Towards Robust Named Entity Recognition for Historic German

Stefan Schweter and Johannes Baiter
Bayerische Staatsbibliothek München
Digital Library/Munich Digitization Center
80539 Munich, Germany
{stefan.schweter, johannes.baiter}@bsb-muenchen.de

Abstract

Recent advances in language modeling using deep neural networks have shown that these models learn representations, that vary with the network depth from morphology to semantic relationships like co-reference. We apply pre-trained language models to low-resource named entity recognition for Historic German. We show on a series of experiments that character-based pre-trained language models do not run into trouble when faced with low-resource datasets. Our pre-trained character-based language models improve upon classical CRF-based methods and previous work on Bi-LSTMs by boosting F1 score performance by up to 6%. Our pre-trained language and NER models are publicly available[1].

1 Introduction

Named entity recognition (NER) is a central component in natural language processing tasks. Identifying named entities is a key part in systems e.g. for question answering or entity linking. Traditionally, NER systems are built using conditional random fields (CRFs). Recent systems are using neural network architectures like bidirectional LSTM with a CRF-layer ontop and pre-trained word embeddings (Ma and Hovy, 2016; Lample et al., 2016a; Reimers and Gurevych, 2017; Lin et al., 2017).

Pre-trained word embeddings have been shown to be of great use for downstream NLP tasks (Mikolov et al., 2013; Pennington et al., 2014). Many recently proposed approaches go beyond these pre-trained embeddings. Recent works have proposed methods that produce different representations for the same word depending on its contextual usage (Peters et al., 2017, 2018a; Akbik et al., 2018; Devlin et al., 2018). These methods have shown to be very powerful in the fields of named entity recognition, coreference resolution, part-of-speech tagging and question answering, especially in combination with classic word embeddings.

Our paper is based on the work of Riedl and Padó (2018). They showed how to build a model for German named entity recognition (NER) that performs at the state of the art for both contemporary and historical texts. Labeled historical texts for German named entity recognition are a low-resource domain. In order to achieve robust state-of-the-art results for historical texts they used transfer-learning with labeled data from other high-resource domains like CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003) or GermEval (Benikova et al., 2014). They showed that using Bi-LSTM with a CRF as the top layer and word embeddings outperforms CRFs with hand-coded features in a big-data situation.

We build up upon their work and use the same low-resource datasets for Historic German. Furthermore, we show how to achieve new state-of-the-art results for Historic German named entity recognition by using only unlabeled data via pre-trained language models and word embeddings. We also introduce a novel language model pre-training objective, that uses only contemporary texts for training to achieve comparable state-of-the-art results on historical texts.

2 Model

In this paper, we use contextualized string embeddings as proposed by Akbik et al. (2018), as they have shown to be very effective in named entity recognition. We use the FLAIR[2] (Akbik et al., 2018) library to train all NER and pre-trained language models. We use FastText (Wikipedia and

[1] https://github.com/stefan-it/historic-ner

[2] https://github.com/zalandoresearch/flair

Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019), pages 96–103
Florence, Italy, August 2, 2019.

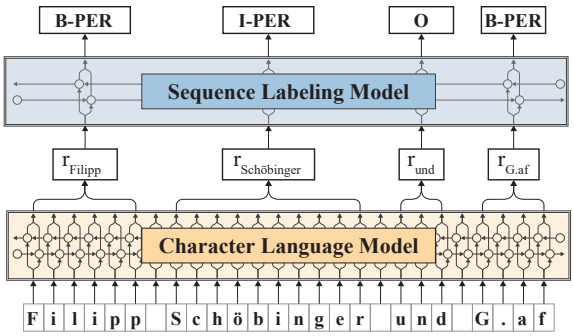

Figure 1: High level overview of our used model. A sentence is input as a character sequence into a pre-trained bidirectional character language model. From this LM, we retrieve for each word a contextual embedding that we pass into a vanilla Bi-LSTM-CRF.

