A Bit of a Problem: Measurement Disparities in Dataset Sizes Across Languages

Catherine Arnett*[1], **Tyler A. Chang***[2,3], **Benjamin K. Bergen**[2]
[1] Department of Linguistics,
[2] Department of Cognitive Science,
[3] Halıcıoğlu Data Science Institute
UC San Diego
{ccarnett, tachang, bkbergen}@ucsd.edu

Abstract

How should text dataset sizes be compared across languages? Even for content-matched (parallel) corpora, UTF-8 encoded text can require a dramatically different number of bytes for different languages. In our work, we define the byte premium between two languages as the ratio of bytes used to encode content-matched text in those languages. We compute byte premiums for 1155 languages, and we use linear regressions to estimate byte premiums for other languages. We release a tool to obtain byte premiums for any two languages, enabling comparisons of dataset sizes across languages for more equitable multilingual model development and data practices.

Keywords: multilinguality, datasets, low-resource languages.

1. Introduction

Large language datasets serve as the foundation for modern natural language technologies. However, an often ignored question is how to compare dataset sizes across languages. For standard multilingual language models such as XLM-R, BLOOM, and XGLM, dataset sizes are reported in bytes (Conneau et al., 2020; Scao et al., 2022; Lin et al., 2022).[1] However, content-matched (i.e. parallel) text in two languages does not generally have the same size in bytes, with some languages taking over $5\times$ as many bytes as others (§3).

Here, we compute **byte premiums** (cf. tokenization premiums in Petrov et al., 2024), the ratios of bytes taken to encode text in 1155 different languages. We find that these byte premiums are highly correlated across datasets. We fit linear regressions to estimate byte premiums for languages not included in our parallel datasets, and we release a simple Python tool to retrieve or predict the byte premium between any two languages.[2] Our work enables comparisons of dataset sizes across languages, with implications for equitable multilingual model development and resource distribution.

*Equal contribution.

[1]Dataset sizes are also often reported in tokens, which depend on model-specific tokenizers and which exhibit similar cross-language disparities to bytes (Petrov et al., 2024).

[2]https://github.com/catherinearnett/byte-premium-tool

2. Related Work

Using UTF-8 encoding, which is by far the most widespread text encoding (Davis, 2012), characters take between one and four bytes to encode (Unicode Consortium, 2022). Numbers and Latin characters without diacritics are one byte, and all non-Latin scripts use two or more bytes per character. This alone introduces a disparity in measured dataset sizes in bytes (Costa-jussà et al., 2017), but it must be balanced with the fact that different scripts encode different amounts of "information" per character. For example, Mandarin has high UTF-8 bytes-per-character, but it generally requires fewer characters than Latin-script languages to encode the same content. To account for this tradeoff, previous work has used parallel text, finding that byte-level tokenizers encode parallel text in some languages using more "tokens" (bytes) than others ("tokenization premiums"; Petrov et al., 2024). We tie these results to dataset storage and training dataset size measurement, we compute the byte premium for 1155 languages, and we present a method to predict the byte premium for novel languages. All our results use UTF-8 encoded text.

3. Computing Byte Premiums

In this section, we calculate the **byte premium** $\mathrm{BP}_{A/B}$ for different language pairs, which we define as the ratio of bytes taken to encode a comparable amount of information in language A relative to language B. For example, if A on average takes twice as many UTF-8 bytes to encode the same information (parallel text) as B, then $\mathrm{BP}_{A/B}$ would be 2.0. These byte premiums are useful when mea-

SIGUL2024 Workshop, pages 1–9

suring "how much" content is in each language in a corpus. In multi-parallel corpora, we note that the byte premiums must satisfy:

$$\mathrm{BP}_{A/B} = \frac{\mathrm{Bytes}_A}{\mathrm{Bytes}_C} * \frac{\mathrm{Bytes}_C}{\mathrm{Bytes}_B} = \frac{\mathrm{BP}_{A/C}}{\mathrm{BP}_{B/C}} \quad (1)$$

