

# AIN: Fast and Accurate Sequence Labeling with Approximate Inference Network

Xinyu Wang<sup>◇</sup>, Yong Jiang<sup>†</sup>, Nguyen Bach<sup>†</sup>, Tao Wang<sup>†</sup>, Zhongqiang Huang<sup>†</sup>, Fei Huang<sup>†</sup>, Kewei Tu<sup>◇\*</sup>

<sup>◇</sup>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

<sup>†</sup>DAMO Academy, Alibaba Group

{wangxy1, tukw}@shanghaitech.edu.cn

{yongjiang.jy, nguyen.bach, leeo.wangt, z.huang, f.huang}@alibaba-inc.com

## Abstract

The linear-chain Conditional Random Field (CRF) model is one of the most widely-used neural sequence labeling approaches. Exact probabilistic inference algorithms such as the forward-backward and Viterbi algorithms are typically applied in training and prediction stages of the CRF model. However, these algorithms require sequential computation that makes parallelization impossible. In this paper, we propose to employ a parallelizable approximate variational inference algorithm for the CRF model. Based on this algorithm, we design an approximate inference network that can be connected with the encoder of the neural CRF model to form an end-to-end network, which is amenable to parallelization for faster training and prediction. The empirical results show that our proposed approaches achieve a 12.7-fold improvement in decoding speed with long sentences and a competitive accuracy compared with the traditional CRF approach.

## 1 Introduction

Sequence labeling assigns each token with a label in a sequence. Tasks such as Named Entity Recognition (NER) (Sundheim, 1995), Part-Of-Speech (POS) tagging (DeRose, 1988) and chunking (Tjong Kim Sang and Buchholz, 2000) can all be formulated as sequence labeling tasks. BiLSTM-CRF (Huang et al., 2015; Lample et al., 2016; Ma and Hovy, 2016) is one of the most successful neural sequence labeling architectures. It feeds pre-trained (contextual) word representations into a single layer bi-directional LSTM (BiLSTM) encoder to extract contextual features and then feeds these features into a CRF (Lafferty et al., 2001) decoder layer to produce final predictions. The CRF

layer is a linear-chain structure that models the relation between neighboring labels. In the traditional CRF approach, exact probabilistic inference algorithms such as the forward-backward and Viterbi algorithms are applied for training and prediction respectively. In many sequence labeling tasks, the CRF layer leads to better results than the simpler method of predicting each label independently.

In practice, we sometimes require very fast sequence labelers for training (e.g., on huge datasets like WikiAnn (Pan et al., 2017)) and prediction (e.g. for low latency online serving). The BiLSTM encoder and the CRF layer both contain sequential computation and require  $O(n)$  time over  $n$  input words even when parallelized on GPU. A common practice to improve the speed of the encoder is to replace the BiLSTM with a CNN structure (Collobert et al., 2011; Strubell et al., 2017), distill larger encoders into smaller ones (Tsai et al., 2019; Mukherjee and Awadallah, 2020) or in other settings (Tu and Gimpel, 2018; Yang et al., 2018; Tu and Gimpel, 2019; Cui and Zhang, 2019). The CRF layer, however, is more difficult to replace because of its superior accuracy compared with faster alternatives in many tasks.

In order to achieve sublinear time complexity on the CRF layer, we must parallelize the CRF prediction over the tokens. In this paper, we apply Mean-Field Variational Inference (MFVI) to approximately decode the linear-chain CRF. MFVI iteratively passes messages among neighboring labels to update their distributions locally. Unlike the exact probabilistic inference algorithms, MFVI can be parallelized over different positions in the sequence, achieving time complexity that is constant in  $n$  with full parallelization. Previous work (Zheng et al., 2015) showed that such an algorithm can be unfolded as an RNN for grid CRF structure. We expand on the work for the linear-chain CRF structure and unfold the algorithm as an RNN

\*Kewei Tu is the corresponding author. This work was conducted when Xinyu Wang was interning at Alibaba DAMO Academy.

