More Embeddings, Better Sequence Labelers?

Xinyu Wang[◇‡], **Yong Jiang**[†*], **Nguyen Bach**[†],
Tao Wang[†], **Zhongqiang Huang**[†], **Fei Huang**[†], **Kewei Tu**[◇*]
[◇]School of Information Science and Technology, ShanghaiTech University
[◇]Shanghai Engineering Research Center of Intelligent Vision and Imaging
[◇]University of Chinese Academy of Sciences
[†]DAMO Academy, Alibaba Group
{wangxy1,tukw}@shanghaitech.edu.cn
{yongjiang.jy,nguyen.bach}@alibaba-inc.com
{leeo.wangt,z.huang,f.huang}@alibaba-inc.com

Abstract

Recent work proposes a family of contextual embeddings that significantly improves the accuracy of sequence labelers over non-contextual embeddings. However, there is no definite conclusion on whether we can build better sequence labelers by combining different kinds of embeddings in various settings. In this paper, we conduct extensive experiments on 3 tasks over 18 datasets and 8 languages to study the accuracy of sequence labeling with various embedding concatenations and make three observations: (1) concatenating more embedding variants leads to better accuracy in rich-resource and cross-domain settings and some conditions of low-resource settings; (2) concatenating contextual sub-word embeddings with contextual character embeddings hurts the accuracy in extremely low-resource settings; (3) based on the conclusion of (1), concatenating additional similar contextual embeddings cannot lead to further improvements. We hope these conclusions can help people build stronger sequence labelers in various settings.

1 Introduction

In recent years, sequence labelers equipped with contextual embeddings have achieved significant accuracy improvement (Peters et al., 2018; Akbik et al., 2018; Devlin et al., 2019; Martin et al., 2019) over approaches that use static non-contextual word embeddings (Mikolov et al., 2013) and character embeddings (Santos and Zadrozny, 2014). Different types of embeddings have different inductive biases to guide the learning process. However, little work has been done to study how to concatenate these contextual embeddings and non-contextual embeddings to build better sequence labelers in multilingual, low-resource, or cross-domain settings over various sequence labeling tasks. In this paper, we empirically investigate the effectiveness of concatenating various kinds of embeddings for multilingual sequence labeling and try to answer the following questions:

1. In rich-resources settings, does combining different kinds of contextual embeddings result in a better sequence labeler? Are non-contextual embeddings helpful when the models are equipped with contextual embeddings?

2. When we train models in low-resource and cross-domain settings, do the conclusions from the rich-resource settings still hold?

3. Can sequence labelers automatically learn the importance of each kind of embeddings when they are concatenated?

2 Model Architecture

2.1 Sequence Labeling

We use the BiLSTM structure for all the sequence labeling tasks, which is one of the most popular approaches to sequence labeling (Huang et al., 2015; Ma and Hovy, 2016). Given a n word sentence $\mathbf{x} = \{x_1, \cdots, x_n\}$ and L kinds of embeddings, we feed the sentence to generate the l-th kind of word embeddings $\{\mathbf{e}_1^l, \cdots, \mathbf{e}_n^l\}$:

$$\mathbf{e}_i^l = \text{embed}^l(\mathbf{x})$$

We concatenate these embeddings to generate the word representations $\{\mathbf{r}_1, \cdots, \mathbf{r}_n\}$ as the input of the BiLSTM layer:

$$\mathbf{r}_i = \mathbf{e}_i^1 \oplus \cdots \oplus \mathbf{e}_i^L$$

where $\oplus$ represents the vector concatenation operation. We feed the word representations into a single-layer BiLSTM to generate the contextual hidden

[*] Yong Jiang and Kewei Tu are the corresponding authors.
[‡]: This work was conducted when Xinyu Wang was interning at Alibaba DAMO Academy.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 3992–4006
November 16 - 20, 2020.

layer of each word. Then we use either a Softmax layer (the MaxEnt approach) or a Conditional Random Field layer (the CRF approach) (Lafferty et al., 2001; Lample et al., 2016; Ma and Hovy, 2016) fed with the hidden layers to generate the conditional probability $p(\mathbf{y}|\mathbf{x})$. Given the corresponding sequence of gold labels $\mathbf{y}^* = \{y_1^*, \cdots, y_n^*\}$ for the input sentence, the loss function for a model with parameters θ is:

$$\mathcal{L}_\theta = -\log p(\mathbf{y}^*|\mathbf{x}; \theta)$$

2.2 Embeddings

There are mainly four kinds of embeddings that have been proved effective on the sequence labeling task: contextual sub-word embeddings, contextual character embeddings, non-contextual word embeddings and non-contextual character embeddings[1]. As we conduct our experiments in multilingual settings, we need to select suitable embeddings from each category for the concatenation.

Contextual Sub-word Embeddings (CSEs) **CSEs** such as OpenAI GPT (Radford et al.) and BERT (Devlin et al., 2019) are based on transformer (Vaswani et al., 2017) and use WordPiece embeddings (Sennrich et al., 2016; Wu et al., 2016) as input. Much research has focused on improving BERT model's performance such as better masking strategy (Liu et al., 2019) and cross-lingual training (Conneau and Lample, 2019). Since we focus on the multilingual settings of sequence labeling tasks, we use multilingual BERT (M-BERT), as recent researches shows its strong generalizability over various languages and tasks (Pires et al., 2019; Karthikeyan et al., 2020).

Contextual Character Embeddings (CCEs) Liu et al. (2018) proposed a character language model by applying the BiLSTM over the sentence and trained jointly with the sequence labeling task. (Pooled) Contextual string embeddings (Flair) (Akbik et al., 2018, 2019) are pretrained on a large amount of unlabeled data and result in significant improvements for sequence labeling tasks. We use the Flair embeddings due to their high accuracy for sequence labeling task[2].

Non-contextual Word Embeddings (NWEs) The most common approach to the **NWEs** is Word2vec (Mikolov et al., 2013), which is a skip-gram model learning word representations by predicting neighboring words. Based on this approach, GloVe (Pennington et al., 2014) creates a co-occurrence matrix for global information and fastText (Bojanowski et al., 2017) represents each word as an n-gram of characters. We use fastText in our experiments as there are pretrained embeddings for 294 languages.

Non-contextual Character Embeddings (NCEs) Using character information to represent the embeddings of word is proposed by Santos and Zadrozny (2014) with a lot of following work using a CNN structure to encode character representation (dos Santos and Guimarães, 2015; Chiu and Nichols, 2016; Ma and Hovy, 2016). Lample et al. (2016) utilized BiLSTM on the character sequence of each word. We follow this approach as it usually results in better accuracy (Yang et al., 2018).

3 Experiments and Results

For simplicity, we use **M** to represent M-BERT embeddings, **F** to represent Flair embeddings, **W** to represent fastText embeddings, **C** to represent non-contextual character embeddings, **All** to represents the concatenation of all types of embeddings and the operator "+" to represent the concatenation operation. We use the MaxEnt approach for all experiments[3]. Due to the space limit, some detailed experiment settings, extra experiments and discussions are included in the appendix.

3.1 Settings

Datasets We use datasets from three multilingual sequence labeling tasks over 8 languages in our experiments: WikiAnn NER datasets (Pan et al., 2017), UD Part-Of-Speech (POS) tagging datasets (Nivre et al., 2016), and CoNLL 2003 chunking datasets (Tjong Kim Sang and De Meulder, 2003). We use language-specific fastText and Flair embeddings depending on the dataset.

Embedding Concatenation Since experimenting on all 15 concatenation combinations of the four embeddings is not essential for evaluating the effectiveness of each kind of embeddings, we experiment on the following 7 concatenations: **F**, **F+W**,

[1] We do not use contextual word embeddings such as ELMo (Peters et al., 2018) since Akbik et al. (2018) showed that concatenating Flair embeddings with ELMo embeddings cannot further improve the accuracy.

[2] We do not use the pooled version of Flair due to its slower speed in training.

[3] We find that the observations from the MaxEnt experiments do not change in all experiments with the CRF approach.

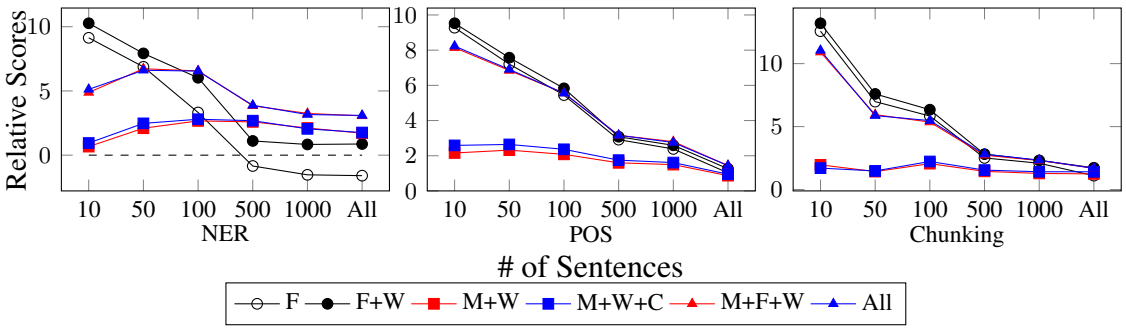

Figure 1: Relative score improvements against models with M-BERT embeddings for three tasks.

	Embeddings				Tasks			
	M	F	W	C	NER	POS	Chunk	Avg.
1.	✗	✓	✗	✗	82.1	96.3	92.3	90.2
2.	✗	✓	✓	✗	84.6	96.5	**92.9**	91.4
3.	✓	✗	✗	✗	83.8	95.3	91.3	90.1
4.	✓	✗	✓	✗	85.5	96.1	92.5	91.4
5.	✓	✗	✓	✓	85.5	96.2	92.6	91.5
6.	✓	✓	✓	✗	**86.8**	**96.7**	**92.9**	**92.1**
7.	✓	✓	✓	✓	**86.8**	**96.7**	**92.9**	**92.1**

Table 1: Averaged F1 scores over languages for each task with different embedding concatenations.