This implies that if the byte premium is known for every language relative to some language C, then all pairwise byte premiums are determined. Thus, we only calculate a single byte premium $\mathbf{BP}_A = \mathrm{BP}_{A/C}$ per language, all relative to reference language C. We use $C =$ English as our reference language, but using any other reference language C_0 would simply multiply all our byte premiums by a constant BP_{C/C_0}. In later sections, we refer to byte premiums relative to English unless otherwise noted. In contrast to Petrov et al. (2024), calculating a single byte premium per language allows byte premiums to be used for multilingual corpora beyond just pairwise corpora.[3]

3.1. NLLB

Computing byte premiums requires parallel corpora in the desired languages. We first use NLLB (Costa-jussà et al., 2022), a dataset of pairwise parallel text segments in 188 languages. We sample the first 100K parallel text segments for each language pair (A, B), and we compute $\mathrm{BP}_{A/B}$ as the mean ratio of bytes used in language A versus B, averaged over all segments. This produces a byte premium value for every language pair.

To fit a single byte premium $\mathrm{BP}_A = \mathrm{BP}_{A/C}$ for each language relative to a reference language C (in our case English), we minimize the mean squared error of $\mathrm{BP}_A/\mathrm{BP}_B$ relative to the ground truth $\mathrm{BP}_{A/B}$ (Equation 1) over all language pairs (A, B). In other words, we fit 188 byte premium values (one per language) based on all 2656 pairwise byte premium values. Fitting these single byte premiums ensures that Equation 1 holds for all pairs.

Byte premiums computed from NLLB are reported in Appendix Table A.1. For example, Burmese has byte premium 5.10, so on average it takes $5.10\times$ as many UTF-8 bytes to encode text in Burmese versus English. These byte premiums are consistent when computed from different subsets of the NLLB corpus, with Pearson's $r > 0.999$ for byte premiums computed from ten disjoint subsets of 10% of the NLLB corpus. Notably, byte premiums computed from only 100 lines of text per language pair correlate with the byte premiums computed from the full NLLB dataset with Pearson's $r = 0.955$, indicating that byte premiums can be computed from fairly small parallel corpora.

[3]For example, if Equation 1 does not hold, then English-Mandarin and Arabic-Mandarin byte premiums could produce conflicting comparable dataset sizes when adding Mandarin data to an English+Arabic corpus.

	NLLB	FLORES	Bible	UDHR
FLORES	**0.919**		**0.938**	0.737
Bible	**0.921**	**0.938**		0.177
UDHR	0.592	0.737	0.177	

Table 1: Pearson correlations between byte premiums calculated from different datasets. Correlations are high between NLLB, FLORES, and the Bible.

3.2. Other Parallel Corpora

For comparison, we also compute byte premiums from three multi-parallel corpora: FLORES-200 (Costa-jussà et al., 2022; 204 languages), the Bible (eBible, 2023; 1027 languages), and the Universal Declaration of Human Rights (Vatanen et al., 2010; UDHR; 241 languages). For each language A in each dataset, we compute $\mathrm{BP}_A = \mathrm{Bytes}_A/\mathrm{Bytes}_C$ relative to reference language $C =$ English. Because each dataset is comprised of parallel text across all included languages, these byte premiums already satisfy Equation 1.

Computed byte premiums are highly correlated between NLLB, FLORES, and the Bible (Table 1; Pearson's $r > 0.90$), suggesting that byte premiums are fairly consistent across datasets. We posit that lower correlations with UDHR byte premiums may be because the UDHR corpora are much shorter (roughly twenty total lines of text) and potentially more domain-specific than the other corpora. For this reason, we do not use UDHR in later sections.