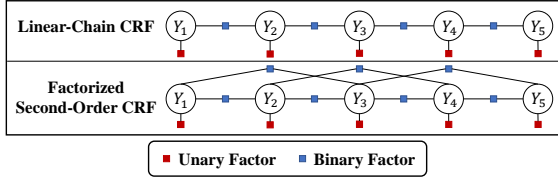


Figure 1: Factor graphs of different CRFs.  $Y_i$  is the random variable representing the  $i$ -th label.

which can be connected with the encoder to form an end-to-end neural network that is amenable to parallelization for both training and prediction. We call the unfolded RNN an approximate inference network (AIN). In addition to linear-chain CRFs, we also apply AIN to factorized second-order CRF models, which consider relations between more neighboring labels. Our empirical results show that AIN significantly improves the speed and achieves competitive accuracy against the traditional CRF approach on 4 tasks with 15 datasets.

## 2 Approaches

Given an input sequence with  $n$  tokens  $\mathbf{x} = [x_1, x_2, \dots, x_n]$  and a corresponding label sequence  $\mathbf{y} = [y_1, y_2, \dots, y_n]$  with a label set of size  $L$ , the conditional probability of  $\mathbf{y}$  given  $\mathbf{x}$  specified by a CRF with position-wise factorization is:

$$P(\mathbf{y}|\mathbf{x}) = \frac{\exp\{\sum_{i=1}^n \psi(\mathbf{x}, \mathbf{y}, i)\}}{\sum_{\mathbf{y}' \in \mathcal{Y}(\mathbf{x})} \exp\{\sum_{i=1}^n \psi(\mathbf{x}, \mathbf{y}', i)\}}$$

where  $\mathcal{Y}(\mathbf{x})$  is the set of all possible label sequences for  $\mathbf{x}$  and  $\psi(\mathbf{x}, \mathbf{y}, i)$  is a potential function.

In the simplest case, the potential function is just a softmax function that outputs the distribution of each label independently. We call it the MaxEnt approach. In a typical linear-chain CRF, the potential function is decomposed into a unary potential  $\psi_u$  and a binary potential  $\psi_b$  (called the emission and transition functions respectively):

$$\psi(\mathbf{x}, \mathbf{y}, i) = \psi_u(\mathbf{x}, y_i) + \psi_b(y_{i-1}, y_i) \quad (1)$$

$$\psi_u(\mathbf{x}, y_i) = \mathbf{r}_i \mathbf{W} \mathbf{v}_{y_i}$$

$$\psi_b(y_{i-1}, y_i) = \mathbf{U}_{y_{i-1}, y_i} \quad (2)$$

where  $\mathbf{r}_i$  is the contextual feature of  $x_i$  output from the CNN or BiLSTM encoder with dimension  $d$ ,  $\mathbf{v}_{y_i}$  is a one-hot vector for label  $y_i$ ,  $\mathbf{W}$  is a  $d \times L$  matrix and  $\mathbf{U}$  is an  $L \times L$  matrix containing the transition scores between two labels. The factor

graph of a linear-chain CRF is shown at the top of Figure 1.

The exact probabilistic inference algorithms (Viterbi and forward-backward) for the CRF layer are significantly slower than the MaxEnt approach. They take  $O(nL^2)$  and  $O(n \log L)$  time on CPU and GPU<sup>1</sup> respectively, while the decoder in MaxEnt takes  $O(nL)$  and  $O(\log L)$ .

### 2.1 AIN on Linear-Chain CRF

In order to speed up the training and prediction time of the CRF layer, we propose the approximate inference network (AIN), which is a neural network derived from MFVI for approximate decoding in linear-chain CRF.

MFVI approximates the distribution  $P(\mathbf{y}|\mathbf{x})$  with a factorized distribution  $Q(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^n Q_i(y_i|\mathbf{x})$  and update it iteratively to minimize the KL divergence  $KL(Q||P)$ . The update formula of  $Q_i(y_i|\mathbf{x})$  at iteration  $m$  is:

$$s(i, j, k) := \sum_{y_i=1}^L Q_i^{k-1}(y_i|\mathbf{x}) \psi_b(y_{\min\{i,j\}}, y_{\max\{i,j\}})$$

$$Q_i^m(y_i|\mathbf{x}) \propto \exp\{\psi_u(\mathbf{x}, y_i) + s(i-1, i, m) + s(i+1, i, m)\}$$

where  $s(i, j, k)$  represents the message from node  $i$  to node  $j$  at time step  $k$ .  $Q_i^0(y_i|\mathbf{x})$  is set by normalizing the unary potential  $\psi_u(\mathbf{x}, y_i)$ . Upon convergence, the label sequence with the highest approximate probability  $Q(\mathbf{y}|\mathbf{x})$  can be found by optimizing  $Q_i(y_i|\mathbf{x})$  at each position  $i$ :

$$\hat{y}_i = \underset{y_i \in \{1, \dots, L\}}{\operatorname{argmax}} Q_i(y_i|\mathbf{x})$$

