Pooled Contextualized Embeddings for Named Entity Recognition

Alan Akbik
Zalando Research
Mühlenstraße 25
10243 Berlin

Tanja Bergmann
Zalando Research
Mühlenstraße 25
10243 Berlin

Roland Vollgraf
Zalando Research
Mühlenstraße 25
10243 Berlin

{firstname.lastname}@zalando.de

Abstract

Contextual string embeddings are a recent type of contextualized word embedding that were shown to yield state-of-the-art results when utilized in a range of sequence labeling tasks. They are based on character-level language models which treat text as distributions over characters and are capable of generating embeddings for any string of characters within any textual context. However, such purely character-based approaches struggle to produce meaningful embeddings if a rare string is used in a underspecified context. To address this drawback, we propose a method in which we dynamically aggregate contextualized embeddings of each unique string that we encounter. We then use a pooling operation to distill a global word representation from all contextualized instances. We evaluate these *pooled contextualized embeddings* on common named entity recognition (NER) tasks such as CoNLL-03 and WNUT and show that our approach significantly improves the state-of-the-art for NER. We make all code and pre-trained models available to the research community for use and reproduction.

1 Introduction

Word embeddings are a crucial component in many NLP approaches (Mikolov et al., 2013; Pennington et al., 2014) since they capture latent semantics of words and thus allow models to better train and generalize. Recent work has moved away from the original "one word, one embedding" paradigm to investigate *contextualized* embedding models (Peters et al., 2017, 2018; Akbik et al., 2018). Such approaches produce different embeddings for the same word depending on its context and are thus capable of capturing latent contextualized semantics of ambiguous words.

Recently, Akbik et al. (2018) proposed a character-level contextualized embeddings approach they refer to as *contextual string embeddings*. They leverage pre-trained character-level language models from which they extract hidden states at the beginning and end character positions of each word to produce embeddings for any string of characters in a sentential context. They showed these embeddings to yield state-of-the-art results when utilized in sequence labeling tasks such as named entity recognition (NER) or part-of-speech (PoS) tagging.

B-PER	E-PER		S-LOC		S-ORG
Fung	Permadi	(	Taiwan	) v	Indra

Figure 1: Example sentence that provides underspecified context. This leads to an underspecified contextual word embedding for the string "*Indra*" that ultimately causes a misclassification of "*Indra*" as an organization (ORG) instead of person (PER) in a downstream NER task.

Underspecified contexts. However, such contextualized character-level models suffer from an inherent weakness when encountering rare words in an underspecified context. Consider the example text segment shown in Figure 1: "*Fung Permadi (Taiwan) v Indra*", from the English CoNLL-03 test data split (Tjong Kim Sang and De Meulder, 2003). If we consider the word "*Indra*" to be rare (meaning no prior occurrence in the corpus used to generate word embeddings), the underspecified context allows this word to be interpreted as either a person or an organization. This leads to an underspecified embedding that ultimately causes an incorrect classification of "*Indra*" as an organization in a downstream NER task.

Pooled Contextual Embeddings. In this paper, we present a simple but effective approach to address this issue. We intuit that entities are normally only used in underspecified contexts if they are expected to be known to the reader. That is, they are either more clearly introduced in an earlier sentence, or part of general in-domain knowl-

Proceedings of NAACL-HLT 2019, pages 724–728
Minneapolis, Minnesota, June 2 - June 7, 2019.

