FullStop: Multilingual Deep Models for Punctuation Prediction

Oliver Guhr[1] **Anne-Kathrin Schumann**[2] **Frank Bahrmann**[1] **Hans-Joachim Böhme**[1]
[1]University of Applied Science (HTW) Dresden, Germany
[2]t2k GmbH, Dresden, Germany
{oliver.guhr, frank.bahrmann, hans-joachim.boehme}@htw-dresden.de
anne-kathrin.schumann@text2knowledge.de

Abstract

This paper describes our contribution to the SEPP-NLG Shared Task in multilingual sentence segmentation and punctuation prediction. The goal of this task consists in training NLP models that can predict the end of sentence (EOS) and punctuation marks on automatically generated or transcribed texts. We show that these tasks benefit from crosslingual transfer by successfully employing multilingual deep language models. Our multilingual model achieves an average F_1-score of 0.94 for EOS prediction on English, German, French, and Italian texts and an average F_1-score of 0.78 for punctuation mark prediction.

1 Introduction

The prediction of EOS and punctuation marks in automatically generated or transcribed texts is a relatively novel task. While sentence segmentation is a core, and low-level, natural language processing (NLP) task, punctuation has, in the past, primarily been studied in the context of error correction and the normalisation of automatic speech recognition (ASR) output. However, with the recent rise of conversational agents and other NLP systems that are able to generate new texts, the injection of punctuation and EOS marks has gained wider interest. This is hardly surprising because punctuation affects the readability of the text produced by the NLP system and, thus, its perceived overall performance. The SEPP-NLG Shared Task offers two subtasks, namely:

- **Subtask1 – Sentence segmentation:** Fullstop prediction on fully unpunctuated, lowercased documents.
- **Subtask 2 – Punctuation prediction:** Prediction of all punctuation marks on fully unpunctuated, lowercased documents, where the possible punctuation marks are members of the set $p = \{: -, ?.0\}$, with 0 indicating no punctuation.

The task is carried out on the German, English, French, and Italian sections of the Europarl corpus (Koehn, 2005), since it offers transcripts of spoken texts for multiple languages. We developed models for both tasks based on the Transformers library by Wolf et al. (2020). These models and our code are publicly available [1]

2 Related Work

Earlier studies on EOS and punctuation prediction reflect the various fields of application of this technology. The task is mostly modeled as token-wise prediction. Over the last few years, consistent performance improvements have – unsurprisingly – been achieved with the help of neural network approaches and large-scale neural language models.

The work by Attia et al. (2014) constitutes a rather traditional approach to spelling and punctuation correction, in this case for Arabic. The authors report that in their data set, punctuation errors constitute 40 % of all errors. The task is modeled as token-wise classification with context windows varying between 4-8 words. Classification is carried out with Support Vector Machines and Conditional Random Field (CRF) classifiers, using part-of-speech (POS) and morphological information. The authors obtain the best result, an F1-score of 0.56, with the CRF classifier and a window size of five tokens.

Che et al. (2016) experiment with three different neural network architectures, using pretrained GloVe (Pennington et al., 2014) embeddings as inputs. Since their goal is to predict punctuation marks specifically on ASR output, they evaluate

[1] https://github.com/oliverguhr/fullstop-deep-punctuation-prediction.

their models on ASR transcripts of TED talks. Predicting the positions of commas, periods, and question marks, their best result in this 4-class classification task is an F_1-score of 0.54.

Treviso et al. (2017) study sentence segmentation – not punctuation – in narrative transcripts that were generated in the context of examining patients for symptoms of language-impairing dementia. They work on three different Portuguese data sets. Input data is modeled by means of POS features, word embeddings, and prosodic information. They then combine convolutional and recurrent neural network layers, achieving F_1-scores between 0.7 and 0.8 on two evaluation data sets.