M, **M+W**, **M+W+C**, **M+F+W**, **All**. Through these concatenations, we can answer the following questions: (1) whether **NWEs** are still helpful (**F** vs. **F+W** and **M** vs. **M+W**); (2) whether **NCEs** are still helpful (**M+W** vs. **M+W+C** and **M+F+W** vs. **All**); (3) whether concatenating different contextual embeddings results in a better sequence labeler (**F+W** vs. **M+F+W** and **M+W** vs. **M+F+W**); (4) which one is the best concatenation.

3.2 Rich-resource and Low-resource Settings

How to build better sequence labelers through embedding concatenations in both rich-resource and low-resource settings is the most important concern for users. We report the results of various concatenations of embeddings for the tasks in Table 1 for rich-resource settings and in Figure 1 for low-resource settings. From the results, we have the following observations.

Observation #1. Concatenating more embedding variants results in better sequence labelers: In rich-resource settings, concatenating more embedding variants (**M+F+W** and **All**) results in best scores in most of the cases, which indicates that the inductive biases in various kind of embeddings are helpful to train a better sequence labeler. In low-resource settings, **M+F+W** and **All** performs inferior to the **F+W** when the number of sentences are lower than 100. However, when the training set gets larger, the gap between these concatenations becomes smaller and reverses when the training set becomes larger than 100 for NER and POS tagging and the gap also disappears for Chunking. A possible reason is that using **CSEs** makes the model sample inefficient so that **CSEs** requires more training samples to improve accuracy than **CCEs**. The observation suggests that concatenating more embedding variants performs better if the training set is not extremely small.

Observation #2. NCEs become less effective when concatenated with CSEs and CCEs: Concatenating **NCEs** with **CSEs** only marginally improves the accuracy. There is almost no improvement when concatenated with both **CSEs** and **CCEs** but the **NCEs** does not hurt the accuracy as well. A possible reason is that the **CSEs** and **CCEs** largely contain the information in **NCEs**[4].

Observation #3. NWEs are significantly helpful on top of contextual embeddings: Although models based on contextual embeddings have proved to be stronger than models based on **NWEs** for sequence labeling, concatenating **NWEs** with contextual embeddings can still improve the accuracy significantly. The results imply that the contextual embeddings contain more contextual information over the input but lack static word information.

From these observations, we find that in most of rich-resource and low-resource settings, concatenating all embeddings variants or all embeddings variants except **NCEs** is the simplest choice for a better sequence labeler.

3.3 Cross-domain Settings

Another concern for users is that we want to build better sequence labelers not only in in-domain set-

[4]The observation is consistent with the observation of Akbik et al. (2018), but we experimented on more languages and tasks with the M-BERT embeddings.

	F	F+W	M	M+W	M+W+C	M+F+W	All
AVG.	46.3	48.6	47.4	48.4	48.7	49.9	**50.4**

Table 2: Cross-domain transfer from the Wikipedia domain to the news domain on the NER task.

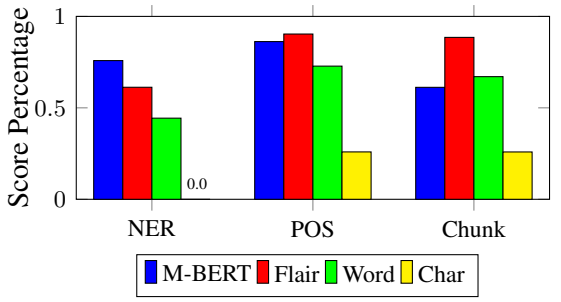

Figure 2: Importance of each embedding over the concatenation of **All** embeddings. The score percentage represents the average score preserving only one kind of embeddings divided by the score without masking.

tings but in out-of-domain settings as well. We conduct experiments in cross-domain settings to show how the embedding concatenations impact the accuracy when the distribution of training data and test data are different. We evaluate our Wikipedia NER models on CoNLL 2002/2003 NER (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003) datasets from the news domain. The results (Table 2) are almost consistent with rich-resource settings, suggesting that concatenating more embedding variants results in better sequence labelers.

3.4 Importance of Embeddings

To study the effectiveness of concatenating embeddings from another perspective, we preserve only one kind of embedding in **All** and mask out the other embeddings as 0 to study how the models rely upon each kind of embeddings. To avoid the impact of embedding dimensions, we train the model by linearly projecting each kind of embeddings into the same dimension of 4096. The results (Figure 2) show that the accuracy of preserved embeddings has a positive correlation with the results in Table 1. For example, **M** gets higher accuracy than other embeddings in NER and Table 1 also shows that the model with **F** performs inferior to the model with **M** only. The models with concatenated embeddings almost do not rely on **NCEs** and relies mostly on **CSEs** or **CCEs** depending on the task. These results show that models with concatenated embeddings can extract helpful information from each kind of embeddings to improve accuracy.

EMBEDDINGS						TASKS		
M	F	W	C	B	MF	NER	POS	CHUNK
+En-BERT (English)								
✓	✓	✓	✓	✗	✗	81.8	97.0	91.6
✗	✓	✓	✓	✓	✗	80.5	**97.2**	**91.8**
✓	✓	✓	✓	✓	✗	**82.1**	**97.2**	91.6
+M-Flair (All languages)								
✓	✓	✓	✓	✗	✗	**86.8**	**96.7**	**92.9**
✓	✗	✓	✓	✗	✓	86.1	96.5	92.8
✓	✓	✓	✓	✗	✓	**86.8**	**96.7**	**92.9**

Table 3: Comparisons of the effectiveness for additionally concatenating the same category of embeddings. **B** represents the En-BERT embeddings and **MF** represents the M-Flair embeddings.

EMBEDDINGS	TASKS		
	NER	POS	CHUNK
F+W	32.7	81.7	78.2
F+W+Proj.	**33.2**	**82.3**	**79.0**
All	27.5	80.4	76.1

Table 4: Comparisons of **F+W**, **All**, and **F+W+proj** (**F+W** with linearly projecting the hidden size into the hidden size of **All**) in three tasks with 10-sentence low-resource settings. The accuracy is averaged over tasks.

3.5 On Concatenating Similar Embeddings

Since concatenating more embeddings variants results in better sequence labelers, we additionally concatenate multilingual Flair embeddings (M-Flair) or English BERT embeddings (En-BERT) with **All** embeddings to show whether concatenating the same category of embeddings can further improve the accuracy. We evaluate the addition of En-BERT on English and M-Flair on all languages in each task. The results are shown in Table 3. It can be seen that additionally concatenating the same category of embeddings does not further improve the accuracy in most cases except for concatenating En-BERT on English WikiAnn NER. A possible reason is that the BERT models are trained on the same domain as WikiAnn and hence the inductive biases of BERT embeddings help improve the accuracy.

We also find that concatenating En-BERT with **All** only improves the accuracy of WikiAnn English NER. We think the possible reason for the improvement is that the BERT and the training data have the same domain of Wikipedia. We conduct the same concatenation on the CoNLL English NER dataset for comparison. The results in Table 7 show that concatenating En-BERT with **All** does not further improve the accuracy on CoNLL English NER.

	Embeddings					Tasks		
	M	F	W	C	B	NER	POS	Chunk
Low-Resource: 10 Sentences								
1.	✗	✓	✓	✗	✗	**35.5**±1.4	**80.2**±0.1	**73.3**±0.6
2.	✓	✓	✓	✓	✗	25.4±0.8	77.9±0.2	70.8±0.5
3.	✗	✓	✓	✓	✓	29.3±0.8	79.6±0.2	67.9±0.5
Low-Resource: 50 Sentences								
1.	✗	✓	✓	✗	✗	**48.6**±0.3	88.8±0.0	**82.2**±0.0
2.	✓	✓	✓	✓	✗	48.5±0.4	87.5±0.1	80.3±0.3
3.	✗	✓	✓	✓	✓	43.4±0.9	**88.9**±0.0	78.8±0.1
Low-Resource: 100 Sentences								
1.	✗	✓	✓	✗	✗	54.8±0.5	90.6±0.1	**83.7**±0.0
2.	✓	✓	✓	✓	✗	**56.8**±0.1	90.3±0.0	82.4±0.0
3.	✗	✓	✓	✓	✓	50.2±0.5	**91.4**±0.1	82.9±0.1
Low-Resource: 500 Sentences								
1.	✗	✓	✓	✗	✗	68.3±0.2	92.8±0.0	86.8±0.0
2.	✓	✓	✓	✓	✗	**69.1**±0.2	93.0±0.1	86.7±0.1
3.	✗	✓	✓	✓	✓	67.3±0.1	**93.9**±0.1	**86.9**±0.0
Low-Resource: 1000 Sentences								
1.	✗	✓	✓	✗	✗	72.0±0.1	94.0±0.1	87.1±0.1
2.	✓	✓	✓	✓	✗	**75.2**±0.3	94.4±0.1	87.1±0.2
3.	✗	✓	✓	✓	✓	70.8±0.1	**95.0**±0.0	**87.6**±0.1
Rich-Resource								
1.	✗	✓	✓	✗	✗	79.9±0.3	96.7±0.0	91.7±0.1
2.	✓	✓	✓	✓	✗	**81.7**±0.2	97.0±0.1	91.6±0.1
3.	✗	✓	✓	✓	✓	80.5±0.2	**97.2**±0.0	**91.8**±0.1

Table 5: Comparisons of using English BERT instead of M-BERT in English datasets. **B** represents the En-BERT embeddings. We also provide the concatenation of Flair and pretrained word embeddings for reference.

Embeddings	Tasks		
	NER	POS	Chunk
All	**86.8**	**96.7**	**92.9**
All+50d Proj.	83.8	96.3	92.0
All+1024d Proj.	84.8	96.5	92.2
All+4096d Proj.	85.1	96.5	92.2

Table 6: Comparisons of **All** with different linear projection size in three tasks with rich-resource settings. The accuracy is averaged over tasks.

