Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 2440–2452
Marseille, 11–16 May 2020

Wiki-40B: Multilingual Language Model Dataset

Mandy Guo, Zihang Dai, Denny Vrandečić, Rami Al-Rfou
Google Research
1600 Amphitheatre Parkway, Mountain View CA
{xyguo, zihangd, vrandecic, rmyeid}@google.com

Abstract

We propose a new multilingual language model benchmark that is composed of 40+ languages spanning several scripts and linguistic families. With around 40 billion characters, we hope this new resource will accelerate the research of multilingual modeling. We train monolingual causal language models using a state-of-the-art model (Transformer-XL) establishing baselines for many languages. We also introduce the task of multilingual causal language modeling where we train our model on the combined text of 40+ languages from Wikipedia with different vocabulary sizes and evaluate on the languages individually. We released the cleaned-up text of 40+ Wikipedia language editions, the corresponding trained monolingual language models, and several multilingual language models with different fixed vocabulary sizes.

Keywords: Language Modeling, Wikipedia, Multilinguality, Low Resource Languages

1. Introduction

Language modeling has received a significant attention for its role as a benchmark to test new network architectures (Vaswani et al., 2017; Dai et al., 2019; Al-Rfou et al., 2019) and learning representations for down-stream tasks (Peters et al., 2018; Devlin et al., 2018; Radford et al., 2018; Yang et al., 2019). Causal language models are a category of language models that aim to predict the next token given the previously observed tokens. They have been deployed in a wide variety of practical applications such as machine translation, automatic speech recognition, spelling correction and writing assistants (Kannan et al., 2016; Henderson et al., 2017; Chen et al., 2019).

Language model evaluation metrics such as perplexity and bits per character are intrinsic in nature. They enable researchers to use this task to evaluate different networks and training algorithms robustly. Extrinsic evaluation, on the other hand, relies on down-stream tasks which are usually small or quite limited in scope, if they exist at all for low resource languages. Causal language model evaluation datasets can be scaled easily to avoid robustness issues that come with small down-stream evaluation tasks.

Despite all these benefits, most of the released datasets with benchmarked results are limited to English (Prasad et al., 2008; Mahoney, 2009; Chelba et al., 2013). This limits the ability of researchers to understand many aspects of language modeling that are related to open vocabulary and complex morphology given that compiling a small vocabulary for English can already achieve a high coverage rate (Baayen, 1996; Kageura, 2012).

It is uncommon for researchers to study causal language modeling within a multilingual corpus that contains a mixture of languages. However, with the rising interests in multilingual research, multilingual language modeling could be of great interest to researchers in the fields of "Multi-Domain" or "Multi-Task" modeling. We hope that this novel setup will pose new challenges for researchers. We aim to transfer knowledge learned from high resource languages to low resource languages. This requires to construct optimal vocabularies and develop new model architectures. At the same time we aim to minimize the interference from the low resource languages on the performance of the high resource languages.

To summarize our main contributions, we are:

- Releasing high quality processed Wikipedia text in 40+ languages (listed in Table 2) split into train, dev, and test sets.
- Releasing pre-trained monolingual causal language models using transformer-XL network for each language, establishing the first baselines for many languages.
- Releasing pre-trained multilingual causal language models for 40+ languages in Wikipedia using SentencePiece (SPM) (Kudo and Richardson, 2018) with different vocabulary sizes.

2. Wikipedia Corpus

We choose Wikipedia as our benchmark dataset for its permissive licensing, availability in many languages, and wide coverage of topics. Each Wikipedia content is organized into pages and its text formatted using special markup within each page (called Wikitext). To maximize the utility of this data for language modeling, we construct a preprocessing pipeline to remove non-content pages and Wikitext, keeping only few structural markers, such as article and section boundaries. In the following subsections, we outline our process in detail.

2.1. Page Filtering

Many Wikipedia pages are non-content pages and do not hold significant amounts of text. We aim to keep the pages that represents articles covering topics and entities. We define the following set of rules to remove those non-article pages:

- **Disambiguation pages:**[1] These pages are used to resolve conflicts in article titles that occur when a single

[1] https://en.wikipedia.org/wiki/Wikipedia:Disambiguation

TF Dataset cleanup	Olindo Guerrini (14 October 1845 - 21 October 1916) was an Italian poet who also published under the pseudonyms Lorenzo Stecchetti and Argìa Sbolenfi. He was born at Forlì, but grew up in Sant'Alberto, Ravenna, and after studying law took to a life of letters, ... References External links Guerrini bio (in Italian) The Song of Hate (Il Canto dell'Odio) Category:Italian poets
Our Dataset	Olindo Guerrini (14 October 1845 - 21 October 1916) was an Italian poet who also published under the pseudonyms Lorenzo Stecchetti and Argìa Sbolenfi. He was born at Forlì, but grew up in Sant'Alberto, Ravenna, and after studying law took to a life of letters, ...
TF Dataset cleanup	- - - was a town located in Echi District, Shiga Prefecture, Japan. "Aitō" means "eastern Echi".
Our Dataset	Aitō was a town located in Echi District, Shiga Prefecture, Japan. "Aitō" means "eastern Echi".

Table 1: Comparison between our final processed text and the one produced by the TensorFlow Wikipedia Dataset. The "Olindo Guerrini" article shows that our processed corpus removes sections that correspond to lists leaving the more relevant content to our task. The "Aitō" article shows we succeed in extracting the full sentence from the markup while the TensorFlow dataset omits the town name.

term is associated with more than one topic. For example, the page at `Joker`[2] is a disambiguation page, leading to all the alternative usages of `Joker`, such as a playing card, a comic book character, or a song. The page content is often just a list of named entities that have similar page titles. Given the content of these pages does not resemble natural language, we decide not to include them.

- **Redirect pages:**[3] These pages do not hold any content, their mere functionality is to automatically send a query using synonyms to their canonical page. For example, if you search for `UK` in Wikipedia, it will take you to the page `United Kingdom` with a note saying `Redirected from UK`.[4] We filter them out to avoid including duplicate pages.

- **Deleted pages**: These pages are not accessible any more by readers while they might still be available in the Wikipedia dump.

- **Non-Entity Pages**: We utilize Wikidata (Vrandečić and Krötzsch, 2014) to identify which pages correspond to entities. We found this heuristic to be an effective way to identify content-heavy pages of high quality. Non-entity pages tend to be full of lists, infoboxes, and images. For example, `List of Dutch-language films`.[5]

2.2. Page Processing

Wikipedia page content is stored using a markup language.[6] This markup defines text styling features as well as functional templates that enhance the reader experience. These templates could be nested, outdated, and ill-defined which makes processing them quite a complex task.

