Are All Languages Created Equal in Multilingual BERT?

Shijie Wu and **Mark Dredze**
Department of Computer Science
Johns Hopkins University
shijie.wu@jhu.edu, mdredze@cs.jhu.edu

Abstract

Multilingual BERT (mBERT) (Devlin, 2018) trained on 104 languages has shown surprisingly good cross-lingual performance on several NLP tasks, even without explicit cross-lingual signals (Wu and Dredze, 2019; Pires et al., 2019). However, these evaluations have focused on cross-lingual transfer with high-resource languages, covering only a third of the languages covered by mBERT. We explore how mBERT performs on a much wider set of languages, focusing on the quality of representation for low-resource languages, measured by within-language performance. We consider three tasks: Named Entity Recognition (99 languages), Part-of-speech Tagging, and Dependency Parsing (54 languages each). mBERT does better than or comparable to baselines on high resource languages but does much worse for low resource languages. Furthermore, monolingual BERT models for these languages do even worse. Paired with similar languages, the performance gap between monolingual BERT and mBERT can be narrowed. We find that better models for low resource languages require more efficient pretraining techniques or more data.

1 Introduction

Pretrained contextual representation models trained with language modeling (Peters et al., 2018; Yang et al., 2019) or the cloze task objectives (Devlin et al., 2019; Liu et al., 2019) have quickly set a new standard for NLP tasks. These models have also been trained in multilingual settings. As the authors of BERT say "[...] (they) do not plan to release more single-language models", they instead train a single BERT model with Wikipedia to serve 104 languages, without any explicit cross-lingual links, yielding a multilingual BERT (mBERT) (Devlin, 2018). Surprisingly, mBERT learn high-quality cross-lingual representation and show strong zero-shot cross-lingual transfer performance (Wu and Dredze, 2019; Pires et al., 2019). However, evaluations have focused on high resource languages, with cross-lingual transfer using English as a source language or within language performance. As Wu and Dredze (2019) evaluated mBERT on 39 languages, this leaves the majority of mBERT's 104 languages, most of which are low resource languages, untested.

Does mBERT learn equally high-quality representation for its 104 languages? If not, which languages are hurt by its massively multilingual style pretraining? While it has been observed that for high resource languages like English, mBERT performs worse than monolingual BERT on English with the same capacity (Devlin, 2018). It is unclear that for low resource languages (in terms of monolingual corpus size), how does mBERT compare to a monolingual BERT? And, does multilingual joint training help mBERT learn better representation for low resource languages?

We evaluate the representation quality of mBERT on 99 languages for NER, and 54 for part-of-speech tagging and dependency parsing. In this paper, we show mBERT does not have equally high-quality representation for all of the 104 languages, with the bottom 30% languages performing much worse than a non-BERT model on NER. Additionally, by training various monolingual BERT for low-resource languages with the same data size, we show the low representation quality of low-resource languages is not the result of the hyper-parameters of BERT or sharing the model with a large number of languages, as monolingual BERT performs worse than mBERT. On the contrary, by pairing low-resource languages with linguistically-related languages, we show low-resource languages benefit from multilingual joint training, as bilingual BERT outperforms monolingual BERT while still lacking behind mBERT,

Our findings suggest, with small monolingual corpus, BERT does not learn high-quality representation for low resource languages. To learn better representation for low resource languages, we suggest either collect more data to make low resource language high resource (Conneau et al., 2019), or consider more data-efficient pretraining techniques like Clark et al. (2020). We leave exploring more data-efficient pretraining techniques as future work.

2 Related Work

Multilingual Contextual Representations Deep contextualized representation models such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) have set a new standard for NLP systems. Their application to multilingual settings, pretraining one model on text from multiple languages with a single vocabulary, has driven forward work in cross-language learning and transfer (Wu and Dredze, 2019; Pires et al., 2019; Mulcaire et al., 2019). BERT-based pretraining also benefits language generation tasks like machine translation (Conneau and Lample, 2019). BERT can be further improve with explicit cross-language signals including: bitext (Conneau and Lample, 2019; Huang et al., 2019) and word translation pairs from a dictionary (Wu et al., 2019) or induced from a bitext (Ji et al., 2019).