3.6 English BERT vs. M-BERT

We use English BERT embeddings instead of M-BERT embeddings to see whether the language-specific **CSEs** impact the observations. The results (Table 5) show that our observations do not change in both rich-resource and low-resource settings. Using a language-specific BERT embedding can even get better sequence labelers for the POS tagging and chunking tasks in rich-resource settings.

3.7 Hidden Sizes and Accuracy

In low-resource settings with 10 sentences, we find that models with **All** perform inferior to the models with **F+W**. One possible concern is that whether the larger hidden size of **All** introduces more parameters in the model and makes the model over-fits the training set. We linearly project the hidden size of **F+W** (4396) to the same hidden size as **All** (5214). Table 4 shows that with linear projection, **F+W** performs even better. Therefore, the cause for over-fitting is not the inferior accuracy of **All** but possibly the sample inefficiency for **CSEs**.

Embeddings					Task
M	F	W	C	B	English NER
✓	✓	✓	✓	✗	**92.1**±0.1
✗	✓	✓	✓	✓	92.0±0.1
✓	✓	✓	✓	✓	**92.1**±0.1

Table 7: Comparisons of concatenating En-BERT with **All** on CoNLL NER. **B** represents the En-BERT.

Another concern is whether we can project each embedding to a larger hidden size to improve the accuracy. Since we try a projection to 4096 for each embedding in **F+W+proj** (Section 3.4), we further project each embedding variants to see how the projection affect the accuracy in rich-resource settings. The results (Table 6) show that the linear projection for each embedding significantly decreases the accuracy of the models.

From the two experiments, we find that the hidden sizes of concatenated embeddings do not impact the observations.

4 Conclusion

In this paper, we analyze how to get a better sequence labeler by concatenating various kinds of embeddings. We make several empirical observations that we hope can guide future work to build better sequence labelers: (1) in most settings, concatenating more embedding variants leads to better results, while in extremely low-resource settings, only using **CSEs** and **NWEs** performs better; (2) **NCEs** become less effective when concatenated with contextual embeddings, while **NWEs** are still beneficial; (3) neural models can automatically learn which embeddings are beneficial to the task; (4) additionally concatenating similar contextual embeddings with the best concatenations from (1) cannot further improve the accuracy in most cases.

Acknowledgements

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

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.

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.

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.

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

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7057–7067.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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

K Karthikeyan, Zihan Wang, Stephen Mayhew, and Dan Roth. 2020. Cross-lingual ability of multilingual bert: An empirical study. In *International Conference on Learning Representations*.

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 âĂŹ01, page 282âĂŞ289, 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.

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*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2019. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

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.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

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

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Cícero dos Santos and Victor Guimarães. 2015. Boosting named entity recognition with neural character embeddings. In *Proceedings of the Fifth Named Entity Workshop*, pages 25–33, Beijing, China. Association for Computational Linguistics.

Cicero D Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st international conference on machine learning (ICML-14)*, pages 1818–1826.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. 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 Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

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 Appendix

In this appendix, we use ISO 639-1 codes[5] to represent each language for simplification.

A.1 Settings

Datasets We use the following datasets for experiments:

- **Named Entity Recognition (NER)**: We use **WikiAnn**[6] (Pan et al., 2017) datasets and CoNLL 2002/2003 NER[7] (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003) datasets for experiments. The **WikiAnn** datasets contain silver standard NER tags over 282 languages. We select 8 languages from the dataset. We randomly choose 5000 sentences from the dataset for each language except English with 12000 sentences. We split the dataset by 3:1:1 for training/development/test. We use the standard training/development/test split for the CoNLL NER experiments.

- **Part-Of-Speech (POS) tagging**: We use universal POS tagging annotations in the **Universal Dependencies** (UD) (Nivre et al., 2016) datasets[8]. We choose one treebank for each language from the same 8 languages that are used in the WikiAnn experiments. The list of treebank are shown in Table 8. We use the

[5] https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes
[6] https://elisa-ie.github.io/wikiann/
[7] https://www.clips.uantwerpen.be/conll2003/ner/
[8] https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2837

official train/development/test split for experiments.

- **Chunking**: We use the chunking datasets from the CoNLL 2003 shared task (Tjong Kim Sang and De Meulder, 2003), which contain two languages for chunking. We use the official train/development/test split for experiments.

Model Configuration and Running For the embeddings, the hidden sizes for fastText and Flair embeddings[9] are 300 and 4096, respectively. The dimension of character embeddings is set to 50[10] following previous work (Lample et al., 2016). For M-BERT embeddings, we use the cased version that is trained on 104 languages for all datasets. We use the official release of *bert-base-cased* model in the experiments for English BERT. The word embeddings are fine-tuned and character embeddings are trained for tasks while the Flair and BERT embeddings are fixed. Our codes are mainly based on the official release of Flair[11] which is based on PyTorch v1.1.0 (Paszke et al., 2019). We run our experiments on a GPU server with NVIDIA Tesla V100 GPU. For model training, we set the mini-batch size to 2,000 tokens for better GPU utilization. Following the official release of Flair, we use an SGD optimizer with a learning rate of 0.1 for training all models and set the hidden size of BiLSTM to 256. We anneal the learning rate by 0.5 if there is no improvement on the development sets for 10 and 100 epochs when training rich-resource and low-resource datasets respectively. We fix these hyper-parameters for all experiments because we find that tuning these hyper-parameters does not impact the observation and usually results in lower accuracy. We average over 5 runs for each experiment and report the macro-average score over all languages for each task.

Pre-processing and Evaluation We evaluate the NER and chunking by the F1 score and POS tagging by the accuracy. We use the evaluation script in the official release of Flair. We convert the BIO format into BIOES format for all NER and chunking datasets.

[9]Details of Flair embeddings https://github.com/flairNLP/flair/blob/master/resources/docs/embeddings/FLAIR_EMBEDDINGS.md

[10]We did not observe further gains when increasing the dimension size.

[11]https://github.com/flairNLP/flair

Language	Treebank
ar	PADT
cs	FicTree
de	GSD
en	EWT
es	GSD
fr	Sequoia
nl	LassySmall
ta	TTB

Table 8: The list of treebank that we used in UD POS tagging.

A.2 Detailed Results

For the models using the CRF layer, similar to the main paper, we plot our results in the rich-resource and low-resource settings in Figure 3. The figures have similar trends as the MaxEnt models, showing that output structures do not impact the observations.

Table 10 shows the importance of each kind of embeddings for each language and task (Section 3.4 in the main paper). Table 11, 13 and 15 show average scores over each language for each task in the rich-resource and low-resource settings (Section 3.2). Table 12, 14 and 16 show average scores over each language for each task in the rich-resource and low-resource settings. Table 9 shows the average scores for each language in our cross-domain experiments (Section 3.3). Table 17 show the detailed comparison for additionally concatenating M-Flair embeddings with **All** for all datasets (Section 3.5).

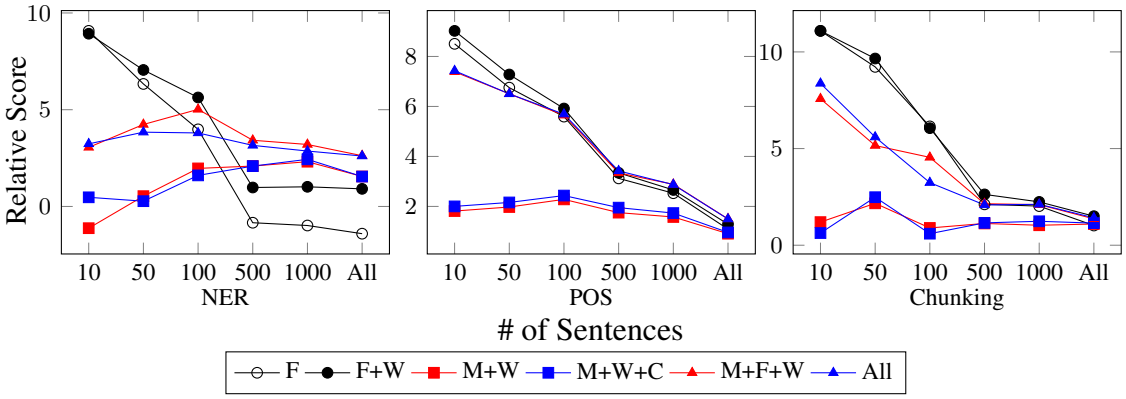

Figure 3: Relative score improvements against models with M-BERT embeddings for three tasks. Models are equipped with the CRF layer.

	EMBEDDINGS				MaxEnt models on WikiAnn NER				
	M	**F**	**W**	**C**	de	en	es	nl	avg
1.	✗	✓	✗	✗	41.0±0.7	46.9±1.4	47.4±1.7	49.8±0.6	46.3
2.	✗	✓	✓	✗	43.2±0.6	48.1±1.0	**50.1**±0.3	52.9±0.3	48.6
3.	✓	✗	✗	✗	45.2±0.9	49.2±0.7	45.2±2.8	49.9±0.7	47.4
4.	✓	✗	✓	✗	43.5±0.1	50.9±0.7	46.9±1.3	52.5±0.6	48.4
5.	✓	✗	✓	✓	44.4±1.2	50.6±0.6	47.1±0.1	52.9±0.6	48.7
6.	✓	✓	✓	✗	**46.7**±0.4	51.1±0.7	48.0±2.1	53.9±0.9	49.9
7.	✓	✓	✓	✓	46.6±0.4	**52.3**±0.5	47.9±0.3	**54.7**±0.3	**50.4**

Table 9: Detailed results of cross-domain transfer from the Wikipedia domain to the news domain on the NER task. We use the ISO 639 language code to represent each language.

