

# FLERT: Document-Level Features for Named Entity Recognition

Stefan Schweter

schweter.ml  
stefan@schweter.eu

Alan Akbik

Humboldt-Universität zu Berlin  
alan.akbik@hu-berlin.de

## Abstract

Current state-of-the-art approaches for named entity recognition (NER) using BERT-style transformers typically use one of two different approaches: (1) The first *fine-tunes* the transformer itself on the NER task and adds only a simple linear layer for word-level predictions. (2) The second uses the transformer only to provide *features* to a standard LSTM-CRF sequence labeling architecture and thus performs no fine-tuning. In this paper, we perform a comparative analysis of both approaches in a variety of settings currently considered in the literature. In particular, we evaluate how well they work when *document-level* features are leveraged. Our evaluation on the classic CoNLL benchmark datasets for 4 languages shows that document-level features significantly improve NER quality and that fine-tuning generally outperforms the feature-based approaches. We present recommendations for parameters as well as several new state-of-the-art numbers. Our approach is integrated into the FLAIR framework to facilitate reproduction of our experiments.

## 1 Introduction

Named entity recognition (NER) is the well-studied NLP task of predicting shallow semantic labels for sequences of words, used for instance for identifying the names of persons, locations and organizations in text. Current approaches for NER often leverage pre-trained transformer architectures such as BERT (Devlin et al., 2019) or XLM (Lample and Conneau, 2019).

**Document-level features.** While NER is traditionally modeled at the sentence-level, transformer-based models offer a natural option for capturing document-level features by passing a sentence with its surrounding context. As Figure 1 shows, this context can then influence the word representations of a sentence: The example sentence "I love Paris"

is passed through the transformer together with the next sentence that begins with "The city is", potentially helping to resolve the ambiguity of the word "Paris". A number of prior works have employed such document-level features (Devlin et al., 2019; Virtanen et al., 2019; Yu et al., 2020) but only in combination with other contributions and thus have not evaluated the impact of using document-level features in isolation.

**Contributions.** With this paper, we close this experimental gap and present an evaluation of document-level features for NER. As there are two conceptually very different approaches for transformer-based NER that are currently used across the literature, we evaluate document-level features in both:

1. In the first, we *fine-tune* the transformer itself on the NER task and only add a simple linear layer for word-level predictions (Devlin et al., 2019).
2. In the second, we use the transformer only to provide *features* to a standard LSTM-CRF sequence labeling architecture (Huang et al., 2015) and thus perform no fine-tuning.

We discuss the differences between both approaches and explore best hyperparameters for each. In their best determined setup, we then perform a comparative evaluation using document-level features from different transformer models. We find that document-level features significantly improve NER quality and that fine-tuning generally outperforms feature-based approaches.

We use document-level features in the best determined fine-tuning setup and report a number of new state-of-the-art scores on the classic CoNLL benchmark datasets. Our approach is integrated into the FLAIR framework (Akbik et al., 2019a) to facilitate reproduction of our experiments.