Similar to Zheng et al. (2015), we unfold the MFVI algorithm as a recurrent neural network that is parameterized by the linear-chain CRF potential functions. We fix the number of iterations to  $M$  and call the resulting network AIN. AIN can be connected with the encoding network that computes the potential functions and together they form an end-to-end neural network. However, different from previous work (Krähenbühl and Koltun, 2011; Zheng et al., 2015) using the MFVI algorithm for intractable problems of the grid-structured probabilistic models to get better accuracy, we propose to

<sup>1</sup>We assume that the number of threads is enough for full parallelization on GPU and the parallel reduction (e.g., sum and max) for a  $L$  elements vector takes  $O(\log L)$  time (Harris et al., 2007).

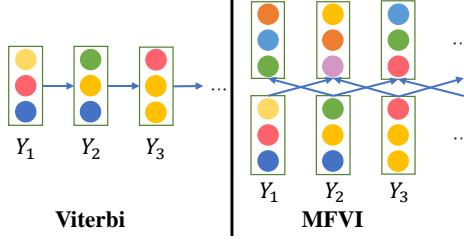


Figure 2: Illustration of the computation graphs for the Viterbi decoding and one iteration of our MFVI inference on the CRF model.  $Y_i$  is the random variable representing the  $i$ -th label with three possible values. The illustrated vectors represent Viterbi scores and  $Q_i$  distributions respectively.

employ the MFVI algorithm to accelerate the speed of tractable problems of the sequence-structured probabilistic models.

The time complexity of each iteration of the MFVI algorithm is  $O(nL^2)$ , which is on par with the time complexity of the exact probabilistic inference algorithms. However, in each iteration, the update of each distribution  $Q_i(y_i|\mathbf{x})$  depends only on its two neighboring distributions from the previous iteration, so each iteration can be parallelized over positions. A comparison between the Viterbi algorithm and the MFVI algorithm is shown in Figure 2. The time complexity of our AIN decoder with full GPU parallelization is  $O(M \log L)$ , while the time complexity of the exact probabilistic inference algorithms with GPU parallelization is  $O(n \log L)$ . We set the value of  $M$  to 3s, which is much smaller than the typical value of sequence length  $n$ .

## 2.2 AIN on Factorized Second-Order CRF

We can extend AIN to the second-order CRF with a ternary potential function over every three consecutive labels. In the second-order CRF, the potential function in Eq. 1 becomes:

$$\psi(\mathbf{x}, \mathbf{y}, i) = \psi_u(\mathbf{x}, y_i) + \psi_t(y_{i-2}, y_{i-1}, y_i)$$

However, the second-order CRF has space and time complexity that is cubic in  $L$ . Therefore, we factorize its ternary potential function and reduce its complexity to be quadratic in  $L$ :

$$\begin{aligned} \psi_t(y_{i-2}, y_{i-1}, y_i) &= \psi_{b'}(y_{i-2}, y_i) + \psi_b(y_{i-1}, y_i) \\ \psi_{b'}(y_{i-2}, y_i) &= \tilde{\mathbf{U}}_{y_{i-2}, y_i} \end{aligned}$$

where the matrix  $\tilde{\mathbf{U}}$  has the same shape as  $\mathbf{U}$  in Eq. 2. The factor graph of our factorized second-order CRF is shown at the bottom of Figure 1. The

update formula is similar to that of our first-order approach but with more neighbors:

$$\begin{aligned} Q_i^m(y_i|\mathbf{x}) &\propto \exp\{\psi_u(\mathbf{x}, y_i) + s'(i-2, i, m) \\ &\quad + s(i-1, i, m) + s(i+1, i, m) + s'(i+2, i, m)\} \end{aligned}$$

where  $s'(i, j, k)$  has a similar definition as  $s(i, j, k)$  by replacing  $\psi_b$  with  $\psi_{b'}$ . The time complexity of this approach is also  $O(nL^2)$  for each iteration and  $O(M \log L)$  with full GPU parallelization for  $M$  iterations. Following the first approach, we also unfold MFVI of this approach as an AIN.