	M-BERT	Flair	Word	Char	**All**
WikiAnn NER					
ar	53.0±1.4	43.7±1.1	44.9±2.6	0.0±0.0	82.6±0.1
cs	71.6±0.4	54.5±2.4	45.9±2.1	0.0±0.0	88.0±0.1
de	67.0±0.8	58.1±2.5	33.6±1.9	0.0±0.0	85.6±0.2
en	67.7±0.9	46.5±0.9	22.8±0.5	0.0±0.0	88.4±0.7
es	74.9±1.0	54.6±2.5	35.0±2.0	0.0±0.0	79.9±0.6
fr	75.9±1.5	48.0±1.3	35.2±3.9	0.0±0.0	84.2±0.1
nl	63.4±3.7	59.0±1.6	40.3±0.9	0.0±0.0	86.7±0.4
ta	41.3±0.9	51.5±1.5	43.5±1.4	0.0±0.0	83.4±0.2
Avg.	64.3	52.0	37.6	0.0	84.8
UD POS tagging					
ar	84.5±1.7	88.3±0.7	79.3±0.6	27.4±0.9	92.0±0.6
cs	84.0±0.6	90.8±0.1	68.5±0.6	25.6±3.2	96.5±0.1
de	78.3±2.9	88.3±0.2	71.3±0.8	26.3±3.6	98.8±0.0
en	85.1±0.9	85.8±0.2	65.3±1.8	32.5±1.1	97.2±0.0
es	83.2±1.6	92.4±0.5	80.9±1.6	20.4±5.9	96.6±0.0
fr	92.6±0.7	85.2±0.6	63.7±0.2	15.8±3.0	95.2±0.0
nl	80.9±0.3	89.9±0.1	70.9±1.9	18.9±1.7	98.7±0.0
ta	76.8±1.8	76.6±0.3	62.0±0.0	33.1±4.8	96.7±0.0
Avg.	83.2	87.2	70.2	25.0	96.5
Chunking					
de	66.3±4.0	90.2±0.3	52.9±2.9	28.0±0.5	93.5±0.1
en	46.6±2.1	73.0±0.8	70.7±0.6	19.8±0.3	90.8±0.1
Avg.	56.4	81.6	61.8	23.9	92.2

Table 10: Detailed results on importance of embeddings for each language.

	Embeddings				MaxEnt models on WikiAnn NER								
	M	F	W	C	ar	cs	de	en	es	fr	nl	ta	Avg.
Low-Resource: 10 Sentences													
1.	✗	✓	✗	✗	33.6±1.3	36.9±1.5	25.5±0.8	**35.8**±1.2	**34.4**±2.0	39.4±2.0	26.2±1.7	20.6±0.7	31.5
2.	✗	✓	✓	✗	**34.4**±1.4	**40.5**±1.4	**26.0**±0.4	35.5±1.4	33.9±3.1	**41.6**±0.2	28.6±1.9	**20.8**±1.1	**32.7**
3.	✓	✗	✗	✗	29.6±2.4	31.0±0.9	13.8±1.9	23.2±1.7	23.2±1.5	24.6±2.3	16.2±1.5	17.5±0.8	22.4
4.	✓	✗	✓	✗	28.8±1.9	32.2±0.9	15.7±1.6	22.1±0.6	23.1±1.0	24.2±2.5	20.0±2.9	18.6±0.7	23.1
5.	✓	✗	✓	✓	27.8±0.6	32.1±1.5	15.8±2.2	23.1±0.5	23.9±1.3	26.3±2.3	19.2±1.8	18.6±0.4	23.4
6.	✓	✓	✓	✗	29.6±1.9	34.8±0.5	22.3±0.9	26.4±0.9	25.2±1.0	29.8±1.5	29.5±2.1	20.7±0.6	27.3
7.	✓	✓	✓	✓	30.3±3.0	34.5±0.6	23.0±1.4	25.4±0.8	25.3±1.3	30.8±1.3	**30.6**±3.2	20.5±0.5	27.5
Low-Resource: 50 Sentences													
1.	✗	✓	✗	✗	47.6±1.0	54.6±1.9	52.7±1.4	47.3±0.3	**54.8**±0.8	**54.5**±0.5	49.2±0.6	**45.3**±0.6	50.8
2.	✗	✓	✓	✗	**49.8**±0.8	**58.0**±0.4	**55.7**±1.0	**48.6**±0.3	54.2±1.6	54.0±1.1	49.7±0.9	44.5±0.6	**51.8**
3.	✓	✗	✗	✗	40.7±1.3	52.8±2.1	42.1±1.6	45.4±0.6	42.7±2.6	49.0±1.3	46.4±2.0	31.9±1.4	43.9
4.	✓	✗	✓	✗	44.3±1.2	54.6±0.4	46.6±2.3	46.9±0.5	45.8±1.2	49.2±1.2	46.7±1.3	33.6±1.1	46.0
5.	✓	✗	✓	✓	44.1±0.6	55.7±0.9	47.0±4.1	47.1±0.6	44.9±2.0	49.1±1.2	47.6±1.5	35.4±1.1	46.4
6.	✓	✓	✓	✗	49.4±1.7	**58.0**±1.3	50.1±1.1	48.5±0.1	50.4±1.0	53.4±1.3	**52.5**±0.9	42.5±2.6	50.6
7.	✓	✓	✓	✓	48.3±1.4	**58.0**±1.2	49.6±1.3	48.5±0.4	51.0±1.5	52.4±0.9	51.9±0.5	44.3±2.3	50.5
Low-Resource: 100 Sentences													
1.	✗	✓	✗	✗	56.5±0.7	57.8±0.8	55.0±1.9	52.3±0.3	66.1±0.9	56.8±2.8	55.4±1.0	50.6±0.8	56.3
2.	✗	✓	✓	✗	**58.7**±1.2	61.9±1.0	**56.9**±1.1	54.8±0.5	**67.0**±1.0	60.5±0.7	57.9±1.3	**54.6**±1.3	59.0
3.	✓	✗	✗	✗	47.6±3.0	57.7±3.8	49.7±3.7	54.9±0.5	59.2±1.2	57.5±0.9	54.9±1.0	42.3±3.9	53.0
4.	✓	✗	✓	✗	51.0±3.2	59.8±0.8	52.3±1.3	56.0±0.3	60.0±1.6	58.4±0.4	59.0±3.1	48.8±4.0	55.7
5.	✓	✗	✓	✓	51.2±2.9	61.1±0.9	52.8±1.7	55.9±0.7	61.0±0.9	60.4±2.2	57.3±2.2	46.6±1.6	55.8
6.	✓	✓	✓	✗	57.4±1.5	**64.3**±1.7	55.3±1.0	**57.0**±0.4	65.7±2.3	62.2±0.7	61.3±0.7	53.2±1.3	**59.6**
7.	✓	✓	✓	✓	58.2±1.1	62.8±1.0	54.4±1.5	56.8±0.1	66.0±0.3	**62.7**±0.7	**61.8**±1.3	53.7±0.6	59.5
Low-Resource: 500 Sentences													
1.	✗	✓	✗	✗	69.2±0.8	77.2±1.1	72.1±0.7	65.6±0.3	77.6±0.6	73.6±0.4	74.6±1.5	61.9±1.1	71.5
2.	✗	✓	✓	✗	**73.2**±0.7	78.6±1.3	72.9±1.0	68.3±0.2	78.2±1.3	75.1±1.1	75.0±1.3	66.1±1.2	73.4
3.	✓	✗	✗	✗	67.3±1.0	77.0±0.5	71.8±1.0	67.7±0.2	77.6±0.7	76.7±0.9	75.5±1.7	64.9±1.3	72.3
4.	✓	✗	✓	✗	72.0±0.3	78.6±0.8	75.2±1.7	68.3±0.0	78.6±1.5	77.7±1.1	78.0±0.9	70.8±1.0	74.9
5.	✓	✗	✓	✓	72.1±0.9	78.5±0.8	74.7±0.6	68.1±0.2	78.5±1.3	79.4±0.1	**78.2**±1.4	70.4±1.3	75.0
6.	✓	✓	✓	✗	72.9±0.7	79.0±0.4	76.2±0.4	**69.7**±0.2	**82.0**±0.4	**80.5**±0.4	78.0±0.6	**71.0**±0.8	**76.2**
7.	✓	✓	✓	✓	72.9±0.7	**79.9**±1.0	**76.3**±0.5	69.1±0.2	**82.0**±0.5	**80.5**±0.6	78.0±1.8	70.8±0.5	**76.2**
Low-Resource: 1000 Sentences													
1.	✗	✓	✗	✗	74.7±0.6	80.2±0.4	75.6±0.6	68.9±0.0	82.9±0.4	75.9±0.8	80.1±0.7	73.8±0.4	76.5
2.	✗	✓	✓	✗	77.8±0.2	82.6±0.5	77.2±0.4	72.0±0.1	84.0±0.3	78.4±1.0	83.3±0.5	75.7±0.7	78.9
3.	✓	✗	✗	✗	73.6±0.4	81.7±0.4	77.3±0.2	72.2±0.1	84.8±0.6	82.2±0.9	81.4±0.4	71.2±0.8	78.0
4.	✓	✗	✓	✗	77.0±0.2	83.3±0.4	78.7±0.6	73.4±0.5	85.2±0.8	83.1±0.4	84.2±0.6	76.3±0.7	80.2
5.	✓	✗	✓	✓	77.9±0.6	83.3±0.7	78.9±0.3	73.6±0.0	84.6±0.8	82.3±0.9	84.2±0.7	76.0±0.1	80.1
6.	✓	✓	✓	✗	78.3±0.3	**84.0**±0.4	**79.8**±0.5	**75.3**±0.3	86.3±0.4	83.3±0.6	**85.0**±1.0	**78.2**±0.5	**81.3**
7.	✓	✓	✓	✓	**78.4**±0.6	83.8±0.3	**79.8**±0.3	75.2±0.3	**86.4**±0.2	**83.6**±0.3	84.5±0.6	77.9±0.5	81.2
Rich-Resource													
1.	✗	✓	✗	✗	81.0±0.5	83.0±2.5	80.4±0.5	76.9±0.1	86.2±0.4	81.8±0.6	85.7±0.5	82.3±0.5	82.2
2.	✗	✓	✓	✗	84.5±0.3	86.9±0.6	81.8±0.4	79.9±0.3	88.4±0.5	83.8±0.3	88.2±0.5	83.5±0.6	84.6
3.	✓	✗	✗	✗	79.4±0.4	86.2±0.4	83.1±0.3	79.6±0.4	87.5±0.8	86.9±0.4	87.8±0.6	79.5±0.4	83.7
4.	✓	✗	✓	✗	83.2±0.4	87.1±0.6	84.2±0.3	80.3±0.6	88.4±0.5	87.4±0.3	89.3±0.4	83.8±0.4	85.5
5.	✓	✗	✓	✓	83.2±0.8	87.4±0.3	83.9±0.2	81.0±0.1	88.2±0.4	87.5±0.3	89.2±0.4	83.7±0.6	85.5
6.	✓	✓	✓	✗	84.2±0.5	**88.8**±0.5	85.4±0.5	**81.8**±0.1	90.4±0.4	88.1±0.5	**90.4**±0.3	85.4±0.3	**86.8**
7.	✓	✓	✓	✓	**84.6**±0.2	88.4±0.2	**85.5**±0.3	81.8±0.2	**90.6**±0.2	**88.4**±0.5	90.1±0.3	**85.5**±0.3	**86.8**

Table 11: Averaged F1 scores over 8 languages for WikiAnn NER.