We wanted to use an existing solution to clean-up the raw Wikipedia text instead of creating a new one. The available libraries to clean up Wikipedia text (Zesch et al., 2008; Milne and Witten, 2013) do not provide downloadable dumps of the results, so we were left with two options: TensorFlow Datasets or Google's internal cleaned-up version of Wikipedia's text.

Initially, we set to use Wikipedia processed dumps as released and maintained by TensorFlow Datasets.[7] However we quickly noticed that TensorFlow relies on the `mwparserfromhell`[8] library which does not remove `References` and `External Links` that produce many short phrases instead of full sentences. Moreover, other issues include omitting text displayed within templates which leads to broken sentences.

In Table 1, we show how TensorFlow Dataset removes certain template invocations deleting portions of the text by mistake. For example, the TensorFlow Dataset would mistakenly remove the entity name at the beginning of a sentence because it was surrounded by a `nihongo` Wikipedia template, a template that indicates the pronunciation of a Japanese word.

These flaws produce text that is less than ideal and does not resemble natural usage of language. Thus, we use Google's internal markup cleaning and annotation process to clean up the Wikipedia source text. We publish the full cleaned-up text so that the results can be compared.

We follow these steps to process the pages content:

- **Non-Content Sections**: We remove sections such as `See also`, `References`, etc. These sections are lists of external links or articles related to the page content.

- **Structured Objects**: We remove images, captions, tables, and lists. These sections often contain short and incomplete sentences.

[2] https://en.wikipedia.org/wiki/Joker

[3] https://en.wikipedia.org/wiki/Wikipedia:Redirect

[4] https://en.wikipedia.org/wiki/UK

[5] https://en.wikipedia.org/wiki/List_of_Dutch-language_films

[6] https://en.wikipedia.org/wiki/Help:Wikitext

[7] https://www.tensorflow.org/datasets/catalog/wikipedia

[8] https://github.com/earwig/mwparserfromhell

```
_START_ARTICLE_
2012 Premiership Rugby Sevens Series
_START_SECTION_
Final stage
_START_PARAGRAPH_
The finals were played at The
Recreation Ground, Bath on Friday
3 August 2012._NEWLINE_ For the
finals, the 6 qualified teams were
split into two pools of three teams.
Scoring remained the same as in the
previous rounds (4 points for a win,
etc.), and the winner of each pool
progressed to the final.
```

Figure 1: Partial `2012 Premiership Rugby Sevens Series` Wikipedia article displaying our structural markers.

2.3. Structural Markers

We keep four boundary markers in our cleaned up text for two reasons: First, self-attention based models should learn not to rely on information across articles since their order is random in our corpus. Second, these special markers can act as text generation controls when we sample our model later on.
We define four special markers:

- `_START_ARTICLE_`: Each article will start with this special token followed by the page title. Generating this token while sampling the model signals that the previous article has ended.
- `_START_SECTION_`: Each article has several sections (e.g. History, Life Events, etc.). This token signifies the end of the previous section and the start of the new section title.
- `_START_PARAGRAPH_`: This separator divides the section title and the paragraph text within the section.
- `_NEWLINE_`: When available, this marker signifies the end of the current paragraph.

Figure 1 shows an example, an extract from the "2012 Premiership Rugby Sevens Series" article as it looks like after our processing procedure.
In case these markers are not of any use, we expect that removing them will be quite easy since they never occur in the original corpus. Finally, we add those markers as special tokens in our SentencePiece models to avoid lengthening our sequences unnecessarily with many tokens per marker. This way, each marker occupies only one time step in our sequence.

2.4. Dataset Splits

For each language, we split our articles into three parts; train (90%), dev (5%), and test (5%). We are aware that other researchers (including us) are interested in publishing newer versions of Wikipedia. To make sure that no data leaks across from training to test, we split our articles according to the Python3 hash of the Wikidata ID.[9] We calculate the hash value modulo 100, where the first 90 are dedicated to the train split, followed by 5 for dev and the last 5 for test.

2.5. Corpus Statistics

Table 2 shows statistics from our extracted and processed dataset. The entire dataset contains roughly 40B characters from 19.5M Wikipedia pages. We only obtain the accurate number of tokens and characters for dev and test sets for each language. For the training set, the statistics are estimated from the dev and test sizes, since we do not require the accurate training set size for training or evaluation.
Given the diversity of languages included in our dataset, we refrain from using rule-based tokenizers to estimate the number of words. Instead, we train a statistical based text processor, SentencePiece (SPM) (Kudo and Richardson, 2018). The number of tokens produced by a SentencePiece model depends on its vocabulary size. We choose a vocabulary size of 32K for our monolingual SPM models. For each language we train its own SPM model and then process its corpus accordingly. Table 2 shows the general statistics of each language, as well as the number of tokens and characters per split and the average number of characters per token. Note that when we count the number of characters in each dataset, the structural markers are counted as 1 character.

2.6. Data Format and License

We released the data in the TensorFlow Datasets format described as follows. The processed text will be released under the CC-BY-SA license, inheriting the license from the Wikipedia source text.
We use the TensorFlow datasets (`tfds`) API to offer a familiar interface to our data. This will enable researchers to inspect, load, and process the data quickly and with ease. The texts of the different languages are released separately. Each article is stored as a `FeaturesDict`[10] which includes two features for now:

Feature Name	Type	Description
wikidata_id	string	Unique ID given to the respective Wikidata entity (Barack Obama Article → `Q67`)
text	string	Processed text as shown in Figure 1

Listing 1 shows an example code snippet of how to load the training set with a batch size of 30. Each batch contains one Wikipedia article and its Wikidata ID.

3. Vocabulary

We obtain our model vocabularies using SentencePiece models (SPM). SentencePiece, introduced by Kudo and Richardson (2018) is a language-independent tokenizer and detokenizer designed to avoid relying on rule and domain

[9] `https://www.wikidata.org/wiki/Wikidata:Identifiers`
[10] `https://www.tensorflow.org/datasets/api_docs/python/tfds/features/FeaturesDict`

