Impact of Tokenization on Language Models: An Analysis for Turkish

Anonymous ACL submission

Abstract

Tokenization is an important text preprocessing step to prepare input tokens for language models. WordPiece and BPE are de-facto methods employed by large language models, such as BERT and GPT. However, the impact of tokenization can be different for the agglutinative languages having words with prefixes and suffixes, such as Turkic languages. We compare five tokenization methods, including a morphological-level tokenization that takes agglutinative language structure into account. We train tokenizers, and pre-train mini language models using RoBERTa pre-training procedure on Turkish OSCAR corpus. We then fine-tune our models on six downstream tasks. There are two main outcomes: (i) Morphological and word-level tokenizers outperform de-facto tokenizers in particular cases. (ii) Mini models can be competitive to larger state-of-the-art models, such that a 14-times smaller model can recover 94% of the performance of a larger model.

1 Introduction

Tokenization is an important text preprocessing step for deep language models. Input text is split into smaller pieces so that out-of-vocabulary words can still be processed by language models. Moreover, language models can benefit from sub-word tokens to better comprehend text semantics.

Transformer-based language models generally employ two de-facto tokenization algorithms, namely WordPiece (Schuster and Nakajima, 2012) and Byte Pair Encoding (BPE) (Sennrich et al., 2016). BERT (Devlin et al., 2019) uses WordPiece, whereas GPT-2 (Radford et al., 2019) uses BPE. There are other efforts for tokenization, such as SentencePiece (Kudo and Richardson, 2018) to fix input text without space between words.

Large language models are first pre-trained for English; successor pre-trained models in low-resource languages thereby employ the same tokenizers. However, the impact of tokenization can be different for agglutinative languages, such as Turkic and Uralic languages, where words can have prefixes and suffixes. For instance, in Turkish, parsing the word "veremedim" (translated as "I could not give") results in "ver-e-me-di-m" including four suffixes in a single word. A morphological-level tokenizer can output five tokens in this case, providing model with a better understanding of word semantics. An example benefit is that language model would relate that the suffix "-me" provides negation, similar to the word "not" in English.

In this study, we compare the performance of different tokenization methods for Turkish. We select five tokenizers such that their outputs vary from smallest pieces (characters) to whole words. These tokenization methods are character-level, BPE, WordPiece, morphological-level, and word-level. In order to evaluate the performance of the tokenizers, we train a tokenizer for each method, and pre-train small language models using RoBERTa pre-training procedure, called **RoBERTa-TR-mini**, on Turkish OSCAR corpus. We then fine-tune our models on six downstream tasks; namely Text Classification, Sentiment Analysis, Named Entity Recognition, Question Answering, Semantic Text Similarity, and Natural Language Inference.

Our main contributions are two-fold. First, we compare the impact of tokenizers for Turkish language models. We find that morphological and word-level tokenizers outperform de-facto tokenizers (BPE and WordPiece) in some cases. Second, we compare our mini models with a large state-of-the-art one similar to BERT-base, and show that a 14-times smaller model can recover 94% of the performance of the larger one.

2 Related Work

The prevalent tokenization algorithms in the literature, Byte Pair Encoding (BPE) (Sennrich et al., 2016) and WordPiece (Schuster and Nakajima, 2012), are of recent interest in language model pre-training research. BPE is found to be sub-optimal for language pre-training (Bostrom and Durrett, 2020) as it does not effectively utilize the vocabulary space. Nayak et al. (2020) compare the activations of attention layers of BERT with WordPiece and word-level tokenization to assess the effect of including subword tokens. They find out that the vocabulary with frequency-based character combinations hinders the ability of modeling semantically meaningful relations between words.

Alternative tokenization algorithms using morphological analysis are promising candidates for subword tokenization that increase modeling efficiency and downstream performance (Park et al., 2020; Vasiu and Potolea, 2020). Joint and hybrid tokenization approaches combine coarse and fine-grained representations to incorporate word-level and subword representations (Hiraoka et al., 2021; Zhang et al., 2021b).

