GlotScript: A Resource and Tool for Low Resource Writing System Identification

Amir Hossein Kargaran[*◇], **François Yvon**[†], **Hinrich Schütze**[*◇]
[*]Center for Information and Language Processing, LMU Munich, Germany
[◇]Munich Center for Machine Learning (MCML), Germany
[†]Sorbonne Université, CNRS, ISIR, France
amir@cis.lmu.de

Abstract

We present GlotScript, an open resource and tool for low resource writing system identification. GlotScript-R is a resource that provides the attested writing systems for more than 7,000 languages. It is compiled by aggregating information from existing writing system resources. GlotScript-T is a writing system identification tool that covers all 161 Unicode 15.0 scripts. For an input text, it returns its script distribution where scripts are identified by ISO 15924 codes. We also present two use cases for GlotScript. First, we demonstrate that GlotScript can help cleaning multilingual corpora such as mC4 and OSCAR. Second, we analyze the tokenization of a number of language models such as GPT-4 using GlotScript and provide insights on the coverage of low resource scripts and languages by each language model. We hope that GlotScript will become a useful resource for work on low resource languages in the NLP community. GlotScript-R and GlotScript-T are available at https://github.com/cisnlp/GlotScript.

Keywords: multilingual, low resource, natural language processing

1. Introduction

We are interested in automatically identifying the writing system or script a given text is written in. We will refer to this automatic identification of scripts as *script identification*. When doing research on and developing technology for low resource languages, script identification is useful. For example, when compiling a corpus for a low resource language, script identification can serve as part of quality control: texts written in scripts not used for the language can be excluded. Similarly, when training the tokenizer of a language model for low resource languages, an analysis of the learned token vocabulary allows developers to see how well a script is represented, an indication of how well languages written in that script are represented.

In such low resource scenarios, language identification is an alternative to script identification: language identification can also be used for quality control and for the analysis of language model vocabularies. However, language identification for low resource languages is prone to high error rates (Kargaran et al., 2023; Kreutzer et al., 2022; Caswell et al., 2020). Many low resource languages are poorly identified by existing tools, due to data scarcity and high variability in orthography, genres and domains. By contrast, script identification can be performed with a much higher accuracy and it is therefore a useful functionality in the absence of reliable language identification for many low resource languages.

In this paper, we present GlotScript, a resource and tool for low resource identification of writing systems, i.e., low resource script identification.

Our contributions are as follows. (i) We compile and organize GlotScript-R, a comprehensive resource for script identification, associating attested writing systems with language varieties. We make this resource available to the community. (ii) We publish GlotScript-T, a tool for script identification which covers all 161 scripts in Unicode 15.0. It computes the script distribution for any input text. Scripts are identified by their ISO 15924 codes. To the best of our knowledge, no such tool is currently available. (iii) We demonstrate the benefits of GlotScript-T and GlotScript-R for corpus cleaning, and show that the quality of existing low resource corpora can be improved using script identification. (iv) We analyze the tokenization of large language models (LLMs) – including GPT-4, Falcon and Llama2 – using GlotScript-T. This analysis gives valuable insights regarding LLM coverage (or lack of coverage) of low resource languages.

2. Background and Related work

2.1. Script Identification

The Stops library (Andrews et al., 2022), part of the NLLB project (NLLB Team et al., 2022), is capable of detecting the script of a given text in 38 scripts based on ISO 15924. It uses the Unicode blocks defined for each script.

Ács (2019) gathered Unicode data block ranges and mapped them to 18 macro Unicodes. For instance, they categorized ranges like "Basic Latin", "C1 Controls and Latin-1 Supplement", "Latin Extended Additional", "Latin Extended-A", and "Latin Extended-B" into the Latin script. These ranges were then employed with regular expressions to identify the script of an input text.

These methods do not cover all of the 161 scripts that Unicode 15.0 defines. Additionally, since they use entire blocks,[1] the result may not be entirely accurate. For example, the range from U+0000 to U+007F is part of the "Basic Latin" block. However, within this range, there are some common characters that do not belong to a specific script and can be used universally, such as the left square bracket (U+005B). Compared to blocks, using a more granular approach, in particular a per-character approach,[2] can be beneficial.

Python has the built-in library unicodedata.[3] It allows working with the Unicode Database. The command `unicodedata.name(char)` can be used to obtain the name of a character. This command only works for single characters. However, the character's name does not always include the name of its script. Even if the name of the character contains information about its script, there is no direct and consistent correspondence of that information to the codes of the ISO 15924 standard.

2.2. Language Resources

Many existing resources that compile information about the world's languages, such as Ethnologue (Eberhard et al., 2023),[4] Glottolog (Hammarström et al., 2023) and WALS (World Atlas of Language Structures) (Dryer and Haspelmath, 2013) do not contain information about writing systems.

Our work is most closely related to the work of van Esch et al. (2022). Apart from the fact that van Esch et al. (2022) do not provide script identification software for use cases such as corpus cleaning, our approach differs in methodology. van Esch et al. (2022) also aim to establish an extensive metadata repository including writing systems and speaker details. They cover more than 2,800 languages. But their methodology heavily focuses on analyzing online texts from sources such as Wikipedia, JW.org, Crúbadán (Scannell, 2007) and PanLex (Kamholz et al., 2014). They then extend their analysis to projects like Unilex,[5] CorpusCrawler (Brawer, 2017), Bible.is and the LTI corpus for LangID (Brown, 2014). Relying on texts to determine the correct script for a language may not be a robust method, as texts collected online can be noisy or may lack accurate labels. We further discuss van Esch et al. (2022)'s work and its limitations in §3.3, including its tendency to include the Latin script for languages for which romanization is not widely used.