Crawl) as word embeddings. FLAIR allows us to easily combine ("stacking") different embeddings types. For instance, Lample et al. (2016b) combine word embeddings with character features. In our experiments we combined several embedding types and language models. Contextualized string embeddings were trained with a forward and backward character-based language model (LSTM) on two historic datasets. This process is further called "pre-training". We use a Bi-LSTM with CRF on top as proposed by Huang et al. (2015). A high level system overview of our used model is shown in figure 1.

3 Datasets

We use the same two datasets for Historic German as used by Riedl and Padó (2018). These datasets are based on historical texts that were extracted (Neudecker, 2016) from the Europeana collection of historical newspapers[3]. The first corpus is the collection of Tyrolean periodicals and newspapers from the Dr Friedrich Temann Library (LFT). The LFT corpus consists of approximately 87,000 tokens from 1926. The second corpus is a collection of Austrian newspaper texts from the Austrian National Library (ONB). The ONB corpus consists of approximately 35,000 tokens from texts created between 1710 and 1873.

The tagset includes locations (LOC), organizations (ORG), persons (PER) and the remaining entities as miscellaneous (MISC). Figures 1-2 contain an overview of the number of named entities of the two datasets. No miscellaneous entities (MISC) are found in the ONB dataset and only a few are annotated in the LFT dataset. The two corpora pose three challenging problems: they are relatively small compared to contemporary corpora like CoNLL-2003 or GermEval. They also have a different language variety (German and Austrian) and they include a high rate of OCR errors[4] since they were originally printed in Gothic type-face (Fraktur), a low resource font, which has not been the main focus of recent OCR research.

Dataset	LOC	MISC	ORG	PER
Training	1,605	0	182	2,674
Development	207	0	10	447
Test	221	0	16	355

Table 1: Number of named entities in ONB dataset.

Dataset	LOC	MISC	ORG	PER
Training	3,998	2	2,293	4,009
Development	406	0	264	558
Test	441	1	324	506

Table 2: Number of named entities in LFT dataset.

4 Experiments

4.1 Experiment 1: Different Word Embeddings

In the first experiment we use different types of embeddings on the two datasets: (a) FastText embeddings trained on German Wikipedia articles, (b) FastText embeddings trained on Common Crawl and (c) character embeddings, as proposed by Lample et al. (2016b). We use pre-trained FastText embeddings[5] without subword information, as we found out that subword information could harm performance (0.8 to 1.5%) of our system in some cases.

Table 3 shows, that combining pre-trained FastText for Wikipedia and Common Crawl leads to a F1 score of 72.50% on the LFT dataset. Adding character embeddings has a positive impact of 2% and yields 74.50%. This result is higher than the reported one by Riedl and Padó (2018)

[3] https://www.europeana.eu/portal/de

[4] Typical OCR errors would be segmentation and hyphenation errors or misrecognition of characters (e.g. *B i f m a r c k* instead of *B i ſ m a r c k*).

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

	Configuration	F-Score
LFT	Wikipedia	69.59%
	Common Crawl	68.97%
	Wikipedia + Common Crawl	72.00%
	Wikipedia + Common Crawl + Character	**74.50%**
	Riedl and Padó (2018) (no transfer-learning)	69.62%
	Riedl and Padó (2018) (with transfer-learning)	74.33%
ONB	Wikipedia	75.80%
	CommonCrawl	78.70%
	Wikipedia + CommonCrawl	79.46%
	Wikipedia + CommonCrawl + Character	**80.48%**
	Riedl and Padó (2018) (no transfer-learning)	73.31%
	Riedl and Padó (2018) (with transfer-learning)	78.56%

Table 3: Results on LFT and ONB dataset with different configurations. Wikipedia and Common Crawl are pre-trained FastText word embeddings. The best configurations reported by Riedl and Padó (2018) used Wikipedia or Europeana word embeddings with subword information and character embeddings.