Schweter and Ahmed (2019) also experiment with the Europarl corpus, however, their task is different from the task presented here, i.e. they model only sentence segmentation by predicting, at each full stop in the input text, whether it is an EOS marker or forms a part of another linguistic unit (for instance, it could mark an abbreviation). Predictions are produced by character-level models that are fed not only the token to disambiguate, but also local contexts in the form of context windows. Working on a wide variety of languages – including often overlooked languages such as Bosnian, Greek, or Romanian, – they achieve F_1-scores between 0.98 and 0.99, with their BiLSTM model performing best on average.

Sunkara et al. (2020) also work in the clinical domain, more precisely, on the output of medical ASR systems. They jointly model punctuation and truecasing by first predicting a punctuation sequence and then the case of each input word. The authors use a pretrained transformer model (Devlin et al., 2019; Liu et al., 2019) in combination with subword embeddings to overcome lexical sparsity in the medical domain. They also carry out a fine-tuning step on medical data and a task adaptation step – randomly masking punctuation marks in the text – before training the actual model. Predicting fullstops and commas, the authors achieve F_1-scores of 0.81 (for commas) and 0.92 (for fullstops) with Bio-BERT (Lee et al., 2019), which was trained on biomedical corpora.

3 Task and Data

The task consists in predicting EOS and punctuation marks on unpunctuated lowercased text. The organizers of the SeppNLG shared task provided 470 MB of English, German, French, and Italian text. This data set consists of a training and a development set. For system ranking, a test set with in-domain and a surprise set with out-of-domain texts were used.

Figure 1 shows the distribution of the punctuation labels for subtask 2, for all languages. As can be seen from the Figure, the distribution of the labels is quite skewed, even if we disregard that the majority of tokens in each data set has the label ”0” (omitted in Figure 1 for better readability). All languages follow the same distribution pattern, however, they exhibit subtle differences. For instance, the difference in frequency between commas and fullstops is particularly pronounced for German and German, in general, has a higher proportion of commas, indicating complex sentence structures. For other language pairs, we observe slight differences in the distribution of hyphens and colons.

Earlier versions of subtask 2 also required predictions for the punctuation marks ”!” and ”;”. During the training phase, the task organizers mapped these symbols to the fullstop to account for strongly skewed distributions and potential HTML artefacts. Sentences containing other punctuation symbols than those already mentioned – parentheses, for instance – were removed by the task organizers because not all instances of parentheses were well-formed (i. e. not for every opening parenthesis there also was a closing parenthesis). These issues leave avenues for future research.

