Private-Use Area Characters in the Wild: Signal or Noise?

Alexander Gutkin° **Adrian Benton**† **Christo Kirov**†
Brian Roark‡ **Lawrence Wolf-Sonkin**†
Google Research, °London, UK; †New York; ‡Portland, OR
{agutkin,adbenton,ckirov,roark,wolfsonkin}@google.com

Abstract

The Private-Use Area (PUA) designation plays an important role in the Unicode standard. It covers several ranges of Unicode code points with no official character assignments. A PUA range is primarily used as a temporary representation mechanism for characters falling outside the official standard, to facilitate text entry and display of orthographies that are not otherwise adequately represented. The primary downside of PUA use is that characters lose their semantics if the pairing with the corresponding display font is broken, in which case they cannot be faithfully displayed in a general setting. Large-scale multilingual web corpora invariably contain PUA code points of unclear provenance, which may commonly be treated as noise and discarded. We investigate the distribution of PUA characters within large-scale web corpora, and analyze the resulting distributions across both scripts and writing systems. We show that, while the proportion of PUA-bearing paragraphs in the original corpora are small, PUA-bearing tokens can signal texts from under-represented languages. We additionally explore whether an off-the-shelf large language model (LLM) can classify PUA characters as constituting relevant orthographic signals versus punctuation or other noise. Our methods identify millions of paragraphs making use of such characters, and we argue that such data is important for the long tail of data-scarce orthographies. Moreover, as a primary Unicode mechanism for poorly represented writing systems, PUA characters are here to stay.

Keywords: text noise, Unicode, Private-Use Area characters, web-crawled corpora, fonts

1. Introduction

The information loss in digital communication due to legacy character encoding standards has been extensively studied by specialists in digital language preservation (Bird and Simons, 2003; Brown and Woods, 2009; Dombrowski et al., 2024), and natural language processing (Stefanovitch, 2022; Yi and Bowern, 2023; Karagöz et al., 2024). Despite wider adoption of the Unicode character encoding standard along with the OpenType format for representing scalable and programmable computer fonts (Haralambous, 2007; Hossain, 2024), the continued use of many incompatible fonts with legacy and custom character encoding schemes—requiring specialized algorithms for font identification and conversion—has been noted for South Asian scripts and beyond (Lehal et al., 2012; Mahi and Verma, 2015; Pine and Turin, 2018; Bradley and Blokland, 2023). Some communities even choose custom encoding schemes and specialized input methods based on these schemes over Unicode because of the representational inadequacies of the latter, as is the case with the traditional Mongolian script (Batjargal et al., 2011; Wang et al., 2016).[1]

The continued evolution of the Unicode standard partially addresses the "digital divide" between established languages and their writing systems on the one hand and endangered ones on the other (Kornai, 2013; Joshi et al., 2019; Zaugg et al., 2022; Simons et al., 2022). Information loss, however, is still possible in a pure Unicode ecosystem. One Unicode feature potentially responsible for such information loss is the set of private-use characters. These are valid code points from a private use area (PUA), which have no interpretable semantics within the Unicode standard. Individuals or organizations, such as font foundries, can assign meaning to such characters by private agreement (Unicode Consortium, 2024, Section 23.5). PUA characters provide a popular mechanism for representing glyphs from low-resource, historic and endangered language orthographies with no official assignments in the Unicode standard at implementation time (Priest, 2007). While flexible, PUA characters lose their meaning in a digital document without an association to (and availability of) the corresponding font, resulting in information loss quite similar to that observed with legacy character encoding schemes mentioned earlier (Anderson, 2018). Furthermore, the same slot in the PUA table may be inadvertently shared by multiple font designers, preventing general interchange (Kempgen, 2008).

With the advent of large language models (LLMs), large-scale corpora based on web-crawled data, such as MADLAD-400 (Kudugunta

[1] Schillo and Turin (2022) discuss typographic issues in representing North American Indigenous language orthographies stemming from the imprecise specification of Unicode character semantics.

Proceedings of the Third Workshop on Computation and Written Language (CAWL 2026) @LREC 2026, pages 9–32
12 May, 2026. ©ELRA Language Resources Association (ELRA), 2026

et al., 2023), GlotCC (Kargaran et al., 2024), FineWeb2 (Penedo et al., 2025) and DCAD-2000 (Shen et al., 2025), have become important sources of model pre-training data, especially for low-resource languages (Caswell et al., 2020; Bapna et al., 2022). When analyzing such web data one inevitably encounters PUA characters. Since PUA characters often fall outside the standard orthography of natural languages, they are usually treated as noise, and noise can degrade model quality (Cooper Stickland et al., 2023; Wang et al., 2024; Sperduti and Moreo, 2025; Meng et al., 2025).

This paper offers a preliminary study of characters from Unicode PUA ranges found in popular web-crawled corpora, a phenomenon which is under-explored in the literature. We mine text containing PUA characters from two popular large-scale web-crawled datasets of over 7 billion web documents and analyze PUA character distributions over script and writing system at the token and paragraph levels. We find that at most 0.01% of paragraphs contain PUA characters, and thus these are unlikely to harm the performance of downstream models on well-resourced languages. However, we hypothesize that this may not be the case for languages for which only small amounts of data are generally available. In such scenarios, text with PUA characters should receive more attention, especially when prioritizing effective modeling for under-resourced languages. We substantiate this argument with examples from under-represented Cyrillic script orthographies. Inferring the identity of a given PUA code point in the absence of the original character semantics is difficult, but the character context can be informative. We conclude by offering a preliminary step towards this inference task, by assessing the ability of an off-the-shelf LLM to follow instructions when asked to discriminate between orthographic ("interesting") and non-orthographic instances of PUA characters.

The overarching goal of this descriptive analysis is to provide actionable insights for NLP practitioners. Specifically, by mapping PUA distributions, we aim to: (1) prevent the inadvertent destruction of unstandardized scripts in text cleaning pipelines; (2) provide a method for discovering "hidden" low-resource corpora masked by legacy font encodings; and (3) highlight active but poorly supported digital language communities. Ultimately, this analysis is a preliminary step towards bridging the gap between raw web scraping and the targeted curation of inclusive NLP datasets.

2. Background

Unicode has three PUA ranges with 137,468 code points reserved for PUA characters. PUA assignments are typically used as a temporary mechanism by academics, governments and vendors for representing new, as yet un-encoded characters before their official assignment to valid non-PUA Unicode code points. For example, among government-endorsed allocations are the Chinese national standard "Tibetan coded character set – Extensions A and B" for representing precomposed Tibetan script ligatures exclusively using PUA code points.[2] Since approvals to get new characters accepted into the next version of the Unicode standard take considerable time, and some proposed character sets may not be accepted for various reasons, documenting the available PUA characters in use by different entities in order to avoid character collisions and improve coordination becomes important.

One example of such a PUA character allocation registry is the ConScript Unicode Registry,[3] which, among dozens of allocations, originally documented the PUA-based encoding of the Phaistos Disc signs before they were officially adopted as part of the Unicode standard—at which point a mapping from the original PUA code points to code points with relevant semantics was provided (Everson, 2008). In the context of historic writing systems and associated fonts there are additional prominent PUA character allocations for Early Slavonic (Cleminson et al., 2010), medieval orthographies (Emiliano, 2012; Haugen, 2013), and in particular Runic (Magin and Smith, 2023) as part of Medieval Unicode Font Initiative (MUFI),[4] among others.

Fonts covering a wide range of rare language orthographies without recourse to PUA are uncommon, mostly confined to academic publishers such as Brill (Rietbroek, 2021), who have painstakingly avoided PUA since the earliest versions of their fonts covering Cyrillic, Greek and Latin scripts. Rietbroek (2014, p. 1) states: "The Unicode Standard is rigorously adhered to: there is no dependence on the Private Use Area (PUA), as it happens frequently in other fonts with regard to characters carrying rare diacritics or combinations of diacritics. Instead, all alphabetic characters can carry any diacritic or combination of diacritics, even stacked, with automatic correct positioning."

An additional often neglected use case of PUA should be noted. Font designers sometimes prefer PUA code points as an economical shortcut to introducing new glyphs even in cases where the glyphs can be encoded using official Unicode assignments via a longer sequence of code points. For example, there is no single Unicode code point for representing the small *O with*

[2] GB/T 20542-2006 and GB/T 22238-2008.

[3] https://www.evertype.com/standards/csur

[4] https://mufi.info

macron in the Cyrillic orthography of the Mansi language (Bradley and Skribnik, 2021). Instead the canonical way of representing this letter is a combination of U+043E (*Cyrillic small letter O*) and U+0304 (*combining macron*). Some Cyrillic fonts, however, represent this glyph as a single PUA code point. One possible explanation of the font designer motivation for such a choice is that it allows them to treat this glyph similarly to its Latin counterpart (*Latin small letter O with macron*), for which a valid single Unicode code point (U+014D) is defined in the Unicode standard.

Before documenting PUA prevalence in large web text corpora, we first present our data mining methods for extracting text samples.

3. Data Mining Methods

Corpora MADLAD-400 is a general domain multilingual dataset based on CommonCrawl[5] that spans 419 languages (Kudugunta et al., 2023). The dataset has `noisy` and `clean` partitions, the latter obtained by filtering the former with a variety of noise-reducing heuristics. Since we are interested in text containing PUA characters, which may have been deemed noise by prior filters, we make use of the `noisy` partition, which includes 5 trillion tokens over 7.2 billion web documents.

DCAD-2000 (Shen et al., 2025) is a recently introduced large-scale multilingual corpus, significantly larger than MADLAD-400 both in size and coverage, supporting 2,282 language-script pairs. It includes a more recent CommonCrawl dump[6] and other multilingual sources such as MaLA (Lin et al., 2024) and FineWeb-2 (Penedo et al., 2025). Similar to MADLAD-400, we process the unfiltered union of the `keep` and `remove` DCAD-2000 partitions, spanning 8B web documents.[7]

We chose these two datasets as sufficiently different, in terms of both recency (DCAD-2000 is significantly newer given the fast pace of appearance of various web-crawled datasets) and scope, both factors that may influence the PUA character distributions found in the datasets.

Pipeline Our PUA data mining pipeline consists of four steps, similar to the data preparation pipeline in Benton et al. (2026). First, each web document is split at newline characters into what we will call *paragraphs*. Paragraphs with less than four whitespace-delimited tokens are merged with the previous paragraph if that one has less than 30 tokens. Unmerged paragraphs with less than four tokens are discarded. We next discard all paragraphs where more than 40% of whitespace-delimited tokens start with a hashtag character. We then discard paragraphs without any valid PUA-containing tokens, where tokens are considered valid PUA-containing if they (a) have at least one PUA character; and (b) contain non-PUA characters, all of which come from the same script (after stripping leading and trailing punctuation). In other words, tokens with multiple scripts or only PUA characters do not count when deciding whether to retain the paragraph. Finally, we apply the state-of-the-art off-the-shelf LID model GlotLID (Kargaran et al., 2023) that covers 2,282 language-script pairs to the resulting paragraphs.[8]

When determining valid PUA-containing tokens above, PUA characters are allowed in any position of the token (*Anywhere*). This may be too permissive as we are primarily interested in retrieving orthographic tokens corresponding to words in natural language, while PUA characters at the token periphery may potentially encode non-linguistic glyphs, such as list delimiters or other forms of general punctuation.[9] Motivated by this observation, we also pursue an alternate strategy, where valid PUA-containing tokens must contain the PUA characters word-internally (*Internal*). In the next section, we compare the general distributions obtained with both strategies.

Filter	MADLAD-400	DCAD-2000
Documents	7 158 832 435	8 033 425 484
Paragraphs	145 459 293 186	159 081 108 657
(1) Too short	−1 100 076 575	−7 191 309 323
(2) Hashtags	−135 492 904	−25 226 491
(3) No PUA	−144 155 975 436	−151 795 290 723
(3) Invalid PUA	−52 546 928	−60 163 490
Output	15 201 343	9 118 630

Table 1: Counts of filters removing paragraphs without valid PUA-containing tokens. Other than initial document count, all counts are paragraphs.

Filter performance Table 1 presents counts from the use of the above pipeline to mine paragraphs with PUA characters in MADLAD-400 and DCAD-2000, where PUA characters are allowed *Anywhere* in the token. The number of input documents and newline-delimited paragraphs are given in the first two rows. Rows three to six present the number of paragraphs discarded due to the various filtering conditions described above. The number of paragraphs passing all filters and retained by the pipeline are displayed in the final row.

[5] Snapshots before August 2022.

[6] CC-MAIN-2024-46: November 2024.

[7] This number excludes documents in Chinese varieties and Japanese which we removed to preserve disk space. We expect PUA use within these sub-corpora to be similar to MADLAD-400.

[8] The same model was used for document-level LID to build DCAD-2000. The presence of PUA characters may inject some noise into predictions by the LID model.

[9] See the end of Section 4 for examples of PUA characters representing linguistic and non-linguistic glyphs.

As can be seen from the table, the "Invalid PUA" filter for MADLAD-400 drops 78% of candidate paragraphs, while for DCAD-2000 87% of the paragraphs are ignored. This is not surprising because the presence of one "invalid" token, by our definition, is enough to discard the entire paragraph. While the number of useful paragraphs thus removed is likely high, it nevertheless suits our purpose to opt for higher precision and lower recall before investigating the resulting PUA character distributions in the next sections.

4. PUA Character Distribution

Here we take a general look at the top scripts associated with PUA characters followed by the individual top writing systems using PUA character tokens. For each dataset, we consider filtering by either the permissive *Anywhere* or restrictive token-*Internal* PUA character placement strategies.

Scripts The distribution of PUA character tokens among different scripts in MADLAD-400 and DCAD-2000 is shown in Table 2 for the two PUA character placement strategies (*Anywhere* in the left and *Internal* in the right table), resulting in four configurations overall. For each configuration, the top five scripts ranked in terms of number of tokens with PUA characters are shown. Counts from the remaining scripts are aggregated under "Other". Each script, represented by its ISO 15924 code (ISO, 2022), is shown with the count of PUA character tokens and their percentage of the overall number of PUA tokens.

Although DCAD-2000 is significantly larger than MADLAD-400, it yields fewer PUA character tokens. This observation holds for both *Anywhere* and *Internal* token selection strategies. One contributing factor is likely that DCAD-2000 includes fresher CommonCrawl snapshots, containing fewer web pages with broken characters (Shen et al., 2025). This noise reduction may be due in part to existing web pages being updated to newer versions, since older page fonts make use of more PUA assignments in their inventory. As can be seen from the table, all four resulting datasets are heavily skewed towards Latin script tokens with PUA characters, alone accounting for close to (or over) 60% of the respective distributions. In three out of four configurations, Cyrillic script tokens are ranked as the second most frequent. The third most frequent PUA character tokens are encountered in Thai script, with the Greek script as an outlier in one of the conditions. We also note that restricting the allowed placement of PUA characters from *Anywhere* to *Internal* results in 2.5 times fewer PUA tokens for MADLAD-400 and 7.2 times fewer PUA tokens for DCAD-2000.

The contents of the *Other* bins differ among the four configurations. The distributions of PUA character tokens among the scripts (excluding the top five) that belong to this category are shown in Table 3. Each configuration is shown along with the name of the dataset, the PUA character placement strategy, the number of scripts in the category N_S and the statistics for the number of PUA characters per-script that includes the mean μ and logarithm of variance $\log(\sigma^2)$.[10] The MADLAD-400 configurations in general have more distinct scripts with PUA character tokens than DCAD-2000 regardless of the character placement strategy. Unsurprisingly, the more restrictive PUA character placement strategy (*Internal*) results in a smaller set of scripts than the more flexible (*Anywhere*) strategy. Nevertheless, in all configurations there is a very diverse mix of both "well-resourced", under-represented and historic scripts that comprise the *Other* bin. For example, the MADLAD-400 61-script configuration obtained with permissive PUA character placement includes reasonably high-frequency (in terms of PUA character tokens n) scripts such as Armenian (`Armn`, n=13625), Hiragana (`Hira`, n=5232), and Coptic (`Copt`, n=4628), while the long tail of the distribution has Saurashtra (`Saur`, n=11), Cuneiform (`Xsux`, n=2), and Tagalog (`Tglg`, n=1). The collection has a single token in Hanifi Rohingya script (`Rohg`), which in its romanized form "Ruwainggya" means the name of the script itself. The token is 10 characters long with a single PUA character that likely corresponds to an older encoding of a nasal marker (Pandey, 2015). In other cases like the two Cuneiform (`Xsux`) examples, the use of PUA characters is non-orthographic. The full PUA token rank-frequency distribution for all scripts and conditions is provided in Appendix A.

Writing systems Next we consider the distribution of PUA paragraphs over writing systems. We count PUA paragraphs rather than tokens since the writing system is inferred at the paragraph rather than token level. Even modern LID models such as GlotLID typically require at least a sentence worth of text to accurately predict language.

The top-10 writing systems, in terms of the number of paragraphs with at least one PUA character token, are shown in Table 4. The two sub-tables on the left display language rankings for the permissive PUA character placement strategy (*Anywhere*) for MADLAD-400 and DCAD-2000, while the two sub-tables on the right show the rankings obtained allowing only token-internal placement (*Internal*). For each writing system, designated by the ISO 639-3 language code and the

[10] The distribution mostly consists of low counts and a few very high counts, hence the logarithm of variance.