3.3. Byte Premiums After Compression

Interestingly, we find that byte premiums persist after compression with the common compression algorithm `gzip` (at maximum compression level 9). When byte premiums are computed from the compressed FLORES corpora, they correlate strongly with the uncompressed byte premiums (Pearson's $r = 0.890$). However, the scale of variation across languages reduces substantially after compression; for example, uncompressed byte premiums of 4.0 are roughly analogous to compressed byte premiums of 1.7 (i.e. compressed data in that language takes only $1.7\times$ as many bytes as the reference language rather than $4.0\times$ as many bytes; Appendix B). This suggests that standard compression algorithms reduce but do not fully alleviate disparities in dataset storage sizes across languages.

4. Predicting Novel Byte Premiums

In many cases, we may want to compute the byte premium for a language A outside of our existing datasets. If a single parallel text is available from A to any language B in our datasets, then the byte premium can easily be calculated as (using reference language C as before):

$$\mathrm{BP}_A = \frac{\mathrm{Bytes}_A}{\mathrm{Bytes}_C} = \frac{\mathrm{Bytes}_A}{\mathrm{Bytes}_B} * \mathrm{BP}_B \tag{2}$$

However, there may be cases where no parallel text is available for language A. In this scenario, we can break the byte premium into (1) the mean bytes-per-character in A and C, and (2) the ratio of characters needed to express the same information in A and C (the "length ratio"):

$$\mathrm{BP}_A = \frac{\mathrm{Bytes}_A}{\mathrm{Bytes}_C} = \frac{\mathrm{Bytes}_A}{\mathrm{Chars}_A} * \frac{\mathrm{Chars}_A}{\mathrm{Chars}_C} * \frac{\mathrm{Chars}_C}{\mathrm{Bytes}_C} \tag{3}$$

The bytes-per-character ratio for A can be calculated with only monolingual text in A. We find that this ratio is highly consistent regardless of the dataset used. The computed bytes-per-character ratios correlate strongly (Pearson's $r > 0.99$) when calculated from any of NLLB, the Bible, or FLORES with 20, 200, or 2000 lines of text. Given the consistency of these bytes-per-character ratios, we find it efficient to break byte premiums down as in Equation 3 such that we only need to predict the length ratio between languages.

4.1. Predicting Length Ratios

We use linear regressions including language family, script (writing system), script type (e.g. alphabet vs. logography), and entropy over characters to predict the length ratio $\mathrm{Chars}_A/\mathrm{Chars}_C$ for a language A relative to the reference language $C =$ English. From the predicted length ratio, we can use Equation 3 to calculate the predicted byte premium for language A. Our results use length ratios, bytes-per-character ratios, and character entropies computed from NLLB, FLORES, or the Bible when available, in order of decreasing priority.[4]

Language Family We predict that typological features (e.g. inflection patterns or morpho-syntactic distinctions) may drive differences in length ratios. Languages that are in the same language family are more likely to share typological features due to their shared historical origin (Moravcsik, 2012).

Script and Script Type Some writing systems may encode higher information content per character than others (e.g. Chinese characters; Perfetti and Liu, 2005), which leads to low length ratios, because the same content takes fewer characters to write. We separate scripts into four script types (alphabet, abjad, abugida, and logography; Appendix C), and we use script type as a predictor for length ratio. We also consider the specific script as a nested predictor (e.g. Latin vs. Cyrillic).

Character Entropy It has been proposed that languages with fewer phonemes (contrastive sounds) in their inventories have longer words, because it requires more sounds per word to generate the number of contrastive sound sequences necessary to communicate (Nettle, 1995).[5] Using the same logic, we predict that a language that tends to use fewer unique characters will require longer character sequences to express information (a high length ratio). We operationalize the number of unique characters in a language as the entropy over the character probability distribution in that language. A higher entropy indicates either a more even distribution over characters or a distribution over more characters. Similar to bytes-per-character ratios (§4), the entropy over characters is highly stable across datasets, even computed from as few as 20 lines of text (Pearson's $r > 0.90$ for the same datasets as §4).