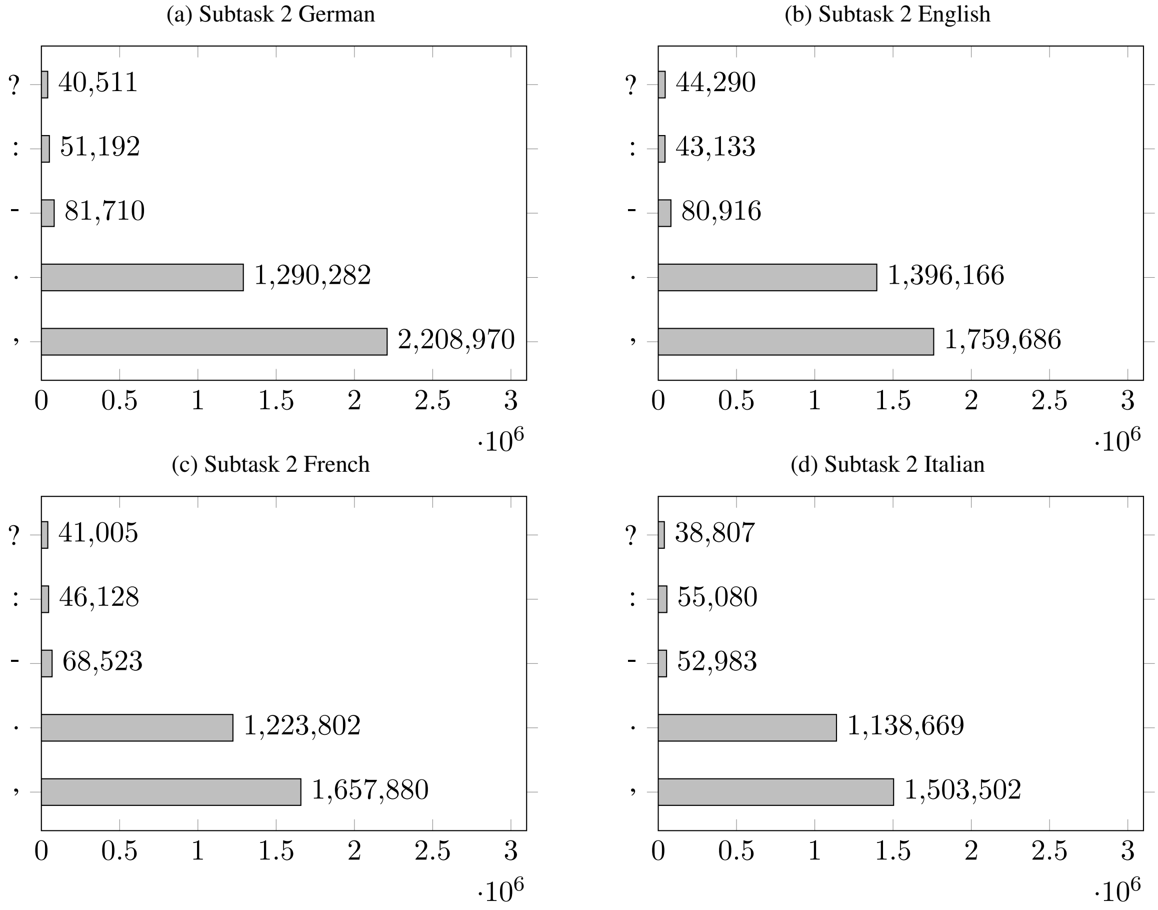
4 Models

4.1 Baselines and Model Selection

The transformer architecture (Vaswani et al., 2017) and transfer learning with transformer-based language models (Devlin et al., 2019) have led to notable performance gains for many NLP tasks. For this reason, we have focused our research on a transformer-based architecture, exploring a number of recent language models and multilingual transfer learning. Following earlier work, we have modelled the task as token-wise prediction.

However, to assess the performance gain enabled by a transformer-based language model, we also trained (for German sentence segmentation) a first, non-neural baseline: a CRF model on the basis of bag-of-words, POS and local context (+/- 2 tokens) features. This model seemed to perform much better than the spaCy[2] baseline provided for sub-

[2] https://spacy.io/.


(a) Subtask 2 German

(b) Subtask 2 English

(c) Subtask 2 French

(d) Subtask 2 Italian

Figure 1: Distribution of punctuation labels for the four languages on the training sets of task 2. Mean document length varies between 10,378 (for Italian) and 12,275 words (for French).

task 1[3], however, since it was outperformed by all transformer-based models by a large margin, we decided to not explore this direction any further.

As a second baseline, we trained a vanilla multilingual Bert model and explored techniques to improve this baseline. In particular, we focused on three different options, namely data augmentation, hyperparameter optimization, and the selection of different architectures and pre-trained models. We have also tested various preprocessing steps to remove special characters and HTML artefacts, but this had no significant effect on our results.

As a first step towards model selection, we trained a set of mono- and multilingual models on 10% of the training data for each task. We then selected the best models per language and the best multilingual model and trained them on the full training data set. This approach helped us to iterate quickly by avoiding long training times (up to 20 hours on a single GPU) just for model selection. We then selected the following architectures for our tests:

- Bert (Devlin et al., 2019)
- Distillbert (Sanh et al., 2019)
- Electra (Clark et al., 2020)
- Roberta (Liu et al., 2019)
- XLM-Roberta (Conneau et al., 2020)
- Camembert (Martin et al., 2020)

First experiments with data augmentation and hyperparameter optimization showed that these techniques had only a minor effect on the models' performance. All of our 10% and full models were trained for 3 epochs using Adafactor (Shazeer and Stern, 2018) and a learning rate of $4e^{-5}$ and batch size of 8. Furthermore we used 16-bit-precision training to improve training speed. We did run hyperparamter optimizations with limited success, for more information please see our ablations in section 7. We then focused on the selection of architectures and pretrained models.