[1]https://unicode.org/Public/15.0.0/ucd/Blocks.txt

[2]https://unicode.org/Public/15.0.0/ucd/Scripts.txt

[3]https://docs.python.org/3/library/unicodedata.html

[4]The free access version.

[5]https://github.com/unicode-org/unilex

2.3. Applications

Corpus cleaning. One of the uses of script identification is corpus cleaning. ImaniGooghari et al. (2023) detect the script for each sentence and treat each language-script as a separate entity. They exclude all corpora for which the language-scripts are found to be incorrect or noisy; for example, when there is a mismatch between language and script, the corpus is removed.

Kreutzer et al. (2022) reported in their manual audit on multilingual datasets that languages written in scripts other than their correct ones, or text mixed with non-linguistic material, are good indicators of a corpus being of low quality.

Analysis of pre-trained models. Ács (2019) studies the mBERT (Devlin et al., 2019) tokenizer vocabulary. They compile unicode ranges into 18 categories and use these ranges with regex to detect the script of vocabulary tokens. Similar to Ács (2019), van Esch et al. (2022) analyzes the vocabulary coverage of three models: mBERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020) and mT5 (Xue et al., 2021). Our analysis covers a wider range of languages models and benefits from the alternative methodology we adopt for GlotScript.

Fonts and keyboards. One application of studying writing systems is the development of Unicode fonts, such as SIL Fonts,[6] or keyboards for devices such as Keyman.[7]

OCR post-correction. Studying writing systems also finds practical use in improving OCR (optical character recognition) systems for rare languages. Take, for instance, the case where a Cyrillic character is erroneously replaced with a visually similar Latin letter. This highlights the importance of integrating post-correction methods into OCR systems (Rijhwani et al., 2020). A simple method to detect such errors would involve script identification performed on any part of the text to determine the percentage of characters in that part belonging to the same script.

3. GlotScript-R

We now describe GlotScript-R, a resource providing writing system metadata for more than 7,000 language varieties. We identify languages based on ISO 639.[8]

[6]https://software.sil.org/fonts/

[7]https://keyman.com/

[8]https://iso639-3.sil.org/

3.1. Source Selection

We conducted an exhaustive search to identify potential sources of information about writing systems that we could use for our goals in creating GlotScript. We focused on sources known for collaborative contributions or recognized for their reliability across a wide range of languages. We summarize the result of this search as follows.

(i) **LREC_2800 metadata**. van Esch et al. (2022) compiled a database containing information on the writing systems of more than 2,800 languages under the CC BY 4.0 license, permitting modification and redistribution.[9]

(ii) **Wikipedia metadata**. Wikipedia hosts pages for each language's ISO 639 code and some of these pages include details about the language's writing system. It also contains information about writing systems that have not yet been incorporated into Unicode. However, not all pages contain metadata for writing systems. This dataset is available under the CC BY-SA 4.0 license, permitting modification and redistribution.[10]

(iii) **ScriptSource metadata.** Developed by SIL, ScriptSource is a dynamic collaborative website serving as a reference for writing systems. It gives information on which languages use which script. This dataset is available under the CC BY-SA 3.0 license, permitting modification and redistribution.[11]

(iv) **LangTag metadata.** The WSTech team of SIL offers writing system metadata for language varieties (format: JSON). This dataset is available under the MIT License, permitting modification and redistribution.[12]

(v) **Other sources.** We came across additional sources during our search, but they have limited coverage of languages. For example, the IANA language subtag registry provides script metadata for 134 languages.[13] Other sources are consulted by sources (i) – (iv), for example, Omniglot[14] in LREC_2800 and Unicode CLDR[15] in LangTag.

Note that none of these sources cover all languages, and there is a potential for some languages to have incorrect scripts listed (see §3.3). To address this, we incorporate all four sources (i) – (iv) in GlotScript-R; this allows us to give preference to script identification decisions that several sources agree upon. We gathered the Wikipedia and ScriptSource data – which are not accessible in tabular format – by crawling.

[9]https://github.com/google-research/url-nlp

[10]https://en.wikipedia.org/wiki/ISO_639:{ISO639}

[11]https://scriptsource.org/scr/{ISO15924}

[12]https://github.com/silnrsi/langtags

[13]https://iana.org/assignments/language-subtag-registry

[14]https://www.omniglot.com/writing/langalph.htm

[15]https://github.com/unicode-org/cldr-json

3.2. Preprocessing

There is a total of 8030 unique three-letter ISO 639 codes that at least one of the sources covers. The most used version of the ISO 639 code set in the NLP community is ISO 639-3; however, not all three-letter codes are part of this subset. For instance, ber (Berber languages) is part of code sets 639-2 and 639-5, but not part of 639-3. To handle this, we include all three-letter ISO codes, not just those from ISO 639-3.

The number of three-letter ISO 639 codes covered is 2836 for LREC_2800, 1726 for Wikipedia, 7875 for ScriptSource and 7901 for LangTag.

3.3. Agreement

We assess agreement between two metadata sources using Jaccard similarity:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

where A and B are sets of scripts given for an ISO code by the two sources in question. Since the Wikipedia data is of a smaller size and does not represent writing systems in a uniform format (e.g., ISO 15924), we use Wikipedia as a secondary source of information when merging information, especially in cases where there is no agreement.

Pair	$\lvert\mathcal{L}\rvert$	**CA**	**PA**	**NA**
(LangTag, LREC_2800)	2814	2385	404	25
(ScriptSource, LREC_2800)	2811	2372	414	25
(LangTag, ScriptSource)	7858	7567	287	4

Table 1: Agreement counts for each pair of sources. CA: complete agreement ($J = 1.0$), PA: partial agreement ($0 < J < 1$), NA: no agreement ($J = 0$), $|\mathcal{L}|$: number of common ISO 639 codes.

