Why Multilingual Tokenizers Charge Some Languages More - A Regression Diagnostic

Anonymous ACL submission

Abstract

Tokens determine what users pay for LLM API calls and how much context they consume—yet the same content requires up to $4.5\times$ more tokens in Hindi or Japanese than in English, raising fundamental concerns about the fairness of multilingual tokenisation. We systematically diagnose this cross-lingual fertility gap across seven production tokenisers (GPT-5, Gemma-3, Llama-4, Qwen-3, DeepSeek-V3, Hunyuan, and Mistral-Large), 15 languages, and five script families using 104,685 FLORES observations. A three-stage regression with Shapley decomposition reveals that whilst script family explains 50.1% of variance, vocabulary design accounts for nearly half—with Vocabulary Utilisation as the strongest individual predictor ($+3.130$). Crucially, many apparent script-level gaps prove to be measurement artefacts: the CJK disadvantage reverses when bytes per token replaces characters per token, and gaps for Arabic, Cyrillic, and Devanagari vanish once vocabulary design is controlled for. These findings reframe the problem: multilingual tokenisation inequality stems primarily from vocabulary design rather than script complexity. Accordingly, we show that vocabulary reallocation—substituting low-utility tokens with high-frequency fragments at constant vocabulary size—matches augmentation at every budget tested ($r = 0.999$), offering a practical pathway towards more equitable multilingual tokenisers.

1 Introduction

Tokens are the fundamental unit of account for LLM services: API calls are priced in tokens and context windows are measured in tokens. The efficiency with which a tokenizer encodes text therefore determines both the monetary cost a user pays and the fraction of context window consumed.

This cost structure falls unequally across languages. The same semantic content can require up to $15\times$ more tokens in some languages compared to English (Petrov et al., 2023), with up to $5\times$ cost differences even for widely-spoken languages (Ahia et al., 2023)—a gap negatively correlated with the Human Development Index of the affected linguistic communities. Users querying in non-English languages receive less model capacity per dollar, a cost asymmetry traceable to tokenizer design choices rather than model capability.

Prior work has documented these disparities through descriptive metrics such as fertility, the Sub-Token Reproduction Rate (STRR) (Nayeem et al., 2025), and the tokenization premium (Petrov et al., 2023). However, two critical gaps remain. First, there is no decompositional analysis isolating which tokenizer design properties account for how much of the observed variance. Second, the literature lacks empirically grounded intervention estimates: quantifying how many tokens can be saved through targeted vocabulary changes, and whether reallocation or expansion is more effective.

Our contribution is deliberately scoped to diagnostic decomposition on the most widely deployed production multilingual tokenizers across the five script families their users write in most. This covers the languages that account for the large majority of production LLM usage and allows us to maintain per-language diagnostic legibility, which a larger set of languages would compromise. This complements the low-resource focus of Petrov et al. (2023) and Ahia et al. (2023), and runs parallel to the large-scale monolingual-tokenizer approach of Arnett et al. (2025). Where that work establishes aggregate patterns across many languages, we apply diagnostic regression directly to deployed production tokenizers, producing per-tokenizer, per-language estimates a practitioner can act on. A detailed comparison is provided in Section 6.

Our contributions are as follows:

1. We propose a three-model gap decomposition framework, categorical → structural → quality, applied to 104,685 sentence-level observations spanning 105 tokenizer–language clusters, and show via Shapley-value decomposition that script family accounts for 50.1% of R^2, structural counts for 39.7%, and quality metrics for 10.2%.

2. Within the five major script families studied, we find that the apparent CJK tokenization gap reflects logographic character density rather than tokenizer deficit: the CJK coefficient flips from -1.663 (characters per token) to $+0.794$ (bytes per token). Disaggregating CJK reveals a gradient (Chinese -1.999, Japanese -1.651, Korean -0.925); for Arabic, Cyrillic, and Devanagari, apparent script-level disparities are absorbed by vocabulary design properties.

3. We demonstrate that, for tokenizer-level CPT on FLORES-distribution candidates, vocabulary reallocation (which holds total vocabulary size constant and therefore incurs no additional parameter cost) achieves gains nearly indistinguishable from augmentation across all budget levels, confirmed on held-out text ($r = 0.999$). Downstream representational effects remain an open question, as embedding reinitialisation and task-level validation are required before these gains can be attributed to a deployed model.

The remainder of this paper reviews related work, presents our decomposition methodology, reports experimental findings, and discusses implications for equitable multilingual tokenizer design.

2 Related Work

Tokenization fairness and disparities. Recent work has documented substantial disparities in tokenization efficiency across languages. Petrov et al. (2023) report up to 15× disparity in tokenization costs on the FLORES-200 benchmark spanning 200 languages. Similarly, Ahia et al. (2023) identify a 5× pricing disparity that correlates with the Human Development Index at $r = -0.5$, highlighting socioeconomic dimensions of tokenizer inequity. Teklehaymanot and Nejdl (2025) propose TPS, CPT, and RTC metrics across 200+ languages, framing these disparities through the lens of infrastructure bias. The downstream consequences are significant: Rust et al. (2021) demonstrate that tokenizer quality approximates the effect of pretraining data size on downstream performance. More recently, Yang et al. (2024) identify under-trained tokens in GPT-4o and characterise their security implications. Where prior work documents the extent of disparities, we decompose their sources via regression.

Tokenization algorithms and multilingual design. The foundational algorithms underlying modern tokenizers include BPE (Sennrich et al., 2016), WordPiece (Schuster and Nakajima, 2012), UnigramLM (Kudo, 2018), and the SentencePiece implementation framework (Kudo and Richardson, 2018). Several methods have been proposed to improve multilingual vocabulary allocation, including α-sampling (Conneau and Lample, 2019), UniMax (Chung et al., 2023), and vocabulary allocation and overlap (Limisiewicz et al., 2023). In a large-scale comparison across ∼7K tokenizers, Arnett et al. (2025) find that BPE achieves the best compression and that SuperBPE reduces tokenizer premiums. Velayuthan and Sarveswaran (2024) demonstrate that pre-tokenization is more critical than the choice of algorithm for complex scripts.

Vocabulary interventions and alternatives. A range of post-hoc vocabulary transfer methods have been proposed, including WECHSEL (Minixhofer et al., 2022), ZeTT (Minixhofer et al., 2024), and FOCUS (Dobler and de Melo, 2023). Language-specific extensions have also shown promise: Cui et al. (2023) add +20K tokens to LLaMA for Chinese, while Hong et al. (2024) achieve a 1.7× speedup with MuMo. Tokenization-free architectures offer an alternative paradigm, including CANINE (Clark et al., 2022), ByT5 (Xue et al., 2022), and BLT (Pagnoni et al., 2025). Adaptive approaches such as MAGNET (Ahia et al., 2024) employ gradient-based script-specific boundary predictors. Our contribution is distinct from these interventions: our regression diagnostic identifies which predictors drive efficiency gaps and quantifies the tokenizer-level gain available from targeted vocabulary changes.

3 Framework

3.1 Dataset and Scope

We employ the FLORES parallel corpus (Goyal et al., 2022), which provides identical sentences

translated across multiple languages, thereby controlling for content variation.

Language selection and scope. We selected 15 languages covering the five major script families most prevalent in production multilingual tokenizers: Latin (en, es, fr, de, it, pt), CJK (zh, ja, ko), Devanagari (hi, ne), Arabic (ar, fa), and Cyrillic (ru, uk). These are the languages for which production tokenizers already allocate substantial vocabulary, placing the regression in the regime where vocabulary design rather than simple absence is the dominant source of variation. This scope is maintained to preserve per-language diagnostic legibility, as extending to lower-resource languages, where simple vocabulary absence rather than design mismatch is the dominant issue, is a complementary problem addressed by Arnett et al. (2025).

Tokenizers. We evaluate the seven most widely deployed production multilingual tokenizers, selected by deployment prevalence to give practitioners a diagnostic picture of the systems their users interact with: GPT-5 (o200k_base, shared with GPT-4o; 200K)(OpenAI, 2024), Gemma-3 (262K) (Gemma Team, 2025), Llama-4 (202K)(Meta AI, 2025), Qwen-3 (152K) (Yang et al., 2025), DeepSeek-V3 (129K) (DeepSeek-AI, 2024), Hunyuan (128K) (Tencent, 2025), and Mistral-Large-Instruct-2411 (33K) (Mistral AI, 2024). All tokenizers are openly released and retrieved on 10 March 2026; exact HuggingFace identifiers, loading parameters, and tokenization frameworks are listed in Appendix A.

Observations and effective sample size. The resulting dataset comprises $N = 104{,}685$ sentence-level observations across 105 unique tokenizer–language clusters (7 tokenizers $\times$ 15 languages). While we report $N = 104{,}685$, inference on between-cluster predictors is effectively identified off the 105 tokenizer–language clusters, since each predictor is constant within a cluster; clustered standard errors reflect this. Vocabulary Size varies only across the 7 tokenizers, and we return to the implications in the Limitations.