Language	Language Code	# Pages	# Sections	# SPM Tokens (M)			# Characters (M)			# Chars / # Tokens
				train	dev	test	train	dev	test	
English	en	3,426,657	11,378,343	1988.8	111.0	110.0	8862.1	494.7	489.9	4.456
German	de	1,752,761	5,466,644	901.6	50.1	50.1	4210.3	234.0	233.8	4.670
French	fr	1,540,579	4,989,635	689.3	38.4	38.2	2894.9	161.3	160.4	4.200
Russian	ru	1,060,586	2,701,885	513.8	28.6	28.5	2113.4	117.8	117.1	4.113
Spanish	es	1,018,751	3,017,131	532.3	29.3	29.8	2370.1	130.5	132.8	4.453
Italian	it	957,432	2,827,294	385.4	21.5	21.3	1730.3	96.7	95.6	4.489
Japanese	ja	889,932	2,651,078	359.6	19.9	20.0	693.9	38.5	38.6	1.930
Chinese Simplified	zh-cn	660,505	1,630,116	195.9	11.0	10.8	302.9	17.0	16.6	1.546
Chinese Traditional	zh-tw	652,328	1,611,524	199.7	11.2	11.0	308.2	17.3	17.0	1.543
Polish	pl	605,658	1,290,306	198.6	11.0	11.0	858.5	47.7	47.7	4.322
Ukrainian	uk	562,612	1,306,643	205.6	11.3	11.5	828.6	45.6	46.4	4.029
Dutch	nl	523,689	1,309,822	182.0	10.3	9.9	819.6	46.5	44.6	4.502
Swedish	sv	518,253	925,777	111.9	6.2	6.2	490.3	27.4	27.1	4.382
Portuguese	pt	485,005	1,294,787	212.4	11.8	11.8	935.6	52.0	52.0	4.406
Serbian	sr	373,632	628,691	76.2	4.4	4.1	286.2	16.5	15.3	3.755
Hungarian	hu	327,488	830,651	116.6	6.4	6.6	500.1	27.4	28.2	4.289
Catalan	ca	321,737	929,496	155.5	8.5	8.8	644.0	35.2	36.4	4.143
Czech	cs	307,913	925,119	120.1	6.7	6.6	489.2	27.3	27.1	4.074
Finnish	fi	296,389	632,095	95.5	5.3	5.3	441.4	24.5	24.6	4.622
Arabic	ar	283,820	766,236	108.3	6.0	6.0	421.3	23.5	23.3	3.890
Korean	ko	256,885	630,014	78.1	4.4	4.3	167.3	9.3	9.3	2.143
Persian	fa	245,533	510,137	60.1	3.3	3.4	235.4	12.9	13.3	3.919
Norwegian	no	228,481	524,044	73.9	4.2	4.1	317.6	17.9	17.4	4.296
Vietnamese	vi	223,825	556,671	85.8	4.8	4.7	331.6	18.6	18.2	3.864
Hebrew	he	187,522	605,551	120.8	6.7	6.8	450.7	24.9	25.2	3.731
Indonesian	id	185,343	422,884	55.8	3.0	3.2	280.6	15.0	16.1	5.026
Romanian	ro	175,565	379,418	60.7	3.2	3.6	259.6	13.6	15.3	4.276
Turkish	tr	170,378	487,296	53.0	3.0	2.9	244.6	13.6	13.5	4.611
Bulgarian	bg	150,458	335,644	52.2	2.9	2.9	215.4	12.0	11.9	4.126
Estonian	et	130,535	268,757	34.4	2.0	1.9	147.2	8.3	8.1	4.277
Malay	ms	130,177	262,004	23.0	1.3	1.3	115.5	6.4	6.4	5.010
Danish	da	128,613	291,434	47.8	2.6	2.7	206.8	11.3	11.7	4.327
Slovak	sk	122,325	280,724	32.4	1.7	1.9	129.9	6.8	7.7	4.004
Croatian	hr	119,781	390,199	47.8	2.7	2.6	197.6	11.0	10.9	4.136
Greek	el	107,317	314,647	62.4	3.4	3.5	270.3	14.8	15.2	4.331
Lithuanian	lt	98,319	191,785	26.2	1.5	1.5	111.2	6.2	6.2	4.240
Slovenian	sl	74,567	198,295	31.0	1.7	1.8	127.6	7.0	7.2	4.111
Thai	th	71,295	185,766	22.2	1.2	1.3	100.7	5.4	5.8	4.542
Hindi	hi	64,970	224,452	26.2	1.5	1.5	102.1	5.7	5.7	3.890
Latvian	lv	39,350	93,571	14.8	0.8	0.8	64.6	3.5	3.7	4.367
Filipino	tl	30,586	48,052	5.3	0.3	0.3	22.6	1.3	1.2	4.276
Total		19,507,552	54,314,618	8363.2	465.0	464.2	34299.8	1906.8	1904.3	4.101

Table 2: Statistics for the dataset organized by languages. The number of tokens is determined by our SPM trained with a vocabulary size of 32k. The number of SPM tokens and number of characters for the train split are estimated based on the sizes of dev and test.

```
import tensorflow_datasets as tfds

def loop_wikipedia(training_data_path,
                   num_articles=30,
                   iterations=100):
  data, info = tfds.load(data_path,
                         with_info=True)
  train = data['train'].cache()
  train = train.shuffle(iterations).batch(num_articles)
  batch = next(iter(train))
  return zip(batch['wikidata_id'], batch['text'])
```

Listing 1: An example of how to load the features of the training set.

knowledge based tokenizers to process text. We train separate SPM models for each monolingual model and a combined one for multilingual models with varying vocabulary sizes.

For monolingual models, we train an SPM with vocabulary size of 32k for each individual language using 100k articles from the corresponding training set. Each of our monolingual SPM reaches 99.9% coverage on both the dev and test sets for its corresponding language.

In the case of multilingual models, we train two SentencePiece models with 64k and 128k vocabulary sizes. To train the multilingual models, we sampled 10k articles from the training set of each language to obtain 410k articles in total. By sampling languages equally, we avoid high volume

languages such as English dominating the vocabulary. Our coverage test on the dev and test sets of the individual languages shows that we achieve 99.8%-99.9% coverage on all languages.

Table 3 shows the average characters per token of the 64k and 128k SPM measured on each language's dev and test sets,[11] and compared to the monolingual SPM.

	Chars/Token		
	Mono	Multi	
Language	32k	64k	128k
en	4.456	3.662	4.021
de	4.670	3.718	4.097
fr	4.200	3.373	3.689
es	4.453	3.615	3.951
ru	4.113	2.947	3.343
zh-tw	1.543	1.235	1.264
zh-cn	1.546	1.227	1.255
ar	3.890	2.594	2.932
vi	3.864	3.296	3.530
el	4.331	2.818	3.272
bg	4.126	2.984	3.355
tr	4.611	3.420	3.842
hi	3.890	2.515	2.897
th	4.542	2.708	3.125

Table 3: Average number of characters per token measured on each individual language's dev and test sets using the multilingual SPM in comparison to the monolingual SPM.

4. Models

To provide a solid starting point for the proposed dataset, we use Transformer-XL (Dai et al., 2019), the state-of-the-art architecture for language modeling, as the baseline model. In a nutshell, Transformer-XL extends the standard Transformer (Vaswani et al., 2017) with (1) a segment-level recurrence mechanism and (2) a relative positional encoding scheme. As a benefit, Transformer-XL is able to reuse hidden states from previous segments as additional context in language modeling training, achieving the effect of truncated back-propagation through time (T-BPTT).