## 2.3 Learning

Given a sequence  $\mathbf{x}$  with corresponding gold labels  $\mathbf{y}^* = \{y_1^*, \dots, y_n^*\}$ , the learning objective of our approaches is:

$$\mathcal{L}_{\text{NLL}} = - \sum_{i=1}^n \log Q_i^M(y_i^*|\mathbf{x})$$

Since AINs are end-to-end neural networks, the objective function can be optimized by any gradient-based method in an end-to-end manner.

## 3 Experiments

### 3.1 Settings

**Datasets** We evaluate our approaches on four tasks: NER, POS tagging, chunking and slot filling. For NER, we use the corpora from the CoNLL 2002 and CoNLL 2003 shared tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003). For POS tagging, we use universal POS tag annotations with 8 languages from the Universal Dependencies (UD) (Nivre et al., 2018) dataset. For chunking, we use the corpora from the CoNLL 2003 shared task. We use the Air Travel Information System (ATIS) (Hemphill et al., 1990) dataset for slot filling.

**Encoder** In our experiments, we use three types of encoders. The first is a BiLSTM fed with word and character embeddings, which captures contextual information globally. The second is a single layer CNN with only word embedding as input, which captures contextual information locally. The third is a single linear layer with word embeddings as input, which does not capture any contextual information. We use these settings for a better understanding of how the decoders perform on each task when the encoders capture different levels of contextual information.

# Words	WORD-CHAR-BiLSTM								WORD-CNN			
	Training				Prediction				Training		Prediction	
	32		128		32		128		32	128	32	128
	All	Dec.	All	Dec.	All	Dec.	All	Dec.	All	All	All	All
MaxEnt*	6.8×	-	13.1×	-	3.0×	-	5.9×	-	12.9×	40.1×	6.3×	18.6×
AIN-1O	4.3×	7.7×	10.2×	31.4×	1.7×	2.4×	4.4×	12.7×	5.6×	21.5×	2.4×	6.8×
AIN-F2O	3.5×	5.3×	8.7×	20.1×	1.5×	1.9×	4.1×	10.6×	4.4×	16.7×	1.8×	5.5×

Table 1: Relative speedup over the **CRF** model with 10,000 sentences of 32/128 words. **All** represents the speed of the full model. **Dec.** represents the speed of decoder. \*: For reference.

	WORD-CHAR-BiLSTM					WORD-CNN					WORD ONLY				
	NER	POS	Chunk	SF	Avg.	NER	POS	Chunk	SF	Avg.	NER	POS	Chunk	SF	Avg.
MaxEnt*	83.74	94.84	92.58	95.47	91.65	75.19	94.00	87.05	91.07	86.83	52.27	90.53	78.17	62.93	70.98
CRF	84.17	94.91	<b>92.88</b>	95.52	91.87	<b>79.44</b>	94.26	<b>89.21</b>	92.24	<b>88.79</b>	<b>72.28</b>	92.79	<b>89.39</b>	76.82	82.82
AIN-1O	<b>84.22</b>	<b>94.97</b>	92.87	<b>95.59</b>	<b>91.91</b>	78.47	94.29	88.86	92.18	88.45	70.23	92.84	88.69	88.76	85.13
AIN-F2O	84.11	94.91	92.85	95.58	91.86	78.71	<b>94.32</b>	88.75	<b>92.26</b>	88.51	71.16	<b>93.03</b>	88.80	<b>88.86</b>	<b>85.46</b>

Table 2: Averaged F1 score and accuracy on four tasks. **SF** represents the slot filling task. \*: For reference.

**Decoder** We use the MaxEnt approach, the traditional CRF approach and AINs with the first-order and factorized second-order CRFs for decoding. We denote these approaches by **MaxEnt**, **CRF**, **AIN-1O** and **AIN-F2O** respectively. We set the iteration number  $M$  to 3 in AINs because we find that more iterations do not result in further improvement in accuracy.

### 3.2 Results

**Speed** We report the relative speed improvements over the **CRF** model based on our PyTorch (Paszke et al., 2019) implementation run on a GPU server with Nvidia Tesla V100. Following Tsai et al. (2019), we report the training and prediction speed with 10,000 sentences of 32 and 128 words, respectively. The results (Table 1) show that AINs are significantly faster than **CRF** in terms of both the full model speed and the decoder speed. The speed advantage of AINs is especially prominent with long sentences, suggesting their usefulness in tasks like document-level NER.