We present the results for each pair in LangTag, ScriptSource and LREC_2800 in Table 1. LangTag and ScriptSource completely agree ($J = 1.0$) for 96% of ISO codes. This is not surprising, given that both sources are from SIL. However, some disagreements still exist. Additionally, it appears that LangTag aligns more closely with LREC_2800, as it shares a greater number of ISO 639 codes, fewer partial agreements ($0 < J < 1$) and no disagreements ($J = 0$).

To understand the discrepancies between different metadata sources, we conducted a manual analysis, and observed the following trends.[16]

(i) **Rare or historic scripts.** ScriptSource and LangTag metadata tend to include rare and historic

[16]We analyse discrepancies for languages which Wikipedia data knowledge is available. Our approach involves relying on information sourced from Wikipedia and other web-based resources for each language.

scripts. For instance, in the case of Turkish (tur), alongside the primary Latin script, these sources also list Arabic, Greek and Cyrillic. In contrast, LREC_2800 exclusively lists Latin, the current official script.

(ii) **Romanized versions.** LREC_2800 often introduces a Latin version for a language, even if it is rarely used. For instance, there is a Latin entry for fas (Farsi) in LREC_2800, despite it not being the official script and not widely used, even in social networks. Scriptsource and langtag only report Arabic.[17]

(iv) **Partial information.** There are instances where each source only partially supports certain scripts. For example, ScriptSource and LangTag mention that aat (Arvanitika Albanian) uses the Greek script while LREC_2800 reports the use of the Latin script. However, resolving this conflict is difficult because there is no proper definition. There is disagreement among Arvanitika speakers about using the Greek versus the Latin script. Additionally, this language is classified as an endangered language. The writing history for this language is outlined in Sasse (1991).

(v) **Errors.** There are instances where it is clear that a language is highly unlikely to be written in a particular script. One of these cases is var (Huarijio) in LREC_2800, which is indicated to be written in Devanagari. In our judgement, this is an error since var is a Uto-Aztecan language spoken in northwestern Mexico and Devanagari is only used for South Asia languages.

3.4. Compilation

We now explain the process of compiling GlotScript-R by combining different metadata sources. As will be apparent from our discussion, creating a reliable writing system resource is not straightforward.

Two of our main desiderata are *usefulness for NLP* and *accuracy*.

As far as usefulness for NLP is concerned, if we accepted all scripts that any of the sources lists for a language, then we would include errors and scripts that in practical NLP contexts are very unlikely to be relevant. The most important instance of this is that some sources give Latin as a valid script in many cases where its use is extremely rare. Including Latin for such languages would be harmful for use cases such as corpus quality control. For example, a non-Farsi subcorpus written in Latin cannot be excluded using script identification if we accept Latin as a standard script for Farsi.

On the other hand, the desideratum of accuracy demands that we do not simply adopt a criterion of perfect agreement of the four sources. Such a heuristic would exclude important language metadata that might be useful to the NLP community.

To allow users of GlotScript-T to trade off usefulness against accuracy, we define two metadata categories: CORE and AUXILIARY. The CORE metadata give the primary scripts based on consensus among the metadata sources. The AUXILIARY metadata give secondary scripts, those that are only specified as admissible by a single source.

Given the 96% complete agreement between LangTag and ScriptSource, we prioritize resolving disagreements between these two sources using information from Wikipedia and LREC_2800. We merge LangTag and ScriptSource as one consolidated group named SIL, which is the aggregation of LangTag and ScriptSource if they match or if the discrepancies can be resolved based on additional resources. Only those discrepancies that cannot be resolved this way are collected in SIL2-aux.

As a result of this consolidation, we have now three metadata sources: SIL, LREC_2800 and Wikipedia. Given a language l identified by an ISO 639 code, we categorize a script for l as CORE if this is supported by at least two of the three sources (e.g., the CORE metadata specify Kpel as one of the admissible scripts for kpe (Kpelle) since SIL and Wikipedia agree on it even though LREC_2800 does not) or if only one of three sources provides information about admissible scripts for l.

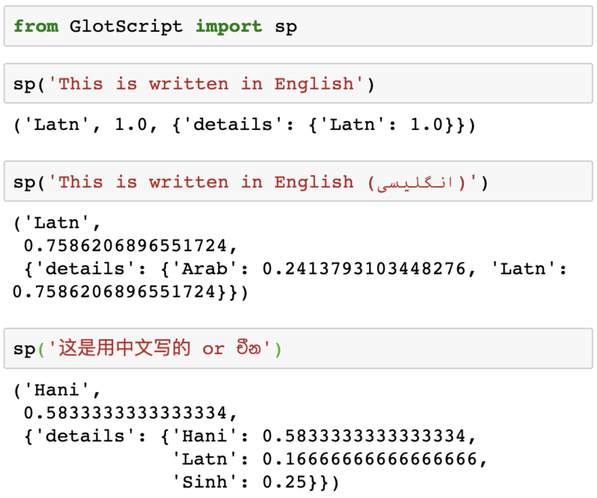
(i) In cases of partial information, such as for aat (Arvanitika Albanian), where both LREC_2800 and Wikipedia agree on Latin, and both Wikipedia and SIL agree on Greek, we include both Latin and Greek in CORE.

(ii) If only one metadata source reports a script and not the others, the script is placed in the auxiliary category specific to that source. Wiki-aux, LREC2800-aux, and SIL-aux are used for Wikipedia, LREC_2800, and SIL, respectively. SIL2-aux is exclusively used for discrepancies between ScriptSource and LangTag.

4. GlotScript-T

We now describe GlotScript-T, an open-source Python tool that identifies the writing systems of input text. It supports the 161 scripts in Unicode 15.0, identified as ISO 15924 codes. GlotScript-T is the first tool to provide labels based on ISO 15924 with this level of coverage. Figure 1 gives an example of how to use GlotScript-T.