Specifically, we denote the m-th layer hidden states of two consecutive segments as $H_{\tau-1}^{(m)}$ and $H_{\tau}^{(m)}$ respectively. Then, to produce the higher-layer hidden $H_{\tau}^{(m+1)}$, the standard Transformer performs self-attention based only on $H_{\tau}^{(m)}$:

$$H_{\tau}^{(m+1)} \leftarrow \text{Attn}\left(\text{Q} = H_{\tau}^{(m)}, \text{KV} = H_{\tau}^{(m)}\right).$$

In comparison, Transformer-XL utilizes relative attention to reuse the hidden states from previous segment $H_{\tau-1}$ to provide additional context information:

$$H_{\tau}^{(m+1)} \leftarrow \text{Rel-Attn}\left(\text{Q} = H_{\tau}, \text{KV} = [\text{SG}(H_{\tau-1}^{(m)}), H_{\tau}^{(m)}]\right),$$

where $[\cdot,\cdot]$ denotes concatenation and $\text{SG}(\cdot)$ means stop gradient, emphasizing the fact that the gradient is not passed across segments. In theory, one can reuse more than one previous segment, leading to an even larger context. This strategy is usually used during evaluation to fully exploit the model's capacity.

4.1. Model Usage

Our benchmarking models are publicly available on TensorFlow Hub (tfhub).[12] Listing 2 illustrates getting the model's likelihood of a given text, and getting the text embeddings. See also the project website for updates to the code example.

```
import tensorflow_hub as hub
import tensorflow as tf

module = hub.Module(path_to_model)

log_likelihood = module(dict(
    text=["The capital of the United States is
     Washington D.C."]),
    signature="log_likelihood", as_dict=True)
log_likelihood # >>> log_likelihood = -5.365

embeddings = module(
    dict(text=["Barack Obama is 58 years old."]),
    signature="embeddings", as_dict=True)
tf.shape(embeddings) # >>> [1, 512, 768]
```

Listing 2: Example of getting the log likelihood and embeddings from our models.

5. Evaluation

5.1. Bits per Character

Our models output token-level perplexity. However, SPM models with different vocabulary sizes will generate a different number of tokens for the dev and test sets, and therefore, producing incomparable numbers. To compare results from models trained on different text segmentations, we follow (Al-Rfou et al., 2019) and calculate bits per character (bpc) over the set under consideration. The calculation is shown as the following:

$$\text{bpc} = \log_2(\text{ppl per token}) \times \frac{\text{\# tokens}}{\text{\# chars}}$$

Note that the bpc values of different languages are not meant to be compared to each other. We report the bpc values to compare the performance of the different models for a given language.

5.2. Monolingual Benchmark

In this experiment, we train a model for each language separately to set our monolingual benchmark for these languages. For each language, we train an SPM with a vocabulary size of 32K, a medium sized Transformer-XL of 12 layers, with hidden size 768, and 12 attention heads with 64 dimensions, leading to a total number of 141.4M parameters.

During training, we use a segment length of 512 and only reuse one previous segment. The batch size ranges from 512 to 32 depending on the dataset size. To account for the

[11] We report the numbers of 14 chosen languages in the paper, and the full report are available in the appendix, and on the project website: https://www.tensorflow.org/datasets/catalog/wiki40b

[12] https://www.tensorflow.org/hub/

Language	dev			test		
	# SPM tokens	# characters	bpc	# SPM tokens	# characters	bpc
en	111,018,982	494,743,191	0.861	109,963,773	489,931,919	0.860
de	50,073,983	234,000,586	0.846	50,099,687	233,811,691	0.844
fr	38,404,735	161,251,396	0.772	38,185,922	160,399,436	0.773
es	29,296,731	130,522,477	0.795	29,847,318	132,819,185	0.795
ru	28,629,847	117,757,871	0.851	28,463,806	117,061,332	0.850
zh-tw	11,204,195	17,287,915	2.787	10,980,467	16,951,793	2.800
zh-cn	10,990,477	17,019,128	2.794	10,776,116	16,639,874	2.806
ar	6,022,834	23,500,808	1.060	6,010,250	23,310,912	1.055
vi	4,819,344	18,623,431	0.891	4,715,795	18,220,009	0.891
el	3,422,437	14,843,833	0.754	3,511,864	15,190,485	0.760
bg	2,914,683	12,016,269	0.760	2,885,102	11,913,832	0.759
tr	2,955,386	13,647,118	0.800	2,938,855	13,530,147	0.810
hi	1,456,834	5,671,760	0.838	1,458,692	5,670,195	0.818
th	1,175,301	5,368,067	0.761	1,287,601	5,818,713	0.752

Table 4: Monolingual Benchmark. Full table for all languages in Appendix.

Vocab	Lang	dev			test		
Size	Code	tokens	chars	bpc	tokens	chars	bpc
64k	en	135,084,994	494,743,191	0.998	133,787,289	489,931,919	0.998
	de	62,920,839	234,000,586	0.952	62,917,616	233,811,691	0.951
	fr	47,796,872	161,251,396	0.977	47,550,298	160,399,436	0.978
	es	36,098,031	130,522,477	1.007	36,753,941	132,819,185	1.009
	ru	39,959,378	117,757,871	1.050	39,729,534	117,061,332	1.050
	zh-tw	13,998,653	17,287,915	3.555	13,717,242	16,951,793	3.576
	zh-cn	13,845,393	17,019,128	3.568	13,578,848	16,639,874	3.574
	ar	9,043,446	23,500,808	1.549	9,002,051	23,310,912	1.546
	vi	5,646,618	18,623,431	1.190	5,530,379	18,220,009	1.195
	el	5,263,897	14,843,833	1.151	5,394,558	15,190,485	1.159
	bg	4,029,562	12,016,269	1.179	3,988,805	11,913,832	1.179
	tr	3,990,217	13,647,118	1.255	3,956,033	13,530,147	1.259
	hi	2,252,476	5,671,760	1.535	2,258,029	5,670,195	1.529
	th	1,978,178	5,368,067	1.475	2,153,547	5,818,713	1.461
128k	en	123,035,697	494,743,191	0.975	121,851,443	489,931,919	0.975
	de	57,092,340	234,000,586	0.925	57,093,429	233,811,691	0.923
	fr	43,706,988	161,251,396	0.951	43,474,952	160,399,436	0.952
	es	33,023,795	130,522,477	0.979	33,625,843	132,819,185	0.980
	ru	35,222,993	117,757,871	1.022	35,013,541	117,061,332	1.022
	zh-tw	13,679,748	17,287,915	3.500	13,406,647	16,951,793	3.527
	zh-cn	13,536,010	17,019,128	3.510	13,276,633	16,639,874	3.514
	ar	7,995,449	23,500,808	1.490	7,967,704	23,310,912	1.488
	vi	5,272,687	18,623,431	1.153	5,164,711	18,220,009	1.159
	el	4,531,987	14,843,833	1.098	4,646,593	15,190,485	1.110
	bg	3,582,487	12,016,269	1.141	3,550,987	11,913,832	1.140
	tr	3,552,468	13,647,118	1.208	3,520,978	13,530,147	1.212
	hi	1,956,262	5,671,760	1.475	1,959,353	5,670,195	1.481
	th	1,713,594	5,368,067	1.409	1,866,275	5,818,713	1.400

