Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019)

Deep-EOS: General-Purpose Neural Networks for Sentence Boundary Detection

Stefan Schweter
Bayerische Staatsbibliothek München
Digital Library/Munich Digitization Center
Munich, Germany
{*stefan.schweter*}@*bsb-muenchen.de*

Sajawel Ahmed
Text Technology Lab
Goethe University Frankfurt
Frankfurt, Germany
{*sahmed*}@*em.uni-frankfurt.de*

Abstract

In this paper, we present three general-purpose neural network models for sentence boundary detection. We report on a series of experiments with long short-term memory (LSTM), bidirectional long short-term memory (BiLSTM) and convolutional neural network (CNN) for sentence boundary detection. We show that these neural networks architectures outperform the popular framework of *OpenNLP*, which is based on a maximum entropy model. Hereby, we achieve state-of-the-art results both on multi-lingual benchmarks for 12 different languages and on a *zero-shot* scenario, thus concluding that our trained models can be used for building a robust, language-independent sentence boundary detection system.

1 Introduction

The task of sentence boundary detection is to identify sentences within a text. Many natural language processing (NLP) tasks take a sentence as an input unit, such as part-of-speech tagging (Manning, 2011), dependency parsing (Yu and Vu, 2017), named entity recognition or machine translation. Thus, this foundational task stands at the beginning of various NLP processes and decisively determines their downstream-performance.

Sentence boundary detection is a nontrivial task, because of the ambiguity of the period sign ".", which has several functions (Grefenstette and Tapanainen, 1994), e.g.:

- End of sentence
- Abbreviation
- Acronyms and initialism
- Mathematical numbers

A sentence boundary detection system has to resolve the use of ambiguous punctuation characters to determine if the punctuation character is a true end-of-sentence marker[1].

In the present work, we train different deep architectures of neural networks, such as long short-term memory (LSTM), bidirectional long short-term memory (BiLSTM) and convolutional neural network (CNN), and compare the results with *OpenNLP*[2]. *OpenNLP* is a state-of-the-art tool and uses a maximum entropy model for sentence boundary detection. To test the robustness of our models, we use the *Europarl* corpus for German and English, the *SETimes* corpus for nine different Balkan languages, and the *Leipzig* corpus (Goldhahn et al., 2012) for one Semitic language, namely Arabic. This makes our model language-independent, in which further languages can be used, given the associated training resources are available.

Additionally, we use a *zero-shot* scenario to test our model on unseen abbreviations. We show that our models outperform *OpenNLP* both for each language and on the zero-shot learning task. Therefore, we conclude that our trained models can be used for building a robust, language-independent state-of-the-art sentence boundary detection system.

The remainder of the paper is organized as follows: Section 2 reviews related work. Section 3 presents a sketch of the underlying neural models and the choice of hyperparameters. Section 4 describes the text data and its preprocessing for our twofold experimental setup of a) mono-lingual, and b) zero-shot training. Section 5 reports our results, and, finally, Section 6 discusses our results and draws a conclusion.

2 Related Work

Various approaches have been employed to achieve sentence boundary detection in different languages.

[1] In this paper, we define "?!:;." as potential end-of sentence markers.

[2] *OpenNLP 1.8.4*: `https://opennlp.apache.org`

Recent research in sentence boundary detection focus on machine learning techniques, such as hidden Markov models (Mikheev, 2002), maximum entropy (Reynar and Ratnaparkhi, 1997), conditional random fields (Tomanek et al., 2007), decision tree (Wong et al., 2014) and neural networks (Palmer and Hearst, 1997). Kiss and Strunk (2006) use an unsupervised sentence detection system called *Punkt*, which does not depend on any additional resources. The system use collocation information as evidence from unannotated corpora to detect e.g. abbreviations or ordinal numbers.

The sentence boundary detection task can be treated as a classification problem. Our work is similar to the *SATZ* system, proposed by Palmer and Hearst (1997), which uses a fully-connected feed-forward neural network. The *SATZ* system disambiguates a punctuation mark given a context of k surrounding words. This is different to our approach, as we use a char-based context window instead of a word-based context window.

Further high-performers such as *Elephant* (Evang et al., 2013) or *Cutter* (Graën et al., 2018) follow a sequence labeling approach. However, they require a prior language-dependent tokenization of the input text. In contrast to these works, we construct an end-to-end approach which does not depend on the performance of any tokenization method, thus making our *Deep End-Of-Sentence detector* (Deep-EOS) more robust to multi-lingual settings.

