Bridging the Gap between Subword and Character Segmentation in Pretrained Language Models

Shun Kiyono Sho Takase Shengzhe Li Toshinori Sato
LINE Corporation
shun.kiyono@linecorp.com, sho.takase@linecorp.com,
shengzhe.li@linecorp.com, toshinori.sato@linecorp.com

Abstract

Pretrained language models require the use of consistent segmentation (e.g., subword- or character-level segmentation) in pretraining and finetuning. In NLP, many tasks are modeled by subword-level segmentation better than by character-level segmentation. However, because of their format, several tasks require the use of character-level segmentation. Thus, in order to tackle both types of NLP tasks, language models must be independently pretrained for both subword and character-level segmentation. However, this is an inefficient and costly procedure. Instead, this paper proposes a method for training a language model with unified segmentation. This means that the trained model can be finetuned on both subword- and character-level segmentation. The principle of the method is to apply the subword regularization technique to generate a mixture of subword- and character-level segmentation. Through experiment on BERT models, we demonstrate that our method can halve the computational cost of pretraining.

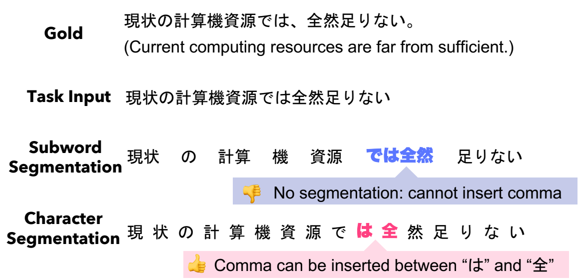


Figure 1: Overview of punctuation restoration. Character-level segmentation must be used to insert a missing comma in a given input sentence.

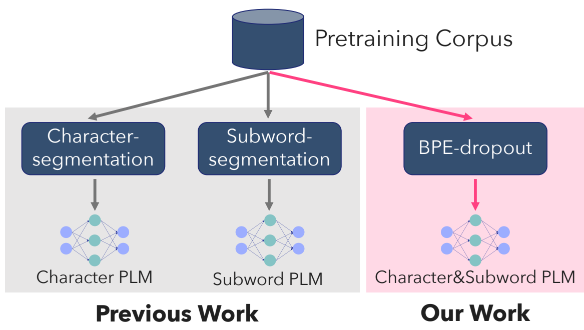


Figure 2: Overview of our method. Previously, subword- and character-level pretraining were conducted independently (left). Conversely, in our method, BPE-dropout enables the training of the language model with unified segmentation (right).

1 Introduction

The use of large pretrained language models (PLMs) has become the dominant approach for tackling NLP tasks and applications (Devlin et al., 2019; Bommasani et al., 2021; Kaneko et al., 2020; Konno et al., 2021). One notable characteristic of these models is that the segmentation algorithm must be determined before pretraining the model. Given a pretrained model, users are expected to employ a consistent segmentation algorithm.

For example, a common convention is to use a family of subword-level segmentation algorithms (Sennrich et al., 2016; Kudo, 2018; Song et al., 2021) with a sufficiently large vocabulary; for example, 8k (Kiyono et al., 2019), 30k (Devlin et al., 2019), 50k (Radford et al., 2019), or 250k (Scao et al., 2022). The subword-level segmentation is usually preferred over the character-level segmentation, because subword models often outperform character models (Libovický et al., 2022) and are more computationally efficient (Xue et al., 2022).

However, such predetermined subword-level segmentation may cause a *segmentation incompatibility problem*, depending on the target downstream task. More specifically, this problem occurs when the pretrained model uses subword-level segmentation but the target task requires a character-level

Proceedings of Recent Advances in Natural Language Processing, pages 568–577
Varna, Sep 4–6, 2023
https://doi.org/10.26615/978-954-452-092-2_062

segmentation. A typical example of a character-level task is punctuation restoration for Japanese text. Punctuation restoration is a post-processing module that is applied to the output of an automatic speech recognition system to improve the readability of transcripts (Tilk and Alumäe, 2016). We present an overview of punctuation restoration in Figure 1. Figure 1 shows that, because the positions of punctuation marks do not necessarily correspond to the positions of subword-level segmentations, character-level segmentation must be employed to tackle this task. In addition, there are several other Japanese tasks, including spelling error correction and text normalization, that also require the character-level segmentation.