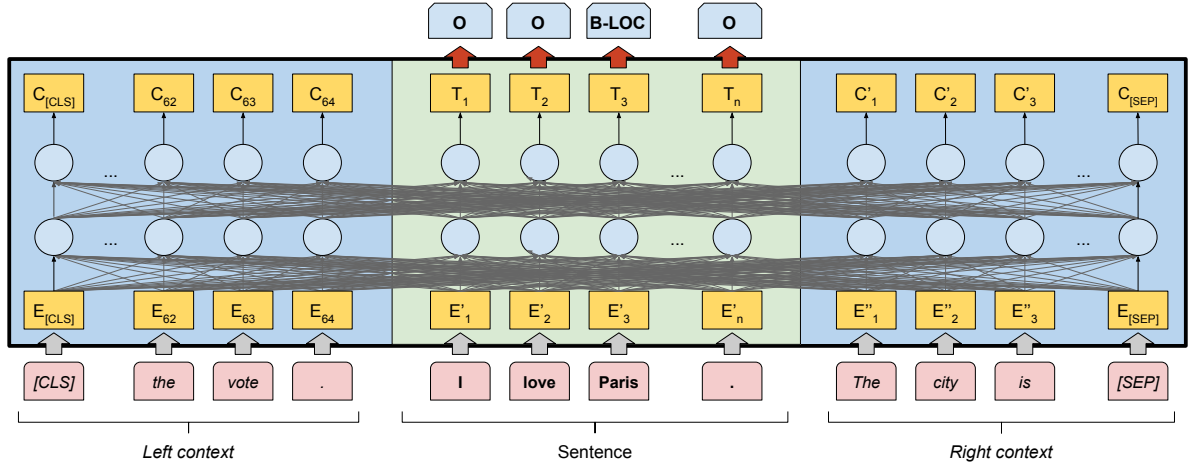


Figure 1: To obtain document-level features for a sentence that we wish to tag (“I love Paris”, shaded green), we add 64 tokens of left and right tokens each (shaded blue). As self-attention is calculated over all input tokens, the representations for the sentence’s tokens are influenced by the left and right context.

## 2 Document-Level Features

In a transformer-based architecture, document-level features can easily be realized by passing a sentence with its surrounding context to obtain word embeddings, as illustrated in Figure 1.

**Prior approaches.** This approach was first employed by [Devlin et al. \(2019\)](#) with what they described as a “maximal document context”, though technical details were not listed. Subsequent work has used variants of this approach. For instance, [Virtanen et al. \(2019\)](#) experiment with adding the following (but not preceding) sentence as context to each sentence. [Yu et al. \(2020\)](#) instead use a 64 surrounding token window for each token in a sentence, thus calculating a large context on a per-token basis. By contrast, [Luoma and Pyysalo \(2020\)](#) adopt a multi-sentence view in which they evaluate different ways for passing sequences of multiple complete sentences through a transformer, evaluate the impact of the position of a sentence in this multi-sentence view and present ways to combine predictions from different windows and sentence positions.

**Our approach.** In this paper, we instead use a conceptually simple variant in which we create context on a per-sentence basis. That is, for each sentence we wish to classify, we add 64 subtokens of left and right context, as shown in Figure 1. This has computational and implementation advantages in that each sentence and its context need only be passed through the transformer once and that added context is limited to a relatively small window of 64 subtokens. Furthermore, we can still follow stan-

dard procedure in shuffling sentences at each epoch during training, since context is encoded on a per-sentence level. We use this approach throughout this paper.

## 3 Baseline Parameter Experiments

As mentioned in the introduction, there are two common architectures for transformer-based NER, namely fine-tuning and feature-based approaches. In this section, we briefly introduce the differences between both approaches and conduct a study to identify best hyperparameters for each. The best respective setups are then used in the final comparative evaluation in Section 4.

### 3.1 Setup

**Data set.** We use the development datasets of the CoNLL shared tasks ([Tjong Kim Sang and De Meulder, 2003](#); [Tjong Kim Sang, 2002](#)) for NER on four languages (English, German, Dutch and Spanish). Following [Yu et al. \(2020\)](#) we report results for both original and revisited dataset for German (denoted as  $DE_{06}$ ).

**Transformer model.** In all experiments in this section, we employ the multilingual XLM-RoBERTa (XLM-R) transformer model proposed by [Conneau et al. \(2019\)](#). We use the `xlm-roberta-large` model in our experiments, trained on 2.5TB of data from a cleaned Common Crawl corpus ([Wenzek et al., 2020](#)) for 100 different languages

**Embeddings (+WE).** For each setup we experiment with adding classic word embeddings that are concatenated to the word-level representations obtained from the transformer model. Following [Ak-](#)