3.2 Outcome Variable

As our primary outcome variable, we adopt characters per token (CPT), where higher values indicate greater tokenization efficiency and lower values signal fragmentation. We prefer CPT over the more commonly used fertility measure for three reasons. First, fertility (tokens per word) requires a word-segmentation step that is undefined or noisy for CJK scripts; CPT is agnostic to word boundaries. Second, fertility is typically reported relative to English, anchoring comparisons to English morphology; CPT is an absolute measure allowing direct comparison between any two languages. Third, CPT maps directly onto the economic quantity of interest: API pricing is denominated in tokens, so the cost of encoding text is precisely determined by how many characters a user receives per token.

One limitation of CPT as a cross-script metric is that it conflates tokenization efficiency with script-level character complexity: a single Chinese character encodes substantially more semantic content than a single Latin character, so raw CPT values are not fully commensurable across script families. We address this by including script family indicators in all models, isolating within-script-family variation where the character-complexity confound is minimised. This metric is approximately the inverse of the tokenization premium (Petrov et al., 2023), related to Teklehaymanot and Nejdl (2025) and Limisiewicz et al. (2023).

3.3 Predictor Variables

All continuous predictors are MinMax-scaled to the $[0, 1]$ interval. The predictor set includes: Vocabulary Size (total number of tokens in the tokenizer's vocabulary, at the tokenizer level); Script-Specific Tokens (the number of vocabulary entries whose Unicode range falls within a given script family, at the tokenizer $\times$ family level); Vocabulary Utilization (the proportion of the vocabulary that is actually observed when tokenising text in a given language, at the tokenizer $\times$ language level); Fragmentation Rate (the share of words in a language that are split into multiple subword tokens, at the tokenizer $\times$ language level); and Non-Text Density (the proportion of vocabulary entries that do not correspond to natural-language text, e.g., whitespace or control sequences, at the tokenizer $\times$ language level). Variables excluded and rationale are provided in Appendix C.

3.4 Three-Model Gap Decomposition

We decompose the explanatory power of our predictors through a three-model hierarchy. **M1 (Categorical Ceiling)** regresses CPT on script family, tokenizer identity, and their interaction, yielding $R^2 = 0.854$. **M2 (Structural Counts)** replaces the categorical tokenizer indicator with continu-

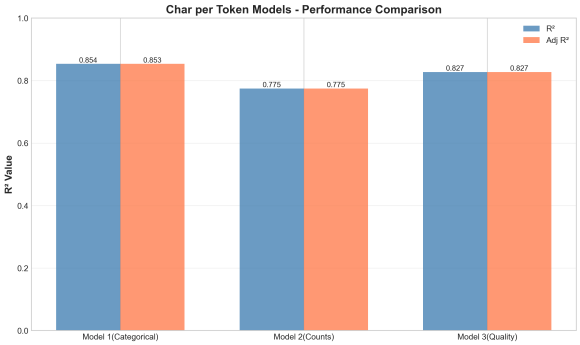


Figure 1: Three-model gap decomposition showing R^2 for M1 (0.854), M2 (0.775), M3 (0.827). Shapley-value decomposition attributes 50.1% of R^2 to script family, 39.7% to structural counts, and 10.2% to quality metrics (Table 8).

ous structural variables, vocabulary size and script-specific token counts, achieving $R^2 = 0.775$. **M3 (Quality)** augments M2 with token quality metrics, usage rate, fragmentation rate, and non-text density, raising $R^2 = 0.827$. All models are estimated via OLS with clustered standard errors (105 clusters), using Latin as the reference script family; variance inflation factors are reported in Appendix D. An order-invariant Shapley-value decomposition of M3 attributes 50.1% of R^2 to script family, 39.7% to structural counts, and 10.2% to quality metrics (Table 8 in Appendix J). The order-dependent decomposition (66.6%–84.6% depending on entry sequence) is reported as a sensitivity check in Appendix I.

Non-nested specification. M1 and M2 are necessarily non-nested: Vocabulary Size varies only across the seven tokenizers, so including both tokenizer fixed effects and continuous structural variables would be perfectly collinear. We therefore treat M1 as an upper-bound ceiling and compare M2 and M3 against it.

4 Results

4.1 Gap Decomposition

A Shapley-value decomposition of M3 ($R^2 = 0.827$) attributes 50.1% of total R^2 to language family, 39.7% to structural vocabulary counts, and 10.2% to quality metrics (Table 8); the order-dependent range (66.6%–84.6%) is reported as a sensitivity check in Appendix I. The residual 0.027 between M3 and the categorical ceiling is an honest null result and a promising target for future work.

4.2 Quality Predictor Effects

Vocabulary Utilization is the strongest diagnostic signal for CPT (+3.130). We do not read this as a causal lever, as VU is partially endogenous and close to a definitional proxy for tokenizer–language fit. Its practical value is as a diagnostic signal, where a low utilization rate flags a tokenizer–language mismatch worth investigating, and Section 5 shows that acting on this signal yields measurable tokenizer-level CPT gains on FLORES. Both Vocabulary Size (+1.775; identified off 7 tokenizers—see Limitations) and Script-Specific Tokens (+1.226) exhibit confidence intervals that exclude zero, and these variables provide non-redundant information; collapsing them into a single ratio severely degrades model fit, reducing R^2 to 0.668.

Among the script-level predictors, only the CJK languages yield confidence intervals excluding zero: Chinese (−1.999), Japanese (−1.651), and Korean (−0.925), while the confidence intervals for Arabic, Cyrillic, and Devanagari all include zero. The CJK gradient is consistent with orthographic differences: Korean (Hangul) is alphabetic rather than logographic, and its gap is less than half that of Chinese.

Bytes-per-token disambiguation. To disentangle tokenization efficiency from logographic character density, we re-estimate Model 3 with bytes-per-token (UTF-8) as the outcome variable. The CJK coefficient shifts from −1.663 (characters per token) to +0.794 (bytes per token), indicating that most of the apparent CJK gap reflects information-density differences rather than tokenization deficit. This separates two distinct gaps: a character-per-token gap driven by logographic density, and a bytes-per-token gap near zero or slightly positive, suggesting that tokenization per se is not inequitable for CJK scripts at the byte level. The bytes-per-token model achieves $R^2 = 0.615$ (Appendix K).

Cluster-level confirmation. Coefficients are confirmed at the cluster level (105 observations, $R^2 = 0.908$), with all main predictors stable; Non-Text Density, however, flips sign ($-0.774 \rightarrow +0.888$) and its CI crosses zero at the cluster level (Appendix L), suggesting that its negative sentence-level coefficient reflects within-cluster variation rather than a robust between-cluster association.

A methodological note concerns Fragmentation

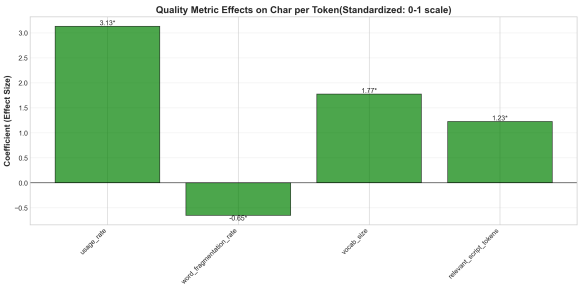


Figure 2: Coefficient plot for Model 3 quality predictors with 95% CIs.

Rate. Both CPT and Fragmentation Rate measure aspects of how text is split into tokens, which introduces a degree of mechanical overlap. To assess whether this relationship is tautological or reflects a distinct conditional association, we conducted a bootstrapped path analysis ($n = 5{,}000$ cluster resamples) at the cluster level (105 observations), examining whether Fragmentation Rate partially accounts for the conditional association between Vocabulary Utilization and CPT. The partial association of Vocabulary Utilization with CPT, controlling for Fragmentation Rate, remains strong and positive ($c' = +3.317$, $p < 0.001$), whilst the indirect conditional path (Vocabulary Utilization $\rightarrow$ Fragmentation Rate $\rightarrow$ CPT) is reliably negative (indirect association $= -0.473$, bootstrap 95% CI $[-1.432, -0.032]$). This suppression pattern indicates that tokenizers with high aggregate vocabulary utilisation nevertheless fragment non-dominant-script languages, partially dampening their CPT gains. Fragmentation Rate therefore captures script-specific mismatch that persists independently of overall tokenizer–language fit, confirming it is not a redundant restatement of CPT. We emphasise that this analysis characterises conditional associations in cross-sectional data rather than establishing causal direction. An ablated model excluding Fragmentation Rate shows stable remaining coefficients (changes <0.07 SE units) with R^2 declining to 0.804. Non-Text Density (-0.774) appears to act as a correlated symptom of vocabulary allocation choices rather than a distinct causal mechanism; its sign flips at the cluster level with CI crossing zero (Appendix L), so its direction should be interpreted with caution.