A naive way to solve the segmentation incompatibility problem is to independently pretrain language models for both subword- and character-level segmentations[1]. In fact, this is a common practice in current Japanese language models. For example, both subword-level BERT[2] and character-level BERT[3] models are distributed and actively used in the NLP community. Our organization has also been following this practice for constructing in-house BERT models. Specifically, we regularly pretrain both subword- and character-level language models from scratch, on the latest Web corpus, to keep them updated with news information. However, pretraining is an extremely computationally intensive process that requires very large GPU clusters (Strubell et al., 2019). This fact encouraged us to develop a means of training a single language model with *unified segmentation* (i.e., a model that can handle both subword and character-level segmentations) and thereby eliminate the need for independent pretraining on each type of segmentation.

To achieve the goal of unified segmentation, we use the subword regularization technique (Kudo, 2018; Provilkov et al., 2020) during the pretraining (Figure 2). Subword regularization trains the model with multiple segmentation candidates to improve the model's robustness and generalization. Instead, in this paper, we use it as a means of simultaneously incorporating subword- and character-level segmentation into the pretraining. Our method is extremely simple and it requires no additional model parameters.

In our experiments, we demonstrate the effectiveness of our method on the pretraining of BERT (Devlin et al., 2019), which is one of the most popular PLMs. Our experimental results indicate that the BERT model with unified segmentation performs on par with models that are pretrained only on subword- or character-level segmentation, and therefore the computational cost of pretraining can be halved.

2 Background

As explained in Section 1, our method is based on a subword segmentation algorithm and a corresponding regularization technique, namely, subword regularization (Kudo, 2018). In this paper, we employ byte pair encoding (BPE) (Sennrich et al., 2016) and BPE-dropout (Provilkov et al., 2020) for subword segmentation and subword regularization, respectively[4]. This section briefly describes the main ideas underlying both methods.

2.1 Byte Pair Encoding (BPE)

Byte Pair Encoding (BPE) (Sennrich et al., 2016) is an algorithm for obtaining subword-level segmentations of a given token.

BPE uses a table of merge rules to define the segmentation procedure (Figure 3, left). Here, each merge rule represents how two consecutive tokens should be concatenated to form a longer subword. In addition, each merge rule has a priority: a merge rule that appears earlier in the table has a higher priority than the later rules. To obtain the merge rules, BPE counts the frequencies of all consecutive token pairs of a given corpus, and the token pair with the highest frequency is iteratively appended at the very end of the merge rules. The construction of the merge rules ends when the number of merge rules reaches a predefined size, which is a hyperparameter.

Segmentation of a given token proceeds by iteratively applying the set of merge rules in a deterministic manner (Figure 3, right). First, a token is rep-

[1] Technically, it is possible to finetune a subword-level pretrained model on a character-level segmentation. However, as we demonstrate using experimental results (Section 4.3), the performance of such an approach is suboptimal compared with the character-level finetuning of a character-level pretrained model.

[2] `https://huggingface.co/cl-tohoku/bert-base-japanese-v2`

[3] `https://huggingface.co/cl-tohoku/bert-base-japanese-char-v2`

[4] Our method does not depend on BPE. That is, another subword segmentation algorithm (e.g., BERT-WordPiece (Devlin et al., 2019; Song et al., 2021) or the unigram language model (Kudo, 2018)) may be used as an alternative. Details are discussed in Section 6.