Several factors need to be considered in understanding mBERT. First, the 104 most common Wikipedia languages vary considerably in size (Table 1). Therefore, mBERT training attempted to equalize languages by up-sampling words from low resource languages and down-sampling words from high resource languages. Previous work has found that shared strings across languages provide sufficient signal for inducing cross-lingual word representations (Lample et al., 2018; Artetxe et al., 2017). While Wu and Dredze (2019) finds the number of shared subwords across languages correlates with cross-lingual performance, multilingual BERT can still learn cross-lingual representation without any vocabulary overlap across languages (Wu et al., 2019; K et al., 2020). Additionally, Wu et al. (2019) find bilingual BERT can still achieve decent cross-lingual transfer by sharing only the transformer layer across languages. Artetxe et al. (2019) shows learning the embedding layer alone while using a fixed transformer encoder from English monolingual BERT can also produce decent cross-lingual transfer performance. Second, while each language may be similarly represented in the training data, subwords are not evenly distributed among the languages. Many languages share common characters and cognates, biasing subword learning to some languages over others. Both of these factors may influence how well mBERT learns representations for low resource languages.

Finally, Baevski et al. (2019) show that in general larger pretraining data for English leads to better downstream performance, yet increasing the size of pretraining data exponentially only increases downstream performance linearly. For a low resource language with limited pretraining data, it is unclear whether contextual representations outperform previous methods.

Representations for Low Resource Languages Embeddings with subword information, a non-contextual representation, like fastText (Bojanowski et al., 2017) and BPEmb (Heinzerling and Strube, 2018) are more data-efficient compared to contextual representation like ELMo and BERT when a limited amount of text is available. For low resource languages, there are usually limits on **monolingual corpora** and **task specific supervision**. When task-specific supervision is limited, e.g. sequence labeling in low resource languages, mBERT performs better than fastText while underperforming a single BPEmb trained on all languages (Heinzerling and Strube, 2019). Contrary to this work, we focus on mBERT from the perspective of representation learning for each language in terms of monolingual corpora resources and analyze how to improve BERT for low resource languages. We also consider parsing in addition to sequence labeling tasks.

Concurrently, Conneau et al. (2019) train a multilingual masked language model (Devlin et al., 2019) on 2.5TB of CommonCrawl filtered data covering 100 languages and show it outperforms a Wikipedia-based model on low resource languages (Urdu and Swahili) for XNLI (Conneau et al., 2018). Using CommonCrawl greatly increases monolingual resource especially for low resource languages, and makes low resource languages in terms of Wikipedia size high resource. For example, Mongolian has 6 million and 248 million tokens in Wikipedia and CommonCrawl, respectively. Indeed, a 40-fold data increase of Mongolian (mn) increases its WikiSize, a measure of monolingual corpus size introduced in §3.1, from 5 to roughly 10, as shown in Tab. 1, making it

relatively high resource with respect to mBERT.

3 Experimental Setup

We begin by defining high and low resource languages in mBERT, a description of the models and downstream tasks we use for evaluation, followed by a description of the masked language model pretraining.

3.1 High/Low Resource Languages

Since mBERT was trained on articles from Wikipedia, a language is considered a high or low resource for mBERT based on the size of Wikipedia in that language. Size can be measured in many ways (articles, tokens, characters); we use the size of the raw dump archive file;[1] for convenience we use $\log_2$ of the size in MB (**WikiSize**). English is the highest resource language (15.5GB) and Yoruba the lowest (10MB).[2] Tab. 1 shows languages and their relative resources.

3.2 Downstream Tasks

mBERT supports 104 languages, and we seek to evaluate the learned representations for as many of these as possible. We consider three NLP tasks for which annotated task data exists in a large number of languages: named entity recognition (NER), universal part-of-speech (POS) tagging and universal dependency parsing. For each task, we train a task-specific model using within-language supervised data on top of the mBERT representation with fine-tuning.

For NER we use data created by Pan et al. (2017) automatically built from Wikipedia, which covers 99 of the 104 languages supported by mBERT. We evaluate NER with entity-level F1. This data is in-domain as mBERT is pretrained on Wikipedia. For POS tagging and dependency parsing, we use Universal Dependencies (UD) v2.3 (Nivre et al., 2018), which covers 54 languages (101 treebanks) supported by mBERT. We evaluate POS with accuracy (ACC) and Parsing with label attachment score (LAS) and unlabeled attachment score (UAS). For POS, we consider UPOS within the treebank. For parsing, we only consider universal dependency labels. The domain is treebank-specific so we use all treebanks of a language for completeness.

[1]The size of English (en) is the size of this file: https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2

[2]The ordering does not necessarily match the number of speakers for a language.