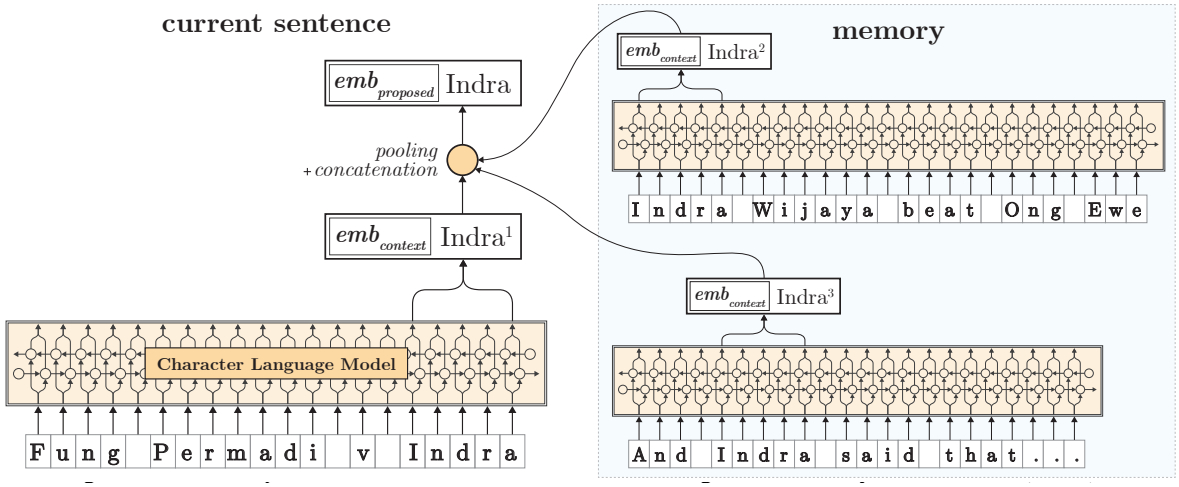

Figure 2: Example of how we generate our proposed embedding ($emb_{proposed}$) for the word "*Indra*" in the example text segment "*Fung Permadi v Indra*". We extract a contextual string embedding ($emb_{context}$) for this word and retrieve from the *memory* all embeddings that were produced for this string on previous sentences. We pool and concatenate all local contextualized embeddings to produce the final embedding.

edge a reader is expected to have. Indeed, the string "*Indra*" in the CoNLL-03 data also occurs in the earlier sentence "*Indra Wijaya (Indonesia) beat Ong Ewe Hock*".

Based on this, we propose an approach in which we dynamically aggregate contextualized embeddings of each unique string that we encounter as we process a dataset. We then use a pooling operation to distill a global word representation from all contextualized instances that we use in combination with the current contextualized representation as new word embedding. Our approach thus produces *evolving word representations* that change over time as more instances of the same word are observed in the data.

We evaluate our proposed embedding approach on the task of named entity recognition on the CoNLL-03 (English, German and Dutch) and WNUT datasets. In all cases, we find that our approach outperforms previous approaches and yields new state-of-the-art scores. We contribute our approach and all pre-trained models to the open source Flair[1] framework (Akbik et al., 2019), to ensure reproducibility of these results.

2 Method

Our proposed approach (see Figure 2) dynamically builds up a "memory" of contextualized embeddings and applies a pooling operation to distill a global contextualized embedding for each word. It requires an $embed()$ function that produces a contextualized embedding for a given $word$ in a *sentence* context (see Akbik et al. (2018)). It also requires a $memory$ that records for each unique word all previous contextual embeddings, and a $pool()$ operation to pool embedding vectors.

This is illustrated in Algorithm 1: to embed a word (in a sentential context), we first call the $embed()$ function (line 2) and add the resulting embedding to the memory for this word (line 3). We then call the pooling operation over all contextualized embeddings for this word in the memory (line 4) to compute the pooled contextualized embedding. Finally, we concatenate the original contextual embedding together with the pooled representation, to ensure that both local and global interpretations are represented (line 5). This means that the resulting *pooled contextualized embedding* has twice the dimensionality of the original embedding.

Algorithm 1 Compute pooled embedding

Input: $sentence$, $memory$

1: **for** $word$ in $sentence$ **do**
2: $emb_{context} \leftarrow$ embed($word$) within $sentence$
3: add $emb_{context}$ to $memory[word]$
4: $emb_{pooled} \leftarrow$ pool($memory[word]$)
5: $word.embedding \leftarrow$ concat(emb_{pooled}, $emb_{context}$)
6: **end for**

Pooling operations. We experiment with different pooling operations: $mean$ pooling to average a word's contextualized embedding vectors, and min and max pooling to compute a vector of all