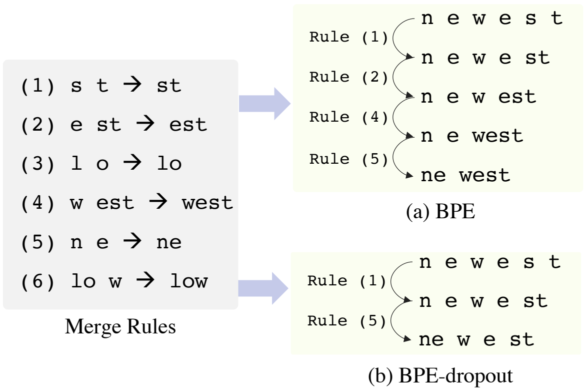


Figure 3: Example of BPE-based segmentation. A token `newest` is first represented as a sequence of characters. In (b) BPE-dropout, some merge rules are randomly dropped with a probability of p. As a result, its final segmentation `ne w e st` differs from that of (a) vanilla BPE, `ne west`.

resented as a sequence of characters. Second, two adjacent tokens are iteratively merged according to the merge rules and their corresponding priority. For example, in Figure 3, merge rule (1) has the highest priority; therefore, this rule is applied at the beginning of the process. These merge operations are repeated until no applicable merge rules are available.

2.2 Subword Regularization for BPE

Subword regularization (Kudo, 2018) is a technique for improving a model's robustness to noise. To achieve this, this technique incorporates multiple segmentations of a given token into the training. BPE-dropout (Provilkov et al., 2020) is a subword regularization technique developed for BPE, which enables BPE to obtain multiple segmentations from a given token. The original BPE and BPE-dropout are compared in Figure 3.

BPE-dropout randomly discards each merge rule with a probability of p. Thus, for a given token, the segmentation results may be different for each merge process. A higher value of p corresponds to a more aggressive dropout. For example, BPE-dropout with $p = 1.0$ discards the entire set of merge rules, and the result is equivalent to character-level segmentation. Conversely, if $p = 0.0$, BPE-dropout is identical to the original BPE, that is, segmentation is deterministic.

3 Method

Originally, BPE-dropout was developed for the purpose of regularization, that is, to improve a model's robustness to noise and segmentation errors. Conversely, in this study, we used this technique as a means of training a language model that is compatible with both subword- and character-level segmentations. Our idea originated from the characteristics of the segmentation performed by BPE-dropout (Figure 3 (b)), that is, a sequence of two subwords `ne west` can be segmented as a sequence of both characters and subwords `ne w e st`. We expect that a model trained with such a mixed segmentation can be compatible with both subword- and character-level segmentation. As a result, the need for independently pretraining language models for dedicated types of segmentations can be eliminated, and thus, the computational cost of pretraining can be halved.

Our method is extremely straightforward: during pretraining, we simply apply the off-the-shelf BPE-dropout algorithm to the input. Thus, the method requires neither modification of the model architecture nor the addition of model parameters. Once the model is pretrained, we set the dropout probability p according to the desirable segmentation, and then perform finetuning. For example, if a task of interest requires character-level segmentation, we set $p = 1.0$ and then finetune the model.

4 Experiments

We demonstrate the effectiveness of unified segmentation on pretrained BERT (Devlin et al., 2019) models on Japanese benchmark datasets. Specifically, we demonstrate that unified segmentation achieves performance comparable to that of both subword- and character-level BERT. It should be noted that the aim of unified segmentation is neither to achieve state-of-the-art performance on benchmark datasets, nor to outperform its counterparts (i.e., BERT models pretrained on either subword- or character-level segmentation alone). Instead, we aim to achieve comparable performance. This is because, given such results, the independent training of subword- and character-level BERT models can be eliminated, thereby saving the computational cost of pretraining.

4.1 Experimental Configuration

4.1.1 Pretraining Dataset

We pretrained the BERT-base model (Devlin et al., 2019) on the Japanese Wikipedia corpus[5]. We

[5] We used a dump data as of October 2020.

first tokenized the corpus using the MeCab tokenizer[6] with UniDic dictionary v2.1.2. We then performed subword tokenization using the BPE algorithm with the SentencePiece toolkit (Kudo and Richardson, 2018). We set the vocabulary size and character coverage ratio to 32,000 and 0.9995, respectively.

