MRT: Regularizing Tokenization with Morphology for Low-Resource Indic Languages

Anonymous authors
Paper under double-blind review

Abstract

Subword tokenizers such as BPE, Unigram and WordPiece select merges from corpus statistics alone, producing tokens that straddle morpheme boundaries. For morphologically rich, low-resource languages this inflates vocabulary sparsity and degrades downstream performance. Existing remedies either block cross-boundary merges outright, requiring supervised morphological annotation and offering no compression-fidelity control, or learn unsupervised segmentations that degrade with limited data. We introduce **Morphology-Regularized Tokenization (MRT)**, which adds morphological priors to BPE using only compact, curated suffix lists. *MRT-Base* pre-segments words at morpheme boundaries, imposing a hard constraint against cross-boundary merges. *MRT-Penalized* penalises each candidate merge by the fraction of its occurrences that cross a boundary, with a single strength parameter that recovers standard BPE at zero and morpheme-only merges in the limit. We evaluate on nine Indic languages from two families and two corpora, against our oracle and external gold segmentations, with language models trained from scratch at three scales and five extrinsic tasks. Under a tokenizer-invariant bits-per-character metric, an 8-layer GPT-2 trained on MRT-Penalized achieves the lowest held-out loss in 7 of 9 languages (5 by a margin of 1.6-2.8%, 2 by margins we treat as ties), is level with Unigram in the other two, and outperforms the byte encoding MYTE in all nine. The MRT variants also achieve the best score in 37 of 45 downstream comparisons, recover more externally annotated morpheme boundaries than every baseline in 42 of 48 settings, and render Bengali suffixes in about 4× fewer distinct subword forms, with the penalty strength confined to $\lambda \in [5.5, 7.0]$ across all languages and corpora.

1 Introduction

Subword tokenization is the dominant interface between text and modern neural language models. Byte-Pair Encoding (Sennrich et al., 2016), WordPiece (Schuster & Nakajima, 2012), and Unigram (Kudo, 2018), with the SentencePiece (Kudo & Richardson, 2018) backend used in models such as GPT-2 (Radford et al., 2019), learn merges from corpus statistics alone, optimising compression without any notion of linguistic structure. This suffices for typologically simple languages like English. However, it breaks down for morphologically rich, agglutinative languages in the Indic family, where a word typically consists of a root followed by suffixes encoding case, number, tense, and derivation. The Bengali word "বইগুলোতে" (*boigulote*, "in the books"), for example, decomposes into "বই" (root), "গুলো" (plural), and "তে" (locative), yet a frequency-driven tokenizer has no reason to respect these boundaries, and if the spanning trigram is frequent, BPE merges them into a token that is neither a morpheme nor semantically meaningful. We term this phenomenon *boundary violation*, with three downstream consequences.

First, it increases lexical sparsity, because inconsistent splits represent the same morpheme by different token sequences, so a model must tie several surface forms to one unit. Second, as a motivating hypothesis rather than a result, consistent segmentation should ease compositional generalisation. A model that always sees "গুলো" as one unit receives a single, repeated signal for plurality, whereas one that sometimes sees "গু" + "লো" must learn the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

same regularity from fewer, scattered examples. Third, it weakens interpretability, since a token that straddles a boundary corresponds to no linguistic unit, which obscures case and number marking from any analysis that operates on tokens. These costs matter most in the low-resource setting. Under a fixed token and data budget, segmentations that consolidate morphemes reduce the number of redundant surface forms a model must learn from limited text, and tokenizer choice has been shown to change downstream behaviour on low-resource Indic tasks such as zero-shot NER (Pattnayak et al., 2025). A morphologically coherent tokenization places linguistic priors such as plural, case, and tense markers directly in the vocabulary, which is a cheap scaffold when data are scarce.

Existing remedies trade off along several axes, which we review in Section 6. Hard-constraint methods (Asgari et al., 2026; García et al., 2025; Brahma et al., 2025) rely on supervised analysers or annotated segmentations and make a binary accept/reject decision on boundary-crossing merges, with no control over the compression-fidelity trade-off. Unsupervised segmenters (Creutz & Lagus, 2007) need no annotation, although they degrade with limited data. Byte-level re-encodings such as MYTE (Limisiewicz et al., 2024) equalise sequence length across languages. However, they do not change which units a subword learner merges. To overcome these drawbacks, we introduce **Morphology-Regularized Tokenization (MRT)** with two complementary variants, *MRT-Base* (**MRT-B**), which pre-segments words at morpheme boundaries to impose a hard constraint, and *MRT-Penalized* (**MRT-P**), which applies a soft, tunable penalty in the BPE merge score. As a drop-in SentencePiece replacement, MRT integrates into existing multilingual Indic systems without architectural change. The contributions are as follows.

1. **MRT-P**, a soft morphological penalty for BPE. A normalised penalty on each merge's boundary-crossing rate that interpolates between standard BPE (λ=0) and morpheme-only merges ($\lambda \to \infty$), exposing the compression-fidelity trade-off through a single parameter (Sections 2.4 and 2.5).
2. **MRT-B**, an annotation-free hard constraint. Pre-segmentation with an 11-44 entry suffix list per language, in place of the supervised analysers or gold segmentations required by prior hard-constraint methods (Asgari et al., 2026; García et al., 2025) (Sections 2.2 and 2.3).
3. A tokenizer-invariant multilingual evaluation. Nine Indic languages from two families, two corpora (Wikimedia Foundation, 2023; Khan et al., 2024), language models trained from scratch at three scales and scored in bits per character against four subword baselines and MYTE (Limisiewicz et al., 2024), external-gold morphological evaluation for seven languages, and five extrinsic tasks (Sections 3 and 4).
4. Robustness. The per-language optima of λ all fall in $[5.5, 7.0]$, and on Bengali and Hindi, perturbing up to half of the suffix list changes MRT-P's bits per character by at most 0.3% (Section 5, Appendix F.5).

2 Methodology

We present two morphology-regularised tokenisers that share a common rule-based suffix oracle μ. **MRT-B** enforces a hard morpheme-boundary constraint via pre-segmentation, while **MRT-P** applies a soft, tunable penalty to the BPE merge score. In the following page, Figure 1 summarises the complete MRT methodology.

2.1 Problem Formulation

Let $\mathcal{C}$ be a monolingual corpus with word vocabulary $\mathcal{W}$ and word frequencies $\text{freq}(w)$. A subword tokenizer $\tau : \mathcal{W} \to V^*$ with $|V| = K$ splits each word into tokens $\tau(w) = (t_1, \ldots, t_{m_w})$ with $t_1 \oplus \cdots \oplus t_{m_w} = w$ ($\oplus$ is concatenation), and its cut positions form the *token-boundary set* $\mathcal{P}_\tau(w) \subseteq \{1, \ldots, |w| - 1\}$. A morphological segmentation $\mu(w) = (\mu_1, \ldots, \mu_{r_w})$ of w induces the *morpheme-boundary set* $\mathcal{B}(w)$. A cut $p \in \mathcal{P}_\tau(w) \setminus \mathcal{B}(w)$ is a *boundary violation*, and a missing cut $p \in \mathcal{B}(w) \setminus \mathcal{P}_\tau(w)$ is a *morpheme fusion*, and τ is *morphologically aligned* at w when $\mathcal{P}_\tau(w) = \mathcal{B}(w)$. At corpus level we measure the two error types by the frequency-weighted *boundary-level violation rate* bBVR (the share of cuts that are wrong) and the *gold-boundary recall* RecB (the share of morpheme boundaries realised

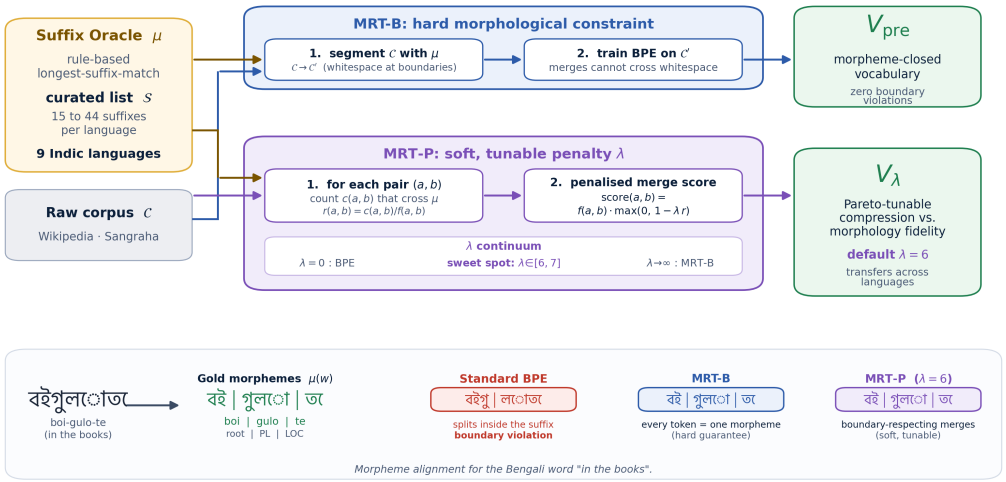


Figure 1: Overview of MRT. A rule-based suffix oracle μ, built from a curated suffix list $\mathcal{S}$ (11-44 entries per language), marks the morpheme boundaries $\mathcal{B}(w)$ of each word in the corpus $\mathcal{C}$. **MRT-B** (top) inserts whitespace at every boundary and runs standard BPE on the segmented corpus $\mathcal{C}'$, so no learned token crosses a boundary. **MRT-P** (bottom) keeps $\mathcal{C}$ intact and scores each merge by $f(a,b)\cdot\max(0, 1-\lambda\, r(a,b))$, where $r(a,b)$ is the fraction of occurrences of (a,b) that cross a boundary, and λ interpolates between BPE (λ=0) and MRT-B ($\lambda\to\infty$). The bottom strip is schematic, and actual segmentations are in Table 26 in Appendix G.1.

as cuts), both defined in Appendix B. Both are needed, since a tokenizer that never splits has bBVR = 0. Given $\mathcal{C}$, μ and K, **MRT** seeks a tokenizer with low bBVR and high RecB that remains competitive in held-out likelihood and downstream accuracy. MRT-B enforces bBVR = 0 on the training vocabulary, whereas MRT-P trades bBVR against compression through a single scalar λ. The token-level rate BVR reported in our tables counts tokens rather than cut positions (Appendix B).

2.2 Morphological Segmentation Oracle

Both MRT variants require a morphological segmentation function μ. Rather than relying on data-driven segmenters such as Morfessor (Creutz & Lagus, 2007) or supervised analysers (Bhat & Sharma, 2012; Ramasamy & Žabokrtský, 2012), we exploit the agglutinative structure of Indic languages with a lightweight *iterative longest-suffix-match* rule that needs only a curated suffix list and degrades gracefully on unseen words (Algorithm 1). Let $\mathcal{S} = \{s_1, \ldots, s_L\}$ be a curated list of morphological suffixes for the target language, sorted by decreasing length ($|s_1| \geq \cdots \geq |s_L|$). Given $\mathcal{S}$, Algorithm 1 returns $\mu(w)$, stripping suffixes only while a root of at least m=2 characters remains, and $\mathcal{B}(w)$ follows from Section 2.1 as the partial-sum positions of its segment lengths. For instance, "বইগুলোতে" segments as (বই, গুলো, তে), yielding $\mathcal{B} = \{2, 6\}$. Curated lists cover nine Indic languages with 11-44 entries each, authored by native speakers and cross-checked against standard grammars. Per-language sizes, suffix coverage, and average morphemes-per-word are reported in Table 6 in Appendix E.1.

Algorithm 1 Longest-Suffix-Match

```
function Segment(w, S, m=2)
    segs ← [ ]
    while some s ∈ S has w = u ⊕ s with |u| ≥ m do
        s ← the longest such suffix
        segs.Prepend(s)
        w ← u
    end while
    return segs.Prepend(w)
end function
```

Algorithm 1 is a greedy approximation to the true segmentation μ^* (exact with respect to $\mathcal{S}$ when $\mathcal{S}$ is suffix-free, by Proposition 2.1). Its accuracy depends on (i) *suffix completeness*, meaning every true suffix appears in $\mathcal{S}$, and (ii) *non-spuriousness*, meaning no $s \in \mathcal{S}$ matches a root-internal position. Writing $\mathcal{B}^*(w)$ for the gold boundary set, we report oracle quality through the standard precision and recall $\mathrm{Prec}(w) = |\mathcal{B} \cap \mathcal{B}^*|/|\mathcal{B}|$ and $\mathrm{Rec}(w) = |\mathcal{B} \cap \mathcal{B}^*|/|\mathcal{B}^*|$. Measured against external gold from Universal Dependencies and UniMorph (Appendix F.4), the curated lists are more precise than complete for Bengali, Hindi, Marathi and Tamil (boundary precision 0.30-0.90, above recall in every case, with recall 0.08-0.50), and weak

in both directions for Gujarati, Telugu and Assamese (precision 0.07-0.27). The oracle is therefore a deliberately cheap, incomplete prior, and our experiments measure what such a prior buys rather than assuming a near-perfect one.

Proposition 2.1 (Greedy optimality). *If $\mathcal{S}$ is* suffix-free*, no element is a proper suffix of another, Algorithm 1 returns the unique maximum-cardinality boundary set consistent with $\mathcal{S}$.*

The proof is one line. Suffix-freeness makes the longest match at each step the only choice that enables further stripping, so greedy is non-dominated and hence optimal. When suffix-freeness fails (e.g. both "এর" and "র" in $\mathcal{S}$) the algorithm still prefers the longer match. In practice we enforce suffix-freeness at curation time by keeping only the longer of any overlapping pair, so all nine lists used below are suffix-free by construction.

2.3 MRT-Base: a Hard Morphological Constraint

MRT-B applies μ to every word of $\mathcal{C}$, inserts a whitespace marker at each morpheme boundary to form $\mathcal{C}'$, and then trains standard BPE on $\mathcal{C}'$ under the usual rule that merges may not cross whitespace. Writing $\mathrm{off}(a) \in \{0, \ldots, |w|\}$ for the left offset of token a inside w under the current partial tokenisation $\tau_k(w)$, the admissible merge candidates at step k are

$$\mathrm{Cand}_k^{\mathrm{pre}} = \Big\{ (a,b) \;\Big|\; \forall w \in \mathcal{W}\colon\ \nexists\, p \in \mathcal{B}(w) \text{ with } \mathrm{off}(a) < p < \mathrm{off}(a) + |a| + |b| \Big\},$$

so cross-boundary merges are eliminated by construction. Because $\mathcal{C}'$ is a disjoint union of morpheme strings, MRT-B inherits BPE's $O(|\mathcal{C}| \log K)$ training cost, and the only overhead is a single $O(L)$ suffix match per unique word during preprocessing.