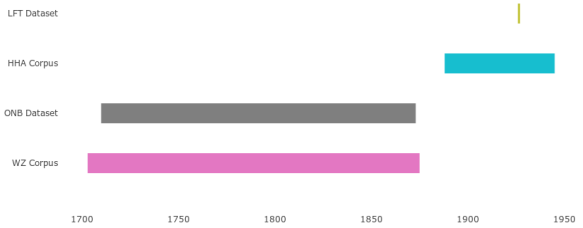

Figure 2: Temporal overlap for language model corpora and historic datasets.

(74.33%), who used transfer-learning with more labeled data. Table 3 also shows the same effect for ONB: combining Wikipedia and Common Crawl embeddings leads to 79.46% and adding character embeddings marginally improves the result to 80.48%. This result is also higher than the reported one by Riedl and Padó (2018) (78.56%).

4.2 Experiment 2: Language model pre-training

For the next experiments we train contextualized string embeddings as proposed by Akbik et al. (2018). We train language models on two datasets from the Europeana collection of historical newspapers. The first corpus consists of articles from the Hamburger Anzeiger newspaper (HHA) covering 741,575,357 tokens from 1888 - 1945. The second corpus consists of articles from the Wiener Zeitung newspaper (WZ) covering 801,543,845 tokens from 1703 - 1875. We choose the two corpora, because they have a temporal overlap with the LFT corpus (1926) and the ONB corpus (1710 - 1873). Figure 2 shows the temporal overlap for the language model corpora and the datasets used in the downstream task. There is a huge temporal overlap between the ONB dataset and the WZ corpus, whereas the overlap between the LFT dataset and the HHA corpus is relatively small.

Additionally we use the BERT model, that was trained on Wikipedia for 104 languages[6] for comparison. We perform a per-layer analysis of the multi-lingual BERT model on the development set to find the best layer for our task. For the German language model, we use the same pre-trained language model for German as used in Akbik et al. (2018). This model was trained on various sources (Wikipedia, OPUS) with a training data set size of half a billion tokens.

Table 4 shows that the temporal aspect of training data for the language models has deep impact on the performance. On LFT (1926) the language model trained on the HHA corpus (1888 - 1945) leads to a F1 score of 77.51%, which is a new state-of-the art result on this dataset. The result is 3.18% better than the result reported by Riedl and Padó (2018), which uses transfer-learning with more labeled training data. The language model trained on the WZ corpus (1703-1875) only achieves a F1 score of 75.60%, likely because the time period of the data used for pre-training (19th century) is too far removed from

[6] https://github.com/google-research/bert/blob/f39e881b169b9d53bea03d2d341b31707a6c052b/multilingual.md

	Configuration	Pre-trained LM	Pre-training data	F-Score
LFT (1926)	German	✓	Wikipedia, OPUS	76.04%
	Hamburger Anzeiger (HHA)	✓	Newspaper (1888 - 1945)	**77.51%**
	Wiener Zeitung (WZ)	✓	Newspaper (1703 - 1875)	75.60%
	Multi-lingual BERT	✓	Wikipedia	74.39%
	SMLM (synthetic corpus)	✓	Wikipedia	77.16%
	Riedl and Padó (2018)	-	-	69.62%
	Riedl and Padó (2018)†	-	-	74.33%
ONB (1710-1873)	German	✓	Wikipedia, OPUS	80.06%
	Hamburger Anzeiger (HHA)	✓	Newspaper (1888 - 1945)	83.28%
	Wiener Zeitung (WZ)	✓	Newspaper (1703 - 1875)	**85.31%**
	Multi-lingual BERT	✓	Wikipedia	77.19%
	SMLM (synthetic corpus)	✓	Wikipedia	82.15%
	Riedl and Padó (2018)	-	-	73.31%
	Riedl and Padó (2018)†	-	-	78.56%