Effects of SentencePiece, word-level, and syllable-level tokenization strategies are investigated for low-resource languages, such as Thai (Lowphansirikul et al., 2021). Morphological analysis is used to propose a tokenization system (Ahmadi, 2020) for Kurdish. Exploiting pre-trained models with parameter freezing and additional intermediate layers is beneficial for Uyghur-Chinese machine translation (Zhang et al., 2021a). Although there are some efforts for Turkish pre-training[1], such as BERTurk (Schweter, 2020), the effect of tokenization algorithms including a morphological-level one is yet to be studied. To the best of our knowledge, this is the first study that investigates the impact of tokenization on Turkish.

3 Impact of Tokenization

We develop a pipeline that consists of choosing a tokenization method, pre-training a language model by using the selected tokenizer, and then fine-tuning the model on different downstream tasks to evaluate the performance of the tokenizer.

3.1 Tokenization Methods

- **Character-level**: Unlike the tokenization methods performing on word or sub-word units, byte or character level models split words into the smallest parts. They can be utilized in any language. Since character-level tokenizer requires no training to learn a vocabulary, we employ the ByT5 tokenization (Xue et al., 2021).
- **BPE**: Byte Pair Encoding (BPE) is a frequently used method for pre-trained language models (Sennrich et al., 2016). In this method, all unique words are first extracted. Then, a base vocabulary is constituted from all symbols occurring in the unique words. The final vocabulary is built by merging the symbols according to the frequencies of consecutive symbols or sub-words.
- **WordPiece**: Similar to BPE, WordPiece is also based on merging characters in the documents (Schuster and Nakajima, 2012). Main difference from BPE is that, WordPiece merges symbols towards maximizing the language model likelihood, i.e., when the probability of the merged symbol divided by individual probabilities of the symbols is greater than any other symbol pair.
- **Morphological-level**: Since Turkish is an agglutinative language, morphological analysis can provide suffixes and word stems that are semantically more meaningful and valuable than the tokens obtained with overlapping frequency or likelihood. Therefore, we propose to use the parsing output (without tags) of morphological analysis as input tokens. We use Zemberek morphological analysis tool (Akın and Akın, 2007) before training the tokenizer.
- **Word-level**: This is a basic method that splits text with spaces between words, i.e. considers whole words as tokens. One explicit disadvantage is that this model requires more vocabulary size compared to other methods. We therefore set vocabulary size of this model higher than others.

3.2 Pre-train: RoBERTa-TR-mini

The OSCAR Turkish deduplicated corpus[2] constitutes the main pre-training data of our model (Ortiz Suárez et al., 2019). We filter out 95,152 instances that are not in Turkish with an automated language detector[3]. The tokenization process is conducted in three steps: Applying normalization, training the tokenizer (except char-level), mapping the tokenizer to obtain tokenized data. We apply lowercase conversion and NFC normalization[4]. We train BPE and WordPiece with vocabulary size of 50k, and word-level and morph-level with vocabulary size of 100k to decrease unknown (out-of-vocabulary) tokens due to conjugations in agglutinative languages.

[1] https://github.com/Loodos/turkish-language-models

[2] https://huggingface.co/datasets/oscar

[3] https://pypi.org/project/langdetect

[4] Unicode normalization is important for Turkish, since there are special characters (ç, ğ, ı, ö, ş, ü) in the Turkish alphabet that are not observed in English. We note that NFC Unicode normalization provides all letters in Turkish.

	BERTurk-base	RoBERTa-TR-mini
Parameters	110.62 M	7.79 M
Train data	35 GB	27 GB
Layers	12	4
Heads	12	4
Hidden size	768	256
Batch size	n/a	264
Max length	512 tokens	514 tokens
Train time	9.63 days	1.04 days*
Hardware	TPU v3-8	2x Nvidia RTX2080 Ti

Table 1: **Pre-training configurations**. (*) Train time and hardware are given for BPE and WordPiece. Time can differ for other tokenizers, e.g. morph-level tokenizer outputs more tokens, and its train time is 1.58d.