[17]Ultimately, the meaning of “widely used” depends on the intended application. If data entered on smartphones is the primary interest of a company, then Latin may actually be not infrequent because text messages may be entered in the Latin script, for example, by Farsi speakers in countries where the Latin script dominates.



```
from GlotScript import sp
```

```
sp('This is written in English')
```

```
('Latn', 1.0, {'details': {'Latn': 1.0}})
```

```
sp('This is written in English (انگلیسی)')
```

```
('Latn',
 0.7586206896551724,
 {'details': {'Arab': 0.2413793103448276, 'Latn':
0.7586206896551724}})
```

```
sp('这是用中文写的 or චීන')
```

```
('Hani',
 0.5833333333333334,
 {'details': {'Hani': 0.5833333333333334,
              'Latn': 0.16666666666666666,
              'Sinh': 0.25}})
```

Figure 1: How to use GlotScript-T: three examples. GlotScript-T returns a tuple consisting of the main script, the percentage of characters in the main script and detailed information on the distribution of scripts.

4.1. Development

We first sorted unicode ranges into different script categories, based on the Unicode Character Database.[18] We then matched these ranges with ISO 15924 code names from Wikipedia.[19]

For an input text, GlotScript-T identifies the unicode range of each character, maps it to an ISO 15924 code and then calculates the percentage of each script. GlotScript-T returns the main script (the one that most characters belong to) and detailed information on the distribution of scripts.

4.1.1. Special Codes

GlotScript-T also uses three special codes that are not proper scripts.

(i) **Zzzz.** This code is used for unknown Unicode ranges. We also add the replacement character (U+FFFD �) as Zzzz.

(ii) **Zinh.** This code is assigned to a character who inherits its script from the previous character. For example, the zero width joiner character (U+200D) is used for joining characters. It does not belong to any script, but rather inherits its script code from the immediately preceding character.

(iii) **Zyyy.** This is the ISO 15924 code for undetermined script. This script code covers characters like punctuation, symbols, mathematical notation and musical notation that are used across many different scripts.

[18]https://unicode.org/Public/15.0.0/ucd/Scripts.txt
[19]https://en.wikipedia.org/wiki/ISO_15924

4.1.2. Efficiency

We randomly generate a test set of 1 million sentences, each with a length of 100, using characters from different Unicode ranges. The walltime of processing this test set with GlotScript-T on a single core of an Intel Xeon E7-8857 3GHz CPU is 80.790 seconds, i.e., about 8×10^{-5} seconds per sentence.

5. Experimental Setup

We present experiments for two tasks to demonstrate the usefulness of GlotScript-T and GlotScript-R.

(i) **Corpus quality assessment.** We investigate multilingual datasets by determining if a text assigned by the corpus metadata to a particular language is written in a script that is admissible for that language. If this is not the case for a particular text, it hints at mislabeling and suggests that the text most likely belongs to another language or is noise. This part of our experiments highlights the benefits of a script identification tool for creating high-quality corpora for low resource languages.

(ii) **Multilingual models.** We quantify the presence of each script within the vocabulary of several multilingual language models, focusing on large multilingual language models. We evaluate the level of representation of each script, which sheds light on the quality of representations of languages using that script.

5.1. Corpus Quality Assessment

GlotScript-R lists for each language l, identified by an ISO 639 code, the scripts that are commonly used for l. Recall that, as shown in Figure 1, the function $sp(s)$ provided by GlotScript-T computes the percentage of each script in the input and identifies the main script.

Let s be an input sentence from a corpus that is assigned the ISO 639 code l by the corpus metadata. The predicted main script for s – i.e., $sp(s)$ – is either one of the admissible scripts (according to GlotScript-R) for l. We call this a match. Or it is not one of the admissible scripts. We call this a mismatch. In case we find a mismatch for s, we evaluate this as an error. We refer to this heuristic as the **script mismatch rule**. We determine for each sentence of the corpus whether it is a match or a mismatch and then report the proportion of errors.

5.1.1. Evaluation Corpora

We select two corpora that have been recognized in multiple past studies for their multilinguality and the inclusion of lower resource languages.

(i) Multilingual C4 (mC4) (Xue et al., 2021) is a document-level dataset used for training the mT5 language model. It uses CLD3 (Botha et al., 2017; Salcianu et al., 2018) language identification (LID). CLD3 supports 107 languages. Accordingly, mC4 contains monolingual texts in 107 languages.

(ii) OSCAR 22.01 (Ortiz Suárez et al., 2019; Abadji et al., 2021) is a set of monolingual corpora for 151 languages. It is deduplicated and uses Fasttext (Joulin et al., 2017) FT176[20] LID on a line by line level.

Both corpora are sourced from CommonCrawl. Kreutzer et al. (2022) performed a manual audit on 100 sentences (or less) per language for these two corpora.

5.1.2. Setup

We load both datasets using the Hugging Face API.[21] Each row in the dataset is split by `\n` (which we consider to be the sentence delimiter) and deduplicated.

For both corpora, we randomly select 1000 sentences per language. We exclude languages for which there are fewer than 1000 sentences available, resulting in a coverage of 118 languages from OSCAR 22.01. For example, for dsb (Lower Sorbian), diq (Dimli/Zaza) and eml (Emiliano-Romagnolo), there is only one sentence each available in OSCAR 22.01, so we exclude these languages. We map the language identifiers provided by the corpus metadata to three-letter ISO 639 codes.

We apply GlotScript-T to the 1000-sentence subsets per language and obtain the main script for each sentence. We apply the script mismatch rule to identify the sentence as correct or incorrect. However, if the corpus metadata specify the script in addition to the language (e.g., bg-Latn), then we only consider the script given as a candidate script for that sentence by the metadata (e.g., we only consider Latn for bg-Latn).