**Accuracy** We run each approach on each dataset for 5 times and compute its average accuracy. Because of space limit, we report the accuracy averaged over all the datasets for each task in Table 2. Please refer to the supplementary material for the complete results. AINs achieve competitive overall accuracy with **CRF**, even though AINs take significantly less time than **CRF**. With the BiLSTM encoder which has the capability to capture global contextual information, AINs achieves almost the same average accuracy as **CRF**, demonstrating that AINs performing approximate inference with local

contextual information are competitive with **CRF** with globally exact decoding. With the CNN encoder that encodes local contextual information, AINs are inferior to **CRF** because both the CNN layer and our approaches utilize only local information. Without any contextual encoders (Word Only), the accuracy of these decoders vary significantly over tasks. For NER and chunking, **CRF** is the strongest, but our approaches only marginally underperform **CRF** while significantly outperform **MaxEnt**. For POS tagging and slot filling, our approaches outperform **CRF**, which implies that local information might be more beneficial for these tasks. Comparing **AIN-1O** and **AIN-F2O**, **AIN-F2O** is stronger when the encoder is weak, but their performance gap becomes smaller and eventually disappears when the encoder gets stronger.

## 4 Conclusion

In this paper, we propose approximate inference networks (AIN) that use Mean-Field Variational Inference (MFVI) instead of exact probabilistic inference algorithms such as the forward-backward and Viterbi algorithms for training and prediction on the conditional random field for sequence labeling. The MFVI algorithm can be unfolded as a recurrent neural network and connected with the encoder to form an end-to-end neural network. AINs can be parallelized over different positions in the sequence. Empirical results show that AINs are significantly faster than traditional CRF and do very well in tasks that require more local information. Our approaches achieve competitive accuracy on 4 tasks with 15 datasets over three encoder types.



## Acknowledgements

The authors wish to thank Chao Lou for his helpful comments and suggestions. This work was supported by the National Natural Science Foundation of China (61976139).

## 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.
- Joakim Nivre et al. 2018. [Universal dependencies 2.2](#). LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.
- 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.
- Steven J. DeRose. 1988. [Grammatical category disambiguation by statistical optimization](#). *Computational Linguistics*, 14(1).
- Mark Harris et al. 2007. Optimizing parallel reduction in cuda. *Nvidia developer technology 2.4*.
- Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. [The ATIS spoken language systems pilot corpus](#). In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.
- Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.
- Philipp Krähenbühl and Vladlen Koltun. 2011. [Efficient inference in fully connected crfs with gaussian edge potentials](#). In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 109–117. Curran Associates, Inc.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, ICML 2001*, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- 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.
- 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.
- Subhabrata Mukherjee and Ahmed Awadallah. 2020. Tinymbert: Multi-stage distillation framework for massive multi-lingual ner. *arXiv preprint arXiv:2004.05686*.
- Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. [Cross-lingual name tagging and linking for 282 languages](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. 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 8024–8035. Curran Associates, Inc.
- 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.
- Beth M. Sundheim. 1995. Named entity task definition, version 2.1. In *Proceedings of the Sixth Message Understanding Conference*, pages 319–332.
- 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.

Henry Tsai, Jason Riesa, Melvin Johnson, Naveen Ariavazhagan, Xin Li, and Amelia Archer. 2019. [Small and practical BERT models 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 3632–3636, Hong Kong, China. Association for Computational Linguistics.

Lifu Tu and Kevin Gimpel. 2018. [Learning approximate inference networks for structured prediction](#). In *International Conference on Learning Representations*.

Lifu Tu and Kevin Gimpel. 2019. [Benchmarking approximate inference methods for neural structured prediction](#). 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 3313–3324, Minneapolis, Minnesota. 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.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. 2015. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537.

## A Appendix

### A.1 Datasets

**Named Entity Recognition (NER)** We use the corpora from the CoNLL 2002 and CoNLL 2003 shared tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003), which contain

four languages in total. We use the standard training/development/test split for experiments.<sup>2</sup>

**Chunking** The chunking datasets are also from the CoNLL 2003 shared task (Tjong Kim Sang and De Meulder, 2003) that contains English and German datasets. We use the same standard split as in NER.

