When Being Unseen from mBERT is just the Beginning: Handling New Languages With Multilingual Language Models

Benjamin Muller[†] **Antonis Anastasopoulos**[‡] **Benoît Sagot**[†] **Djamé Seddah**[†]
[†]Inria, Paris, France
[‡]Department of Computer Science, George Mason University, USA
firstname.lastname@inria.fr antonis@gmu.edu

Abstract

Transfer learning based on pretraining language models on a large amount of raw data has become a new norm to reach state-of-the-art performance in NLP. Still, it remains unclear how this approach should be applied for unseen languages that are not covered by any available large-scale multilingual language model and for which only a small amount of raw data is generally available. In this work, by comparing multilingual and monolingual models, we show that such models behave in multiple ways on unseen languages. Some languages greatly benefit from transfer learning and behave similarly to closely related high resource languages whereas others apparently do not. Focusing on the latter, we show that this failure to transfer is largely related to the impact of the script used to write such languages. Transliterating those languages improves very significantly the ability of large-scale multilingual language models on downstream tasks.

1 Introduction

Language models are now a new standard to build state-of-the-art Natural Language Processing (NLP) systems. In the past year, monolingual language models have been released for more than 20 languages including Arabic, French, German, Italian, Polish, Russian, Spanish, Swedish, and Vietnamese (Antoun et al., 2020; Martin et al., 2020b; de Vries et al., 2019; Cañete et al., 2020; Kuratov and Arkhipov, 2019; Schweter, 2020, et alia). Additionally, large-scale multilingual models covering more than 100 languages are now available (XLM-R by Conneau et al. (2020) and mBERT by Devlin et al. (2019)). Still, most of the 7000+ spoken languages in the world are not covered—remaining unseen—by those models. Even languages with millions of native speakers like Sorani Kurdish (about 7 million speakers in the Middle East) or Bambara (spoken by around 5 million people in Mali and neighboring countries) are not covered by any available language models.

Even if training multilingual models that cover more languages and language varieties is tempting, the curse of multilinguality described by Conneau et al. (2020) makes it an impractical solution, as it would require to train ever larger models. Furthermore, as shown by Wu and Dredze (2020), large-scale multilingual language models reach sub-optimal performance for languages that only account for a small portion of the pretraining data.

In this paper, we describe and analyze task and language adaptation experiments to get usable language model-based representations for understudied low resource languages. We run experiments on 16 typologically diverse unseen languages on three NLP tasks with different characteristics: part-of-speech (POS) tagging, dependency parsing (DEP) and named entity recognition (NER).

Our results bring forth a great diversity of behaviors that we classify in three categories reflecting the abilities of pretrained multilingual language models to be used for low-resource languages. Some languages, the “Easy” ones, largely behave like high resource languages. Fine-tuning large-scale multilingual language models in a task-specific way leads to state-of-the-art performance. The “Intermediate” languages are harder to process as large-scale multilingual language models lead to sub-optimal performance as such. However, adapting them using unsupervised fine-tuning on available raw data in the target language leads to a significant boost in performance, reaching or extending the state of the art. Finally, the “Hard” languages are those for which large-scale multilingual models fail to provide decent downstream performance even after unsupervised adaptation.

“Hard" languages include both stable and endangered languages, but they predominantly are languages of communities that are majorly underserved by modern NLP. Hence, we direct our attention to these “Hard" languages. For those languages, we show that the script they are written in is a critical element in the transfer abilities of pretrained multilingual language models. We show that transliterating them into the script of a possibly-related high resource language leads to large gains in performance leading to outperforming non-contextual strong baselines.

To sum up, our main contributions are the following:

- Based on our empirical results, we propose a new categorization of low-resource languages that are currently not covered by any available language models: the Hard, the Intermediate and the Easy languages.
- We show that Hard languages can be better addressed by transliterating them into a better-handled script (typically Latin), providing a promising direction for rendering multilingual language models useful for a new set of unseen languages.

2 Background and Motivations

As Joshi et al. (2020) vividly illustrate, there is a great divergence in the coverage of languages by NLP technologies. The majority of the 7000+ of the world’s languages are not studied by the NLP community. Some languages have very few or no annotated datasets, making the development of systems challenging.

The development of such models is a matter of first importance for the inclusion of communities, the preservation of endangered languages and more generally to support the rise of tailored NLP ecosystems for such languages (Schmidt and Wiegand, 2017; Stecklow, 2018; Seddah et al., 2020) In that regard, the advent of the Universal Dependency project (Nivre et al., 2016) and the WikiAnn dataset (Pan et al., 2017) have greatly opened the number of covered languages by providing annotated datasets for respectively 90 languages for dependency parsing and 282 languages for Named Entity Recognition.

Regarding modeling approaches, the emergence of multilingual representation models, first with static word embeddings (Ammar et al., 2016) and then with language model-based contextual representations (Devlin et al., 2019; Conneau et al., 2020) enabled transfer from high to low-resource languages, leading to significant improvements in downstream task performance (Rahimi et al., 2019; Kondratyuk and Straka, 2019). Furthermore, in their most recent forms, multilingual models, such as mBERT, process tokens at the sub-word level using SentencePiece tokenization (Kudo and Richardson, 2018). This means that they work in an open vocabulary setting.[1] This flexibility enables such models to process any language, even those that are not part of their pretraining data.

When it comes to low-resource languages, one direction is simply to train such contextualized embedding models on whatever data is available. Another option is to adapt/finetune a multilingual pretrained model to the language of interest. We briefly discuss these two options.