Proposition 2.2 (Morpheme closure). *Every $t \in V_{\mathrm{pre}} \setminus V_0$ is a contiguous substring of a single morpheme type. Equivalently, $\mathcal{P}_{\tau_{\mathrm{B}}}(w) \subseteq \mathcal{B}(w)$ for all $w \in \mathcal{W}$, so* $\mathrm{bBVR}(\tau_{\mathrm{B}}) = 0$ *on the training corpus.*

The claim follows by induction on merge steps, since V_0 contains only characters and every admissible merge extends two within-morpheme fragments (Appendix A.1).

Train-inference asymmetry. The oracle is invoked only at vocabulary-construction time, and at inference the learnt merge rules are applied to raw text. Proposition 2.2 is therefore *training-time exact* and only *inference-time approximate*, since a merge rule learnt in a non-boundary context may re-fire at a boundary in an unseen word, a failure mode MRT-P avoids by encoding boundary information directly into the merge score.

2.4 MRT-Penalized: a Soft Morphological Penalty

MRT-P keeps $\mathcal{C}$ intact and instead reweights BPE's merge-selection rule with a continuous boundary-crossing penalty, so morpheme alignment is encoded directly into the merge objective rather than into the corpus.

Definition 2.1 (Penalised merge score). At merge step k, let

$$f(a,b) = \sum_{w \in \mathcal{W}} \mathrm{freq}(w)\, \mathbf{1}\big[(a,b) \text{ adj. in } \tau_k(w)\big],$$

$$c(a,b) = \sum_{w \in \mathcal{W}} \mathrm{freq}(w)\, \mathbf{1}\big[(a,b) \text{ adj.}\big]\, \mathbf{1}\big[\mathrm{off}(a)+|a| \in \mathcal{B}(w)\big]$$

be the standard pair frequency and its boundary-crossing subset. Since $c(a,b) \le f(a,b)$ the *crossing rate* $r(a,b)$ is well defined, and the *penalised score* follows by multiplicative reweighting

$$r(a,b) = \frac{c(a,b)}{f(a,b)} \in [0,1], \qquad \mathrm{score}(a,b) = f(a,b) \cdot \max\big(0,\ 1 - \lambda\, r(a,b)\big).$$

The expected number of boundary violations introduced by committing (a, b) is exactly $c(a, b)$, so minimising $\lambda r(a, b)$ penalises the per-unit-frequency contribution of (a, b) to bBVR(τ).

Algorithm 2 MRT-P BPE

```
function MRT-P-BPE(C, S, K, λ)
    W ← WordFreqTable(C)
    B(w) ← Boundaries(w, S) for all w ∈ W
    tok(w) ← CharList(w) for all w ∈ W
    V ← specials ∪ {unique chars}
    for k = 1, ..., K − |V| do
        count f(a, b), c(a, b) over pairs in all tok(w)
        (a*, b*) ← arg max f · max(0, 1−λ c/f)
        if score(a*, b*) ≤ 0 then break        ▷ early stop
        V ← V ∪ {a* ⊕ b*}
        merge (a*, b*) in each tok(w)           ▷ inv. index
    end for
    return V
end function
```

Normalising by $f(a, b)$ in Definition 2.1 renders the penalty *scale-independent* (a pair that always crosses a boundary has $r = 1$ regardless of corpus size), and the $\max(0, \cdot)$ clamp zeros out any pair with $r > 1/\lambda$ rather than driving the score negative. The scalar $\lambda \in [0, \infty)$ thus acts as a single knob trading compression for alignment. Setting $\lambda = 0$ recovers vanilla BPE, $\lambda = 1$ eliminates pairs that always cross ($r = 1$), and $\lambda \to \infty$ admits only pairs with $r = 0$, coinciding with MRT-B (Proposition 2.4). The merge loop (Algorithm 2) stops early as soon as $\max_{(a,b)} \text{score}(a, b) \leq 0$, so at high λ this yields a vocabulary with fewer than $K - |V_0|$ merges, composed entirely of morphologically safe subwords. We maintain an inverted index from pairs to containing words so that each step re-scans only affected words, reducing per-step cost from $O(|\mathcal{W}|)$ to $O(|\text{affected}|)$.

2.5 Theoretical Properties

We establish three formal properties of MRT-P, namely monotone boundary suppression, its interpretation as a Lagrangian relaxation of a constrained optimisation problem, and the OOV-free coverage guarantee inherited from BPE. All proofs are deferred to Appendix A.1, and Appendices A.2-A.4 further show that the saturation threshold is tight, bound BVR under a noisy oracle, and relate cross-boundary tokens to predictive entropy.

Proposition 2.3 (Monotone Boundary Suppression). *For any fixed corpus $\mathcal{C}$, suffix list $\mathcal{S}$, and vocabulary budget K, the total number of cross-boundary merges selected by MRT-P is non-increasing in λ.*

Proposition 2.4 (Interpolation). *MRT-P with parameter λ realises a continuous homotopy between standard BPE ($\lambda = 0$) and MRT-B ($\lambda \to \infty$)*

$$\tau_0 = \tau_{\text{BPE}}, \qquad \lim_{\lambda \to \infty} \tau_\lambda = \tau_{\text{MRT-B}}$$

The penalised criterion is the Lagrangian relaxation of constrained BPE, namely $\max_M \sum_{(a,b)\in M} f(a, b)$ subject to a crossing-rate budget $\sum_{(a,b)\in M} r(a, b) \leq B$ and $|M| \leq K - |V_0|$, where $\lambda \geq 0$ is the multiplier for the budget. Tuning λ therefore traces the Pareto frontier between compression and morphological fidelity, with the $\max(0, \cdot)$ clamp enforcing complementary slackness.

Proposition 2.5 (OOV-Free Coverage). *For any word $w \in \Sigma^*$, both MRT-B and MRT-P admit a valid tokenization $\tau(w) \in V^*$.*

2.6 Complexity Analysis

Let $|\mathcal{W}|$ be the number of word types, $\ell_{\max}$ the maximum word length, $L = |\mathcal{S}|$ the suffix list size, and $|\text{aff}|_k$ the word types affected at merge step k. The oracle (Algorithm 1) costs $O(|\mathcal{W}| \cdot L \cdot \ell_{\max})$, with boundary sets stored as $O(1)$-lookup hash sets.

With an inverted index from pairs to containing word types, MRT-P (Algorithm 2) updates the counts $f(a, b), c(a, b)$ in $O(|\text{aff}|_k \cdot \ell_{\max})$ per step, giving total merge cost $T_{\text{MRT-P}} = O\big(|\mathcal{W}| \cdot L \cdot \ell_{\max} + (K - |V_0|) \cdot \overline{|\text{aff}|} \cdot \ell_{\max}\big)$. Since $\overline{|\text{aff}|} \ll |\mathcal{W}|$ in practice, the oracle preprocessing dominates and MRT-P matches standard BPE's asymptotic complexity (measured wall-clock times are in Appendix E.2). MRT-B pays the same $O(|\mathcal{W}| \cdot L \cdot \ell_{\max})$ preprocessing and then trains BPE on the segmented corpus $\mathcal{C}'$ with $|\mathcal{W}'| \leq \sum_w |\mu(w)|$ and a smaller effective

$\ell'_{\max}$, so its merge loop is typically faster than MRT-P's. Space is $O(|\mathcal{W}| \cdot \ell_{\max})$ across token sequences and the inverted index.

3 Experimental Setup

We use nine Indic languages, seven Indo-Aryan (Assamese, Bengali, Gujarati, Hindi, Marathi, Nepali, Punjabi) and two Dravidian (Tamil, Telugu), from Wikipedia and Sangraha-Verified (Khan et al., 2024), training on nested 100–100K-line splits and scoring on a deduplicated held-out set (Appendices D.1, D.2). Baselines are BPE, Unigram (Uni), WordPiece (WP), Morfessor (Morf) and, for LM, byte-level MYTE (Limisiewicz et al., 2024). K is set by VOLT (Xu et al., 2021) and MRT-P's $\lambda \in \{5.0, \ldots, 7.0\}$ by held-out LM loss (Section 5). Intrinsic metrics are MAS (morpheme-matching tokens), BVR, RecB and SCon (suffix consistency), against the oracle and external gold (Appendix B). LMs are trained from scratch per tokenizer (2- and 8-layer GPT-2 (Radford et al., 2019), BERT-base (Devlin et al., 2019)) and compared by tokenizer-independent bits per character (BPC). Downstream probes are POS (UD (Nivre et al., 2020)), NER (WikiANN (Pan et al., 2017)), and topic classification and similarity (XL-Sum (Hasan et al., 2021)) with BiLSTM/CNN models (Appendix D.4). All runs use seed 42; **bold**/underline mark best/second best.

4 Results

We address three research questions as below, and extended results are in Appendix F.

> **RQ1.** Does MRT produce higher-quality tokenizations? (Section 4.1)
> **RQ2.** Does morphological fidelity translate to better language modelling? (Section 4.2)
> **RQ3.** Do MRT tokenizers improve performance on downstream NLP tasks? (Section 4.3)

4.1 Morphological Fidelity (RQ1)

We measure token-morpheme *alignment* (MAS, BVR, RecB) and *suffix consistency* (SCon).

Against the oracle: On Bengali Wikipedia (10K), MRT-B raises MAS from 0.559 (BPE) to 0.657 and lowers BVR from 0.441 to 0.343, as Proposition 2.2 predicts. Across 15 frequent Bengali suffixes (Figure 2), BPE produces 7,565 distinct renderings against 1,877 for MRT-B and 1,836 for MRT-P (4.1× fewer), and the genitive র alone drops from 2,824 to 138 (the fertility cost of these extra splits is analysed in Appendix F.11, and example segmentations are in Appendix G.1). Because these metrics use the oracle that MRT optimises, we also evaluate against external gold.

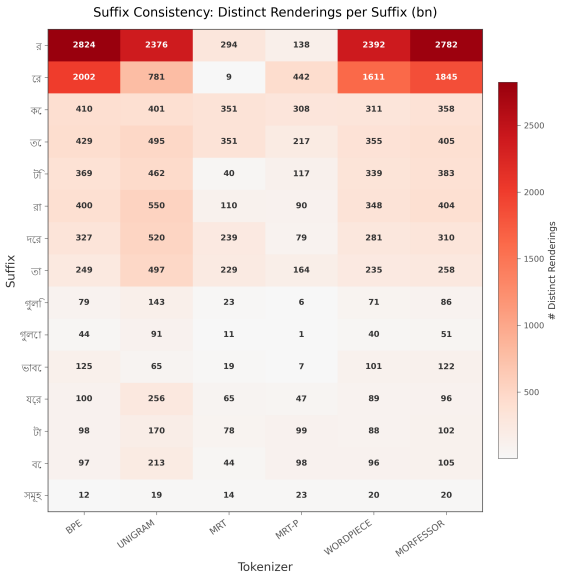


Figure 2: Suffix consistency (Bengali), measured as distinct renderings per suffix.

Against external gold: Using gold segmentations derived from Universal Dependencies and UniMorph for seven languages (48 settings, Appendix F.4), an MRT variant has the highest gold-boundary recall in 42 of 48 settings (MRT-P alone in 35). On held-out corpus text, BPE recovers at most 0.044 of the gold boundaries, whereas MRT-P recovers 0.09-0.62 in every setting except Telugu Sangraha. Token-level MAS against gold is mixed (an MRT variant is best in 17 of 48), since MRT-P fragments most and MAS divides by the token count. Suffix

consistency also holds under a single shared 9-language vocabulary, with 10-14 renderings per suffix for the MRT variants against 32-37 for the baselines (Appendix F.6).

4.2 Language Modelling (RQ2)

Table 1 is our primary LM result. Under an 8-layer GPT-2, MRT-P achieves the lowest bits per character in 7 of 9 languages, 1.2% below the strongest baseline on average, with clear margins of 1.6-2.8% in five, while Unigram is level in Bengali and Hindi. Bits per byte and per-word perplexity give the same ordering, and on the quality-filtered Telugu Sangraha split the margin is 2.6% (Appendices F.1 and D.2).

MYTE trails MRT-P in all nine languages and the strongest subword tokenizer by 6.4% on average, although its 56% shorter encoding reproduces on our data (Appendix F.2). Under BERT-base, MRT-P has the lowest per-token pseudo-perplexity in all nine languages (Table 14(a) in Appendix F.3). Since the fertility-scaled pseudo-BPC favours the baselines (Appendix F.3), we rest the tokenizer-invariant claim on the causal LM. Vocabulary sizes are not matched (MRT-P exports about half the pieces, see Table 11 and Appendix G.2), and under a vocabulary shared by all nine languages MRT-P is 0.5% behind Unigram in joint BPC (Appendix F.6).

4.3 Downstream Task Performance (RQ3)

On the Wikipedia 100K split across all 9 languages (full results in Table 21(a) in Appendix F.7, with per-tokenizer Sangraha results in Appendix F.10), MRT-P or MRT-B achieves the best score in 37 of 45 comparisons (82.2%), including all five metrics on Telugu and Gujarati. Table 22 in Appendix F.7 details Telugu, a Dravidian language with productive case stacking. These probes use small BiLSTM and CNN models so that differences are attributable to the tokenizer, and they complement, rather than replace, the language-model evaluation above. On Telugu, MRT-P gains +3.3 points absolute on classification macro-F1 and +18% relative on sentence similarity R@10 over the strongest baseline, and Gujarati, where an MRT variant also achieves the best score on all five metrics, shows a similar pattern. We relate these gains to the degree of suffixation of each language, measured by suffix coverage and morphemes per word, in Appendix E.1. Across the 9 languages, MRT-P has the best classification accuracy in 7 (BPE leads on Assamese, Morfessor on Punjabi) and the best sentence-similarity MRR in 8 (Unigram leads on Hindi), while NER gains are modest where entity spans are morphologically simple proper nouns.

4.4 Selection Recipe for MRT-B vs. MRT-P

MRT-B forbids cross-boundary merges outright and needs no hyperparameter beyond K, which guarantees morpheme-internal tokens on the training corpus (Proposition 2.2). MRT-P instead penalises crossing merges through λ, so the boundary prior is built into the learned merge order and no oracle is needed at inference. MRT-P gives the lowest bits per character in 7 of 9 languages, the MRT variants together lead in 37 of 45 downstream comparisons, and MRT-B gives the fewest suffix renderings and the highest MAS under a shared vocabulary. The optimum of λ is broad (Section 5), and perturbing up to half of the suffix list changes MRT-P's BPC by at most 0.3% on Bengali and Hindi (Appendix F.5). We therefore recommend MRT-P with λ=6.5 as the default for a new language, and MRT-B when a hard no-crossing guarantee is required or no λ search is possible.

Table 1: Held-out bits per character (↓) of an 8-layer GPT-2 per tokenizer (Wikipedia 10K). C is the number of held-out characters, and Δ compares MRT-P with the strongest of BPE, Uni, WP and Morf.