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

Approach	CoNLL-03 En	CoNLL-03 De	CoNLL-03 Nl	WNUT-17
Pooled Contextualized Embeddings$_{min}$	**93.18** $\pm$ 0.09	**88.27** $\pm$ 0.30	90.12 $\pm$ 0.14	49.07 $\pm$ 0.31
Pooled Contextualized Embeddings$_{max}$	93.13 $\pm$ 0.09	88.05 $\pm$ 0.25	90.26 $\pm$ 0.10	49.05 $\pm$ 0.26
Pooled Contextualized Embeddings$_{mean}$	93.10 $\pm$ 0.11	87.69 $\pm$ 0.27	**90.44** $\pm$ 0.20	**49.59** $\pm$ 0.41
Contextual String Emb. (Akbik et al., 2018)	92.86 $\pm$ 0.08	87.41 $\pm$ 0.13	90.16 $\pm$ 0.26	49.49 $\pm$ 0.75
best published				
BERT (Devlin et al., 2018)†	92.8			
CVT+Multitask (Clark et al., 2018)†	92.6			
ELMo (Peters et al., 2018)†	92.22			
Stacked Multitask (Aguilar et al., 2018)†				45.55
Character-LSTM (Lample et al., 2016)†	90.94	78.76	81.74	

Table 1: Comparative evaluation of proposed approach with different pooling operations (min, max, $mean$) against current state-of-the-art approaches on four named entity recognition tasks († indicates reported numbers). The numbers indicate that our approach outperforms all other approaches on the CoNLL datasets, and matches baseline results on WNUT.

element-wise minimum or maximum values.

Training downstream models. When training downstream task models (such as for NER), we typically make many passes over the training data. As Algorithm 2 shows, we reset the memory at the beginning of each pass over the training data (line 2), so that it is build up from scratch at each epoch.

Algorithm 2 Training

```
1: for epoch in epochs do
2:     memory ← map of word to list
3:     train and evaluate as usual
4: end for
```

This approach ensures that the downstream task model learns to leverage pooled embeddings that are built up (e.g. *evolve*) over time. It also ensures that pooled embeddings during training are only computed over training data. After training, (i.e. during NER prediction), we do not reset embeddings and instead allow our approach to keep expanding the memory and evolve the embeddings.

3 Experiments

We verify our proposed approach in four named entity recognition (NER) tasks: We use the English, German and Dutch evaluation setups of the CoNLL-03 shared task (Tjong Kim Sang and De Meulder, 2003) to evaluate our approach on classic newswire data, and the WNUT-17 task on emerging entity detection (Derczynski et al., 2017) to evaluate our approach in a noisy user-generated data setting with few repeated entity mentions.

3.1 Experimental Setup

We use the open source Flair framework in all our experiments. It implements the standard BiLSTM-CRF sequence labeling architecture (Huang et al., 2015) and includes pre-trained contextual string embeddings for many languages. To Flair, we add an implementation of our proposed *pooled contextualized embeddings*.

Hyperparameters. For our experiments, we follow the training and evaluation procedure outlined in Akbik et al. (2018) and follow most hyperparameter suggestions as given by the in-depth study presented in Reimers and Gurevych (2017). That is, we use an LSTM with 256 hidden states and one layer (Hochreiter and Schmidhuber, 1997), a locked dropout value of 0.5, a word dropout of 0.05, and train using SGD with an annealing rate of 0.5 and a patience of 3. We perform model selection over the learning rate $\in \{0.01, 0.05, 0.1\}$ and mini-batch size $\in \{8, 16, 32\}$, choosing the model with the best F-measure on the validation set. Following Peters et al. (2017), we then repeat the experiment 5 times with different random seeds, and train using both train and development set, reporting both average performance and standard deviation over these runs on the test set as final performance.

Standard word embeddings. The default setup of Akbik et al. (2018) recommends contextual string embeddings to be used in combination with standard word embeddings. We use GloVe embeddings (Pennington et al., 2014) for the English tasks and FastText embeddings (Bojanowski et al., 2017) for all newswire tasks.

Baselines. Our baseline are contextual string embeddings without pooling, i.e. the original setup proposed in Akbik et al. (2018)[2]. By comparing against this baseline, we isolate the impact of our proposed pooled contextualized embeddings.

[2]Our reproduced numbers are slightly lower than we reported in Akbik et al. (2018) where we used the official CoNLL-03 evaluation script over BILOES tagged entities. This introduced errors since this script was not designed for S-tagged entities.

Approach	CoNLL-03 En	CoNLL-03 De	CoNLL-03 Nl	WNUT-17
Pooled Contextualized Embeddings (*only*)	**92.42** $\pm$ 0.07	**86.21** $\pm$ 0.07	**88.25** $\pm$ 0.11	**44.29** $\pm$ 0.59
Contextual String Embeddings (*only*)	91.81 $\pm$ 0.12	85.25 $\pm$ 0.21	86.71 $\pm$ 0.12	43.43 $\pm$ 0.93

Table 2: Ablation experiment using contextual string embeddings without word embeddings. We find a more significant impact on evaluation numbers across all datasets, illustrating the need for capturing global next to contextualized semantics.