	Embeddings				CRF models on WikiAnn NER								
	M	F	W	C	ar	cs	de	en	es	fr	nl	ta	Avg.
Low-Resource: 10 Sentences													
1.	✗	✓	✗	✗	31.0±5.2	38.6±1.1	23.5±0.7	**35.1**±2.2	**37.6**±3.6	41.2±2.2	27.4±3.4	19.6±1.1	**31.8**
2.	✗	✓	✓	✗	**32.0**±3.3	**38.9**±4.6	**24.1**±1.3	33.3±0.5	31.7±2.0	**43.1**±4.5	**30.4**±3.9	19.6±3.7	31.6
3.	✓	✗	✗	✗	31.5±4.7	30.0±2.8	12.5±4.9	23.1±1.0	23.5±5.2	24.7±3.2	18.3±2.7	17.8±1.0	22.7
4.	✓	✗	✓	✗	28.4±3.2	30.9±1.9	14.2±1.3	21.3±2.9	21.5±1.9	25.4±3.4	12.5±8.0	18.5±1.0	21.6
5.	✓	✗	✓	✓	30.9±1.8	32.8±2.2	11.0±5.7	19.7±1.4	24.7±1.4	28.1±0.3	19.3±3.8	18.8±1.3	23.2
6.	✓	✓	✓	✗	29.3±1.4	33.0±2.5	20.9±1.5	22.4±0.8	23.8±2.0	28.5±1.9	26.5±4.6	**21.7**±1.7	25.7
7.	✓	✓	✓	✓	30.0±2.4	32.2±1.6	22.2±2.2	22.1±0.0	23.1±1.9	30.0±3.6	28.5±2.1	19.2±2.0	25.9
Low-Resource: 50 Sentences													
1.	✗	✓	✗	✗	49.6±2.0	55.0±1.3	**54.3**±0.6	43.8±1.7	55.3±2.7	**57.6**±1.0	53.9±1.3	45.4±2.7	51.9
2.	✗	✓	✓	✗	**51.3**±4.6	**58.6**±1.8	52.8±1.0	45.1±0.3	**55.9**±1.7	55.6±2.1	**55.2**±1.3	**46.1**±2.2	**52.6**
3.	✓	✗	✗	✗	45.5±2.7	55.3±3.3	42.2±3.0	44.4±0.4	44.7±2.7	52.3±1.7	47.0±3.7	32.8±5.2	45.5
4.	✓	✗	✓	✗	45.9±1.2	54.6±1.3	45.3±2.6	40.6±2.6	45.5±2.1	51.4±2.2	47.7±1.2	37.5±3.4	46.1
5.	✓	✗	✓	✓	44.1±4.4	54.8±1.7	47.0±3.5	40.9±1.3	47.1±2.4	50.8±2.7	46.8±3.0	34.9±7.3	45.8
6.	✓	✓	✓	✗	48.2±4.7	57.9±2.3	51.6±1.8	41.7±0.5	51.0±2.7	53.7±3.1	51.3±2.0	42.6±2.8	49.8
7.	✓	✓	✓	✓	45.0±6.2	57.1±1.2	50.0±3.3	**45.8**±0.6	51.5±1.4	53.0±2.6	51.1±4.2	41.3±3.2	49.4
Low-Resource: 100 Sentences													
1.	✗	✓	✗	✗	55.6±3.0	62.8±0.9	**59.3**±1.3	52.5±1.7	66.2±1.6	62.7±2.1	58.9±2.5	52.4±1.2	58.8
2.	✗	✓	✓	✗	**60.7**±1.9	63.1±0.6	58.2±1.9	50.5±1.1	**66.7**±1.7	**66.1**±0.8	61.6±0.8	**56.6**±1.3	**60.4**
3.	✓	✗	✗	✗	55.1±3.1	59.5±2.1	47.9±3.3	52.6±3.4	61.2±1.4	59.9±3.0	56.4±4.4	45.9±0.9	54.8
4.	✓	✗	✓	✗	53.9±1.5	61.7±2.4	51.1±2.5	51.4±0.9	62.9±1.8	62.5±0.5	60.9±1.2	49.8±2.3	56.8
5.	✓	✗	✓	✓	53.5±2.8	62.8±2.3	52.2±2.1	**52.8**±1.3	61.9±2.6	61.6±1.8	57.9±4.2	48.7±4.0	56.4
6.	✓	✓	✓	✗	58.4±2.2	**65.0**±1.0	55.5±3.0	52.6±2.5	66.3±1.6	62.6±1.4	**64.7**±1.2	53.4±1.3	59.8
7.	✓	✓	✓	✓	59.1±2.1	63.0±2.4	54.5±2.4	52.0±2.9	63.9±1.5	64.1±1.7	61.2±1.3	51.0±2.1	58.6
Low-Resource: 500 Sentences													
1.	✗	✓	✗	✗	69.3±0.7	78.0±0.8	73.0±1.9	65.3±0.7	80.4±0.7	76.0±0.5	76.2±0.8	66.8±0.8	73.1
2.	✗	✓	✓	✗	73.0±2.0	79.5±0.8	74.2±0.6	67.4±0.6	81.1±0.9	77.8±0.8	77.4±0.4	69.2±1.1	75.0
3.	✓	✗	✗	✗	70.2±1.0	77.7±0.9	73.1±1.2	67.5±0.9	80.8±1.1	79.0±1.2	76.1±0.7	67.3±0.7	74.0
4.	✓	✗	✓	✗	74.1±1.0	79.1±0.3	76.3±0.6	67.5±0.3	80.7±0.6	80.0±0.3	79.3±1.5	71.6±0.6	76.1
5.	✓	✗	✓	✓	73.6±0.5	78.5±1.2	75.5±1.0	68.1±0.5	81.5±1.2	80.3±0.8	78.9±1.5	72.1±0.4	76.1
6.	✓	✓	✓	✗	**74.5**±0.4	**80.3**±0.6	**76.7**±0.6	**70.3**±1.2	82.6±0.4	81.5±0.9	**80.5**±1.0	**72.9**±0.9	**77.4**
7.	✓	✓	✓	✓	74.2±0.7	80.2±1.0	75.8±1.0	69.4±0.1	**83.9**±1.0	**81.7**±0.8	79.4±0.7	72.4±1.3	77.1
Low-Resource: 1000 Sentences													
1.	✗	✓	✗	✗	76.9±0.6	80.7±0.7	77.0±0.5	69.4±0.1	84.2±0.7	78.0±0.5	81.4±0.4	75.5±0.6	77.9
2.	✗	✓	✓	✗	79.4±0.3	83.6±0.7	77.8±0.5	70.9±0.4	85.7±0.4	81.0±0.4	83.4±0.3	77.3±0.8	79.9
3.	✓	✗	✗	✗	75.3±0.7	82.7±0.5	76.9±0.5	72.5±1.4	85.9±0.4	82.4±0.6	82.0±0.4	73.4±0.4	78.9
4.	✓	✗	✓	✗	79.2±0.5	84.4±0.8	79.7±0.8	73.3±0.5	86.7±0.4	83.8±0.5	84.0±0.6	78.5±0.6	81.2
5.	✓	✗	✓	✓	78.7±0.8	84.7±1.1	79.4±0.4	73.7±0.1	86.8±0.2	84.1±0.1	84.8±0.4	78.3±0.4	81.3
6.	✓	✓	✓	✗	**79.6**±0.7	**85.0**±0.7	**80.8**±0.4	**74.4**±0.7	**87.4**±0.7	**84.5**±0.6	**85.6**±0.8	**79.3**±0.7	**82.1**
7.	✓	✓	✓	✓	79.4±0.6	84.8±0.4	80.2±0.6	74.3±0.2	87.1±0.8	84.3±0.5	84.6±0.5	79.2±0.5	81.7
Rich-Resource													
1.	✗	✓	✗	✗	82.8±0.3	85.8±0.5	81.0±0.6	78.4±0.1	87.1±0.5	82.9±0.7	86.2±0.4	82.8±0.5	83.4
2.	✗	✓	✓	✗	**85.2**±0.5	87.9±0.2	83.0±0.1	81.1±0.2	89.0±0.4	85.8±0.4	88.8±0.4	84.6±0.3	85.7
3.	✓	✗	✗	✗	80.3±0.4	87.1±0.4	84.2±0.4	81.2±0.1	88.8±0.2	87.8±0.4	87.6±0.6	81.3±0.5	84.8
4.	✓	✗	✓	✗	84.2±0.3	88.0±0.3	84.7±0.3	82.3±0.3	89.1±0.4	87.9±0.4	89.7±0.6	84.9±0.2	86.3
5.	✓	✗	✓	✓	84.0±0.4	87.9±0.4	85.0±0.4	82.2±0.2	89.3±0.5	87.6±0.4	89.6±0.5	85.0±0.3	86.3
6.	✓	✓	✓	✗	85.1±0.4	**89.6**±0.0	85.5±0.6	**82.9**±0.1	90.6±0.3	88.6±0.4	**90.8**±0.1	**86.1**±0.4	**87.4**
7.	✓	✓	✓	✓	85.0±0.4	89.3±0.2	**85.8**±0.1	82.8±0.2	**91.0**±0.3	**88.7**±0.3	90.4±0.3	86.0±0.3	**87.4**

Table 12: Averaged F1 scores over 8 languages for WikiAnn NER with the CRF layer.