PU character location: *Anywhere* in token					
MADLAD-400			DCAD-2000		
Script	Tokens	%	Script	Tokens	%
Latn	32 240 234	80.4	Latn	12 835 255	81.0
Cyrl	2 803 337	7.0	Cyrl	1 071 969	6.8
Thai	2 672 535	6.7	Grek	644 169	4.1
Hani	825 219	2.1	Arab	396 244	2.5
Arab	507 019	1.3	Thai	296 941	1.9
Other	1 056 045	2.6	*Other*	616 509	3.9
Total	40 104 389			15 861 087	

PU character location: Token-*internal* only					
MADLAD-400			DCAD-2000		
Script	Tokens	%	Script	Tokens	%
Latn	11 739 280	73.4	Latn	1 283 007	58.6
Thai	2 092 819	13.1	Cyrl	464 904	21.2
Cyrl	1 266 526	7.9	Thai	144 511	6.6
Hani	440 618	2.8	Arab	115 386	5.3
Arab	161 666	1.0	Hang	67 066	3.1
Other	287 384	1.8	*Other*	116 425	5.3
Total	15 988 293			2 191 299	

Table 2: Distribution of top-5 scripts among the tokens with PUA characters in MADLAD-400 paragraphs (left) and DCAD-2000 (right) under both *Anywhere* (left table) and *Internal* (right table) strategies.

Dataset details		Script-token distribution		
Name	PU Position	N_S	μ	$\log(\sigma^2)$
MADLAD-400	*Anywhere*	61	17 312.2	22.1
	Internal	51	6386.3	19.8
DCAD-2000	*Anywhere*	49	12 581.8	20.6
	Internal	33	3528.0	17.1

Table 3: Scripts and PUA tokens in the *Other* bin.

ISO 15924 script code, the total number of paragraphs N_P that contain PUA character token(s) are shown along with the corresponding percentage of the total number of such paragraphs observed in the data. In addition to the top-10 ranked writing systems, the information for the remaining writing systems is accumulated in the *Other* bin. The numbers of writing systems N_L in this bin for each resulting collection are shown in Table 5.

The PUA paragraph distributions in Table 4 are dominated by the Latin script and, in particular, English, Spanish and French. One outlier among the top-3 in the DCAD-2000 collection obtained with *Internal* character placement is Czech (ces) which, along with Polish, is also present in the top-10 of DCAD-2000 *Anywhere*. This collection also prominently features Modern Greek (ell), which is not present in any other top-10 ranking. Another interesting entry obtained for MADLAD-400 *Internal* collection is the relatively high-frequency group of paragraphs marked as "no linguistic content" (zxx) by GlotLID, and shown in red. These likely correspond to paragraphs of questionable provenance, such as *mojibake*, text graphics or intentionally obfuscated content. We also note that in addition to Latin, the only other script that features in all four collections is Cyrillic, with the surprising inclusion of low-resource Mansi (mns) orthography in the DCAD-2000 *Internal* collection. Finally, the Thai, Mandarin Chinese and Korean scripts also appear frequently in the top-10 writing systems in three out of four collections. There is a very long tail of well over 700 other language-script pairs present in the *Other* bin (Table 5) that corresponds to about one third or more of the total number of paragraphs with PUA characters obtained for all the collections.

Does size matter? The low number of paragraphs extracted by our pipeline indicates that PUA characters are relatively infrequent in MADLAD-400 and DCAD-2000. The largest collection of paragraphs with PUA characters extracted from MADLAD-400 with permissive PUA character placement (*Anywhere*) consists of 15M paragraphs (see Table 1), which is equivalent to 0.01% of all the paragraphs examined by our pipeline for this condition. For the smallest collection, DCAD-2000 in restrictive PUA character placement mode (*Internal*), this value reduces to 0.006%. Given how infrequently these PUA characters are used, they are likely to have negligible effect on any downstream models constructed using such text for a range of well-resourced languages. It is also likely that the situation will be different for under-represented languages, especially from the long tail of the distribution, for which very small amounts of data are present in such corpora. Importantly, however, the presence of such characters may be quite useful as markers of under-represented orthographies or orthographies "in flux".

Rank-proportion view The analysis in this section relies on the rank-frequency distribution induced by the absolute counts. An alternative is to normalize by the count of tokens and paragraphs in the original corpora, ranking by proportions instead of raw counts. The resulting ranking better highlights the heavy-tail structure and the relative prevalence of PUAs in lower-resourced scripts and writing systems (detailed in Appendix B).

Cyrillic at a glance To demonstrate the above point on the usefulness of this type of data for mining under-represented orthographies, we zoom in on the Cyrillic script portion of the PUA character paragraphs extracted from MADLAD-400 using the permissive character placement strategy. The ranking of the top-10 Cyrillic writing systems in terms of number of paragraphs before and after applying our PUA filters is shown in Table 6. Beyond the obvious presence of well-resourced orthographies, PUA filtering exposes new low-resource orthographies of Khanty, Mansi and Church Slavic,

PU character location: *Anywhere* in token							
MADLAD-400				DCAD-2000			
Lang.	Script	N_P	%	Lang.	Script	N_P	%
eng	Latn	4 424 545	29.1	eng	Latn	1 877 073	20.6
spa	Latn	1 444 067	9.5	fra	Latn	767 446	8.4
fra	Latn	979 665	6.4	spa	Latn	607 533	6.7
rus	Cyrl	918 580	6.0	ell	Grek	597 827	6.6
por	Latn	736 590	4.8	pol	Latn	550 285	6.0
deu	Latn	468 212	3.1	deu	Latn	537 344	5.9
ind	Latn	396 848	2.6	nld	Latn	424 145	4.7
ita	Latn	388 052	2.5	ces	Latn	377 946	4.1
nld	Latn	350 477	2.3	ita	Latn	370 327	4.0
tha	Thai	271 180	1.8	rus	Cyrl	363 222	3.9
Other		4 823 127	31.7	*Other*		2 645 482	29.0
Total		15 201 343				9 118 630	

PU character location: Token-*internal* only							
MADLAD-400				DCAD-2000			
Lang.	Script	N_P	%	Lang.	Script	N_P	%
eng	Latn	1 243 931	29.7	ces	Latn	247 631	24.9
spa	Latn	382 776	9.1	eng	Latn	104 765	10.5
fra	Latn	362 742	8.6	fra	Latn	69 341	6.9
tha	Thai	207 384	4.9	spa	Latn	59 096	5.9
deu	Latn	159 843	3.8	rus	Cyrl	35 634	3.6
zxx	Zzzz	158 484	3.7	mns	Cyrl	30 605	3.1
rus	Cyrl	152 232	3.6	kor	Hang	30 210	3.0
por	Latn	119 231	2.8	pol	Latn	27 436	2.8
cmn	Hani	118 495	2.8	ita	Latn	26 381	2.6
ita	Latn	117 797	2.8	tha	Thai	25 315	2.5
Other		1 171 210	28.0	*Other*		339 636	34.1
Total		4 194 125				996 050	

Table 4: Distribution of top-10 languages among the paragraphs with PUA characters in MADLAD-400 (left) and DCAD-2000 (right) under both *Anywhere* (left table) and *Internal* (right table) strategies.

Name	*Anywhere* (N_L)	*Internal* (N_L)
MADLAD-400	1480	1204
DCAD-2000	1021	716

Table 5: Unique writing systems with PUA characters in the *Other* bin.

Original Distribution		PUA Distribution		Change
Language	%	Language	%	
Russian	88.6	Russian	75.9	↓
Ukrainian	5.2	Ukrainian	9.6	↑
Bulgarian	3.1	Mansi	2.6	↑
Serbian	0.6	Bulgarian	2.4	↓
Kazakh	0.4	Khanty	1.6	↑
Macedonian	0.3	Old Russian	1.6	↑
Belarusian	0.3	Belarusian	0.8	↑
Mongolian	0.2	Kazakh	0.7	↑
Old Russian	0.2	Church Slavic	0.7	↑
Undetermined	0.2	Mongolian	0.6	↑
Other	0.9	*Other*	3.3	↑

Table 6: Top-10 Cyrillic writing systems in MADLAD-400 before and after PUA filtering.

while the proportion of some of the "big" orthographies, such as Russian and Bulgarian, is reduced. Also note the more prominent presence of Old Russian, which we consider low-resource (Franklin, 1985; Schaeken, 2018).

It turns out that lower-resource orthographies can be discovered simply by manually inspecting the data. A sample of Cyrillic script paragraphs with PUA characters from 12 distinct low-resource writing systems is presented in Table 7 along with the corresponding language assignments. In addition to a Cyrillic rendition of Vedic Sanskrit shown in the table, we could identify other historic orthographies such as Middle Bulgarian and Middle/Old Russian, as well as more modern but now somewhat obsolete Cyrillic orthographies of the Azerbaijani and Uzbek languages (Ergun, 2010; Hasanova, 2020). Not all the uses of PUA characters shown in the table are linguistic. For example, while the PUA character <U+F529> in a Mansi token "Н<U+F529>врамыт" is likely a "я" with macron (U+044F + U+0304) from the original Mansi word for "children" (Balandin and Vahrusheva, 1958, p. 68), the character <U+F074> in the Adyghe token "<U+F074>Убалъэм" represents an itemized list delimiter.[11] PUA characters in Cyrillic text can frequently be traced to specific fonts, as detailed in Appendix C.

5. PUA Character Class Prediction

Determining the original semantics of PUA characters in the resulting collections is not trivial. Simple cases like tokens consisting of contiguous runs of identical PUA characters or singleton PUA characters are weeded out by the filters described in Section 3. However, even in cases of singleton PUA character tokens, how can one be certain that it does not represent a valid orthographic word in some writing system, encoded using a single Unicode code point? In general, determining the intended glyphs requires linguistic knowledge of its context. For example, deciding whether the first token in the Adyghe sentence in Table 7 has a non-orthographic vs. linguistically informative leading PUA character is straightforward based on morphology—in the Northwest Caucasian languages Kabardian and Adyghe this is a lexeme of a lemma "убалъэ" meaning "mortar". Hence, we can assume this leading character is some non-linguistic grapheme, e.g., a list delimiter or some element of graphical embellishment.

Manual disambiguation of PUA characters in large collections, as in the example above, is difficult. In this section, we investigate whether this task could be automated by an off-the-shelf LLM which, for each input paragraph in the collection, is prompted to disambiguate each PUA character in the paragraph. Given that LLMs are pre-trained on large quantities of multilingual text, the model may have enough linguistic and orthographic knowledge to correctly classify each character.

[11] Adyghe source: https://apkbr.ru/node/293.

Sample Sentence	Language		
	Name	Code	Family
<U+F074>Убалъэм ихуар къикIыжыгъуафIэ хъуркъым.	Adyghe	ady	Northwest Caucasian
<U+F0AB>Республика йәштәр мәғлүмәт үҙәге<U+F0BB>дәүләт учреждениеһы филиалы сығыш яһаны	Bashkir	bak	Turkic
Шина са<U+F019>гIе хьаьдда молла, цхьа а доцуш, висна	Chechen	che	Northeast Caucasian
Қикви морыкы, мэкгыпы а<U+F62D>қагыргын морыкы вагъэ?	Chukot	ckt	Chukotko-Kamchatkan
Налксема пак<U+10FC00>сянть ваксс тееви кирькс-те нарму<U+10FC00>нень пекстамка кардо.	Erzya	myv	Uralic
Күүкд улсас т<U+F09A>р<U+F09A>н ворошиловск мөрч болсинь Бугшан Чимидова.	Kalmyk	xal	Mongolic
Дыв “Буран” нӈемпи тӳтӓӈ хопн<U+F52B>ӈхлӓт.	Khanty	kca	Uralic
Чилик хик<U+F049>из, къудгъун жезва цавариз.	Lezgian	lez	Northeast Caucasian
Н<U+F529>врамыт ӯщлахтын колытт л<U+F511>ккарыт вос рӯпит<U+F523>гыт.	Mansi	mns	Uralic
Садйо х<U+F115>дй аварудхйате ‘тра.	Sanskrit	san	Indo-European
<U+F034>байырлалга албан чемненир, шимченир болза эки.	Tuvan	tyv	Turkic
Олунньу 13 кҮнэ – тереебут тыл уонна сурук-бичик кҮнэ<U+F4D6>	Yakut	sah	Turkic

Table 7: Sample of MADLAD-400 Cyrillic sentences with PUA character tokens in different languages.

1. Role

You are a specialist in Unicode characters with the special expertise on private-use area (PUA) characters. The PUA character is a character in one of three Unicode ranges: `U+E000`–`U+F8FF`, `U+F0000`–`U+FFFFD`, `U+100000`–`U+10FFFD`.

2. Instructions

- The sentence below contains one or more PUA characters. The number of PUA characters in the sentence is provided to you after the sentence, separated by a semicolon.
- Determine the language of the sentence. Ignore *mojibake*.
- Find each PUA character and annotate it with one of the following 2 labels:
  1. `"LETTER"`: The character is a valid letter in a word that belongs to the language of the sentence.
  2. `"OTHER"`: The character belongs to numbers, punctuation, icons, list delimiters, emoji, symbols.

Your output should consist of only a list of labels, one for each PUA character in the sentence, preceded by analysis of each label. The labels should be output in the same order as the PUA characters in the input sentence.

3. Few-shot examples

INPUT: ⊠Убалъэм ихуар ; 1
ANALYSIS: There is one PUA character. This character is a list delimiter because it occurs at the beginning of the word “Убалъэм”, which is a valid word.
OUTPUT: `[OTHER]`

INPUT: рӯпит⊠гыт үҙәге⊠дәүләт ; 2
ANALYSIS: Two PUA characters. The first PUA character is a valid substring in a word “рӯпитәгыт”. The second PUA character is punctuation in “үҙәге» дәүләт”.
OUTPUT: `[LETTER, OTHER]`

INPUT: loo na spoluprái s⊠Michalem\nSuchánkem a⊠Richardem ; 2
ANALYSIS: Two PUA characters. The first and second PUA characters are punctuation in “s Michalem” and “a Richardem”.
OUTPUT: `[OTHER, OTHER]`

INPUT: Тыныӈ омакем, с⊠рни омакем ; 1
ANALYSIS: The PUA character in “сӧрни” is a valid letter “cyrillic o with macron”.
OUTPUT: `[LETTER]`

INPUT: ɑıU⊠uѡɔ ; 1
ANALYSIS: One PUA character. The full word without the garbled character is “tu̇uѡɔ”.
OUTPUT: `[LETTER]`

INPUT: }rdf⊠lh⊠⊠hgf fj}sgrk⊠ ; 4
ANALYSIS: Four PUA characters. This is not natural language and is probably a *mojibake*.
OUTPUT: `[OTHER, OTHER, OTHER, OTHER]`

4. Input

`SENTENCE: {{text}} ; {{num_puas}}`

Figure 1: LLM prompt for classifying PUA characters into two basic types.

$\|S\|$	N_S	N_T	A	E_U	E_O	R_C	R_L
32K	991 309	8 060 123	98.0	1.2	0.9	1.9	40.7
512	954 470	6 537 913	98.2	1.0	0.8	1.9	45.9
256	916 663	5 526 594	98.4	0.9	0.7	1.8	44.1
128	875 300	4 785 262	98.5	0.8	0.7	1.7	41.3
64	817 177	4 021 097	98.7	0.7	0.6	1.6	36.6

Table 8: LLM instruction following: input length limit ($|S|$), number of paragraphs N_S satisfying the limit, total number of PUA characters N_T observed, and the values of five corresponding metrics.

We judge the suitability of LLM for this task in two ways. First, we assess whether the number of predictions generated by the model matches the number of PUAs in the input paragraph that are to be disambiguated. Being able to produce the same number of labels as there are PUA characters in the input paragraph is a necessary precondition for using an LLM for PUA character class prediction. Second, we assess an LLM's ability to disambiguate PUA characters into those that encode LETTERs, diacritics, or other linguistic characters, or OTHER non-linguistic characters such as list delimiters, whitespace, or unintentional corruption. We judge the performance of the LLM with respect to a 100 paragraph reference set, doubly annotated and reconciled manually by three of the paper authors.

We use Gemini 2.5 Flash (Comanici et al., 2025) as the off-the-shelf LLM this experiment. Gemini 2.5 Flash is a multi-billion parameter transformer model (Vaswani et al., 2017), using a sparse mixture-of-experts (Shazeer et al., 2017) with a long context window and thinking capabilities. We frame the task as binary classification of PUA character types—valid LETTERs in a word or OTHER forms of punctuation or non-linguistic graphemes—using few-shot prompting (Brown et al., 2020) with six examples, as shown in Figure 1. For each example (in Latin, Cyrillic and Thai scripts), in addition to input (a paragraph), the number of relevant PUA characters, and corresponding model output (a list of types, one per-character), we include a sample of an intermediate manual reasoning stage to guide the model (Wei et al., 2022). We arrived at this version of the prompt after several iterations of prompt engineering. Devising a more detailed taxonomy of character tags (e.g., general punctuation, other forms of delimiters, *emoji*) or restoring the originally intended characters where possible, introduces additional complexity beyond our relatively simple current setup, which we leave for future work.