Pretraining language models on a small amount of raw data Even though the amount of pre-training data seems to correlate with downstream task performance (e.g. compare BERT and RoBERTa), several attempts have shown that training a new model from scratch can be efficient even if the amount of data in that language is limited. Indeed, Suárez et al. (2020) showed that pretraining ELMo models (Peters et al., 2018) on less than 1GB of Wikipedia text leads to state-of-the-art performance while Martin et al. (2020a) showed for French that pretraining a BERT model on as few as 4GB of diverse enough data results in state-of-the-art performance. This was further confirmed by Micheli et al. (2020) who demonstrated that decent performance was achievable with as low as 100BM of raw text data.

Adapting large-scale models for low-resource languages Multilingual language models can be used directly on unseen languages, or they can also be adapted using unsupervised methods. For example, Han and Eisenstein (2019) successfully used unsupervised model adaptation of the English BERT model to Early Modern English for sequence labeling. Instead of finetuning the whole model, Pfeiffer et al. (2020) recently showed that adapter layers (Houlsby et al., 2019) can be injected into multilingual language models to provide parameter efficient task and language transfer.

Still, as of today, the availability of monolingual or multilingual language models is limited to

[1] As long as the input text is written in a script that is used in the training languages.

approximately 120 languages, leaving many languages without access to valuable NLP technology, although some are spoken by millions of people, including Bambara, Maltese and Sorani Kurdish.

What can be done for unseen languages? Unseen languages vary greatly in the amount of available data, in their script (many languages use non-Latin scripts such as Sorani Kurdish and Mingrelian), and in their morphological or syntactical properties (most largely differ from high-resource Indo-European languages). This makes the design of a methodology to build contextualized models for such languages very challenging. In this work, by experimenting with 16 typologically diverse unseen languages, (i) we show that there is a diversity of behavior depending on the script, the amount of available data, and the relation to pretraining languages; (ii) Focusing on the unseen languages that lag in performance compared to their easier-to-handle counterparts, we show that the script plays a critical role in the transfer abilities of multilingual language models. Transliterating such languages to a script which is used by a related language seen during pretraining leads to very significant improvement in downstream performance.[2]

3 Experimental Setting

We will refer to any languages that are not covered by pretrained language models as "unseen." We select a small portion of those languages within a large scope of language families and scripts. Our selection is constrained to 16 typologically diverse languages for which we have evaluation data for at least one of our three downstream tasks. Our selection includes low-resource Indo-European and Uralic languages, as well as members of the Bantu, Semitic, and Turkic families. None of these 16 languages are included in the pretraining corpora of mBERT. We report in table 1 information about their scripts, language families, and amount of raw data available.

3.1 Raw Data

To perform pretraining and fine-tuning on monolingual data, we use the deduplicated datasets from the OSCAR project (Ortiz Suárez et al., 2019). OSCAR is a corpus extracted from a Common Crawl Web snapshot.[3] It provides a significant amount of data for all the *unseen* languages we work with, except for Narabizi, Naija and Faroese, for which we use data respectively collected by Seddah et al. (2020), Caron et al. (2019) and Biemann et al. (2007), as well as for Buryat, Meadow Mari, Erzya and Livvi for which we use Wikipedia dumps.

Language (iso)	Script	Family	#sents	source
Bambara (bm)	Latin	Niger-Congo	1k	OSCAR
Wolof (wo)	Latin	Niger-Congo	10k	OSCAR
Swiss* (gsw)	Latin	West Germanic	250k	OSCAR
Naija (pcm)	Latin	Pidgin (En)	237k	Other
Faroese (fao)	Latin	North Germanic	297k	Leipzig
Maltese (mlt)	Latin	Semitic	50k	OSCAR
Narabizi (nrz)	Latin	Semitic**	87k	Other
Sorani (ckb)	Arabic	Indo-Iranian	380k	OSCAR
Uyghur (ug)	Arabic	Turkic	105k	OSCAR
Sindhi (sd)	Arabic	Indo-Aryan	375k	OSCAR
Mingrelian (xmf)	Georg.	Kartvelian	29k	Wiki
Buryat (bxu)	Cyrillic	Mongolic	7k	Wiki
Mari (mhr)	Cyrillic	Uralic	58k	Wiki
Erzya (myv)	Cyrillic	Uralic	20k	Wiki
Livvi (olo)	Latin	Uralic	9.4k	Wiki

Table 1: Unseen Languages used for downstream experiments. #sents indicates the number of raw sentences used for MLM-TUNING
*short for Swiss German **code-mixed with French *

3.2 Language Models

In all our experiments, we pretrain and fine-tune our language models using the Transformers library (Wolf et al., 2019).

MLM from scratch The first approach we evaluate is to train a dedicated language model from scratch on the available raw data we have. To do so, we train a language-specific SentencePiece tokenizer (Kudo and Richardson, 2018) before training a Masked-Language Model (MLM) using the RoBERTa (base) architecture and objective functions (Liu et al., 2019). As we work with significantly smaller pretraining sets than in the original setting, we reduce the number of layers to 6 layers in place of the original 12 layers.

Multilingual Language Models We want to assess how large-scale multilingual language models can be used and adapted to languages that are not in their pretraining corpora. We work with the multilingual version of BERT (mBERT) trained on the concatenation of Wikipedia corpora in 104 languages (Devlin et al., 2019). We also run experiments with the XLM-R base version (Conneau et al., 2020) trained on 100 languages using data extracted from the Web. As the observed behav-

[2] Also see the discussion in Section §3.2 on the script distributions in mBERT.