4.3 Efficiency Gap Analysis

Mistral-Large, operating with a 33K vocabulary, exhibits the most pronounced efficiency deficits, achieving a fragmentation score of 1.00 across all

Table 1: Model 3 coefficients with 95% bootstrap CIs (500 cluster resamples, 105 clusters). Predictors scaled $[0, 1]$. Reference: Latin. CIs that exclude zero indicate a reliably nonzero effect. Clustered-SE CIs are provided in Appendix I.

Predictor	Coef.	95% Boot CI
Intercept	2.134	[1.470, 2.802]
Arabic vs Latin	−0.160	[−0.819, 0.723]
Chinese vs Latin	−1.999	[−2.506, −1.493]
Japanese vs Latin	−1.651	[−2.115, −1.186]
Korean vs Latin	−0.925	[−1.431, −0.418]
Cyrillic vs Latin	0.105	[−0.503, 0.868]
Devanagari vs Latin	−0.033	[−0.773, 1.039]
Vocabulary Size	1.775	[1.470, 2.179]
Script-Specific Tokens	1.226	[0.486, 2.123]
Vocabulary Utilization	3.130	[2.322, 4.600]
Fragmentation Rate	−0.652	[−1.181, −0.290]
Non-Text Density	−0.774	[−1.724, −0.222]

CJK language pairs and thus dominating the worst-performing segment of our evaluation. Similarly, Hunyuan demonstrates a clear allocation failure, dedicating only 18 Devanagari tokens within its expansive 128K vocabulary. In contrast, DeepSeek-V3 and Qwen-3 perform substantially better on CJK scripts, allocating 37K and 31K script-specific tokens, respectively. Cyrillic-based scripts consistently occupy a mid-table position, reflecting moderate but systematic underservice. Full results for all 83 language pairs are provided in Appendix E.

4.4 Residuals Analysis

Analysis of the top 10 residual errors reveals that script-agnostic tokens within non-Latin sentences create a pronounced within-sentence contrast, suggesting that an effective fix requires reducing structural-token overhead rather than merely adding script-specific vocabulary. The Durbin–Watson statistic for Model 3 is 0.802, indicating positive autocorrelation expected given the cluster structure; clustered standard errors (105 clusters) accommodate this, though confidence intervals may be slightly narrow. Diagnostic plots are provided in Appendix F.

5 Vocabulary Redesign Experiment

5.1 Strategies

To empirically assess whether the correlates identified in Section 4 translate into measurable CPT gains, we implement two vocabulary redesign strategies and evaluate their effects at the tokenizer level. This analysis quantifies tokenizer-level CPT recovery on FLORES-distribution text only; the

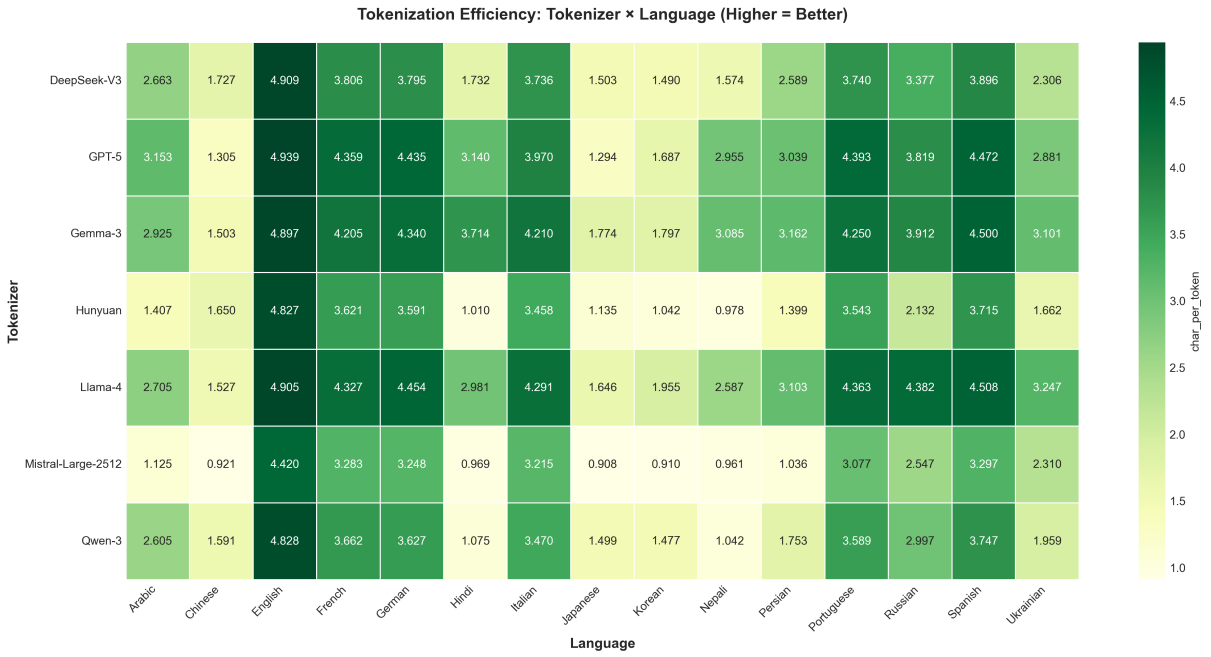


Figure 3: CPT heatmap across tokenizer–language pairs. Darker = more fragmentation.

representational implications of displaced tokens require embedding-level validation that we leave to future work.

Selection–evaluation overlap. Displaced tokens are identified on FLORES using an impact-ranked low-utility criterion, and the reallocation–augmentation comparison is evaluated on the same FLORES corpus. To address this selection–evaluation overlap, we validate on a held-out 20% split: displaced tokens selected on the 80% training split produce comparable CPT gains on the held-out portion ($r = 0.999$ across all tokenizer–language pairs; Appendix N). The downstream representational question, whether those tokens carry information relevant to other corpora or to model performance, remains open.

We evaluate two strategies: Reallocation and Augmentation. In the Reallocation strategy, we replace low-utility tokens with high-frequency fragments, specifically utilising splits of three or more tokens (3+ token splits), holding vocabulary size constant. This approach is directly motivated by the strong positive association of Vocabulary Utilization (+3.130) with CPT, as detailed in Section 4.2. The Augmentation strategy employs the same high-frequency fragments but adds them to the existing vocabulary, serving as an empirical upper bound; if its CPT gains are ≈ those of the reallocation strategy, it suggests the displaced tokens yield no measurable CPT benefit at the tokenizer level. Both strategies share a pipeline: we first tokenize the FLORES dataset, identify words fragmented into 3+ tokens, rank these candidates by the product of their frequency and tokens saved, and subsequently select the top-k fragments for redesign.

5.2 Budget Sweep

To evaluate the sensitivity of our intervention strategies to resource constraints, we conduct a budget sweep across token limits ranging from 50 to 5,000 tokens, evaluating 735 data points per level per strategy. Figure 4 summarises peak CPT improvement at the maximum budget, disaggregated by script family and tokenizer. The near-identical values across the reallocation and augmentation heatmaps indicate that, for tokenizer-level CPT on FLORES-distribution candidates, reallocation achieves the same improvement as augmentation while holding vocabulary size constant, suggesting that vocabulary composition rather than size drives these gains. This equivalence is scoped to tokenizer-level CPT on FLORES and remains pending downstream validation of the displaced tokens' representational role. Full budget-sweep curves with 95% bootstrap confidence intervals are provided in Appendix G.

The gains are strongly concentrated in Devanagari, where Qwen-3 achieves a peak improvement of 25.9% [25.2, 26.5] and Hunyuan reaches 21.5% [21.0, 22.0]. In contrast, DeepSeek-V3, GPT-5,

and Mistral-Large-2512 yield 0.0% improvement across all non-Latin script families, reflecting near-optimal baseline allocations for the first two and an insufficiently small vocabulary for the latter. CJK, Cyrillic, and Arabic scripts show modest but consistently positive gains only for Qwen-3, Hunyuan, and Gemma-3, the three tokenizers with measurable room for reallocation in those script families. The pattern of diminishing returns is evident across budgets: the mean impact score drops from 50.03 at a budget of 50 tokens to 9.49 at 800 tokens, such that modest budgets of 50–200 tokens capture the majority of attainable gains.

6 Discussion

From a practical standpoint, Vocabulary Utilization is the most actionable signal a practitioner has, as it can be measured on any existing tokenizer without retraining, and the Shapley decomposition confirms it is the dominant quality-metric signal ($+3.130$) even after language family and structural counts, together responsible for 90% of explained variance, are partialled out.