Task Models For sequence labeling tasks (NER and POS), we add a linear function with a softmax on top of mBERT. For NER, at test time, we adopt a simple post-processing heuristic as a structured decoder to obtain valid named entity spans. Specifically, we rewrite stand-alone prediction of `I-X` to `B-X` and inconsistent prediction of `B-X I-Y` to `B-Y I-Y`, following the final entity. For dependency parsing, we replace the LSTM in the graph-based parser of Dozat and Manning (2017) with mBERT. For the parser, we use the original hyperparameters. Note we do not use universal part-of-speech tags as input for dependency parsing. We fine-tune all parameters of mBERT for a specific task. We use a maximum sequence length of 128 for sequence labeling tasks. For sentences longer than 128, we use a sliding window with 64 previous tokens as context. For dependency parsing, we use sequence length 128 due to memory constraints and drop sentences with more than 128 subwords. We also adopt the same treatment for the baseline (Che et al., 2018) to obtain comparable results. Since mBERT operates on the subword-level, we select the first subword of each word for the task-specific layer with masking.

Task Optimization We train all models with Adam (Kingma and Ba, 2014). We warm up the learning rate linearly in the first 10% steps then decrease linearly to 0. We select the hyperparameters based on dev set performance by grid search, as recommended by Devlin et al. (2019). The search includes a learning rate (2e-5, 3e-5, and 5e-5), batch size (16 and 32). As task-specific supervision size differs by language or treebank, we fine-tune the model for 10k gradient steps and evaluate the model every 200 steps. We select the best model and hyperparameters for a language or treebank by the corresponding dev set.

Task Baselines We compare our mBERT models with previously published methods: Pan et al. (2017) for NER; For POS and dependency parsing the best performing system ranked by LAS in the 2018 universal parsing shared task (Che et al., 2018) [3], which use ELMo as well as word embeddings. Additionally, Che et al. (2018) is trained on POS and dependency parsing jointly while we trained mBERT to perform each task separately. As a result, the dependency parsing with mBERT

[3]The shared task uses UD v2.2 while we use v2.3. However, treebanks contain minor changes from version to version.

WikiSize	Languages	# Languages	Size Range (GB)
3	io, pms, scn, **yo**	4	[0.006, 0.011]
4	cv, lmo, mg, min, su, vo	6	[0.011, 0.022]
5	an, bar, br, ce, fy, ga, gu, is, jv, ky, lb, **mn**, my, nds, ne, pa, pnb, sw, tg	19	[0.022, 0.044]
6	**af**, ba, cy, kn, la, mr, oc, sco, sq, tl, tt, uz	12	[0.044, 0.088]
7	az, bn, bs, eu, hi, ka, kk, lt, **lv**, mk, ml, nn, ta, te, ur	15	[0.088, 0.177]
8	ast, be, bg, da, el, et, gl, hr, hy, ms, sh, sk, sl, th, war	15	[0.177, 0.354]
9	fa, fi, he, id, ko, no, ro, sr, tr, vi	10	[0.354, 0.707]
10	ar, ca, cs, hu, nl, sv, uk	7	[0.707, 1.414]
11	ceb, it, ja, pl, pt, zh	6	[1.414, 2.828]
12	de, es, fr, ru	4	[2.828, 5.657]
14	en	1	[11.314, 22.627]

Table 1: List of 99 languages we consider in mBERT and its pretraining corpus size. Languages in **bold** are the languages we consider in §5.

does not have access to POS tags. By comparing mBERT to these baselines, we control for task and language-specific supervised training set size.

3.3 Masked Language Model Pretraining

We include several experiments in which we pretrain BERT from scratch. We use the PyTorch (Paszke et al., 2019) implementation by Conneau and Lample (2019).[4] All sentences in the corpus are concatenated. For each language, we sample a batch of N sequence and each sequence contains M tokens, ignoring sentence boundaries. When considering two languages, we sample each language uniformly. We then randomly select 15% of the input tokens for masking, proportionally to the exponentiated token count of power -0.5, favoring rare tokens. We replace selected masked token with <MASK> 80% of the time, the original token 10% of the time, and uniform random token within the vocabulary 10% of the time. The model is trained to recover the original token (Devlin et al., 2019). We drop the next sentence prediction task as Liu et al. (2019) find it does not improve downstream performance.

Data Processing We extract text from a Wikipedia dump with Gensim (Řehůřek and Sojka, 2010). We learn vocabulary for the corpus using SentencePiece (Kudo and Richardson, 2018) with the unigram language model (Kudo, 2018). When considering two languages, we concatenate the corpora for the two languages while sampling the same number of sentences from both corpora when learning vocabulary. We learn a vocabulary of size V, excluding special tokens. Finally, we tokenized the corpora using the learned SentencePiece model and did not apply any further preprocessing.