We fit linear regressions to predict length ratios from three different subsets of our predictors. This allows us to predict novel byte premiums depending on the available information about the novel language. We consider the following three subsets: (I) character entropy, language family, script, and script type, (II) character entropy, script, and script type, and (III) character entropy and script type. The predicted length ratios can be used to predict byte premiums using Equation 3.

5. Evaluating Byte Premium Predictions

We validate the byte premium predictions from our linear regressions by looping through languages with known byte premiums (from NLLB, FLORES, or the Bible, in that order of priority), evaluating the byte premium prediction for that language when holding it out from regression fitting.[6] We report

[4] As with byte premiums, the choice of reference language C only multiplies all length ratios by a constant. NLLB length ratios are computed in the same way as byte premiums, but using characters instead of bytes. We obtain similar regression results using length ratios, bytes-per-character ratios, and character entropies computed from NLLB, FLORES, or the Bible (Appendix D).

[5] We also measure the number of phonemes per language (PHOIBLE; Moran et al., 2014), but it does not help predict length ratios ($R^2 = 0.002$). Therefore we do not include it in our linear regressions.

[6] To prevent skew of regression coefficients, we clip byte premiums to a maximum of 4.0 (three languages; Appendix A).

	Regression		
	I	II	III
Scripts with count ≥ 5	**0.261**	0.288	0.290
Scripts with count < 5	0.770	0.739	**0.589**

Table 2: RMSEs when predicting byte premiums using different regressions, for languages with common and uncommon scripts.

the root mean squared error (RMSE) for the three linear regressions described in the previous section (I, II, and III). We compute separate RMSEs for (1) languages whose script is shared by less than five languages in our datasets, and (2) languages whose script is shared by five or more languages in our datasets. Languages whose script is uncommon may have more poorly fitted script coefficients (and potentially language family coefficients), so we might expect them to exhibit larger byte premium prediction errors.

Results are reported in Table 2. For languages with common scripts (scripts with count ≥ 5), the regressions improve as predictors are added (III, II, then I). For these languages, RMSEs reach 0.261, indicating that the predicted byte premiums are on average approximately 0.261 away from the true byte premiums.

As expected, we also find that languages with uncommon scripts (scripts with count < 5) have higher errors in their predicted byte premiums, indicating that their script and family coefficients are poorly fitted. Likely due to these poorly fitted coefficients, for those languages, the regression with the lowest validation error is regression III, which only includes character entropy and script type as predictors. The validation RMSE is 0.589, indicating that predicted byte premiums for languages with uncommon scripts are on average approximately 0.589 away from the true byte premiums. Given that byte premiums can range from below 0.75 to over 5.00, even this simple regression is a substantial improvement over a naive assumption that languages take equal bytes to encode information (i.e. byte premium 1.0).

6. Introducing the Tool

Finally, we introduce a Python tool that returns pre-computed or predicted byte premiums for any language pair. The tool is available at `https://github.com/catherinearnett/byte-premium-tool`. If both input languages are in our set of 1155 languages, the pairwise byte premium is computed from Equation 1 using our pre-computed byte premiums. Otherwise, the byte premium is computed from a user-provided parallel text (if available). If no parallel text is available, the tool asks for a small monolingual corpus in the novel language(s), from which it can compute the character entropy and bytes-per-character ratio per language, to use in the regressions from §4. Following the validation results in §5, the tool uses regression **I**, **II**, or **III** (in order of decreasing priority) for languages with common scripts. For languages with uncommon scripts, regression III is always used. Aside from character entropy (which is computed from the user-provided monolingual text), regression III requires only the script type for the novel language(s), which can easily be found on sites such as Wikipedia. Thus, our tool provides a simple interface from which to obtain the pairwise byte premium between any two languages, enabling easy dataset size conversions.