4.1.2 Finetuning Dataset

Subword Task: JGLUE To evaluate performance in subword-level segmentation, we used the public JGLUE dataset (Kurihara et al., 2022), which is a Japanese version of the widely-used GLUE benchmark (Wang et al., 2018). We used this dataset in order to compare the unified BERT model with its counterparts, namely, character-level BERT and subword-level BERT. We report the scores for three tasks: natural language inference (JNLI), sentiment analysis (MARC-ja), and semantic textual similarity (JSTS). Because the original JGLUE does not include an official test set, we randomly split the official validation set into two sets, which we use as a validation set and a test set.

Character Task: Punctuation Restoration We also conducted an experiment on the Japanese punctuation restoration task, which restores missing commas and periods in a given text. This task requires the character-level segmentation of the input text. We constructed the benchmark dataset from the Japanese raw corpus as follows. First, we randomly sampled 100k sentences from the Japanese portion of the CC-100 corpus (Wenzek et al., 2020; Conneau et al., 2020). Second, we removed Japanese commas and periods from the corpus. Third, we assigned a label for each character, namely, no action, comma insertion, or period insertion. Finally, we concatenated consecutive sentences into a single sequence; each sequence contains at most three sentences. For a given pretrained BERT model, we formulated this task as a sequential labeling task, as described in Devlin et al. (2019). Specifically, we fed the BERT model's final hidden layer output to a linear classifier to predict the label.

4.1.3 Models

We compared the following three segmentation settings.

Pretraining	
Architecture	BERT-base
Implementation	Megatron-LM (Shoeybi et al., 2019)
Optimizer	Adam
Learning Rate Schedule	Linear warmup and decay
Warmup Steps	12,500
Max Learning Rate	5e-4
Initial Learning Rate	1e-07
Dropout	0.1
Gradient Clipping	1.0
Weight Decay	0.01
Mini-batch Size	2,048
Number of Updates	250,000
Max Sequence Length	512
Vocabulary Size	32,000
BPE-dropout rate (p)	0.1
Finetuning	
Optimizer	Adam
Learning Rate Schedule	Linear warmup and decay
Warmup Steps	5% of total gradient steps
Max Learning Rate	2e-5
Dropout	0.1
Gradient Clipping	1.0
Weight Decay	0.01
Mini-batch Size	32
Number of Epochs	10

Table 1: List of hyperparameters for pretraining and finetuning.

- SUBWORD: An input text is deterministically segmented into subwords, i.e., we set $p = 0.0$.
- CHARACTER: An input text is deterministically segmented into characters, i.e., we set $p = 1.0$.
- BPE-DROPOUT: An input text is stochastically segmented using BPE-dropout.

The hyperparameters are listed in Table 1. We used the Megatron-LM implementation (Shoeybi et al., 2019) for the pretraining . The choice of hyperparameters (e.g., large batch size and high learning rate, etc) mostly follows recommendations made in reports of previous studies (Liu et al., 2019; Shoeybi et al., 2019; Mosbach et al., 2021; Zhang et al., 2021).

4.2 Results in Subword Task: JGLUE

Table 2 shows the results on the JGLUE dataset. The comparison of models (c) and (a) demonstrates that the performance of SUBWORD derived from BPE-DROPOUT (c) achieved performance comparable with that of the SUBWORD-only model (a), especially on the test set. In addition, with respect to character-level segmentation, the CHARACTER

[6] https://taku910.github.io/mecab/

Model ID	Pretraining	Finetuning	JNLI Valid	JNLI Test	MARC-ja Valid	MARC-ja Test	JSTS Valid	JSTS Test
(a)	SUBWORD	SUBWORD	88.55	89.43	95.74	95.19	85.09	87.71
(b)	CHARACTER	CHARACTER	85.54	86.91	94.65	95.08	82.97	84.75
(c)†	BPE-DROPOUT	SUBWORD	88.00	88.69	95.54	95.26	84.52	87.64
(d)†	BPE-DROPOUT	CHARACTER	87.37	88.93	95.21	95.39	82.91	86.26
(e)	SUBWORD	CHARACTER	86.50	87.78	94.38	94.69	80.04	82.36

Table 2: Performance in JGLUE tasks. We report the accuracy for JNLI and MARC-ja. We report the Spearman's rank correlation coefficient ρ for JSTS. All values are averages of three different random seeds. † indicates our method.