	EMBEDDINGS				MaxEnt models on UD POS tagging								
	M	F	W	C	ar	cs	de	en	es	fr	nl	ta	Avg.
LOW-RESOURCE: 10 SENTENCES													
1.	✗	✓	✗	✗	86.4±0.3	**83.0**±0.4	**83.4**±0.6	79.5±0.1	88.7±0.1	85.5±0.2	**72.1**±0.5	72.5±0.4	81.4
2.	✗	✓	✓	✗	**87.1**±0.1	82.5±0.2	83.1±0.3	**80.2**±0.1	**88.8**±0.1	**86.2**±0.2	**72.1**±0.6	73.2±0.2	**81.7**
3.	✓	✗	✗	✗	80.7±0.7	71.0±1.3	73.4±1.1	72.5±0.1	78.5±1.2	76.4±0.7	62.8±1.3	61.7±1.6	72.1
4.	✓	✗	✓	✗	82.4±0.9	74.4±0.8	75.1±0.9	73.7±0.1	80.2±0.9	78.5±0.5	65.0±0.9	65.0±1.4	74.3
5.	✓	✗	✓	✓	82.6±0.6	74.7±0.3	75.9±0.4	73.9±0.4	81.4±0.8	78.3±0.9	64.5±0.9	66.4±1.4	74.7
6.	✓	✓	✓	✗	86.6±0.2	80.8±0.2	81.8±0.2	77.9±0.0	86.9±0.4	82.6±0.7	**72.1**±0.8	**73.5**±0.7	80.3
7.	✓	✓	✓	✓	86.8±0.2	81.1±0.2	81.9±0.2	77.9±0.2	86.9±0.3	82.6±0.5	**72.1**±0.9	**73.5**±0.4	80.4
LOW-RESOURCE: 50 SENTENCES													
1.	✗	✓	✗	✗	91.9±0.1	91.1±0.2	**91.4**±0.1	88.6±0.1	93.3±0.0	92.2±0.1	83.3±0.1	85.9±0.3	89.7
2.	✗	✓	✓	✗	**92.3**±0.1	**91.6**±0.1	91.3±0.2	**88.8**±0.0	**93.6**±0.1	**92.3**±0.1	**84.0**±0.2	86.7±0.3	**90.1**
3.	✓	✗	✗	✗	87.7±0.3	83.9±1.3	83.5±0.5	82.2±0.2	87.9±0.3	86.0±0.5	71.9±0.8	76.9±0.4	82.5
4.	✓	✗	✓	✗	89.3±0.3	86.3±0.4	85.5±0.8	83.9±0.1	89.5±0.7	88.1±0.5	75.1±1.0	81.0±0.7	84.8
5.	✓	✗	✓	✓	89.6±0.1	86.5±1.1	86.0±0.7	84.1±0.2	90.0±0.5	88.4±0.3	75.1±0.7	81.4±0.5	85.1
6.	✓	✓	✓	✗	91.6±0.1	91.1±0.3	90.8±0.2	87.5±0.2	92.5±0.3	91.7±0.1	82.7±0.1	**87.0**±0.2	89.3
7.	✓	✓	✓	✓	91.6±0.1	91.1±0.3	91.0±0.2	87.5±0.1	92.5±0.2	91.8±0.2	82.8±0.5	86.9±0.2	89.4
LOW-RESOURCE: 100 SENTENCES													
1.	✗	✓	✗	✗	93.5±0.1	93.6±0.1	**92.2**±0.1	90.2±0.1	94.2±0.0	94.4±0.1	88.2±0.2	88.5±0.7	91.8
2.	✗	✓	✓	✗	**93.7**±0.1	**93.9**±0.1	**92.2**±0.0	**90.6**±0.1	**94.6**±0.1	**94.5**±0.2	**89.1**±0.1	89.3±0.1	**92.2**
3.	✓	✗	✗	✗	90.4±0.0	88.8±0.2	85.9±0.5	85.7±0.1	90.4±0.2	90.2±0.4	77.9±0.8	81.9±0.4	86.4
4.	✓	✗	✓	✗	91.8±0.1	90.6±0.2	87.5±0.3	87.4±0.1	92.1±0.3	91.7±0.3	81.0±1.7	85.7±0.4	88.5
5.	✓	✗	✓	✓	91.9±0.1	90.9±0.1	87.8±0.2	87.7±0.1	92.1±0.3	91.8±0.2	81.9±1.7	85.9±0.5	88.8
6.	✓	✓	✓	✗	93.6±0.1	93.6±0.1	92.0±0.3	90.4±0.1	94.3±0.2	94.3±0.3	87.8±0.2	89.7±0.2	92.0
7.	✓	✓	✓	✓	93.6±0.1	93.6±0.1	91.9±0.1	90.3±0.0	94.4±0.1	94.3±0.1	87.8±0.5	**89.8**±0.3	91.9
LOW-RESOURCE: 500 SENTENCES													
1.	✗	✓	✗	✗	95.2±0.0	96.0±0.1	94.3±0.1	92.7±0.1	95.9±0.1	97.0±0.1	93.0±0.2	92.2±0.3	94.5
2.	✗	✓	✓	✗	95.3±0.0	**96.3**±0.1	**94.4**±0.0	92.8±0.0	96.0±0.0	97.4±0.1	93.2±0.2	92.3±0.6	94.7
3.	✓	✗	✗	✗	93.5±0.1	92.7±0.1	90.2±0.1	90.0±0.1	93.9±0.1	95.6±0.1	90.3±0.2	86.9±0.1	91.6
4.	✓	✗	✓	✗	94.5±0.1	94.7±0.1	91.5±0.1	91.8±0.1	95.1±0.1	96.7±0.1	91.6±0.1	89.9±0.2	93.2
5.	✓	✗	✓	✓	94.7±0.1	94.9±0.1	91.6±0.1	92.0±0.0	95.2±0.1	96.9±0.1	91.8±0.1	89.7±0.6	93.4
6.	✓	✓	✓	✗	**95.5**±0.1	96.1±0.1	94.0±0.1	**93.0**±0.0	**96.1**±0.1	**97.5**±0.0	**93.4**±0.1	**92.6**±0.4	**94.8**
7.	✓	✓	✓	✓	**95.5**±0.0	96.2±0.1	94.0±0.1	**93.0**±0.1	96.0±0.2	**97.5**±0.0	**93.4**±0.1	**92.6**±0.3	**94.8**
LOW-RESOURCE: 1000 SENTENCES													
1.	✗	✓	✗	✗	95.7±0.0	96.8±0.1	**94.7**±0.1	93.6±0.1	96.2±0.0	97.5±0.1	94.5±0.2	92.3±0.2	95.2
2.	✗	✓	✓	✗	95.8±0.0	**97.0**±0.1	94.6±0.1	94.0±0.1	96.3±0.1	98.0±0.0	94.8±0.0	92.4±0.2	95.4
3.	✓	✗	✗	✗	94.3±0.0	93.9±0.1	91.2±0.1	91.7±0.1	94.7±0.0	96.8±0.1	92.5±0.1	87.1±0.2	92.8
4.	✓	✗	✓	✗	95.1±0.1	95.8±0.0	92.5±0.1	93.4±0.0	96.0±0.1	97.6±0.0	94.0±0.1	89.7±0.3	94.3
5.	✓	✗	✓	✓	95.2±0.0	96.1±0.1	92.6±0.1	93.5±0.1	96.0±0.1	97.6±0.1	94.2±0.1	89.8±0.5	94.4
6.	✓	✓	✓	✗	**95.9**±0.0	**97.0**±0.1	94.6±0.1	**94.4**±0.1	**96.6**±0.1	**98.2**±0.1	**95.0**±0.0	**92.9**±0.2	**95.6**
7.	✓	✓	✓	✓	**95.9**±0.0	**97.0**±0.0	**94.7**±0.1	**94.4**±0.1	**96.6**±0.1	98.1±0.1	**95.0**±0.0	92.6±0.5	95.5
RICH-RESOURCE													
1.	✗	✓	✗	✗	96.7±0.1	98.6±0.1	94.9±0.1	96.3±0.0	97.0±0.1	98.6±0.1	96.3±0.0	91.9±0.4	96.3
2.	✗	✓	✓	✗	96.9±0.0	98.6±0.0	95.0±0.1	96.7±0.0	97.1±0.1	98.9±0.0	**96.7**±0.0	92.4±0.5	96.5
3.	✓	✗	✗	✗	96.3±0.0	97.8±0.0	94.9±0.1	95.8±0.0	96.6±0.1	98.4±0.1	95.5±0.1	86.9±0.2	95.3
4.	✓	✗	✓	✗	96.7±0.0	98.6±0.1	95.2±0.1	96.6±0.0	97.0±0.1	98.9±0.0	96.2±0.1	89.9±0.3	96.1
5.	✓	✗	✓	✓	96.8±0.0	98.6±0.0	95.2±0.1	96.7±0.0	97.1±0.1	99.0±0.1	96.3±0.1	90.1±0.2	96.2
6.	✓	✓	✓	✗	96.9±0.0	**98.8**±0.0	95.3±0.1	**97.0**±0.1	**97.3**±0.0	**99.1**±0.1	**96.7**±0.1	**92.6**±0.4	**96.7**
7.	✓	✓	✓	✓	**97.0**±0.1	**98.8**±0.0	**95.4**±0.1	**97.0**±0.1	**97.3**±0.1	**99.1**±0.1	**96.7**±0.1	92.5±0.4	**96.7**

Table 13: Averaged accuracy scores over 8 languages for UD POS tagging.