[3] https://sites.google.com/view/sentence-segmentation/.

Base Model	Task 1 F_1	Task 2 F_1
English		
distilbert-base-uncased	0.849048	0.581294
google/electra-base-generator	0.867502	0.426554
google/electra-small-generator	0.872033	0.590815
bert-base-uncased	0.885560	0.647669
google/electra-large-generator	0.901298	0.558433
bert-large-uncased	0.903943	0.699679
roberta-base	0.921170	0.719705
xlm-roberta-large	**0.932057**	**0.740402**
roberta-large	**0.935672**	**0.742778**
German		
bert-base-multilingual-uncased	0.931668	0.708220
dbmdz/bert-base-german-uncased	0.943437	0.746249
deepset/gbert-base	0.943571	0.753979
german-nlp-group/electra-base-german-uncased	**0.950070**	**0.759387**
French		
bert-base-multilingual-uncased	0.881648	0.658968
camembert-base	0.914799	0.702187
camembert/camembert-large	**0.935436**	**0.756594**
Italian		
dbmdz/electra-base-italian-xxl-cased-generator	0.866070	0.496291
bert-base-multilingual-uncased	0.867798	0.586234
dbmdz/bert-base-italian-cased	0.897765	0.658520
dbmdz/bert-base-italian-xxl-uncased	**0.910585**	**0.693615**
multilingual		
bert-base-multilingual-uncased	0.887909	0.683688
xlm-roberta-base	0.915930	0.716822
xlm-roberta-large	**0.935946**	**0.753770**

Table 1: We trained all base models in this Table on 10% of the language-specific data or on 10% of all languages for the multilingual models. All models were trained for 3 epochs using Adafactor and a learning rate of $4e^{-5}$. For Task 1 we report the F_1 score of the EOS class. For task 2 the macro average F_1 of all classes is shown.

We trained a 10 % and 100 % model for all architecture types to ensure that the architectures scale well with the increased data. Comparing the results from Table 1 and 2, we found that the models for task 1 gain between 0.1 % to 1 % by scaling from 10% to 100% and the model for task 2 gain between 3 % to 5 %.

4.2 Windowing Approach

All selected architectures are limited with respect to the number of tokens they can process, typically 512. Since most documents are longer than this limit (see Figure 1), we needed a strategy to handle longer sequences.

The simplest method to achieve that is by splitting the text into chunks of 200 words before processing. The number of 200 words was chosen empirically to account for the fact that words get tokenized into more than one token. The disadvantage of this approach is that it is inefficient since most sequences will not utilize the full 512-token capacity of the model.

Overlapping Tokens	F_1 Score Task 1
0	0.87893
10	0.87933
100	**0.88556**
200	0.88375

Table 2: We found that an overlap of 100 tokens between consecutive sequences improves the models performance.

We therefore chose to first tokenize each document and then split it into sequences of 512 tokens. However, this approach, just like the first one, can produce sequences that start with the last word of a sentence or end with the first word of a sentence, giving the model no context for the prediction. To address this issue, we used a sliding window approach and ran experiments with different step sizes similar to the stride parameter in convolutional neural networks. This method ensures that the model has additional context for making predictions. For training, we ran a grid search to find the optimal length of the overlapping window, using an English Bert base model on 10% of the data. Based on the results shown in Table 2, we choose an overlapping window size of 100 for training our models. The loss was calculated for the whole sequence, including the overlapping part. Since this method also generates new training sequences, it also acts as a data-augmentation.