BERT Models Following mBERT, We use 12 Transformer layers (Vaswani et al., 2017) with 12 heads, embedding dimensions of 768, hidden dimension of the feed-forward layer of 3072, dropout of 0.1 and GELU activation (Hendrycks and Gimpel, 2016). We tied the output softmax layer and input embeddings (Press and Wolf, 2017). We consider both a 12 layer model (**base**) and a smaller 6 layer model (**small**).

BERT Optimization We train BERT with Adam and an inverse square root learning rate scheduler with warmup (Vaswani et al., 2017). We warm up linearly for 10k steps and the learning rate is 0.0001. We use batch size $N = 88$ and mixed-precision training. We trained the model for roughly 115k steps and save a checkpoint every 23k steps, which correspond to 10 epochs. We select the best out of five checkpoints with a task-specific dev set. We train each model on a single NVIDIA RTX Titan with 24GB of memory for roughly 20 hours.

4 Are All Languages Created Equal in mBERT?

Fig. 1 shows the performance of mBERT and the baseline averaged across all languages by Wikipedia size (see Tab. 1 for groupings). For WikiSize over 6, mBERT is comparable or better than baselines in all three tasks, with the exception of NER. For NER in very high resource languages (WikiSize over 11, i.e. top 10%) mBERT performs worse than baseline, suggesting high resource languages could benefit from monolingual pretraining.