[3] http://commoncrawl.org/

iors are very similar between both models, we report only the results using mBERT. We note that mBERT is highly biased toward Indo-Europeans languages written in the Latin script. The basic statistics of the vocabulary shows that more than 77% of the vocabulary subword types are in the Latin script, about 11.5% are in the Cyrillic script, the Arabic scripts takes up about 4%, and smaller scripts like the Georgian one only make up less than 1% of the vocabulary (with less than 1,000 subwords) (Ács, 2019).

Adapting Multilingual Language Models to unseen languages with MLM-TUNING Following previous work (Han and Eisenstein, 2019; Wang et al., 2019; Pfeiffer et al., 2020), we adapt large-scale multilingual models by fine-tuning them with their Mask-Language-Model objective directly on the available raw data in the unseen target language. We refer to this process as MLM-TUNING. We will refer to a MLM-tuned mBERT model as mBERT+MLM.

3.3 Downstream Tasks

We perform experiments on POS tagging, Dependency Parsing (DEP), and Name Entity Recognition (NER). We use annotated data from the Universal Dependency project (Nivre et al., 2016) for POS tagging and parsing, and the WikiAnn dataset (Pan et al., 2017) for NER. For POS tagging and NER, we append a linear classifier layer on top of the language model. For parsing, following Kondratyuk and Straka (2019), we append a Bi-Affine Graph prediction layer (Dozat and Manning, 2016). For each task, following Devlin et al. (2019), we only back-propagate through the first token of each word. We refer to the process of fine-tuning a language model in a task-specific way as TASK-TUNING.

3.4 Optimization

For all pretraining and fine-tuning runs, we use the Adam optimizer (Kingma and Ba, 2014). For fine-tuning, we select the hyperparameters that minmize the loss on the validation set. The reported results are the average score of 5 runs with different random seeds computed on the test split of each dataset. We report details about the hyperparameters for TASK-TUNING in the Appendix in Table 12 and about pretraining and MLM-TUNING in Table 13.

3.5 Dataset Splits

For each task and language, we use the provided training, validation and test dataset split except for the ones that have less than 500 training sentences. In this case, we concatenate the training and test set and perform 8-folds cross-Validation and use the validation set for early stopping. If no validation set is available, we isolate one of the folds for validation and report the test scores as the average of the other folds. This enables uses to run training on at least 500 sentences in all our experiments (except for Swiss German for which we only have 100 training examples) and reduce the impact of the annotated dataset size on our analysis. As doing cross-validation results in training on very limited number of examples, we refer to training in this cross-validation setting as *few-shot* learning.

3.6 Non-contextual Baselines

For parsing and POS tagging, we use the UDPipe future system (Straka, 2018) as our baseline. This model is a LSTM-based (Hochreiter and Schmidhuber, 1997) recurrent architecture trained using pretrained static word embedding (Mikolov et al., 2013) (hence our non-contextual characterization) along with character-level embeddings. This system was ranked in the very first positions for parsing and tagging in the CoNLL shared task 2018 (Zeman and Hajič, 2018). For NER we use the LSTM-CRF model similar with character and word level embedding based on Qi et al. (2020) implementation.

4 The Three Categories of Unseen Languages

For each unseen language, we experiment with our three modeling approaches: (a) Training a language model from scratch on the available raw data and then fine-tuning it on any available annotated data in the target language for each task. (b) Fine-tuning mBERT with TASK-TUNING directly on the target language. (c) Finally, adapting mBERT to the unseen language using MLM-TUNING before fine-tuning it in a supervised way on the target task and language. We then compare all these experiments to our non-contextual baselines. By doing so we can assess if language models are a practical solution to handle each unseen language.

Interestingly we find a great diversity of behaviors across languages regarding those language model training techniques. As summarized in

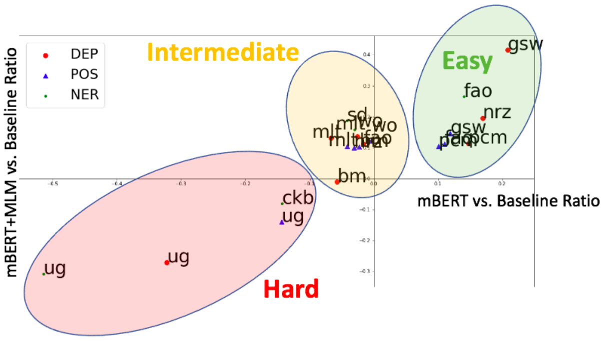

Figure 1: Visualizing our Typology of Unseen Languages. X,Y positions are computed for each language as follows: X = f(mBERT), Y = f(mBERT+MLM) with f(x) = $\frac{x-Baseline}{Baseline}$ Easy Languages are the ones on which mBERT works without MLM-TUNING, the Intermediate languages are the ones that require MLM-TUNING while the Hard languages are the ones for which mBERT does not work

Figure 1, we observe three clear clusters of languages. The first cluster, which we dub "Easy", corresponds to the languages that do not require extra MLM-fine-tuning for mBERT to achieve good performance. mBERT has the modeling abilities to process such languages without a large amount of raw data and can outperform strong non-contextual baselines as such. In the second cluster, the "Intermediate" languages require MLM-fine-tuning. mBERT is not able to beat strong non-contextual baselines using only TASK-TUNING, but MLM-TUNING enables it to do so. Finally, Hard languages are those on which mBERT fails to deliver any decent performance even after MLM- and task-fine-tuning. mBERT simply does not have the capacity to learn and process such languages.