We pre-train a language model using Turkish (TR) text, using RoBERTa pre-training procedure and configuration, but smaller in terms of layers, attention heads, and hidden size (similar to BERT-mini (Devlin et al., 2019)). We thereby call the model as *RoBERTa-TR-mini*.

The pre-training details of our mini model is given in Table 1. We compare the results of our model with the current state-of-the-art performance for sanity check, i.e. the rationality of our results. To do so, we employ the BERTurk model (Schweter, 2020), which is a Turkish pre-trained version of BERT-base. Since we examine the effect of different tokenization strategies in Turkish, we keep the pre-training procedure computationally simpler because extensive pre-training might overshadow possible advantages of tokenization algorithms. When a model is extensively pre-trained, the performance can converge to high scores, even with character-level encoding (Xue et al., 2021).

3.3 Fine-tuning Tasks

We evaluate the performance of our models by fine-tuning six downstream tasks.

- **Text Classification (TC)**: We use a Turkish news classification dataset (Toraman et al., 2011) that has approximately 7.5k news articles over eight news categories, such as economy and sports.
- **Sentiment Analysis (SA)**: The task is binary classification of text sequences as positive or negative. We use a Turkish dataset including movie reviews (Demirtas and Pechenizkiy, 2013).
- **Named Entity Recognition (NER)**: We use a Turkish dataset including news articles with named entity tags (Tür et al., 2003). For morph-level tokenization, ground truth labels are reorganized according to new tokens after morphological analysis.

		TC	SA	NER	QA	STS	NLI
mini	Epochs	10	10	10	50	25	3
	Length	514	514	514	514	514	514
	BS	32	32	32	32	32	32
BERT	Epochs	3	3	3	3	25	3
	Length	256	256	256	256	256	256
	BS	32	32	32	16	32	16
	LR	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5
	Size	7.5k	10.7k	23.2k	1.2k	8.6k	569.0k

Table 2: **Fine-tuning configurations**. *mini* refers to RoBERTa-TR-mini, and *BERT* refers to BERTurk. We modify configurations for BERTurk due to its space complexity. *BS* refers to Batch Size, *LR* to Learning Rate, *Length* to max sequence length, *Size* to the number of instances in the dataset. We apply constant learning rate for all tasks, except linear decay learning rate in NLI. For char-level models, max length is set to 1024, and batch size to 16.

- **Question Answering (QA)**: Given a context information or passage, the task is to find the correct part of the context representing the answer. Text span is extracted by predicting where the answer starts and ends in the passage. We use the Turkish split of the XQuaD dataset (Artetxe et al., 2020).
- **Semantic Text Similarity (STS)**: In this task, semantic similarity of two text sequences are measured. Sentences are annotated from 1 to 5 indicating their similarity degree. Different from classification tasks, this problem is handled as a regression problem. We use a Turkish STS dataset (Beken Fikri et al., 2021).
- **Natural Language Inference (NLI)**: Given two sentences, the task is to predict the semantic relation of the latter to the former, in terms of *entailment, neutral, contradiction*. We use a Turkish NLI dataset (Budur et al., 2020).

Note that we select two tasks (TC and SA) for single sequence classification, two tasks (NER and QA) for token classification, and two tasks (STS and NLI) for semantic similarity.

4 Experiments

4.1 Experimental Setup

For fine-tuning our models, the configurations along with dataset sizes are given in Table 2. For pre-training, we use AdamW optimizer (β_1 is 0.90, β_2 is 0.98, ϵ is 1e-6), linear scheduling with warmup ratio of 1e-2 and peak learning rate of 5e-5, and gradient accumulation with 22 steps. Other hyperparameters are set to the RoBERTa configuration.