In addition, we list the best reported numbers for the four tasks. This includes the recent BERT approach using bidirectional transformers by Devlin et al. (2018), the semi-supervised multitask learning approach by Clark et al. (2018), the ELMo word-level language modeling approach by Peters et al. (2018), and the best published numbers for WNUT-17 (Aguilar et al., 2018) and German and Dutch CoNLL-03 (Lample et al., 2016).

3.2 Results

Our experimental results are summarized in Table 1 for each of the four tasks.

New state-of-the-art scores. We find that our approach outperforms all previously published results, raising the state-of-the-art for CoNLL-03 on English to 93.18 F1-score (↑0.32 pp vs. previous best), German to 88.27 (↑0.86 pp) and Dutch to 90.44 (↑0.28 pp). The consistent improvements against the contextual string embeddings baseline indicate that our approach is generally a viable option for embedding entities in sequence labeling.

Less pronounced impact on WNUT-17. However, we also find no significant improvements on the WNUT-17 task on emerging entities. Depending on the pooling operation, we find comparable results to the baseline. This result is expected since most entities appear only few times in this dataset, giving our approach little evidence to aggregate and pool. Nevertheless, since recent work has not yet experimented with contextual embeddings on WNUT, as side result we report a new state-of-the-art of 49.59 F1 vs. the previous best reported number of 45.55 (Aguilar et al., 2018).

Pooling operations. Comparing the pooling operations discussed in Section 2, we generally find similar results. As Table 1 shows, min pooling performs best for English and German CoNLL, while $mean$ pooling is best for Dutch and WNUT.

3.3 Ablation: Character Embeddings Only

To better isolate the impact of our proposed approach, we run experiments in which we do not use any classic word embeddings, but rather rely solely on contextual string embeddings. As Table 2 shows, we observe more pronounced improvements of pooling vis-a-vis the baseline approach in this setup. This indicates that pooled contextualized embeddings capture global semantics words similar in nature to classical word embeddings.

4 Discussion and Conclusion

We presented a simple but effective approach that addresses the problem of embedding rare strings in underspecified contexts. Our experimental evaluation shows that this approach improves the state-of-the-art across named entity recognition tasks, enabling us to report new state-of-the-art scores for CoNLL-03 NER and WNUT emerging entity detection. These results indicate that our embedding approach is well suited for NER.

Evolving embeddings. Our dynamic aggregation approach means that embeddings for the same words will change over time, even when used in exactly the same contexts. Assuming that entity names are more often used in well-specified contexts, their pooled embeddings will improve as more data is processed. The embedding model thus continues to "learn" from data even after the training of the downstream NER model is complete and it is used in prediction mode. We consider this idea of constantly evolving representations a very promising research direction.

Future work. Our pooling operation makes the conceptual simplification that all previous instances of a word are equally important. However, we may find more recent mentions of a word - such as words within the same document or news cycle - to be more important for creating embeddings than mentions that belong to other documents or news cycles. Future work will therefore examine methods to learn weighted poolings of previous mentions. We will also investigate applicability of our proposed embeddings to tasks beside NER.

Public release. We contribute our code to the Flair framework[3]. This allows full reproduction of all experiments presented in this paper, and al-

[3]The proposed embedding is added to Flair in release 0.4.1. as the `PooledFlairEmbeddings` class (see Akbik et al. (2019) for more details).

lows the research community to use our embeddings for training downstream task models.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 732328 ("FashionBrain").

References

Gustavo Aguilar, Adrian Pastor Lopez Monroy, Fabio González, and Thamar Solorio. 2018. Modeling noisiness to recognize named entities using multitask neural networks on social media. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1401–1412.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics: System Demonstrations*.

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.

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.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc V. Le. 2018. Semi-supervised sequence modeling with cross-view training. In *EMNLP*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

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*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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

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. 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.

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, Vancouver, Canada. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

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 (EMNLP)*, pages 338–348, Copenhagen, Denmark.

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-Volume 4*, pages 142–147. Association for Computational Linguistics.