5 Results

Table 1 shows the results of the 10% model comparison training. All the models that performed best on task 1 also performed best on task 2. For English, we selected two models, XLM RoBERTa Large and RoBERTa Large since their scores were about even. An Electra-based model achieved the best results for the German language, whereas, surprisingly, English and Italian Electra models scored below baseline Bert models. For French, we selected Camembert large, a 335 million parameters RoBERTa-based model which scores notably better than Camembert base using 110 million parameters. The digital library team at the Bavarian State Library (dbmdz) published two different Italian Bert-based models, the XXL version of the model was trained on the larger corpus and achieved the best result. The multilingual XLM RoBERTa base model achieved better scores than the older multilingual Bert model using the same number of parameters. The larger 335 million parameter version of this model achieved the best multilingual model score, on par with the language-specific models. Note that the scores of the multilingual models are evaluated on a multilingual development set.

We trained the selected models on the full training set for each task and evaluated them on the development sets. The results of this evaluation can be found in Table 3 for both subtasks 1 and 2. For both tasks, the large multilingual XLM RoBERTa outperformed all language-specific models. Therefore we submitted our XLM RoBERTa based models for task 1 and 2. For the Italian language, the XLM-RoBERTa-based model scored notably better than the best language-specific model. However, for the other languages, the performance gains are not that significant. The scores of the German Electra-based model are comparable to those of XLM RoBERTa, despite using 110 million parameters in contrast to the 550 million parameters of XLM RoBERTa large. This indicates that there is room for possible performance improvements.

5.1 Final Models and Evaluation

Since the multilingual models outperformed almost all monolingual models, we selected these for subtasks 1 and 2. Furthermore, we submitted one smaller monolingual model to evaluate its performance on the test set and out-of-domain test set (surprise test).

FullStop Multilingual Task 1: This model is based on the 550-million-parameter XLM RoBERTa large model and was trained on the labeled data of task 1. Across all four languages this model archived an average F_1 score of 0.94 on the test set and an average F_1 score of 0.78 on the surprise test set.

FullStop German Task 1: This model is based on the 110-million-parameter German Electra base model. It was trained on the labeled data for task 1 and an additional data set consisting of data from speeches of the German parliament (Bundestag, 134 MB[4]) and a text crawl from the Leipzig corpora collection (245 MB[5]), containing a mixture of news texts and Wikipedia articles. For the German language, this model archived an F_1 score of 0.95 on the test set and an F_1 score of 0.80 on the surprise test set.

FullStop Multilingual Task 2: This model is also based on XLM RoBERTa large and was trained on the labeled data for task 2. As shown in Figure 2 and Table 4, the model performs well on EOS marks across all languages. In contrast, the performance for colons and hyphens is lower. We suspect that this is due to the properties of the data set as described in section 3. We have seen that hyphens and colons are not only infrequent in the training data for all languages, they also exhibit unstable

[4] https://github.com/Datenschule/offenesparlament-data

[5] https://wortschatz.uni-leipzig.de/de/download/German

Model	Test Language	F_1 Score Task 1	F_1 Score Task 2
roberta-large	EN	0.941992	0.772326
xlm-roberta-large	EN	0.938764	0.765496
electra-base-german-uncased	DE	0.953894	0.795759
electra-base-german-uncased with data augmentation	DE	0.954782	–
camembert-large	FR	0.937222	0.778617
bert-base-italian-xxl-uncased	IT	0.919729	0.732624
xlm-roberta-large	EN	**0.945746**	**0.774601**
	DE	**0.958591**	**0.813861**
	FR	**0.941974**	**0.781834**
	IT	**0.934144**	**0.761775**

Table 3: All models for subtasks 1 and 2 where trained on the full data set for each languages. For tasks 1, we report the F_1 score of the sentence end class and for task 2 the macro average F_1 score.