Lang.	C	BPE	Uni	WP	Morf	MRT-B	MRT-P	MYTE	Δ
Assamese	326,634	2.0215	1.9752	2.0115	2.0123	1.9691	**1.9192**	2.0639	−2.8%
Bengali	292,402	1.9371	**1.8639**	1.9324	1.9259	1.9145	1.8648	1.9968	+0.0%
Gujarati	266,315	2.1713	2.1178	2.1480	2.1141	2.1372	**2.0798**	–	−1.6%
Hindi	323,245	2.0476	**1.9916**	2.0334	2.0296	2.0331	1.9980	2.1354	+0.3%
Marathi	264,202	2.1716	2.1439	2.1662	2.1647	2.1589	**2.1052**	2.2233	−1.8%
Nepali	327,160	2.0101	1.9624	1.9982	1.9605	1.9522	**1.9110**	2.0615	−2.5%
Punjabi	279,918	2.0508	2.0133	2.0435	2.0422	2.0436	**2.0114**	2.1383	−0.1%
Tamil	295,067	1.9099	1.8711	1.8897	1.8644	1.8634	**1.8194**	1.8601	−2.4%
Telugu	98,917	1.9041	1.8572	1.8799	1.8702	1.8722	**1.8532**	–	−0.2%



Figure 3: Effect of the morphological penalty λ on various metrics for Bengali Wikipedia, 10K split, vocabulary 32K. Each subplot demonstrates the metric trajectory across $\lambda \in [0.5, 9.0]$, and the red ring marks the value of λ that achieves the best score for that metric.

5 Analysis

Performance Measurements: We sweep $\lambda \in \{0.5, 1.0, \ldots, 9.0\}$ (step 0.5) on Bengali Wikipedia (10K, 32K vocab) to validate the compression-fidelity trade-off (Proposition 2.4). Figure 3 annotates the best λ per metric. Held-out per-token perplexity of the 2-layer GPT-2, which compares λ values within one tokenizer family, drops steeply from 335.3 at λ=0.5 to 205.4 at λ=2.5 and bottoms out at 194.7 (λ=7.0, Table 23 in Appendix F.8), so even moderate regularisation has most of the effect. The optimum shifts with the task. POS accuracy peaks early (λ=3.5, 0.616), while semantically richer tasks prefer stronger penalty, CLF macro-F1 (λ=6.5), NER F1 (λ=8.5), and SIM Recall@5 (λ=6.5). All optima fall in $\lambda \in [3.5, 8.5]$, so MRT-P requires no fragile tuning. λ=6.5 balances the task optima and is our recommended default for a new language. For the reported results, λ is instead selected per language and corpus by held-out LM loss from $\{5.0, \ldots, 7.0\}$ (Section 3, Appendix D.3), which gives λ=7.0 for Bengali Wikipedia.

Error analysis: This analysis is defined against the same oracle boundaries $\mathcal{B}(w)$ that MRT optimises, so it cannot on its own establish utility, so we use it as a diagnostic that explains the mechanism behind the extrinsic results, whose independent support comes from the external-gold recall of Section 4.1. We classify tokenized words into five categories by comparing $\tau(w)$ against $\mathcal{B}(w)$, namely **Correct** (exact), **Over Segmentation** ($\tau \supset \mathcal{B}$), **Under Segmentation** ($\tau \subset \mathcal{B}$), **Boundary Shift** (± 1 char offset), and **Suffix Mangling** (a token straddles a boundary, fusing root and suffix). Suffix Mangling produces chimeric tokens that conflate two morphosyntactic variables (Proposition S.3 in Appendix A.4). Figure 4 shows the distribution for Telugu (10K split, 82K words analysed). Suffix Mangling dominates BPE errors at 73.4%, with WordPiece similarly high at 70.6%. MRT-P (λ=6.0) reduces Suffix Mangling to 19.9%, a 3.7× reduction, replacing it primarily with Over Segmentation (69.5%), a category in which each resulting token still lies within a single oracle morpheme. MRT-B achieves an intermediate profile, with 24.6% Suffix Mangling with the highest Correct rate (16.4%) among all methods. This error-type shift is consistent with, though not by itself evidence for, the downstream gains, the lower held-out loss of the language model, and the more consistent suffix tokenization reported above.

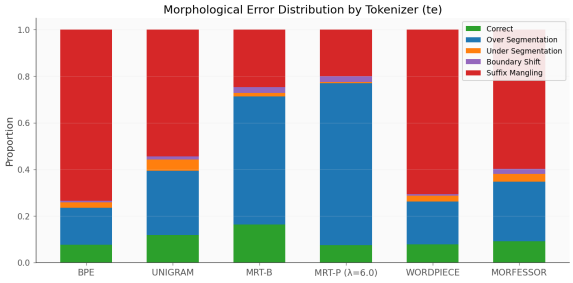


Figure 4: Error distribution (Telegu, Wiki 10k).

Segmentation case analysis. Table 2 depicts representative Gujarati words. We notice three patterns. First, BPE fails to segment common case markers. For example, જેલમાંથી ("from jail", gold જેલ|માં|થી) is kept whole by BPE, obscuring the locative માં and ablative

Table 2: Segmentation case study (Wiki 10K). Token boundaries are denoted by "|".

Word	Gold	BPE	Uni	MRT-B	MRT-P	WP	Morf
૧૯૪૯ના	૧૯૪૯|ના	૧૯૪૯ના	૧૯૪૯|ના	૧૯૪૯|ના	૧૯૪૯|ના	૧૯૪૯ના	૧૯૪૯|ના
કરવામાં	કરવા|માં	કરવામાં	કરવા|માં	કરવા|માં	કરવા|માં	કરવામાં	કરવામાં
જેલમાંથી	જેલ|માં|થી	જેલમાંથી	જેલમાં|થી	જેલ|માં|થી	જેલ|માં|થી	જેલમાંથી	જેલમાં|થી
કલકત્તાના	કલકત્|તા|ના	કલકત્તાના	કલકત્તા|ના	કલકત્|તાન|ા	કલક|ત|ૃતાન|ા	કલકત્તાના	કલકત્તા|ના
રોજ	રોજ	રોજ	રોજ	રોજ	રોજ	રોજ	રોજ

થી, whereas MRT-B recovers the gold segmentation exactly. Second, proper nouns expose a limitation. The word કલકત્તાના ("of Kolkata") is over-segmented by both MRT variants because the oracle treats તા as a productive suffix, though here it is part of the stem, a known edge case for rule-based oracles. Third, the mono-morphemic word રોજ remains intact across all tokenizers, so no spurious splits are introduced when no boundary is identified.

6 Related Work

MorphBPE (Asgari et al., 2026) blocks BPE merges that cross supervised morpheme boundaries, García et al. (2025) train BPE on semi-supervised pre-segmentations of Spanish, and MorphTok (Brahma et al., 2025) applies morphology-aware pre-tokenization to Indian languages. All three make a binary accept/reject decision and depend on annotation or a trained segmenter. MRT-B is the closest analogue to this family and appears in every table, whereas MRT-P uses a continuous penalty that recovers the hard constraint only as $\lambda \to \infty$ (Proposition 2.4) and needs only an 11-44 entry suffix list. A direct reproduction was not possible (no released implementation for MorphBPE or MorphTok, and García et al. (2025) is Spanish-specific), so we compare against MRT-B throughout. MYTE (Limisiewicz et al., 2024) re-encodes UTF-8 with morpheme-informed byte codes to equalise sequence length across languages. It changes the encoding rather than the merge objective, and we run it as a baseline. Morfessor (Creutz & Lagus, 2007) induces morphs without supervision and degrades on small corpora, supervised analysers and treebanks (Bhat & Sharma, 2012; Ramasamy & Žabokrtský, 2012) need annotation that most Indic languages lack, and morphology-aware segmentation for machine translation (Ataman et al., 2017; Saleva & Lignos, 2021) targets individual language pairs. VOLT (Xu et al., 2021) selects the vocabulary size and BPE-dropout (Provilkov et al., 2020) regularises segmentation, and both leave the merge criterion purely statistical, the component MRT modifies. Multilingual Indic models such as IndicBERT (Kakwani et al., 2020) and MuRIL (Khanuja et al., 2021) share subword vocabularies that are not tuned to any single language's morphology, and Pattnayak et al. (2025) show that tokenizer choice affects low-resource Indic NER.

7 Conclusion

We introduced **Morphology-Regularized Tokenization (MRT)**, with two algorithms, MRT-B (hard constraint) and MRT-P (soft penalty), that inject morphological priors into BPE merge scoring from a compact suffix list. Across 9 Indic languages and two corpora, MRT-P achieves the lowest tokenizer-invariant held-out loss in 7 of 9 languages under an 8-layer GPT-2 (5 by clear margins, 2 by tie-level margins), is level with Unigram in the other two, and outperforms MYTE in all nine. Under a BERT-base masked LM it gives the lowest per-token pseudo-perplexity in all nine languages, whereas it trails the strongest baseline in fertility-scaled pseudo-BPC, two estimators biased in opposite directions. The MRT variants achieve the best score in 37 of 45 downstream comparisons, recover more externally annotated morpheme boundaries than every baseline in 42 of 48 settings, and render Bengali suffixes in about 4× fewer distinct forms. The per-language optima of λ all fall in the narrow band $[5.5, 7.0]$, so only a coarse search is needed for a new language, and training adds negligible overhead over BPE (Section 2.6).

Limitations. MRT relies on a short curated suffix list and a suffix-stripping oracle, so it currently targets suffixing languages, and our language models are trained from scratch at up to 110M parameters so that the tokenizer is the only variable. Extending the framework to other morphological types and to large pretrained multilingual models is the natural next step, and Appendix H discusses further limitations.

References

Ehsaneddin Asgari, Yassine El Kheir, MohammadAli SadraeiJavaheri, and Ali Nazari. MorphBPE: Morphology-aware tokenization for efficient LLM training. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 41610–41621, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.2068. URL https://aclanthology.org/2026.findings-acl.2068/.

Duygu Ataman, Matteo Negri, Marco Turchi, and Marcello Federico. Linguistically motivated vocabulary reduction for neural machine translation from turkish to english. *The Prague Bulletin of Mathematical Linguistics*, 108(1):331–342, 2017. doi: 10.1515/pralin-2017-0031. URL https://ufal.mff.cuni.cz/pbml/108/art-ataman-negri-turchi-federico.pdf.

Khuyagbaatar Batsuren, Omer Goldman, Salam Khalifa, Nizar Habash, Witold Kieraś, Gábor Bella, Brian Leonard, Garrett Nicolai, Kyle Gorman, Yustinus Ghanggo Ate, Maria Ryskina, Sabrina Mielke, Elena Budianskaya, Charbel El-Khaissi, Tiago Pimentel, Michael Gasser, William Abbott Lane, Mohit Raj, Matt Coler, Jaime Rafael Montoya Samame, Delio Siticonatzi Camaiteri, Esaú Zumaeta Rojas, Didier López Francis, Arturo Oncevay, Juan López Bautista, Gema Celeste Silva Villegas, Lucas Torroba Hennigen, Adam Ek, David Guriel, Peter Dirix, Jean-Philippe Bernardy, Andrey Scherbakov, Aziyana Bayyr-ool, Antonios Anastasopoulos, Roberto Zariquiey, Karina Sheifer, Sofya Ganieva, Hilaria Cruz, Ritván Karahóǧa, Stella Markantonatou, George Pavlidis, Matvey Plugaryov, Elena Klyachko, Ali Salehi, Candy Angulo, Jatayu Baxi, Andrew Krizhanovsky, Natalia Krizhanovskaya, Elizabeth Salesky, Clara Vania, Sardana Ivanova, Jennifer White, Rowan Hall Maudslay, Josef Valvoda, Ran Zmigrod, Paula Czarnowska, Irene Nikkarinen, Aelita Salchak, Brijesh Bhatt, Christopher Straughn, Zoey Liu, Jonathan North Washington, Yuval Pinter, Duygu Ataman, Marcin Wolinski, Totok Suhardijanto, Anna Yablonskaya, Niklas Stoehr, Hossep Dolatian, Zahroh Nuriah, Shyam Ratan, Francis M. Tyers, Edoardo M. Ponti, Grant Aiton, Aryaman Arora, Richard J. Hatcher, Ritesh Kumar, Jeremiah Young, Daria Rodionova, Anastasia Yemelina, Taras Andrushko, Igor Marchenko, Polina Mashkovtseva, Alexandra Serova, Emily Prud'hommeaux, Maria Nepomniashchaya, Fausto Giunchiglia, Eleanor Chodroff, Mans Hulden, Miikka Silfverberg, Arya D. McCarthy, David Yarowsky, Ryan Cotterell, Reut Tsarfaty, and Ekaterina Vylomova. UniMorph 4.0: Universal Morphology. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pp. 840–855, Marseille, France, June 2022. European Language Resources Association. URL https://aclanthology.org/2022.lrec-1.89/.

Riyaz Ahmad Bhat and Dipti Misra Sharma. Dependency treebank of Urdu and its evaluation. In Nancy Ide and Fei Xia (eds.), *Proceedings of the Sixth Linguistic Annotation Workshop*, pp. 157–165, Jeju, Republic of Korea, July 2012. Association for Computational Linguistics. URL https://aclanthology.org/W12-3623/.

Riyaz Ahmad Bhat, Rajesh Bhatt, Annahita Farudi, Prescott Klassen, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, Ashwini Vaidya, Sri Ramagurumurthy Vishnu, and Fei Xia. The Hindi/Urdu treebank project. In Nancy Ide and James Pustejovsky (eds.), *Handbook of Linguistic Annotation*, pp. 659–697. Springer, 2017. doi: 10.1007/978-4-431-55858-3_25.

Mausoom Brahma, N. J. Karthika, Anmol Singh, Dheeraj Adiga, Shripad Bhate, Ganesh Ramakrishnan, et al. Morphtok: Morphologically grounded tokenization for indian languages. In *International Conference on Machine Learning (ICML)*, 2025. URL https://openreview.net/forum?id=32wHtvZOqH.

Mathias Creutz and Krista Lagus. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1), 2007. ISSN 1550-4875. doi: 10.1145/1187415.1187418.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Andrew Thomas Dyer, Riffat Sharmin, and Sadia Afrin. Universal dependencies treebank UD_ Bengali-Sabdakosh (release 2.15). http://handle.net, 2024. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Alba Táboas García, Piotr Przybyła, and Leo Wanner. Exploring morphology-aware tokenization: A case study on Spanish language modeling. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 30505–30518, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1552. URL https://aclanthology.org/2025.emnlp-main.1552/.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005. ISSN 0893-6080. doi: 10.1016/j.neunet.2005.06.042.

Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 4693–4703, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.413. URL https://aclanthology.org/2021.findings-acl.413/.