Matching the Correct Label Count For this evaluation, we examine the smallest of four collections of paragraphs, corresponding to DCAD-2000 with token-internal PUA characters. We evaluate how well the LLM follows the instructions given by the prompt in Figure 1 by comparing the number of PUA characters in each input paragraph with the number of character classes (LETTER or OTHER) produced by the model. The findings are presented in Table 8. Each row in the table corresponds to the maximum number of tokens in the input paragraph denoted $|S|$. For each configuration, the total number of paragraphs N_S satisfying the input sequence limit and the total number of PUA characters N_T observed in this data are shown along with five additional metrics. The first is the measure A of how well the model adheres to instructions, *i.e.*, the percentage of paragraphs where the number of input PUA characters in a sequence matches the number of predicted classes. Two additional complementary metrics measure the percentage of sequence mismatches, where the number of per-sequence predicted character classes N_P^s is either under-generated E_U ($N_P^s<N_T^s$) or over-generated E_O ($N_P^s>N_T^s$). The measure R_C is a ratio between the total number of generated character classes N_P and the total number of PUA characters observed in the data N_T. Finally, R_L represents the percentage of LETTER class predictions of the total number of generated classes.

The adherence rate A shown in Table 8 is quite high, ranging from 98.9% for paragraphs less than 64 tokens long to 98.7% considering paragraphs in their entirety. While the LLM tends to under/over-predict at similar rates (E_U=1.2% vs. E_O=0.9%), it tends to generate many more labels than expected when it overpredicts, as evidenced by R_C=1.9. Overall, the LLM tends to label PUA characters as LETTER 40.7% of the time. While the LLM does not adhere perfectly to the instructions, it adheres close enough that one could consider using it to resolve PUA character class.

We also compute the above measures on a paragraph script basis in Appendix D. We determine the script from the top language-script hypothesis for a paragraph provided by the LID model. There are 156 scripts overall reported by the LID model for the paragraphs in the collection analyzed in this section. Among the highest frequency scripts reported in Table 2, the model adheres closely to the instructions for all of them. The only exception is the Han (Hani) script, for which the model only has 90.9% adherence. For the other scripts that adherence rate ranges from 96.8% for Thai up to 99.5% for Arabic. While poor adherence in Hani examples may be a product of relatively little pre-training data, the adherence rates could also be a function of the example length and number of PUA characters present per paragraph by script. More PUA characters per paragraph and longer paragraphs make it harder for the LLM to adhere to its instruc-

gold/pred	letter	other
letter	192	10
other	3	180

Table 9: Confusion matrix for predicting character class of individual PUA characters.

tions. Similar to our observations above, adherence decreases slightly as the limit on the number of input sequence tokens is raised. For example, adherence decreases for Latin script paragraphs from 98.9% when limiting to 64-token examples to 98.2% for all paragraphs.

LLM prompt wording is critical in producing the correct number of labels. During our exploration, detailed in Appendix E, we found that using a prompt that does not explicitly include the number of PUA characters in the input paragraph results in only 64.8% of examples having the correct number of predicted labels, with 72.0% adherence even with 64-token inputs. In addition, this prompt yielded an R_C of 31.7, suggesting massive overgeneration of predicted labels on average. Without experimenting with LLM prompts, mismatch in predicted label count would preclude using LLMs to solve this task at even the most basic level.

Predicting Character Class We sampled 100 paragraphs from the set of paragraphs where the LLM adhered to the labeling instructions, restricted to a maximum character length of 2,000 and number of PUA characters of 20. Over 95% of paragraphs satisfied these two criteria, and they limited the difficulty of the annotation task. For each PUA, in each paragraph, two annotators manually labeled its character class independently. Any disagreements were reconciled by discussion after an independent annotation. Before discussion, annotators had an agreement rate of 91% at the paragraph level and 97.1% at the PUA character level.

Table 9 gives the confusion matrix for the LLM's character-level class prediction. Overall, the LLM achieves 91% paragraph-level accuracy and 96.6% at the character level. This compares favorably with the annotator agreement rate prior to reconciliation: 91% at the paragraph level and 97.1% at the character level. Note that a majority classifier, labeling each PUA as LETTER would have achieved only 52.5% accuracy at this binary prediction task. While the LLM appears to be able to resolve PUA characters to character class at human agreement rates, resolution of PUA to glyph remains untested.

It is worth noting that only 2 out of the 100 paragraphs were labeled heterogeneously, i.e., with both LETTER and OTHER classes being assigned at least once to PUA characters in the paragraph. This suggests that for most instances, predicting a single label of LETTER vs. OTHER for all PUA characters in a paragraphs is sufficient. The two examples with heterogeneous labels were: "*B*☒☒☒☒☒☒☒*F*☒☒☒☒☒*JM AND BRAZ-F* ☒☒☒☒☒*R. 2001. Bowdenol*", a reference where most PUA characters encode the letters in the authors' names, but some represent token delimiters and "*stylem, logem i*☒*typogra*☒*í je Studio Najbrt.*", where the first character is a nonbreaking whitespace character typically inserted after Czech prepositions and the second is likely a Latin letter "f".

6. Conclusions

This work explored the landscape of Unicode PUA characters within MADLAD-400 and DCAD-2000, two large-scale, web-crawled corpora frequently used to pre-train popular LLMs. To extract this data, we introduced a simple pipeline using heuristic-based filters, alongside two position-based strategies for defining "valid" PUA tokens. Our analysis revealed that the extracted paragraph-level data exhibits notable rank-frequency and rank-proportion distributions across scripts and writing systems—a dynamic we highlighted using under-represented Cyrillic orthographies. Although PUA paragraphs constitute a minute fraction of the overall data, they hold significant potential for under-represented languages, provided the identity of underlying glyphs can be accurately resolved. To address this, we tested an off-the-shelf LLM's ability to classify PUA character data into orthographic versus non-linguistic graphemes. We found that Gemini 2.5 Flash generates well-formed predictions for 98% of input paragraphs and distinguished between orthographic and non-orthographic PUA usage with 96.6% accuracy on a small sample, closely tracking our human agreement rate of 97.1%. While these initial classification results are highly encouraging, the ability of current LLMs to reliably resolve PUA characters to their exact intended graphemes remains an open question for future research.

Ideally some dynamic community resource would become available to document attested and/or discovered PUA assignments, with detailed information about the intended grapheme (and perhaps the origin of the assignment), to assist in grapheme recovery when links to fonts are lost. Such a resource would have to be dynamic to allow for ongoing use of PUA characters – in contrast to some of the more static (and sometime stale) resources cited earlier in the paper – and would likely be augmented through semi-automatic means, i.e., computer-assisted discovery and labeling, followed by human validation. This paper presents a hint of what at least some part of such an undertaking would entail.

7. Limitations

Without reliable automated tools, resolving PUA characters like those in Table 7 (Section 4) remains a time-consuming and brittle manual process. Establishing a standard methodology is difficult because each instance demands a unique approach. For example, identifying a Mansi PUA character as a valid letter required running the paragraph through a search engine, examining matching documents to find the context in which the keyword appears, and finally looking it up in an online dictionary.[12] In contrast, identifying the Adyghe PUA character as a list delimiter simply required locating the source newspaper article and examining its visual layout. As reviewers noted, incorporating richer metadata context could significantly improve both this manual effort and the automated LLM-based pipeline (Section 5). Valuable metadata signals include source URLs, web-page font specifications, and established PUA character allocation registries (Section 2).

We presented an exploratory LLM-based approach that classifies PUA characters into two broad categories. While this serves as a foundational step, a significantly more useful disambiguation system would go beyond these broad classes toward a finer-grained character taxonomy. As noted by a reviewer, a proper grapholinguistic analysis requires establishing the graphetic identity of the glyph, its script affiliation, and its relationship to existing Unicode code points (*e.g.*, distinguishing between a historically attested character awaiting encoding and an idiosyncratic font shortcut). These nuanced requirements significantly complicate prompt structure. It remains to be seen whether modern LLMs possess sufficient internal grapholinguistic knowledge to resolve this information independently, without resorting to the external metadata discussed earlier. Consequently, rigorous prompt engineering will remain critical—as demonstrated in this paper, iterative prompt refinement is essential for significantly improving instruction adherence even in binary classification tasks.

Accurate and detailed models to disambiguate PUA characters will ultimately facilitate NLP and grapholinguistic research across several avenues: (1) mining new sources of fully disambiguated, clean text for low-resource languages from large-scale web corpora for inclusion in NLP models; (2) constructing data-driven knowledge graphs of grapholinguistic data encountered in the wild; and (3) augmenting, cross-referencing, and tagging hand-curated PUA character allocation registries. Finally, as aptly noted by a reviewer, the development of robust PUA disambiguation methods is crucial for the primary users of these characters—the communities actively developing or reforming their orthographies.

[12]The most detailed results are returned by Yandex search engine for such queries. Also, finding an online dictionary for a low-resource language is challenging.

8. Ethics Statement

The goal of this work was to provide a preliminary study of the Unicode Private-Use Area characters as found in online data. This type of data tends to be treated as noise and ignored in natural language processing yet, as we argue, may provide a useful source of linguistic signals. Developing techniques to categorize such characters and "fix" them by automatically providing valid Unicode alternative(s), where possible, may help provide additional sources of clean data in scenarios where the need for such data is acutely felt, especially for under-represented orthographies or orthographies "in flux".

9. Acknowledgements

The authors thank Cibu Johny and the anonymous reviewers for many useful comments on this paper.

10. Bibliographical References

Robert Alessi and Antonis Tsolomitis. 2023. Old Standard: A Unicode font for classical and medieval studies. Technical Report v2.7–2023/12/15, The Comprehensive TeX Archive Network.

Deborah Anderson. 2018. Bridging the divide: Supporting the minority and historic scripts in fonts: Problems and recommendations. In *Proceedings of the 2018 Digital Humanities Conference (DH2018)*, pages 328–330, Mexico City, Mexico.

Aleksandr Andreev and Nikita Simmons. 2020. Church Slavonic fonts. Technical Report 2.2, Ponomar Project, Slavonic Computing Initiative.

A. N. Balandin and M. P. Vahrusheva. 1958. *Mansi-Russian dictionary with lexical parallels from Southern Mansi (Kondin) dialect*. Ministry of Education of Russian Soviet Federative Socialist Republic (RSFSR), Leningrad, USSR. In Russian. Russian title: *Mansijsko-russkij slovar' s leksicheskimi parallelyami iz yuzhno-mansijskogo (kondinskogo) dialekta*.

Ankur Bapna, Isaac Caswell, Julia Kreutzer, Orhan Firat, Daan van Esch, Aditya Siddhant,

Mengmeng Niu, Pallavi Baljekar, Xavier Garcia, Wolfgang Macherey, Theresa Breiner, Vera Axelrod, Jason Riesa, Yuan Cao, Mia Xu Chen, Klaus Macherey, Maxim Krikun, Pidong Wang, Alexander Gutkin, Apurva Shah, Yanping Huang, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2022. Building machine translation systems for the next thousand languages. *arXiv preprint arXiv:2205.03983*.

Biligsaikhan Batjargal, Garmaabazar Khaltarkhuu, Fuminori Kimura, and Akira Maeda. 2011. A study of traditional Mongolian script encodings and rendering: Use of Unicode in OpenType fonts. *International Journal on Asian Language Processing*, 21(1):23–44.

Adrian Benton, Alexander Gutkin, Christo Kirov, and Brian Roark. 2026. Mining naturally romanized seed corpora without romanizations. In *Proceedings of the 2026 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC 2026)*, Palma de Mallorca, Spain. Language Resources Association (ELRA). To appear.

Steven Bird and Gary Simons. 2003. Seven dimensions of portability for language documentation and description. *Language*, 79(3):557–582.

Rogier Blokland. 2023. Zyrian Komi. In *The Uralic Languages*, 2nd edition, chapter 14, pages 614–664. Routledge, London, UK.

Jeremy Bradley and Rogier Blokland. 2023. Mansi et al. in print before and under Unicode. *Linguistica Uralica*, 59(4):243–257.

Jeremy Bradley and Elena Skribnik. 2021. The many writing systems of Mansi: challenges in transcription and transliteration. *Multilingual Facilitation*, pages 12–24.

Geoffrey Brown and Kam Woods. 2009. Born broken: Fonts and information loss in legacy digital documents. In *Proceedings of 6th International Conference on Preservation of Digital Objects (iPRES)*, pages 30–37, San Francisco, California.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th Conference on Advances in Neural Information Processing Systems (NeurIPS 2020)*, volume 33, pages 1877–1901. Curran Associates, Inc.

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. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6588–6608, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ralf Cleminson, Victor Baranov, Achim Rabus, David Birnbaum, and Heinz Mitlas. 2010. Proposal for a unified encoding of Early Cyrillic glyphs in the Unicode Private Use Area. *Scripta & e-Scripta: the Journal of Interdisciplinary Mediaeval Studies*, 8:9–26. Sofia: Bulgarian Academy of Sciences.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, et al. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Asa Cooper Stickland, Sailik Sengupta, Jason Krone, Saab Mansour, and He He. 2023. Robustification of multilingual language models to real-world noise in crosslingual zero-shot settings with robust contrastive pretraining. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1375–1391, Dubrovnik, Croatia. Association for Computational Linguistics.

Quinn Dombrowski, Manish Goregaokar, Ben Joeng (Yang), and Abeera Kamran. 2024. Encoding multilingualism: Technical affordances of multilingual publication from manuscripts to Unicode and OpenType. *The Journal of Electronic Publishing (JEP)*, 27(1):309–329.

António Emiliano. 2012. Issues in the typographic representation of medieval primary sources. In Yuji Kawaguchi, Makoto Minegishi, and Wolfgang Viereck, editors, *Corpus-based Analysis and Diachronic Linguistics*, Tokyo University of Foreign Studies. Studies in Linguistics 3, pages 153–173. John Benjamins Publishing Company.

Ayça Ergun. 2010. Politics of romanisation in Azerbaijan (1921–1992). *Journal of the Royal Asiatic Society*, 20(1):33–48.

Michael Everson. 2008. Phaistos ConScript to Unicode table. ConScript Unicode Registry (CSUR). Table version 0.01, Unicode version 5.1.

Simon Franklin. 1985. Literacy and documentation in early medieval Russia. *Speculum*, 60(1):1–38.

Yannis Haralambous. 2007. *Fonts & Encodings*. O'Reilly Media, Sebastopol, CA.

Dilia Hasanova. 2020. Linguistic landscape of Uzbekistan: The rise and fall of Uzbek, Russian, Tajik, and English. In Stanley D. Brunn and Roland Kehrein, editors, *Handbook of the Changing World Language Map*, pages 3015–3028. Springer International Publishing, Cham, Switzerland.

Odd Einar Haugen. 2013. Dealing with glyphs and characters: Challenges in encoding medieval scripts. *Document numérique*, 16(3):97–111.

Anushah Hossain. 2024. Text standards for the "rest of world"': The making of the Unicode standard and the OpenType format. *IEEE Annals of the History of Computing*, 46(1):20–33.

ISO. 2022. ISO 15924: Codes for the representation of names of scripts. https://www.iso.org/obp/ui/#iso:std:iso:15924:ed-2:v1:en. International Organization for Standardization, Technical Committee ISO/TC 46, Information and Documentation.

Pratik Joshi, Christain Barnes, Sebastin Santy, Simran Khanuja, Sanket Shah, Anirudh Srinivasan, Satwik Bhattamishra, Sunayana Sitaram, Monojit Choudhury, and Kalika Bali. 2019. Unsung challenges of building and deploying language technologies for low resource language communities. In *Proceedings of the 16th International Conference on Natural Language Processing*, pages 211–219, International Institute of Information Technology, Hyderabad, India. NLP Association of India.

Fatih Karagöz, Berat Doğan, and Şaziye Betül Özateş. 2024. Towards a clean text corpus for Ottoman Turkish. In *Proceedings of the First Workshop on Natural Language Processing for Turkic Languages (SIGTURK 2024)*, pages 62–70, Bangkok, Thailand and Online. Association for Computational Linguistics.

Amir Hossein Kargaran, Ayyoob Imani, François Yvon, and Hinrich Schuetze. 2023. GlotLID: Language identification for low-resource languages. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6155–6218, Singapore. Association for Computational Linguistics.

Amir Hossein Kargaran, François Yvon, and Hinrich Schütze. 2024. GlotCC: An open broad-coverage commoncrawl corpus and pipeline for minority languages. In *The Thirty-Eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada.

Sebastian Kempgen. 2008. Unicode 5.1, Old Church Slavonic, remaining problems–and solutions, including OpenType features. In *Slovo: Towards a Digital Library of South Slavic Manuscripts; Proceedings of the International Conference*, Sofia, Bulgaria. Bulgarian Academy of Sciences.

András Kornai. 2013. Digital language death. *PloS One*, 8(10):e77056.

Alexey Kryukov. 2011. Old Standard: A Unicode font for classical and medieval studies. Technical Report 2.3, The Comprehensive TeX Archive Network. Edited for CTAN by Bob Tennent.

Sneha Kudugunta, Isaac Caswell, Biao Zhang, Xavier Garcia, Derrick Xin, Aditya Kusupati, Romi Stella, Ankur Bapna, and Orhan Firat. 2023. MADLAD-400: A multilingual and document-level large audited dataset. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:67284–67296.