In this section, we present in detail our results in each of these language clusters and provide insights into their linguistic properties.

4.1 Easy

Easy languages are the one one which mBERT delivers high performance out-of-the-box, compared to strong baselines. We find that those languages match two conditions:

- They are closely related to languages used during MLM pre-training
- These languages use the same script as such closely related languages.[4]

Such languages benefit from multilingual models, as cross-lingual transfer is easy to achieve and hence quite effective. In practice, one can obtain very high performance even in zero-shot settings for such languages, by performing task-tuning on related languages.

	Model	UPOS	LAS	NER
Zero-Shot				
(1)	FaroeseBERT	66.4	35.8	-
(2)	mBERT	79.4	67.5	-
(3)	mBERT +MLM	**83.4**	**67.8**	-
Few-Shot (CV with around 500 instances)				
(4)	Baseline	95.36	83.02	50.11
(5)	FaroeseBERT	91.12	67.66	39.3
(6)	mBERT	96.31	84.02	52.1
(6)	mBERT +MLM	**96.52**	**86.41**	**58.3**

Table 2: Faroese is an "easy" unseen language: a multilingual model (+ language-specific MLM) easily outperforms all baselines. Zero-shot performance, after task-tuning only on related languages (Danish, Norwegian, Swedish) is also high.

Perhaps the best example of such an "easy" setting is Faroese. mBERT has been trained on several languages of the north Germanic genus of the Indo-European language family, all of which use the Latin script. As a result, the multilingual mBERT model performs much better than the monolingual FaroeseBERT model that we trained on the available Faroese text (cf rows 1–2 and 5–6 in Table 2). Finetuning mBERT on the Faroese text is even more effective (rows 3 and 6 in Table 2), leading to further improvements, reaching more than 96.5% POS-tagging accuracy, 86% LAS for dependency parsing, and 58% NER F1 in the few-shot setting, surpassing the non-contextual baseline. In fact, even in zero-shot conditions, where we task-tune only on related languages (Danish, Norwegian, and Swedish), the model achieves remarkable performance of over 83% POS-tagging accuracy and 67.8% LAS dependency parsing.

Swiss German is another example of a language for which one can easily adapt a multilingual model and obtain good performance even in zero-shot settings. As in Faroese, simple MLM fine-tuning of the mBERT model with 200K sentences leads to an improvement of more than 25 points in both POS tagging and dependency parsing (Table 3) in zero-shot settings, with similar improvement trends

[4] The second condition can somewhat reliably be rephrased to "these languages use the Latin script," if we take the Europe-centricity of data availability and model creating as granted.

	Model	UPOS	LAS
Zero-Shot			
(1)	SwissGermanBERT	64.7	30.0
(2)	mBERT	62.7	41.2
(3)	mBERT +MLM	**87.9**	**69.6**
Few-Shot (CV with around 100 instances)			
(4)	Baseline	75.22	32.18
(5)	SwissGermanBERT	65.42	30.0
(6)	mBERT	76.66	41.2
(7)	mBERT +MLM	**78.68**	**69.6**

Table 3: Swiss German is an "easy" unseen language: a multilingual model (+ language-specific MLM) outperforms all baselines in both zero-shot (task-tuning on the related High German) and few-shot settings.

in the few-shot setting.

The potential of similar-language pre-training along with script similarity is also showcased in the case of Naija (also known as Nigerian English or Nigerian Pidgin), an English creole spoken by millions in Nigeria. As Table 4 shows, with results after language- and task-tuning on 6K training examples, the multilingual approach surpasses the monolingual baseline.

Model	UPOS	LAS
NaijaBERT	87.1	63.02
mBERT	89.3	**71.6**
mBERT +MLM	**89.6**	69.2

Table 4: Performance on Naija, an English creole, is very high, so we also classify it as an "easy" unseen language.

On a side note, we can rely on the results of Han and Eisenstein (2019) to also classify Early Modern English as an easy language. Similarly, the work of Chau et al. (2020) allows us to also classify Singlish (Singaporean English) as an easy language. In both cases, these two languages are technically unseen by mBERT, but the fact that they are variants of English allows them to be easily handled by mBERT.

4.2 Intermediate

The second type of languages (which we dub "Intermediate") are generally harder to process with pretrained multilingual language models out-of-the-box. In particular, pretrained multilingual language models are typically outperformed by a non-contextual strong baselines. Still, MLM-TUNING has an important impact and leads to usable state-of-the-art models.

A good example of such an intermediate language is Maltese, a member of the Semitic language but using the Latin script, Maltese has not been seen by mBERT. Other Semitic languages though, namely Arabic and Hebrew, have been included in the pre-training languages. The results on Maltese are outlined in Table 5, where it is clear that the non-contextual baseline outperforms mBERT. Additionally, a monolingual MLM trained on only 50K sentences matchs mBERT performance for both NER and POS tagging. However, the best results are reached with MLM-TUNING: the proper use of monolingual data and the advantage of similarity to other pre-training languages render Maltese a tackle-able language by outperforming significantly our strong non-contextual baseline.

Model	UPOS	LAS	NER
Baseline	95.99	79.71	65.1
MalteseBERT	92.1	66.5	62.5
mBERT	92.0	74.4	61.2
mBERT +MLM	**96.4**	**82.2**	**66.7**

Table 5: Maltese is an "Intermediate" unseen language: a multilingual model requires language-specific MLM and task-tuning to achieve performance competitive to a monolingual baseline.