Table 5: Multilingual Benchmark. Full table for all languages in Appendix.

large variance in dataset size across the languages in consideration, we also vary the dropout rate to prevent overfitting especially for low resource languages. During evaluation, we dramatically increase the reuse length to the previous 4096 tokens.

We stop the training when the dev performance stops improving more than 0.1 for 5 consecutive checkpoints (50k steps), and we test the checkpoint with the lowest token-level perplexity on the dev set.

Table 4 shows the `bpc` of our monolingual models on the dev and test sets for a sample of 14 languages.[13]

5.3. Multilingual Benchmark

The multilingual models follow the same Transformer-XL structure as the monolingual models. We experiment with two different vocabulary setups for SPM: 64k and 128k, both trained on 40+ languages sampled equally. During training, we simply mix the text of all languages to-

[13]Full results are in the appendix and on the project website.

gether without balancing the data sampling across languages. Since the combined training set is quite large, we remove any dropout and always use a batch size of 512. Other hyper-parameters are kept the same as in the monolingual models.

For evaluation, instead of evaluating the model on a mix of all languages, we evaluate the model on each language separately while using the corresponding multilingual vocabulary (multilingual SPM). Similar to the monolingual model training, we stop the training when the dev performance on all languages we are evaluating on stops improving more than 0.1 for 5 consecutive checkpoints (50k steps), and we test the checkpoint with the lowest average ppl on dev.

Table 5 shows the bpc on our multilingual models with different vocabulary sizes. We report our evaluation on the same 14 languages reported in the monolingual benchmark, and the full results are available in the appendix and on the project website page.

6. Results & Discussion

Table 4 shows the results of our monolingual models evaluated on their respective dev and test sets. With exception of Chinese and Japanese (Table 7 in Appendix), all models achieve a compression rate of less than two bits per character, many even less than one bit.

Table 5 shows the results of evaluating two models with shared vocabularies against 14 languages; a large model with 128K pieces and a small one with 64K pieces. First, we observe that the large model produces better results than the small one across all languages. A larger embedding table provides higher capacity for each individual language. Second, multilingual models under-perform across all benchmarks in comparison to the monolingual models. While we hope that we can transfer knowledge and statistical strength from rich resource languages to low resource ones, the main mechanism actively influencing our results seems to be interference. An interesting approach would be to make the mix sensitive to the typology or language family of the involved languages (Gerz et al., 2018). Adding more languages to train against complicates the dynamics of learning significantly. Similar results have been reported previously (Conneau et al., 2019; Aharoni et al., 2019).

Figure 2 shows several examples of generated text from our multilingual model. The examples are seeded with variable length input in English, German, and Chinese. The seed text is taken from the corresponding dev sets. Our model is able to understand the context of the input and generate an on-topic structured completion in the same language.

7. Related Work

Language modeling is a self-supervised task that aims to reconstruct the data given partial information. Causal language models predict future tokens having access only to past ones. This factorization makes them applicable in estimating the likelihood of sentences and generating text for translation and assisted writing applications. This likelihood estimation offers an intrinsic evaluation metric that allows researchers to test different modeling ideas and model architectures.

_START_ARTICLE_
BBC World Service
_START_SECTION_
History
_START_PARAGRAPH_
BBC World Service World Service launched service on August 23, 1956 at Barry Island power station, Port Charlotte.
In July 2003, BBC World Service acquired at random power station Docklands at Port Charlotte, Port Charlotte. On 30 December 2005, formally Coronation Castle, Portsmouth, stood on the site of Great British Railway.

_START_ARTICLE_
Stephen J. Solarz
_START_PARAGRAPH_
Stephen Joshua Solarz (* 12. September 1940 in New York City; † 29. November 2010 in Washington, D.C.) war ein US-amerikanischer Politiker der Demokratischen Partei, der von 1993 bis 1999 Stabschef Präsident Woodrow Wilson (parteilos) war und die Rückkehr in die offizielle Position seit 30. September 2000, in den er ursprünglich oft als politischer Mitarbeiter angestellt war.

_START_ARTICLE_
冰火山
_START_PARAGRAPH_ 冰火山（拉丁文作cryovolcano，字面涵即cryo-：冰和volcano：火山），是存在于地外天体上的与火山相似的一地貌
形，其却深入研究了地球的大特征以火山当其高度地点而得名。
搭配火山在另一方面，英国火星在地外天体上的活常被是很重要的原因，有火山可能是由物自来而来原有的物化形式流于虚圈岩石上叫做冰火山。

Figure 2: Generated text from our 64K multilingual model in English, German, and Chinese. Seed text is bold, and is taken from the corresponding dev sets.

7.1. Datasets

Several datasets have been proposed to evaluate modeling architectures:

lm1b is a processed form of data obtained from WMT11.[14] The data adds up to one billion words cover-

[14] http://statmt.org/wmt11/training-monolingual.tgz

ing solely English news. The sentences have been shuffled in the original data limiting the ability to model longer term dependencies across sentences.

Our effort differs in that we only shuffle articles, therefore the structure within an article is kept intact. In reporting the size of our dataset, we report the number of characters and estimated counts of tokens.

enwik8/enwik9/text8 is an English Wikipedia dump of March 3rd 2016 that is extensively used as a benchmark for text compression for the Hutter's prize competition. The data is available both in a processed form (`text8`) and with Wikipedia markup kept in place (`enwik8`, `enwik9`).

Similar to this effort, we utilize Wikipedia on a way larger scale with a fresher dump of data. Moreover, we take a more conservative approach in dealing with the markup language. We keep a minimal set of sequence control markers since they could help with generation tasks (Keskar et al., 2019).

Penn Treebank is a corpus composed of only 4.5 million words and is getting used less often given the ease of training large models. Modeling can easily overfit on such a small corpus (Marcus et al., 1993; Prasad et al., 2008).