Label	EN	DE	FR	IT
,	0.819	0.945	0.831	0.798
-	0.425	0.435	0.431	0.421
.	0.948	0.961	0.945	0.942
0	0.991	0.997	0.992	0.989
:	0.575	0.652	0.620	0.588
?	0.890	0.893	0.871	0.832
macro avg	0.775	0.814	0.782	0.762

Table 4: Per class F_1 scores for the FullStop Multilingual Task 2 model on the dev data set.

distribution patterns across languages. Intuitively, this is not surprising as hyphens and colons, in many cases, are optional in the sense that they can be substituted by either a comma or a full stop, i. e. the rules for their usage are not only grammatical and syntactic, but also stylistic. Performance increases might be achieved through targeted training with adversarial examples. The model achieves an average F_1 of 0.78 on the test set. Similar to the other models, the performance degrades to an average F_1 of 0.61 for the out-of-domain surprise set.

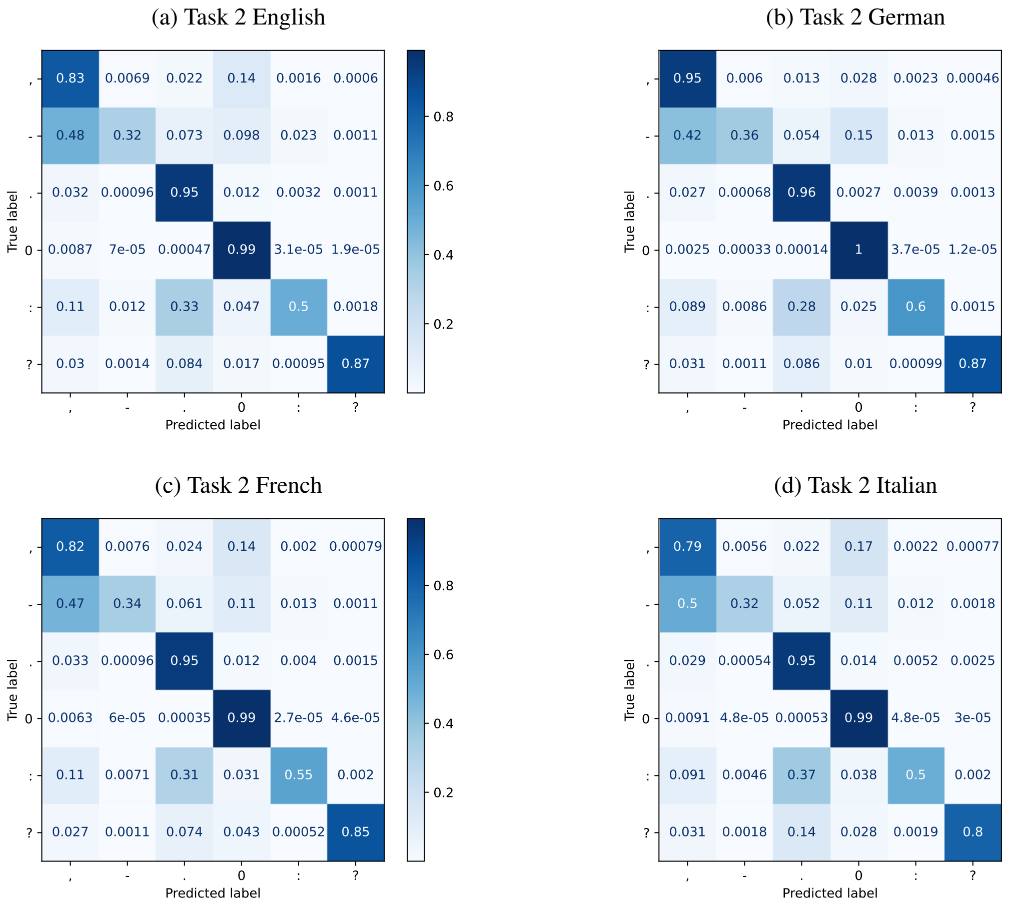
Inference on the complete test and surprise set (470 MB) takes about 1 hour for each multilingual FullStop model using an Nvidia 3090 GPU.