Mayank Jobanputra, Maitrey Mehta, and Çağrı Çöltekin. A Universal Dependencies treebank for Gujarati. In Archna Bhatia, Gosse Bouma, A. Seza Doğruöz, Kilian Evang, Marcos Garcia, Voula Giouli, Lifeng Han, Joakim Nivre, and Alexandre Rademaker (eds.), *Proceedings of the Joint Workshop on Multiword Expressions and Universal Dependencies (MWE-UD) @ LREC-COLING 2024*, pp. 56–62, Torino, Italia, May 2024. ELRA and ICCL. URL https://aclanthology.org/2024.mwe-1.9/.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4948–4961, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.445. URL https://aclanthology.org/2020.findings-emnlp.445/.

Mohammed Safi Ur Rahman Khan, Priyam Mehta, Ananth Sankar, Umashankar Kumaravelan, Sumanth Doddapaneni, Suriyaprasaad B, Varun G, Sparsh Jain, Anoop Kunchukuttan, Pratyush Kumar, Raj Dabre, and Mitesh M. Khapra. IndicLLMSuite: A blueprint for creating pre-training and fine-tuning datasets for Indian languages. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15831–15879, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.843. URL https://aclanthology.org/2024.acl-long.843/.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Singhal, Subhashini Venugopalan, Sunita Sarawagi, and Partha Talukdar. Muril: Multilingual representations for indian languages, 2021. URL https://arxiv.org/abs/2103.10730.

Yoon Kim. Convolutional neural networks for sentence classification. In Alessandro Moschitti, Bo Pang, and Walter Daelemans (eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. URL https://aclanthology.org/D14-1181/.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sabrina J. Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. UniMorph 2.0: Universal Morphology. In Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga (eds.), *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://aclanthology.org/L18-1293/.

Bhadriraju Krishnamurti. *The Dravidian Languages*. Cambridge Language Surveys. Cambridge University Press, Cambridge, 2003. ISBN 9780521771115.

Bhadriraju Krishnamurti and John Peter Lucius Gwynn. *A Grammar of Modern Telugu*. Oxford University Press, Delhi, 1985. ISBN 978-0195616644.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 66–75, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1007. URL https://aclanthology.org/P18-1007/.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Eduardo Blanco and Wei Lu (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012/.

Tomasz Limisiewicz, Terra Blevins, Hila Gonen, Orevaoghene Ahia, and Luke Zettlemoyer. MYTE: Morphology-driven byte encoding for better and fairer multilingual language modeling. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15059–15076, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.804. URL https://aclanthology.org/2024.acl-long.804/.

Colin P. Masica. *The Indo-Aryan Languages*. Cambridge Language Surveys. Cambridge University Press, Cambridge, 1991. ISBN 9780521234207.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. Universal Dependencies v2: An evergrowing multilingual treebank collection. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 4034–4043, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.497/.

Martha Palmer, Rajesh Bhatt, Bhuvana Narasimhan, Owen Rambow, Dipti Misra Sharma, and Fei Xia. Hindi syntax: Annotating dependency, lexical predicate-argument structure, and phrase structure. In *The 7th International Conference on Natural Language Processing (ICON-2009)*, pp. 14–17, Hyderabad, India, 2009.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. Cross-lingual name tagging and linking for 282 languages. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1946–1958, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1178. URL https://aclanthology.org/P17-1178/.

Priyaranjan Pattnayak, Hitesh Laxmichand Patel, and Amit Agarwal. Tokenization matters: Improving zero-shot ner for indic languages. In *2025 IEEE International Conference on Electro/Information Technology (eIT)*, 2025. doi: 10.1109/11103625. URL https://ieeexplore.ieee.org/document/11103625/.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. BPE-dropout: Simple and effective subword regularization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1882–1892, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.170. URL https://aclanthology.org/2020.acl-main.170/.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Taraka Rama and Sowmya Vajjala. A Telugu treebank based on a grammar book. In Jan Hajič (ed.), *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories*, pp. 119–128, Prague, Czech Republic, 2017. URL https://aclanthology.org/W17-7616/.

Loganathan Ramasamy and Zdeněk Žabokrtský. Prague dependency style treebank for Tamil. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pp. 1888–1894, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL https://aclanthology.org/L12-1242/.

Vinit Ravishankar. A Universal Dependencies treebank for Marathi. In Jan Hajič (ed.), *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories*, pp. 190–200, Prague, Czech Republic, 2017. URL https://aclanthology.org/W17-7623/.

Jonne Saleva and Constantine Lignos. The effectiveness of morphology-aware segmentation in low-resource neural machine translation. In Ionut-Teodor Sorodoc, Madhumita Sushil, Ece Takmaz, and Eneko Agirre (eds.), *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 164–174, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-srw.22. URL https://aclanthology.org/2021.eacl-srw.22/.

M. Schuster and K. Nakajima. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5149–5152, 2012. doi: 10.1109/icassp.2012.6289079.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162/.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Wikimedia Foundation. Wikimedia downloads: Database backup dumps. https://dumps.wikimedia.org/, November 2023. Snapshot date: 2023-11-01.

Jingjing Xu, Hao Zhou, Chun Gan, Zaixiang Zheng, and Lei Li. Vocabulary learning via optimal transport for neural machine translation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7361–7373, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.571. URL https://aclanthology.org/2021.acl-long.571/.

Contents of Supplementary Material

A Supplementary Proofs
- A.1 Proofs of Theoretical Properties (Section 2.5)
- A.2 Tightness of the Saturation Threshold
- A.3 Robustness to Oracle Errors
- A.4 Information-Theoretic Connection to Perplexity

B Intrinsic Metric Definitions

C Extrinsic Task and LM Metric Descriptions

D Detailed Experimental Setup
- D.1 Setup Details Omitted from the Main Text
- D.2 Data
- D.3 Tokenizers and λ Selection
- D.4 Model Training Details

E Implementation Details
- E.1 Suffix Statistics and Degree of Suffixation
- E.2 Architectures and Compute

F Extended Experimental Results
- F.1 All Language-Model Metrics for GPT-2 8L
- F.2 MYTE Baseline
- F.3 BERT-base Pseudo-Bits per Character
- F.4 Evaluation Against External Gold Segmentations
- F.5 Sensitivity to Noisy and Incomplete Suffix Lists
- F.6 Shared Multilingual Vocabulary
- F.7 Full Downstream Comparison
- F.8 Full Lambda Sweep
- F.9 Per-Token Perplexity of Small-Scale Language Models (All 9 Languages)
- F.10 Per-Task Detailed Results (Sangraha, 100K)
- F.11 Fidelity vs. Word Length

G Qualitative Analysis
- G.1 Tokenization Examples
- G.2 Vocabulary Overlap and Coverage Analysis
- G.3 Correlation between Intrinsic Metrics, Statistical Significance, and Downstream Performance

H Limitations

I Broader Impacts

J Statements

A Supplementary Proofs

A.1 Proofs of Theoretical Properties (Section 2.5)

This appendix provides the proofs of Propositions 2.3, 2.4, and 2.5 stated in Section 2.5. Appendix A.2 shows that the saturation threshold of Proposition 2.4 is tight, Appendix A.3 bounds BVR under a noisy oracle, and Appendix A.4 relates cross-boundary tokens to predictive entropy.

Proof of Proposition 2.3 (Monotone Boundary Suppression). Let $\mathcal{M}_\lambda$ denote the ordered sequence of merges selected under parameter λ (noting that early stopping may yield $|\mathcal{M}_\lambda| \leq K - |V_0|$), and let $\mathrm{CB}(\mathcal{M}_\lambda) = \sum_{(a,b)\in\mathcal{M}_\lambda} \mathbf{1}[r(a,b) > 0]$ count cross-boundary merges. Suppose, for contradiction, that $\lambda' > \lambda$ yet $\mathrm{CB}(\mathcal{M}_{\lambda'}) > \mathrm{CB}(\mathcal{M}_\lambda)$. Then there exists a step k at which (a^*, b^*) is selected under λ' with $r(a^*, b^*) > 0$, while some non-crossing alternative (a', b') with $r(a', b') = 0$ satisfies

$$f(a^*, b^*) \cdot \max\big(0,\, 1 - \lambda' \cdot r(a^*, b^*)\big) \geq f(a', b')$$

Since $r(a^*, b^*) > 0$ and $\lambda' > \lambda$, the factor $\max(0, 1 - \lambda' r)$ is at most $\max(0, 1 - \lambda r)$. Thus $f(a^*, b^*) \cdot \max(0, 1 - \lambda \cdot r(a^*, b^*)) \geq f(a^*, b^*) \cdot \max(0, 1 - \lambda' \cdot r(a^*, b^*)) \geq f(a', b')$, so (a^*, b^*) would also have been preferred under λ. Moreover, early stopping can only reduce the number of merges (and hence cross-boundary merges) at higher λ. By induction over merge steps, $\mathrm{CB}(\mathcal{M}_{\lambda'}) \leq \mathrm{CB}(\mathcal{M}_\lambda)$, a contradiction. □

Proposition 2.3 concerns the number of boundary-crossing merges in the vocabulary, the source of the wrong cuts counted by bBVR. It does not by itself make the token-level $\mathrm{BVR}(\tau_\lambda)$ of Appendix B monotone in λ, since BVR divides by the number of tokens and therefore also depends on fragmentation, which changes with λ.

Proof of Proposition 2.4 (Interpolation). At $\lambda = 0$, $\mathrm{score}(a, b) = f(a, b)$ exactly recovers the BPE criterion. For any cross-boundary pair (a, b) with $r(a, b) > 0$, define the saturation threshold $\Lambda(a, b) = 1/r(a, b)$. For all $\lambda \geq \Lambda(a, b)$, the clamped factor $\max(0, 1 - \lambda \cdot r(a, b)) = 0$, so the pair receives score zero and is never selected over any non-crossing alternative with positive frequency. Since $r(a, b) \in (0, 1]$ for all cross-boundary pairs, $\lambda^* = \max_{(a,b):r(a,b)>0} 1/r(a, b)$ is finite, and for all $\lambda > \lambda^*$ no cross-boundary merge is ever selected. The resulting vocabulary is thus identical to that of MRT-B, which prohibits cross-boundary merges by construction. □

The Lagrangian interpretation given in Section 2.5 follows by treating $\lambda \geq 0$ as the multiplier for the crossing-rate budget constraint $\sum_{(a,b)\in M} r(a, b) \leq B$ in the constrained BPE problem. The inequality $|M| \leq K - |V_0|$ (rather than equality) accounts for the early-stopping behaviour at high λ, and the $\max(0, \cdot)$ clamp enforces complementary slackness.

Proof of Proposition 2.5 (OOV-Free Coverage). The initial vocabulary V_0 contains every Unicode character present in the training corpus. The BPE merge procedure only extends V and never removes members of V_0. Therefore $V \supseteq V_0$, and any word w can be decomposed character-by-character. MRT-B trains on a segmented corpus and applies the resulting vocabulary to raw text, and the character fallback remains valid since no characters are withheld. MRT-P modifies only the pair-selection criterion, so V_0 is unchanged. □

A.2 Tightness of the Saturation Threshold

Proposition 2.4 establishes that MRT-P converges to MRT-B as $\lambda \to \infty$, with the saturation threshold $\lambda^* = \max_{(a,b):r(a,b)>0} 1/r(a, b)$ where $r(a, b) = c(a, b)/f(a, b)$ is the crossing rate. We now show this bound is tight.

> **Proposition S.1 (Tightness of λ^*).** *For any $\epsilon > 0$, there exists a corpus $\mathcal{C}$ and suffix list $\mathcal{S}$ such that MRT-P with $\lambda = \lambda^* - \epsilon$ performs at least one cross-boundary merge.*

Proof. Let (a^*, b^*) be the pair achieving $\lambda^* = 1/r(a^*, b^*)$, i.e., the pair with the lowest crossing rate among all cross-boundary pairs. At parameter $\lambda^* - \epsilon$ we have

$$\text{score}(a^*, b^*) = f(a^*, b^*) \cdot \max\big(0,\ 1 - (\lambda^* - \epsilon) \cdot r(a^*, b^*)\big) = f(a^*, b^*) \cdot \epsilon \cdot r(a^*, b^*) > 0$$

Construct $\mathcal{C}$ such that (a^*, b^*) is the unique pair with the highest frequency and it always crosses a morpheme boundary ($r(a^*, b^*) = 1$, i.e., $\lambda^* = 1$), and all other pairs have frequency strictly less than $f(a^*, b^*) \cdot \epsilon$. Then (a^*, b^*) has score $f \cdot \epsilon > 0$, strictly exceeding all alternatives, and is selected. □

Corollary S.1. The interval $[0, \lambda^*)$ is the maximal open region in which MRT-P strictly differs from MRT-B with respect to the set of cross-boundary merges performed.

A.3 Robustness to Oracle Errors

In practice the oracle μ is imperfect, since the suffix list may be incomplete (false negatives-real boundaries not in $\hat{\mathcal{B}}$) or spurious (false positives-predicted boundaries that are not real). We characterise the degradation in BVR under noisy oracles.

Setting. Let $\hat{\mathcal{B}}(w)$ be the noisy oracle boundary set. Define the per-word false negative rate $\beta(w) = |\mathcal{B}^*(w) \setminus \hat{\mathcal{B}}(w)|/|\mathcal{B}^*(w)|$ and false positive rate $\alpha(w) = |\hat{\mathcal{B}}(w) \setminus \mathcal{B}^*(w)|/|\hat{\mathcal{B}}(w)|$, with corpus-level averages $\bar{\alpha}, \bar{\beta}$.

Proposition S.2 (BVR under Noisy Oracle). *Let $BVR^*(\lambda)$ denote the BVR under the perfect oracle. Under independently distributed oracle errors, the expected BVR satisfies*

$$\mathbb{E}[\text{BVR}(\hat{\tau}_\lambda)] \leq \text{BVR}^*(\lambda) + \bar{\beta}(1 - \text{BVR}^*(\lambda))$$

Proof sketch. False negatives (rate $\bar{\beta}$) cause real cross-boundary positions to go unpenalised, as a fraction $\bar{\beta}$ of actual boundary crossings receive penalty zero instead of λ, making them effectively equivalent to non-crossing pairs in the scoring function. In the worst case, when these unpenalised pairs are selected, BVR increases by exactly $\bar{\beta}(1 - \text{BVR}^*(\lambda))$, since only non-violating merges (fraction $1 - \text{BVR}^*$) can be displaced. False positives penalise non-boundary positions, making some valid merges less likely, which cannot increase BVR. □

Implication. BVR degrades linearly in $\bar{\beta}$ and is insensitive to $\bar{\alpha}$. Measured against external gold (Appendix F.4), the implied false-negative rate of our curated lists, $\bar{\beta} = 1 - \text{Rec}$, is 0.50-0.97, so for these lists the bound is loose and we do not rely on it. Instead, Appendix F.5 measures the effect of suffix-list noise directly, by perturbing the lists and retraining the tokenizer and the language model from scratch.