Model ID	Pretraining	Finetuning	Valid	Test
(b)	CHARACTER	CHARACTER	80.86	81.13
(d)†	BPE-DROPOUT	CHARACTER	81.88	82.06
(e)	SUBWORD	CHARACTER	78.49	78.98

Table 3: Performance in the punctuation restoration task. We report the micro-F_1 score. All values are averages of three different random seeds. † indicates our method.

finetuning of BPE-DROPOUT (d) outperformed the CHARACTER-only model (b). These results demonstrate that, with BPE-DROPOUT pretraining, we can effectively train a model with unified segmentation. It is worth noting that a naive CHARACTER finetuning of a SUBWORD model was ineffective; this is because the model (e) consistently underperformed our model (d). That is, a pretraining involving character-level segmentation is crucial for CHARACTER finetuning to achieve high performance.

4.3 Results in Character Task: Punctuation Restoration

Table 3 shows the results on punctuation restoration task. Similarly to the results on Table 2, CHARACTER finetuning of the BPE-DROPOUT model (d) outperformed the pure CHARACTER model (b), thereby demonstrating the effectiveness of our method. We also conducted an experiment with CHARACTER finetuning of the SUBWORD model (e). However, model (e) consistently underperformed the other two models. Given the effectiveness of BPE-DROPOUT in both the subword task (Section 4.2) and the character task (Section 4.3), we believe that BPE-DROPOUT can be used as a drop-in replacement for the conventional independent pretraining of the SUBWORD and CHARACTER models.

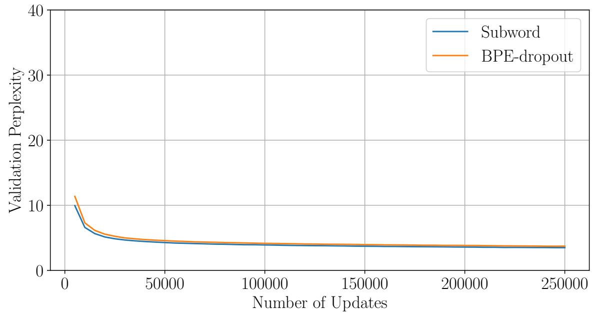


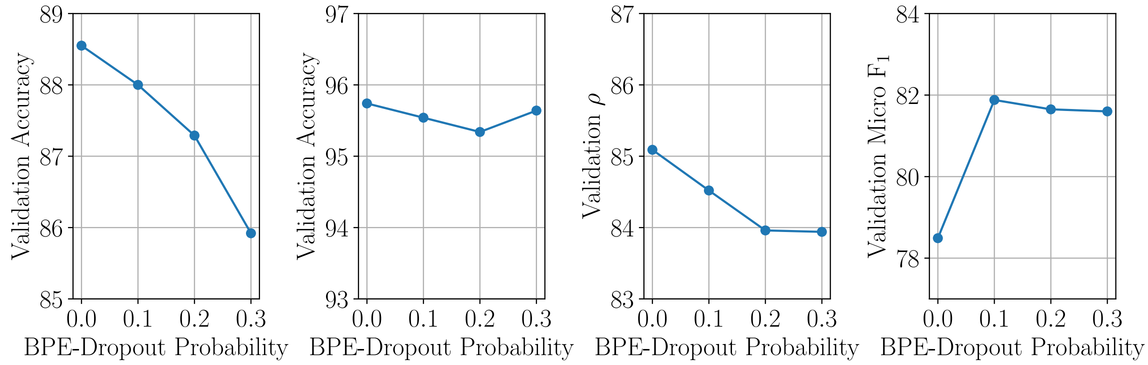
Figure 4: Comparison of validation perplexity curves of subword and BPE-dropout models during BERT pretraining. Both methods converged at a similar rate.