Fine-tuning Approach	EN	DE	DE <sub>06</sub>	NL	ES
<i>XLM-R</i>	96.64 $\pm$ 0.14	89.06 $\pm$ 0.18	91.86 $\pm$ 0.41	93.41 $\pm$ 0.19	88.95 $\pm$ 0.19
+ WE	96.82 $\pm$ 0.13	88.96 $\pm$ 0.10	92.12 $\pm$ 0.10	93.51 $\pm$ 0.09	89.09 $\pm$ 0.36
+ Context	96.82 $\pm$ 0.07	<b>89.79</b> $\pm$ 0.13	<b>93.09</b> $\pm$ 0.06	94.19 $\pm$ 0.14	90.34 $\pm$ 0.27
+ WE + Context	<b>97.02</b> $\pm$ <b>0.09</b>	89.74 $\pm$ 0.46	92.83 $\pm$ 0.12	94.01 $\pm$ 0.27	90.17 $\pm$ 0.25
<i>XLM-R-CRF</i>	96.79 $\pm$ 0.11	88.52 $\pm$ 0.10	92.21 $\pm$ 0.07	93.61 $\pm$ 0.15	88.77 $\pm$ 0.20
+ WE	96.79 $\pm$ 0.15	88.84 $\pm$ 0.15	91.97 $\pm$ 0.09	93.36 $\pm$ 0.04	88.63 $\pm$ 0.47
+ Context	96.90 $\pm$ 0.06	89.67 $\pm$ 0.24	92.87 $\pm$ 0.21	94.16 $\pm$ 0.07	<b>90.56</b> $\pm$ 0.09
+ WE + Context	96.87 $\pm$ 0.00	89.69 $\pm$ 0.22	92.88 $\pm$ 0.26	<b>94.34</b> $\pm$ <b>0.13</b>	90.37 $\pm$ 0.14

Table 1: Evaluation of different variants using the fine-tuning approach. The evaluation is performed against the **development set** of all 4 languages of the CoNLL-03 shared task for NER.

bik et al. (2018), we use GLOVE embeddings (Pennington et al., 2014) for the English tasks and FAST-TEXT embeddings (Bojanowski et al., 2017) for all other languages.

### 3.2 First Approach: Fine-Tuning

Fine-tuning approaches typically only add a single linear layer to a transformer and fine-tune the entire architecture on the NER task. To bridge the difference between subtoken modeling and token-level predictions, they apply *subword pooling* to create token-level representations which are then passed to the final linear layer. A common subword pooling strategy is "first" (Devlin et al., 2019) which uses the representation of the first subtoken for the entire token. See Figure 2 for an illustration.

Parameter	Value
Transformer layers	last
Learning rate	5e-6
Mini batch size	4
Max epochs	20
Optimizer	AdamW
Scheduler	One-cycle LR
Subword pooling	first

Table 2: Parameters used for fine-tuning.

To train this architecture, prior works typically use the AdamW (Loshchilov and Hutter, 2019) optimizer, a very small learning rate and a small, fixed number of epochs as a hard-coded stopping criterion (Conneau et al., 2019). We adopt a one-cycle training strategy (Smith, 2018), inspired from the HuggingFace transformers (Wolf et al., 2019) implementation, in which the learning rate linearly decreases until it reaches 0 by the end of the training. Table 2 lists the architecture parameters we use across all our experiments.

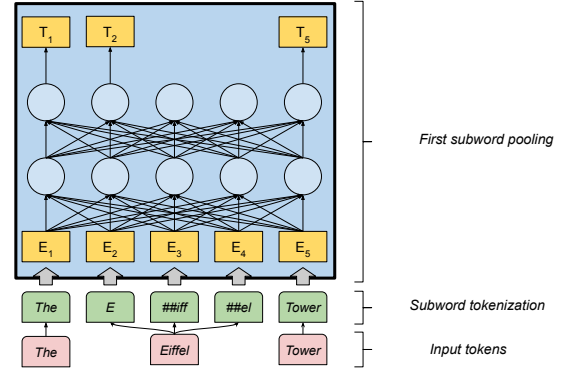


Figure 2: Illustration of first subword pooling. The input "The Eiffel Tower" is subword-tokenized, splitting "Eiffel" into three subwords (shaded green). Only the first ("E") is used as representation for "Eiffel".