Gurpreet Singh Lehal, Tejinder Singh Saini, and Savleen Kaur Chowdhary. 2012. An omni-font Gurmukhi to Shahmukhi transliteration system. In *Proceedings of COLING 2012: Demonstration Papers*, pages 313–320, Mumbai, India. The COLING 2012 Organizing Committee.

Peiqin Lin, Shaoxiong Ji, Jörg Tiedemann, André F. T. Martins, and Hinrich Schütze. 2024. MaLA-500: Massive language adaptation of large language models. *arXiv preprint arXiv:2401.13303*.

Elisabeth Maria Magin and Marcus Smith. 2023. "(R)Unicode: Encoding and sustainability issues in runology". *Digital Humanities in the Nordic and Baltic Countries Publications (DHNB)*, 5(1):121–136.

Gurjot Singh Mahi and Amandeep Verma. 2015. Wrecked Indian fonts: A problem for digitalization of Indic documents. In *Proceedings of 60th International Conference on Embedded Librarianship and Technological challenges in Digital Age*, pages 905–914, Chandigarh, India. Indian Library Association.

Yan Meng, Di Wu, and Christof Monz. 2025. How to learn in a noisy world? self-correcting the real-world data noise in machine translation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 7451–7467, Albuquerque, New Mexico. Association for Computational Linguistics.

Janusz Marian Nowacki. 2005. Antykwa Toruńska. Technical Report 2.03, The Polish TEX Users Group, GUST. Type designer: Zygfryd Gardzielewski, font author: Janusz Marian Nowacki.

Anshuman Pandey. 2015. Proposal to encode the Hanifi Rohingya script in Unicode. 2015-10-27 L2/15-278, Unicode Consortium.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. 2025. FineWeb2: One pipeline to scale them all–adapting pre-training data processing to every language. *arXiv preprint arXiv:2506.20920*.

Aidan Pine and Mark Turin. 2018. Seeing the Heiltsuk orthography from font encoding through to Unicode: A case study using Convertextract. In *Proceedings of the LREC 2018 Workshop "CCURL 2018–Sustaining knowledge diversity in the digital age"*, pages 27–30, Miyazaki, Japan. European Language Resources Association.

Lorna A. Priest. 2007. Unicode on the front lines: Endangered languages and Unicode. In *31st Internationalization and Unicode Conference*, pages 1–24, San José, CA. The Object Management Group, Standards Development Organization. Presentation.

Pim Rietbroek. 2014. The Brill typeface user guide & complete list of characters. Technical Report Brill Typeface Version 2.06, Brill, Leiden, Netherlands.

Pim Rietbroek. 2021. Brill typeface version 4.0 character list. Technical Report Brill Typeface Version 4.0, De Gruyter Brill, Leiden, Netherlands.

Jos Schaeken. 2018. *Voices on birchbark: Everyday communication in medieval Russia*, volume 43 of *Studies in Slavic and General Linguistics*. De Gruyter Brill, Leiden, The Netherlands.

Julia Schillo and Mark Turin. 2022. Type right: Examining the underlying causes of common typeface and font errors for Indigenous orthographies, and a possible path forward. *Language Documentation & Conservation*, 16:364–398.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *Proceedings of 5th International Conference on Learning Representations*, Toulon, France. Poster.

Yingli Shen, Wen Lai, Shuo Wang, Xueren Zhang, Kangyang Luo, Alexander Fraser, and Maosong Sun. 2025. DCAD-2000: A multilingual dataset across 2000+ languages with data cleaning as anomaly detection. *arXiv preprint arXiv:2502.11546*.

Gary F. Simons, Abbey L. L. Thomas, and Chad K. K. White. 2022. Assessing digital language support on a global scale. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4299–4305, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Gianluca Sperduti and Alejandro Moreo. 2025. Misspellings in natural language processing: A survey. *arXiv preprint arXiv:2501.16836*.

Nicolas Stefanovitch. 2022. Recovering text from endangered languages corrupted PDF documents. In *Proceedings of the Fifth Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 78–82, Dublin, Ireland. Association for Computational Linguistics.

Unicode Consortium. 2024. The Unicode Standard (version 16.0.0). Technical report, Unicode Consortium, South San Francisco, CA.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*, pages 5998–6008, Long Beach, CA. Curran Associates Inc.

Bin Wang, Chengwei Wei, Zhengyuan Liu, Geyu Lin, and Nancy F. Chen. 2024. Resilience of large language models for noisy instructions. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11939–11950, Miami, Florida, USA. Association for Computational Linguistics.

Boli Wang, Xiaodong Shi, and Yidong Chen. 2016. Coping with problems of Unicoded traditional Mongolian. In *Proceedings of the 4th International Symposium on Natural Language Processing Based on Naturally Annotated Big Data*, pages 125–131, Yantai, China. Springer-Verlag, Berlin, Heidelberg.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th Conference on Advances in Neural Information Processing Systems (NeurIPS 2022)*, volume 35, pages 24824–24837, New Orleans, LA, USA. Curran Associates, Inc.

Christopher Wyrod. 2008. A social orthography of identity: the N'ko literacy movement in West Africa. *International Journal of the Sociology of Language*, 2008(192).

Irene Yi and Claire Bowern. 2023. FileLingR: An R script validation tool for depositors and users of digital language collections. In *Proceedings of the Sixth Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 84–88, Remote. Association for Computational Linguistics.

Isabelle A. Zaugg, Anushah Hossain, and Brendan Molloy. 2022. Digitally-disadvantaged languages. *Internet Policy Review*, 11(2):1–11.

A. PUA Tokens Across Scripts: Full Rank-Frequency Distribution for Absolute Counts

Table 2 provided the top-5 items of the rank-frequency distribution of PUA character tokens across scripts. The remaining scripts were grouped under the "Other" category (Table 3). This distribution is based on the absolute counts of tokens with PUA characters in the relevant corpora under various PUA character placement conditions.

This appendix provides a fuller picture of this distribution by including the PUA token distributions across all scripts essentially "expanding" the "Other" bin. The distributions for tokens with PUA characters for the permissive (*Anywhere*) PUA character placement condition across scripts are shown in Table 10 for MADLAD-400 (left) and DCAD-2000 (right). As can be seen from the tables, MADLAD-400 has a greater variety of scripts than DCAD-2000. For the restrictive PUA character placement condition (*Internal*) the distributions of PUA character tokens across scripts are shown in Table 11 for MADLAD-400 (left-hand side) and DCAD-2000 (right-hand side).

B. PUA Tokens and Paragraphs Across Scripts and Languages: Rank-Proportion Distribution

In our analysis in Section 4, the distributions considered in Tables 2 to 5, as well as the supplementary data in Appendix A, involve rank-frequency distributions based on absolute counts of PUA character tokens or paragraphs containing such tokens. We found that for these distributions the "interesting" low-resource script and writing system signals are mostly confined to the "Other" bin because the heads of the distributions are dominated by well-resourced scripts and languages.

Using the permissive PUA character placement strategy for MADLAD-400 we also computed the ratio of PUA tokens to all the original tokens for each of the scripts. Table 12 shows the ranking of all scripts with over 100,000 original (non-PUA) tokens ranked in terms of decreasing proportion of the number of PUA tokens, shown in the third column, to the number of all the original tokens for each script. Beyond the tokens recognized as code (`Zinh` and `Zyyy`), the top-ranking scripts include liturgical (Coptic, `Copt` and Syriac, `Syrc`), historic (Ancient Egyptian, `Egyp`) and Southeast Asian scripts, such as Khmer, Lao and Thai. Interestingly, the well-resourced scripts, such as Latin, Cyrillic and Arabic rank significantly lower.

For the same permissive PUA character placement in the MADLAD-400 configuration we computed the ratio of PUA paragraphs to all the original paragraphs for each of the language-script pairs emitted by GlotLID, where the count of original paragraphs exceeds 5,000. The results are shown in Table 13, where the rows are sorted according to decreasing PUA paragraph ratio. The top-ranking language-script pairs are completely dominated by low-resource (Mansy, Khanty, Tsakhur, Abaza and Northern Kurdish) orthographies in the Cyrillic script, among othera. There is a strong presence of historic orthographies such as Coptic and Church Slavic, as well as an unknown language in Old Permic script, which is likely medieval Komi (Blokland, 2023).

C. Legacy and Modern Cyrillic Fonts

Beyond specialized fonts from academic publishers, such as Brill fonts mentioned in Section 2, the use of PUA characters is prevalent in public Cyrillic fonts available with most modern operating systems. Some of the fonts distributed with the popular *TeX Live* typesetting package that include support for Cyrillic script are shown in Table 14.[13] The table shows four popular and well-maintained font

[13] https://tug.org/texlive/

Code	Name	Count
Latn	Latin	32 240 234
Cyrl	Cyrillic	2 803 337
Thai	Thai	2 672 535
Hani	Han (Hanzi, Kanji, Hanja)	825 219
Arab	Arabic	507 019
Hang	Hangul (Hangŭl, Hangeul)	423 992
Grek	Greek	240 064
Hebr	Hebrew	95 085
Deva	Devanagari (Nagari)	76 671
Zinh	Code for inherited script	45 175
Taml	Tamil	44 904
Zyyy	Code for undetermined script	19 883
Beng	Bengali	17 326
Khmr	Khmer	14 829
Armn	Armenian	13 625
Kana	Katakana	11 270
Geor	Georgian (Mkhedruli)	11 022
Mlym	Malayalam	5812
Hira	Hiragana	5232
Copt	Coptic	4628
Laoo	Lao	4092
Knda	Kannada	3992
Sinh	Sinhala	3909
Gujr	Gujarati	3653
Ethi	Ethiopic (Ge'ez)	2163
Telu	Telugu	1901
Guru	Gurmukhi	1761
Mymr	Myanmar (Burmese)	1544
Cans	Unified Canadian Aboriginal Syllabics	1524
Syrc	Syriac	403
Orya	Oriya	280
Glag	Glagolitic	241
Ogam	Ogham	161
Cher	Cherokee	145
Tibt	Tibetan	143
Tfng	Tifinagh (Berber)	131
Lana	Tai Tham (Lanna)	80
Bopo	Bopomofo	70
Linb	Linear B	55
Thaa	Thaana	43
Egyp	Egyptian hieroglyphs	35
Nkoo	N'Ko	29
Bali	Balinese	29
Yiii	Yi	26
Java	Javanese	19
Mong	Mongolian	16
Runr	Runic	13
Samr	Samaritan	13
Saur	Saurashtra	11
Sund	Sundanese	9
Bamu	Bamum	9
Lisu	Lisu (Fraser)	4
Merc	Meroitic Cursive	3
Cham	Cham	3
Bass	Bassa Vah	2
Xsux	Cuneiform, Sumero-Akkadian	2
Tale	Tai Le	2
Limb	Limbu	2
Rohg	Hanifi Rohingya	2
Hung	Old Hungarian (Hungarian Runic)	1
Tglg	Tagalog (Baybayin, Alibata)	1
Hano	Hanunoo (Hanunóo)	1
Tavt	Tai Viet	1
Kali	Kayah Li	1
Rjng	Rejang (Redjang, Kaganga)	1
Mtei	Meitei Mayek (Meithei, Meetei)	1
All		40 104 389

Code	Name	Count
Latn	Latin	12 835 255
Cyrl	Cyrillic	1 071 969
Grek	Greek	644 169
Arab	Arabic	396 244
Thai	Thai	296 941
Hang	Hangul (Hangŭl, Hangeul)	166 927
Zinh	Code for inherited script	133 628
Hani	Han (Hanzi, Kanji, Hanja)	42 923
Hebr	Hebrew	36 066
Deva	Devanagari (Nagari)	27 268
Armn	Armenian	24 780
Knda	Kannada	21 990
Ethi	Ethiopic (Ge'ez)	21 144
Sinh	Sinhala	19 588
Khmr	Khmer	19 261
Geor	Georgian (Mkhedruli)	13 067
Gujr	Gujarati	12 891
Laoo	Lao	12 572
Mlym	Malayalam	10 966
Mymr	Myanmar (Burmese)	8764
Java	Javanese	8428
Taml	Tamil	7143
Guru	Gurmukhi	6623
Beng	Bengali	5251
Telu	Telugu	4318
Orya	Oriya	3977
Zyyy	Code for undetermined script	3945
Lana	Tai Tham (Lanna)	2295
Copt	Coptic	1305
Kana	Katakana	510
Cans	Unified Canadian Aboriginal Syllabics	291
Thaa	Thaana	197
Tibt	Tibetan	118
Egyp	Egyptian hieroglyphs	82
Mong	Mongolian	76
Hira	Hiragana	33
Syrc	Syriac	29
Nkoo	N'Ko	10
Sund	Sundanese	9
Cher	Cherokee	6
Tfng	Tifinagh (Berber)	6
Yiii	Yi	5
Dsrt	Deseret (Mormon)	2
Lina	Linear A	2
Tang	Tangut	2
Bopo	Bopomofo	2
Bamu	Bamum	2
Vaii	Vai	1
Xsux	Cuneiform, Sumero-Akkadian	1
Limb	Limbu	1
Batk	Batak	1
Runr	Runic	1
Mero	Meroitic Hieroglyphs	1
Bugi	Buginese	1
All		15 861 087

Table 10: The full rank-frequency distribution of PUA character tokens in MADLAD-400 (left) and DCAD-2000 (right) with *permissive* PUA character placement: ISO 15924 script codes, names and token counts.

Code	Name	Count
Latn	Latin	11 739 280
Thai	Thai	2 092 819
Cyrl	Cyrillic	1 266 526
Hani	Han	440 618
Arab	Arabic	161 666
Hang	Hangul	122 167
Hebr	Hebrew	51 355
Grek	Greek	35 904
Deva	Devanagari	25 680
Taml	Tamil	18 031
Khmr	Khmer	10 958
Beng	Bengali	4208
Laoo	Lao	3417
Copt	Coptic	3271
Geor	Georgian	1590
Zyyy	Code	1584
Kana	Katakana	1449
Mlym	Malayalam	1324
Zinh	Code	911
Telu	Telugu	844
Hira	Hiragana	758
Cans	Unified	568
Gujr	Gujarati	568
Sinh	Sinhala	548
Guru	Gurmukhi	400
Knda	Kannada	388
Mymr	Myanmar	336
Ethi	Ethiopic	248
Orya	Oriya	186
Armn	Armenian	170
Glag	Glagolitic	157
Syrc	Syriac	137
Ogam	Ogham	80
Tfng	Tifinagh	44
Tibt	Tibetan	23
Lana	Tai	18
Egyp	Egyptian	14
Cher	Cherokee	13
Runr	Runic	9
Bopo	Bopomofo	5
Mong	Mongolian	5
Samr	Samaritan	4
Nkoo	N'Ko	3
Rohg	***	2
Bamu	Bamum	2
Tavt	Tai	1
Sund	Sundanese	1
Cham	Cham	1
Thaa	Thaana	1
Bali	Balinese	1
All		15 988 293

Code	Name	Count
Latn	Latin	1 283 007
Cyrl	Cyrillic	464 904
Thai	Thai	144 511
Arab	Arabic	115 386
Hang	Hangul	67 066
Hebr	Hebrew	23 255
Khmr	Khmer	12 373
Grek	Greek	11 775
Ethi	Ethiopic	10 594
Java	Javanese	8427
Hani	Han	8390
Laoo	Lao	7931
Deva	Devanagari	7057
Mymr	Myanmar	3382
Sinh	Sinhala	3342
Orya	Oriya	3058
Guru	Gurmukhi	2879
Knda	Kannada	2043
Gujr	Gujarati	1760
Zinh	Code	1585
Mlym	Malayalam	1295
Geor	Georgian	1293
Lana	Tai	1220
Telu	Telugu	1093
Copt	Coptic	963
Beng	Bengali	892
Taml	Tamil	731
Zyyy	Code	495
Armn	Armenian	433
Mong	Mongolian	52
Cans	Unified	46
Tibt	Tibetan	29
Kana	Katakana	17
Sund	Sundanese	9
Bopo	Bopomofo	2
Hira	Hiragana	2
Thaa	Thaana	1
Xsux	Cuneiform	1
All		2 191 299

Table 11: The full rank-frequency distribution of PUA character tokens in MADLAD-400 (left) and DCAD-2000 (right) with *word-internal* PUA character placement: ISO 15924 script codes, names and token counts.

families containing multiple fonts with a good support for Cyrillic glyphs (among other scripts, if Latin and Greek are also provided). However, a lot of these glyphs are mapped to PUA characters as can be seen from the values of population mean μ, which represents the average number of PUA characters per font. It is likely that a reasonably high proportion of such characters end up in the resulting digital documents that use these fonts and therefore introduce character-level noise into the web-crawled corpora.