5.2. Multilingual Models

We analyze the representation of common writing systems in state-of-the-art pretrained models. Most of these models are claimed to be highly multilingual. We approach this analysis employing the following two methods.

(i) Following (van Esch et al., 2022; Ács, 2019), we examine the writing systems present in the vocabulary of each model's tokenizer.

(ii) We tokenize the multi-parallel corpora of UDHR[22] using each model's tokenizer. For each writing system, we then measure the number of tokens generated and the percentage of unknown tokens (UNK) generated. Similar experiments are also conducted by Ahia et al. (2023) and Petrov et al. (2024) on FLORES-200 (Goyal et al., 2021; NLLB Team et al., 2022). UDHR dataset supports a greater variety of languages and scripts compared to FLORES-200.

5.2.1. Model Selection

We select ten state-of-the-art models for their multilingual capabilities or for their frequent use: GPT-4 (OpenAI, 2023), Falcon (Penedo et al., 2023), Llama 2 (Touvron et al., 2023), BLOOM (Scao et al., 2022), Glot500 (ImaniGooghari et al., 2023), XLM-R (Conneau et al., 2020), mBERT and BERT (Devlin et al., 2019), mT5 (Xue et al., 2021) and NLLB (NLLB Team et al., 2022).

5.2.2. UDHR

UDHR consists of more than 500 translations of the Universal Declaration of Human Rights, each containing 30 short articles. We remove all the translations that are incomplete (fewer than 89 sentences) or noisy (e.g., lines consisting of the single English word 'missing'). We ensure that all 30 articles are available in a translation and that it has a valid ISO 639-3 code (not undetermined). In cases where multiple versions are available for a pair of ISO 639-3 and ISO 15924, we make a random selection. This procedure selects a subset of UDHR that covers 396 different language-scripts.

6. Results and Analysis

6.1. Corpus Quality Assessment

Table 2 reports the five top and bottom languages (in terms of inferred accuracy of their metadata) for each corpus, along with correct and incorrect scripts. Scripts highlighted in green are deemed correct based on GlotScript-R CORE. yellow represents the scripts that were returned for AUXILIARY. Note that quite a few languages have Latin as an AUXILIARY script, based on the LREC_2800 metadata. The ACC column displays the accuracy of the correct script based only on CORE.

For the 118 selected languages in OSCAR, we obtain an average script accuracy of 0.947. For the 107 languages in mC4, the average score is 0.917. These averages are high, indicating a favorable quality overall. However, when examining the bottom five languages with the lowest correct script scores, the average drops to 0.823 for OSCAR and 0.566 for mC4.

Based on our audit of common errors in the OSCAR corpus, we can confirm that incorrect Latin

[20] https://fasttext.cc/docs/en/language-identification.html
[21] https://huggingface.co/datasets
[22] http://unicode.org/udhr/d/

		Corpus Code: ISO 639-3	Scripts	ACC↑	ACC70↑	ACC50↑
Highest ACC	mC4	st:sot (S Sotho)	Latn:1000	**1.000**	**1.000**	**1.000**
		fil:fil (Filipino)	Latn:998, Cyrl:1, Hani:1	0.998	0.999	**1.000**
		ro:ron (Romanian)	Latn:996, Zyyy:4, Cyrl:1	0.995	0.997	**1.000**
		id:ind (Indonesian)	Latn:995, Zyyy:3, Hani:1, Hebr:1	0.995	**1.000**	**1.000**
		sw:swa (Swahili)	Latn:995, Zyyy:5	0.995	**1.000**	**1.000**
Lowest ACC		ne:nep (Nepali)	Deva:609, Hani:219, Latn:88, Hang:44, Thai:12, Laoo:8, Zyyy:8, Orya:7, Other:5	0.609	0.730	**0.797**
		mn:mon (Mongolian)	Cyrl:502, Hebr:348, Latn:135, Zyyy:14, Hani:1	0.502	0.557	**0.570**
		cy:cym (Welsh)	Grek:603, Latn:367, Zyyy:11, Hebr:9, Cyrl:5, Zzzz:4, Arab:1	**0.367**	0.338	0.295
		sd:snd (Sindhi)	Latn:654, Arab:329, Zyyy:12, Zzzz:2, Cyrl:1, Hang:1, Telu:1	**0.329**	0.271	0.222
		mr:mar (Marathi)	Hani:454, Thai:252, Latn:119, Deva:116, Zyyy:34, Guru:10, Beng:4, Khmr:3, Other: 8	0.116	0.136	**0.141**
Highest ACC	OSCAR	id:ind (Indonesian)	Latn:998, Zyyy:2	0.998	**1.000**	**1.000**
		war:war (Waray)	Latn:997, Zyyy:3	**0.997**	**0.997**	0.996
		als:gsw (Swiss G)	Latn:996, Zyyy:3, Cyrl:1	0.996	0.996	**1.000**
		vo:vol (Volapük)	Latn:994, Arab:4, Cyrl:1	0.994	**1.000**	**1.000**
		nds:nds (Low G)	Latn:994, Zyyy:2, Cyrl:2, Hang:1, Thaa:1	0.994	**1.000**	**1.000**
Lowest ACC		am:amh (Amharic)	Ethi:822, Latn:164, Zyyy:12, Hani:1, Arab:1	0.822	0.883	**0.940**
		gu:guj (Gujarati)	Gujr:802, Latn:180, Zyyy:12, Deva:6	0.802	0.863	**0.883**
		si:sin (Sinhala)	Sinh:801, Latn:188, Zyyy:11	0.801	0.905	**0.948**
		th:tha (Thai)	Thai:800, Latn:181, Zyyy:18, Hani:1	0.800	0.883	**0.917**
		te:tel (Telugu)	Telu:799, Latn:188, Zyyy:9, Deva:3, Cyrl:1	0.799	0.880	**0.908**