Conceptually, fine-tuning approaches have the advantage that everything is modeled in a single architecture that is fine-tuned as a whole.

#### 3.2.1 Fine-Tuning Variants

We compare two variants of fine-tuning:

**XLM-R** In the first, we use the standard approach of adding a simple linear classifier on top of the transformer to directly predict tags.

**XLM-R-CRF** In the second, we evaluate if it is helpful to add a conditional random fields (CRF) decoder between the transformer and the linear classifier (Souza et al., 2019).

We evaluate both in all possible combinations of adding standard word embeddings ("WE") and using document-level features ("Context"). Results are listed in Table 1.

#### 3.2.2 Best Fine-Tuning Variant

As Table 1 shows, the best results are observed with different configurations on different datasets.

Feature-based Approach	EN
XLM-R (last-four-layers)	91.17 $\pm$ 0.29
+ WE	92.19 $\pm$ 0.46
+ Context	94.23 $\pm$ 0.19
+ WE + Context	94.61 $\pm$ 0.10
XLM-R (all-layer-mean)	94.37 $\pm$ 0.06
+ WE	95.63 $\pm$ 0.04
+ Context	96.09 $\pm$ 0.07
+ WE + Context	<b>96.53 <math>\pm</math> 0.10</b>

Table 3: Evaluation of feature-based approach (development set).

In particular, we find that using additional word embeddings and using a CRF decoder improves results only for some languages, and often only minimally so. On the other hand, the results clearly show document-level context to be of significant importance. We thus select the standard fine-tuning approach with document features (i.e. without word embeddings or the CRF) as fine-tuning architecture for our comparative evaluation in Section 4.

### 3.3 Second Approach: Feature-Based

Feature-based approaches instead use the transformer only to generate embeddings for each word in a sentence and use these as input into a standard sequence labeling architecture, most commonly a LSTM-CRF (Huang et al., 2015). The transformer weights remain frozen so that training is limited to the LSTM-CRF. Training typically uses SGD with a larger learning rate that is annealed against the development data. Training terminates when the learning rate becomes too small. Advantages of this approach include the use of a real stopping criterion and the relative ease of combining BERT embeddings with other types of word embeddings (e.g. "embedding stacking" (Akbik et al., 2019a)).

Parameter	Value
LSTM hidden size	256
Learning rate	0.1
Mini batch size	16
Max epochs	500
Optimizer	SGD
Subword pooling	first

Table 4: Parameters for feature-based approach.

The parameters used for training a feature-based model are shown in Table 4.

#### 3.3.1 Feature-Based Variants

We compare two variants to produce token features:

**All-layer-mean** In the first, we obtain embeddings for each token using mean pooling across all transformer layers, including the word embedding layer. This representation has the same length as the hidden size for each transformer layer. This approach is inspired by the ELMO-style (Peters et al., 2018) "scalar mix", that was shown to achieve better results than the best individual layer on most tasks (Liu et al., 2019a; Tenney et al., 2019).

**Last-four-layers** In the second, we follow Devlin et al. (2019) to only use the last four transformer layers for each token and concatenate their representations into a final representation for each token. This representation thus has four times the length of the transformer layer hidden size.

We again evaluate both in all possible combinations of adding standard word embeddings "(+WE)" and using document-level features "(+Context)".

#### 3.3.2 Best Feature-Based Variant

The results of parameter combinations for English<sup>2</sup> are shown in Table 3. The results clearly show that averaging over all layers outperforms using only the last four layers. In addition, the results clearly show that additionally using word embeddings as well as document-level context improves development set F1 score. For this reason, we chose a setup with all-layer-mean, word embeddings and document features for the feature-based experiments.

### 3.4 Summary of Best Configurations

This section identified the best configurations for the fine-tuning and feature-based approaches to NER using development data. In both cases, experiments clearly showed that using document-level features improves NER quality. For the feature-based approach, we also find that an all-layer-mean strategy as well as the addition of word embeddings yields the best results. We employ these configurations in our final evaluation in the next section.