A.4 Information-Theoretic Connection to Perplexity

We provide a formal motivation for why cross-boundary tokens should be harder for a language model to predict. **Setup.** Consider a word $w = \mu_1 \oplus \mu_2$ composed of two morphemes. Under BPE, suppose w is tokenized as a single token $t = w$ (full fusion) or split into $t_1 = \mu_1[:\ell] \oplus \mu_2[:\ell']$ and $t_2 = \mu_2[\ell':]$ (cross-boundary split). Under MRT, the split is $t_1^* = \mu_1$, $t_2^* = \mu_2$ (aligned). Let R denote the root class of μ_1 and S the suffix class of μ_2.

Proposition S.3 (Cross-Boundary Token Entropy). *Let $t_\times = \mu_1[tail] \oplus \mu_2[head]$ be a cross-boundary token and $t_{align} \in \{\mu_1, \mu_2\}$ be an aligned token. If R and S are independent conditioned on context, then*

$$H(t_\times \mid \text{ctx}) \geq H(t_{\text{align}} \mid \text{ctx})$$

Proof. By the independence assumption, $t_\times$ is a function of two independent random variables, $X = \mu_1[\text{tail}]$ (determined by R) and $Y = \mu_2[\text{head}]$ (determined by S). The joint entropy satisfies $H(X, Y \mid \text{ctx}) = H(X \mid \text{ctx}) + H(Y \mid \text{ctx})$ by independence. Since $t_\times = f(X, Y)$ for a deterministic concatenation f, $H(t_\times \mid \text{ctx}) = H(X, Y \mid \text{ctx}) \geq H(X \mid \text{ctx}) = H(t_{\text{align}} \mid \text{ctx})$. □

Implication. Under the stated independence assumption, cross-boundary tokens (the Suffix Mangling category of Section 5) are harder to predict because their surface form conflates

two morphosyntactic variables, whereas morphologically aligned tokens each carry a single source of variance. The proposition concerns entropy *per token*. Because tokenizers emit different numbers of tokens for the same text, we test its consequence empirically with a per-character measure (bits per character, Section 4.2) rather than with per-token perplexity.

B Intrinsic Metric Definitions

We define the intrinsic metrics that characterise tokenization quality without requiring downstream model training. All metrics are computed over the full word set $\mathcal{W}$ of a corpus split, using the reference segmentation $\mu(w)$ produced by the longest-suffix-match oracle (Section 2.2), and in Appendix F.4 the same formulas are evaluated with $\mu(w)$ replaced by external gold segmentations. Together, they capture complementary aspects, namely token-level alignment (MAS, BVR), boundary-level alignment in both directions (bBVR, RecB), oracle coverage (SC), and uniformity (SCon).

Morphological Alignment Score (MAS). The fraction of tokens that exactly match a gold morpheme,

$$\mathrm{MAS}(\tau) = \frac{\sum_{w\in\mathcal{W}} \sum_{i=1}^{|\tau(w)|} \mathbf{1}[t_i \in \mu(w)]}{\sum_{w\in\mathcal{W}} |\tau(w)|}$$

Boundary Violation Rate (BVR). The canonical, token-level quantity reported in all tables, defined as the fraction of non-empty tokens that fail to match any reference morpheme (BVR $\approx 1 - $ MAS),

$$\mathrm{BVR}(\tau) = \frac{\sum_{w\in\mathcal{W}} \sum_{i=1}^{|\tau(w)|} \mathbf{1}[t_i \notin \mu(w) \ \wedge \ t_i \neq \varepsilon]}{\sum_{w\in\mathcal{W}} |\tau(w)|}$$

Boundary-level violation rate (bBVR) and gold-boundary recall (RecB). The position-level quantities of Section 2.1, which measure the two directions of boundary alignment separately. bBVR is the fraction of token cuts that are not morpheme boundaries (wrong cuts), and RecB is the fraction of morpheme boundaries that are realised as token cuts (the complement of missing cuts),

$$\mathrm{bBVR}(\tau) = \frac{\sum_w \mathrm{freq}(w)\,|\mathcal{P}_\tau(w) \setminus \mathcal{B}(w)|}{\sum_w \mathrm{freq}(w)\,|\mathcal{P}_\tau(w)|}, \qquad \mathrm{RecB}(\tau) = \frac{\sum_w \mathrm{freq}(w)\,|\mathcal{P}_\tau(w) \cap \mathcal{B}(w)|}{\sum_w \mathrm{freq}(w)\,|\mathcal{B}(w)|}.$$

Token-level MAS and BVR, and bBVR, cannot distinguish a tokenizer that respects morpheme boundaries from one that rarely splits, because a tokenizer that emits whole words makes no wrong cuts. RecB is the quantity that penalises such morpheme fusion, which is why we report it alongside them.

Suffix Coverage (SC). Proportion of words for which the oracle identifies at least one suffix,

$$\mathrm{SC} = \frac{|\{w \in \mathcal{W} : |\mu(w)| > 1\}|}{|\mathcal{W}|}$$

Suffix Consistency (SCon). Measures whether the same suffix is tokenized uniformly across the words it occurs in. For each observed suffix $s \in \mathcal{S}^+$, let $R_\tau(s)$ be the set of distinct subword renderings of s produced by τ across the words w in which s occurs as a morpheme. We report

$$\mathrm{SCon}_\tau(s) = |R_\tau(s)|, \qquad \overline{\mathrm{SCon}}(\tau) = \frac{1}{|\mathcal{S}^+|} \sum_{s\in\mathcal{S}^+} |R_\tau(s)|,$$

where $\mathcal{S}^+ = \{s \in \mathcal{S} : s \text{ occurs in at least one word}\}$. Lower is more consistent, and a tokenizer that always renders s the same way attains $\mathrm{SCon}_\tau(s) = 1$. The suffix is scored in context, inside the words it occurs in, because a suffix encoded in isolation always receives the same tokenization. Figure 2 and Appendix F.6 report this quantity.

C Extrinsic Task and LM Metric Descriptions

All downstream tasks are designed to probe whether improved tokenization quality translates into measurable gains on standard NLP benchmarks. Each task isolates a different linguistic competency, syntax (POS), named entities (NER), document semantics (classification), and sentence-level semantics (similarity), to test whether MRT's morphological regularisation benefits multiple levels of representation. Architectures are deliberately kept small and deterministic (fixed seeds) so that performance differences are attributable to the tokenizer rather than model capacity or stochasticity.

Language Modelling. For each tokenizer τ with vocabulary V, a language model p_θ is trained from scratch on the training split and scored on the held-out split. Writing $\mathrm{NLL}_{\mathrm{total}} = -\sum_{i=1}^{T} \log p_\theta(t_i \mid t_{<i})$ for the summed held-out negative log-likelihood in nats, T for the number of tokens τ emits on the held-out text, and C, B, W for its numbers of characters, UTF-8 bytes and whitespace words, we report

$$\mathrm{BPC} = \frac{\mathrm{NLL}_{\mathrm{total}}}{C \ln 2}, \qquad \mathrm{BPB} = \frac{\mathrm{NLL}_{\mathrm{total}}}{B \ln 2},$$
$$\mathrm{PPL}_{\mathrm{word}} = \exp\Big(\frac{\mathrm{NLL}_{\mathrm{total}}}{W}\Big), \qquad \mathrm{PPL}_{\mathrm{token}} = \exp\Big(\frac{\mathrm{NLL}_{\mathrm{total}}}{T}\Big).$$

The first three divide the same loss by a denominator that depends only on the text and are comparable across tokenizers, and BPC is our primary metric. $\mathrm{PPL}_{\mathrm{token}}$ divides by the tokenizer's own token count, so a tokenizer that emits more tokens is scored on a finer, easier prediction problem, so it is reported only as a secondary, labelled quantity. For the masked LM, only the ≈15% masked positions are scored, and $\mathrm{NLL}_{\mathrm{total}}$ is extrapolated as the mean masked-token NLL times T, so the resulting pseudo-BPC equals (mean masked NLL) $\times\, T/(C \ln 2)$ and is therefore proportional to fertility by construction (Appendix F.3). The three LM configurations are listed in Table 8.

POS Tagging. A BiLSTM (Graves & Schmidhuber, 2005) tagger (embedding 64, hidden 128, 2 layers) trained on Universal Dependencies (Nivre et al., 2020) treebanks. Accuracy on test splits. This measures whether coherent tokenization preserves syntactic information.

Named Entity Recognition. A two-layer BiLSTM NER tagger on WikiANN (PAN-X) (Pan et al., 2017) with BIO tagging (PER, ORG, LOC). Each word is represented by the hidden state at its first subword. Entity-level P/R/F1. NER is sensitive to tokenization because entity boundaries must be recovered from subword boundaries.

Text Classification. A 1D-CNN (Kim, 2014) classifier (filter widths $\{2, 3, 4, 5\}$, 128 filters each) on XL-Sum (Hasan et al., 2021) topic labels. Accuracy and macro-F1. Probes whether coherent tokenization yields more informative sentence representations.

Sentence Similarity. A BiLSTM encoder with mean-pooling trained via InfoNCE (van den Oord et al., 2018) contrastive loss on XL-Sum title-summary pairs. Evaluated via MRR and Recall@k ($k \in \{5, 10\}$). Tests whether aligned tokens produce more semantically discriminative sentence embeddings.

D Detailed Experimental Setup

D.1 Setup Details Omitted from the Main Text

Data. We evaluate on 9 Indic languages from two families, Indo-Aryan (Assamese, Bengali, Gujarati, Hindi, Marathi, Nepali, Punjabi) and Dravidian (Tamil, Telugu), using two monolingual corpora, (i) **Wikipedia** (HuggingFace `wikimedia/wikipedia`, snapshot `20231101`) and (ii) the **Sangraha-Verified** web corpus (Khan et al., 2024), a quality-filtered Indic web crawl covering heterogeneous web text. We claim no data contribution. After script-specific cleaning, we take *nested* subsets of 100, 500, 10K, 100K and all lines from the top of each cleaned corpus, so that every smaller split lies inside the larger ones, and hold out a disjoint validation set. Tokenizers and LMs are trained per corpus. For the language-model comparison of Section 4.2, held-out lines that repeat a training line exactly, or share at

least 80% of their word 5-grams with the training split, are removed before scoring. This filter removes 62% of the Telugu and 17% of the Gujarati Wikipedia held-out lines, which contain bot-generated, census-templated village stubs, and at most 4% of any other split (Appendix D.2). Sampling indices and seeds are released with the code.

Language modelling. For each tokenizer we train a language model *from scratch*, with no pre-trained weights, so that the tokenizer is the only variable within a language. We use three configurations, namely a 2-layer, 128-d GPT-2 (Radford et al., 2019) (**GPT-2 2L**, 2 heads, context 128), used for all nine languages on both corpora at the 10K split, for additional Bengali split sizes, and for the λ sweep, an 8-layer, 512-d GPT-2 (**GPT-2 8L**, 30-42M parameters), and a 12-layer, 768-d BERT-base masked LM (Devlin et al., 2019) (92-110M parameters). For the 8-layer GPT-2 and BERT-base, the encoded corpus is packed into one token stream cut into 128-token windows and scored with a sliding window, so every tokenizer is trained and scored on identical text, whereas the 2-layer configuration encodes each line separately, truncated at 128 tokens (Appendix D.4). Our *primary* LM metric is bits per character, $\mathrm{BPC} = \mathrm{NLL}_{\mathrm{total}}/(C \ln 2)$, where $\mathrm{NLL}_{\mathrm{total}}$ is the summed held-out negative log-likelihood and C is the number of held-out characters. C is fixed by the text, not the tokenizer, so it is identical for all tokenizers within a language and corpus. Per-token perplexity, $\exp(\mathrm{NLL}_{\mathrm{total}}/T)$, divides by the number of tokens T that the tokenizer under test emits, so it rewards fragmentation and is not comparable across tokenizers, so we report it only as a secondary quantity, labelled as such.

D.2 Data

Corpora and preprocessing. For each of the 9 Indic languages with full extrinsic coverage, we build two parallel corpora, (i) a **Wikipedia** dump via HuggingFace's `wikimedia/wikipedia` dataset (snapshot `20231101`), and (ii) the **Sangraha** web corpus, a quality-filtered Indic web crawl. Both pass through the same pipeline. Raw lines are filtered with a script-specific Unicode regex to remove non-native characters (e.g., Latin loanwords in Bengali text), and lines with fewer than 10 words are discarded to eliminate stubs, headings, and table fragments.

Nested subsets of increasing size (100, 500, 10K, 100K and all lines) are taken from the top of each cleaned corpus, so that each smaller split is contained in the larger ones and the held-out set stays disjoint from every training split. We sample in order rather than at random so that splits are reproducible and nested, and the line indices and the seed are released with the code. For Bengali Wikipedia, the full corpus contains approximately 104,781 lines after cleaning, and the Sangraha Bengali corpus is roughly an order of magnitude larger before subsampling. The 10K split used for all language-model comparisons has 9,000 training and 1,000 held-out lines per language. Table 3 gives the held-out character counts C, which depend only on the text and are therefore identical for all tokenizers and MYTE within a language and corpus, and Gujarati and Telugu Wikipedia are scored on their overlap-filtered subsets (Appendix D.2). Tokenizers and LMs are trained *per corpus* (a Wikipedia tokenizer is never used to encode Sangraha text or vice versa), so reported gains reflect within-corpus generalisation rather than cross-domain transfer.

Morphological segmentation is applied using the rule-based longest-suffix-match oracle (Section 2.2). The segmented corpus (with whitespace-marked boundaries) is used exclusively for MRT-B training, and all other tokenizers receive the unsegmented raw corpus.

Train–held-out overlap filtering. Before scoring, we scan every committed split for held-out lines that also occur in training. A held-out line is removed if it occurs verbatim in the training split (*exact*), if at least 80% of its word 5-grams occur in the training split (*near-duplicate*, which catches templated text that differs only in names and numerals), or if it repeats an earlier held-out line. Table 3 reports the scan for all 18 language-corpus splits at 10K. Two splits are affected. Telugu Wikipedia is dominated by bot-generated village articles written from census templates, whose sentences about electricity supply, sanitation, transport and demographics recur across thousands of articles. In total, 33% of its training lines repeat another training line, and 62% of its held-out lines are removed. Gujarati Wikipedia loses 17% of held-out lines. Every other split, including Telugu Sangraha, is

at least 96% clean and is scored on its full held-out set. All tokenizers are scored on the same filtered subset, and BPC's denominator is that subset's character count, so the filtered scores remain comparable across tokenizers.

Table 3: Held-out data (10K split), with character count C before filtering, train–held-out overlap, and what the overlap filter retains.