5 Analysis

Does BPE-dropout Require Longer Pretraining Time? As explained in Section 2.2, BPE-dropout belongs to a family of *regularization* techniques. A potential drawback of BPE-dropout is that, when pretraining a model with it, it may take longer for the model to converge. In the worst case, BPE-dropout has no practical advantages over independent training of subword and character models, with respect to computational cost. To verify this, we plotted a validation perplexity curve, as shown in Figure 4. The figure demonstrates that the speed of convergence is indeed the same for both the subword and BPE-dropout models.

Effectiveness of BPE-dropout Probability In the main experiment (Section 4), we set the BPE-dropout probability p to 0.1, following the previous study (Provilkov et al., 2020). Here, we investigated the effectiveness of changing the BPE-dropout probability p for the BERT pretraining. Specifically, we report the performance of SUBWORD finetuning in subword tasks and CHARACTER finetuning in a character task (punctuation restoration).

Figure 5a-5c demonstrate that a higher dropout probability consistently reduced subword-level per-



(a) JNLI (b) MARC-ja (c) JSTS (d) Punctuation Restoration

Figure 5: Effectiveness of changing BPE-dropout probability p for pretraining. Note that $p = 0.0$ is equivalent to BPE pretraining (i.e., SUBWORD).

formance. When the dropout probability was high, BPE-dropout almost always segmented the subword tokens into smaller units. This may have caused an insufficient pretraining with subword tokens that consist of many characters, leading to performance degradation of SUBWORD finetuning. Conversely, for a character task (Figure 5d), a small dropout probability (0.1) could already significantly improve the performance over the SUBWORD pretraining. These results support our choice of dropout probability $p = 0.1$ in the main experiment.

6 Related Work

6.1 Subword Regularization

Subword regularization (Kudo, 2018) is a technique for improving the model's robustness to corpus noise and segmentation errors. The underlying idea is to virtually augment the given training data by generating multiple segmentation candidates. Specifically, Kudo (2018) developed a subword algorithm based on a unigram language model, and performed sampling-based segmentation. In contrast to the subword regularization of Kudo (2018), which samples subwords according to the likelihood of a given sequence, Hiraoka et al. (2022) proposed a method of re-sampling subwords according to the length of each subword, to construct a more robust model. Moreover, Takase et al. (2022) indicated that using multiple segmentations improves the performance during inference.

Originally, subword regularization was only available for the subword algorithm based on unigram language model. Recently, several recent follow-up studies have made the technique applicable for other algorithms. For example, Provilkov et al. (2020) proposed BPE-dropout for BPE. Similarly, Hiraoka (2022) proposed MaxMatch-dropout for BERT-WordPiece (Devlin et al., 2019; Song et al., 2021)[7].

In this study, we employed BPE to develop a model with unified segmentation. This is because BPE is the most popular subword algorithm in the NLP literature. Because of the simplicity of our method, it is technically applicable to other subword algorithms; the only requirement is that the algorithm has a corresponding subword regularization method. However, such an exploration is outside the scope of this paper.

6.2 Segmentation for Pretrained Language Model

Currently, the use of subword segmentation is a *de facto* standard for PLMs (Mielke et al., 2021). However, the use of subword algorithms, which determine the segmentation according to frequency, poses several problems. First, these algorithms do not take lexical or semantic information into account. As a result, the segmentation aligns poorly with morphology, and this misalignment causes suboptimal performance in downstream tasks (Bostrom and Durrett, 2020). Second, imbalanced vocabulary allocation occurs when multilingual subword models are constructed (Rust et al., 2021; Scao et al., 2022).