## 4 Comparative Evaluation

Using the best identified configurations, we conduct a final comparative evaluation on the test splits

<sup>2</sup>Other languages omitted to preserve space, but they show the same clear results.



Approach	Context?	EN	DE	DE <sub>06</sub>	NL	Es
<i>XLM-R</i>						
- Feature-based	no	91.83 $\pm$ 0.06	82.88 $\pm$ 0.28	87.35 $\pm$ 0.17	89.87 $\pm$ 0.45	88.78 $\pm$ 0.08
- Feature-based	yes	93.12 $\pm$ 0.14	84.86 $\pm$ 0.11	89.88 $\pm$ 0.26	91.73 $\pm$ 0.21	88.98 $\pm$ 0.11
- Fine-tuning	no	92.79 $\pm$ 0.10	86.60 $\pm$ 0.43	90.04 $\pm$ 0.37	93.50 $\pm$ 0.15	89.94 $\pm$ 0.24
- Fine-tuning	yes	93.64 $\pm$ 0.05	86.99 $\pm$ 0.24	91.55 $\pm$ 0.07	93.99 $\pm$ 0.16	<b>90.14 <math>\pm</math> 0.14</b>
<i>XLM-R (+ Dev)</i>						
- Fine-tuning	yes	<b>93.96 <math>\pm</math> 0.11</b>	<b>88.21 <math>\pm</math> 0.43</b>	<b>92.29 <math>\pm</math> 0.18</b>	<b>94.66 <math>\pm</math> 0.12</b>	89.93 $\pm$ 0.03
<i>Monolingual*</i>						
- Fine-tuning	yes	93.84 $\pm$ 0.12	85.77 $\pm$ 0.13	89.56 $\pm$ 0.55	92.15 $\pm$ 0.12	89.16 $\pm$ 0.11
<i>Best published</i>						
Akbik et al. (2019b)	pooling	93.18 $\pm$ 0.09	-	88.27 $\pm$ 0.30	90.44 $\pm$ 0.20	-
Yu et al. (2020)	yes	93.5	86.4	90.3	93.7	90.3
Straková et al. (2019)	yes	93.38	85.10	-	92.69	88.81

Table 5: Comparative evaluation of best configurations of fine-tuning and feature-based approaches on test data. \* corresponds to the following monolingual models: for English we use RoBERTa (Liu et al., 2019b), DBMDZ<sup>1</sup>-BERT for German, BERTje (Vries et al., 2019) for Dutch and BETO (Cañete et al., 2020) for Spanish.

of the CoNLL-03 shared task datasets. The results of this evaluation are listed in Table 5. For a meaningful comparison we also list previous state-of-the-art results. We additionally report two ablations: (1) fine-tuning with both development and test data, and (2) fine-tuning monolingual transformers.

#### 4.1 Main Results

We make the following observations:

**Fine-tuning document-level features best.** As Table 5 shows, we find that fine-tuning outperforms the feature-based approach across all experiments ( $\approx$ 2 pp on average). Similarly, we find that document-level features clearly outperform sentence-level features ( $\uparrow$ 1.15 pp on average). Similar to our results on development data, we thus find fine-tuning with document-level features to work best across all languages.

**New state-of-the-art results.** For the best approach, we also report results obtained with a training strategy in which we train over both train and development splits, anneal against training loss (instead of development F1) and perform no model selection using development data. This yields new state-of-the-art F1 scores that in most cases outperform all currently published numbers: For German (original), our approach outperforms the best previous approach (Yu et al., 2020) by  $\uparrow$ 1.81 pp, and  $\approx$ 2 pp on the revised German dataset. For Dutch and English, our approach outperforms the previous best by  $\uparrow$ 0.96 pp and  $\uparrow$ 0.46 pp respectively. For Spanish, however, our results lie slightly below those reported by (Yu et al., 2020), with a difference of  $\downarrow$ 0.16 pp.