7. Discussion and Conclusion

Measuring Dataset Sizes One implication of our work is that researchers currently may overestimate the amount of data that multilingual NLP models are trained on for non-Latin script languages (languages with high byte premiums). These languages are often already underrepresented in NLP (van Esch et al., 2022). For example, if it is reported that a model is trained on 1GB of Georgian data, then based on its byte premium of 4.34 relative to English, we should consider the model to be effectively trained on the Georgian equivalent of about 230MB of English data.

As a preliminary investigation into whether scaling training data quantities by byte premiums per language is indeed a "better" measure of training data quantity, we use this scaled measure to predict multilingual language model performance on various per-language benchmarks. Across models and tasks, we find that the scaled data proportions do predict performance in different languages better than reported proportions, but not significantly ($p = 0.13$; see Appendix E for details).

Byte-Level Tokenization Our results also have implications for dataset tokenization. Previous work has argued that byte-level tokenizers enable more uniform treatment of different languages in a model (Zhang and Xu, 2022; Xue et al., 2022), but our byte premiums demonstrate that some languages may still be at a disadvantage with byte-level tokenizers. Tokenization length inequalities can lead to higher costs, longer latencies, and restricted effective context lengths for some languages (Ahia et al., 2023; Petrov et al., 2024), in this case languages with high byte premiums.

Equitable Resource Costs Finally, languages with high byte premiums require more storage space than other languages to store comparable content, and they are likely to require higher bandwidth connections to transmit text content. In cases where storage is charged per (giga)byte or Internet connections are charged based on bandwidth and usage, uniform pricing rates across languages may lead to higher technology costs for low-resource language communities. While only a marginal step towards solving such issues, our work makes it possible to take byte premiums into account when measuring text data sizes across languages.

8. Acknowledgements

We would like to thank the other members of the UCSD Language and Cognition Lab for valuable discussion. Tyler Chang is partially supported by the UCSD Halıcıoğlu Data Science Institute graduate fellowship.

9. Bibliographical References

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

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

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

Marta R. Costa-jussà, Carlos Escolano, and José A. R. Fonollosa. 2017. Byte-based neural machine translation. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 154–158, Copenhagen, Denmark. Association for Computational Linguistics.

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

Peter T Daniels. 1990. Fundamentals of grammatology. *Journal of the American Oriental Society*, pages 727–731.

Mark Davis. 2012. Unicode over 60 percent of the web. Google Blog.

Guosheng Ding, Danling Peng, and Marcus Taft. 2004. The nature of the mental representation of radicals in Chinese: A priming study. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 30(2):530.

eBible. 2023. eBible.

Mandy Guo, Zihang Dai, Denny Vrandečić, and Rami Al-Rfou. 2020. Wiki-40B: Multilingual language model dataset. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2440–2452, Marseille, France. European Language Resources Association.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, Veselin Stoyanov, and Xian Li. 2022. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Steven Moran, Daniel McCloy, and Richard Wright. 2014. PHOIBLE online.

Edith A Moravcsik. 2012. *Introducing language typology*. Cambridge University Press.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15991–16111, Toronto, Canada. Association for Computational Linguistics.

Daniel Nettle. 1995. Segmental inventory size, word length, and communicative efficiency. *Linguistics*, 33(2):359–367.

Charles A Perfetti and Ying Liu. 2005. Orthography to phonology and meaning: Comparisons across and within writing systems. *Reading and Writing*, 18:193–210.

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

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. XCOPA: A multilingual dataset for causal commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2362–2376, Online. Association for Computational Linguistics.

Teven Le Scao, Angela Fan, Christopher Akiki, Elizabeth-Jane Pavlick, Suzana Ili'c, Daniel Hesslow, Roman Castagn'e, Alexandra Sasha Luccioni, Franccois Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Rose Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, et al. 2022. BLOOM: A 176b-parameter open-access multilingual language model. *arXiv*.

Unicode Consortium. 2022. The Unicode Standard.