Arguably the widest coverage of modern Cyrillic writing systems is provided by the ParaType type foundry. According to the ParaType metadata, 81 living Cyrillic writing systems are supported by its

Code	Name	Ratio (%)	Tokens PUA	Tokens Orig.
Copt	Coptic	0.896	4628	516 649
Zinh	Code	0.326	45 175	13 871 088
Tibt	Tibetan	0.092	143	155 482
Khmr	Khmer	0.066	14 829	22 302 230
Bopo	Bopomofo	0.058	70	119 873
Zyyy	Code	0.047	19 883	41 876 698
Thai	Thai	0.047	2 672 535	5 643 856 638
Cher	Cherokee	0.034	145	429 135
Egyp	Egyptian	0.026	35	135 049
Laoo	Lao	0.019	4092	21 654 614
Syrc	Syriac	0.015	403	2 685 983
Tfng	Tifinagh	0.013	131	1 011 416
Cans	Unified	0.012	1524	12 906 115
Mong	Mongolian	0.011	16	141 788
Hani	Han	0.009	825 219	9 433 282 806
Hang	Hangul	0.004	423 992	9 938 441 074
Hira	Hiragana	0.003	5232	173 292 558
Taml	Tamil	0.003	44 904	1 582 361 376
Mymr	Myanmar	0.003	1544	55 233 135
Armn	Armenian	0.002	13 625	725 541 909
Ethi	Ethiopic	0.002	2163	125 084 695
Geor	Georgian	0.001	11 022	848 345 475
Hebr	Hebrew	0.001	95 085	7 406 031 175
Grek	Greek	0.001	240 064	20 328 695 640
Kana	Katakana	0.001	11 270	1 038 104 139
Deva	Devanagari	0.001	76 671	7 791 717 690
Latn	Latin	0.001	32 240 234	3 349 046 897 307
Knda	Kannada	0.001	3992	423 791 112
Sinh	Sinhala	0.001	3909	424 759 844
Beng	Bengali	0.001	17 326	2 103 453 752
Cyrl	Cyrillic	0.001	2 803 337	344 698 119 171
Mlym	Malayalam	0.001	5812	763 444 007
Arab	Arabic	0.001	507 019	69 314 190 045
Gujr	Gujarati	0.001	3653	527 343 472
Guru	Gurmukhi	0.001	1761	261 678 655
Orya	Oriya	0.001	280	55 081 256
Telu	Telugu	0.000	1901	683 656 693
Thaa	Thaana	0.000	43	77 112 730

Table 12: The proportion of PUA *tokens* (shown in the third column) to all tokens for 38 scripts that have over 100,000 tokens, ranked in decreasing order.

Lang.	Script	Lang. Name	Ratio (%)	PUA	Original
mns	Cyrl	Mansi	32.344	31 639	97 819
kca	Cyrl	Khanty	19.276	19 796	102 697
und	Perm	Undetermined	1.071	57	5323
cop	Copt	Coptic	1.036	740	71 446
tkr	Cyrl	Tsakhur	0.928	53	5713
und	Khmr	Undetermined	0.886	3184	359 403
chu	Cyrl	Church Slavic	0.856	8544	998 622
mdy	Ethi	Male	0.596	35	5870
und	Shaw	Undetermined	0.557	39	6998
abq	Cyrl	Abaza	0.535	101	18 874
und	Kthi	Undetermined	0.531	512	96 358
kmr	Cyrl	Northern Kurdish	0.530	242	45 623
und	Mand	Undetermined	0.504	56	11 105
und	Sidd	Undetermined	0.494	360	72 840
und	Thai	Undetermined	0.475	1357	285 724
und	Nshu	Undetermined	0.430	1781	414 466
zsm	Arab	Standard Malay	0.397	31	7817
chr	Cher	Cherokee	0.394	189	47 988
und	Cham	Undetermined	0.392	106	27 053
yrk	Cyrl	Nenets	0.370	144	38 925
bbj	Latn	Ghomálá'	0.366	40	10 928
crk	Cans	Plains Cree	0.356	19	5340
dwr	Latn	Dawro	0.349	586	168 050
und	Copt	Undetermined	0.345	345	99 872
nnw	Latn	Southern Nuni	0.318	26	8182
nio	Cyrl	Nganasan	0.312	20	6410
und	Mong	Undetermined	0.301	768	255 042
und	Limb	Undetermined	0.296	148	50 010
und	Cher	Undetermined	0.295	345	116 813
itv	Latn	Itawit	0.274	45	16 448
und	Lyci	Undetermined	0.260	168	64 570
wsg	Telu	Adilabad Gondi	0.260	21	8076
und	Linb	Undetermined	0.258	638	247 520
sgh	Cyrl	Shughni	0.252	181	71 825
und	Grek	Undetermined	0.247	3583	1 452 750
und	Tang	Undetermined	0.244	3636	1 492 143
und	Hmnp	Undetermined	0.243	179	73 603
und	Sarb	Undetermined	0.224	19	8464

Table 13: The proportion of PUA *paragraphs* (shown in the fourth column) to all paragraphs for top 38 unique language-script pairs that have over 5,000 paragraphs, ranked in decreasing order.

fonts.[14] While the majority of glyph inventories for these writing systems map trivially to the assigned Cyrillic Unicode code points, there are 27 orthographies, shown in Table 15, that include some glyphs that involve PUA allocations. While the presence of PUA characters in well-resourced language orthographies (e.g., Russian) can be attributed to ParaType support for historic versions of these orthographies, for other languages, such as Avar, Even and Selkup, the relatively high number of PUA characters correlates with how established these orthographies are in terms of their Unicode support over time as well as the orthographic complexity of these languages.

A centralized mapping of ParaType glyph-to-PUA character assignments across all writing systems includes 109 unique assignments. These assignments are shown in Table 16 along with the frequency of appearance N in ParaType orthographies. As can be seen from the figure, the most frequent glyphs are CYRILLIC CAPITAL/SMALL LETTER A WITH MACRON and CYRILLIC CAPITAL/SMALL LETTER YA WITH MACRON, which appear in the orthographies of 10 languages. We hypothesize that such characters were added for compatibility with the existing non-PUA Unicode assignments for Latin script for which similar mappings (such as LATIN CAPITAL LETTER A WITH MACRON) form part of the Unicode standard.

[14] https://github.com/paratype/paratype.github.io/. Accessed: 25th December 2024.

D. Per-script LLM Instruction Metrics

This appendix provides a per-script breakdown of various LLM instruction following metrics discussed in Section 5 for the collection of paragraphs with PUA token-internal characters mined from DCAD-2000. This collection has paragraphs in 156 scripts as determined by GlotLID. Table 17 shows the relevant metrics for configuration with a longest (32K-token) limit imposed on input paragraphs. The entries are sorted by decreasing adherence A (fourth column). Table 18 shows the relevant metrics for the same collection obtained for a shortest (64-token) input sequence limit. The entries in this table are similarly ranked by decreas-

Family Name	Documentation	Scripts	# Fonts	PUA	
				μ	σ^2
Antykwa Toruńska	Nowacki (2005)	Cyrillic, Greek, Latin	16	401.2	1.1
Computer Modern Unicode	Multiple sources	Cyrillic, Greek, Latin	33	271.0	116.4
Church Slavonic	Andreev and Simmons (2020)	Cyrillic	18	180.3	337.5
Old Standard	Kryukov (2011) Alessi and Tsolomitis (2023)	Cyrillic, Greek, Latin	5	73.0	44.0

Table 14: Examples of publicly available popular modern typefaces with extensive Cyrillic support. Some font families support multiple scripts, while others focus on Cyrillic only. Each typeface is shown along with the number of fonts it contains as well as the mean and population variance for number of PUA code points for individual fonts. Note that for multi-script fonts not all PUA code points necessarily represent Cyrillic letters.

Orthography	Code(s)	Family	# PUAs
Avar	ava	Northeast Caucasian	32
Bashkir	bak	Turkic	6
Bulgarian	bul	Indo-European	4
Chechen	che	Northeast Caucasian	12
Chuvash	chv	Turkic	8
Enets	enf, env	Uralic	2
Even (Lamut)	eve	Tungusic	20
Evenki (Tungus)	evn	Tungusic	16
Ingush	inh	Northeast Caucasian	4
Itelmen	itl	Chukotko-Kamchatkan	2
Judeo-Tat	jdt	Indo-European	2
Karachay-Balkar	krc	Turkic	2
Khanty	kca	Uralic	18
...	...	...	...

Orthography	Code(s)	Family	# PUAs
Macedonian	mkd	Indo-European	2
Mansi	mns	Uralic	14
Eastern Mari	mhr	Uralic	2
Nanai	gld	Tungusic	16
Negidal	neg	Tungusic	18
Nenets (Yurak)	yrk	Uralic	2
Nganasan	nio	Uralic	4
Nivkh (Gilyak)	niv	Language isolate	2
Russian	rus	Indo-European	22
Kildin Saami	sjd	Uralic	14
Selkup	sel	Uralic	26
Serbian	srp	Indo-European	6
Ulch	ulc	Tungusic	16
Yakut	sah	Turkic	2

Table 15: PUA code points in ParaType Cyrillic orthographies of some languages. Each orthography in the table is shown along the corresponding ISO 639-3 language code, language family and the even number of PUA code points it uses to represent lower and upper-case letters. The higher-resource language orthographies in the Table (Bulgarian, Macedonian, Russian and Serbian) include historic characters.

ing adherence. Adherence is high for almost all scripts. For over half of the unique scripts, adherence is over 98% over all paragraphs. While restricting to shorter paragraphs tends to improve adherence slightly, *e.g.*, Latin script adherence moves to 98.9% from 98.2%, the gains are modest.

There is a reasonably short tail of scripts that do not display high adherence. If the adherence threshold for acceptable performance is set to 80%, there are 24 such scripts for the first condition (Table 17), and 19 scripts for the second condition (Table 18). For both sequence length limit conditions, among the scripts with over 500 observed PUA paragraphs, the worst performers include Yi (Yiii), Ancient Egyptian (Egyp), and Tangut (Tang), all three representing either logographic writing systems or, in the case of Yi, possibly a syllabary if the source sentences belong to modern Yi orthography. Ignoring the 500 paragraph limit, the worst-performing script in terms of LLM instruction adherence is a modern N'Ko (Nkoo) alphabet originally developed for the Bambara language (Wyrod, 2008).

E. LLM Instruction Following with Number of PUA Characters Excluded from the Prompt

This appendix provides overall and per-script breakdown of the LLM instruction following metrics discussed in Section 5 when the number of PUA characters is not explicitly given in the prompt. Table 19 gives metrics for LLM instruction following with this older, exploratory prompt.

Without explicitly giving the number of PUA characters in the prompt, this task is not trivial. In fact, the performance heavily depends on the input sequence length. Overall instruction adherence is quite low with the best adherence of 72% obtained for the shortest input sequences with a limit of 64 tokens. For all the conditions apart from the longest input sequence limit, the model under- and over-generates the PUA class tags roughly equally according to the E_U and E_O metrics. However, according to R_C, when the model over-generates, it does so heavily and this holds for all the conditions and especially the first one, where for the 32K sequence limit the model hallucinates nearly 32 times more PUA characters than the number

Code	N	Description
U+F43A	1	CAPITAL LETTER STRAIGHT U WITH MACRON
U+F43B	1	SMALL LETTER STRAIGHT U WITH MACRON
U+F460	2	CAPITAL LETTER A WITH ACUTE
U+F461	2	CAPITAL LETTER IE WITH ACUTE
U+F462	2	CAPITAL LETTER I WITH ACUTE
U+F463	2	CAPITAL LETTER O WITH ACUTE
U+F464	2	CAPITAL LETTER U WITH ACUTE
U+F465	2	CAPITAL LETTER YERU WITH ACUTE
U+F466	2	CAPITAL LETTER E WITH ACUTE
U+F467	2	CAPITAL LETTER YU WITH ACUTE
U+F468	2	CAPITAL LETTER YA WITH ACUTE
U+F469	2	CAPITAL LETTER IE WITH DIAERESIS AND ACUTE
U+F46A	2	SMALL LETTER A WITH ACUTE
U+F46B	2	SMALL LETTER IE WITH ACUTE
U+F46C	2	SMALL LETTER I WITH ACUTE
U+F46D	2	SMALL LETTER O WITH ACUTE
U+F46E	2	SMALL LETTER U WITH ACUTE
U+F46F	2	SMALL LETTER YERU WITH ACUTE
U+F470	2	SMALL LETTER E WITH ACUTE
U+F471	2	SMALL LETTER YU WITH ACUTE
U+F472	2	SMALL LETTER YA WITH ACUTE
U+F473	2	SMALL LETTER IE WITH DIAERESIS AND ACUTE
U+F498	1	CAPITAL LETTER YU WITH DIAERESIS
U+F499	1	SMALL LETTER YU WITH DIAERESIS
U+F49A	3	CAPITAL LETTER BARRED O WITH MACRON
U+F49B	3	SMALL LETTER BARRED O WITH MACRON
U+F49C	1	CAPITAL LETTER SCHWA WITH MACRON
U+F49D	1	SMALL LETTER SCHWA WITH MACRON
U+F49E	1	CAPITAL LETTER SHHA WITH BAR
U+F49F	1	SMALL LETTER SHHA WITH BAR
U+F4C6	1	CAPITAL LETTER CHE WITH HOOK
U+F4C7	1	SMALL LETTER CHE WITH HOOK
U+F4CC	1	CAPITAL LETTER U WITH ACUTE
U+F4CD	1	SMALL LETTER U WITH ACUTE
U+F4D2	2	CAPITAL LETTER O WITH GRAVE
U+F4D3	2	SMALL LETTER O WITH GRAVE
U+F4D4	1	CAPITAL LETTER ER WITH CARON
U+F4D5	1	SMALL LETTER ER WITH CARON
U+F4D6	1	CAPITAL LETTER E WITH DOT ABOVE
U+F4D7	1	SMALL LETTER E WITH DOT ABOVE
U+F4D8	2	CAPITAL LETTER YA WITH DIAERESIS
U+F4D9	2	SMALL LETTER YA WITH DIAERESIS
U+F50A	2	CAPITAL LETTER ZE WITH TAIL
U+F50B	2	SMALL LETTER ZE WITH TAIL
U+F50C	3	CAPITAL LETTER ES WITH CEDILLA
U+F50D	3	SMALL LETTER ES WITH CEDILLA
U+F50E	10	CAPITAL LETTER A WITH MACRON
U+F50F	10	SMALL LETTER A WITH MACRON
U+F510	9	CAPITAL LETTER IE WITH MACRON
U+F511	9	SMALL LETTER IE WITH MACRON
U+F512	7	CAPITAL LETTER IE WITH DIAERESIS AND MACRON
U+F513	7	SMALL LETTER IE WITH DIAERESIS AND MACRON
U+F514	1	CAPITAL LETTER IE WITH DIAERESIS AND BREVE
U+F515	1	SMALL LETTER IE WITH DIAERESIS AND BREVE
...	...	...

Code	N	Description
...	...	...
U+F516	1	CAPITAL LETTER ZE WITH CARON
U+F517	1	SMALL LETTER ZE WITH CARON
U+F518	9	CAPITAL LETTER O WITH MACRON
U+F519	9	SMALL LETTER O WITH MACRON
U+F51A	3	CAPITAL LETTER O WITH BREVE
U+F51B	3	SMALL LETTER O WITH BREVE
U+F51C	1	CAPITAL LETTER BARRED O WITH BREVE
U+F51D	1	SMALL LETTER BARRED O WITH BREVE
U+F51E	1	CAPITAL LETTER BARRED O WITH DIAERESIS AND BREVE
U+F51F	1	SMALL LETTER BARRED O WITH DIAERESIS AND BREVE
U+F520	7	CAPITAL LETTER YERU WITH MACRON
U+F521	7	SMALL LETTER YERU WITH MACRON
U+F522	9	CAPITAL LETTER E WITH MACRON
U+F523	9	SMALL LETTER E WITH MACRON
U+F524	1	CAPITAL LETTER E WITH BREVE
U+F525	1	SMALL LETTER E WITH BREVE
U+F526	1	CAPITAL LETTER UKRAINIAN IE WITH DIAERESIS
U+F527	1	SMALL LETTER UKRAINIAN IE WITH DIAERESIS
U+F528	10	CAPITAL LETTER YA WITH MACRON
U+F529	10	SMALL LETTER YA WITH MACRON
U+F52A	1	CAPITAL LETTER YA WITH BREVE
U+F52B	1	SMALL LETTER YA WITH BREVE
U+F52C	9	CAPITAL LETTER YU WITH MACRON
U+F52D	9	SMALL LETTER YU WITH MACRON
U+F52E	1	CAPITAL LETTER YU WITH BREVE
U+F52F	1	SMALL LETTER YU WITH BREVE
U+F532	1	CAPITAL LETTER SHORT I WITH HOOK
U+F533	1	SMALL LETTER SHORT I WITH HOOK
U+F536	2	CAPITAL LETTER YERU WITH BREVE
U+F537	2	SMALL LETTER YERU WITH BREVE
U+F538	1	CAPITAL LETTER REVERSED ZE WITH DIAERESIS
U+F539	1	SMALL LETTER REVERSED ZE WITH DIAERESIS
U+F53C	1	CAPITAL LETTER GHE WITH STROKE
U+F53D	1	SMALL LETTER GHE WITH STROKE
U+F53E	1	CAPITAL LETTER ES WITH TAIL
U+F53F	1	SMALL LETTER ES WITH TAIL
U+F830	1	CAPITAL LETTER A WITH CIRCUMFLEX
U+F831	1	SMALL LETTER A WITH CIRCUMFLEX
U+F833	1	CAPITAL LETTER IE WITH CIRCUMFLEX
U+F834	1	SMALL LETTER IE WITH CIRCUMFLEX
U+F839	1	CAPITAL LETTER O WITH CIRCUMFLEX
U+F83A	1	SMALL LETTER O WITH CIRCUMFLEX
U+F86F	1	CAPITAL LETTER KA WITH MACRON
U+F870	1	SMALL LETTER KA WITH MACRON
U+F872	1	CAPITAL LETTER EL WITH MACRO
U+F873	1	SMALL LETTER EL WITH MACRON
U+F875	1	CAPITAL LETTER ES WITH MACRO
U+F876	1	SMALL LETTER ES WITH MACRON
U+F878	1	CAPITAL LETTER HA WITH MACRO
U+F879	1	SMALL LETTER HA WITH MACRON
U+F87B	1	CAPITAL LETTER TSE WITH MACRO
U+F87C	1	SMALL LETTER TSE WITH MACRON
U+F87E	1	CAPITAL LETTER CHE WITH MACRO
U+F87F	1	SMALL LETTER CHE WITH MACRON

Table 16: Inventory of PUA assignments from ParaType type foundry along with the number of Cyrillic writing systems N, where the particular code point is used. The prefix CYRILLIC has been omitted from the original character names.

of characters found in the actual data. The overall number of hallucinations decreases as one imposes shorter limits on the input sequence length. According to the R_L metric, in the first condition the OTHER class heavily dominates the outputs with LETTER classes comprising only 13.4% of the overall predictions. For rest of the conditions this measure is reasonably constant in the 42–45% ballpark.