[4] https://github.com/facebookresearch/XLM

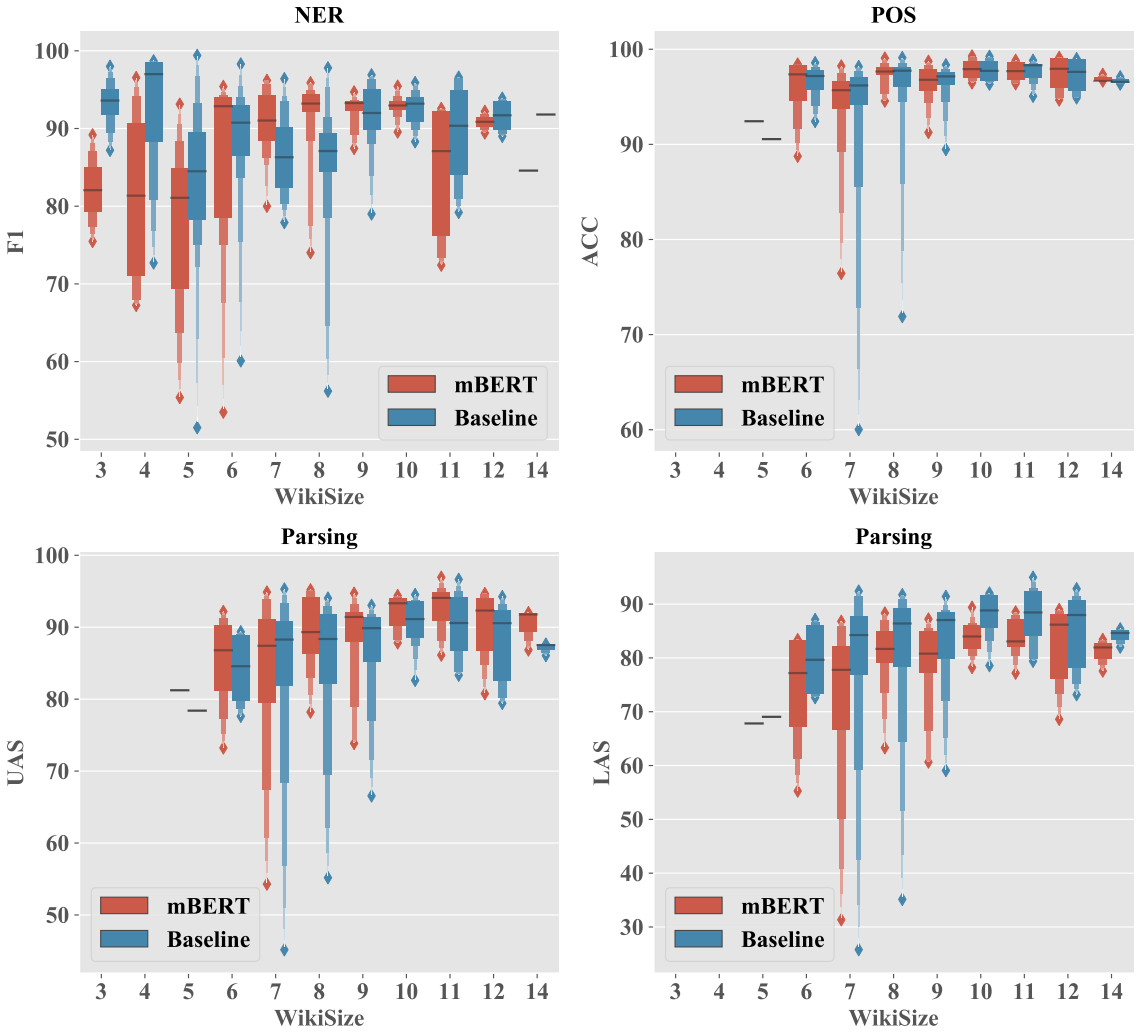

Figure 1: mBERT vs baseline grouped by WikiSize. mBERT performance drops much more than baseline models on languages lower than WikiSize 6 – the bottom 30% languages supported by mBERT – especially in NER, which covers nearly all mBERT supported languages.

Note mBERT has strong UAS on parsing but weak LAS compared to the baseline; Wu and Dredze (2019) finds adding POS to mBERT improve LAS significantly. We expect multitask learning on POS and parsing could further improve LAS. While POS and Parsing only cover half (54) of the languages, NER covers 99 of 104 languages, extending the curve to the lowest resource languages. mBERT performance drops significantly for languages with WikiSize less than 6 (bottom 30% languages). For the smallest size, mBERT goes from being competitive with state-of-the-art to being *over 10 points behind.* Readers may find this surprising since while these are very low resource languages, mBERT training up-weighted these languages to counter this effect.

Fig. 2 shows the performance of mBERT (only) for NER over languages with *different resources*, where we show how much task-specific supervised training data was available for each language. For languages with only 100 labeled sentences, the performance of mBERT drops significantly as these languages also had less pretraining data. While we may expect that pretraining representations with mBERT would be most beneficial for languages with only 100 labels, as Howard and Ruder (2018) show pretraining improve data-efficiency for English on text classification, our results show that on low resource languages this strategy performs much worse than a model trained directly on the available task data. Clearly, mBERT provides variable quality representations depending on the language. While we confirm the finding of others that mBERT is excellent for high resource languages, it

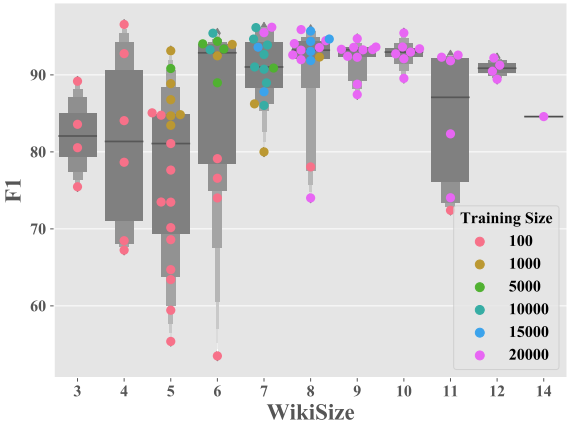

Figure 2: NER with mBERT on 99 languages, ordered by size of pretraining corpus (WikiSize). Task-specific supervised training size differs by language. Performance drops dramatically with less pretraining and supervised training data.

	Coefficient	p-value	CI
Univariate			
Training Size	0.035	<0.001	[0.029, 0.041]
Training Vocab	0.021	<0.001	[0.017, 0.025]
WikiSize	0.015	<0.001	[0.007, 0.023]
Multivariate			
Training Size	0.029	<0.001	[0.023, 0.035]
WikiSize	-0.014	<0.001	[-0.022, -0.006]

Table 2: Statistical analysis on what factors predict downstream performance. We fit two types of linear models, which consider either single factor or multiple factors.

is much worse for low resource languages. Our results suggest caution for those expecting a reliable model for *all* 104 mBERT languages.

5 Why Are All Languages Not Created Equal in mBERT?

5.1 Statistical Analysis

We present a statistical analysis to understand why mBERT does so poorly on some languages. We consider three factors that might affect the downstream task performance: pretraining Wikipedia size (WikiSize), task-specific supervision size, and vocabulary size in task-specific data. Note we take $\log_2$ of training size and training vocab following WikiSize. We consider NER because it covers nearly all languages of mBERT.