6 Key Findings

The type and amount of data used for pretraining has a significant impact on the final model's performance. Table 1 shows that, for Italian, there is a 5% difference for task 2 between the two monolingual Bert-based models. Both models use the same 110 million parameters of the Bert architecture, but were trained on different corpus sizes. The "bert-base-italian-uncased" model was trained on a 13GB corpus and the "bert-base-italian-xxl-uncased" model was trained on a 81 GB corpus. The positive effect of larger corpus sizes on model performance has also been verified for other transformer architectures, for instance by Conneau et al. (2020) and Clark et al. (2020).

Model architectures do not work equally well for different languages. Electra is the best-performing monolingual German model, but for English and Italian, results obtained with Electra are well behind those obtained from mono- and multilingual Bert models. We conducted a series of tests with different hyperparameters for the English Electra models, but could not further improve the results.

Both Tasks benefit from multilingual models and training data. To our surprise, the multilingual XLM-Roberta-based model outperformed all monolingual models, even though earlier multilingual Bert models were, in most cases, outperformed by their language-specific counterparts. We suspected that this could be explained by the much larger number of parameters used by XLM-Roberta large. To test this hypothesis, we trained a monolingual English model based on XLM-RoBERTa and another English model based on the monolingual RoBERTa. As shown in Table 3, both models are outperformed by the XLM-RoBERTa model, showing that the model benefits from multilinguality. Although we have no direct explanation for the superior performance of the multilingual model, we would like to accentuate that it is in line with earlier work (Muller et al., 2021) confirming (for mBERT) that the lower layers of multilingual models act as multilingual encoders by representing linguistic knowledge for various languages. If this is true here as well, the larger number of multilingual training


(a) Task 2 English

(b) Task 2 German

(c) Task 2 French

(d) Task 2 Italian

Figure 2: Confusion matrices for the XLM RoBERTa-based multilingual FullStop model for task 2. Note that all values are rounded.

examples might indeed improve performance for the punctuation task. Our successful pruning experiments also point towards this direction. However, these hypotheses need empirical validation.

Punctuation patterns are domain-specific and robust punctuation prediction requires training on diverse data sets. The data set that we trained on (Europarl) consists of data from a single domain, i. e. political speeches. As our scores on the surprise set revealed, the performance of our models degrades on texts from other domains. The performance of our task-1 model drops from 0.94 (average across all languages) on the in-domain test set to an F_1 of 0.78 on the out-of-domain surprise set. The other models participating in the shared task suffer from similar performance degradations.

7 Ablations

What are the optimal hyperparameters for each model? We ran a hyperparameter search for the Adam optimizer using the Akiba et al. (2019) framework with a budget of 200 trails on the German Electra base model. For the hyperparameter-search, we configured the following search space: learning rates between $1 \cdot 10^{-2}$ and $1 \cdot 10^{-5}$, 1 to 5 training epochs, batch sizes from 2^2 to 2^7, a weight decay from $1 \cdot 10^{-1}$ to $1 \cdot 10^{-12}$ and Adam epsilon from $1 \cdot 10^{-6}$ to $1 \cdot 10^{-10}$.

We have compared these settings with Adafactor (Shazeer and Stern, 2018), using a learning rate of $4e^5$. For both optimizers, we have trained models for task 1 and 2 on 10% of the training data. The results of this comparison are shown in table 5. Adafactor matches the performance of Adam, but eliminates the need for a time-consuming hyperparameter search, therefore we decided to use Adafactor for all models.

Is it possible to use one model for both tasks? The labels of task 2 are a super-set of the labels for task 1, therefore one can use a model trained for task 2 on task 1. We changed the classification result of task 2 by mapping the sentence end labels ”.” and ”?” to label 1 and all other labels to label

Task	Adafactor	Adam	diff in p.P.
1	0.95007	**0.95087**	-0.0008
2	**0.75939**	0.75587	+0,00352

Table 5: In a comparison between Adam with optimized hyperparameters and Adafactor, we found only minor differences in the resulting F_1 score.

0. The results in Table 6 show that this method decreases the final scores only marginally. For many applications, it is sufficient to train one model that processes all four languages for both tasks. For this shared task, we trained and submitted two different models, since a dedicated model for task 1 slightly improves the results.