Table 20 shows the relevant metrics for configuration with a longest (32K-token) limit imposed on input paragraphs. The entries are sorted by decreasing adherence (fourth column). Table 21 shows the relevant metrics for the same collection obtained for a shortest (64-token) input sequence limit. The entries in this table are similarly ranked by decreasing adherence.

As can be seen from these tables, the performance of adherence and other metrics is highly script- and paragraph length-specific. Among the best consistently performing scripts there are some scripts with very low counts of paragraphs and tokens, such as Elymaic (Elym), Palmyrene (Palm) and Marchen (Marc). For these scripts the metrics do not provide any useful evidence. Among the scripts with sufficient counts, Odiya script (Orya) consistently performs best among all the scripts for both configurations of input para-

Code	N_S	N_T	A	E_U	E_O	R_C	R_L
Adlm	9	60	100.0	0.0	0.0	1.0	6.7
Phnx	3	81	100.0	0.0	0.0	1.0	75.3
Phli	3	98	100.0	0.0	0.0	1.0	19.4
Perm	2	10	100.0	0.0	0.0	1.0	20.0
Palm	1	2	100.0	0.0	0.0	1.0	100.0
Ogam	5	198	100.0	0.0	0.0	1.0	53.0
Nbat	13	318	100.0	0.0	0.0	1.0	96.5
Nand	2	22	100.0	0.0	0.0	1.0	18.2
Nagm	2	32	100.0	0.0	0.0	1.0	90.6
Mult	7	212	100.0	0.0	0.0	1.0	24.1
Modi	25	537	100.0	0.0	0.0	1.0	64.2
Mero	3	76	100.0	0.0	0.0	1.0	21.1
Merc	12	296	100.0	0.0	0.0	1.0	59.5
Medf	403	2892	100.0	0.0	0.0	1.0	1.2
Aghb	2	46	100.0	0.0	0.0	1.0	100.0
Mand	1	22	100.0	0.0	0.0	1.0	100.0
Maka	3	85	100.0	0.0	0.0	1.0	67.1
Kthi	2	47	100.0	0.0	0.0	1.0	85.1
Kali	12	160	100.0	0.0	0.0	1.0	31.9
Ital	3	109	100.0	0.0	0.0	1.0	0.0
Hmng	10	309	100.0	0.0	0.0	1.0	97.4
Phlp	1	62	100.0	0.0	0.0	1.0	0.0
Prti	1	40	100.0	0.0	0.0	1.0	0.0
Hano	1	44	100.0	0.0	0.0	1.0	0.0
Rjng	1	40	100.0	0.0	0.0	1.0	0.0
Yezi	6	105	100.0	0.0	0.0	1.0	32.4
Wcho	2	22	100.0	0.0	0.0	1.0	0.0
Ugar	1	6	100.0	0.0	0.0	1.0	0.0
Toto	3	60	100.0	0.0	0.0	1.0	83.3
Tnsa	8	191	100.0	0.0	0.0	1.0	88.0
Tirh	3	130	100.0	0.0	0.0	1.0	86.2
Thaa	8	310	100.0	0.0	0.0	1.0	16.8
Tglg	2	29	100.0	0.0	0.0	1.0	24.1
Tfng	7	135	100.0	0.0	0.0	1.0	31.9
Tavt	8	205	100.0	0.0	0.0	1.0	47.3
Takr	4	265	100.0	0.0	0.0	1.0	9.8
Tagb	5	79	100.0	0.0	0.0	1.0	72.2
Soyo	4	77	100.0	0.0	0.0	1.0	0.0
Sogo	3	71	100.0	0.0	0.0	1.0	18.3
Sogd	3	39	100.0	0.0	0.0	1.0	71.8
Sind	1	3	100.0	0.0	0.0	1.0	0.0
Shaw	3	29	100.0	0.0	0.0	1.0	69.0
Sarb	1	54	100.0	0.0	0.0	1.0	0.0
Samr	3	122	100.0	0.0	0.0	1.0	65.6
Hatr	2	24	100.0	0.0	0.0	1.0	0.0
Marc	7	30	100.0	0.0	0.0	1.0	0.0
Buhd	1	7	100.0	0.0	0.0	1.0	0.0
Dsrt	13	295	100.0	0.0	0.0	1.0	50.8
Gong	4	92	100.0	0.0	0.0	1.0	35.9
Bhks	7	269	100.0	0.0	0.0	1.0	43.1
Bali	3	37	100.0	0.0	0.0	1.0	40.5
Elym	3	9	100.0	0.0	0.0	1.0	55.6
Dogr	2	8	100.0	0.0	0.0	1.0	0.0
Diak	9	157	100.0	0.0	0.0	1.0	69.4
Elba	3	13	100.0	0.0	0.0	1.0	92.3
Armi	6	182	100.0	0.0	0.0	1.0	52.2
Ahom	5	72	100.0	0.0	0.0	1.0	98.6
Batk	3	24	100.0	0.0	0.0	1.0	0.0
Gran	2	17	100.0	0.0	0.0	1.0	0.0
Orya	2904	3366	100.0	0.0	0.0	1.0	88.4
Khmr	6568	26 009	99.6	0.2	0.2	1.0	86.8
Armn	396	1948	99.5	0.3	0.3	1.0	30.0
Guru	1901	11 160	99.5	0.3	0.2	1.0	25.5
Arab	10 674	52 799	99.5	0.2	0.3	2.1	30.1
Gujr	903	6560	99.4	0.3	0.2	1.9	19.6
Deva	5404	24 033	99.4	0.3	0.3	2.0	77.2
Geor	870	4887	99.1	0.7	0.2	1.0	12.5
Mlym	719	4186	99.0	0.6	0.4	1.0	21.5
Mymr	1664	11 196	98.9	0.4	0.7	1.5	30.0
Grek	7616	44 845	98.8	0.8	0.4	1.7	40.1
Telu	1485	4240	98.8	0.8	0.4	2.9	6.9
Beng	7830	11 531	98.8	1.0	0.3	1.2	30.2
Laoo	1993	23 184	98.7	0.8	0.6	1.6	94.5
Hang	30 204	186 766	98.7	0.9	0.4	1.4	72.6
Taml	1120	4933	98.5	1.0	0.5	3.6	92.4
Latn	760 493	4 840 954	98.2	1.0	0.8	1.8	29.2
Sinh	1296	22 935	98.1	1.5	0.3	1.0	85.6
Cyrl	100 166	910 319	98.1	1.5	0.4	1.3	76.5
...	...	...	...	...	...	...	...

Code	N_S	N_T	A	E_U	E_O	R_C	R_L
...	...	...	...	...	...	...	...
Plrd	47	1240	97.9	0.0	2.1	1.0	75.8
Knda	835	10 982	97.8	1.1	1.1	1.7	42.5
Jpan	359	5178	97.2	1.4	1.4	1.6	13.9
Thai	25 533	342 758	96.8	2.3	0.8	1.6	83.8
Talu	31	815	96.8	0.0	3.2	1.0	88.8
Newa	26	838	96.2	3.8	0.0	1.0	84.7
Nshu	52	2280	96.2	3.8	0.0	0.9	41.8
Osge	25	496	96.0	0.0	4.0	1.0	49.1
Sidd	50	1185	96.0	4.0	0.0	1.0	92.5
Hebr	9410	125 431	95.8	3.8	0.4	1.0	12.2
Xpeo	20	713	95.0	0.0	5.0	1.0	40.6
Sund	18	960	94.4	0.0	5.6	13.0	2.3
Brah	35	1004	94.3	0.0	5.7	16.9	14.0
Copt	196	4919	93.9	4.1	2.0	1.0	53.6
Mtei	16	649	93.8	0.0	6.2	1.0	63.2
Lana	104	3766	93.3	5.8	1.0	0.9	92.1
Bamu	257	16 334	93.0	3.9	3.1	0.8	52.1
Cprt	14	367	92.9	0.0	7.1	1.1	56.6
Glag	14	700	92.9	7.1	0.0	0.4	25.7
Dupl	41	2476	92.7	4.9	2.4	7.2	95.0
Sgnw	48	2154	91.7	2.1	6.2	1.7	79.3
Java	12	915	91.7	0.0	8.3	1.6	2.0
Hani	1784	55 538	90.9	4.6	4.5	2.3	20.1
Avst	11	209	90.9	0.0	9.1	1.0	40.3
Bopo	31	2333	90.3	3.2	6.5	1.4	23.8
Cpmn	40	1879	90.0	2.5	7.5	1.1	84.8
Vith	10	325	90.0	10.0	0.0	1.0	37.5
Cakm	10	342	90.0	0.0	10.0	1.0	43.2
Tibt	95	4306	89.5	5.3	5.3	1.9	62.0
Cans	111	6090	89.2	4.5	6.3	2.1	29.7
Hung	18	442	88.9	11.1	0.0	1.0	63.6
Mong	96	4290	88.5	6.2	5.2	2.0	88.5
Brai	43	2924	88.4	4.7	7.0	2.8	67.5
Khar	8	406	87.5	12.5	0.0	1.0	18.5
Ethi	978	20 769	86.7	9.0	4.3	1.2	86.8
Syrc	66	10 461	86.4	4.5	9.1	1.4	24.5
Orkh	7	538	85.7	0.0	14.3	1.1	79.1
Lyci	14	408	85.7	14.3	0.0	1.0	85.6
Mend	27	1304	85.2	11.1	3.7	1.0	42.2
Kits	70	4204	84.3	4.3	11.4	1.6	73.6
Runr	12	528	83.3	0.0	16.7	1.0	47.1
Lepc	6	261	83.3	16.7	0.0	1.0	77.3
Mani	6	828	83.3	0.0	16.7	7.4	1.0
Pauc	6	141	83.3	0.0	16.7	1.1	53.7
Cham	11	529	81.8	18.2	0.0	1.0	40.3
Linb	65	4158	81.5	9.2	9.2	5.9	13.4
Phag	37	812	81.1	13.5	5.4	1.0	93.4
Saur	131	9763	80.2	11.5	8.4	1.4	30.3
Zanb	5	383	80.0	20.0	0.0	1.0	35.9
Osma	5	253	80.0	0.0	20.0	1.0	73.6
Gonm	5	121	80.0	20.0	0.0	1.0	71.7
Shrd	24	1308	79.2	8.3	12.5	1.0	31.5
Hira	33	4035	78.8	12.1	9.1	1.9	7.3
Kana	37	2421	78.4	10.8	10.8	0.9	39.8
Lina	75	5787	77.3	14.7	8.0	1.7	34.8
Limb	38	11 146	76.3	10.5	13.2	0.9	10.2
Cher	29	3313	75.9	13.8	10.3	2.3	36.1
Mahj	4	115	75.0	0.0	25.0	1.3	43.1
Rohg	8	522	75.0	25.0	0.0	0.7	31.0
Olck	7	295	71.4	14.3	14.3	1.0	99.7
Cari	17	1359	70.6	11.8	17.6	8.2	4.3
Kawi	10	495	70.0	20.0	10.0	5.0	7.3
Wara	64	6591	68.8	20.3	10.9	3.5	71.7
Xsux	480	71 375	67.7	12.9	19.4	2.9	51.2
Zzzz	54	6707	64.8	5.6	29.6	12.2	3.3
Vaii	53	12 064	64.2	15.1	20.8	2.0	71.4
Egyp	829	182 167	63.6	11.8	24.6	2.9	48.6
Hluw	348	52 052	62.1	9.8	28.2	3.3	40.3
Tang	1588	322 750	60.0	14.4	25.6	3.4	49.3
Hmnp	12	2246	58.3	16.7	25.0	1.9	1.1
Mroo	9	1258	55.6	11.1	33.3	3.1	97.2
Yiii	1790	502 871	52.0	17.2	30.9	3.0	52.0
Lisu	10	714	50.0	30.0	20.0	3.2	2.3
Ougr	2	49	50.0	50.0	0.0	0.8	0.0
Sora	2	167	50.0	50.0	0.0	1.0	31.2
Bass	12	1927	41.7	16.7	41.7	3.4	0.0
Nkoo	45	867	28.9	71.1	0.0	0.5	66.7
All	991 309	8 060 123	98.0	1.2	0.9	1.9	40.7

Table 17: LLM instruction following metrics sorted by adherence: 32K-token input sequence limit.

Code	N_S	N_T	A	E_U	E_O	R_C	R_L
Adlm	9	60	100.0	0.0	0.0	1.0	6.7
Mero	3	76	100.0	0.0	0.0	1.0	21.1
Phli	3	98	100.0	0.0	0.0	1.0	19.4
Perm	2	10	100.0	0.0	0.0	1.0	20.0
Palm	1	2	100.0	0.0	0.0	1.0	100.0
Ogam	5	198	100.0	0.0	0.0	1.0	53.0
Nbat	13	318	100.0	0.0	0.0	1.0	96.5
Nand	2	22	100.0	0.0	0.0	1.0	18.2
Nagm	2	32	100.0	0.0	0.0	1.0	90.6
Mult	7	212	100.0	0.0	0.0	1.0	24.1
Modi	25	537	100.0	0.0	0.0	1.0	64.2
Merc	12	296	100.0	0.0	0.0	1.0	59.5
Hano	1	44	100.0	0.0	0.0	1.0	0.0
Medf	403	2892	100.0	0.0	0.0	1.0	1.2
Aghb	2	46	100.0	0.0	0.0	1.0	100.0
Mani	4	153	100.0	0.0	0.0	1.0	37.9
Mand	1	22	100.0	0.0	0.0	1.0	100.0
Maka	3	85	100.0	0.0	0.0	1.0	67.1
Kthi	2	47	100.0	0.0	0.0	1.0	85.1
Kali	12	160	100.0	0.0	0.0	1.0	31.9
Ital	2	55	100.0	0.0	0.0	1.0	0.0
Hmng	10	309	100.0	0.0	0.0	1.0	97.4
Phlp	1	62	100.0	0.0	0.0	1.0	0.0
Phnx	3	81	100.0	0.0	0.0	1.0	75.3
Prti	1	40	100.0	0.0	0.0	1.0	0.0
Rjng	1	40	100.0	0.0	0.0	1.0	0.0
Yezi	6	105	100.0	0.0	0.0	1.0	32.4
Wcho	2	22	100.0	0.0	0.0	1.0	0.0
Ugar	1	6	100.0	0.0	0.0	1.0	0.0
Toto	3	60	100.0	0.0	0.0	1.0	83.3
Tnsa	8	191	100.0	0.0	0.0	1.0	88.0
Tirh	3	130	100.0	0.0	0.0	1.0	86.2
Thaa	8	310	100.0	0.0	0.0	1.0	16.8
Tglg	2	29	100.0	0.0	0.0	1.0	24.1
Tfng	6	80	100.0	0.0	0.0	1.0	53.8
Tavt	8	205	100.0	0.0	0.0	1.0	47.3
Takr	4	265	100.0	0.0	0.0	1.0	9.8
Tagb	5	79	100.0	0.0	0.0	1.0	72.2
Soyo	4	77	100.0	0.0	0.0	1.0	0.0
Sogo	3	71	100.0	0.0	0.0	1.0	18.3
Sogd	3	39	100.0	0.0	0.0	1.0	71.8
Sind	1	3	100.0	0.0	0.0	1.0	0.0
Shaw	3	29	100.0	0.0	0.0	1.0	69.0
Sarb	1	54	100.0	0.0	0.0	1.0	0.0
Samr	3	122	100.0	0.0	0.0	1.0	65.6
Hatr	2	24	100.0	0.0	0.0	1.0	0.0
Marc	7	30	100.0	0.0	0.0	1.0	0.0
Batk	3	24	100.0	0.0	0.0	1.0	0.0
Dsrt	13	295	100.0	0.0	0.0	1.0	50.8
Dogr	2	8	100.0	0.0	0.0	1.0	0.0
Ahom	5	72	100.0	0.0	0.0	1.0	98.6
Bali	3	37	100.0	0.0	0.0	1.0	40.5
Buhd	1	7	100.0	0.0	0.0	1.0	0.0
Gong	4	92	100.0	0.0	0.0	1.0	35.9
Elba	3	13	100.0	0.0	0.0	1.0	92.3
Bopo	28	1072	100.0	0.0	0.0	1.0	55.0
Bhks	7	269	100.0	0.0	0.0	1.0	43.1
Diak	9	157	100.0	0.0	0.0	1.0	69.4
Elym	3	9	100.0	0.0	0.0	1.0	55.6
Gran	2	17	100.0	0.0	0.0	1.0	0.0
Armi	6	182	100.0	0.0	0.0	1.0	52.2
Orya	2006	2327	100.0	0.0	0.0	1.0	84.5
Beng	377	843	99.7	0.3	0.0	1.0	18.8
Khmr	6436	22 846	99.7	0.1	0.2	1.1	86.1
Arab	7220	34 923	99.6	0.1	0.2	1.0	29.8
Geor	687	3492	99.6	0.4	0.0	1.0	8.8
Hang	14 278	65 885	99.5	0.3	0.1	1.0	68.7
Deva	3411	15 501	99.5	0.3	0.2	1.1	59.4
Armn	366	1779	99.5	0.3	0.3	1.0	30.0
Gujr	842	6173	99.4	0.4	0.2	2.0	17.5
Guru	1075	8364	99.3	0.4	0.3	1.0	11.6
Grek	7209	34 234	99.3	0.5	0.2	1.2	18.3
Mymr	1570	8673	99.3	0.3	0.4	1.5	32.8
Mlym	707	4004	99.2	0.4	0.4	1.0	22.4
Laoo	1919	20 651	99.1	0.6	0.4	1.7	95.1
Latn	657 219	2 447 839	98.9	0.5	0.6	1.5	20.2
Telu	344	1578	98.8	0.9	0.3	0.9	27.8
Cyrl	80 795	596 194	98.6	1.2	0.2	1.0	81.5
...	...	...	...	...	...	...	...