We fit a linear model to predict task performance (F1) using a single factor. Tab. 2 shows that each factor has a statistically significant positive correlation. One unit increase of training size leads to the biggest performance increase, then training vocabulary followed by WikiSize, all in log scale. Intuitively, training size and training vocab correlate with each other. We confirm this with a log-likelihood ratio test; adding training vocabulary to a linear model with training size yields a statistically insignificant improvement. As a result, when considering multiple factors, we consider training size and WikiSize. Interestingly, Tab. 2 shows training size still has a positive but slightly smaller slope, but the slope of WikiSize change sign, which suggests WikiSize might correlate with training size. We confirm this by fitting a linear model with training size as x and WikiSize as y and the slope is over 0.5 with $p < 0.001$. This finding is unsurprising as the NER dataset is built from Wikipedia so larger Wikipedia size means larger training size.

In conclusion, the larger the task-specific supervised dataset, the better the downstream performance on NER. Unsurprisingly, while pretraining improve data-efficiency (Howard and Ruder, 2018), it still cannot solve a task with limited supervision. Training vocabulary and Wikipedia size correlate with training size, and increasing either one factor leads to better performance. A similar conclusion could be found when we try to predict the performance ratio of mBERT and the baseline instead. Statistical analysis shows a correlation between resource and mBERT performance but can not give a causal answer on why low resource languages within mBERT perform poorly.

5.2 mBERT vs monolingual BERT

We have established that mBERT does not perform well in low-resource languages. Is this because we are relying on a multilingual model that favors high-resource over low-resource languages? To answer this question we train mono-lingual BERT models on several low resource languages with different hyperparameters. Since pretraining a BERT model from scratch is computationally intensive, we select four low resource languages: Latvian (lv), Afrikaans (af), Mongolian (mn), and Yoruba (yo). These four languages (bold font in Tab. 3) reflect varying amounts of monolingual training data.

It turns out that these low resource languages are reasonably covered by mBERT's vocabulary: 25% to 50% of the subword types within the mBERT 115K vocabulary appear in these lan-

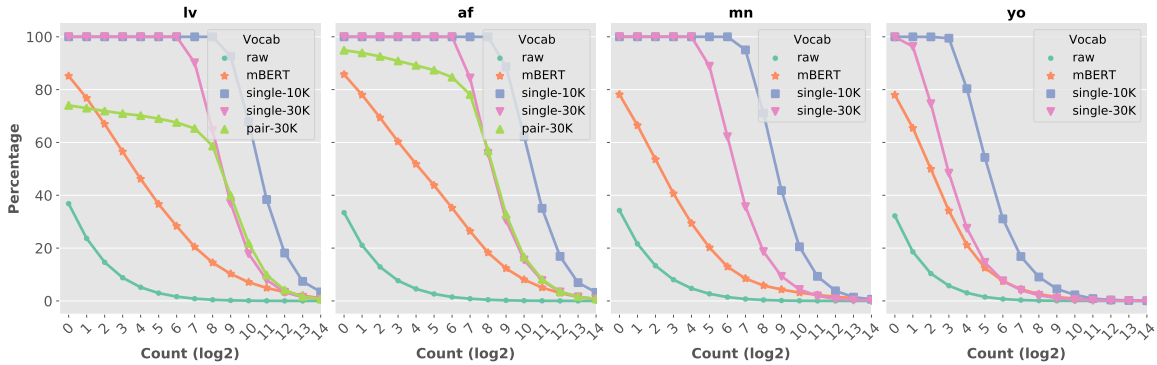

Figure 3: Percentage of vocabulary containing word count larger than a threshold. "Raw" is the vocabulary segmented by space. Single-30K and Single-10K are 30K/10K vocabularies learned from single languages. Pair-30K is 30K vocabulary learned from the selected language and a closely related language, described in §5.3.

	lv	af	mn	yo
Genus	Baltic	Germanic	Mongolic	Defoid
Family	Indo-Eur	Indo-Eur	Altaic	Niger-Congo
WikiSize	7	6	5	3
# Sentences (M)	2.9	2.3	0.8	0.1
# Tokens (M)	21.8	28.8	6.4	0.9
mBERT vocab (K)	56.6	59.0	42.3	29.3
mBERT vocab (%)	49.2	51.3	36.8	25.5

Table 3: Statistic of four low resource languages.

guages' Wikipedia. However, the mBERT vocabulary is by no means optimal for these languages. Fig. 3 shows that a large amount of the mBERT vocabulary that appears in these languages is low frequency while the language-specific SentencePiece vocabulary has a much higher frequency. In other words, the vocabulary of mBERT is not distributed uniformly.