3 Model

We use three different architectures of neural networks: long short-term memory (LSTM), bidirectional long short-term memory (BiLSTM) and convolutional neural network (CNN). All three models capture information at the character level. Our models disambiguate potential end-of-sentence markers followed by a whitespace or line break given a context of k surrounding characters. The potential end-of-sentence marker is also included in the context window. Table 1 shows an example of a sentence and its extracted contexts: left context, middle context and right context. We also include the whitespace or line break after a potential end-of-sentence marker.

LSTM We use a standard LSTM (Hochreiter and Schmidhuber, 1997; Gers et al., 2000) network with an embedding size of 128. The number of hidden states is 256. We apply dropout with probability of 0.2 after the hidden layer during training. We apply a sigmoid non-linearity before the prediction layer.

Input sentence	Left	Middle	Right
I go to Mr. Pete Tong	to Mr	.	␣Pete

Table 1: Example for input sentence and extracted context of window size 5.

BiLSTM Our bidirectional LSTM network uses an embedding size of 128 and 256 hidden states. We apply dropout with a probability of 0.2 after the hidden layer during training, and we apply a sigmoid non-linearity before the prediction layer.

CNN For the convolutional neural network we use a $1D$ convolution layer with 6 filters and a stride size of 1 (Waibel et al., 1989). The output of the convolution filter is fed through a global max pooling layer and the pooling output is concatenated to represent the context. We apply one 250-dimensional hidden layer with ReLU non-linearity before the prediction layer. We apply dropout with a probability of 0.2 during training.

Other Hyperparameters Our proposed character-based model disambiguates a punctuation mark given a context of k surrounding characters. In our experiments we found that a context size of 5 surrounding characters gives the best results. We found that it is very important to include the end-of-sentence marker in the context, as this increases the F1-score of 2%. All models are trained with averaged stochastic gradient descent with a learning rate of 0.001 and mini-batch size of 32. We use Adam for first-order gradient-based optimization. We use binary cross-entropy as loss function. We do not tune hyperparameters for each language. Instead, we tune hyperparameters for one language (English) and use them across languages. Table 2 shows the number of trainable parameters for each model.

Model	# Parameters
LSTM	420,097
BiLSTM	814,593
CNN	33,751

Table 2: Number of trainable parameters for LSTM, bidirectional LSTM and CNN.

4 Experimental Setup

Data Similar to Wong et al. (2014) we use the *Europarl* corpus (Koehn, 2005) for our experiments. The *Europarl* parallel corpus is extracted from the proceedings of the European Parliament and is originally created for the research of statistical machine translation systems. We only use German and English from *Europarl*. Wong et al. (2014) does not mention that the *Europarl* corpus is not fully sentence-segmented. The *Europarl* corpus has a one-sentence per line data format. Unfortunately, in some cases one or more sentences appear in a line. Thus, we define the *Europarl* corpus as "quasi"-sentence segmented corpus. We use the *SETimes* corpus (Tyers and Alperen, 2010) as a second corpus for our experiments. The *SETimes* corpus is based on the content published on the *SETimes.com news portal* and contains parallel texts in ten languages. Aside from English the languages contained in the *SETimes* corpus fall into several linguistic groups: Turkic (Turkish), Slavic (Bulgarian, Croatian, Macedonian and Serbian), Hellenic (Greek), Romance (Romanian) and Albanic (Albanian). The *SETimes* corpus is also a "quasi"-sentence segmented corpus. For our experiments we use all the mentioned languages except English, as we use an English corpus from *Europarl*. We do not use any additional data like abbreviation lists. We use the *Leipzig* corpus as the third and final corpus to include the non-European language Arabic into the scope of our investigations. For a *zero-shot* scenario we extracted 80 German abbreviations including their context in a sentence from Wikipedia. These abbreviations do not exist in the German *Europarl* corpus.

Preprocessing All corpora are not tokenized. Text tokenization (or, equivalently, segmentation) is highly non-trivial for many languages (Schütze, 2017). It is problematic even for English as word tokenizers are either manually designed or trained. For our proposed sentence boundary detection system we use a similar idea from Lee et al. (2017). They use a character-based approach without explicit segmentation for neural machine translation. We also use a character-based context window, so no explicit segmentation of input text is necessary.

For all corpora we use the following preprocessing steps: (a) we remove duplicate sentences, (b) we extract only sentences with ends with a potential end-of-sentence marker. Each text collection for a language is split into train, dev and test sets. Table 3 shows a detailed summary of the training, development and test sets used for each language.

Language	# Train	# Dev	# Test
German	1,476,653	184,580	184,580
English	1,474,819	184,352	184,351
Arabic	1,647,906	274,737	276,172
Bulgarian	148,919	18,615	18,614
Bosnian	97,080	12,135	12,134
Greek	159,000	19,875	19,874
Croatian	143,817	17,977	17,976
Macedonian	144,631	18,079	18,078
Romanian	148,924	18,615	18,615
Albanian	159,323	19,915	19,915
Serbian	158,507	19,813	19,812
Turkish	144,585	18,073	18,072

Table 3: Number of sentences in *Europarl*, *SETimes* and *Leipzig* corpus for each language for training, development and test set.