Table 4: Results on LFT and ONB with different language models. The German language model refers to the model used in Akbik et al. (2018). We perform a per-layer analysis for BERT on the development set and use the best layer. For all experiments we also use pre-trained FastText embeddings on Wikipedia and Common Crawl as well as character embeddings. † indicates the usage of additional training data (GermEval) for transfer learning.

that of the downstream task (mid-1920s). Table 4 also shows the results of pre-trained language models on the ONB (1710 - 1873) dataset. The language models, that were trained on contemporary data like the German Wikipedia (Akbik et al., 2018) or multi-lingual BERT do not perform very well on the ONB dataset, which covers texts from the 18-19th century. The language model trained on the HHA corpus performs better, since there is a substantially temporal overlap with the ONB corpus. The language model trained on the WZ corpus (1703-1875) leads to the best results with a F1 score of 85.31%. This result is 6.75% better than the reported result by Riedl and Padó (2018), which again uses transfer-learning with additionally labeled training data.

4.3 Experiment 3: Synthetic Masked Language Modeling (SMLM)

We also consider the masked language modeling (MLM) objective of Devlin et al. (2018). However, this technique cannot be directly used, because they use a subword-based language model, in contrast to our character-based language model. We introduce a novel masked language modeling technique, synthetic masked language modeling (SMLM) that randomly adds noise during training.

The main motivation for using SMLM is to transfer a corpus from one domain (e.g. "clean" contemporary texts) into another (e.g. "noisy" historical texts). SMLM uses the vocabulary (characters) from the target domain and injects them into the source domain. With this technique it is possible to create a synthetic corpus, that "emulates" OCR errors or spelling mistakes without having any data from the target domain (except all possible characters as vocabulary). Furthermore, SMLM can also be seen as a kind of domain adaption.

To use SMLM we extract all vocabulary (characters) from the ONB and LFT datasets. We refer to these characters as target vocabulary. Then we obtained a corpus consisting of contemporary texts from Leipzig Corpora Collection (Goldhahn et al., 2012) for German. The resulting corpus has 388,961,352 tokens. During training, the following SMLM objective is used: Iterate overall characters in the contemporary corpus. Leave the character unchanged in 90% of the time. For the remaining 10% we employ the following strategy: in 20% of the time replace the character with a masked character, that does not exist in the target vocabulary. In 80% of the time we randomly replace the character by a symbol from the target vocabulary.

Table 4 shows that the language model trained with SMLM achieves the second best result on LFT with 77.16%. The ONB corpus is more challenging for SMLM, because it includes texts from

a totally different time period (18-19th century). SMLM achieves the third best result with a F-Score of 82.15%. This result is remarkable, because the language model itself has never seen texts from the 18-19th century. The model was trained on contemporary texts with SMLM only.

5 Data Analysis

	LM	Perplexity		F-Score
		Forward	Backward	
LFT	German	8.30	8.7	76.04%
	HHA	**6.31**	**6.64**	**77.51%**
	WZ	6.72	6.97	75.60%
	Synthetic	7.87	8.20	77.16%
ONB	German	8.58	8.77	80.06%
	HHA	6.71	7.22	83.28%
	WZ	**4.72**	**4.95**	**85.31%**
	Synthetic	8.65	9.64	82.15%

Table 5: Averaged perplexity for all sentences in the test dataset for LFT for all pre-trained language models.

The usage of pre-trained character-based language models boosts performance for both LFT and ONB datasets. The results in table 4 show, that the selection of the language model corpus plays an important role: a corpus with a large degree of temporal overlap with the downstream task performs better than corpus with little to no temporal overlap. In order to compare our trained language models with each other, we measure both the perplexity of the forward language model and the backward language model on the test dataset for LFT and ONB. The perplexity for each sentence in the test dataset is calculated and averaged. The results for LFT and ONB are shown in table 5. For all language models (except one) there is a clear correlation between overall perplexity and F1 score on the test dataset: lower perplexity (both for forward and backward language model) yields better performance in terms of the F1 score on the downstream NER tasks. But this assumption does not hold for the language model that was trained on synthetic data via SMLM objective: The perplexity for this language model (both forward and backward) is relatively high compared to other language models, but the F1 score results are better than some other language models with lower perplexity. This variation can be observed both on LFT and ONB test data. We leave this anomaly here as an open question: Is perplexity a good measure for comparing language models and a useful indicator for their results on downstream tasks?