		TC			SA			NER			QA			STS		NLI		
		P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**corr**	**p-value**	**P**	**R**	**F1**
	BERT	0.918	0.917	0.917	0.927	0.927	0.927	0.926	0.941	0.933	0.582	0.666	0.484	0.862	<1e-178	0.852	0.852	0.852
R-TR-mini	Char	0.501	0.539	0.513	0.640	0.637	0.636	0.350	0.380	0.362	0.155	0.531	0.150	0.196	<1e-1	0.468	0.469	0.467
	BPE	0.851	0.851	0.846	0.869	0.869	0.869	0.338	0.194	0.242	0.128	0.399	0.111	**0.310**	<1e-8	**0.701**	**0.702**	**0.701**
	WP	0.852	0.850	0.843	**0.872**	**0.872**	**0.872**	0.602	0.689	0.641	0.035	0.184	0.037	0.262	<1e-3	0.656	0.656	0.656
	Morph	0.818	0.815	0.800	0.825	0.825	0.825	**0.643**	**0.739**	**0.687**	**0.177**	**0.617**	**0.176**	0.246	<1e-1	0.610	0.610	0.610
	Word	**0.858**	**0.856**	**0.850**	0.862	0.862	0.862	0.638	0.707	0.670	0.044	0.247	0.048	0.263	<1e-6	0.643	0.643	0.642

Table 3: **Fine-tuning results on six NLP tasks using Turkish datasets**. The average of 10-fold cross validation is reported in terms of precision (P), recall (R), and weighted F1. For STS, Pearson correlation (corr) is reported with p-value. *R-TR-mini* refers to our pre-trained model for Turkish text, *RoBERTa-TR-mini*, along with each tokenization method. *Char* refers to character-level tokenizer, *BPE* refers to Byte Pair Encoding, *WP* refers to WordPiece, *Morph* refers to morphological-level tokenizer, *Word* refers to world-level tokenizer. Highest score among tokenizers is given as bold. *BERT* refers to BERTurk, which is structurally similar to BERT-base, but pre-trained for Turkish text.

We measure weighted precision, recall, and F1 score for all tasks, except STS where Pearson correlation is reported with p-value. We apply 10-fold cross-validation and report the average scores.

4.2 Experimental Results

We report the fine-tuning results in Table 3. There are two main aspects in this experiment. First, we compare the performance of tokenizers (rows) using RoBERTa-TR-mini for Turkish downstream tasks (columns). Second, we analyze the performance of our mini model, compared to a larger state-of-the-art model. To do so, we report the performance of BERTurk, a Turkish model with the similar size of BERT-base, in the first row.

Characters are not for our mini models. Character-level tokenization achieves the worst performance for Turkish in most tasks. We argue that our mini models are inadequate to comprehend the relations among characters, which could be better modeled by larger language models (Xue et al., 2021).

Word-level tokenizer performs better with less unknown tokens. Word-level tokenization provides a head start to the model by exploiting word semantics. This high-level modeling can benefit sequence classification, rather than token classification. Indeed, word-level tokenizer outperforms others in Text Classification (TC). However, this observation is not valid for another sequence classification task, Sentiment Analysis (SA). The reason would be that the ratio of unknown tokens is approximately 5% for TC, and 15% for SA. Word-level tokenization would perform better as the number of unknown tokens decreases.

Morph-level tokenizer is better for token classification. When tokenizers are compared among each other, we observe that morphological-level tokenizer outperforms others in Named Entity Recognition and Question Answering. We argue that suffixes provide useful information for such tasks that employ token classification in Turkish.

De-facto tokenizers are better for semantic similarity. BPE works better than others in Semantic Text Similarity and Natural Language Inference. We observe that the sub-words that BPE outputs work better than others for such semantic tasks in Turkish.

Mini models can be competitive to larger ones. We expect that the performance of our mini models is worse than larger models, i.e. BERTurk, due to the computational advantages of larger models. However, we find that the performance gap is narrow for particular tasks. Our 14-times smaller model recovers 93% of BERTurk's performance in TC, 94% in SA, and 83% in NLI.

5 Conclusion

We analyze the impact of five tokenization algorithms on language models for Turkish. The results are interesting such that word-level and morph-level tokenizers outperform de-facto tokenizers in particular tasks, showing that agglutinative languages can benefit from such tokenizers. Moreover, our mini language models are competitive to a larger state-of-the-art model in particular tasks, showing a trade-off between size and performance.