To solve above problems, several studies have proposed the use of character-level segmentation for PLMs. Character BERT (El Boukkouri et al.,

[7]We use the name BERT-WordPiece to refer to the algorithm that uses a greedy longest-match strategy for segmentation, to distinguish it from the original WordPiece algorithm, which is a variant of BPE (Schuster and Nakajima, 2012; Wu et al., 2016).

2020) replaces the word embedding layer with a character convolutional layer to construct an open-vocabulary model. ByT5 (Xue et al., 2022) uses byte-level sequences to eliminate the tokenization procedure. In contrast to these approaches, our method enables the model to be trained with unified segmentation, that is, the model can use both character- and subword-level segmentations.

Some studies (Hiraoka et al., 2020, 2021) have proposed methods to modify segmentations according to their performance in downstream tasks. Because these methods can be combined with any pretrained model, we can use these methods with our proposed model to further improve the performance.

6.3 Efficient Pretraining of Language Models

Several previous studies have focused on improving the training efficiency of language models (Izsak et al., 2021; Geiping and Goldstein, 2022). For example, Izsak et al. (2021) proposed a recipe for training a BERT model within 24 hours, namely, 24h BERT. 24h BERT applies insightful techniques, including an efficient implementation and the use of a larger model for faster convergence. Levine et al. (2021) proposed a sophisticated masking strategy for BERT, which is based on pointwise mutual information (PMI-Masking). PMI-Masking enables faster BERT training than the conventional random masking strategy. These studies are all orthogonal to our study, that is, their findings can be combined with our method to further reduce the computational cost.

7 Conclusion

In this study, we investigated the effectiveness of incorporating subword regularization as a means of training a language model with unified segmentation. Our method enables the pretraining of a single model that is applicable to both subword- and character-level segmentation. This can significantly reduce the computational cost of pretraining. As a future work, we will investigate the effectiveness of this method to the pretraining of other language models, such as the encoder-decoder model (Raffel et al., 2020) and decoder-only model (Radford et al., 2019).

Acknowledgments

We thank anonymous reviewers for their insightful comments.

References

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren E. Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas F. Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, O. Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir P. Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Benjamin Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, J. F. Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Robert Reich, Hongyu Ren, Frieda Rong, Yusuf H. Roohani, Camilo Ruiz, Jack Ryan, Christopher R'e, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishna Parasuram Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei A. Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2021. On the opportunities and risks of foundation models. *arXiv*.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. 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)*, pages

4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Jun'ichi Tsujii. 2020. CharacterBERT: Reconciling ELMo and BERT for word-level open-vocabulary representations from characters. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6903–6915, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jonas Geiping and Tom Goldstein. 2022. Cramming: Training a language model on a single GPU in one day. *arXiv preprint arXiv:2212.14034*.

Tatsuya Hiraoka. 2022. MaxMatch-dropout: Subword regularization for WordPiece. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4864–4872, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Tatsuya Hiraoka, Sho Takase, Kei Uchiumi, Atsushi Keyaki, and Naoaki Okazaki. 2020. Optimizing word segmentation for downstream task. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1341–1351. Association for Computational Linguistics.

Tatsuya Hiraoka, Sho Takase, Kei Uchiumi, Atsushi Keyaki, and Naoaki Okazaki. 2021. Joint optimization of tokenization and downstream model. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 244–255. Association for Computational Linguistics.

Tatsuya Hiraoka, Sho Takase, Kei Uchiumi, Atsushi Keyaki, and Naoaki Okazaki. 2022. Word-level perturbation considering word length and compositional subwords. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3268–3275, Dublin, Ireland. Association for Computational Linguistics.

Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. How to train BERT with an academic budget. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10644–10652, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4248–4254, Online. Association for Computational Linguistics.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1236–1242, Hong Kong, China. Association for Computational Linguistics.

Ryuto Konno, Shun Kiyono, Yuichiroh Matsubayashi, Hiroki Ouchi, and Kentaro Inui. 2021. Pseudo zero pronoun resolution improves zero anaphora resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3790–3806, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. 2022. JGLUE: Japanese general language understanding evaluation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2957–2966, Marseille, France. European Language Resources Association.

Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. 2021. PMI-Masking: Principled masking of correlated spans. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Jindřich Libovický, Helmut Schmid, and Alexander Fraser. 2022. Why don't people use character-level machine translation? In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2470–2485, Dublin, Ireland. 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. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sabrina J Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y Lee, Benoît Sagot, et al. 2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. *arXiv preprint arXiv:2112.10508*.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. BLOOM: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Xinying Song, Alex Salcianu, Yang Song, Dave Dopson, and Denny Zhou. 2021. Fast WordPiece tokenization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2089–2103, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Sho Takase, Tatsuya Hiraoka, and Naoaki Okazaki. 2022. Single model ensemble for subword regularized models in low-resource machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2536–2541, Dublin, Ireland. Association for Computational Linguistics.

Ottokar Tilk and Tanel Alumäe. 2016. Bidirectional recurrent neural network with attention mechanism for punctuation restoration. In *Interspeech*, pages 3047–3051.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

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 Twelfth Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021. Revisiting few-sample BERT fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Model ID	Pretraining	Finetuning	JNLI Valid	JNLI Test	MARC-ja Valid	MARC-ja Test	JSTS Valid	JSTS Test
(a)	Subword	Subword	88.55	89.43	95.74	95.19	85.09	87.71
(b)	Character	Character	85.54	86.91	94.65	95.08	82.97	84.75
(c)	BPE-Dropout	Subword	88.00	88.69	95.54	95.26	84.52	87.64
(d)	BPE-Dropout	Character	87.37	88.93	95.21	95.39	82.91	86.26
(e)	RandomMix	Subword	87.92	88.66	95.64	95.38	84.54	86.86
(f)	RandomMix	Character	87.98	88.58	95.19	95.30	82.77	85.74

Table 4: Performance in JGLUE tasks. We report the accuracy for JNLI and MARC-ja. We report the Spearman's rank correlation coefficient ρ for JSTS. All values are average of three different random seeds.

A Appendix

A.1 Alternative Approach for Unified Segmentation Model

Background In this paper, we used BPE-dropout for training BERT with unified segmentation. The goal was to simultaneously incorporate subword- and character-level segmentation into pretraining. There exists an alternative approach to achieve this goal: instead of BPE-dropout, we can randomly mix the subword-level segmentation with character-level segmentation in the training data. We refer to this approach as RandomMix.

A comparison of subword-level segmentation, character-level segmentation, BPE-dropout, and RandomMix is presented in Figure 6. The difference between RandomMix and BPE-dropout is that BPE-dropout generates a mixture of character and subword within a sequence, whereas RandomMix always segments a given sequence into characters or subwords. Here, we compare RandomMix with BPE-dropout.

Result We pretrained a BERT model using RandomMix (RandomMix) and evaluated its performance on JGLUE benchmark. For RandomMix, we mixed subword-level segmentation and character-level segmentation in a 1:1 ratio. The experimental setup for pretraining and finetuning was identical to that described in Section 4.

Table 4 presents the results. The table shows that the RandomMix models (e) and (f) achieved almost comparable performance to the BPE-Dropout models (c) and (d) in the JNLI and MARC-ja tasks. However, in the JSTS task, the RandomMix model slightly underperformed BPE-Dropout. Given this result, we decided to use BPE-Dropout instead of RandomMix.

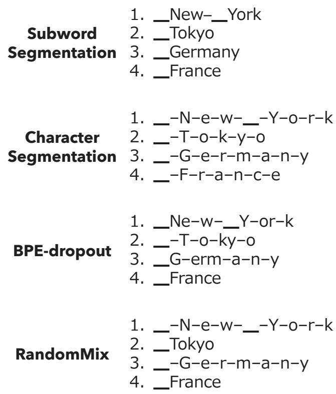


Figure 6: Comparison of four segmentation methods. A dash "–" represents a segmentation boundary. In RandomMix, a given text is always represented as either a subword-level segmentation or a character-level segmentation.