Our Maltese dependency parsing results are in line with those of Chau et al. (2020), who also show that MLM-TUNING leads to significant improvements. They also additionally show that a small vocabulary transformation allows finetuning to be even more effective and gain 0.8 LAS points more. We further discuss the vocabulary adaptation technique of (Chau et al., 2020) in Section §6.

We consider Narabizi, an Arabic dialect spoken in North-Africa written in the Latin script and code-mixed with French, to fall in the same "Intermediate" category, because it follows the same pattern. Our results in Narabizi are listed in Table 6. For both POS tagging and parsing, the multilingual models outperform the monolingual NarabiziBERT. In addition, MLM-TUNING leads to significant improvements over the non-language-tuned mBERT baseline, also outperforming the non-contextual dependency parsing baseline.

Model	UPOS	LAS
Baseline	**84.20**	52.84
NarabiziBERT	71.3	52.8
mBERT	81.6	56.5
mBERT +MLM	**84.22**	**57.8**

Table 6: Narabizi falls into the Intermediate category. MLM-TUNING enables mBERT to match or outperform strong non-contextual baselines

We also categorize Bambara, a Niger-Congo Bantu language spoken in Mali and surrounding countries, as Intermediate, relying mostly on the POS tagging results which follow similar patterns as Maltese and Narabizi (see Table 7). We note that the BambaraBERT that we trained achieves notably poor performance compared to the non-conctextual baseline, a fact we attribute to the extremely low amount of available data (1000 sentences only). We also note that the non-contextual baseline is the best performing model for dependency parsing, which could also potentially classify Bambara as a “Hard" language instead.

Model	UPOS	LAS
Baseline	**92.3**	**76.2**
BambaraBERT	78.1	46.4
mBERT	90.2	71.4
mBERT +MLM	**92.6**	**75.4**

Table 7: Bambara also falls into the Intermediate category. MLM-TUNING enables mBERT to match outperform the strong non-contextual baseline in POS tagging

Our results in Sindhi and Wolof follow the same pattern. The non-contextual baseline achieves a 44.10 F1 score. A monolingual SindhiBERT with an F1-score of 45.2 outperforms mBERT (42.3). However, MLM-TUNING achieves the highest F1-score of 47.9. For Wolof, as reported in table 8, mBERT the non-contextual baseline only after MLM-TUNING.

The importance of script We provide initial supporting evidence for our argument on the importance of having pretrained LMs on languages with similar scripts, even for generally high-resource language families.

We first focus on Uralic languages. Finnish, Estonian, and Hungarian are high-resource representatives of this language family that are typically included in multilingual LMs, also having task-tuning data available in large quantities. For several smaller Uralic languages, however, task-tuning data are generally unavailable.

Model	UPOS	LAS
Baseline	94.09	77.05
WolofBERT	88.41	52.8
mBERT	92.85	73.26
mBERT +MLM	**95.22**	**77.91**

Table 8: Wolof falls into the Intermediate category. MLM-TUNING enables mBERT to match or outperform strong non-contextual baselines

Following a similar procedure as before, we start with mBERT, perform task-tuning on Finnish and Estonian (both of which use the Latin script) and then do zero-shot experiments on Livvi, and Komi, all low-resource Uralic languages (results on the top part of Table 9). We also report results on the Finnish treebanks after task-tuning, for better comparison. The difference in performance on Livvi (which uses the Latin script) and the other languages that use the Cyrillic script is striking.

Language	UPOS	LAS
Task-tuned – Latin script		
Finnish (FTB)	93.1	77.5
Finnish (TDT)	95.0	78.9
Finnish (PUD)	96.8	83.5
Zero-Shot Experiments		
Latin script		
Livvi	72.3	40.3
Cyrillic script		
Erzya	51.5	18.6
Few-Shot Experiments (CV)		
Livvi – *Latin script*		
Baseline	84.1	40.1
mBERT	83.0	36.3
mBERT +MLM	**85.5**	**42.3**
Erzya – *Cyrillic script*		
Baseline	91.1	65.1
mBERT	89.3	61.2
mBERT +MLM	**91.2**	**66.6**

Table 9: The script matters for the efficacy of cross-lingual transfer. The zero-shot performance on Livvi, which is written in the same script as the task-tuning languages (Finnish, Estonian), is almost twice as good as the performance on the Uralic languages that use the Cyrillic script.

Although they are not easy enough to be tackled in a zero-shot setting, we show that the low-resource Uralic languages fall in the "Intermediate" category, since mBERT has been trained on similar languages: a small amount of annotated data are enough to improve over mBERT using task-tuning. The results for Livvi and Erzya using 8-fold cross-validation, with each run only using around 700 training instances, are shown in Table 9. For Erzya, the multilingual model along with MLM-TUNING achieves the best performance, outperforming the non-contextual baseline by more than 1.5 point for parsing and matching its performance for POS tagging.

4.3 Hard

The last category of the hard unseen language is perhaps the most interesting one, as these languages are very hard to process. All available large-scale language models are outperformed by non-contextual baselines as well as by monolingual language models trained from scratch on the available raw data. At the same time, MLM-TUNING over the available raw data has a minimal impact on performance.

Uyghur, a Turkic language with about 10-15 million speakers in central Asia, is a prime example of a hard language for current models. In our experiments, outlined in Table 10, the non-contextual baseline outperforms all contextual variants, both monolingual and multilingual, in the POS tagging task. The monolingual UyghurBert achieves the best dependency parsing results with a LAS of 77, more than 5 points higher than mBERT, with similar trends for NER. Uyghur is also the only case where mBERT with MLM-TUNING does not improve over the unadapted mBERT on dependency parsing.