Entity	EN	DE	DE <sub>06</sub>	NL	Es
LOC	+0.44	+0.23	+1.97	-0.74	+0.17
MISC	+0.22	-0.90	+1.66	+1.16	+0.72
ORG	+1.21	+0.56	+0.74	+1.66	+0.11
PER	+1.19	+1.15	+1.50	-0.34	+0.14

Table 6: Relative change in F1 for different entity types and languages when adding document-level context.

#### 4.2 Analysis

To get a better understanding of the impact of document-level features on different entity types, we perform a per-type analysis to compare average results across entity types with and without document context. Results are shown in Table 6.

We find that overall, the ORG (organization) and PER (person) entity types benefit the most from document-level context. In particular, the ORG type consistently improves across all languages. For other entity types, we find a mix of results depending on the language: For instance, document-level context reduces F1 score for LOC (location) and PER entities in Dutch, whereas these entities improve the most in all other languages. For German (original), document-features improve all entity types except for MISC (miscellaneous,  $\downarrow$ 0.9 pp) whereas for Spanish all entity types improve, albeit only slightly.

#### 4.3 Ablation: Monolingual Transformers

As Table 6 also shows, we surprisingly find monolingual, language-specific transformers to underperform the multilingual XLM-R for most languages: For German, Dutch and Spanish the average F1

score is lower by ↓1.22 pp, ↓1.84 pp and ↓0.98 pp respectively when compared against XLM-R. This could be due to the fact that XML-R was trained over significantly larger data sets. Only for English we find that using RoBERTa (Liu et al., 2019b) instead of XLM-R leads to a slight increase in F1-score (↑0.2 pp).

## 5 Conclusion

In this paper we evaluated document-level features in two commonly used NER architectures. For each architecture, we conducted an evaluation of different variants to determine the best setup. We derive the following recommendations for transformer-based NER:

- For *fine-tuning* approaches, we recommend a straightforward architecture without CRF and additional word embeddings with the hyperparameters as listed in Section 3.2.
- For *feature-based* approaches, we recommend an all-layer-mean with additionally stacking classic word embeddings and the hyperparameters as listed in Section 3.3.

For both architectures, we find document-level features to significantly improve overall F1 score for all four languages in the CoNLL-03 benchmark. We also find fine-tuning to outperform the feature-based approach.

We thus recommend the combination of document-level features and fine-tuning for NER. In our experiments, this yields new state-of-the-art F1 scores for English, German and Dutch and competitive numbers for Spanish. We integrate our approach into the FLAIR framework and share the recommended parameter settings to enable the research community to reproduce our results and apply our new state-of-the-art models in their tasks.

## References

- Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019a. [FLAIR: An easy-to-use framework for state-of-the-art NLP](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.
- Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019b. Pooled contextualized embeddings for named entity recognition. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, page 724–728.
- 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.
- José Cañete, Gabriel Chaperon, Rodrigo Fuentes, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *to appear in PML4DC at ICLR 2020*.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.
- 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. [Bi-directional LSTM-CRF Models for Sequence Tagging](#). *arXiv e-prints*, page arXiv:1508.01991.
- Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*.
- Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. [Linguistic knowledge and transferability of contextual representations](#). 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 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). In *International Conference on Learning Representations*.

- Jouni Luoma and Sampo Pyysalo. 2020. Exploring cross-sentence contexts for named entity recognition with bert. *arXiv preprint arXiv:2006.01563*.
- 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, Doha, Qatar. Association for Computational Linguistics.
- 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.
- Leslie N. Smith. 2018. [A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay](#). *arXiv e-prints*, page arXiv:1803.09820.
- Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2019. Portuguese named entity recognition using bert-crf. *arXiv preprint arXiv:1909.10649*.
- Jana Straková, Milan Straka, and Jan Hajic. 2019. [Neural architectures for nested NER through linearization](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy. Association for Computational Linguistics.
- Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. [What do you learn from context? probing for sentence structure in contextualized word representations](#). In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.
- 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.
- Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: Bert for finnish. *arXiv preprint arXiv:1912.07076*.
- Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. [BERTje: A Dutch BERT Model](#). *arXiv:1912.09582 [cs]*.
- Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. [CCNet: Extracting high quality monolingual datasets from web crawl data](#). In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, pages arXiv–1910.
- Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. [Named entity recognition as dependency parsing](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