To train the monolingual BERTs properly for low resource languages, we consider four different sets of hyperparameters. In **base**, we follow English monolingual BERT on learning vocabulary size $V = 30K$, 12 layers of transformer (base). To ensure we have a reasonable batch size for training using our GPU, we set the training sequence length to $M = 256$. Since a smaller model can prevent overfitting smaller datasets, we consider 6 transformer layers (**small**). We do not change the batch size as a larger batch is observed to improve performance (Liu et al., 2019). As low resource languages have small corpora, 30K vocabulary items might not be optimal. We consider **smaller vocabulary** with $V = 10K$. Finally, since in fine-tuning we only use a maximum sequence length of 128, in **smaller sequence length**, we match the fine-tuning phrase with $M = 128$. As a benefit of half the self-attention range, we can increase the batch size over 2.5 times to $N = 220$.

Tab. 4 shows the performance of monolingual BERT in four settings. The model with smaller sequence length performs best for monolingual BERT and outperforms the base model in 5 out of 8 tasks and languages combination. The model with smaller vocabulary has mixed performance in the low resource languages (mn, yo) but falls short for (relatively) higher resource languages (lv, af). Finally, the smaller model underperforms the base model in 5 out of 8 cases. In conclusion, the best way to pretrain BERT with a limited amount of computation for low resource languages is to use a smaller sequence length to allow a larger batch size. Future work could look into a smaller self-attention span with a restricted transformer (Vaswani et al., 2017) to improve training efficiency.

Despite these insights, no monolingual BERT outperforms mBERT (except Latvian POS). For higher resource languages (lv, af) we hypothesize that training longer with larger batch size could further improve the downstream performance as the cloze task dev perplexity was still improving. Fig. 4 supports this hypothesis showing downstream dev performance of lv and af improves as pretraining continues. Yet for lower resource languages (mn, yo), the cloze task dev perplexity is stuck and we began to overfit the training set. At the same time, Fig. 4 shows the downstream performance of mn fluctuates. It suggests the cloze task dev perplexity correlates with downstream performance when dev perplexity is not decreasing.

The fact that monolingual BERT underperforms mBERT on four low resource languages suggests that mBERT style multilingual training benefits low resource languages by transferring from other

Model Size	Vocabulary	Max Length	lv NER	lv POS	lv Parsing (LAS/UAS)	af NER	af POS	af Parsing (LAS/UAS)	mn NER	yo NER
Baseline										
Baseline			92.10	**96.19**	**84.47**/88.28	**94.00**	97.50	**85.69**/88.67	**76.40**	**94.00**
mBERT			**93.88**	95.69	77.78/**88.69**	93.36	**98.26**	83.18/**89.69**	64.71	80.54
Monolingual BERT (§5.2)										
base	30k	256	93.02	95.76	74.18/85.35	90.90	97.76	80.08/86.92	56.20	72.57
small	-	-	92.75	95.41	71.67/83.34	90.67	98.02	80.60/87.40	58.92	70.80
-	10k	-	92.68	95.65	73.94/85.20	89.55	97.66	79.91/86.93	41.70	80.18
-	-	128	93.38	95.57	73.21/84.53	91.84	97.87	80.83/87.59	55.91	73.45
Bilingual BERT (§5.3)			lv + lt			af + nl				
base	30k	256	93.22	96.03	74.42/85.60	91.85	97.98	81.73/88.55	n/a	n/a

Table 4: Monolingual BERT on four languages with different hyperparameters. Underscore denotes best within monolingual BERT and **bold** denotes best among all models. Monolingual BERT underperforms mBERT in most cases. "-" denotes same as base case.

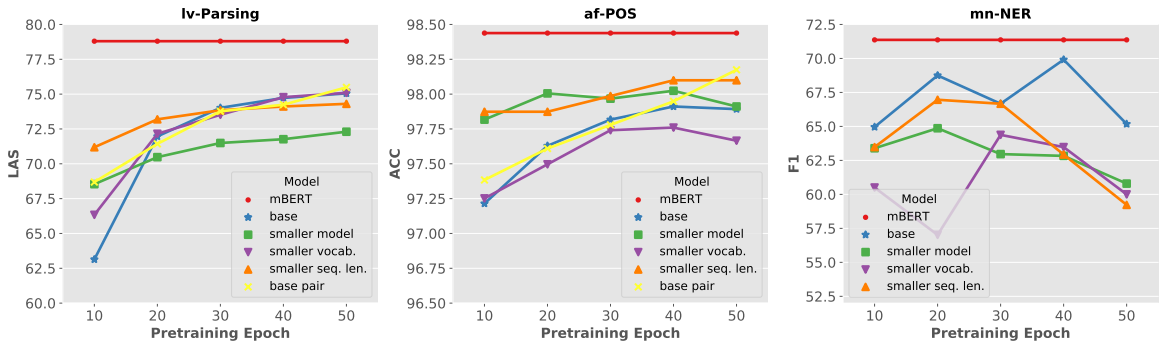

Figure 4: Dev performance with different pretraining epochs on three languages and tasks. Dev performance on higher resources languages (lv, af) improves as training continues, while lower resource languages (mn) fluctuate.

languages; monolingual training produces worse representations due to small corpus size. Additionally, the poor performance of mBERT on low resource languages does not emerge from balancing between languages. Instead, it appears that we do not have sufficient data, or the model is not sufficiently data-efficient.