Model	UPOS	LAS	NER
Baseline	**89.97**	67.09	50.12
UyghurBERT	87.39	**77.33**	41.42
mBERT	76.98	72.26	24.32
mBERT+MLM	88.41	48.91	34.66

Table 10: Uyghur is a hard language. The non-contextual baseline outperforms all mBERT variants on POS tagging, and the UyghurBERT is best for DEP.

We attribute this discrepancy to script differences: Uyghur uses the Perso-Arabic script, when the other Turkic languages that were part of mBERT pre-training use either the Latin (e.g. Turkish) or the Cyrillic script (e.g. Kazakh).

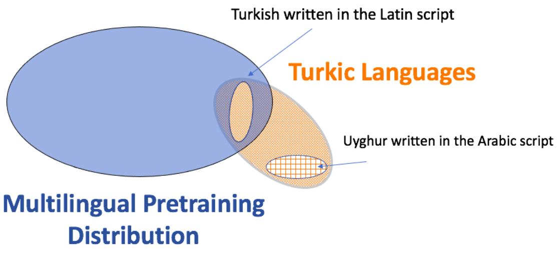
Turkish written in the Latin script
Turkic Languages
Uyghur written in the Arabic script
Multilingual Pretraining Distribution

Figure 2: An illustration of the pretraining distributions and an unseen language distribution in the case of the Turkic Language Family. Uyghur is unseen but related to Turkish which mBERT has been pretrained on. Uyghur is written in the Arabic script while Turkish is written in the Latin Script making it a great challenge for mBERT

Sorani Kurdish (also known as Central Kurdish) is a similarly hard language, mainly spoken in Iraqi Kurdistan by around 8 million speakers, which uses the Sorani alphabet, a variant of the Arabic script. We can solely evaluate on the NER task, where the non-contextual baseline is the best model, achieving a 81.3 F1-score. The SoraniBert that we trained reaches 80.6 F1-score, while mBERT gets 70.4 F1-score. MLM-TUNING on 380K sentences of Sorani texts improves mBERT performance to 75.6 F-score, but it is still lagging behind the baseline.

5 Tackling Hard Languages with Multilingual Language Models

As we have already alluded to, our hypothesis is that the script is a critical element for multilingual pretrained models to efficiently process unseen languages.

To verify this hypothesis, we assess the ability of mBERT to process an unseen language after transliterating it to another script. We focus our experiments on six languages belonging to four language families: Erzya, Bruyat and Meadow Mari (Uralic), Sorani Kurdish (Iranian, Indo-European), Uyghur (Turkic) and Mingrelian (Kartvelian). We apply the following transliteration:

- Erzya/Buryat/Mari:
  Cyrillic Script → Latin Script
- Uyghur: Arabic Script → Latin Script
- Sorani: Arabic Script → Latin Script
- Mingrelian: Georgian Script → Latin Script

Model	POS	LAS	NER	Model	NER
Uyghur (Arabic→Latin)				Sorani (Arabic→Latin)	
UyghurBERT	87.4→86.2	57.3→54.6	41.4→41.7	SoraniBERT	80.6→78.9
mBERT	77.0→87.9	45.7→65.0	24.3→35.7	mBERT	70.5→77.8
mBERT+MLM	77.3→**89.8**	48.9→**66.8**	34.7→**55.2**	mBERT+MLM	75.6→**82.7**
Buryat (Cyrillic→Latin)				Meadow Mari (Cyrillic→Latin)	
BuryatBERT	75.8→75.8	31.4→31.4	–	MariBERT	44.0→45.5
mBERT	83.9→81.6	50.3→45.8	–	mBERT	55.2→58.2
mBERT+MLM	**86.5**→84.6	**52.9**→51.9	–	mBERT+MLM	57.6→**65.9**
Erzya (Cyrillic→Latin)				Mingrelian (Georgian→Latin)	
ErzyaBERT	84.4→84.5	47.8→47.8	–	MingrelianBERT	42.0→42.2
mBERT	89.3→88.2	61.2→58.3	–	mBERT	53.6→41.8
mBERT+MLM	**91.2**→90.5	**66.6**→65.5	–	mBERT+MLM	**68.4**→62.6

Table 11: Transliterating low-resource languages into the Latin script leads to significant improvements in languages like Uyghur, Sorani, and Meadow Mari. For languages like Erzya and Buryat transliteration does not significantly influence results, while it does not help for Mingrelian. In all cases, mBERT+MLM is the best approach.

5.1 Linguistically-motivated transliteration

The strategy we used to transliterate the above-listed language is specific to the purpose of our experiments. Indeed, our goal is for the model to take advantage of the information it has learned during training on a related language written in the Latin script. The goal of our transliteration is therefore to transcribe each character in the source script, which we assume corresponds to a phoneme, into the most frequent (sometimes only) way this phoneme is rendered in the closest related language written in the Latin script, hereafter the target language. This process is not a transliteration strictly speaking, and it is needs not be reversible. It is not a phonetization either, but rather a way to render the source language in a way that maximizes the similarity between the transliterated source language and the target language.