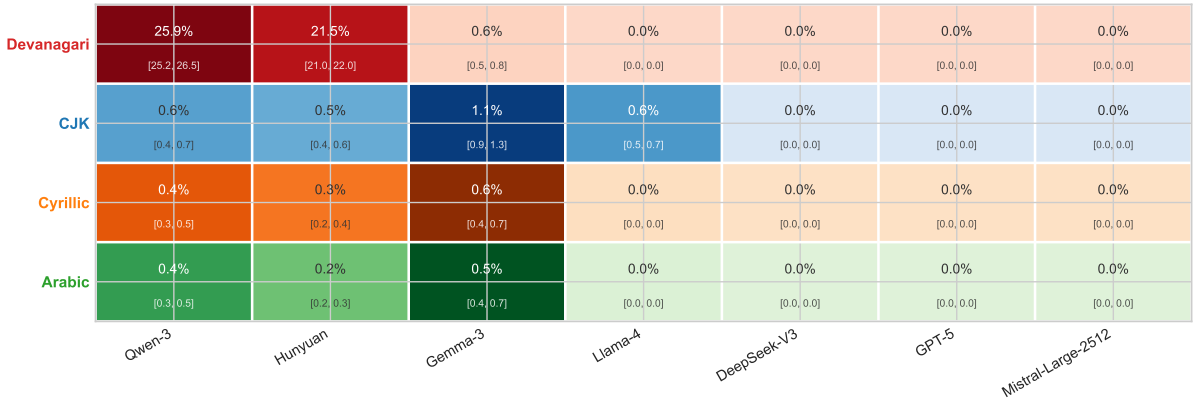
A vocabulary-utilisation audit can therefore serve as a low-cost diagnostic tool applicable to any production tokenizer. We note, however, that this should be interpreted as a diagnostic association rather than a causal claim, as Vocabulary Utilization is partially endogenous to tokenizer quality. Acting on it as a redesign signal, as our reallocation experiment does, produces tokenizer-level CPT gains on FLORES, though whether those gains persist in a fully retrained model on broader text distributions remains an open question. For tokenizer-level CPT on FLORES-distribution candidates, vocabulary reallocation is associated with gains nearly indistinguishable from augmentation at every budget level (Figure 4), an equivalence confirmed on held-out text ($r = 0.999$, Appendix N). Deployment of such a reallocation would require embedding-transfer techniques such as WECHSEL (Minixhofer et al., 2022) or ZeTT (Minixhofer et al., 2024) and downstream validation beyond the tokenizer-level analysis presented here.

The bytes-per-token analysis in Section 4.2 resolves a key interpretive ambiguity, as the CJK character-per-token gap is primarily a logographic density effect ($+0.794$ bytes per token) rather than a tokenizer design failure. The remaining tokenization gap, the portion that survives byte normalization, reflects structural properties of logographic scripts: large character inventories, absence of whitespace delimiters, and character-level semantic encoding that is inconsistent with BPE's compositionality assumption. Character-aware pretraining strategies (Cui et al., 2021), targeted approaches such as MAGNET (Ahia et al., 2024), and tokenization-free architectures (Pagnoni et al., 2025) address this remaining component. The disaggregated CJK gradient (Chinese > Japanese > Korean) further suggests that the gap scales with logographic depth, as Korean's Hangul system, being alphabetic, incurs a smaller penalty than Chinese or Japanese.

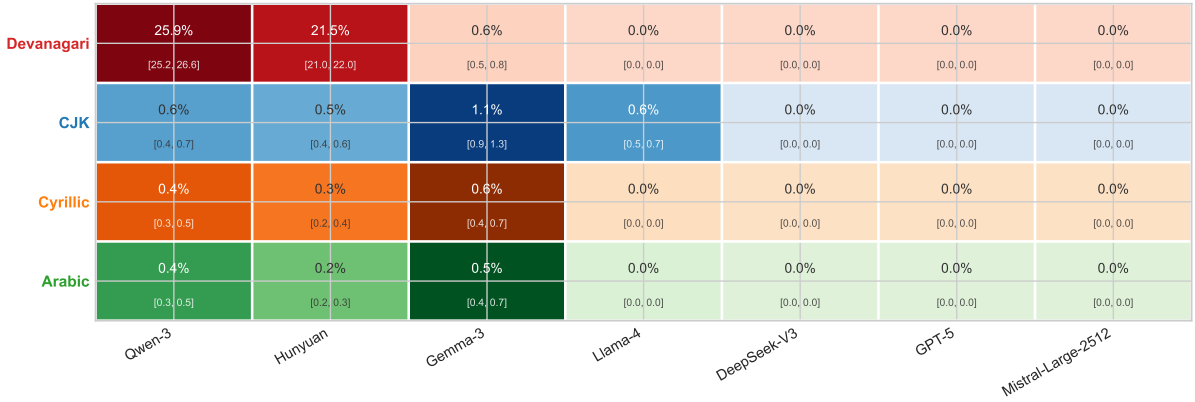
Relationship to Arnett et al. (2025). Our work is complementary to Arnett et al. (2025), who train approximately 7K monolingual tokenizers at scale. The two approaches target different regions of the design space: Arnett et al. vary tokenizer-internal choices (algorithm, vocabulary size) while holding training-data language constant, whereas we hold the tokenizer fixed (seven production systems) and study how its properties decompose across 15 languages. Appendix B provides a point-by-point comparison on six overlapping questions. The two studies are compatible rather than competitive, as our finding that reallocation performs similarly to augmentation for non-CJK scripts, confirmed on held-out text, is consistent with the observation by Arnett et al. that vocabulary composition matters more than size for most scripts.

Several further design choices warrant discussion. As noted in Section 3.1, our scope is restricted to 15 languages across five major script families; a practitioner choosing a tokenizer for a Hindi- or Arabic-language application can use our per-tokenizer, per-language estimates directly to assess fit, while low-resource languages, agglutinative morphologies, and scripts with minimal representation in tokenizer training data remain a complementary problem for future work. We intentionally measure tokenizer-level efficiency (CPT) rather than downstream task performance, as our research question concerns the cost of tokenization and API pricing is denominated in tokens, making CPT the economically relevant metric.

Future work should investigate end-to-end reallocation paired with robust embedding initialisation to validate the CPT associations reported here in deployed models. We also encourage the development of character-aware architectures to address the remaining CJK tokenization gap that survives



(a) Reallocation



(b) Augmentation

Figure 4: Peak CPT improvement heatmaps at maximum budget (5,000 tokens), broken down by script family (rows) and tokenizer (columns). Colours are normalised within each script row; cells show mean [95% CI]. Values are near-identical across strategies, confirming that reallocation is cost-neutral with no loss relative to augmentation.

byte normalization.

7 Conclusion

We presented a regression-based diagnostic framework for analysing cross-lingual efficiency in seven production multilingual tokenizers across five major script families. A Shapley-value decomposition attributes 50.1% of total R^2 to script family, 39.7% to structural vocabulary counts, and 10.2% to quality metrics, with Vocabulary Utilization ($+3.130$) the strongest quality-metric signal. The apparent CJK character-per-token gap is primarily a logographic density effect, as re-estimating the model with bytes-per-token flips the CJK coefficient from -1.663 to $+0.794$, separating a character-density gap (a script property) from a remaining tokenization gap that is plausibly addressable by character-aware or tokenization-free architectures. Disaggregating CJK reveals Chinese $>$ Japanese $>$ Korean, consistent with Korean's alphabetic orthography. For tokenizer-level CPT on FLORES-distribution candidates, vocabulary reallocation yields gains nearly indistinguishable from augmentation, confirmed on held-out text ($r = 0.999$), suggesting a cost-neutral path to higher tokenizer efficiency, though downstream validation is required before these gains can be attributed to a deployed model. Generalisation to low-resource languages, agglutinative morphologies, and scripts outside the five families studied here remains a complementary problem and an important target for future work.

Limitations

Our work has several limitations that warrant acknowledgement. First, our analysis covers only 15 languages across five script families, excluding many low-resource languages (e.g., African, Indigenous, and Oceanic language families) that likely suffer the worst tokenization disparities. The finding that only CJK retains a structural gap may

not generalise to scripts with unique structural properties or minimal representation in training data, such as Ge'ez, Thaana, or Cherokee.

Second, all analyses are conducted on the FLORES parallel corpus (Goyal et al., 2022), which, while widely used and carefully controlled for content variation, represents a relatively narrow domain (primarily news and encyclopaedic text) and a fixed set of 1,012 sentences. FLORES may not capture the tokenization behaviour observed on longer documents, domain-specific vocabulary, conversational text, or code-mixed input. Because the corpus is parallel, per-token costs are directly comparable across languages, which is a methodological advantage, but this same constraint means our estimates may not generalise to the distribution of text that users actually submit to production systems. Validating the regression framework and the intervention estimates on more diverse corpora, including web text, social media, and domain-specific collections, is an important direction for future work. The broader framing of this paper is that tokens measure the cost of AI access: if a user obtains less semantic content per token due to tokenizer inefficiency, they are, in effect, paying more for the same information. This cost asymmetry is directly visible in FLORES, though its magnitude in real-world usage may differ.