Table 2: Script accuracy for mC4 (top) and OSCAR (bottom) corpora. We display the five best-performing and worst-performing languages. Green indicates correct scripts based on GlotScript-R CORE. Yellow indicates correct scripts based on GlotScript-R AUXILIARY. ACC: accuracy, i.e., the proportion of sentences for which the script identified by GlotScript-T is one of the admissible scripts (according to GlotScript-R) of the language provided by corpus metadata for the sentence. ACC70/ACC50: accuracy for the 70%/50% longest sentences. To save space, we write "Other" for multiple scripts with a small number of sentences. The best scores are bolded for each row. S Sotho = Southern Sotho. Swiss/Low G = Swiss/Low German.

sentences are either written in English or are related to website content, such as website functionalities (comment and search sections), URLs and dates. This confirms that including more scripts, especially all the romanized versions, in the writing metadata even when their use is not solidly attested would hamper our ability to identify incorrect sentences in low resource corpora. This is why we decided that when merging different datasets, if a script is not approved by the majority of sources, it will be kept in AUXILIARY (see §3.4). We also noticed that most sentences with script mismatches are short. We therefore run another set of experiments, this time using a length-based filter that keeps either 70% (ACC70 column) or 50% (ACC50 column) of the longest sentences.

For the bottom languages of OSCAR in Table 2, it is clear that length filtering proves to be effective. Notably, for amh (Amharic), the accuracy improves 0.118 when retaining only the 50% longest sentences. However, this is not the case for the bottom languages of mC4, particularly for cym (Welsh) and snd (Sindhi) where the accuracy worsens. Additionally, the correct scripts for these two languages are not the most frequent in their respective corpora. This suggests that the mistakes are not merely short incorrect sentences, but rather lengthy paragraphs in the wrong language. In the case of Welsh, upon closer inspection, it becomes apparent that the incorrectly identified Greek scripts are actually written in ell (Modern Greek). We also observed suspicious patterns in the Latin portion of this data but it also contained many correctly written sentences in cym (Welsh). For snd (Sindhi), the data contains numerous extensive paraphrases in English, and we also suspect a mix of ara (Arabic) and fas (Farsi) in the Arabic script part.

The infrequent instances of incorrect writing systems in OSCAR may indicate the effectiveness of line-level LID filtering. These results hint at the need for further research on LID. Additionally, we recommend that in newly published LID and corpora, along with the language code, a script code should be assigned to each sentence as part of the metadata. Adopting this recommendation would significantly facilitate error prevention.

6.2. Multilingual Models

6.2.1. Tokenizer Vocabulary

We use GlotScript-T to analyze the token vocabulary of the ten language models and determine each token's script. Figure 2 reports the percentage distribution of each script for each tokenizer's vocabulary. Our findings are as follows.

(i) The Cyrillic representation in the BLOOM tokenizer is relatively scarce compared to other models.

(ii) The BERT tokenizer supports not only Latin scripts but also recognizes Hani, Arabic, Cyrillic and some tokens in an additional 12 scripts.

(iii) Glot500 encompasses the highest number of scripts, totaling 88, followed by mT5, which supports 66. However, a significant portion of these scripts in both models has limited presence.