	Embeddings				BiLSTM-CRF models on UD POS tagging								
	M	F	W	C	ar	cs	de	en	es	fr	nl	ta	Avg.
Low-Resource: 10 Sentences													
1.	✗	✓	✗	✗	85.5±0.3	81.6±0.8	**82.9**±0.2	77.1±0.0	87.9±0.3	84.9±0.7	**70.7**±0.7	71.5±0.8	80.2
2.	✗	✓	✓	✗	**86.3**±0.5	**82.0**±0.2	82.5±0.4	**78.9**±0.4	**88.2**±0.3	**85.2**±0.4	70.5±1.1	72.6±0.5	**80.8**
3.	✓	✗	✗	✗	79.7±1.2	72.3±0.4	72.6±1.0	69.0±0.3	78.8±0.6	76.7±0.5	62.5±1.4	62.3±0.8	71.7
4.	✓	✗	✓	✗	81.7±0.4	74.2±0.8	74.2±0.8	70.9±0.4	80.7±0.3	77.8±0.7	62.7±0.7	66.4±1.2	73.6
5.	✓	✗	✓	✓	82.0±0.5	74.0±0.8	73.9±0.8	70.6±1.9	81.0±0.4	78.5±0.5	63.6±1.2	66.4±1.2	73.7
6.	✓	✓	✓	✗	85.8±0.5	79.5±0.4	80.8±0.5	75.3±0.4	86.6±0.2	82.2±0.4	70.3±0.8	72.5±0.4	79.1
7.	✓	✓	✓	✓	85.9±0.4	80.0±0.1	80.5±0.7	74.7±0.1	86.4±0.4	82.4±0.9	**70.7**±0.3	**72.8**±0.5	79.2
Low-Resource: 50 Sentences													
1.	✗	✓	✗	✗	91.7±0.2	90.5±0.1	**90.9**±0.3	87.6±0.2	93.0±0.1	91.6±0.3	81.8±0.7	85.6±0.5	89.1
2.	✗	✓	✓	✗	**91.9**±0.2	**91.2**±0.2	90.8±0.1	**87.9**±0.3	**93.5**±0.1	91.8±0.2	**83.2**±0.4	**86.5**±0.3	**89.6**
3.	✓	✗	✗	✗	87.4±0.4	84.2±0.7	83.6±0.2	80.1±0.4	88.3±0.5	86.4±0.4	72.6±0.3	76.1±0.7	82.3
4.	✓	✗	✓	✗	89.4±0.4	86.3±0.3	85.3±0.9	81.0±0.1	89.7±0.4	88.4±1.0	74.2±1.6	80.3±0.3	84.3
5.	✓	✗	✓	✓	89.4±0.4	86.6±0.6	85.0±0.6	81.5±0.6	89.6±0.3	88.1±0.5	75.1±0.6	80.7±0.6	84.5
6.	✓	✓	✓	✗	91.5±0.2	90.7±0.1	90.2±0.3	85.7±0.1	92.8±0.5	91.2±0.5	82.1±0.6	**86.5**±0.3	88.8
7.	✓	✓	✓	✓	91.5±0.1	90.7±0.5	90.3±0.3	85.3±0.1	92.4±0.2	**91.9**±0.6	82.5±0.2	86.0±0.7	88.8
Low-Resource: 100 Sentences													
1.	✗	✓	✗	✗	93.3±0.1	93.5±0.2	**92.0**±0.3	89.8±0.1	94.0±0.1	94.2±0.2	87.4±0.3	88.3±0.4	91.6
2.	✗	✓	✓	✗	**93.6**±0.1	93.6±0.1	91.8±0.3	**90.0**±0.1	**94.5**±0.1	**94.5**±0.1	**88.5**±0.1	88.7±0.3	**91.9**
3.	✓	✗	✗	✗	90.2±0.2	88.4±0.1	85.2±0.3	84.6±0.2	90.5±0.2	90.0±0.3	78.0±0.7	80.8±0.3	86.0
4.	✓	✗	✓	✗	91.6±0.1	90.6±0.1	87.4±0.3	85.5±0.1	92.5±0.5	91.7±0.2	81.7±0.6	85.2±0.4	88.3
5.	✓	✗	✓	✓	91.8±0.1	91.1±0.5	87.4±0.4	86.1±0.1	92.2±0.1	92.1±0.8	81.2±1.1	85.5±0.2	88.4
6.	✓	✓	✓	✗	93.5±0.2	**93.7**±0.3	91.4±0.2	89.4±0.6	94.3±0.2	94.2±0.2	86.9±0.3	**89.6**±0.4	91.6
7.	✓	✓	✓	✓	**93.6**±0.1	**93.7**±0.3	91.7±0.2	89.3±0.6	94.4±0.1	94.3±0.3	87.0±0.8	89.4±0.2	91.7
Low-Resource: 500 Sentences													
1.	✗	✓	✗	✗	95.2±0.1	96.1±0.1	94.1±0.1	92.6±0.1	95.9±0.0	97.0±0.1	92.8±0.2	91.9±0.4	94.4
2.	✗	✓	✓	✗	95.3±0.1	**96.3**±0.1	**94.3**±0.1	92.6±0.1	95.9±0.1	97.2±0.1	**93.3**±0.1	92.5±0.2	**94.7**
3.	✓	✗	✗	✗	93.4±0.1	92.6±0.0	89.6±0.2	89.7±0.0	93.8±0.2	95.5±0.1	89.5±0.2	86.5±0.2	91.3
4.	✓	✗	✓	✗	94.5±0.2	94.7±0.2	91.4±0.1	91.0±0.0	95.1±0.1	96.7±0.0	91.3±0.2	89.9±0.3	93.1
5.	✓	✗	✓	✓	94.8±0.1	95.0±0.2	91.5±0.1	91.1±0.1	95.2±0.1	97.0±0.1	91.6±0.1	90.0±0.4	93.3
6.	✓	✓	✓	✗	**95.5**±0.0	96.2±0.1	93.8±0.1	92.7±0.1	95.9±0.2	**97.5**±0.1	**93.3**±0.1	92.3±0.2	**94.7**
7.	✓	✓	✓	✓	**95.5**±0.1	96.1±0.1	93.9±0.1	**92.8**±0.1	**96.2**±0.1	97.4±0.1	**93.3**±0.2	**92.7**±0.2	**94.7**
Low-Resource: 1000 Sentences													
1.	✗	✓	✗	✗	95.6±0.0	96.8±0.0	**94.6**±0.1	93.5±0.1	96.3±0.1	97.5±0.1	94.5±0.1	92.0±0.5	95.1
2.	✗	✓	✓	✗	95.8±0.1	96.9±0.0	94.5±0.0	93.8±0.1	96.3±0.1	97.9±0.1	94.8±0.1	91.9±0.4	95.2
3.	✓	✗	✗	✗	94.2±0.1	93.9±0.1	90.9±0.2	91.4±0.0	94.7±0.1	96.7±0.2	92.5±0.1	86.4±0.3	92.6
4.	✓	✗	✓	✗	95.1±0.0	95.9±0.1	92.3±0.1	92.8±0.0	95.9±0.2	97.6±0.2	93.9±0.1	89.7±0.4	94.2
5.	✓	✗	✓	✓	95.2±0.1	96.1±0.1	92.5±0.2	93.3±0.1	96.1±0.1	97.5±0.1	94.0±0.1	89.8±0.4	94.3
6.	✓	✓	✓	✗	**95.9**±0.0	96.9±0.1	94.4±0.2	**94.2**±0.2	**96.5**±0.1	**98.1**±0.1	**95.0**±0.1	**92.7**±0.2	**95.5**
7.	✓	✓	✓	✓	**95.9**±0.1	**97.0**±0.0	94.5±0.1	94.1±0.1	**96.5**±0.0	**98.1**±0.1	**95.0**±0.1	92.5±0.3	**95.5**
Rich-Resource													
1.	✗	✓	✗	✗	96.7±0.0	98.6±0.0	95.0±0.1	96.4±0.1	97.0±0.0	98.5±0.1	96.3±0.0	92.1±0.5	96.3
2.	✗	✓	✓	✗	96.9±0.0	98.7±0.0	95.0±0.1	96.7±0.1	97.1±0.0	98.9±0.0	96.6±0.0	92.4±0.4	96.5
3.	✓	✗	✗	✗	96.3±0.0	97.8±0.0	94.9±0.0	95.8±0.1	96.6±0.1	98.4±0.1	95.4±0.1	86.7±0.3	95.2
4.	✓	✗	✓	✗	96.7±0.0	98.6±0.0	95.3±0.1	96.6±0.1	97.0±0.1	99.0±0.0	96.2±0.1	89.8±0.3	96.1
5.	✓	✗	✓	✓	96.8±0.1	98.6±0.0	95.1±0.0	96.7±0.0	97.0±0.1	99.0±0.1	96.3±0.1	90.0±0.6	96.2
6.	✓	✓	✓	✗	**97.0**±0.0	**98.8**±0.0	**95.4**±0.1	97.0±0.0	**97.3**±0.1	**99.1**±0.1	**96.7**±0.1	**92.7**±0.3	**96.7**
7.	✓	✓	✓	✓	**97.0**±0.0	**98.8**±0.0	**95.4**±0.1	**97.1**±0.0	**97.3**±0.1	**99.1**±0.0	**96.7**±0.1	92.6±0.1	**96.7**

Table 14: Averaged accuracy scores over 8 languages for UD POS tagging with the CRF layer.