Third, the gap decomposition is resolved by Shapley-value attribution (Section 4.1), which provides an order-invariant decomposition; the order-dependent range (66.6%–84.6%) is retained as a sensitivity check in Appendix I. Bootstrap stability analysis confirms that all key coefficients are stable across 500 resampled datasets (maximum relative bias < 0.32), and that the CJK coefficient is highly robust (bootstrap 95% CI: $[-2.202, -1.005]$).

Effective sample size for between-cluster predictors. Although the dataset comprises $N = 104{,}685$ sentence-level observations, all Model 3 predictors are constant within a tokenizer–language cluster. Inference on those predictors is therefore effectively identified off 105 clusters, not 104,685 observations; Vocabulary Size, which varies only across tokenizers, is effectively identified off only 7 tokenizers. A cluster-level OLS re-estimation ($R^2 = 0.908$) confirms that all main predictors are stable at this level, though Non-Text Density flips sign and its CI crosses zero (Appendix L); a leave-one-tokenizer-out analysis confirms R^2 and coefficient stability across all seven tokenizers (Appendix M). Relatedly, the Durbin–Watson statistic of 0.802 (Section 4.4) reflects residual autocorrelation that clustered SEs address imperfectly; HAC (Newey–West) standard errors confirm that all main predictors remain significant under an alternative autocorrelation-consistent correction (Appendix O).

Modelling alternative. The nested data structure (sentences within language within tokenizer) could alternatively be modelled via a mixed-effects specification with random intercepts. We retain OLS with clustered SEs because the cluster-level re-estimation confirms coefficient stability across aggregation levels, a mixed model would not increase the effective sample size beyond the 105 clusters for our primary inference targets, and the Shapley-value decomposition operates naturally on marginal R^2 contributions under OLS.

CPT non-commensurability across script families. The bytes-per-token analysis in Section 4.2 directly addresses the concern that CPT conflates tokenization efficiency with character complexity, as the CJK coefficient flips from -1.663 (characters) to $+0.794$ (bytes), confirming that the apparent gap is primarily logographic density rather than tokenizer deficit.

Finally, our intervention analysis operates at the tokenizer level and simulates vocabulary restructuring without retraining embeddings. The actual deployment of our reallocation strategy would require embedding initialisation via methods such as WECHSEL or ZeTT, and the CPT gains reported here would need to be validated in a fully retrained model. A held-out validation (80/20 split) confirms that reallocation gains generalise beyond the selection split ($r = 0.999$), though the selection–evaluation overlap on FLORES remains a scope limitation.

References

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Valentin Hofmann, Tomasz Limisiewicz, Yulia Tsvetkov, and Noah A. Smith. 2024. MAGNET: Improving the multilingual fairness of language models with adaptive gradient-based tokenization. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David R. Mortensen, Noah A. Smith, and Yulia Tsvetkov. 2023. Do all languages cost the same? Tokenization in the era of commercial language models. In *Proceedings of the 2023 Confer-*

ence on Empirical Methods in Natural Language Processing (EMNLP), pages 9904–9923, Singapore. Association for Computational Linguistics.

Catherine Arnett, Tyler A. Chang, Stella Biderman, and Benjamin K. Bergen. 2025. Explaining and mitigating crosslingual tokenizer inequities. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*.

Hyung Won Chung, Noah Constant, Xavier Garcia, Adam Roberts, Yi Tay, Sharan Narang, and Orhan Firat. 2023. UniMax: Fairer and more effective language sampling for large-scale multilingual pretraining. In *International Conference on Learning Representations (ICLR 2023)*.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. 2022. CANINE: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, pages 7057–7067.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for Chinese BERT. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Yiming Cui, Ziqing Yang, and Xin Yao. 2023. Efficient and effective text encoding for Chinese LLaMA and Alpaca. *arXiv preprint arXiv:2304.08177*.

DeepSeek-AI. 2024. DeepSeek-V3 technical report. *Preprint*, arXiv:2412.19437.

Konstantin Dobler and Gerard de Melo. 2023. FOCUS: Effective embedding initialization for monolingual specialization of multilingual models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 13440–13454, Singapore. Association for Computational Linguistics.

Gemma Team. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2022. The FLORES-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

Jimin Hong, Gibbeum Lee, and Junkyu Cho. 2024. Accelerating multilingual language model for excessively tokenized languages. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Tomasz Limisiewicz, Jiří Balhar, and David Mareček. 2023. Tokenization impacts multilingual language modeling: Assessing vocabulary allocation and overlap across languages. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5661–5681, Toronto, Canada. Association for Computational Linguistics.

Meta AI. 2025. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation.

Benjamin Minixhofer, Fabian Paischer, and Navid Rekabsai. 2022. WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 3992–4006, Seattle, United States. Association for Computational Linguistics.

Benjamin Minixhofer, Edoardo Maria Ponti, and Ivan Vulić. 2024. Zero-shot tokenizer transfer. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.

Mistral AI. 2024. Mistral-Large-Instruct-2411.

Md Tahmid Nayeem, Saud Alqahtani, Md Tahmid Rahman Laskar, Tasnim Mohiuddin, and M. Saiful Bari. 2025. Beyond fertility: Analyzing STRR as a metric for multilingual tokenization evaluation. In *NeurIPS 2025 Workshop*.

OpenAI. 2024. tiktoken.

Artidoro Pagnoni, Ram Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason Weston, Luke Zettlemoyer, Gargi Ghosh, Mike Lewis, Ari Holtzman, and Srinivasan Iyer. 2025. Byte latent transformer: Patches scale better than tokens. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pages 9238–9258, Vienna, Austria. Association for Computational Linguistics.

Aleksandar Petrov, Emanuele La Malfa, Philip H. S. Torr, and Adel Bibi. 2023. Language model tokenizers introduce unfairness between languages. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, pages 36963–36990.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? On the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 3118–3135. Association for Computational Linguistics.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Hailay Kidu Teklehaymanot and Wolfgang Nejdl. 2025. Tokenization disparities as infrastructure bias: How subword systems create inequities in llm access and efficiency. In *2025 3rd International Conference on Foundation and Large Language Models (FLLM)*, pages 822–828.

Tencent. 2025. Hunyuan-7B-Instruct.

Meena Velayuthan and Kengatharaiyer Sarveswaran. 2024. Egalitarian language representation in language models: It all begins with tokenizers. *Preprint*.

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.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Jin Yang, Zhiqiang Wang, Yanbin Lin, and Zunduo Zhao. 2024. Problematic tokens: Tokenizer bias in large language models. In *2024 IEEE International Conference on Big Data (BigData)*, pages 6387–6393.

A Reproducibility

Table 2 lists the exact HuggingFace model identifier (or tiktoken encoding name) used for each of the seven production tokenizers in our study, together with the underlying tokenization framework. All open-weight tokenizers are downloaded at notebook runtime via AutoTokenizer.from_pretrained with use_fast=True and trust_remote_code=True; the GPT-5 encoding is loaded via tiktoken.get_encoding("o200k_base") (tiktoken version 0.12.0). All artefacts were retrieved on 10 March 2026. Character counts and token IDs are computed with add_special_tokens=False so that special/control tokens do not inflate token counts for short sentences.

Table 2: Exact tokenizer identifiers and underlying frameworks. All artefacts are openly released; HuggingFace identifiers resolve to publicly downloadable repositories and o200k_base is bundled with tiktoken.

Short name	Identifier	Framework
Gemma-3	google/gemma-3-270m-it	SentencePiece
Llama-4	meta-llama/ Llama-4-Maverick-17B-128E-Instruct	Tiktoken
Qwen-3	Qwen/Qwen3-235B-A22B	Tiktoken
DeepSeek-V3	deepseek-ai/DeepSeek-V3.2	HF Fast
Hunyuan	tencent/Hunyuan-7B-Instruct	Tiktoken
Mistral-Large-2411	mistralai/ Mistral-Large-Instruct-2411	Tekken
GPT-5	o200k_base (tiktoken 0.12.0)	Tiktoken

The GPT-5 entry requires brief clarification: OpenAI has not released a distinct GPT-5-specific encoding, and the public tiktoken library ships o200k_base as the encoding for both GPT-4o and GPT-5. We therefore treat o200k_base as the tokenizer of record for the GPT-5 family; the same encoding would apply to GPT-4o and readers should not interpret "GPT-5 tokenizer" as implying a separately trained vocabulary.