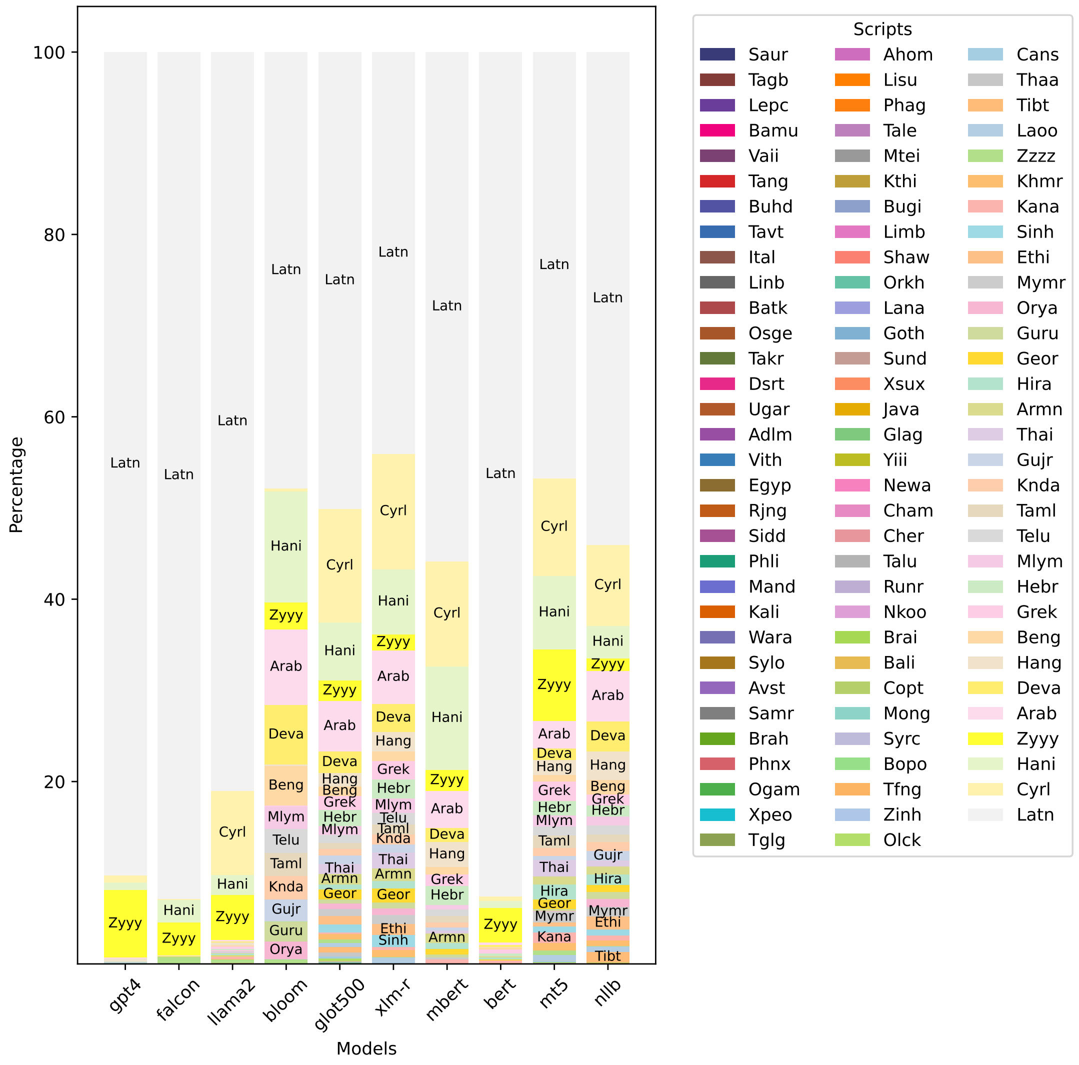


Figure 2: The percentage of each script in the vocabulary of model tokenizers. Scripts with a presence of more than 1% in each tokenizer are text-labeled in the figure.

(iv) Llama2's second most prominent script is Cyrillic.

(v) Falcon's second most prominent script is Hani.

(vi) The GPT-4 tokenizer vocabulary includes representations for 18 scripts, albeit not very comprehensively compared to its coverage of Latin.

(vii) In all tokenizer models combined, a total of 92 scripts has some presence.

6.2.2. UDHR Tokenization

Parsing the UDHR translations with the specific tokenizer associated with each model, we generate a plot illustrating the token count required by each model to tokenize the UDHR translation. Since not all model tokenizers operate at the byte level, this may result in the generation of unknown (UNK) tokens. We only consider tokenizer-translation pairs where fewer than 5% unknown tokens are produced. Figure 3 displays the token count used by each tokenizer (left) and the percentage of unknown tokens (right). Rather than coloring the plot data based on language labels, we choose to use script categories for color representation.

GPT-4, in addition to being trained on English, was also trained on some other languages. For instance, it is capable of translating between English and sin (Sinhala). In tasks such as text generation, the number of generated tokens is particularly important. For example, for the English UDHR translation, the GPT-4 tokenizer produces 1983 tokens. However, for the Sinhala UDHR translation, it generates 20,071 tokens, nearly 10 times more. As the pricing of OpenAI APIs is also based on the number of tokens, this demonstrates that generation of Sinhala is very expensive using GPT-4 in comparison with English.

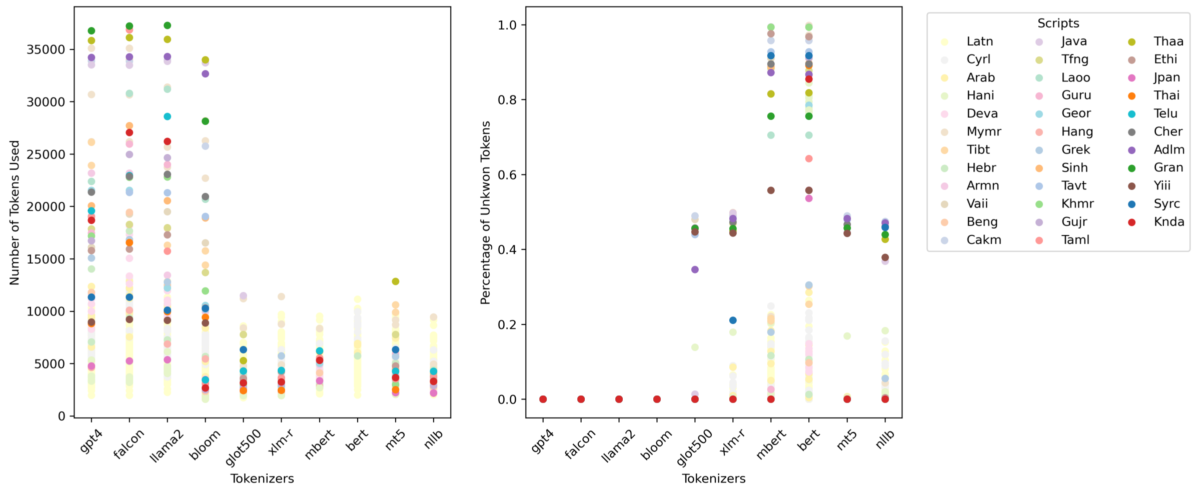


Figure 3: Analysis of the multilinguality of the tokenization of ten language models. This analysis was performed on 396 UDHR translations. Left: the number of tokens into which the UDHR translation is tokenized. We omit a pair of tokenizer and translation with more than 5% unknown tokens. Right: the percentage of unknown tokens generated for a pair of tokenizer and translation.

7. Conclusion

We publish GlotScript-R, an extensive resource covering writing systems for over 7,000 languages, including thousands of typically overlooked “lowest-resource” languages. We open-source GlotScript-T, a script identification tool that supports all 161 scripts in Unicode 15.0. It reports the script distribution within a given text, using ISO 15924 labels. This work is the first to create a highly efficient tool for script identification and output labels based on ISO 15924 with this level of coverage.