Lang.	Corpus	Held-out C	Train dup	Exact	Near-dup	Retained lines	Retained chars
as	Wiki	326,634	0.1%	0.6%	1.0%	990 (99.0%)	324,279 (99.3%)
bn	Wiki	292,402	2.0%	1.0%	1.6%	984 (98.4%)	291,351 (99.6%)
gu	Wiki	331,329	3.0%	3.9%	17.1%	829 (82.9%)	266,315 (80.4%)
hi	Wiki	323,245	0.2%	0.5%	0.9%	991 (99.1%)	322,211 (99.7%)
mr	Wiki	264,202	0.2%	0.5%	0.9%	991 (99.1%)	262,988 (99.5%)
ne	Wiki	327,160	0.6%	1.1%	4.0%	960 (96.0%)	322,762 (98.7%)
pa	Wiki	279,918	0.7%	0.3%	3.3%	967 (96.7%)	277,035 (99.0%)
ta	Wiki	295,067	0.7%	0.8%	1.4%	986 (98.6%)	293,146 (99.3%)
te	Wiki	259,875	33.4%	40.0%	61.9%	379 (37.9%)	98,917 (38.1%)
as	Sangraha	311,575	0.4%	0.6%	0.6%	994 (99.4%)	311,043 (99.8%)
bn	Sangraha	287,541	0.4%	0.6%	0.6%	994 (99.4%)	287,032 (99.8%)
gu	Sangraha	301,750	0.3%	0.9%	0.9%	991 (99.1%)	299,636 (99.3%)
hi	Sangraha	310,215	2.2%	2.5%	2.6%	974 (97.4%)	304,944 (98.3%)
mr	Sangraha	337,197	0.7%	0.9%	1.1%	989 (98.9%)	335,788 (99.6%)
ne	Sangraha	256,705	0.2%	0.1%	0.1%	999 (99.9%)	256,654 (100.0%)
pa	Sangraha	310,304	0.4%	0.5%	0.5%	995 (99.5%)	310,038 (99.9%)
ta	Sangraha	273,088	0.1%	0.4%	0.4%	996 (99.6%)	272,650 (99.8%)
te	Sangraha	337,070	0.6%	1.1%	1.1%	989 (98.9%)	334,964 (99.4%)

Telugu on both corpora. Table 4 places the Telugu results on the two corpora side by side under the same metric and the same 8-layer GPT-2 protocol. On both, MRT-P has the lowest BPC. The margin over the strongest baseline is 0.2% on the filtered Wikipedia set, which rests on 379 lines and should be read as a tie, and 2.5-2.6% on Sangraha, which rests on about 1,000 lines. Telugu Sangraha runs at 2.05-2.17 BPC, in the same range as the other eight languages on Wikipedia (1.82-2.11 for the best tokenizer). We therefore regard the quality-filtered Sangraha split as the better-founded Telugu comparison.

Table 4: Telugu held-out BPC (↓) of GPT-2 8L (10K) on Wikipedia and Sangraha. The margin compares MRT-P with the strongest of BPE, Uni, WP and Morf (negative = MRT-P better).

	Wikipedia (filtered)	Sangraha (full)	Sangraha (filtered)
C	98,917	337,070	334,964
BPE	1.9041	2.1579	2.1650
Unigram	1.8572	2.1006	2.1069
WordPiece	1.8799	2.1456	2.1545
Morfessor	1.8702	2.1539	2.1612
MRT-B	1.8722	2.0898	2.0978
MRT-P	**1.8532**	**2.0453**	**2.0544**
Margin	−0.22%	−2.63%	−2.49%

Data sources and citations. *Corpora and task data.* Sangraha-Verified via IndicLLM-Suite (Khan et al., 2024), Wikipedia (`wikimedia/wikipedia`, snapshot `20231101`) (Wikimedia Foundation, 2023), WikiANN / PAN-X (Pan et al., 2017) and XL-Sum (Hasan et al., 2021).

Universal Dependencies treebanks (Nivre et al., 2020), used for POS tagging and as a source of external gold boundaries, namely Hindi-HDTB (Bhat et al., 2017; Palmer et al., 2009), Tamil-TTB (Ramasamy & Žabokrtský, 2012), Gujarati-GujTB (Jobanputra et al., 2024), Marathi-UFAL (Ravishankar, 2017), Telugu-MTG (Rama & Vajjala, 2017) (whose sentences are drawn from Krishnamurti & Gwynn (1985)) and Bengali-Sabdakosh (Dyer et al., 2024). Assamese, Nepali and Punjabi have no UD treebank.

UniMorph inflection tables for Assamese (`asm`), Bengali (`ben`), Gujarati (`guj`), Hindi (`hin`) and Telugu (`tel`) (Batsuren et al., 2022; Kirov et al., 2018).

D.3 Tokenizers and λ Selection

Tokenizer configurations. Table 5 lists the six tokenizers trained per language and per split, along with their underlying method and the form of training input each receives. MYTE (Appendix F.2) is a fixed byte encoding and is not trained.

Table 5: Tokenizers trained per language per split.

Tokenizer	Method	Training Input
BPE	Byte Pair Encoding	Raw cleaned corpus
Unigram	Unigram LM	Raw cleaned corpus
MRT-B	BPE	Morphologically segmented corpus
MRT-P	Normalised penalised BPE ($\lambda \in [5.0, 7.0]$)	Raw corpus + boundary annotations
WordPiece	WordPiece	Raw cleaned corpus
Morfessor	Unsupervised morphology	Raw cleaned corpus

Vocabulary size is determined dynamically per language and split via VOLT (Xu et al., 2021), which selects K from $\{1000, 2000, 4000, 8000, 16000, 32000\}$ by maximising an entropy-based utility metric, and candidates exceeding corpus capacity (90% of unique tokens) are filtered. The fallback is $K = 8{,}000$. SentencePiece (Kudo & Richardson, 2018) is used as backend for BPE, Unigram, and MRT-B. MRT-P uses a custom implementation exporting SentencePiece-compatible unigram models.

λ hyperparameter selection. Six tokenizers are compared throughout, namely BPE, Unigram, WordPiece, Morfessor, MRT-B and MRT-P, with MYTE added for the language-model comparison. The λ sweep uses the GPT-2 2L configuration (Appendix D.4), and downstream tasks use lightweight BiLSTM/CNN models.

The primary hyperparameter is λ in MRT-P. Because the normalised scoring function $\text{score} = f \cdot \max(0, 1 - \lambda r)$ is scale-independent, λ has a consistent interpretation across corpus sizes. Setting $\lambda = 1$ zeros out pairs where 100% of occurrences cross a boundary, $\lambda = 2$ blocks pairs with $\geq 50\%$ crossing rate, and $\lambda = 5$ blocks pairs with $\geq 20\%$ crossing rate. Selection proceeds in two stages. On Bengali Wikipedia (10K) we sweep $\lambda \in \{0.5, 1.0, \ldots, 9.0\}$ in steps of 0.5 and evaluate the intrinsic metrics, LM perplexity and the downstream tasks at each point (Section 5). For every language and corpus we then search $\lambda \in \{5.0, 5.5, 6.0, 6.5, 7.0\}$, the band around the Bengali optimum, and the selected values (5.5-7.0) are listed in Table 11 and in the per-language tables. The metric-specific optimum is $\lambda^*_m = \arg\text{opt}_\lambda \; m(\tau_\lambda)$, where opt = arg max for alignment/consistency metrics and opt = arg min for violation/fragmentation/perplexity. In practice we select the λ that minimises held-out LM loss, as a task-agnostic intrinsic criterion rather than as a proxy for downstream accuracy, since the downstream optima differ by task (Section 5). Exact values for the Bengali sweep are in Appendix F.8.

D.4 Model Training Details

Language models. All language models are trained from scratch, with no pre-trained weights and no shared initialisation across tokenizers. Table 8 lists the three configurations, and Appendix E.2 reports training times.

GPT-2 2L. The configuration used for the full grid of languages, corpora, split sizes and the λ sweep. Each line is encoded separately and truncated at 128 tokens, so a finer-grained tokenizer is trained and scored on somewhat less text than a coarse one, and per-token perplexity is evaluated on the held-out split after the final epoch (results in Appendix F.9).

GPT-2 8L and BERT-base. The configurations used for the tokenizer-invariant comparison (Wikipedia 10K, all nine languages, 108 runs, plus MYTE). The encoded training text is packed into a single stream with an end-of-line token after every line and cut into 128-token

windows, so every tokenizer is trained on identical text. Causal models are scored with a sliding window of stride 64, so every held-out token is scored exactly once with at least 64 tokens of left context, and the NLL is summed token-weighted rather than averaged over batches. The masked LM uses standard 80/10/10 masking at 15%, with a fixed generator at validation so that the masked subset is identical across tokenizers and epochs. The reported checkpoint is the epoch with the lowest held-out NLL per token.

Downstream tasks. *POS tagging.* BiLSTM tagger (embedding 64, hidden 128, 2 layers, bidirectional), 15 epochs, AdamW (lr 5×10^{-3}). Bengali uses the merged UD_Bengali-BRU (56 sent.) and UD_Bengali-Sabdakosh (252 sent.) from the Universal Dependencies treebanks (Nivre et al., 2020), 246 train / 62 test after 80/20 split.

NER. Two-layer BiLSTM on WikiANN (PAN-X) (Pan et al., 2017), 20 epochs. Bengali has 10K train / 1K val / 1K test. BIO scheme (PER, ORG, LOC).

Text classification. 1D-CNN (filter widths $\{2, 3, 4, 5\}$, 128 filters), 15 epochs on XL-Sum (Hasan et al., 2021) (∼14K train / 3K val for Bengali). Proxy task for languages without XL-Sum.

Sentence similarity. BiLSTM + mean-pooling, InfoNCE (van den Oord et al., 2018) loss, 8 epochs, batch 64, on XL-Sum title-summary pairs (∼16K train / 4K eval for Bengali).

E Implementation Details

E.1 Suffix Statistics and Degree of Suffixation

Table 6 summarises the curated suffix list size, suffix coverage, and average morphemes-per-word produced by the oracle for each of the nine languages.

Table 6: Suffix-list statistics (Wikipedia 10K), with list size $|\mathcal{S}|$, suffix coverage SC and mean morphemes per word $\overline{|\mu|}$.

Language	**Code**	**Script**	$\lvert\mathcal{S}\rvert$	SC	$\overline{\lvert\mu\rvert}$
Assamese	`as`	Bengali	11	0.291	1.34
Bengali	`bn`	Bengali	44	0.262	1.29
Gujarati	`gu`	Gujarati	15	0.253	1.27
Hindi	`hi`	Devanagari	26	0.123	1.13
Marathi	`mr`	Devanagari	17	0.211	1.22
Nepali	`ne`	Devanagari	12	0.231	1.24
Punjabi	`pa`	Gurmukhi	13	0.077	1.08
Tamil	`ta`	Tamil	16	0.133	1.13
Telugu	`te`	Telugu	15	0.291	1.34

Degree of suffixation and where MRT helps. All nine languages are predominantly suffixing. Indo-Aryan languages mark case, number and verbal categories largely with postpositions and suffixes (Masica, 1991), and the Dravidian languages Tamil and Telugu are agglutinative, stacking case and plural suffixes on nominal stems (Krishnamurti, 2003). Within this common profile, the languages differ in how much morphology the curated oracle actually exposes. Table 6 gives two corpus-level indicators, suffix coverage SC, the fraction of word types in which the oracle finds at least one suffix, and the mean number of morphemes per word $\overline{|\mu|}$. By these measures Telugu and Assamese (SC = 0.291, $\overline{|\mu|} = 1.34$), Bengali and Gujarati expose the most suffixation, and Punjabi (0.077, 1.08), Hindi and Tamil the least. These indicators describe the oracle as much as the language. The Tamil list, for example, marks only 0.09 boundaries per word type against 1.06 in UD gold (Appendix F.4), so Tamil's low SC reflects the list rather than the morphology. With that caveat, the pattern is suggestive rather than conclusive. The two languages in which an MRT variant achieves the best score on all five downstream metrics, Telugu and Gujarati, are among those with the highest suffix coverage, and the BERT-base per-token reductions of Table 14(a) are smallest for Hindi and Punjabi, the two Indo-Aryan languages with the lowest coverage.

E.2 Architectures and Compute

Table 7 specifies the architecture and capacity of the downstream evaluation models, and Table 8 lists the three language-model configurations.

Table 7: Downstream model architectures.

Task	Model	Hidden dim	Layers	Other
POS Tagging	BiLSTM	128	2	Embedding 64, Bidirectional
NER	BiLSTM	128	2	Embedding 64, BIO scheme
Text Classification	1D-CNN	128 filters	1	Filter widths $\{2, 3, 4, 5\}$
Sentence Similarity	BiLSTM + mean-pool	256	2	InfoNCE loss, temperature $\tau = 0.07$
Language Model	GPT-2 / BERT	128-768	2-12	see Table 8

Table 8: Language-model configurations, all trained from scratch with AdamW. Parameter counts vary with vocabulary size.

Setting	GPT-2 2L	GPT-2 8L	BERT-base
Objective	causal LM	causal LM	masked LM (15%)
Layers / heads / hidden	2 / 2 / 128	8 / 8 / 512	12 / 12 / 768 (FFN 3072)
Context length	128	128	128
Text layout	per line, truncated	packed stream	packed stream
Dropout	–	0.1	0.1
Peak learning rate	5×10^{-4}	3×10^{-4}	1×10^{-4}
Schedule, weight decay	–	cosine, warmup 0.1, wd 0.01	cosine, warmup 0.1, wd 0.01
Batch size, epochs	16, 15	16, 15	16, 15
Parameters	–	29.6M-41.7M	92.2M-110.4M
Seed	42	42	42

Compute requirements. All experiments were conducted on an NVIDIA DGX Spark with 120 GB VRAM. The full pipeline for one language with all five splits, six tokenizers, and a single GPT-2 2L LM per cell requires approximately the wall-clock times in Table 9.

Table 9: Estimated wall-clock time per language.

Phase	Estimated Wall-clock Time
Data download + preprocessing	10–30 min per language
Tokenizer training (all methods, all splits)	5–20 min per language
GPT-2 LM training (all tokenizers, all splits)	1–3 hours per language
Downstream tasks (POS + NER + CLF + SIM)	1–3 hours per language
Total (single language, full pipeline)	2–6 hours

The tokenizer-invariant comparison (Section 4.2) comprises 108 runs (9 languages $\times$ 6 tokenizers $\times$ 2 models) on Wikipedia 10K. The 54 GPT-2 8L runs took 1h32m in total (mean 1m42s per run) and the 54 BERT-base runs 3h02m (mean 3m23s per run).

F Extended Experimental Results

This section presents the full set of experimental results referenced in the main text. We first give the complete tokenizer-invariant language-model results (the 8-layer GPT-2, MYTE, BERT-base, the external-gold evaluation, the suffix-list noise ablation and the shared multilingual vocabulary), and then the downstream, Sangraha, λ-sweep and small-scale (GPT-2 2L) per-token perplexity tables, followed by the fidelity-vs-word-length analysis. Unless stated otherwise, the 8-layer GPT-2 and BERT-base runs use the Wikipedia 10K split and seed 42, and Gujarati and Telugu Wikipedia are scored on their overlap-filtered held-out sets (Appendix D.2).