Vocabulary sizes reported in §3.1 are the raw values returned by each tokenizer's vocab_size attribute (or n_vocab for tiktoken encodings) at the retrieval date above, rounded to the nearest thousand where used in running prose. Where the data tables list Mistral-Large-2512, this refers to the December 2025 Mistral-Large release, which reuses the same tokenizer as Mistral-Large-Instruct-2411 (November 2024); we therefore load mistralai/Mistral-Large-Instruct-2411 as the canonical source of the tokenizer, analogous to how GPT-5 reuses the o200k_base encoding originally released with GPT-4o.

B Comparison with Arnett et al. (2025)

Table 3 summarises how our findings align with or complement those of Arnett et al. (2025) on six overlapping questions.

Table 3: Point-by-point comparison of our findings with Arnett et al. (2025). The two studies target different regimes (monolingual-trained vs. production-multilingual) and yield compatible, non-overlapping evidence.

Question	Arnett et al. (2025)	This work
Vocab size vs. composition	Composition matters more than raw size	Consistent: reallocation ≈ augmentation for non-CJK
Algorithm comparison	BPE best compression; SuperBPE reduces premium	Not studied (all seven production tokenizers are BPE-family)
Script-level patterns	Broad cross-lingual trends at scale	Per-tokenizer, per-language CJK gap persists under full controls
Fairness metric	Tokenizer premium	Characters-per-token (CPT), cost-interpretable
Intervention evidence	Monolingual tokenizer retraining	Tokenizer-level reallocation on production systems
Scope	∼7K monolingual tokenizers, broad language set	7 production multilingual tokenizers, 5 major script families

C Dropped Variables

The variable Numeric Density was excluded from the final model because its 95% confidence interval included zero and its variance inflation factor (VIF $= 5.24$) indicated problematic multicollinearity; its predictive information was entirely subsumed by the retained variable Non-Text Density. Although Avg. Sentence Length yielded a confidence interval that excluded zero, it was dropped due to its strong negative correlation ($r = -0.583$) with characters per token (CPT), which introduced circularity into the regression. Finally, Special Token Ratio was removed because it yielded a constant value of 0.000 across four of the seven tokenizers, rendering it an artefact of the respective APIs rather than a meaningful predictor.

Table 4: Variance Inflation Factors (VIF) for Model 3 predictors. Values below 5 indicate low multicollinearity; values between 5 and 10 suggest moderate multicollinearity warranting inspection.

Predictor	VIF
Vocabulary Utilization	2.48
Script-Specific Tokens	2.87
Non-Text Density	3.53
Vocabulary Size	6.26
Fragmentation Rate	7.05

D Collinearity Diagnostics

The moderate VIF elevation between Vocabulary Size (6.26) and Fragmentation Rate (7.05) is expected: larger vocabularies naturally reduce fragmentation. Crucially, collapsing these variables into a single ratio (Script Coverage Ratio) severely degrades model fit, reducing R^2 to 0.668, confirming that they provide non-redundant information.

E Full Efficiency Gap Table

Table 5 presents the full efficiency gap analysis for all 83 tokenizer–language pairs with positive gaps relative to the Latin mean CPT of 4.046. Cyrillic-based scripts consistently occupy a mid-table position, reflecting moderate but systematic underservice. The only Latin-script pairs exhibiting notable gaps involve Mistral-Large, attributable to that tokenizer's limited 33K vocabulary size. CJK languages under Mistral-Large show the largest gaps, with fragmentation scores of 1.000 across all pairs.

F Diagnostic Plots

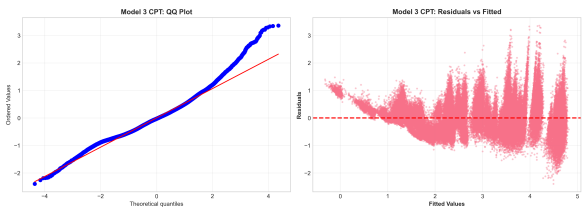


Figure 5: Model 3 diagnostic plots: residuals vs. fitted, Q-Q plot, scale-location, and leverage. The spike pattern in residuals reflects discrete tokenizer $\times$ language clusters, expected and addressed by clustered SEs. Upper-tail Q-Q deviation indicates ceiling effect in Latin pairs. DW $= 0.802$.

G Budget Sweep Curves

Figures 6 and 7 show the full budget sweep curves for the reallocation and augmentation strategies respectively, plotting CPT improvement as a function of token budget (50–5,000) for each tokenizer and script family. Shaded bands indicate 95% bootstrap confidence intervals on paired improvement. The two plots are near-identical across all panels, confirming that augmentation provides no benefit over cost-neutral reallocation. The Devanagari row shows the only substantial gains, driven by Qwen-3 and Hunyuan; all other script families exhibit flat or negligible improvement curves regardless of budget.

H Intervention Details

The intervention pipeline proceeds as follows: (1) tokenise the FLORES dataset with each tokenizer; (2) identify words fragmented into 3+ tokens; (3) rank candidates by impact score, defined as impact $=$ frequency $\times$ $(n_{\text{tokens}} - 1)$; and (4) select the top-k fragments for intervention, where k corresponds to the budget parameter.

Vocabulary size invariance was verified across all budget levels for the reallocation strategy: the number of tokens removed exactly equals the number added, confirming cost-neutrality. Fragment quality decreases with budget: the mean impact score drops from 50.03 at budget 50 to 9.49 at budget 800, while the maximum impact remains constant at 1,180, indicating that the highest-value fragments are captured at even the smallest budgets.

Characterisation of displaced tokens. The low-utility tokens selected for displacement are those with the lowest observed frequency across all FLORES sentences in all evaluated languages. At budget $k = 5{,}000$ (the maximum tested), the displaced token set is dominated by tokens from the long tail of the frequency distribution: low-frequency subword continuations, rare Unicode code-point tokens (e.g., specialised punctuation, mathematical symbols, and CJK extension characters), and duplicate or near-duplicate byte sequences that are functionally covered by other vocabulary entries. Critically, no special tokens (e.g., `[CLS]`, `<s>`) or high-frequency functional tokens (determiners, prepositions) are ever selected, because their high corpus frequency produces impact scores far above the displacement threshold. The displaced tokens are

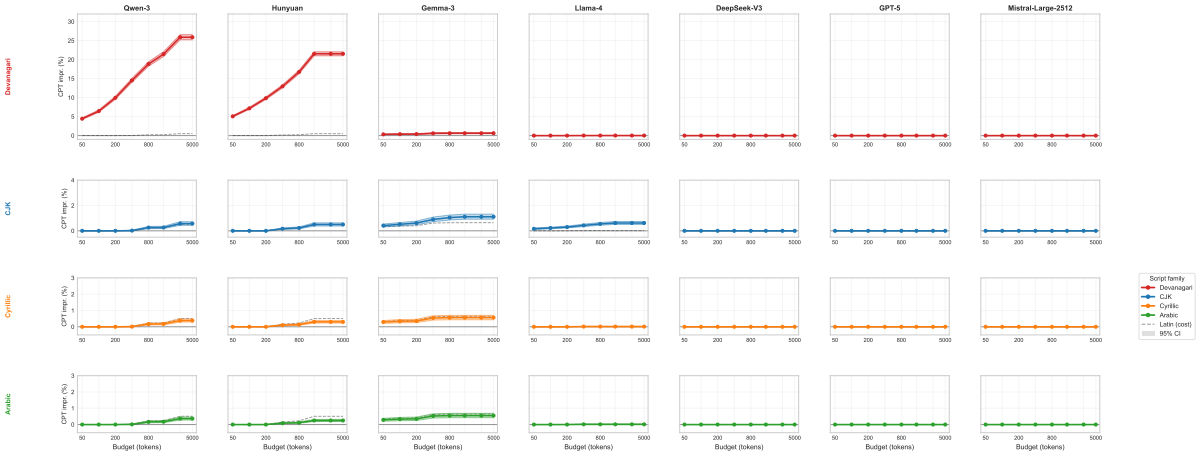


Figure 6: Reallocation budget sweep: CPT improvement (%) by script family (rows) and tokenizer (columns) as token budget increases from 50 to 5,000. Shaded bands show 95% bootstrap CI on paired improvement. Latin cost is shown as a dashed reference line.