Language	Task 1 Model	Task 2 Model
en	**0.945746**	0.941686
de	**0.958591**	0.955926
fr	**0.941974**	0.938254
it	**0.934144**	0.930851

Table 6: We compared the scores of the ”FullStop Multilingual Task 1” model and the remapped output of the ”FullStop Multilingual Task 2” model to match the labels of task 1. This approach leads to a slightly decreased F_1 score.

Do we need a deep model for these tasks?
For the purpose of the shared task, we did not aim at optimizing inference and training efficiency. However, we tested if it is necessary to use all the 12 Bert base layers. To this end, we trained a set of models on 10% of the English data using 3,6, 9 and 10 layers on task 1. To keep the results comparable, we used the same hyperparameters as with all other models, described in section 4. The results in Table 7 show that with this simple layer pruning approach it is possible to retain 99% of the model's performance while removing 1/4 of the last layers. We suggest to explor more advanced optimization techniques in further studies.

Layers	Parameters	F_1 Score Task 1
3	45,102,338	0.74758
6	66,365,954	0.84408
9	87,629,570	0.87776
12	108,893,186	0.88556

Table 7: F_1 scores resulting from a pruned Bert base model at various levels of pruning. Scores are for an English model trained on 10% of the data.

8 Conclusion

In this paper, we have shown that transformer-based architectures can be successfully applied to the tasks of punctuation mark and sentence end prediction. To our surprise, monolingual models are outperformed by multilingual models, showing that these models can transfer knowledge across languages. For the future, we plan to improve on two main aspects. Firstly, we want to reduce the size of our models. Both ”FullStop Multilingual” models use 550 million parameters which leads to computationally expensive inferencing. In our ablations, we have demonstrated a first approach to reducing the number of parameters. Secondly, we would like to improve the out-of-domain performance of our models. The shared task surpriseset showed that there is a performance degradation on texts from unseen domains. We will address this issue in future research.

Acknowledgments

This research has been funded by the European Social Fund (ESF), SAB grant number 100339497 and the European Re-gional Development Funds (ERDF) (ERDF-100346119). Anne-Kathrin Schumann has received funding through the SAB's technology startup scholarship (Technologiegründerstipendium).

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Mohammed Attia, Mohamed Al-Badrashiny, and Mona Diab. 2014. GWU-HASP: Hybrid Arabic Spelling and Punctuation Corrector. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 148–154. Association for Computational Linguistics.

Xiaoyin Che, Cheng Wang, Haojin Yang, and Christoph Meinel. 2016. Punctuation Prediction for Unsegmented Transcript Based on Word Vector. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 654–658. European Language Resources Association (ELRA).

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *ICLR*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stayanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451. Association for Computational Linguistics.

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)*. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the 10th Machine Translation Summit*, pages 79–86. AAMT.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A Robustly Optimized BERT Pretraining Approach. http://arxiv.org/abs/1907.11692.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Larent Romary, Éric Villemont de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. Camembert: a Tasty French Language Model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219. Association for Computational Linguistics.

Benjamin Muller, Yanai Elazar, Benoît Sagot, and Djamé Seddah. 2021. First Align, then Predict: Understanding the Cross-Lingual Ability of Multilingual BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2214–2231. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Stefan Schweter and Sajawel Ahmed. 2019. Deep-EOS: General-Purpose Neural Networks for Sentence Boundary Detection. In *Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019)*, pages 251–255.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR.

Monica Sunkara, Srikanth Ronanki, Kalpit Dixit, Sravan Bodapati, and Katrin Kirchhoff. 2020. Robust Prediction of Punctuation and Truecasing for Medical ASR. In *Proceedings of the 1st Workshop on NLP for Medical Conversations*, pages 53–62. Association for Computational Linguistics.

Marcos Vinícius Treviso, Christopher Shulby, and Sandra Maria Aluísio. 2017. Sentence Segmentation in Narrative Transcripts from Neuropsychological Tests using Recurrent Convolutional Neural Networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 315–325. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.