We have manually developed transliteration scripts for Uighur and Sorani Kurdish, using respectively Turkish and Kurmanji Kurdish as target languages, only Turkish being one of the languages used to train mBERT. Note however that Turkish and Kurmanji Kurdish share a number of conventions for rendering phonemes in the Latin script (for instance, /ʃ/, rendered in English by "sh", is rendered in both languages by "ş"; as a result, the Arabic letter "ش", used in both languages, is rendered as "ş" by both our transliteration scripts). As for Erzya, Buryat and Mari, we used the readily available transliteration package *transliterate*,[5] which performs a standard transliteration.[6] We used the Russian transliteration module, as it covers the Cyrillic script. Finally, for our control experiments on Mingrelian, we used the Georgian transliteration module from the same package.

5.2 Transfer via Transliteration

We train mBERT with MLM-TUNING and TASK-TUNING on the transliterated data. As a control experiment, we also train a monolingual BERT Model from scratch on the transliterated data of each language.

Our results with and without transliteration are listed in Table 11. Transliteration for Sorani and Uyghur generally has a noticeable positive impact. For instance, transliterating Uyghur to Latin leads to an improvement of 16 points in DEP and 20 points in NER. For one of the low-resource Uralic languages, Meadow Mari, we observe an 8 F1-score points improvement on NER, while for other Uralic languages like Erzya the effect of transliteration is very minor.[7] The only case where transliterating to the Latin script leads to a drop in performance for mBERT and mBERT+MLM is Mingrelian.

[5] https://pypi.org/project/transliterate/

[6] In future work, we intend to develop dedicated transliteration scripts using the strategy described above, and to compare the results obtained with it with those described here.

[7] We got similar results for Livvi (not shown in Table 11).

We interpret our results as follows. When running MLM and task- tuning, mBERT associates the target unseen language to a set of similar languages seen during pretraining based on the script. In consequence, mBERT is not able to associate a language to its related language if they are not written in the same script. For instance, transliterating Uyghur enables mBERT to match it to Turkish, a language which accounts for a accounts for a sizeable portion pf mBERT pretraining. In the case of Mingrelian, transliteration has the opposite effect: transliterating Mingrelian in Latin is harming the performance as mBERT is not able to associate it to Georgian which is seen during pre-training and uses the Georgian script.

Our findings are generally in line with previous work. Transliteration to English specifically (Lin et al., 2016; Durrani et al., 2014) and *named entity* transliteration (Kundu et al., 2018; Grundkiewicz and Heafield, 2018) has been proven useful for successful cross-lingual transfer in tasks like NER, entity translation, or entity linking (Rijhwani et al., 2019) and morphological inflection (Murikinati et al., 2020).

The transliteration approach provides a viable path for rendering large pretrained models like mBERT useful for all languages of the world. Indeed, transliterating both Uyghur and Sorani leads to matching or outperforming the performance of non-contextual strong baselines and deliver usable models.

6 Discussion and Conclusion

Pretraining ever larger language models is a research direction that is currently receiving a lot of attention and resources from the NLP research community (Raffel et al., 2019; Brown et al., 2020). Still, a large majority of human languages are under-resourced making the development of monolingual language models very challenging in those settings. Another path is to build large scale multilingual language models.[8] However, such an approach faces the inherent zipfian structure of human languages, making the training of a single model to cover all languages an unfeasible solution (Conneau et al., 2020). Reusing large scale pretrained language models for new unseen languages seems to be a more promising and reasonable solution from a cost-efficiency and environmental perspective (Strubell et al., 2019).

Recently, Pfeiffer et al. (2020) proposed to use adapter layers (Houlsby et al., 2019) to build parameter efficient multilingual language models for unseen languages. However, this solution brings no significant improvement in the supervised setting, compared to a more simple Masked-Language Model fine-tuning. Furthermore, developing a language agnostic adaptation method is an unreasonable wish with regard to the great typological diversity of human languages.

On the other hand, the promising vocabulary adaptation technique of (Chau et al., 2020) which leads to good dependency parsing results on unseen languages when combined with task-tuning has so far been tested only on Latin script languages (Singlish and Maltese). We expect that it will be orthogonal to our transliteration approach, but we leave for future work the study of its applicability and efficacy on more languages and tasks.

In this context, we bring empirical evidence to assess the efficiency of language models pretraining and adaptation methods on 16 low-resource and typologically diverse unseen languages. Our results show that the "Hard" languages are currently out-of-the-scope of any currently available language models and are therefore left outside of the current NLP progress. By focusing on those, we find that this challenge is mostly due to the script. Transliterating them to a script that is used by a related higher resource language on which the language model has been pretrained on leads to large improvements in downstream task performance. Our results shed some new light on the importance of the script in multilingual pretrained models. While previous work suggests that multilingual language models could transfer efficiently across scripts in zero-shot settings (Pires et al., 2019; K et al., 2020), our results show that such cross-script transfer is possible only if the model has seen related languages in the same script during pretraining.

Our work paves the way for a better understanding of the mechanics at play in cross-language transfer learning in low-resource scenarios. We strongly believe that our method could contribute to bootstrapping NLP resources and tools for low-resource languages, thereby favoring the emergence of NLP ecosystems for languages that are currently under-served by the NLP community.

[8] Even though we explore a different research direction, we do acknowledge recent advances in small scale and domain specific language models (Micheli et al., 2020) which suggest such models could also have an important impact for those languages.

Acknowledgments

The Inria authors were partly funded by two French Research National agency projects, namely projects PARSITI (ANR-16-CE33-0021) and SoSweet (ANR-15-CE38-0011), as well as by Benoit Sagot's chair in the PRAIRIE institute as part of the "Investissements d'avenir" programme under the reference ANR-19-P3IA-0001. Antonis thanks Graham Neubig for very insightful initial discussions on this research direction.