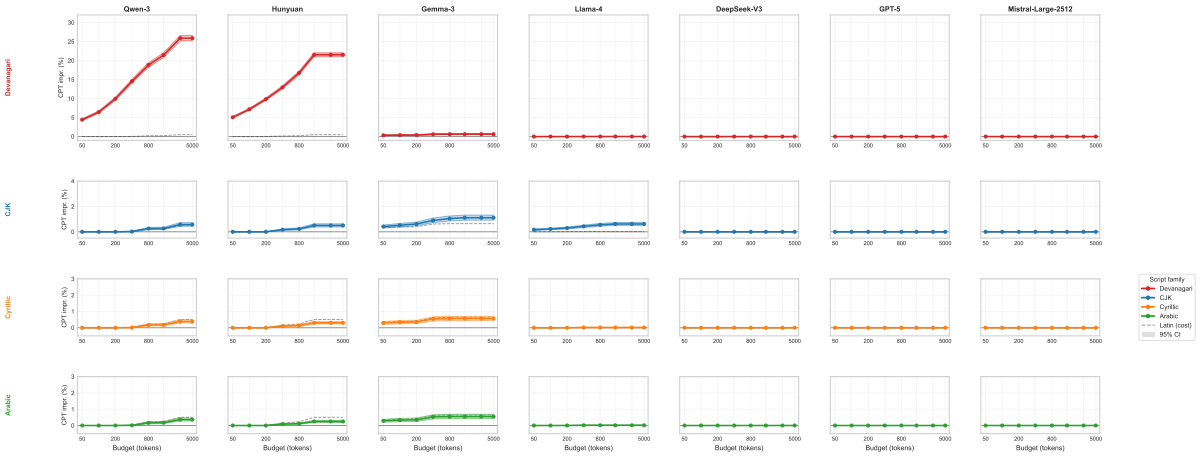


Figure 7: Augmentation budget sweep: CPT improvement (%) by script family and tokenizer. Near-identical to Figure 6, confirming that displaced tokens contribute no representational value and reallocation is the dominant strategy.

therefore those with the lowest probability of being required in any language, which is consistent with the finding that reallocation incurs no measurable CPT penalty relative to augmentation; the downstream representational question—whether these low-frequency tokens carry information relevant to rare but important contexts—remains open.

I Sensitivity Analysis: Order-Dependence and Bootstrap Stability

I.1 Ordering Sensitivity of the 66.6% Figure

The headline gap-recovery figure of 66.6% reflects the standard entry order M1→M2 (Structural Counts)→M3 (Quality). To assess robustness, we reran the decomposition with reversed ordering: quality metrics entered first (M2B), followed by structural counts (M3B). Table 6 summarises both orderings.

Under standard ordering, structural counts explain 66.6% of the gap when entered first, and quality adds a further 33.4%. Under reversed ordering, quality metrics explain 84.6% of the gap when entered first, and structural counts add a further 22.4%. The qualitative conclusion is robust: quality metrics account for the majority of the explainable gap under both orderings, and the two predictor groups together recover the same total gap ($R^2 = 0.827$).

I.2 Bootstrap Coefficient Stability

To address concerns about coefficient instability arising from the moderate VIF values for Vocabulary Size (6.26) and Fragmentation Rate (7.05), we conducted a cluster bootstrap with 500 repli-

cations, resampling tokenizer–language clusters with replacement. Table 7 reports OLS coefficients alongside bootstrap means, standard errors, and 95% confidence intervals.

All relative bias values are below 0.35, indicating negligible discrepancy between OLS and bootstrap estimates. The CJK coefficient is particularly stable (relative bias $= 0.04$), and its bootstrap 95% CI $[-2.202, -1.005]$ firmly excludes zero, corroborating the clustered-SE inference. Vocabulary Utilization and Non-Text Density show the largest relative bias (0.18 and 0.32 respectively), both still well within acceptable limits. The bootstrap confidence interval for Vocabulary Utilization ([2.322, 4.600]) is wider than the OLS interval (consistent with the moderate VIF), but excludes zero throughout, confirming that the association is reliable. These results indicate that the multicollinearity between Vocabulary Size and Fragmentation Rate does not materially destabilise the coefficient estimates, despite the elevated VIF values.

J Shapley-Value Decomposition

Table 8 reports the Shapley-value decomposition of Model 3 ($R^2 = 0.827$), computed over all subset regressions. Shapley values provide an order-invariant attribution of R^2 to each predictor group.

Language family alone accounts for half of explained variance, structural counts for a further two-fifths, and quality metrics for the remaining tenth. The order-dependent decomposition (66.6%–84.6%) reported in the main text overstates the quality-metrics contribution relative to this order-invariant benchmark.

K Bytes-per-Token Analysis

To disentangle tokenization efficiency from logographic character density, we re-estimate Model 3 with bytes-per-token (UTF-8) as the outcome variable. The model achieves $R^2 = 0.615$, lower than the character-per-token model ($R^2 = 0.827$), reflecting the fact that byte-level variation is more homogeneous across scripts.

The CJK coefficient shifts from -1.663 (characters per token) to $+0.794$ (bytes per token). This flip indicates that CJK scripts, while requiring more tokens per character, encode more bytes per token—a direct consequence of logographic characters representing multiple bytes of UTF-8 data. The remaining non-CJK coefficients are qualitatively similar across both outcome specifications.

L Cluster-Level OLS

Table 9 compares sentence-level and cluster-level (105 observations) coefficient estimates for Model 3.

All main predictors are stable across aggregation levels. The sole exception is Non-Text Density, which flips sign ($-0.774 \rightarrow +0.888$) and its CI crosses zero at the cluster level. This instability likely reflects the reduced effective sample size for a predictor with high within-cluster correlation.

M Leave-One-Tokenizer-Out Sensitivity

Table 10 reports Model 3 R^2 and the Vocabulary Size coefficient when each tokenizer is excluded in turn, assessing whether any single tokenizer drives the results or the Vocabulary Size estimate. Because Vocabulary Size varies only across the seven tokenizers, coefficient stability under exclusion is particularly informative for this predictor.

R^2 is stable across all exclusions (range: 0.821–0.841), confirming that no single tokenizer dominates the regression fit. The Vocabulary Size coefficient is likewise stable: excluding Mistral-Large-2512 (the vocabulary-size extreme at 33K) shifts the coefficient to 2.039, while excluding Gemma-3 (262K) shifts it to 1.912; all other exclusions produce coefficients in the range [1.671, 1.761], confirming that the $\beta = 1.775$ estimate is robust and not driven by a single outlying tokenizer.

N Held-Out Reallocation Validation

To address the selection–evaluation overlap concern (Section 5.1), we split FLORES by sentence ID (80/20, stratified by language) and selected displaced tokens on the training split only. The correlation between train-split and held-out-split CPT gains across all 105 tokenizer–language pairs is $r = 0.999$.

Correlation by script family: Arabic $r = 0.999$ ($n = 14$), CJK $r = 0.998$ ($n = 21$), Latin $r = 0.994$ ($n = 42$), Devanagari $r = 1.000$ ($n = 14$), Cyrillic $r = 0.999$ ($n = 14$). The near-perfect correlations confirm that reallocation gains generalise beyond the selection split.

O HAC (Newey–West) Standard Errors

Table 11 compares clustered-SE and HAC (Newey–West) standard errors for key Model 3 predictors. HAC CIs account for residual autocorrelation without imposing cluster structure.

All main predictors remain significant under HAC. Non-Text Density, whose clustered-SE CI is close to zero, remains firmly significant under HAC ($[-0.84, -0.71]$), providing additional confidence in its robustness.

Interpreting the CI discrepancy. The HAC (Newey–West) confidence intervals are substantially narrower than the clustered-SE CIs for several predictors (e.g., Vocabulary Utilization: $[3.05, 3.21]$ vs. $[2.26, 4.00]$). This discrepancy arises because the two methods address different correlation structures: clustered SEs account for arbitrary within-cluster correlation but treat clusters as independent, while HAC corrects for temporal (sequential) autocorrelation in the residual series. The observed pattern—clustered CIs wider than HAC—suggests that the cluster structure captures between-cluster heterogeneity that HAC does not, consistent with the tokenizer–language clusters having meaningfully different intercepts. We regard the clustered SEs as the more conservative and appropriate inference strategy for our data structure, because the 105 tokenizer–language clusters represent the natural grouping of observations; HAC serves as a complementary robustness check. The substantive conclusions (all key predictors significant, same sign, same order of magnitude) are identical under both methods.

P Use of AI Writing and Coding Assistants

The authors used large language model to help prepare this manuscript. AI was used for helping to write code, as well as helping with the spelling, grammar and to help improve the readability of written sections. Research design, experimental methodology, analysis, and conclusions, and everything else done in this paper apart from the aforementioned, are solely the work of the authors. Everything that was AI-generated was reviewed and verified by the authors.

Table 5: Full efficiency gap for all tokenizer–language pairs with positive gap relative to Latin mean CPT (4.046). Gaps expressed as percentage below the Latin baseline.