	Embeddings				MaxEnt models on chunking		
	M	F	W	C	de	en	Avg.
Low-Resource: 10 Sentences							
1.	✗	✓	✗	✗	**83.3**±0.6	71.8±0.3	77.6
2.	✗	✓	✓	✗	83.2±0.6	**73.3**±0.6	**78.2**
3.	✓	✗	✗	✗	64.9±1.1	65.1±1.0	65.0
4.	✓	✗	✓	✗	66.3±3.0	67.7±0.6	67.0
5.	✓	✗	✓	✓	66.5±1.8	67.0±0.3	66.7
6.	✓	✓	✓	✗	81.3±1.6	70.6±0.1	75.9
7.	✓	✓	✓	✓	81.4±0.9	70.8±0.5	76.1
Low-Resource: 50 Sentences							
1.	✗	✓	✗	✗	**87.9**±0.3	80.6±0.7	84.3
2.	✗	✓	✓	✗	87.6±0.4	**82.2**±0.0	**84.9**
3.	✓	✗	✗	✗	80.5±1.3	74.1±0.2	77.3
4.	✓	✗	✓	✗	80.6±0.8	76.8±0.2	78.7
5.	✓	✗	✓	✓	81.3±0.5	76.2±0.3	78.7
6.	✓	✓	✓	✗	86.2±0.7	80.3±0.3	83.2
7.	✓	✓	✓	✓	86.0±1.0	80.3±0.3	83.1
Low-Resource: 100 Sentences							
1.	✗	✓	✗	✗	**88.8**±0.3	82.7±0.0	85.8
2.	✗	✓	✓	✗	**88.8**±0.4	**83.7**±0.0	**86.3**
3.	✓	✗	✗	✗	84.2±0.4	75.7±0.4	79.9
4.	✓	✗	✓	✗	84.9±0.5	79.1±0.4	82.0
5.	✓	✗	✓	✓	85.0±0.4	79.3±0.2	82.2
6.	✓	✓	✓	✗	88.3±0.4	82.3±0.2	85.3
7.	✓	✓	✓	✓	88.4±0.3	82.4±0.0	85.4
Low-Resource: 500 Sentences							
1.	✗	✓	✗	✗	91.3±0.1	86.2±0.2	88.7
2.	✗	✓	✓	✗	91.2±0.1	**86.8**±0.0	**89.0**
3.	✓	✗	✗	✗	89.7±0.1	82.7±0.1	86.2
4.	✓	✗	✓	✗	90.5±0.1	84.9±0.2	87.7
5.	✓	✗	✓	✓	90.3±0.3	85.2±0.0	87.8
6.	✓	✓	✓	✗	**91.4**±0.1	86.6±0.0	**89.0**
7.	✓	✓	✓	✓	**91.4**±0.1	86.7±0.1	**89.0**
Low-Resource: 1000 Sentences							
1.	✗	✓	✗	✗	**92.2**±0.1	86.6±0.1	89.4
2.	✗	✓	✓	✗	92.1±0.1	**87.1**±0.1	**89.6**
3.	✓	✗	✗	✗	90.6±0.1	84.0±0.1	87.3
4.	✓	✗	✓	✗	91.4±0.1	85.7±0.1	88.6
5.	✓	✗	✓	✓	91.5±0.2	86.0±0.1	88.7
6.	✓	✓	✓	✗	**92.2**±0.1	**87.1**±0.1	**89.6**
7.	✓	✓	✓	✓	**92.2**±0.1	**87.1**±0.2	**89.6**
Rich-Resource							
1.	✗	✓	✗	✗	94.1±0.1	90.5±0.0	92.3
2.	✗	✓	✓	✗	**94.2**±0.1	**91.7**±0.1	**92.9**
3.	✓	✗	✗	✗	93.0±0.1	89.4±0.0	91.2
4.	✓	✗	✓	✗	93.6±0.1	91.3±0.1	92.5
5.	✓	✗	✓	✓	93.8±0.1	91.4±0.0	92.6
6.	✓	✓	✓	✗	94.0±0.1	**91.7**±0.0	**92.9**
7.	✓	✓	✓	✓	94.1±0.1	91.6±0.1	**92.9**

Table 15: Averaged F1 scores over 2 languages for chunking.

	Embeddings				CRF models on chunking		
	M	F	W	C	de	en	Avg.
Low-Resource: 10 Sentences							
1.	✗	✓	✗	✗	**83.0**±0.8	70.1±0.1	**76.5**
2.	✗	✓	✓	✗	82.6±1.1	**70.5**±0.1	**76.5**
3.	✓	✗	✗	✗	68.0±2.1	62.9±1.2	65.4
4.	✓	✗	✓	✗	71.2±1.8	62.1±0.6	66.6
5.	✓	✗	✓	✓	71.0±1.0	61.2±0.8	66.1
6.	✓	✓	✓	✗	81.2±0.9	64.9±3.2	73.0
7.	✓	✓	✓	✓	81.3±0.8	66.3±2.9	73.8
Low-Resource: 50 Sentences							
1.	✗	✓	✗	✗	**88.0**±0.4	79.9±0.2	84.0
2.	✗	✓	✓	✗	87.9±0.3	**80.9**±0.4	**84.4**
3.	✓	✗	✗	✗	80.8±0.8	68.7±0.7	74.8
4.	✓	✗	✓	✗	82.8±1.2	71.1±1.1	76.9
5.	✓	✗	✓	✓	83.0±1.5	71.4±1.1	77.2
6.	✓	✓	✓	✗	86.2±1.3	73.6±0.6	79.9
7.	✓	✓	✓	✓	86.5±0.5	74.2±1.4	80.3
Low-Resource: 100 Sentences							
1.	✗	✓	✗	✗	**89.0**±0.5	**82.9**±0.5	**85.9**
2.	✗	✓	✓	✗	**89.0**±0.4	82.6±0.2	85.8
3.	✓	✗	✗	✗	84.7±0.5	74.8±0.6	79.8
4.	✓	✗	✓	✗	86.0±0.4	75.4±1.1	80.7
5.	✓	✗	✓	✓	86.1±0.4	74.6±1.6	80.4
6.	✓	✓	✓	✗	**89.0**±0.6	79.7±0.5	84.3
7.	✓	✓	✓	✓	88.3±0.7	77.7±0.6	83.0
Low-Resource: 500 Sentences							
1.	✗	✓	✗	✗	91.5±0.2	86.1±0.2	88.8
2.	✗	✓	✓	✗	91.4±0.1	**87.1**±0.1	**89.3**
3.	✓	✗	✗	✗	90.1±0.2	83.3±0.3	86.7
4.	✓	✗	✓	✗	90.8±0.1	84.8±0.2	87.8
5.	✓	✗	✓	✓	90.8±0.1	84.8±0.2	87.8
6.	✓	✓	✓	✗	91.5±0.1	86.1±0.1	88.8
7.	✓	✓	✓	✓	**91.6**±0.1	85.9±0.3	88.7
Low-Resource: 1000 Sentences							
1.	✗	✓	✗	✗	92.4±0.1	86.7±0.2	89.6
2.	✗	✓	✓	✗	92.4±0.1	**87.2**±0.2	**89.8**
3.	✓	✗	✗	✗	91.0±0.0	84.0±0.0	87.5
4.	✓	✗	✓	✗	91.5±0.1	85.6±0.2	88.6
5.	✓	✗	✓	✓	91.7±0.2	85.9±0.1	88.8
6.	✓	✓	✓	✗	92.4±0.1	86.9±0.1	89.6
7.	✓	✓	✓	✓	**92.5**±0.1	86.8±0.1	89.7
Rich-Resource							
1.	✗	✓	✗	✗	**94.4**±0.0	91.0±0.1	92.7
2.	✗	✓	✓	✗	**94.4**±0.1	**92.0**±0.0	**93.2**
3.	✓	✗	✗	✗	93.2±0.1	90.2±0.1	91.7
4.	✓	✗	✓	✗	93.8±0.1	91.7±0.0	92.8
5.	✓	✗	✓	✓	94.0±0.1	91.7±0.0	92.8
6.	✓	✓	✓	✗	94.2±0.1	91.8±0.0	93.0
7.	✓	✓	✓	✓	94.3±0.1	91.9±0.1	93.1

Table 16: Averaged F1 scores over 2 languages for chunking with the CRF layer.

EMBEDDINGS					Languages								
M	F	W	C	MF	NER								
					ar	cs	de	en	es	fr	nl	ta	Avg.
✓	✓	✓	✓	✗	84.6±0.2	88.4±0.2	**85.5**±0.3	81.7±0.2	**90.6**±0.2	**88.4**±0.5	**90.1**±0.3	**85.5**±0.3	**86.8**
✓	✗	✓	✓	✓	83.6±0.8	87.7±0.3	84.2±0.3	81.4±0.1	90.0±0.1	87.9±0.1	89.9±0.4	84.2±0.2	86.1
✓	✓	✓	✓	✓	**84.8**±0.5	**88.6**±0.2	85.1±0.3	**82.0**±0.2	90.3±0.4	88.1±0.3	**90.1**±0.2	85.3±0.3	**86.8**
					POS TAGGING								
					ar	cs	de	en	es	fr	nl	ta	Avg.
✓	✓	✓	✓	✗	**97.0**±0.1	**98.8**±0.0	**95.4**±0.1	**97.0**±0.1	**97.3**±0.1	**99.1**±0.1	**96.7**±0.1	92.5±0.4	**96.7**
✓	✗	✓	✓	✓	96.8±0.0	98.7±0.0	95.2±0.1	96.6±0.1	97.2±0.0	99.0±0.0	96.5±0.1	91.2±0.3	96.4
✓	✓	✓	✓	✓	**97.0**±0.0	**98.8**±0.0	95.3±0.0	96.9±0.1	**97.3**±0.1	**99.1**±0.0	**96.7**±0.1	**92.7**±0.5	**96.7**
					CHUNKING								
					de	en							Avg.
✓	✓	✓	✓	✗	94.0±0.0	91.5±0.0							92.8
✓	✗	✓	✓	✓	94.1±0.1	91.6±0.1							**92.9**
✓	✓	✓	✓	✓	**94.2**±0.1	**91.7**±0.1							**92.9**

Table 17: Detailed comparison for additionally concatenating **MF** with **All**. **MF** represents the M-Flair embeddings.