The previous experiments show, that language model pre-training does work very well, even for domains with low data resources. Cotterell and Duh (2017) showed that using CRF-based methods outperform traditional Bi-LSTM in low-resource settings. We argue that this shortcoming can now be eliminated by using Bi-LSTMs in combination with pre-trained language models. Our experiments also showed, that pre-trained language models can also help to improve performance, even when no training data for the target domain is used (SMLM objective).

6 Conclusion

In this paper we have studied the influence of using language model pre-training for named entity recognition for Historic German. We achieve new state-of-the-art results using carefully chosen training data for language models.

For a low-resource domain like named entity recognition for Historic German, language model pre-training can be a strong competitor to CRF-only methods as proposed by Cotterell and Duh (2017). We showed that language model pre-training can be more effective than using transfer-learning with labeled datasets.

Furthermore, we introduced a new language model pre-training objective, synthetic masked language model pre-training (SMLM), that allows a transfer from one domain (contemporary texts) to another domain (historical texts) by using only the same (character) vocabulary. Results showed that using SMLM can achieve comparable results for Historic named entity recognition, even when they are only trained on contemporary texts.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful and valuable comments.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

Darina Benikova, Chris Biemann, and Marc Reznicek. 2014. Nosta-d named entity annotation for german: Guidelines and dataset. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*. European Language Resources Association (ELRA).

Ryan Cotterell and Kevin Duh. 2017. Low-resource named entity recognition with cross-lingual, character-level neural conditional random fields. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 91–96. Asian Federation of Natural Language Processing.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In *In Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC12*.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *arXiv e-prints*, page arXiv:1508.01991.

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. Association for Computational Linguistics.

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

Bill Y. Lin, Frank Xu, Zhiyi Luo, and Kenny Zhu. 2017. Multi-channel bilstm-crf model for emerging named entity recognition in social media. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 160–165. 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. Association for Computational Linguistics.

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.

Clemens Neudecker. 2016. An open corpus for named entity recognition in historic newspapers. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

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, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1765. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, and Christopher Clark Kenton Lee Luke Zettlemoyer Mohit Iyyer, Matt Gardner. 2018a. Deep contextualized word representations. *6th International Conference on Learning Representations*.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018b. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509. 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. 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. Association for Computational Linguistics.

Martin Riedl and Sebastian Padó. 2018. A named entity recognition shootout for german. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 120–125. Association for Computational Linguistics.

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*.

A Supplemental Material

A.1 Language model pre-training

Table 6 shows the parameters that we used for training our language models. As our character-based language model relies on raw text, no pre-processing steps like tokenization are needed. We use 1/500 of the complete corpus for development data and another 1/500 for test data during the language model training.

Parameter	Value
LSTM hidden size	2048
LSTM layer	1
Dropout	0.1
Sequence length (characters)	250
Mini batch size	1
Epochs	1

Table 6: Parameters used for language model pre-training.

A.1.1 SMLM objective

Original sentence
Dann habe der Mann erzählt, wie er in München am Bahnhof mit Blumen begrüßt worden sei.
Sentence after SMLM transformation
Qa ¶n hab5 der MaRy erzählt nie er in MñchenIam Bahnhof mit Blumen begrüß(Corden se¶.

Figure 3: An example of the SMLM transformation for a given input sentence. The special character "¶" is used as masked character symbol.

