94.36±0.06	96.28±0.07	**96.54±0.07**	**94.94±0.09**	**97.62±0.03**	95.20±0.06	**95.19±0.08**
	D-PENTALINEAR	94.12±0.18	92.21±0.22	94.30±0.34	96.25±0.46	96.39±0.08	94.70±0.09	97.55±0.10	95.03±0.08	95.07±0.19
WORD & CHAR	BILSTM-LAN	93.88±0.11	92.38±0.45	95.58±0.08	**97.14±0.05**	96.69±0.03	94.80±0.05	**98.02±0.01**	94.75±0.23	95.41±0.13
	SOFTMAX	93.86±0.10	93.05±0.10	95.59±0.08	97.07±0.06	96.55±0.03	94.93±0.06	97.85±0.05	96.85±0.07	95.72±0.07
	VANILLA CRF	93.69±0.08	**93.21±0.14**	95.58±0.11	97.08±0.11	96.59±0.03	94.80±0.08	97.89±0.06	96.75±0.27	95.70±0.11
	TWOBILINEAR	93.54±0.08	92.91±0.22	95.60±0.13	97.03±0.05	96.59±0.06	94.93±0.05	97.83±0.09	96.67±0.11	95.64±0.10
	THREEBILINEAR	93.62±0.08	93.03±0.13	95.55±0.05	97.09±0.06	96.61±0.04	94.97±0.09	97.79±0.06	96.63±0.19	95.66±0.09
	TRILINEAR	94.02±0.07	92.99±0.29	95.58±0.09	97.10±0.03	96.61±0.07	94.84±0.11	97.98±0.06	96.59±0.15	95.71±0.11
	D-TRILINEAR	94.05±0.15	92.83±0.17	**95.69±0.04**	97.11±0.05	96.64±0.02	94.95±0.08	97.96±0.03	96.80±0.08	95.75±0.08
	D-QUADLINEAR	**94.49± 0.11**	93.03±0.18	95.64±0.05	97.10±0.03	**96.78±0.05**	95.04±0.09	**98.02±0.05**	96.90±0.07	**95.88±0.08**
	D-PENTALINEAR	94.20±0.12	93.03±0.20	95.63±0.02	97.04±0.05	96.70±0.02	**95.08±0.05**	97.91±0.30	**96.93±0.12**	95.82±0.11
BERT EMBEDDING	SOFTMAX	96.54±0.05	93.58±0.13	96.39±0.05	97.52±0.06	96.26±0.10	91.56±0.06	98.24±0.03	96.16±0.06	95.83±0.07
	VANILLA CRF	96.53±0.11	93.57±0.12	96.38±0.08	97.47±0.03	96.31±0.08	**91.75±0.10**	98.24±0.09	96.25±0.11	95.81±0.09
	TWOBILINEAR	96.47±0.05	93.45±0.15	**96.42±0.05**	97.52±0.06	96.32±0.06	**91.75±0.15**	98.16±0.04	96.24±0.15	95.79±0.09
	THREEBILINEAR	96.45±0.10	93.26±0.21	96.32±0.07	97.50±0.06	96.27±0.03	91.73±0.20	98.18±0.04	96.19±0.18	95.74±0.11
	TRILINEAR	96.60±0.08	93.53±0.13	96.22±0.05	97.55±0.06	96.22±0.08	90.97±0.24	98.22±0.07	96.09±0.14	95.67±0.11
	D-TRILINEAR	96.68±0.07	93.68±0.19	96.31±0.08	**97.57±0.06**	96.37±0.07	91.55±0.09	98.26±0.04	96.35±0.09	95.74±0.11
	D-QUADLINEAR	96.74±0.10	**93.69±0.21**	96.39±0.05	97.56±0.03	**96.41±0.07**	91.65±0.18	**98.29±0.02**	**96.44±0.05**	**95.90±0.09**
	D-PENTALINEAR	**97.10±0.02**	93.40±0.03	96.40±0.01	97.54±0.09	96.33±0.11	91.46±0.20	**98.29±0.06**	96.30±0.08	95.85±0.08

Table 10: Results on Fine POS task. BiLSTM-LAN (Cui and Zhang, 2019) is one of the current state-of-the-art sequence labeling approaches.