**Part-Of-Speech (POS) Tagging** Universal Dependencies<sup>3</sup> (UD) (Nivre et al., 2018) contains syntactically annotated corpora of over 70 languages. We use universal POS tag annotations with 8 languages for experiments, the list of treebanks is shown in Table 3. We use the standard training/development/test split for experiments.

**Slot Filling** Slot filling is a task that interprets user commands by extracting relevant slots, which can be formulated as a sequence labeling task. We use the Air Travel Information System (ATIS) (Hemphill et al., 1990) dataset for the task and use the same dataset split as [this repository](#).

### A.2 Settings

**Embeddings** For word embeddings in NER, chunking and slot filling experiments, we use the same word embedding as Lample et al. (2016) except that we use *fastText* (Bojanowski et al., 2017) embedding for Dutch which we find significantly improves the accuracy (more than 5 F1 scores on CoNLL NER). We use *fastText* embeddings for all UD tagging experiments. For character embedding, we use a single layer character CNN with a hidden size of 50, because Yang et al. (2018) empirically showed that it has competitive performance with character LSTM. We concatenate the word embedding and character CNN output for the final word representation.

**Hyper-parameters** For the hyper-parameters, we follow the settings of previous work (Akbik et al., 2018). We use Stochastic Gradient Descent for optimization with a fixed learning rate of 0.1 and a batch size of 32. We fix the hidden size of the CNN and BiLSTM layer to 512 and 256 respectively, and the kernel size of CNN to 3. We anneal the learning rate by 0.5 if there is no improvement in the development sets for 10 epochs when training. For the value of maximum iteration  $M$ , we

<sup>2</sup><https://www.clips.uantwerpen.be/conll2003/ner/>

<sup>3</sup><https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2837>

Language	Treebank
de	GSD
en	EWT
es	GSD
fr	Sequoia
it	PoSTWITA
nl	LassySmall
sl	SST
sv	LinES

Table 3: The list of treebank that we used in UD POS tagging.

tried from 1 to 5 and compared the accuracy on the English NER dataset over 5 runs for different  $M$ .

**Evaluation** We use F1 score to evaluate the NER, slot filling and chunking tasks and use accuracy to evaluate the POS tagging task. We convert the BIO format into BIOES format for NERs, slot filling and chunking datasets and use the official release of CoNLL evaluation script<sup>4</sup> to evaluate the F1 score.

### A.3 Detailed Results

The detailed results for the four tasks are shown in Table 4 and 5. We use ISO 639-1 codes<sup>5</sup> to represent each language for simplification.

<sup>4</sup><https://github.com/chakki-works/seqeval/blob/master/tests/conlleval.pl>

<sup>5</sup>[https://en.wikipedia.org/wiki/List\\_of\\_ISO\\_639-1\\_codes](https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes)

	POS TAGGING								
Model	de	en	es	fr	it	nl	sl	sv	avg
WORD-CHAR-BiLSTM									
MaxEnt	94.19±0.04	95.70±0.07	<b>96.44</b> ±0.05	98.00±0.06	92.76±0.17	95.09±0.13	90.96±0.69	95.56±0.07	94.84
CRF	<b>94.27</b> ±0.11	95.71±0.06	96.37±0.09	<b>98.06</b> ±0.04	92.87±0.15	95.10±0.17	91.38±1.12	95.55±0.09	94.91
AIN-1	94.23±0.06	95.73±0.05	96.39±0.10	98.04±0.10	<b>93.13</b> ±0.19	95.10±0.20	<b>91.42</b> ±0.28	<b>95.69</b> ±0.05	<b>94.97</b>
AIN-F2	94.11±0.22	<b>95.76</b> ±0.05	96.34±0.05	97.99±0.11	92.87±0.20	<b>95.24</b> ±0.16	91.38±0.44	95.59±0.07	94.91
WORD CNN									
MaxEnt	92.36±0.19	93.99±0.12	95.91±0.06	97.62±0.05	92.49±0.08	94.51±0.08	91.39±0.18	93.76±0.15	94.00
CRF	93.06±0.17	<b>94.22</b> ±0.10	<b>96.09</b> ±0.08	97.68±0.07	92.63±0.05	94.63±0.16	91.65±0.23	94.15±0.17	94.26
AIN-1	<b>93.11</b> ±0.14	94.21±0.05	96.02±0.06	97.73±0.05	92.64±0.06	94.58±0.07	91.77±0.20	94.26±0.11	94.29
AIN-F2	92.99±0.12	94.17±0.13	96.00±0.04	<b>97.75</b> ±0.03	<b>92.69</b> ±0.06	<b>94.68</b> ±0.04	<b>91.84</b> ±0.23	<b>94.47</b> ±0.10	<b>94.32</b>
WORD ONLY									
MaxEnt	89.44±0.08	87.57±0.12	93.02±0.05	94.82±0.07	89.23±0.08	91.63±0.17	88.56±0.24	90.01±0.06	90.53
CRF	91.55±0.13	91.04±0.22	94.64±0.05	96.65±0.10	91.56±0.05	93.28±0.12	90.02±0.24	93.55±0.09	92.79
AIN-1	91.53±0.08	91.47±0.09	94.77±0.05	96.67±0.05	91.62±0.03	<b>93.46</b> ±0.03	89.65±0.37	93.54±0.10	92.84
AIN-F2	<b>91.75</b> ±0.09	<b>91.76</b> ±0.12	<b>94.82</b> ±0.03	<b>96.95</b> ±0.05	<b>91.63</b> ±0.06	93.32±0.13	<b>90.17</b> ±0.23	<b>93.86</b> ±0.09	<b>93.03</b>