Figure 3 shows the SMLM objective for a given input sentence and the corresponding output. We use the same parameters as shown in table 6 to train a language model with SMLM objective. We use different values of p in range of $[80, 90, 95]$ for leaving the character unchanged in the SMLM objective and found that $p = 90$ yields the best results.

A.2 Model parameters

Table 7 shows the parameters that we use for training a named entity recognition model with the FLAIR library. We reduce the learning rate by a factor of 0.5 with a patience of 3. This factor determines the number of epochs with no improvement after which learning rate will be reduced.

Parameter	Value
LSTM hidden size	512
Learning rate	0.1
Mini batch size	8
Max epochs	500
Optimizer	SGD

Table 7: Parameters used for training NER models.

A.3 BERT per-layer analysis

We experimentally found that using the last four layers as proposed in Devlin et al. (2018) for the feature-based approach does not work well. Thus, we perform a per-layer analysis that trains a model with a specific layer from the multi-lingual BERT model. Inspired by Peters et al. (2018b) we visualize the performance for each layer of the BERT model. Figure 4 shows the performance of each layer for the LFT development dataset, figure 5 for the ONB development dataset.

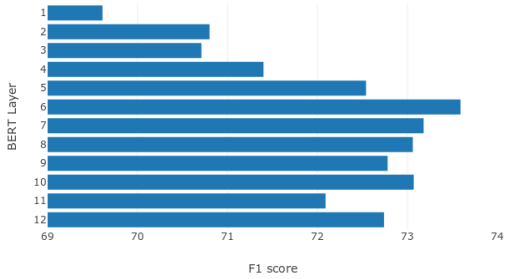

Figure 4: BERT per-layer analysis on the LFT development dataset.

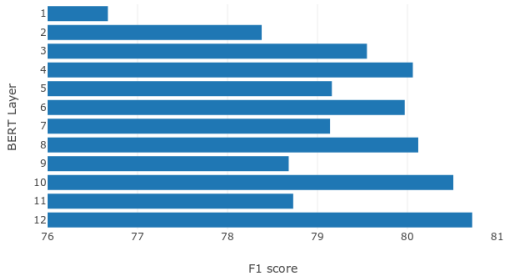

Figure 5: BERT per-layer analysis on the ONB development dataset.

A.4 Evaluation

We train all NER models with IOBES (Ratinov and Roth, 2009) tagging scheme. In the prediction

step we convert IOBES tagging scheme to IOB, in order to use the offical CoNLL-2003 evaluation script[7]. For all NER models we train and evaluate 3 runs and report an averaged F1 score.

A.5 Negative Results

We briefly describe a few ideas we implemented that did not seem to be effective in initial experiments. These findings are from early initial experiments. We did not pursue these experiments further after first attempts, but some approaches could be effective with proper hyperparameter tunings.

- **FastText embeddings with subword information**: We use subword information with FastText embeddings trained on Wikipedia articles. On LFT this model was 0.81% behind a model trained with FastText embeddings without subword information. On ONB the difference was 1.56%. Using both FastText embeddings trained on Wikipedia and CommonCrawl with subword information caused out-of-memory errors on our system with 32GB of RAM.

- **ELMo Transformer**: We trained ELMo Transformer models as proposed by Peters et al. (2018b) for both HH and WZ corpus. We use the default hyperparameters as proposed by Peters et al. (2018b) and trained a ELMo Transformer model for one epoch (one iteration over the whole corpus) with a vocabulary size of 1,094,628 tokens both for the HH and WZ corpus. We use the same model architecture like in previous experiments for training a NER model on both LFT and ONB. On LFT we achieved a F1 score of 72.18%, which is 5.33% behind our new state-of-the-art result. On ONB we achieved a F1 score of 75.72%, which is 9.59% behind our new state-of-the-art result. We assume that training a ELMo Transformer model for more epochs would lead to better results.

[7] https://www.clips.uantwerpen.be/conll2003/ner/bin/conlleval