F.1 All Language-Model Metrics for GPT-2 8L

Table 1 in the main text gives bits per character. Table 10(a) and (b) give the two other tokenizer-invariant normalisations of the same held-out loss, bits per UTF-8 byte and per-word perplexity. Within a language, BPB is BPC divided by a constant number of bytes per character, so its ordering and relative margins are identical to BPC, and it is the figure to use when comparing with byte-level results reported elsewhere. Per-word perplexity exponentiates the loss per whitespace word, so the same relative reduction in loss reads larger. All three give MRT-P the best score in the same 7 of 9 languages, with Bengali and Hindi the exceptions, and the mean per-word perplexity change is −10.7%. Panel (c) of the same table gives per-token perplexity, which is not comparable across tokenizers (MRT-P is lowest in all nine languages, 59% below the strongest baseline on average), and Table 11 the fertility and vocabulary size that explain why. Appendix F.11 breaks fertility down by word length and Appendix G.2 analyses vocabulary composition. MRT-P's SentencePiece export caps the vocabulary at 55% of the distinct pre-segmented pieces, so it lands near half the size of the baselines and vocabulary size is not matched across tokenizers.

Table 10: GPT-2 8L held-out metrics (Wikipedia 10K). Δ compares MRT-P with the strongest of BPE, Uni, WP and Morf. Per-token perplexity is not comparable across tokenizers.

Lang.	BPE	Uni	WP	Morf	MRT-B	MRT-P	Δ
(a) Bits per byte (↓)							
Assamese	0.7510	0.7337	0.7472	0.7475	0.7315	**0.7129**	
Bengali	0.7174	**0.6903**	0.7157	0.7133	0.7090	0.6906	
Gujarati	0.8115	0.7916	0.8028	0.7902	0.7988	**0.7774**	
Hindi	0.7820	**0.7607**	0.7766	0.7752	0.7765	0.7631	
Marathi	0.8018	0.7915	0.7998	0.7992	0.7971	**0.7772**	
Nepali	0.7473	0.7296	0.7429	0.7289	0.7258	**0.7105**	
Punjabi	0.7858	0.7715	0.7830	0.7826	0.7831	**0.7707**	
Tamil	0.6854	0.6714	0.6781	0.6690	0.6687	**0.6529**	
Telugu	0.6933	0.6762	0.6845	0.6809	0.6816	**0.6747**	
(b) Per-word perplexity (↓)							
Assamese	7,484.3	6,098.9	7,160.0	7,186.7	5,937.0	**4,765.0**	−21.9%
Bengali	6,352.6	**4,564.1**	6,219.6	6,040.7	5,736.5	4,582.8	+0.4%
Gujarati	8,958.3	7,159.7	8,125.0	7,049.6	7,766.9	**6,106.2**	−13.4%
Hindi	1,504.7	**1,232.1**	1,430.2	1,411.3	1,429.0	1,260.2	+2.3%
Marathi	23,586.4	20,742.0	22,997.7	22,844.1	22,234.4	**17,334.6**	−16.4%
Nepali	6,677.4	5,419.3	6,337.6	5,373.2	5,182.1	**4,325.8**	−19.5%
Punjabi	1,278.0	1,121.1	1,245.7	1,240.3	1,246.2	**1,113.7**	−0.7%
Tamil	167,166.8	130,919.7	147,216.7	125,535.7	124,762.9	**94,546.1**	−24.7%
Telugu	24,610.4	19,189.4	21,647.5	20,552.6	20,777.3	**18,782.6**	−2.1%
(c) Per-token perplexity							
Assamese	1,352.0	722.3	1,115.6	816.7	384.7	182.3	
Bengali	1,256.4	404.2	1,063.0	757.2	435.9	182.1	
Gujarati	1,217.7	526.6	985.1	693.9	444.0	176.4	
Hindi	473.7	179.8	417.3	364.4	275.4	118.4	
Marathi	1,926.7	936.5	1,599.6	1,018.4	866.1	302.5	
Nepali	834.8	387.3	720.0	487.5	383.0	138.9	
Punjabi	555.4	232.7	495.5	436.1	354.4	151.8	
Tamil	3,025.4	1,294.7	2,178.6	1,051.2	1,583.2	425.5	
Telugu	791.1	365.7	618.0	455.7	364.2	93.2	

F.2 MYTE Baseline

MYTE (Limisiewicz et al., 2024) is run from the upstream code, vendored unmodified, and its encoder is checked against the released MyT5 tokenizer before training. It is trained and scored under exactly the protocol of the subword tokenizers (same splits, same GPT-2 8L and BERT-base configurations, same metrics). MYTE differs from the subword baselines

Table 11: Fertility (tokens per whitespace word) and vocabulary size $|V|$ on Wikipedia 10K, where λ is MRT-P's per-language penalty.

		Fertility						**Vocabulary size $\|V\|$**					
Lang.	λ	BPE	Uni	WP	Morf	MRT-B	MRT-P	BPE	Uni	WP	Morf	MRT-B	MRT-P
as	6.5	1.24	1.32	1.26	1.32	1.46	1.63	32,000	32,000	26,460	21,954	32,000	16,282
bn	7.0	1.23	1.40	1.25	1.31	1.42	1.62	32,000	16,000	26,128	21,245	32,000	16,033
gu	6.0	1.23	1.38	1.25	1.29	1.44	1.67	32,000	32,000	26,567	24,048	32,000	15,996
hi	6.5	1.19	1.37	1.20	1.23	1.29	1.50	32,000	16,000	26,092	21,915	32,000	15,474
mr	7.0	1.33	1.45	1.36	1.45	1.48	1.71	32,000	32,000	26,526	22,304	32,000	16,841
ne	6.0	1.31	1.44	1.33	1.39	1.44	1.70	32,000	32,000	26,921	23,392	32,000	16,530
pa	5.5	1.13	1.29	1.15	1.17	1.21	1.40	32,000	16,000	25,086	21,357	32,000	15,114
ta	6.0	1.50	1.64	1.55	1.69	1.59	1.89	32,000	32,000	27,108	20,993	32,000	17,262
te	6.0	1.21	1.42	1.23	1.26	1.40	1.86	16,000	16,000	13,288	13,869	16,000	8,335

in two ways at once, since it is byte-level and its byte code is morphologically re-encoded, so a BPC difference against a subword tokenizer cannot on its own be attributed to the morphological byte maps. Some quantities are also not held equal, as a consequence of the method rather than of our setup. MYTE has a 260-symbol vocabulary, so its GPT-2 8L has 25.4M parameters against 29.6-41.7M for the subword models, and a 128-token window holds 78-132 characters of MYTE text against 441-775 for the subword tokenizers.

Sequence length. MYTE's central claim is that its encoding is shorter than UTF-8, especially for non-Latin scripts. Table 12 confirms this on our data, as MYTE is shorter than raw UTF-8 for all nine languages, by 56.1% on average, while still emitting 5.1-8.4 times as many tokens as BPE.

Table 12: Sequence length on the Wikipedia 10K held-out split. *vs. UTF-8* is the length reduction relative to raw bytes, and *vs. BPE* is MYTE tokens per BPE token.

Lang.	**Chars**	**UTF-8 bytes**	**MYTE tokens**	**vs. UTF-8**	**vs. BPE**
Assamese	326,634	879,288	536,116	−39.0%	8.44×
Bengali	292,402	789,536	323,960	−59.0%	5.89×
Gujarati	331,329	885,689	389,888	−56.0%	5.73×
Hindi	323,245	846,325	391,629	−53.7%	5.26×
Marathi	264,202	715,608	282,766	−60.5%	5.38×
Nepali	327,160	879,960	344,642	−60.8%	5.09×
Punjabi	279,918	730,498	374,742	−48.7%	5.95×
Tamil	295,067	822,243	307,631	−62.6%	6.31×
Telugu	259,875	710,073	251,618	−64.6%	5.80×

Modelling quality. Table 13 compares MYTE with MRT-P and with the strongest subword tokenizer (best of BPE, Uni, WP, Morf, MRT-B and MRT-P, where positive Δ means MYTE is worse). MYTE was scored only on the unfiltered held-out splits, so all values in that table use them, and its Gujarati and Telugu rows differ from Table 1. Under the causal LM, MYTE does not achieve the best score in any language and trails the strongest subword tokenizer by 6.4% BPC on average. Under the masked LM, its pseudo-BPC is 2.1-3.2 times that of the strongest subword tokenizer in eight languages. The exception is unfiltered Telugu Wikipedia, where MYTE's 0.8939 is below every subword tokenizer (best 0.9805), on a held-out set dominated by templated text that recurs in training (Appendix D.2). The extrapolation behind pseudo-BPC scales with the number of tokens (Appendix F.3), which penalises a byte encoding heavily, so we do not read the masked-LM column as a modelling result. **Encoding fidelity.** MYTE's decoder maps a consonant followed by a nukta to the precomposed character, whereas our NFC text keeps the two-codepoint form (these are Unicode composition exclusions). This affects 48-91% of held-out lines in Assamese, Bengali, Hindi and Punjabi and none in Gujarati, Tamil and Telugu. Because BPC divides by the character count of the original text, MYTE's BPC on the four affected languages is, if anything, very slightly flattered.

Table 13: MYTE vs. MRT-P and the strongest subword tokenizer on the unfiltered held-out splits. Δ is MYTE relative to the strongest subword tokenizer.

	GPT-2 8L, BPC				BERT-base, pseudo-BPC	
Lang.	MYTE	MRT-P	best subword	Δ	MYTE	best subword
Assamese	2.0639	1.9192	1.9192 (MRT-P)	+7.5%	7.0757	2.2348 (BPE)
Bengali	1.9968	1.8648	1.8639 (Uni)	+7.1%	5.0952	2.1723 (WP)
Gujarati	1.8094	1.7094	1.7094 (MRT-P)	+5.9%	5.2658	1.9820 (BPE)
Hindi	2.1354	1.9980	1.9916 (Uni)	+7.2%	5.4168	2.3089 (BPE)
Marathi	2.2233	2.1052	2.1052 (MRT-P)	+5.6%	4.9275	2.3653 (WP)
Nepali	2.0615	1.9110	1.9110 (MRT-P)	+7.9%	4.8392	2.2974 (BPE)
Punjabi	2.1383	2.0114	2.0114 (MRT-P)	+6.3%	5.9222	2.2891 (BPE)
Tamil	1.8601	1.8194	1.8194 (MRT-P)	+2.2%	4.9086	2.0264 (Uni)
Telugu	0.8209	0.7639	0.7639 (MRT-P)	+7.5%	0.8939	0.9805 (Uni)

F.3 BERT-base Pseudo-Bits per Character

Table 14: BERT-base masked LM (Wikipedia 10K). Δ compares MRT-P with the strongest of BPE, Uni, WP and Morf.

Lang.	BPE	Uni	WP	Morf	MRT-B	MRT-P	Δ
(a) Per-token pseudo-perplexity (↓), not tokenizer-invariant							
Assamese	2,892.8	1,727.8	2,670.9	2,124.1	1,140.1	**813.8**	−52.9%
Bengali	5,495.6	1,268.4	2,525.3	2,197.9	1,371.4	**882.2**	−30.4%
Gujarati	803.3	443.4	770.1	623.4	412.8	**260.6**	−41.2%
Hindi	1,039.8	635.5	986.2	893.8	782.3	**548.5**	−13.7%
Marathi	6,769.9	2,077.1	3,151.9	2,290.8	1,848.9	**1,189.8**	−42.7%
Nepali	2,183.6	1,120.1	1,993.5	1,416.7	1,178.5	**748.1**	−33.2%
Punjabi	1,157.4	712.9	1,088.5	1,035.7	920.9	**593.2**	−16.8%
Tamil	9,679.8	2,347.2	7,783.2	5,223.3	3,452.7	**1,394.8**	−40.6%
Telugu	1,231.4	719.5	1,165.8	977.9	810.2	**274.8**	−61.8%
(b) Pseudo-BPC (↓), an estimator that scales with fertility							
Assamese	**2.2348**	2.2368	2.2618	2.2992	2.3284	2.4708	+10.6%
Bengali	2.3377	2.2191	**2.1723**	2.2355	2.2756	2.4303	+11.9%
Gujarati	**1.9820**	2.0222	2.0022	1.9984	2.0785	2.2282	+12.4%
Hindi	**2.3089**	2.4759	2.3232	2.3384	2.4110	2.6394	+14.3%
Marathi	2.5324	2.3935	**2.3653**	2.4181	2.4010	2.6100	+10.3%
Nepali	**2.2974**	2.3121	2.3074	2.2984	2.3211	2.5630	+11.6%
Punjabi	**2.2891**	2.4269	2.3027	2.3329	2.3761	2.5572	+11.7%
Tamil	2.1870	**2.0264**	2.2027	2.2940	2.0606	2.1762	+7.4%
Telugu	**2.0303**	2.0701	2.0656	2.1035	2.1261	2.2952	+13.0%

A masked LM scores only the ≈15% masked positions and has no chain-rule factorisation whose terms sum to the likelihood of the corpus, so no exact character-normalised likelihood exists for it. The corpus NLL can only be extrapolated as the mean masked-token NLL times the number of tokens T, which gives

$$\text{pseudo-BPC} = \frac{\overline{\text{NLL}}_{\text{masked}} \cdot T}{C \ln 2} = \overline{\text{NLL}}_{\text{masked}} \cdot \frac{T/C}{\ln 2}.$$

Pseudo-BPC is therefore directly proportional to fertility, since a tokenizer that emits 1.3× as many tokens must cut its per-masked-token NLL by 23% just to draw level. This is the opposite tilt to per-token perplexity, which rewards fragmentation. Table 14(b) reports pseudo-BPC. MRT-P, the most fragmenting tokenizer (Table 11), has the highest pseudo-BPC among the subword tokenizers in eight of nine languages and trails the strongest baseline by 11.5% on average, while its per-masked-token NLL is the lowest (Table 14(a)). The two estimators bracket the comparison rather than settle it, which is why the tokenizer-invariant claim of this paper rests on the causal LM. In both BERT-base tables, Telugu is scored on its overlap-filtered held-out set, and Gujarati, which was scored only on its unfiltered set under BERT-base, on its full held-out split.