## A Appendix: Figures

Figure 3 gives an overview of the feature-based approach: Word representations are extracted from the transformer by either averaging over all layers (all-layer-mean) or by concatenating the representations of the last four layers (last-four-layers). These are then input into a standard sequence labeling model as features.

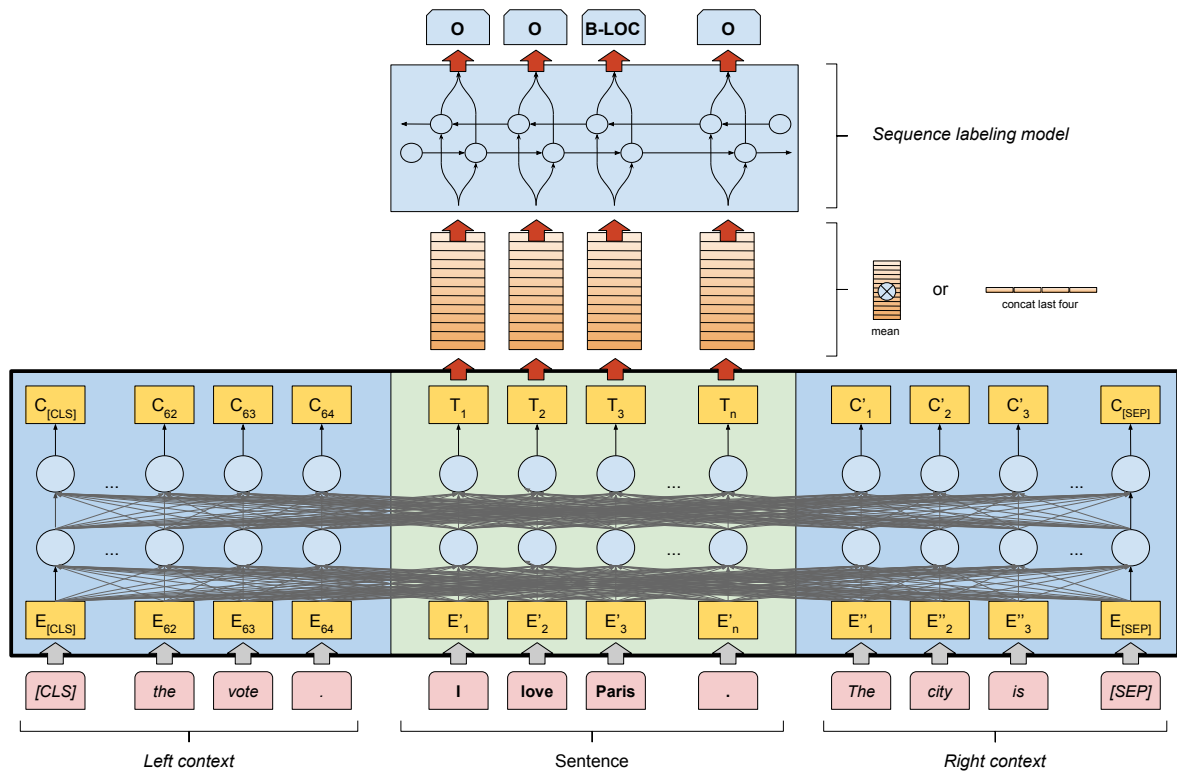


Figure 3: Overview of Feature-based approach. Self-attention is calculated over all input tokens (incl. left and right context). The final representation for each token in the sentence ("I love Paris", shaded green) can be calculated as a) mean over all layers of transformer-based model or b) concatenating the last four layers.