Europarl is a corpus mainly used for machine translation. It also has been recently utilized for its multilinguality to study the complexity of modeling different languages (Koehn, 2005). The main limitation of this approach is that it is limited to European languages.

7.2. Causal Language Models

In the last few years, the field of (causal) language modeling has gradually shifted from N-Gram models (Chen and Goodman, 1999) to neural language models. Neural language modeling was first explored using simple multi-layer perception (MLP) trained on fixed length segments (Bengio et al., 2003; Mnih and Hinton, 2007). Soon after that, based on truncated back-propagation through time training, vanilla recurrent neural networks (RNN) (Mikolov et al., 2010) and an advanced variant long short-term memory (LSTM) (Graves, 2013; Jozefowicz et al., 2016) were employed to capture longer contextual information. Meanwhile, various initialization (Le et al., 2015), optimization (Pascanu et al., 2013) and regularization (Zaremba et al., 2014; Merity et al., 2017) techniques have been proposed to improve RNN training. Later, also convolutional neural networks (CNN) (Dauphin et al., 2017) were considered to improve the speed.

More recently, the newly emerged Transformer architecture (Vaswani et al., 2017) is brought into language modeling which leads to a dramatic performance gain (Al-Rfou et al., 2019). However, similar to the MLP, Transformer can only perform fixed length training, limiting the contextual information it has access to. By properly employing relative attention and designing a segment-level recurrence mechanism, Transformer-XL (Dai et al., 2019) removes this limitation and effectively enables T-BPTT training for the Transformer architecture.

8. Conclusion

We introduce a high quality multilingual Wikipedia dataset with around 40 billion characters for benchmarking the research progress in language modeling for 40+ languages. We consistently split the dataset into train, dev, and test sets so that researchers can fairly compare future model developments on this dataset.

This dataset includes many low resource languages, where the data for down-stream tasks is small if it exists at all. While extrinsic evaluation relies on down-stream tasks, the intrinsic evaluation metrics of causal language modeling enable researchers to evaluate new architectures reliably without down-stream tasks. By releasing this dataset, we hope to provide a standard dataset for training and evaluating language models for many languages, and advance the modeling techniques for those languages.

Moreover, along with the dataset, we release the monolingual models and multilingual models trained using the state-of-the-art Transformer-XL architecture, and we set the initial benchmarks on the 40+ languages with these models.

Training causal language models with a multilingual corpus that contains a mixture of languages is uncommon. From our results, we observe a gap between the performance of monolingual models and multilingual models. We hope this multilingual causal language modeling task can pose new challenges for researchers. Future work is to investigate optimizing vocabulary setups and model structures to improve transfer learning from high resource languages to low resource languages, possibly within language families, while the interference on high resource languages should be minimized.

9. Acknowledgements

We thank Dokook Choe, Ciprian Chelba, and Bryan Perozzi for their valuable feedback, and Etienne Pot and Adam Roberts for their support for adding the dataset to Tensorflow Dataset API. We also thank Jiang Bian, Jie Mao, Yuan Gao, Xiaoyi Ren, Zhicheng Zheng, Cherry Ng, and Wenjie Song for their work on cleaning up and processing the raw Wikipedia text, and Mike Lee, Weizhao Wang, Daphne Luong, and Chuck Wu for their organizational support.

10. Bibliographical References

Aharoni, R., Johnson, M., and Firat, O. (2019). Massively multilingual neural machine translation. In *ACL 2019*, pages 3874–3884. ACL, June.

Al-Rfou, R., Choe, D., Constant, N., Guo, M., and Jones, L. (2019). Character-level language modeling with deeper self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3159–3166.

Baayen, R. H. (1996). The effects of lexical specialization on the growth curve of the vocabulary. *Comput. Linguist.*, 22(4):455–480, December.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Chelba, C., Mikolov, T., Schuster, M., Ge, Q., Brants, T., Koehn, P., and Robinson, T. (2013). One billion word

benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394.

Chen, M. X., Lee, B. N., Bansal, G., Cao, Y., Zhang, S., Lu, J., Tsay, J., Wang, Y., Dai, A. M., Chen, Z., Sohn, T., and Wu, Y. (2019). Gmail smart compose: Real-time assisted writing. *CoRR*, abs/1906.00080.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Dai, Z., Yang, Z., Yang, Y., Cohen, W. W., Carbonell, J., Le, Q. V., and Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. (2017). Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 933–941. JMLR. org.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Gerz, D., Vulić, I., Ponti, E. M., Reichart, R., and Korhonen, A. (2018). On the relation between linguistic typology and (limitations of) multilingual language modeling. In *EMNLP 2018*, pages 316–327. ACL.

Graves, A. (2013). Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Henderson, M., Al-Rfou, R., Strope, B., Sung, Y., Lukács, L., Guo, R., Kumar, S., Miklos, B., and Kurzweil, R. (2017). Efficient natural language response suggestion for smart reply. *CoRR*, abs/1705.00652.

Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., and Wu, Y. (2016). Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

Kageura, K. (2012). *The quantitative analysis of the dynamics and structure of terminologies*, volume 15. John Benjamins Publishing.

Kannan, A., Kurach, K., Ravi, S., Kaufman, T., Miklos, B., Corrado, G., Tomkins, A., Lukacs, L., Ganea, M., Young, P., and Ramavajjala, V. (2016). Smart reply: Automated response suggestion for email. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) (2016)*.

Keskar, N. S., McCann, B., Varshney, L. R., Xiong, C., and Socher, R. (2019). Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

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

Kudo, T. and Richardson, J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR*, abs/1808.06226.

Le, Q. V., Jaitly, N., and Hinton, G. E. (2015). A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*.

Mahoney, M. (2009). Large text compression benchmark. http://www.mattmahoney.net/text/text.html.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Merity, S., Keskar, N. S., and Socher, R. (2017). Regularizing and optimizing lstm language models. *arXiv preprint arXiv:1708.02182*.

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Milne, D. and Witten, I. H. (2013). An open-source toolkit for mining wikipedia. *Artificial Intelligence*, 194:222–239.

Mnih, A. and Hinton, G. (2007). Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM.

Pascanu, R., Mikolov, T., and Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A. K., and Webber, B. L. (2008). The penn discourse treebank 2.0. In *LREC*. Citeseer.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding with unsupervised learning. Technical report, Technical report, OpenAI.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Vrandečić, D. and Krötzsch, M. (2014). Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57:78–85.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Zaremba, W., Sutskever, I., and Vinyals, O. (2014). Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Zesch, T., Müller, C., and Gurevych, I. (2008). Extracting lexical semantic knowledge from wikipedia and wiktionary. In *LREC*, pages 1646–1652.