In future work, we plan to extend our experiments to other agglutinative languages, such as Finnish and Hungarian, and other tokenizers such as SentencePiece (Kudo and Richardson, 2018). Morphological disambiguation (Hakkani-Tür et al., 2018) can be used to improve the quality of morphological analysis. We also plan to compare our results with those of larger models trained with the same tokenization methods.

References

Sina Ahmadi. 2020. A tokenization system for the Kurdish language. In *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 114–127, Spain. ICCL.

Ahmet Afsin Akın and Mehmet Dündar Akın. 2007. Zemberek, an open source NLP framework for Turkic languages. *Structure*, 10:1–5.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the ACL*, pages 4623–4637. ACL.

Figen Beken Fikri, Kemal Oflazer, and Berrin Yanikoglu. 2021. Semantic similarity based evaluation for abstractive news summarization. In *Proceedings of GEM 2021*, pages 24–33, Online. ACL.

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

Emrah Budur, Rıza Özçelik, and Tunga Güngör. 2020. Data and representation for Turkish natural language inference. In *Proceedings of the 2020 Conf. on EMNLP*, pages 8253–8267, Online. ACL.

Erkin Demirtas and Mykola Pechenizkiy. 2013. Cross-lingual polarity detection with machine translation. In *Proceedings of WISDOM 2013, Chicago, IL, USA, August 11, 2013*, pages 9:1–9:8. ACM.

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 ACL: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. ACL.

Dilek Zeynep Hakkani-Tür, Murat Saraçlar, Gökhan Tür, Kemal Oflazer, and Deniz Yuret. 2018. *Morphological Disambiguation for Turkish*, pages 53–67. Springer Int. Publishing, Cham.

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

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 Conf. on EMNLP: System Demonstrations*, pages 66–71.

Lalita Lowphansirikul, Charin Polpanumas, Nawat Jantrakulchai, and Sarana Nutanong. 2021. Wangchanberta: Pretraining transformer-based Thai language models. *arXiv preprint arXiv:2101.09635*.

Anmol Nayak, Hariprasad Timmapathini, Karthikeyan Ponnalagu, and Vijendran Gopalan Venkoparao. 2020. Domain adaptation challenges of BERT in tokenization and sub-word representations of out-of-vocabulary words. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 1–5, Online. ACL.

Pedro Javier Ortiz Suárez, Benoit Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. Proceedings of CMLC. Cardiff, 2019, pages 9 – 16, Mannheim.

Kyubyong Park, Joohong Lee, Seongbo Jang, and Dawoon Jung. 2020. An empirical study of tokenization strategies for various Korean NLP tasks. In *Proceedings of the 1st Conf. of the Asia-Pacific Chapter of the ACL and the 10th IJCNLP*, pages 133–142, Suzhou, China. ACL.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Stefan Schweter. 2020. BERTurk - BERT models for Turkish.

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 ACL (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. ACL.

Cagri Toraman, Fazli Can, and Seyit Koçberber. 2011. Developing a text categorization template for Turkish news portals. In *Int. Sym. on Innovations in Intel. Sys. and Applications, INISTA 2011*, pages 379–383.

Gökhan Tür, Dilek Hakkani-Tür, and Kemal Oflazer. 2003. A statistical information extraction system for Turkish. *Natural Lang. Engineering*, 9(2):181–210.

Mihaela Alexandra Vasiu and Rodica Potolea. 2020. Enhancing tokenization by embedding romanian language specific morphology. In *Proceedings of ICCP 2020*, pages 243–250. IEEE.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, et al. 2021. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *arXiv preprint arXiv:2105.13626*.

Wenbo Zhang, Xiao Li, Yating Yang, and Rui Dong. 2021a. Pre-training on mixed data for low-resource neural machine translation. *Information*, 12(3):133.

Xinsong Zhang, Pengshuai Li, and Hang Li. 2021b. AMBERT: A pre-trained language model with multi-grained tokenization. In *Findings of the ACL: ACL-IJCNLP 2021*, pages 421–435, Online. ACL.