Code	N_S	N_T	A	E_U	E_O	R_C	R_L
...	...	...	...	...	...	...	...
Sinh	1215	22 198	98.3	1.6	0.2	1.0	85.5
Hebr	742	9846	98.2	0.9	0.8	1.3	84.8
Thai	18 020	146 225	98.2	1.5	0.3	1.1	81.3
Taml	260	2607	98.1	1.5	0.4	0.9	76.4
Knda	748	10 261	98.0	1.1	0.9	1.7	41.9
Sidd	47	1157	97.9	2.1	0.0	1.0	92.4
Plrd	46	1158	97.8	0.0	2.2	1.0	81.1
Lana	84	1685	97.6	1.2	1.2	1.1	89.5
Ethi	432	5612	97.2	0.9	1.9	1.2	66.0
Jpan	355	4988	97.2	1.4	1.4	1.6	12.3
Talu	31	815	96.8	0.0	3.2	1.0	88.8
Newa	26	838	96.2	3.8	0.0	1.0	84.7
Nshu	50	2173	96.0	4.0	0.0	0.9	44.2
Osge	25	496	96.0	0.0	4.0	1.0	49.1
Dupl	40	2044	95.0	5.0	0.0	0.9	53.0
Xpeo	20	713	95.0	0.0	5.0	1.0	40.6
Bamu	253	7939	94.5	2.8	2.8	1.2	53.2
Sund	18	960	94.4	0.0	5.6	13.0	2.3
Copt	195	4699	94.4	4.1	1.5	1.0	50.8
Brah	35	1004	94.3	0.0	5.7	16.9	14.0
Mtei	16	649	93.8	0.0	6.2	1.0	63.2
Cprt	14	367	92.9	0.0	7.1	1.1	56.6
Sgnw	48	2154	91.7	2.1	6.2	1.7	79.3
Java	12	915	91.7	0.0	8.3	1.6	2.0
Tibt	92	4014	91.3	3.3	5.4	2.1	61.7
Avst	11	209	90.9	0.0	9.1	1.0	40.3
Hani	1704	53 901	90.9	4.5	4.6	2.3	20.1
Cans	109	4788	90.8	4.6	4.6	1.9	20.1
Brai	42	1813	90.5	4.8	4.8	0.9	34.1
Mong	93	3060	90.3	6.5	3.2	1.4	78.9
Glag	10	644	90.0	10.0	0.0	0.3	7.8
Cpmn	40	1879	90.0	2.5	7.5	1.1	84.8
Vith	10	325	90.0	10.0	0.0	1.0	37.5
Cakm	10	342	90.0	0.0	10.0	1.0	43.2
Hira	29	1677	89.7	6.9	3.4	1.8	7.4
Hung	18	442	88.9	11.1	0.0	1.0	63.6
Syrc	65	3663	87.7	4.6	7.7	1.3	74.6
Zzzz	32	1465	87.5	3.1	9.4	11.9	3.9
Khar	8	406	87.5	12.5	0.0	1.0	18.5
Kits	68	3014	86.8	4.4	8.8	1.0	40.6
Lyci	14	408	85.7	14.3	0.0	1.0	85.6
Cher	26	1321	84.6	11.5	3.8	2.2	12.6
Mend	25	1215	84.0	12.0	4.0	1.0	45.3
Pauc	6	141	83.3	0.0	16.7	1.1	53.7
Runr	12	528	83.3	0.0	16.7	1.0	47.1
Orkh	6	537	83.3	0.0	16.7	1.1	79.0
Olck	6	90	83.3	16.7	0.0	0.8	98.7
Lepc	6	261	83.3	16.7	0.0	1.0	77.3
Linb	64	3920	82.8	9.4	7.8	6.2	13.5
Shrd	23	1179	82.6	4.3	13.0	1.0	34.7
Saur	126	7482	82.5	10.3	7.1	1.3	39.7
Phag	37	812	81.1	13.5	5.4	1.0	93.4
Kana	36	2217	80.6	11.1	8.3	0.8	35.0
Lina	72	4136	80.6	13.9	5.6	1.1	75.4
Zanb	5	383	80.0	20.0	0.0	1.0	35.9
Gonm	5	121	80.0	20.0	0.0	1.0	71.7
Osma	5	253	80.0	0.0	20.0	1.0	73.6
Cham	10	413	80.0	20.0	0.0	1.0	52.2
Limb	37	8105	78.4	8.1	13.5	1.0	11.9
Egyp	700	57 503	75.3	8.1	16.6	3.3	63.0
Mahj	4	115	75.0	0.0	25.0	1.3	43.1
Rohg	8	522	75.0	25.0	0.0	0.7	31.0
Wara	59	3278	74.6	18.6	6.8	0.8	59.8
Xsux	437	33 886	74.1	10.8	15.1	3.0	43.2
Kawi	7	165	71.4	14.3	14.3	12.9	1.9
Vaii	48	6385	70.8	12.5	16.7	2.6	67.8
Cari	17	1359	70.6	11.8	17.6	8.2	4.3
Tang	1352	115 930	70.4	11.2	18.4	3.7	51.2
Hluw	308	31 122	70.1	9.4	20.5	2.2	48.0
Hmnp	10	523	70.0	20.0	10.0	0.7	13.3
Yiii	1392	134 304	66.7	12.1	21.1	3.3	59.3
Mroo	8	630	62.5	12.5	25.0	3.8	95.4
Lisu	9	310	55.6	33.3	11.1	0.9	18.2
Sora	2	167	50.0	50.0	0.0	1.0	31.2
Ougr	2	49	50.0	50.0	0.0	0.8	0.0
Bass	12	1927	41.7	16.7	41.7	3.4	0.0
Nkoo	45	867	28.9	71.1	0.0	0.5	66.7
All	817 177	4 021 097	98.7	0.7	0.6	1.6	36.6

Table 18: LLM instruction following metrics sorted by adherence: 64-token input sequence limit.

1. Role

You are a specialist in Unicode characters with the special expertise on private-use area (PUA) characters. The PUA character is a character in one of three Unicode ranges: `U+E000`–`U+F8FF`, `U+F0000`–`U+FFFFD`, `U+100000`–`U+10FFFD`.

2. Instructions

- The sentence below contains one or more PUA characters.
- Determine the language of the sentence. Ignore *mojibake*.
- Find each PUA character and annotate it with one of the following 2 labels:
  1. **"`LETTER`"**: The character is a valid letter in a word that belongs to the language of the sentence.
  2. **"`OTHER`"**: The character belongs to numbers, punctuation, icons, list delimiters, emoji, symbols.

Your output should consist of only a list of labels, one for each PUA character in the sentence, without additional text or explanation.

3. Input

`SENTENCE: {{text}}`

4. Few-shot examples

INPUT: ⊠Убалъэм ихуар
ANALYSIS: One PUA character. This character is a list delimiter because "Убалъэм" is a valid word.
OUTPUT: `[OTHER]`

INPUT: рӯпитэгыт үҙәге⊠дәүләт
ANALYSIS: Two PUA characters. The first PUA character is a valid substring in a word "рӯпитэгыт". The second PUA character is punctuation in "үҙәге» дәүләт".
OUTPUT: `[LETTER, OTHER]`

INPUT: loo na spolupráci s⊠Michalem\nSuchánkem a⊠Richardem
ANALYSIS: Two PUA characters. The first and second PUA characters are punctuation in "s Michalem" and "a Richardem".
OUTPUT: `[OTHER, OTHER]`

INPUT: ɑιU⊠uʁʍ
ANALYSIS: One PUA character. The full word without the garbled character is "ɑιU̇uʁʍ".
OUTPUT: `[LETTER]`

INPUT: }rdf⊠lh⊠⊠hgf fj}sgrk⊠
ANALYSIS: Four PUA characters. This is not natural language and is probably a *mojibake*.
OUTPUT: `[OTHER, OTHER, OTHER, OTHER]`

INPUT: Тыныӈ омакем, с⊠рни омакем
ANALYSIS: The PUA character in "с⊠рни" is a valid letter "cyrillic o with macron".
OUTPUT: `[LETTER]`

Figure 2: Exploratory LLM prompt for classifying the PUA characters into two basic types, without explicitly specifying the number of PUA characters.

$\|S\|$	N_S	N_T	A	E_U	E_O	R_C	R_L
32K	991 441	8 063 322	64.8	16.0	19.2	31.7	13.4
512	954 470	6 537 913	67.1	16.2	16.7	10.7	43.2
256	916 663	5 526 594	68.9	15.2	15.9	8.8	45.4
128	875 300	4 785 262	70.4	14.6	14.9	6.9	45.7
64	817 177	4 021 097	72.0	14.0	14.0	4.6	42.1

Table 19: Exploratory prompt and LLM instruction following: input length limit ($|S|$), number of paragraphs N_S satisfying the limit, total number of PUA characters N_T observed, and the values of five corresponding metrics. Number of PUA characters omitted from LLM prompt.

graph length limit achieving adherence of 91% and 97%, respectively. On the other hand, the model performs very poorly on long paragraphs in Bengali (Beng) script (32K-token limit) adhering to instructions only 5.6% of the time. In this particular scenario, according to the R_C metric (the ratio of total number of generated PUA character classes to the total number of observed character classes), the model exhibits the most extreme over-generation among all the scripts (R_C=3422.4). The Bengali script adherence significantly improves for the shortest token condition (64-token limit) to 65%, with over-generation reduced drastically (R_C=1.0). Analyzing such discrepancies in LLM performance across scripts and conditions requires further investigation.

Code	N_S	N_T	A	E_{UI}	E_O	R_C	R_L	Code	N_S	N_T	A	E_{UI}	E_O	R_C	R_L
Elym	3	9	100.0	0.0	0.0	1.0	55.6	...	...	...	...	...	...	...	...
Palm	1	2	100.0	0.0	0.0	1.0	100.0	Bass	12	1927	8.3	33.3	58.3	6.1	0.0
Orya	2904	3366	90.7	0.6	8.7	1.2	78.4	Hmnp	12	2246	8.3	58.3	33.3	6.5	0.4
Marc	7	30	85.7	14.3	0.0	0.9	57.1	Runr	12	528	8.3	75.0	16.7	5.0	5.2
Medf	403	2892	79.7	1.0	19.4	1.1	1.7	Lina	75	5787	8.0	65.3	26.7	9.3	32.8
Grek	7616	44 845	75.9	13.4	10.7	11.9	29.4	Nshu	52	2280	7.7	67.3	25.0	10.3	81.6
Latn	760 539	4 844 035	73.6	12.6	13.7	13.6	25.6	Nkoo	45	867	6.7	68.9	24.4	11.1	2.8
Bali	3	37	66.7	33.3	0.0	0.9	62.5	Plrd	47	1240	6.4	66.0	27.7	14.1	98.2
Armn	396	1948	64.1	23.7	12.1	9.3	93.0	Hira	33	4035	6.1	54.5	39.4	8.8	1.5
Arab	10 674	52 799	57.7	14.7	27.6	14.5	23.8	Beng	7830	11 531	5.6	1.3	93.1	3422.4	0.7
Deva	5404	24 033	57.5	15.7	26.8	10.5	57.0	Sund	18	960	5.6	72.2	22.2	0.6	57.9
Mymr	1664	11 196	55.0	19.5	25.5	10.3	18.7	Kana	37	2421	5.4	59.5	35.1	10.5	11.9
Guru	1901	11 160	51.8	5.6	42.6	20.6	56.6	Egyp	829	182 167	4.7	65.5	29.8	8.4	26.9
Aghb	2	46	50.0	0.0	50.0	1.1	94.0	Wara	64	6591	4.7	79.7	15.6	10.1	4.5
Gran	2	17	50.0	0.0	50.0	1.1	68.4	Linb	65	4158	4.6	86.2	9.2	4.5	7.0
Kthi	2	47	50.0	50.0	0.0	0.5	72.7	Sgnw	48	2154	4.2	54.2	41.7	33.6	58.7
Perm	2	10	50.0	50.0	0.0	0.8	12.5	Mend	27	1304	3.7	66.7	29.6	28.9	0.9
Wcho	2	22	50.0	0.0	50.0	1.0	56.5	Zzzz	54	6707	3.7	11.1	85.2	31.6	1.1
Brah	35	1004	48.6	25.7	25.7	20.1	9.2	Hluw	348	52 052	3.2	71.0	25.9	16.8	33.5
Avst	11	209	45.5	27.3	27.3	1.0	43.4	Saur	131	9763	3.1	82.4	14.5	5.4	3.5
Java	12	915	41.7	41.7	16.7	3.5	1.2	Kits	70	4204	2.9	71.4	25.7	13.2	60.3
Gujr	903	6560	40.1	44.0	15.9	5.2	4.0	Limb	38	11 146	2.6	78.9	18.4	0.9	34.3
Ethi	978	20 769	38.9	29.1	32.0	22.8	22.2	Dupl	41	2476	2.4	80.5	17.1	7.9	3.2
Cyrl	100 252	910 437	37.5	26.9	35.5	25.2	15.8	Brai	43	2924	2.3	79.1	18.6	5.9	96.7
Knda	835	10 982	37.5	24.0	38.6	6.2	33.5	Tang	1588	322 750	2.1	65.1	32.8	10.6	24.4
Geor	870	4887	37.4	40.6	22.1	1.9	7.7	Yiii	1790	502 871	1.7	66.5	31.8	9.5	22.9
Cham	11	529	36.4	36.4	27.3	31.6	98.8	Armi	6	182	0.0	100.0	0.0	0.5	25.6
Bamu	257	16 334	36.2	49.0	14.8	8.2	53.4	Batk	3	24	0.0	100.0	0.0	0.5	0.0
Osge	25	496	36.0	48.0	16.0	33.6	99.1	Bhks	7	269	0.0	85.7	14.3	0.7	41.0
Hang	30 204	186 766	33.6	24.5	41.9	278.4	2.6	Bopo	31	2333	0.0	83.9	16.1	1.0	19.9
Ital	3	109	33.3	0.0	66.7	1.2	1.6	Buhd	1	7	0.0	0.0	100.0	2.4	0.0
Mero	3	76	33.3	33.3	33.3	0.8	37.3	Cakm	10	342	0.0	50.0	50.0	0.9	31.8
Pauc	6	141	33.3	33.3	33.3	1.1	59.3	Cher	29	3313	0.0	41.4	58.6	15.2	1.6
Sogd	3	39	33.3	33.3	33.3	1.1	80.5	Cpmn	40	1879	0.0	60.0	40.0	11.0	98.7
Toto	3	60	33.3	33.3	33.3	1.1	6.1	Cprt	14	367	0.0	50.0	50.0	0.9	55.4
Sinh	1296	22 935	32.3	38.9	28.9	2.8	32.9	Dogr	2	8	0.0	50.0	50.0	0.9	0.0
Mtei	16	649	31.2	56.2	12.5	25.8	98.8	Elba	3	13	0.0	66.7	33.3	3.0	71.8
Vith	10	325	30.0	40.0	30.0	51.0	98.9	Hano	1	44	0.0	100.0	0.0	0.6	0.0
Khmr	6568	26 009	29.0	7.1	63.8	14.6	46.9	Hatr	2	24	0.0	100.0	0.0	0.8	45.0
Mlym	719	4186	28.4	62.9	8.8	0.7	15.1	Khar	8	406	0.0	50.0	50.0	7.5	1.6
Jpan	359	5178	27.9	23.4	48.7	6.2	6.4	Lepc	6	261	0.0	66.7	33.3	63.2	0.3
Thai	25 533	342 758	25.2	35.5	39.3	131.9	15.4	Lyci	14	408	0.0	78.6	21.4	0.8	50.3
Gong	4	92	25.0	75.0	0.0	0.7	7.8	Mahj	4	115	0.0	50.0	50.0	1.0	34.8
Merc	12	296	25.0	50.0	25.0	55.9	12.5	Maka	3	85	0.0	66.7	33.3	0.9	87.2
Thaa	8	310	25.0	62.5	12.5	0.8	15.1	Mand	1	22	0.0	0.0	100.0	1.5	100.0
Laoo	1993	23 184	24.9	54.7	20.3	3.8	55.8	Mani	6	828	0.0	66.7	33.3	21.0	0.5
Hani	1784	55 538	24.3	34.4	41.4	19.3	34.5	Mroo	9	1258	0.0	66.7	33.3	16.7	93.2
Adlm	9	60	22.2	55.6	22.2	1.0	16.4	Nagm	2	32	0.0	50.0	50.0	1.0	65.6
Glag	14	700	21.4	42.9	35.7	6.8	2.5	Nand	2	22	0.0	0.0	100.0	2.8	93.4
Ahom	5	72	20.0	60.0	20.0	0.8	39.0	Nbat	13	318	0.0	84.6	15.4	0.7	92.8
Gonm	5	121	20.0	80.0	0.0	0.5	45.2	Orkh	7	538	0.0	57.1	42.9	1.2	92.1
Ogam	5	198	20.0	60.0	20.0	0.7	30.6	Osma	5	253	0.0	80.0	20.0	65.1	0.7
Tibt	95	4306	20.0	44.2	35.8	18.5	77.1	Ougr	2	49	0.0	100.0	0.0	0.5	20.8
Cans	111	6090	18.9	53.2	27.9	14.6	53.7	Phli	3	98	0.0	66.7	33.3	0.6	18.6
Telu	1485	4240	18.9	15.1	66.0	1066.2	3.6	Phlp	1	62	0.0	0.0	100.0	1.0	34.9
Sidd	50	1185	18.0	64.0	18.0	10.4	38.7	Phnx	3	81	0.0	100.0	0.0	0.7	46.6
Cari	17	1359	17.6	64.7	17.6	15.5	0.8	Prti	1	40	0.0	100.0	0.0	0.3	7.1
Taml	1120	4933	17.1	5.9	77.0	1205.8	1.2	Rjng	1	40	0.0	100.0	0.0	0.3	100.0
Hung	18	442	16.7	66.7	16.7	18.5	98.7	Rohg	8	522	0.0	62.5	37.5	31.8	99.5
Kali	12	160	16.7	83.3	0.0	0.7	30.8	Samr	3	122	0.0	100.0	0.0	0.3	19.0
Hebr	9410	125 431	16.3	56.4	27.3	23.3	6.7	Sarb	1	54	0.0	100.0	0.0	0.7	94.7
Dsrt	13	295	15.4	46.2	38.5	13.5	97.8	Shaw	3	29	0.0	100.0	0.0	0.4	16.7
Lana	104	3766	14.4	45.2	40.4	27.1	68.2	Shrd	24	1308	0.0	79.2	20.8	13.1	1.6
Mult	7	212	14.3	85.7	0.0	0.6	52.6	Sind	1	3	0.0	0.0	100.0	669.3	0.0
Olck	7	295	14.3	57.1	28.6	55.8	100.0	Sogo	3	71	0.0	100.0	0.0	0.8	12.3
Copt	196	4919	13.3	76.0	10.7	4.9	4.7	Sora	2	167	0.0	50.0	50.0	0.7	64.6
Tavt	8	205	12.5	75.0	12.5	0.7	71.7	Soyo	4	77	0.0	75.0	25.0	0.8	61.9
Modi	25	537	12.0	76.0	12.0	0.6	48.1	Tagb	5	79	0.0	80.0	20.0	0.8	45.0
Newa	26	838	11.5	61.5	26.9	0.8	83.4	Takr	4	265	0.0	50.0	50.0	0.6	9.1
Diak	9	157	11.1	55.6	33.3	0.8	77.6	Tfng	7	135	0.0	28.6	71.4	1.5	31.5
Phag	37	812	10.8	64.9	24.3	0.7	81.5	Tglg	2	29	0.0	100.0	0.0	0.9	61.5
Xsux	480	71 375	10.6	65.8	23.5	9.9	21.3	Tirh	3	130	0.0	100.0	0.0	0.5	60.3
Hmng	10	309	10.0	90.0	0.0	0.7	96.1	Tnsa	8	191	0.0	50.0	50.0	0.9	57.6
Kawi	10	495	10.0	40.0	50.0	7.3	1.6	Ugar	1	6	0.0	100.0	0.0	0.7	0.0
Lisu	10	714	10.0	60.0	30.0	26.3	0.1	Vaii	53	12 064	0.0	47.2	52.8	14.3	31.3
Talu	31	815	9.7	54.8	35.5	2.6	29.3	Xpeo	20	713	0.0	85.0	15.0	0.7	26.6
Mong	96	4290	9.4	50.0	40.6	7.6	17.9	Yezi	6	105	0.0	16.7	83.3	2.3	70.1
Syrc	66	10 461	9.1	81.8	9.1	3.6	62.5	Zanb	5	383	0.0	60.0	40.0	7.6	2.3
...	...	...	...	...	...	...	...	All	991 441	8 063 322	64.8	16.0	19.2	31.7	13.4