We apply GlotScript-R and GlotScript-T to the task of corpus quality assessment. Our findings indicate that these two components work together effectively to improve the quality of existing low resource corpora. Furthermore, we investigate the tokenizers of large language models like GPT-4. This analysis enables us to assess how well a script is represented, serving as an indicator of the representation quality of languages written in that script.

In future work, we aim to expand GlotScript-R by offering a better categorization of writing systems such as "live", "rare", "historic", "romanization present", "romanization in use". We also would like to include more metadata.

Based on the lessons we learned from conducting this study, we recommend that creators of language identification tools and text corpora provide, along with the language code, a script code for each sentence as part of the metadata. Adopting this recommendation could be of great benefit for error prevention and better quality of language resources.

8. Limitations

We acknowledge that some of the input metadata may contain errors; we rely on consensus to decrease this risk. However, there is a potential risk of excluding a writing system for a language or including a noisy one during the collection and processing of NLP corpora.

9. Ethics Statement

The resources used in our study come from openly available metadata with permission for modification and redistribution. We make our research artifacts, GlotScript-R and GlotScript-T openly available to foster collaboration and reproducibility.

10. Acknowledgements

We would like to thank anonymous reviewers. This work was funded by the European Research Council (grant #740516).

11. Bibliographical References

Julien Abadji, Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2021. Ungoliant: An optimized pipeline for the generation of a very large-scale multilingual web corpus. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-9) 2021. Limerick, 12 July 2021 (Online-Event), pages 1 – 9, Mannheim. Leibniz-Institut für Deutsche Sprache.

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David Mortensen, Noah Smith, and Yulia Tsvetkov. 2023. Do all languages cost the same? tokenization in the era of commercial language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9904–9923, Singapore. Association for Computational Linguistics.

Pierre Andrews, Guillaume Wenzek, Kevin Heffernan, Onur Çelebi, Anna Sun, Ammar Kamran, Yingzhe Guo, Alexandre Mourachko, Holger Schwenk, and Angela Fan. 2022. stopes - Modular Machine Translation Pipelines. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 258–265, Abu Dhabi, UAE. Association for Computational Linguistics.

Jan A. Botha, Emily Pitler, Ji Ma, Anton Bakalov, Alex Salcianu, David Weiss, Ryan McDonald, and Slav Petrov. 2017. Natural language processing with small feed-forward networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2879–2885, Copenhagen, Denmark. Association for Computational Linguistics.

Sascha Brawer. 2017. Corpus crawler.

Ralf Brown. 2014. Non-linear mapping for improved identification of 1300+ languages. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 627–632, Doha, Qatar. Association for Computational Linguistics.

Isaac Caswell, Theresa Breiner, Daan van Esch, and Ankur Bapna. 2020. Language id in the wild: Unexpected challenges on the path to a thousand-language web text corpus. *arXiv preprint arXiv:2010.14571*.

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.

Matthew S Dryer and Martin Haspelmath. 2013. WALS: World Atlas of Language Structures. *Max Planck Institute for Evolutionary Anthropology*.

Eberhard, David M., Gary F. Simons, and Charles D. Fennig (eds.). 2023. Ethnologue: Languages of the world. twenty-sixth edition. *SIL International*.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2021. The flores-101 evaluation benchmark for low-resource and multilingual machine translation.

Harald Hammarström, Robert Forkel, Martin Haspelmath, and Sebastian Bank. 2023. Glottolog 4.8. *Leipzig: Max Planck Institute for Evolutionary Anthropology*.

Ayyoob ImaniGooghari, Peiqin Lin, Amir Hossein Kargaran, Silvia Severini, Masoud Jalili Sabet, Nora Kassner, Chunlan Ma, Helmut Schmid, André Martins, François Yvon, and Hinrich Schütze. 2023. Glot500: Scaling multilingual corpora and language models to 500 languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1082–1117, Toronto, Canada. Association for Computational Linguistics.

Armand Joulin, Édouard Grave, Piotr Bojanowski, and Tomáš Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431.

David Kamholz, Jonathan Pool, and Susan Colowick. 2014. PanLex: Building a resource for panlingual lexical translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3145–3150, Reykjavik, Iceland. European Language Resources Association (ELRA).

Amir Hossein Kargaran, Ayyoob Imani, François Yvon, and Hinrich Schütze. 2023. GlotLID: Language identification for low-resource languages. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, et al. 2022.

Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. 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.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Aleksandar Petrov, Emanuele La Malfa, Philip Torr, and Adel Bibi. 2024. Language model tokenizers introduce unfairness between languages. *Advances in Neural Information Processing Systems*, 36.

Shruti Rijhwani, Antonios Anastasopoulos, and Graham Neubig. 2020. OCR Post Correction for Endangered Language Texts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5931–5942, Online. Association for Computational Linguistics.

Alex Salcianu, Andy Golding, Anton Bakalov, Chris Alberti, Daniel Andor, David Weiss, Emily Pitler, Greg Coppola, Jason Riesa, Kuzman Ganchev, et al. 2018. Compact language detector v3.

Hans-Jürgen Sasse. 1991. *Arvanitika: die albanischen Sprachreste in Griechenland*, volume 1. Otto Harrassowitz Verlag.

Kevin P Scannell. 2007. The crúbadán project: Corpus building for under-resourced languages. In *Building and Exploring Web Corpora (WAC3-2007): Proceedings of the 3rd Web as Corpus Workshop, Incorporating Cleaneval*, volume 4, page 5. Presses univ. de Louvain.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Daan van Esch, Tamar Lucassen, Sebastian Ruder, Isaac Caswell, and Clara Rivera. 2022. Writing system and speaker metadata for 2,800+ language varieties. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5035–5046, Marseille, France. European Language Resources Association.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Judit Ács. 2019. Exploring BERT's vocabulary.