Appendix
for Wiki-40B: Multilingual Language Model Dataset

A Full Report on Avg Chars per Tokens

	Chars/Token		
	Mono	Multi	
Language	32k	64k	128k
ar	3.890	2.594	2.932
bg	4.126	2.984	3.355
ca	4.143	3.429	3.699
cs	4.074	3.165	3.536
da	4.327	3.519	3.864
de	4.670	3.718	4.097
el	4.331	2.818	3.272
en	4.456	3.662	4.021
es	4.453	3.615	3.951
et	4.277	3.181	3.554
fa	3.919	2.733	3.081
fi	4.622	3.434	3.856
fr	4.200	3.373	3.689
he	3.731	2.585	2.923
hi	3.890	2.515	2.897
hr	4.136	3.235	3.591
hu	4.289	3.096	3.486
id	5.026	3.964	4.384
it	4.489	3.572	3.925
ja	1.930	1.434	1.522
ko	2.143	1.532	1.669
lt	4.240	3.197	3.572
lv	4.367	3.243	3.669
ms	5.010	3.949	4.372
nl	4.502	3.594	3.953
no	4.296	3.521	3.865
pl	4.322	3.210	3.629
pt	4.406	3.520	3.863
ro	4.276	3.266	3.594
ru	4.113	2.947	3.343
sk	4.004	3.120	3.445
sl	4.111	3.266	3.614
sr	3.755	2.719	3.029
sv	4.382	3.505	3.850
th	4.542	2.708	3.125
tl	4.276	3.402	3.728
tr	4.611	3.420	3.842
uk	4.029	2.836	3.222
vi	3.864	3.296	3.530
zh-cn	1.546	1.227	1.255
zh-tw	1.543	1.235	1.264

Table 6: Average number of characters per token measured on each individual language's dev and test sets using the multilingual SPM in comparison to the monolingual SPM.

B Full Report on Monolingual Benchmark

Language	dev			test		
	# SPM tokens	# characters	bpc	# SPM tokens	# characters	bpc
ar	6,022,834	23,500,808	1.060	6,010,250	23,310,912	1.055
bg	2,914,683	12,016,269	0.760	2,885,102	11,913,832	0.759
ca	8,489,856	35,192,575	0.782	8,783,540	36,366,223	0.785
cs	6,701,286	27,299,019	0.931	6,642,862	27,061,923	0.915
da	2,613,901	11,299,708	0.843	2,695,443	11,676,182	0.842
de	50,073,983	234,000,586	0.846	50,099,687	233,811,691	0.844
el	3,422,437	14,843,833	0.754	3,511,864	15,190,485	0.760
en	111,018,982	494,743,191	0.861	109,963,773	489,931,919	0.860
es	29,296,731	130,522,477	0.795	29,847,318	132,819,185	0.795
et	1,953,305	8,275,733	0.820	1,870,166	8,078,735	0.817
fa	3,287,657	12,866,631	1.026	3,385,518	13,288,734	1.029
fi	5,292,738	24,457,478	0.794	5,317,879	24,590,021	0.764
fr	38,404,735	161,251,396	0.772	38,185,922	160,399,436	0.773
he	6,660,416	24,870,599	1.224	6,761,302	25,208,735	1.223
hi	1,456,834	5,671,760	0.838	1,458,692	5,670,195	0.818
hr	2,677,654	11,042,019	0.831	2,631,167	10,917,511	0.827
hu	6,395,273	27,402,731	0.866	6,559,416	28,164,651	0.862
id	2,999,455	15,048,875	0.794	3,203,261	16,128,722	0.798
it	21,533,325	96,685,062	0.942	21,289,727	95,566,570	0.942
ja	19,944,020	38,505,964	2.225	20,006,470	38,591,027	2.221
ko	4,355,409	9,333,927	1.899	4,319,838	9,256,659	1.864
lt	1,453,533	6,157,214	0.677	1,461,354	6,203,086	0.698
lv	803,323	3,511,594	1.260	841,063	3,668,947	1.253
ms	1,282,845	6,385,620	0.626	1,277,727	6,442,537	0.624
nl	10,320,546	46,491,533	0.804	9,906,397	44,579,412	0.804
no	4,159,279	17,867,941	0.926	4,054,458	17,421,505	0.930
pl	11,037,356	47,676,997	0.823	11,032,088	47,714,220	0.826
pt	11,806,517	51,994,248	0.878	11,789,242	51,962,617	0.880
ro	3,180,639	13,570,445	0.806	3,563,981	15,269,698	0.798
ru	28,629,847	117,757,871	0.851	28,463,806	117,061,332	0.850
sk	1,691,677	6,762,384	0.799	1,913,170	7,670,401	0.803
sl	1,699,163	6,984,578	0.835	1,750,175	7,196,041	0.832
sr	4,407,772	16,515,617	1.217	4,062,276	15,289,063	1.224
sv	6,247,941	27,405,722	0.801	6,183,053	27,072,949	0.802
th	1,175,301	5,368,067	0.761	1,287,601	5,818,713	0.752
tl	300,620	1,279,276	0.896	286,789	1,232,607	0.866
tr	2,955,386	13,647,118	0.800	2,938,855	13,530,147	0.810
uk	11,334,897	45,625,835	0.885	11,514,315	46,442,543	0.884
vi	4,819,344	18,623,431	0.891	4,715,795	18,220,009	0.891
zh-cn	10,990,477	17,019,128	2.794	10,776,116	16,639,874	2.806
zh-tw	11,204,195	17,287,915	2.787	10,980,467	16,951,793	2.800