Tasks In the first task we train our different models on the *Europarl*, *SETimes* and *Leipzig* corpus. The second task is to perform *zero-shot* sentence boundary detection. For the *zero-shot* scenario the trained models for the German *Europarl* corpus are used.

Setup We evaluate our different models on our three corpora. We measure F1-score for each model. As baseline to our models, we use *OpenNLP*. *OpenNLP* uses a maximum entropy model. *OpenNLP* comes with pretrained models for German and English, but to ensure a fair comparison between our models and *OpenNLP*, we do not use them. Instead, we train a model from scratch for each language with the recommended hyperparameters from the documentation. For the *zero-shot* scenario we use our trained LSTM, BiLSTM and CNN models on the German *Europarl* corpus and the trained model with *OpenNLP* to perform a zero-shot sentence boundary detection on the crawled abbreviations.

5 Results

We train a maximum of 10 epochs for each model. For the German and English corpus (*Europarl*) the time per epoch is 55 minutes for the BiLSTM model, 28 minutes for the LSTM model and 5 minutes for the CNN model. For each language from the *SETimes* corpus the time per epoch is 5 minutes

Lang.	LSTM	BiLSTM	CNN	*OP*
German	**97.59**	**97.59**	97.50	97.38
English	98.61	**98.62**	98.55	98.40
Arabic	**99.86**	99.83	81.97	99.76
Bulg.	99.22	**99.27**	99.22	98.87
Bosn.	**99.58**	99.52	99.53	99.25
Greek	99.67	**99.70**	99.66	99.25
Croat.	**99.46**	99.44	99.44	99.07
Maced.	98.04	**98.09**	97.94	97.86
Roman.	99.05	99.05	**99.06**	98.89
Alban.	**99.52**	99.51	99.47	99.34
Serbian	98.72	**98.76**	98.73	98.32
Turkish	98.56	**98.58**	98.54	98.08

Table 4: Results on test set for *Europarl*, *SETimes* and *Leipzig* corpus against *OpenNLP* (OP). The highest F1-score for each task on each language is marked in bold face.

for the Bi-LSMT model, 3 minutes for the LSTM model and 20 seconds for the CNN model. Timings are performed on a server machine with a single Nvidia Tesla K20Xm and Intel Xeon E5-2630.

The results on test set on the *SETimes* corpus are shown in Table 4. For each language the best neural network model outperforms *OpenNLP*. On average, the best neural network model is 0.38% better than *OpenNLP*. The worst neural network model also outperforms *OpenNLP* for each language. On average, the worst neural network model is 0.33% better than *OpenNLP*. In half of the cases the bi-directional LSTM model is the best model. In almost all cases the CNN model performs worse than the LSTM and bi-directional LSTM model, but it still achieves better results than the *OpenNLP* model. This suggests that the CNN model still needs more hyperparameter tuning.

The first two rows in Table 4 show the results on test set on the *Europarl* corpus. For both German and English the best neural network model outperforms *OpenNLP*. The CNN model performs worse than the LSTM and bi-directional LSTM model but still achieves better results than *OpenNLP*. The bi-directional LSTM model is the best model and achieves the best results for German and English. On average, the best neural network model is 0.22% better than *OpenNLP*, whereas the worst neural network model is still 0.14% better than *OpenNLP*.

Table 5 shows the results for the *zero-shot* scenario. The CNN model outperforms *OpenNLP* by a large margin and is 6% better than *OpenNLP*. The CNN model also outperforms all other neural network models. Interestingly, the CNN model performs better in a *zero-shot* scenario than in the previous tasks (*Europarl* and *SETimes*). That suggests that the CNN model generalizes better than LSTM or BiLSTM for unseen abbreviations. The worst neural network model (LSTM model) still performs $1,6\%$ better than *OpenNLP*.

Model	Precision	Recall	F1
LSTM	56.62	96.25	71.29
BiLSTM	60.00	97.50	74.29
CNN	61.90	97.50	**75.12**
OpenNLP	54.60	96.25	69.68

Table 5: Results on the *zero-shot* scenario for unseen German abbreviations.

6 Discussion & Conclusion

In this paper, we propose a general-purpose system for sentence boundary detection using different architectures of neural networks. We use the *Europarl*, *SETimes* and *Leipzig* corpus and compare our proposed models with *OpenNLP*. We achieve state-of-the-art results.