Table 20: Exploratory prompt: LLM instruction following metrics sorted by adherence: 32K-token input sequence limit. Number of PUA characters omitted from LLM prompt.

Code	N_S	N_T	A	E_{UI}	E_O	R_C	R_L	Code	N_S	N_T	A	E_{UI}	E_O	R_C	R_L
Elym	3	9	100.0	0.0	0.0	1.0	55.6	...	...	...	...	...	...	...	...
Palm	1	2	100.0	0.0	0.0	1.0	100.0	Talu	31	815	9.7	54.8	35.5	2.6	29.3
Orya	2006	2327	96.6	0.2	3.1	1.0	83.9	Syrc	65	3663	9.2	81.5	9.2	9.4	69.4
Marc	7	30	85.7	14.3	0.0	0.9	57.1	Bass	12	1927	8.3	33.3	58.3	6.1	0.0
Medf	403	2892	79.7	1.0	19.4	1.1	1.7	Lina	72	4136	8.3	66.7	25.0	5.1	6.0
Latn	657 219	2 447 839	79.1	10.4	10.4	3.2	31.7	Runr	12	528	8.3	75.0	16.7	5.0	5.2
Grek	7209	34 234	78.5	12.7	8.9	3.3	21.0	Nshu	50	2173	8.0	68.0	24.0	10.7	81.9
Guru	1075	8364	68.8	5.0	26.1	4.6	3.1	Hira	29	1677	6.9	62.1	31.0	3.1	5.1
Bali	3	37	66.7	33.3	0.0	0.9	62.5	Nkoo	45	867	6.7	68.9	24.4	11.1	2.8
Armn	366	1779	65.6	23.2	11.2	10.1	93.6	Plrd	46	1158	6.5	67.4	26.1	15.0	98.8
Beng	377	843	65.3	19.6	15.1	1.0	24.0	Egyp	700	57 503	5.6	72.7	21.7	8.6	41.5
Taml	260	2607	65.0	20.8	14.2	7.0	5.0	Kana	36	2217	5.6	61.1	33.3	10.7	12.7
Arab	7220	34 923	63.1	14.6	22.3	3.3	59.8	Sund	18	960	5.6	72.2	22.2	0.6	57.9
Ethi	432	5612	58.6	17.1	24.3	9.4	55.8	Wara	59	3278	5.1	83.1	11.9	5.6	6.6
Deva	3411	15 501	58.4	21.5	20.1	3.0	34.9	Linb	64	3920	4.7	85.9	9.4	4.8	6.9
Mymr	1570	8673	56.4	19.1	24.5	7.1	14.4	Sgnw	48	2154	4.2	54.2	41.7	33.6	58.7
Hebr	742	9846	51.3	29.4	19.3	19.0	82.8	Mend	25	1215	4.0	72.0	24.0	29.1	1.0
Telu	344	1578	50.3	39.0	10.8	0.8	27.9	Hluw	308	31 122	3.6	76.6	19.8	13.0	38.4
Aghb	2	46	50.0	0.0	50.0	1.1	94.0	Saur	126	7482	3.2	85.7	11.1	3.5	6.4
Gran	2	17	50.0	0.0	50.0	1.1	68.4	Zzzz	32	1465	3.1	12.5	84.4	62.7	2.5
Ital	2	55	50.0	0.0	50.0	1.2	1.5	Kits	68	3014	2.9	73.5	23.5	9.6	59.5
Kthi	2	47	50.0	50.0	0.0	0.5	72.7	Limb	37	8105	2.7	78.4	18.9	1.2	34.3
Perm	2	10	50.0	50.0	0.0	0.8	12.5	Dupl	40	2044	2.5	82.5	15.0	1.6	19.4
Wcho	2	22	50.0	0.0	50.0	1.0	56.5	Tang	1352	115 930	2.5	73.2	24.3	11.8	37.2
Hang	14 278	65 885	49.0	33.4	17.5	2.9	42.1	Brai	42	1813	2.4	81.0	16.7	0.5	42.8
Brah	35	1004	48.6	25.7	25.7	20.1	9.2	Yiii	1392	134 304	2.2	79.1	18.8	11.5	34.9
Avst	11	209	45.5	27.3	27.3	1.0	43.4	Armi	6	182	0.0	100.0	0.0	0.5	25.6
Java	12	915	41.7	41.7	16.7	3.5	1.2	Batk	3	24	0.0	100.0	0.0	0.5	0.0
Cyrl	80 795	596 194	40.1	26.3	33.6	2.2	82.4	Bhks	7	269	0.0	85.7	14.3	0.7	41.0
Cham	10	413	40.0	30.0	30.0	40.2	99.4	Bopo	28	1072	0.0	85.7	14.3	0.7	57.6
Geor	687	3492	39.9	42.6	17.5	0.9	14.8	Buhd	1	7	0.0	0.0	100.0	2.4	0.0
Knda	748	10 261	38.4	25.9	35.7	5.0	44.0	Cakm	10	342	0.0	50.0	50.0	0.9	31.8
Gujr	842	6173	37.8	46.2	16.0	2.9	7.1	Cher	26	1321	0.0	46.2	53.8	11.5	3.7
Bamu	253	7939	36.8	49.4	13.8	12.2	61.3	Cpmn	40	1879	0.0	60.0	40.0	11.0	98.7
Osge	25	496	36.0	48.0	16.0	33.6	99.1	Cprt	14	367	0.0	50.0	50.0	0.9	55.4
Thai	18 020	146 225	34.7	41.3	24.0	12.2	72.0	Dogr	2	8	0.0	50.0	50.0	0.9	0.0
Mero	3	76	33.3	33.3	33.3	0.8	37.3	Elba	3	13	0.0	66.7	33.3	3.0	71.8
Pauc	6	141	33.3	33.3	33.3	1.1	59.3	Hano	1	44	0.0	100.0	0.0	0.6	0.0
Sogd	3	39	33.3	33.3	33.3	1.1	80.5	Hatr	2	24	0.0	100.0	0.0	0.8	45.0
Toto	3	60	33.3	33.3	33.3	1.1	6.1	Khar	8	406	0.0	50.0	50.0	7.5	1.6
Mtei	16	649	31.2	56.2	12.5	25.8	98.8	Lepc	6	261	0.0	66.7	33.3	63.2	0.3
Glag	10	644	30.0	40.0	30.0	5.0	2.6	Lyci	14	408	0.0	78.6	21.4	0.8	50.3
Vith	10	325	30.0	40.0	30.0	51.0	98.9	Mahj	4	115	0.0	50.0	50.0	1.0	34.8
Sinh	1215	22 198	29.9	41.2	29.0	1.3	74.6	Maka	3	85	0.0	66.7	33.3	0.9	87.2
Khmr	6436	22 846	29.5	6.7	63.9	8.2	39.7	Mand	1	22	0.0	0.0	100.0	1.5	100.0
Mlym	707	4004	28.3	63.6	8.1	0.7	16.3	Mani	4	153	0.0	100.0	0.0	0.8	63.5
Jpan	355	4988	27.9	23.7	48.5	6.4	5.6	Mroo	8	630	0.0	75.0	25.0	7.4	69.1
Laoo	1919	20 651	25.4	55.0	19.5	3.1	74.9	Nagm	2	32	0.0	50.0	50.0	1.0	65.6
Gong	4	92	25.0	75.0	0.0	0.7	7.8	Nand	2	22	0.0	0.0	100.0	2.8	93.4
Merc	12	296	25.0	50.0	25.0	55.9	12.5	Nbat	13	318	0.0	84.6	15.4	0.7	92.8
Thaa	8	310	25.0	62.5	12.5	0.8	15.1	Orkh	6	537	0.0	66.7	33.3	1.0	90.7
Hani	1704	53 901	24.8	35.7	39.6	18.4	36.3	Osma	5	253	0.0	80.0	20.0	65.1	0.7
Adlm	9	60	22.2	55.6	22.2	1.0	16.4	Ougr	2	49	0.0	100.0	0.0	0.5	20.8
Tibt	92	4014	20.7	44.6	34.8	19.7	77.1	Phli	3	98	0.0	66.7	33.3	0.6	18.6
Ahom	5	72	20.0	60.0	20.0	0.8	39.0	Phlp	1	62	0.0	0.0	100.0	1.0	34.9
Gonm	5	121	20.0	80.0	0.0	0.5	45.2	Phnx	3	81	0.0	100.0	0.0	0.7	46.6
Ogam	5	198	20.0	60.0	20.0	0.7	30.6	Prti	1	40	0.0	100.0	0.0	0.3	7.1
Cans	109	4788	19.3	53.2	27.5	16.3	61.0	Rjng	1	40	0.0	100.0	0.0	0.3	100.0
Cari	17	1359	17.6	64.7	17.6	15.5	0.8	Rohg	8	522	0.0	62.5	37.5	31.8	99.5
Hung	18	442	16.7	66.7	16.7	18.5	98.7	Samr	3	122	0.0	100.0	0.0	0.3	19.0
Kali	12	160	16.7	83.3	0.0	0.7	30.8	Sarb	1	54	0.0	100.0	0.0	0.7	94.7
Lana	84	1685	16.7	51.2	32.1	4.1	43.5	Shaw	3	29	0.0	100.0	0.0	0.4	16.7
Olck	6	90	16.7	66.7	16.7	0.9	96.4	Shrd	23	1179	0.0	78.3	21.7	14.4	1.6
Dsrt	13	295	15.4	46.2	38.5	13.5	97.8	Sind	1	3	0.0	0.0	100.0	669.3	0.0
Sidd	47	1157	14.9	68.1	17.0	0.7	90.8	Sogo	3	71	0.0	100.0	0.0	0.8	12.3
Kawi	7	165	14.3	57.1	28.6	1.0	30.9	Sora	2	167	0.0	50.0	50.0	0.7	64.6
Mult	7	212	14.3	85.7	0.0	0.6	52.6	Soyo	4	77	0.0	75.0	25.0	0.8	61.9
Copt	195	4699	13.3	76.4	10.3	4.1	5.7	Tagb	5	79	0.0	80.0	20.0	0.8	45.0
Tavt	8	205	12.5	75.0	12.5	0.7	71.7	Takr	4	265	0.0	50.0	50.0	0.6	9.1
Modi	25	537	12.0	76.0	12.0	0.6	48.1	Tfng	6	80	0.0	33.3	66.7	1.5	50.8
Xsux	437	33 886	11.7	70.5	17.8	9.8	34.9	Tglg	2	29	0.0	100.0	0.0	0.9	61.5
Newa	26	838	11.5	61.5	26.9	0.8	83.4	Tirh	3	130	0.0	100.0	0.0	0.5	60.3
Diak	9	157	11.1	55.6	33.3	0.8	77.6	Tnsa	8	191	0.0	50.0	50.0	0.9	57.6
Lisu	9	310	11.1	66.7	22.2	53.0	0.2	Ugar	1	6	0.0	100.0	0.0	0.7	0.0
Phag	37	812	10.8	64.9	24.3	0.7	81.5	Vaii	48	6385	0.0	52.1	47.9	20.5	39.4
Hmng	10	309	10.0	90.0	0.0	0.7	96.1	Xpeo	20	713	0.0	85.0	15.0	0.7	26.6
Hmnp	10	523	10.0	70.0	20.0	16.3	0.8	Yezi	6	105	0.0	16.7	83.3	2.3	70.1
Mong	93	3060	9.7	48.4	41.9	10.4	16.8	Zanb	5	383	0.0	60.0	40.0	7.6	2.3
...	...	...	...	...	...	...	...	All	817 177	4 021 097	72.0	14.0	14.0	4.6	42.1

Table 21: Exploratory prompt: LLM instruction following metrics sorted by adherence: 64-token input sequence limit. Number of PUA characters omitted from LLM prompt.