Table 7: Full Report on Monolingual Benchmark

C Full Report on Multilingual Benchmark

Vocab size	Language code	dev # SPM tokens	dev # characters	dev bpc	test # SPM tokens	test # characters	test bpc
64k	ar	9,043,446	23,500,808	1.549	9,002,051	23,310,912	1.546
	bg	4,029,562	12,016,269	1.179	3,988,805	11,913,832	1.179
	ca	10,258,702	35,192,575	1.007	10,610,690	36,366,223	1.009
	cs	8,628,469	27,299,019	1.274	8,548,911	27,061,923	1.276
	da	3,209,696	11,299,708	1.227	3,320,050	11,676,182	1.228
	de	62,920,839	234,000,586	0.952	62,917,616	233,811,691	0.951
	el	5,263,897	14,843,833	1.151	5,394,558	15,190,485	1.159
	en	135,084,994	494,743,191	0.998	133,787,289	489,931,919	0.998
	es	36,098,031	130,522,477	1.007	36,753,941	132,819,185	1.009
	et	2,612,040	8,275,733	1.438	2,529,282	8,078,735	1.435
	fa	4,709,457	12,866,631	1.441	4,859,812	13,288,734	1.454
	fi	7,123,950	24,457,478	1.182	7,159,734	24,590,021	1.186
	fr	47,796,872	161,251,396	0.977	47,550,298	160,399,436	0.978
	he	9,620,139	24,870,599	1.566	9,750,234	25,208,735	1.565
	hi	2,252,476	5,671,760	1.535	2,258,029	5,670,195	1.529
	hr	3,416,813	11,042,019	1.300	3,372,057	10,917,511	1.303
	hu	8,854,414	27,402,731	1.284	9,090,904	28,164,651	1.282
	id	3,800,868	15,048,875	1.115	4,063,809	16,128,722	1.113
	it	27,068,104	96,685,062	1.027	26,754,708	95,566,570	1.027
	ja	26,845,285	38,505,964	2.751	26,916,128	38,591,027	2.748
	ko	6,088,721	9,333,927	2.628	6,045,231	9,256,659	2.611
	lt	1,925,610	6,157,214	1.375	1,940,418	6,203,086	1.362
	lv	1,079,913	3,511,594	1.381	1,134,364	3,668,947	1.385
	ms	1,620,079	6,385,620	1.110	1,628,108	6,442,537	1.106
	nl	12,935,375	46,491,533	1.050	12,405,558	44,579,412	1.049
	no	5,079,302	17,867,941	1.205	4,942,734	17,421,505	1.201
	pl	14,861,943	47,676,997	1.090	14,856,299	47,714,220	1.093
	pt	14,778,797	51,994,248	1.076	14,757,449	51,962,617	1.076
	ro	4,156,428	13,570,445	1.153	4,673,383	15,269,698	1.145
	ru	39,959,378	117,757,871	1.050	39,729,534	117,061,332	1.050
	sk	2,169,568	6,762,384	1.287	2,456,919	7,670,401	1.289
	sl	2,139,454	6,984,578	1.365	2,203,030	7,196,041	1.362
	sr	6,068,711	16,515,617	1.275	5,629,151	15,289,063	1.282
	sv	7,814,743	27,405,722	1.138	7,729,640	27,072,949	1.140
	th	1,978,178	5,368,067	1.475	2,153,547	5,818,713	1.461
	tl	375,734	1,279,276	1.425	362,515	1,232,607	1.424
	tr	3,990,217	13,647,118	1.255	3,956,033	13,530,147	1.259
	uk	16,101,411	45,625,835	1.139	16,361,243	46,442,543	1.139
	vi	5,646,618	18,623,431	1.190	5,530,379	18,220,009	1.195
	zh-cn	13,845,393	17,019,128	3.568	13,578,848	16,639,874	3.574
	zh-tw	13,998,653	17,287,915	3.555	13,717,242	16,951,793	3.576

Table 8: Full Report on Multilingual Benchmark (64k vocabulary size)

Vocab size	Language code	dev # SPM tokens	dev # characters	dev bpc	test # SPM tokens	test # characters	test bpc
128k	ar	7,995,449	23,500,808	1.490	7,967,704	23,310,912	1.488
	bg	3,582,487	12,016,269	1.141	3,550,987	11,913,832	1.140
	ca	9,509,073	35,192,575	0.977	9,834,353	36,366,223	0.979
	cs	7,723,026	27,299,019	1.218	7,650,785	27,061,923	1.220
	da	2,923,938	11,299,708	1.179	3,021,614	11,676,182	1.182
	de	57,092,340	234,000,586	0.925	57,093,429	233,811,691	0.923
	el	4,531,987	14,843,833	1.098	4,646,593	15,190,485	1.110
	en	123,035,697	494,743,191	0.975	121,851,443	489,931,919	0.975
	es	33,023,795	130,522,477	0.979	33,625,843	132,819,185	0.980
	et	2,339,336	8,275,733	1.358	2,261,849	8,078,735	1.353
	fa	4,178,457	12,866,631	1.384	4,310,721	13,288,734	1.394
	fi	6,342,948	24,457,478	1.131	6,376,257	24,590,021	1.136
	fr	43,706,988	161,251,396	0.951	43,474,952	160,399,436	0.952
	he	8,509,397	24,870,599	1.492	8,625,416	25,208,735	1.492
	hi	1,956,262	5,671,760	1.475	1,959,353	5,670,195	1.481
	hr	3,078,872	11,042,019	1.249	3,037,053	10,917,511	1.251
	hu	7,866,214	27,402,731	1.235	8,073,247	28,164,651	1.232
	id	3,438,411	15,048,875	1.077	3,672,901	16,128,722	1.076
	it	24,636,227	96,685,062	0.996	24,344,353	95,566,570	0.995
	ja	25,291,638	38,505,964	2.709	25,371,338	38,591,027	2.705
	ko	5,588,291	9,333,927	2.561	5,549,816	9,256,659	2.537
	lt	1,723,908	6,157,214	1.306	1,736,306	6,203,086	1.297
	lv	953,995	3,511,594	1.315	1,003,184	3,668,947	1.318
	ms	1,463,675	6,385,620	1.068	1,470,774	6,442,537	1.061
	nl	11,759,797	46,491,533	1.019	11,275,929	44,579,412	1.018
	no	4,627,665	17,867,941	1.148	4,503,170	17,421,505	1.152
	pl	13,146,023	47,676,997	1.042	13,142,976	47,714,220	1.045
	pt	13,464,556	51,994,248	1.047	13,445,759	51,962,617	1.047
	ro	3,776,677	13,570,445	1.123	4,247,067	15,269,698	1.114
	ru	35,222,993	117,757,871	1.022	35,013,541	117,061,332	1.022
	sk	1,965,747	6,762,384	1.240	2,223,469	7,670,401	1.240
	sl	1,933,005	6,984,578	1.301	1,990,581	7,196,041	1.300
	sr	5,450,390	16,515,617	1.242	5,050,831	15,289,063	1.249
	sv	7,112,800	27,405,722	1.096	7,036,956	27,072,949	1.099
	th	1,713,594	5,368,067	1.409	1,866,275	5,818,713	1.400
	tl	343,044	1,279,276	1.400	330,711	1,232,607	1.389
	tr	3,552,468	13,647,118	1.208	3,520,978	13,530,147	1.212
	uk	14,170,617	45,625,835	1.107	14,401,153	46,442,543	1.105
	vi	5,272,687	18,623,431	1.153	5,164,711	18,220,009	1.159
	zh-cn	13,536,010	17,019,128	3.510	13,276,633	16,639,874	3.514
	zh-tw	13,679,748	17,287,915	3.500	13,406,647	16,951,793	3.527

Table 9: Full Report on Multilingual Benchmark (128k vocabulary size)