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

Tommi Vatanen, Jaakko J. Väyrynen, and Sami Virpioja. 2010. Language identification of short text segments with n-gram models. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, pages 3423–3430. European Language Resources Association (ELRA).

Eric-Jan Wagenmakers and Simon Farrell. 2004. AIC model selection using Akaike weights. *Psychonomic bulletin & review*, 11:192–196.

Clay Williams and Thomas Bever. 2010. Chinese character decoding: a semantic bias? *Reading and Writing*, 23:589–605.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Mengjiao Zhang and Jia Xu. 2022. Byte-based multilingual NMT for endangered languages. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4407–4417, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Appendices

A. NLLB Byte Premiums

Byte premiums calculated from NLLB are reported in Table A.1.

B. Byte Premiums After Compression

Byte premiums after compression by `gzip`, compared to those before compression, are plotted in Figure B.1.

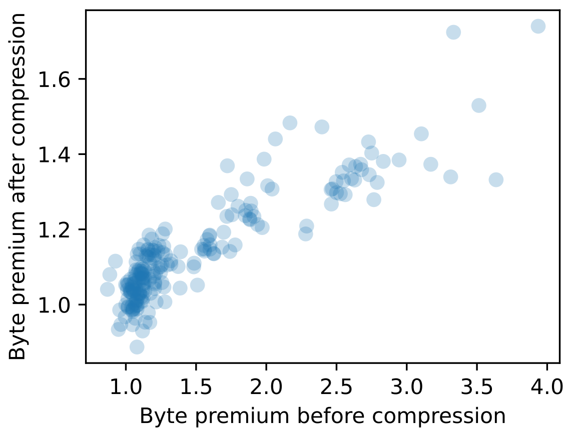

Figure B.1: Byte premiums before and after compression by `gzip`. Each point is a language's byte premium relative to English.

C. Writing System Types

Our regressions in §4 require the script type for each language. The four possible script types are described below.

Alphabet Alphabets are writing systems where each segment (either consonant or vowel) is represented by a symbol (Daniels, 1990). Latin script is one of the most widely used alphabets. Other alphabets include Greek, Cyrillic, and Mkhedruli (Georgian).

Abjad Abjads are writing systems which represent each consonant with a symbol (Daniels, 1990), but vowels are often not represented. Arabic and Hebrew are written with abjads, for example.

Abugida Abugidas, also sometimes referred to as *neosyllabaries*, represent consonant-vowel sequences, often with vowel notation secondary to consonant notation (Daniels, 1990). Examples of abugidas include Devanagari (e.g. Hindi), Ge'ez (e.g. Amharic), and Canadian syllabics (e.g. Ojibwe).

Logography Logographies are different from alphabets, abjads, and abugidas in that they represent semantic information as well as phonetic information. Chinese characters are the only logography that remains in use. The majority of Chinese characters are composed of one semantic component and one phonetic component (Williams and Bever, 2010). A relatively small number of characters are also pictographs or ideographs, representing only semantic information (Ding et al., 2004).

D. Validation from Different Datasets

In Table D.1, we report validation RMSEs for each regression (§5) when computing character entropies and bytes-per-character ratios from different datasets. Within each dataset, we separate the languages for which there are less than five other languages with the same script in the dataset from those which have five or more languages with the same script in the dataset. RMSE results are similar regardless of the dataset used to compute character entropies and bytes-per-character ratios.

		Regression		
		I	II	III
NLLB	Script ct. ≥ 5	0.201	0.244	0.240
	Script ct. < 5	0.700	0.744	0.637
Flores (20 lines)	Script ct. ≥ 5	0.203	0.246	0.250
	Script ct. < 5	0.682	0.557	0.538
Flores (200)	Script ct. ≥ 5	0.204	0.252	0.254
	Script ct. < 5	0.702	0.615	0.544
Flores (2000)	Script ct. ≥ 5	0.206	0.266	0.271
	Script ct. < 5	0.703	0.647	0.558
Bible (4 books)	Script ct. ≥ 5	0.272	0.294	0.298
	Script ct. < 5	0.766	0.680	0.577
Bible (1 book)	Script ct. ≥ 5	0.271	0.293	0.297
	Script ct. < 5	0.760	0.672	0.566

Table D.1: RMSEs when predicting byte premiums using different datasets to compute character entropies and bytes-per-character ratios. Results are separated into common and uncommon scripts.