The results on the three corpora show that the trained neural network models perform well for all languages. We tune hyperparameters just for one language (English) and share these hyperparameter settings across other languages. This suggests that the proposed neural network models can adopt other languages as well, which makes them language-independent. Our character-based context approach requires no explicit text segmentation and is robust against unknown words.

In a *zero-shot* scenario, in which no manifestation of the test abbreviations is observed during training, our system is also robust against unseen abbreviations. It shows that our proposed neural network models can detect abbreviations "on the fly", after the model has already been trained.

The fact that our proposed neural network models perform well on different languages and on a *zero-shot* scenario leads us to the conclusion that *Deep-EOS* is a *general-purpose* system[3]. Our system can be used for a wide variety of practical use cases, e.g. in the scope of the *BIOfid* project where unstructured OCR text data on biodiversity has to

[3] https://github.com/stefan-it/deep-eos

be processed for the task of biological Named Entitiy Recognition (Ahmed and Mehler, 2018; Ahmed et al., 2019).

Acknowledgments

We would like to thank the *Leibniz-Rechenzentrum der Bayerischen Akademie der Wissenschaften* (LRZ) for giving us access to the NVIDIA *DGX-1* supercomputer. Special thanks go to Prof. R. V. Zicari and Prof. A. Mehler for their constructive comments on the final manuscript, to Prof. J. Leidner for his remarks on the related work, and last but not least to Prof. U. Meyer and *Goethe Research Academy for Early Career Researchers* (GRADE) for funding the publication process of this paper.

References

Sajawel Ahmed and Alexander Mehler. 2018. Resource-Size matters: Improving Neural Named Entity Recognition with Optimized Large Corpora. In *Proceedings of the 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*.

Sajawel Ahmed, Manuel Stoeckel, Christine Driller, Adrian Pachzelt, and Alexander Mehler. 2019. BIOfid Dataset: Publishing a German Gold Standard for Named Entity Recognition in Historical Biodiversity Literature. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*. Association for Computational Linguistics. accepted.

Kilian Evang, Valerio Basile, Grzegorz Chrupała, and Johan Bos. 2013. Elephant: Sequence labeling for word and sentence segmentation. In *EMNLP 2013*.

Felix A. Gers, Jürgen A. Schmidhuber, and Fred A. Cummins. 2000. Learning to Forget: Continual Prediction with LSTM. *Neural Comput.*, 12(10):2451–2471, October.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In *LREC*.

Johannes Graën, Mara Bertamini, and Martin Volk. 2018. Cutter–a Universal Multilingual Tokenizer. In *Proceedings of the 3rd Swiss Text Analytics Conference-SwissText*, pages 75–81.

Gregory Grefenstette and Pasi Tapanainen. 1994. What is a word, What is a sentence? Problems of Tokenization. In *COMPLEX*, pages 79–87.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November.

Tibor Kiss and Jan Strunk. 2006. Unsupervised Multilingual Sentence Boundary Detection. *Comput. Linguist.*, 32(4):485–525, December.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *TACL*, 5:365–378.

Christopher D. Manning. 2011. Part-of-speech Tagging from 97% to 100%: Is It Time for Some Linguistics? In *CICLing*, pages 171–189, Berlin, Heidelberg. Springer-Verlag.

Andrei Mikheev. 2002. Periods, Capitalized Words, etc. *Comput. Linguist.*, 28(3):289–318, September.

David D. Palmer and Marti A. Hearst. 1997. Adaptive Multilingual Sentence Boundary Disambiguation. *Comput. Linguist.*, 23(2):241–267, June.

Jeffrey C. Reynar and Adwait Ratnaparkhi. 1997. A Maximum Entropy Approach to Identifying Sentence Boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, ANLC '97, pages 16–19, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hinrich Schütze. 2017. Nonsymbolic Text Representation. In *EACL*, pages 785–796.

Katrin Tomanek, Joachim Wermter, and Udo Hahn. 2007. Sentence and token splitting based on conditional random fields. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 49–57.

Francis M Tyers and Murat Serdar Alperen. 2010. South-east european times: A parallel corpus of balkan languages. In *Proceedings of the LREC Workshop on Exploitation of Multilingual Resources and Tools for Central and (South-) Eastern European Languages*, pages 49–53.

Alex Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. 1989. Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37(3):328–339.

Derek F Wong, Lidia S Chao, and Xiaodong Zeng. 2014. iSentenizer-μ: Multilingual sentence boundary detection model. *The Scientific World Journal*, 2014.

Xiang Yu and Ngoc Thang Vu. 2017. Character Composition Model with Convolutional Neural Networks for Dependency Parsing on Morphologically Rich Languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 672–678, Vancouver, Canada, July. Association for Computational Linguistics.