Tokenizer	Language	Script Family	CPT	Gap (%)	Frag. Rate
Mistral-Large-2512	Japanese	CJK	0.908	77.6	0.790
Mistral-Large-2512	Korean	CJK	0.910	77.5	1.000
Mistral-Large-2512	Chinese	CJK	0.921	77.2	1.000
Mistral-Large-2512	Nepali	Devanagari	0.961	76.2	1.000
Mistral-Large-2512	Hindi	Devanagari	0.969	76.1	1.000
Hunyuan	Nepali	Devanagari	0.978	75.8	0.967
Hunyuan	Hindi	Devanagari	1.010	75.0	0.994
Mistral-Large-2512	Persian	Arabic	1.036	74.4	1.000
Hunyuan	Korean	CJK	1.042	74.3	0.512
Qwen-3	Nepali	Devanagari	1.042	74.2	0.962
Qwen-3	Hindi	Devanagari	1.075	73.4	0.992
Mistral-Large-2512	Arabic	Arabic	1.125	72.2	1.000
Hunyuan	Japanese	CJK	1.135	72.0	0.790
GPT-5	Japanese	CJK	1.294	68.0	0.790
GPT-5	Chinese	CJK	1.305	67.7	0.333
Hunyuan	Persian	Arabic	1.399	65.4	0.958
Hunyuan	Arabic	Arabic	1.407	65.2	0.995
Qwen-3	Korean	CJK	1.477	63.5	0.344
DeepSeek-V3	Korean	CJK	1.490	63.2	0.405
Qwen-3	Japanese	CJK	1.499	63.0	0.790
DeepSeek-V3	Japanese	CJK	1.503	62.9	0.790
Gemma-3	Chinese	CJK	1.503	62.8	1.000
Llama-4	Chinese	CJK	1.527	62.3	1.000
DeepSeek-V3	Nepali	Devanagari	1.574	61.1	0.947
Qwen-3	Chinese	CJK	1.591	60.7	0.188
Llama-4	Japanese	CJK	1.646	59.3	0.790
Hunyuan	Chinese	CJK	1.650	59.2	0.179
Hunyuan	Ukrainian	Cyrillic	1.662	58.9	0.877
GPT-5	Korean	CJK	1.687	58.3	0.364
DeepSeek-V3	Chinese	CJK	1.727	57.3	0.149
DeepSeek-V3	Hindi	Devanagari	1.732	57.2	0.858
Qwen-3	Persian	Arabic	1.753	56.7	0.763
Gemma-3	Japanese	CJK	1.774	56.2	0.790
Gemma-3	Korean	CJK	1.797	55.6	1.000
Llama-4	Korean	CJK	1.955	51.7	1.000
Qwen-3	Ukrainian	Cyrillic	1.959	51.6	0.842
Hunyuan	Russian	Cyrillic	2.132	47.3	0.846
DeepSeek-V3	Ukrainian	Cyrillic	2.306	43.0	0.822
Mistral-Large-2512	Ukrainian	Cyrillic	2.310	42.9	1.000
Mistral-Large-2512	Russian	Cyrillic	2.547	37.0	1.000
Llama-4	Nepali	Devanagari	2.587	36.1	1.000
DeepSeek-V3	Persian	Arabic	2.589	36.0	0.711
Qwen-3	Arabic	Arabic	2.605	35.6	0.762
DeepSeek-V3	Arabic	Arabic	2.663	34.2	0.844
Llama-4	Arabic	Arabic	2.705	33.1	1.000
GPT-5	Ukrainian	Cyrillic	2.881	28.8	0.782
Gemma-3	Arabic	Arabic	2.925	27.7	1.000
GPT-5	Nepali	Devanagari	2.955	27.0	0.897
Llama-4	Hindi	Devanagari	2.981	26.3	1.000
Qwen-3	Russian	Cyrillic	2.997	25.9	0.794
GPT-5	Persian	Arabic	3.039	24.9	0.673
Mistral-Large-2512	Portuguese	Latin	3.077	23.9	1.000
Gemma-3	Nepali	Devanagari	3.085	23.8	1.000
Gemma-3	Ukrainian	Cyrillic	3.101	23.3	1.000
Llama-4	Persian	Arabic	3.103	23.3	1.000
GPT-5	Hindi	Devanagari	3.140	22.4	0.671
GPT-5	Arabic	Arabic	3.153	22.1	0.777
Gemma-3	Persian	Arabic	3.162	21.9	1.000
Mistral-Large-2512	Italian	Latin	3.215	20.5	1.000
Llama-4	Ukrainian	Cyrillic	3.247	19.8	1.000
Mistral-Large-2512	German	Latin	3.248	19.7	1.000
Mistral-Large-2512	French	Latin	3.283	18.9	1.000
Mistral-Large-2512	Spanish	Latin	3.297	18.5	1.000
DeepSeek-V3	Russian	Cyrillic	3.377	16.5	0.770
Hunyuan	Italian	Latin	3.458	14.5	0.622
Qwen-3	Italian	Latin	3.470	14.2	0.622
Hunyuan	Portuguese	Latin	3.543	12.4	0.589
Qwen-3	Portuguese	Latin	3.589	11.3	0.588
Hunyuan	German	Latin	3.591	11.2	0.656
Hunyuan	French	Latin	3.621	10.5	0.532
Qwen-3	German	Latin	3.627	10.3	0.650
Qwen-3	French	Latin	3.662	9.5	0.524

Table 6: Gap-recovery decomposition under standard and reversed variable entry orders. The full model (M3) is identical in both orderings; only the attribution of the gap changes.

Metric	Standard order (Structural first)	Reversed order (Quality first)
R^2 M1 (Categorical ceiling)	0.854	0.854
R^2 M2	0.775	0.801
R^2 M3 (Full model)	0.827	0.827
M1–M2 gap	0.079	0.053
M2–M3 gain	0.052	0.026
Gap recovery (%)	66.6%	49.1%
Quality-first attribution	—	84.6%
Structural-first attribution (reversed)	—	22.4%

Table 7: Bootstrap coefficient stability for Model 3 (500 cluster-bootstrap replications). Relative bias = |OLS coef − bootstrap mean| / bootstrap SE. All relative bias values < 0.35 indicate stable estimates.

Predictor	OLS Coef.	Boot Mean	Boot SE	95% CI (Boot)
CJK vs Latin	−1.663	−1.652	0.307	[−2.202, −1.005]
Arabic vs Latin	−0.160	−0.126	0.375	[−0.819, 0.723]
Cyrillic vs Latin	0.105	0.123	0.344	[−0.503, 0.868]
Devanagari vs Latin	−0.033	0.053	0.476	[−0.773, 1.039]
Vocabulary Size	1.775	1.798	0.186	[1.470, 2.179]
Script-Specific Tokens	1.226	1.237	0.417	[0.486, 2.123]
Vocabulary Utilization	3.130	3.231	0.572	[2.322, 4.600]
Fragmentation Rate	−0.652	−0.665	0.230	[−1.181, −0.290]
Non-Text Density	−0.774	−0.898	0.390	[−1.724, −0.222]

Table 8: Shapley-value decomposition of Model 3 R^2. Values sum to the full model $R^2 = 0.827$.

Predictor Group	ΔR^2	% of R^2
Script Family (categorical)	0.414	50.1%
Structural Counts	0.329	39.7%
Quality Metrics	0.084	10.2%
Total	0.827	100.0%

Table 9: Model 3 coefficients at sentence level ($N = 104{,}685$) and cluster level ($N = 105$). Bootstrap 95% CIs in brackets.

Predictor	Sentence β	Cluster β
CJK	−1.663	−1.545
Vocabulary Size	1.775	1.777
Vocabulary Utilization	3.130	3.274
Fragmentation Rate	−0.652	−0.634
Non-Text Density	−0.774	0.888
R^2	0.827	0.908

Table 10: Model 3 R^2 and Vocabulary Size coefficient under leave-one-tokenizer-out. Full-model $R^2 = 0.827$; full-model Vocab Size $\beta = 1.775$.

Excluded Tokenizer	R^2	Vocab Size β
None (full model)	0.827	1.775
Gemma-3	0.838	1.912
Llama-4	0.841	1.683
Qwen-3	0.832	1.761
DeepSeek-V3	0.835	1.722
Hunyuan	0.821	1.671
Mistral-Large-2512	0.830	2.039
GPT-5	0.827	1.736

Table 11: Model 3 coefficient inference under clustered SEs (105 clusters) and HAC (Newey–West) standard errors.

Predictor	Coef.	Clustered 95% CI	HAC 95% CI
CJK	−1.663	[−2.17, −1.16]	[−1.71, −1.62]
Vocabulary Size	1.775	[1.44, 2.11]	[1.75, 1.80]
Vocabulary Utilization	3.130	[2.26, 4.00]	[3.05, 3.21]
Fragmentation Rate	−0.652	[−1.04, −0.27]	[−0.68, −0.62]
Non-Text Density	−0.774	[−1.48, −0.07]	[−0.84, −0.71]