E. Downstream Performance

To evaluate the impact of byte premiums on downstream performance, we compile reported training data proportions (measured based on bytes) per language for existing massively multilingual models. We adjust each training data proportion by dividing the reported proportion by the byte premium for that language. After re-scaling to sum to 1.0, this provides the estimated effective proportion of data for each language. If adjusted data proportions are indeed "better" estimates of effective data quantities, then we expect them to predict downstream task performance better than the original reported training data proportions.

We evaluate ten models from three model families: XGLM (Lin et al., 2022), BLOOM (Scao et al., 2022), and mT0 (Muennighoff et al., 2023). We compile results from XGLM 7.5B, four sizes of BLOOM (560M, 1.1B, 3B, 7.1B), and five sizes of mT0 (small, base, large, xl, xxl). We use benchmark scores from five multilingual benchmarks: XStoryCloze (Lin et al., 2022), XCOPA (Ponti et al., 2020), XNLI (Conneau et al., 2018), Wikipedia next word prediction (Guo et al., 2020), and XWinograd (Muennighoff et al., 2023). These benchmarks cover 22 languages: Arabic, Bulgarian, German, Greek, English, Estonian, French, Haitian Creole, Hindi, Indonesian, Italian, Japanese, Burmese, Portuguese, Russian, Spanish, Swahili, Telugu, Turkish, Urdu, Vietnamese, and Chinese (simplified and traditional). Benchmark scores are compiled from the Big Science evaluation results on Hugging Face.[7]

We fit two linear mixed effects models. Each predicts the benchmark score for each language (all scores between 0.0 and 1.0) from the training data proportion for that language (either the original proportion or those scaled according to our byte premiums) as well as language family, with random intercepts for model and task. We calculate the AICs of the two non-nested models, along with their relative

[7]https://huggingface.co/datasets/bigscience/evaluation-results

log likelihoods (Wagenmakers and Farrell, 2004). While the the data proportions scaled by byte premiums better predict benchmark performance (lower AIC and higher log likelihood), it is not a significant difference ($p = 0.13$), using significance testing as in Wagenmakers and Farrell (2004). This non-significance may be because there are many other factors that impact downstream performance apart from dataset size. A larger meta-analysis would lead to more reliable inferences.