5.3 mBERT vs Bilingual BERT

Finally, we consider a middle ground between monolingual training and massively multilingual training. We train a BERT model on a low resource language (lv and af) paired with a related higher resource language. We pair Lithuanian (lt) with Latvian and Dutch (nl) with Afrikaans.[5] Lithuanian has a similar size to Latvian while Dutch is over 10 times bigger. Lithuanian belong to the same Genus as Latvian while Afrikaans is a daughter language of Dutch. The **base pair** model has the same hyperparameters as the base model.

Tab. 4 shows that pairing low resource languages with closely related languages improves downstream performance. The Afrikaans-Dutch BERT improves more compared to Latvian-Lithuanian, possibly because Dutch is much larger than Afrikaans, as compared to Latvian and Lithuanian. These experiments suggest that pairing linguistically related languages can benefit representation learning and adding extra languages can further improve the performance as demonstrated by mBERT. It echos the finding of Conneau and Lample (2019) where multilingual training improves uni-directional language model perplexity for low resource languages. Concurrent work shows similar findings as the performance of low resource languages (Urdu and Swahili) improves on XNLI when more languages are trained jointly then decrease with an increasing number of languages (Conneau et al., 2019). However, they do not consider the effect of language similarity.

[5] We did not consider mn and yo since neither has a closely related language in mBERT.

6 Discussion

While mBERT covers 104 languages, the 30% languages with least pretraining resources perform worse than using no pretrained language model at all. Therefore, we caution against using mBERT alone for low resource languages. Furthermore, training a monolingual model on low resource languages does no better. Training on pairs of closely related low resource languages helps but still lags behind mBERT. On the other end of the spectrum, the highest resource languages (top 10%) are hurt by massively multilingual joint training. While mBERT has access to numerous languages, the resulting model is worse than a monolingual model when sufficient training data exists.

Developing pretrained language models for low-resource languages remains an open challenge. Future work should consider more efficient pretraining techniques, how to obtain more data for low resource languages, and how to best make use of multilingual corpora.

Acknowledgments

This research is supported in part by ODNI, IARPA, via the BETTER Program contract #2019-19051600005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Vancouver, Canada. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *arXiv preprint arXiv:1910.11856*.

Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. 2019. Cloze-driven pretraining of self-attention networks. 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 5359–5368, Hong Kong, China. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7057–7067.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Jacob Devlin. 2018. Multilingual bert readme document.

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.

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations*.

Benjamin Heinzerling and Michael Strube. 2018. BPEmb: Tokenization-free pre-trained subword embeddings in 275 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Benjamin Heinzerling and Michael Strube. 2019. Sequence tagging with contextual and non-contextual subword representations: A multilingual evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 273–291, Florence, Italy. Association for Computational Linguistics.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Haoyang Huang, Yaobo Liang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, and Ming Zhou. 2019. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. 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 2485–2494, Hong Kong, China. Association for Computational Linguistics.

Baijun Ji, Zhirui Zhang, Xiangyu Duan, Min Zhang, Boxing Chen, and Weihua Luo. 2019. Cross-lingual pre-training based transfer for zero-shot neural machine translation. *arXiv preprint arXiv:1912.01214*.

Karthikeyan K, Zihan Wang, Stephen Mayhew, and Dan Roth. 2020. Cross-lingual ability of multilingual bert: An empirical study. In *International Conference on Learning Representations*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Herv Jgou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

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*.

Phoebe Mulcaire, Jungo Kasai, and Noah A. Smith. 2019. Polyglot contextual representations improve crosslingual transfer. 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 3912–3918, Minneapolis, Minnesota. Association for Computational Linguistics.

Joakim Nivre, Manying Zhang, and Hanzhi Zhu. 2018. Universal dependencies 2.3. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics:*

Volume 2, Short Papers, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Shijie Wu, Alexis Conneau, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Emerging cross-lingual structure in pretrained language models. *arXiv preprint arXiv:1911.01464*.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. 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 833–844, Hong Kong, China. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.