References

Judit Ács. 2019. Exploring bert's vocabulary. Http://juditacs.github.io/2019/02/19/bert-tokenization-stats.html.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith. 2016. Massively multilingual word embeddings. arXiv:1602.01925.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. arXiv:2003.00104.

Chris Biemann, Gerhard Heyer, Uwe Quasthoff, and Matthias Richter. 2007. The leipzig corpora collection-monolingual corpora of standard size. *Proceedings of Corpus Linguistic*, 2007.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. arXiv:2005.14165.

Bernard Caron, Marine Courtin, Kim Gerdes, and Sylvain Kahane. 2019. A surface-syntactic ud treebank for naija. In *Proceedings of the 18th International Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2019)*, pages 13–24.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*.

Ethan C Chau, Lucy H Lin, and Noah A Smith. 2020. Parsing with multilingual bert, a small corpus, and a small treebank. arXiv:2009.14124.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. arXiv:1611.01734.

Nadir Durrani, Hassan Sajjad, Hieu Hoang, and Philipp Koehn. 2014. Integrating an unsupervised transliteration model into statistical machine translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 148–153, Gothenburg, Sweden. Association for Computational Linguistics.

Roman Grundkiewicz and Kenneth Heafield. 2018. Neural machine translation techniques for named entity transliteration. In *Proceedings of the Seventh Named Entities Workshop*, pages 89–94, Melbourne, Australia. Association for Computational Linguistics.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. 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 4238–4248, Hong Kong, China. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. arXiv:1902.00751.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

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:1412.6980.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. 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 2779–2795, Hong Kong, China. 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.

Soumyadeep Kundu, Sayantan Paul, and Santanu Pal. 2018. A deep learning based approach to transliteration. In *Proceedings of the Seventh Named Entities Workshop*, pages 79–83, Melbourne, Australia. Association for Computational Linguistics.

Yuri Kuratov and Mikhail Arkhipov. 2019. Adaptation of deep bidirectional multilingual transformers for russian language. *CoRR*, abs/1905.07213.

Ying Lin, Xiaoman Pan, Aliya Deri, Heng Ji, and Kevin Knight. 2016. Leveraging entity linking and related language projection to improve name transliteration. In *Proceedings of the Sixth Named Entity Workshop*, pages 1–10, Berlin, Germany. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. arXiv:1907.11692.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020a. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020b. Camembert: a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Vincent Micheli, Martin d'Hoffschmidt, and François Fleuret. 2020. On the importance of pre-training data volume for compact language models.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Nikitha Murikinati, Antonios Anastasopoulos, and Graham Neubig. 2020. Transliteration for cross-lingual morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 189–197, Online. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo

Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. In *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019*, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

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.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. Mad-x: An adapter-based framework for multi-task cross-lingual transfer. arXiv:2005.00052.

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.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. arXiv:2003.07082.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv:1910.10683.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Shruti Rijhwani, Jiateng Xie, Graham Neubig, and Jaime Carbonell. 2019. Zero-shot neural transfer for cross-lingual entity linking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6924–6931.

Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10, Valencia, Spain. Association for Computational Linguistics.

Stefan Schweter. 2020. Berturk - bert models for turkish.

Djamé Seddah, Farah Essaidi, Amal Fethi, Matthieu Futeral, Benjamin Muller, Pedro Javier Ortiz Suárez, Benoît Sagot, and Abhishek Srivastava. 2020. Building a user-generated content North-African Arabizi treebank: Tackling hell. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1139–1150, Online. Association for Computational Linguistics.

Steve Stecklow. 2018. Why Facebook is losing the war on hate speech in Myanmar, Reuters. `https://www.reuters.com/investigates/special-report/myanmar-facebook-hate/`.

Milan Straka. 2018. Udpipe 2.0 prototype at conll 2018 ud shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207.

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

Pedro Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. arXiv:2006.06202.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. Bertje: A dutch BERT model. *CoRR*, abs/1912.09582.

Zihan Wang, Stephen Mayhew, Dan Roth, et al. 2019. Cross-lingual ability of multilingual bert: An empirical study. arXiv:1912.07840.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual bert? *arXiv preprint arXiv:2005.09093*.

Daniel Zeman and Jan Hajič, editors. 2018. *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, Brussels, Belgium.

7 Appendices

7.1 Reproducibility

7.1.1 Infrastructure

Our experiments were ran on a shared cluster on the equivalent of 15 Nvidia Tesla T4 GPUs.[9]

7.1.2 Data Sources

We base our experiments on data originated from two sources. The Universal Dependency project (Nivre et al., 2016) downloadable here `https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2988` and the WikiNER dataset (Pan et al., 2017). We also make use of the CoNLL-2003 shared task NER English dataset `https://www.clips.uantwerpen.be/conll2003/`

7.1.3 Optimization

Params.	Parsing	NER	POS	Bounds
batch size	32	16	16	[1,256]
learning rate	5e-5	3.5e-5	5e-5	[1e-6,1e-3]
epochs (best)	15	6	6	[1,+ inf]
#grid	60	60	180	-
Run-time (min)	32	24	75	-

Table 12: Fine-tuning best hyper-parameters for each task as selected on the validation set with bounds. #grid: number of grid search trial. Run-time is reported in average for training and evaluation.

Parameter	Value
batch size	64
learning rate	5e-5
optimizer	Adam
warmup	linear
warmup steps	10% total
epochs (best of)	10

Table 13: Unsupervised fine-tuning hyper-parameters

[9] https://www.nvidia.com/en-sg/data-center/tesla-t4/