Language	Byte premium	Language	Byte premium	Language	Byte premium
ace_latn	1.2419926	hye_armn	1.7241548	oci_latn	1.0146652
afr_latn	1.0373004	ibo_latn	1.3451287	ory_orya	2.5109372
aka_latn	1.5750612	ilo_latn	1.0765437	pag_latn	1.0439418
als_latn	1.1673181	ind_latn	1.1788023	pan_guru	2.2208951
amh_ethi	1.7210862	isl_latn	1.1543925	pbt_arab	1.7366557
arb_arab	1.4651134	ita_latn	1.0669230	pes_arab	1.5973940
asm_beng	2.5264323	jav_latn	1.1468920	plt_latn	1.1512264
ast_latn	1.7490516	jpn_jpan	1.3220250	pol_latn	1.0774161
awa_deva	2.7014324	kab_latn	1.0287174	por_latn	1.0979270
ayr_latn	1.0976628	kac_latn	1.3451812	quy_latn	1.1639224
azb_arab	1.4901878	kam_latn	1.2177037	ron_latn	1.1151666
azj_latn	1.0761036	kan_knda	2.6420061	run_latn	1.1193204
bak_cyrl	2.2716371	kas_arab	1.7762307	rus_cyrl	1.8228284
bam_latn	1.2569819	kas_deva	2.5259810	sag_latn	1.1632489
ban_latn	1.2695671	kat_geor	4.3381046	san_deva	2.5428913
bem_latn	1.1553301	kbp_latn	1.4408085	sat_beng	2.1131754
ben_beng	2.4308225	kea_latn	0.7821679	shn_mymr	2.8224643
bho_deva	2.5153669	khk_cyrl	1.8046135	sin_sinh	2.4463506
bod_tibt	2.6040539	khm_khmr	3.9051643	slk_latn	1.0415468
bug_latn	1.2279017	kik_latn	1.2930516	slv_latn	0.9722273
bul_cyrl	1.8123562	kin_latn	1.1340740	sna_latn	1.1192729
cat_latn	1.0926706	kir_cyrl	1.9635570	snd_arab	1.5880165
ceb_latn	1.1134194	kmr_latn	1.0351712	som_latn	1.4224149
ces_latn	1.0358867	knc_arab	2.5022926	sot_latn	1.1661078
ckb_arab	1.6521034	knc_latn	1.1769876	spa_latn	1.0838621
ckb_arab	1.6521034	kor_hang	1.2933602	srp_cyrl	1.4249495
cym_latn	1.0265667	lao_laoo	2.7071355	sun_latn	1.0970417
dan_latn	1.0211031	lij_latn	1.1438412	swe_latn	1.0210256
deu_latn	1.0537171	lin_latn	1.1393024	swh_latn	1.0696621
dik_latn	1.1239299	lit_latn	1.0300780	tam_taml	2.7292892
diq_latn	0.9590188	ltg_latn	1.0028570	taq_latn	1.2093634
dyu_latn	1.1545521	ltz_latn	1.2253827	tat_cyrl	1.8543562
dzo_tibt	3.2736977	lug_latn	1.2175185	tel_telu	2.6198705
ell_grek	1.9673049	luo_latn	1.0358323	tgk_cyrl	1.7469201
ewe_latn	1.0783440	lus_latn	1.1689564	tgl_latn	1.1176348
fao_latn	1.1557437	lvs_latn	1.2070388	tir_ethi	1.7631466
fij_latn	1.2107666	mag_deva	2.5555142	tuk_latn	1.7850561
fin_latn	1.0589051	mai_deva	2.3896953	tur_latn	1.0444815
fon_latn	1.5413204	mal_mlym	2.8852389	tzm_tfng	1.9259158
fra_latn	1.1742064	mar_deva	2.4793638	uig_arab	2.3082357
fur_latn	1.0672371	min_latn	0.9497956	ukr_cyrl	1.7514786
fuv_latn	1.1109194	mkd_cyrl	1.8349890	umb_latn	1.1673612
gla_latn	0.9934613	mlt_latn	1.0884567	urd_arab	1.7079714
gle_latn	1.9749562	mni_beng	3.0027416	uzn_latn	1.6455453
glg_latn	1.0590246	mos_latn	1.1413713	vie_latn	1.3493725
guj_gujr	2.1627759	mri_latn	1.1826053	wol_latn	1.0787309
hau_latn	1.1766293	mya_mymr	5.1034592	xho_latn	1.1988860
heb_hebr	1.3555346	nld_latn	1.0516739	ydd_hebr	1.8074376
hin_deva	2.3701629	nob_latn	0.9977426	yor_latn	1.3750599
hrv_latn	0.9897218	npi_deva	2.4202344	zsm_latn	1.1438457
hun_latn	1.0199851	nus_latn	1.2935254	zul_latn	1.1639372

Table A.1: NLLB byte premiums. The byte premium for eng_latn is 1.0. Each language code is comprised of the ISO 639-3 (language) and ISO 15924 (script) codes separated by an underscore.