Table 4: Averaged accuracy scores on POS tagging.

Models	NER					CHUNK			SF
	de	en	es	nl	avg	de	en	avg	en
<b>WORD-CHAR-BiLSTM</b>									
MaxEnt	75.63±0.23	91.00±0.23	84.53±0.50	83.78±0.38	83.74	93.80±0.14	91.36±0.10	92.58	95.47±0.06
CRF	<b>76.46</b> ±0.24	91.14±0.16	85.29±0.36	83.80±0.33	84.17	<b>94.06</b> ±0.07	91.70±0.08	<b>92.88</b>	95.52±0.10
AIN-1O	76.34±0.34	91.07±0.10	<b>85.37</b> ±0.07	<b>84.12</b> ±0.53	<b>84.22</b>	94.03±0.02	<b>91.71</b> ±0.05	92.87	<b>95.59</b> ±0.11
AIN-F2O	76.17±0.28	<b>91.22</b> ±0.20	85.30±0.32	83.76±0.57	84.11	94.02±0.04	91.69±0.08	92.85	95.58±0.14
<b>WORD CNN</b>									
MaxEnt	69.40±0.15	84.86±0.41	70.02±0.62	76.46±0.28	75.19	88.29±0.10	85.80±0.65	87.05	91.07±0.01
CRF	<b>71.12</b> ±0.25	<b>87.58</b> ±0.21	<b>80.34</b> ±0.58	<b>78.70</b> ±0.30	<b>79.44</b>	<b>89.68</b> ±0.21	<b>88.73</b> ±0.18	<b>89.21</b>	92.24±0.27
AIN-1O	70.00±0.28	86.94±0.43	78.95±0.51	77.98±0.38	78.47	89.21±0.11	88.51±0.15	88.86	92.18±0.14
AIN-F2O	70.08±0.92	87.01±0.22	79.80±0.38	77.95±0.47	78.71	89.33±0.12	88.16±0.30	88.75	<b>92.26</b> ±0.26
<b>WORD ONLY</b>									
MaxEnt	36.24±1.77	63.68±1.08	52.42±1.73	56.73±0.77	52.27	81.21±0.33	75.14±0.41	78.17	62.93±0.33
CRF	55.10±2.87	<b>81.76</b> ±0.39	<b>76.53</b> ±0.80	<b>75.71</b> ±0.39	<b>72.28</b>	<b>90.56</b> ±0.24	<b>88.21</b> ±0.34	<b>89.39</b>	76.82±0.57
AIN-1O	<b>57.25</b> ±2.16	79.68±0.25	70.44±0.72	73.55±0.21	70.23	90.04±0.18	87.35±0.29	88.69	88.76±0.65
AIN-F2O	56.36±5.97	81.16±0.37	73.03±1.86	74.09±0.24	71.16	90.04±0.15	87.56±0.24	88.8	<b>88.86</b> ±0.41

Table 5: Averaged F1 scores on NER, chunking and slot filling for each language. **SF** represents the slot filling task. \*: For reference.