F.4 Evaluation Against External Gold Segmentations

Gold construction. Gold boundaries come from annotation that the suffix lists never saw. (i) *UD multi-word tokens.* Treebanks that expand a surface word into syntactic words (Tamil-TTB, Marathi-UFAL) give a direct gold surface segmentation, and components are aligned left to right, and where sandhi makes the concatenation inexact the boundary is placed at the longest common prefix. (ii) *UD lemmas.* For a suffixing language, the root/inflection boundary is where form and lemma diverge, so the gold boundary is at $|\mathrm{LCP}(\mathrm{form}, \mathrm{lemma})|$. Forms identical to their lemma give an empty gold set, and suppletive pairs with $\mathrm{LCP} < 2$ are dropped. (iii) *UniMorph* inflection tables, with the same LCP derivation. UniMorph is the only external gold for Assamese. Words shorter than four characters are excluded, since the oracle never splits them. Nepali and Punjabi have neither a UD treebank nor a UniMorph table and are not evaluable. Each gold source is evaluated on two word sets, the *gold* set (every analysable word of the gold resource) and the *corpus* set (held-out corpus words that carry a gold analysis, weighted by corpus frequency). Seven languages, two corpora, one or two gold sources and two word sets give 48 settings. Resource citations are in Appendix D.2.

Oracle precision and recall. Table 15 populates the precision and recall of Section 2.2 against this gold. Only 6% of Gujarati UD types carry any boundary (marked ‡ in the table), so precision there mostly measures splits of words the treebank never analysed, whereas recall remains meaningful. Lemma-derived gold marks only the root/inflection cut, so an oracle that correctly strips two suffixes is charged a false positive for the second. The *leftmost-boundary* column, the fraction of word types (where both predict a boundary) on which the oracle's leftmost boundary is the gold one, is not subject to this penalty. Two regimes appear. For Bengali, Hindi, Marathi and Tamil the lists are more precise than complete (precision above recall in every row). When the oracle posits a boundary the placement is usually right (leftmost-boundary agreement 0.54-0.90 in seven of the eight rows and 0.36 for Marathi UD lemma), but it misses most gold boundaries. For Gujarati, Telugu and Assamese the lists are weak in both directions (precision 0.07-0.27, leftmost-boundary agreement 0.08-0.32, except Gujarati UD lemma at 0.65, which rests on the 6% of types that carry a boundary). The Tamil list illustrates one cause, as its entries are citation forms with independent vowels, whereas word-finally the suffixes surface as vowel signs, so the oracle marks only 0.09 boundaries per word type against 1.06 in gold.

Table 15: Oracle boundary precision (P), recall (R) and F1 against external gold, leftmost-boundary agreement, and the implied false-negative rate $\bar{\beta} = 1 - \mathrm{R}$.

Lang.	Gold source	P	R	F1	Leftmost	$\bar{\beta}$
Bengali	UD lemma	0.586	0.495	0.537	0.752	0.505
Bengali	UniMorph	0.590	0.356	0.444	0.635	0.644
Hindi	UD lemma	0.446	0.398	0.421	0.654	0.602
Hindi	UniMorph	0.531	0.356	0.426	0.642	0.644
Marathi	UD multi-word token	0.696	0.233	0.349	0.671	0.767
Marathi	UD lemma	0.300	0.173	0.219	0.362	0.827
Tamil	UD multi-word token	0.902	0.077	0.141	0.902	0.923
Tamil	UD lemma	0.500	0.125	0.200	0.536	0.875
Gujarati	UD lemma‡	0.066	0.341	0.111	0.650	0.659
Gujarati	UniMorph	0.072	0.027	0.039	0.088	0.973
Assamese	UniMorph	0.268	0.248	0.258	0.322	0.752
Telugu	UniMorph	0.097	0.054	0.069	0.080	0.946

Tokenizers against gold. Table 16 counts, over the 48 settings, how often an MRT variant achieves the best score on each metric, and Table 17 gives gold-boundary recall for every setting. Against the oracle, an MRT variant has the highest MAS in 45 of 48 settings. Against gold, the token-level picture is mixed. An MRT variant has the highest MAS and the lowest boundary-level violation rate in only 17 of 48 settings each, and Unigram (16 settings) and BPE (11) often lead on MAS, because MRT-P is the most fragmenting tokenizer and MAS divides by the token count. The quantity that tests morphological fidelity in the missing-cut direction, gold-boundary recall, favours MRT clearly, as an MRT variant is best

in 42 of 48 settings, MRT-P alone in 35. On corpus sets, BPE recovers at most 0.044 of the gold boundaries in any setting. Of the six settings in which a baseline leads, three are Telugu (one of them a tie), consistent with its weak suffix list, two are Tamil and one is Gujarati.

Table 16: Number of the 48 external-gold settings in which MRT-B or MRT-P achieves the best score.

Metric	**MRT variant best in**
MAS against the oracle	45 / 48
MAS against gold	17 / 48
Boundary-level violation rate (bBVR) against gold	17 / 48
Gold-boundary recall (RecB)	**42 / 48** (MRT-P alone in 35)

Table 17: Gold-boundary recall RecB (↑) in all 48 settings. *gold* covers all analysable gold words, and *corpus* covers held-out corpus words with a gold analysis.

Lang.	**Gold**	**Set**	**Corpus**	BPE	Uni	WP	Morf	MRT-B	MRT-P
Assamese	UniMorph	corpus	Sang.	0.044	0.122	0.049	0.065	0.349	**0.358**
Assamese	UniMorph	corpus	Wiki	0.042	0.166	0.047	0.064	0.447	**0.470**
Assamese	UniMorph	gold	Sang.	0.606	0.817	0.627	0.638	0.764	**0.825**
Assamese	UniMorph	gold	Wiki	0.637	0.828	0.668	0.652	0.792	**0.857**
Bengali	UD lemma	corpus	Sang.	0.002	0.079	0.003	0.009	**0.449**	0.308
Bengali	UD lemma	corpus	Wiki	0.003	0.080	0.007	0.014	**0.295**	0.293
Bengali	UD lemma	gold	Sang.	0.127	0.316	0.143	0.177	0.472	**0.501**
Bengali	UD lemma	gold	Wiki	0.269	0.556	0.291	0.333	0.493	**0.620**
Bengali	UniMorph	corpus	Sang.	0.013	0.058	0.014	0.022	**0.308**	0.171
Bengali	UniMorph	corpus	Wiki	0.007	0.067	0.009	0.015	0.201	**0.218**
Bengali	UniMorph	gold	Sang.	0.548	0.530	0.580	0.602	0.659	**0.678**
Bengali	UniMorph	gold	Wiki	0.538	0.718	0.543	0.535	0.652	**0.743**
Gujarati	UD lemma	corpus	Sang.	0.000	**0.160**	0.000	0.009	0.072	0.148
Gujarati	UD lemma	corpus	Wiki	0.001	0.048	0.004	0.015	0.584	**0.615**
Gujarati	UD lemma	gold	Sang.	0.078	0.392	0.098	0.157	0.294	**0.490**
Gujarati	UD lemma	gold	Wiki	0.118	0.353	0.137	0.216	0.412	**0.529**
Gujarati	UniMorph	corpus	Sang.	0.018	0.059	0.021	0.017	0.076	**0.223**
Gujarati	UniMorph	corpus	Wiki	0.008	0.104	0.008	0.006	0.056	**0.209**
Gujarati	UniMorph	gold	Sang.	0.196	0.410	0.221	0.236	0.234	**0.466**
Gujarati	UniMorph	gold	Wiki	0.213	0.493	0.250	0.227	0.268	**0.530**
Hindi	UD lemma	corpus	Sang.	0.012	0.278	0.018	0.032	0.311	**0.517**
Hindi	UD lemma	corpus	Wiki	0.009	0.339	0.013	0.030	0.307	**0.581**
Hindi	UD lemma	gold	Sang.	0.037	0.349	0.046	0.069	0.359	**0.496**
Hindi	UD lemma	gold	Wiki	0.050	0.423	0.063	0.094	0.376	**0.532**
Hindi	UniMorph	corpus	Sang.	0.020	0.094	0.027	0.033	0.257	**0.486**
Hindi	UniMorph	corpus	Wiki	0.013	0.099	0.016	0.023	0.300	**0.615**
Hindi	UniMorph	gold	Sang.	0.455	0.627	0.492	0.531	0.538	**0.680**
Hindi	UniMorph	gold	Wiki	0.440	0.636	0.460	0.509	0.552	**0.654**
Marathi	UD lemma	corpus	Sang.	0.003	0.032	0.004	0.002	**0.118**	0.096
Marathi	UD lemma	corpus	Wiki	0.011	0.067	0.018	0.015	**0.185**	0.174
Marathi	UD lemma	gold	Sang.	0.135	0.272	0.186	0.183	0.252	**0.362**
Marathi	UD lemma	gold	Wiki	0.233	0.314	0.258	0.261	0.330	**0.398**
Marathi	UD MWT	corpus	Sang.	0.005	0.048	0.008	0.025	**0.276**	0.119
Marathi	UD MWT	corpus	Wiki	0.004	0.035	0.006	0.018	**0.334**	0.149
Marathi	UD MWT	gold	Sang.	0.239	0.433	0.266	0.382	0.424	**0.534**
Marathi	UD MWT	gold	Wiki	0.251	0.430	0.307	0.397	0.439	**0.642**
Tamil	UD lemma	corpus	Sang.	0.008	0.031	0.009	0.018	0.058	**0.131**
Tamil	UD lemma	corpus	Wiki	0.006	0.032	0.012	0.023	0.069	**0.121**
Tamil	UD lemma	gold	Sang.	0.114	0.160	0.146	0.196	0.204	**0.284**
Tamil	UD lemma	gold	Wiki	0.101	0.186	0.150	0.208	0.233	**0.324**
Tamil	UD MWT	corpus	Sang.	0.027	0.150	0.031	0.037	0.071	**0.216**
Tamil	UD MWT	corpus	Wiki	0.010	**0.276**	0.022	0.054	0.057	0.168
Tamil	UD MWT	gold	Sang.	0.087	0.258	0.110	0.166	0.149	**0.319**
Tamil	UD MWT	gold	Wiki	0.143	**0.330**	0.168	0.188	0.202	0.329
Telugu	UniMorph	corpus	Sang.	0.018	0.022	0.018	**0.025**	0.020	**0.025**
Telugu	UniMorph	corpus	Wiki	0.040	0.067	0.067	0.054	0.054	**0.094**
Telugu	UniMorph	gold	Sang.	0.309	**0.422**	0.334	0.335	0.258	0.301
Telugu	UniMorph	gold	Wiki	0.494	0.414	0.488	**0.516**	0.375	0.318

F.5 Sensitivity to Noisy and Incomplete Suffix Lists

Proposition S.2 (Appendix A.3) bounds one intrinsic metric under a noisy oracle. However, it says nothing about language modelling or downstream accuracy, and its assumption on $\bar{\beta}$ does not hold for our lists (Table 15). We therefore degrade the suffix lists on purpose and retrain everything. *Drop* k% removes k% of the curated entries at random (raising $\bar{\beta}$), *add* k% adds spurious entries equal to k% of the list size, mined from frequent word-final character n-grams that are neither curated entries nor overlap one (raising $\bar{\alpha}$), and *swap* k% does both at constant list size, so that list quality is not confounded with list length. Each variant is a full re-run on Bengali (44 entries, λ=7.0) and Hindi (26 entries, λ=6.5) Wikipedia 10K, with a perturbed list, retrained tokenizer, GPT-2 8L trained from scratch, and NER with the retrained tokenizer (the other downstream datasets were not part of this sweep). The curated control is retrained in the same sweep, and its MRT-B BPC reproduces Table 1 exactly, while its MRT-P BPC differs slightly (1.8679 vs. 1.8648 for Bengali, 1.9971 vs. 1.9980 for Hindi).

Table 18: Suffix-list perturbations (Wikipedia 10K, GPT-2 8L). ΔBPC is the relative change against the retrained curated control (positive = worse).

	Bengali				**Hindi**			
	MRT-B		MRT-P		MRT-B		MRT-P	
List	ΔBPC	NER	ΔBPC	NER	ΔBPC	NER	ΔBPC	NER
curated	–	0.883	–	0.878	–	0.754	–	0.773
drop 10%	+0.8%	0.883	−0.2%	0.890	+0.2%	0.748	−0.2%	0.771
drop 25%	+0.0%	0.880	−0.0%	0.890	+0.4%	0.772	−0.1%	0.770
drop 50%	+0.2%	0.885	−0.2%	0.894	+0.1%	0.767	−0.2%	0.764
add 10%	−0.1%	0.891	+0.0%	0.889	+0.4%	0.759	+0.1%	0.783
add 25%	+0.0%	0.887	+0.2%	0.892	−0.2%	0.763	−0.1%	0.779
add 50%	+0.0%	0.890	+0.1%	0.875	+0.3%	0.770	+0.1%	0.782
swap 10%	+0.0%	0.891	+0.0%	0.890	+0.0%	0.762	−0.1%	0.771
swap 25%	+0.2%	0.878	−0.1%	0.881	+0.2%	0.758	+0.3%	0.776
swap 50%	+0.6%	0.890	−0.1%	0.879	+0.0%	0.747	−0.0%	0.778

Table 18 reports the results. Across all nine perturbations and both languages, MRT-P's BPC moves by at most 0.3% and its NER F1 stays within 1.6 points of the curated control in either direction, while MRT-B moves by at most 0.8%. The perturbations are not cosmetic, since dropping half of the list lowers the oracle's recall against gold from 0.356 to 0.133 (Bengali, UniMorph) and from 0.398 to 0.122 (Hindi, UD lemma). Two caveats apply. First, in this sweep the retrained curated MRT-P trails Unigram by 0.2-0.3% BPC on both languages, which are the two languages where Unigram is level with MRT-P in Table 1, so the ablation shows robustness to list noise rather than a margin over the baselines. Second, each variant is a single draw with seed 42. For reference, the retrained curated controls reach BPC 1.9145 (MRT-B) and 1.8679 (MRT-P) on Bengali and 2.0331 and 1.9971 on Hindi, and the baselines, which do not use the suffix list, reach Unigram BPC 1.8639 and 1.9916 and NER F1 0.888-0.896 and 0.753-0.763. On these two languages, the quality of the suffix list is not the binding constraint on the method.

F.6 Shared Multilingual Vocabulary

Multilingual models share one vocabulary across languages, which is where tokenizer capacity is most contested. We train one vocabulary per tokenizer family jointly over all nine languages at K=32,000 on the 81,000-line joint Wikipedia 10K training corpus (24.3M characters), and one GPT-2 8L per tokenizer on the same joint corpus (6 epochs), scored on each language's own held-out split. Joint BPC is the tokenizer-invariant metric, and joint per-token perplexity is shown for reference only. Sharing compresses the ≈272K pieces of the nine monolingual vocabularies into 32K, about 3.5K per language. MRT-B and MRT-P use the union of the nine suffix lists (160 entries), and no language identification is used anywhere. Because MRT-P's export caps the vocabulary at about half the requested size, we also train a *matched-K* variant that learns twice as many merges and exports a genuine 32K

vocabulary. The nine languages occupy six disjoint Unicode blocks, so the union list can interfere only within Devanagari (Hindi, Marathi, Nepali) and the Bengali script (Assamese, Bengali), where it changes the segmentation of 0-15.7% of validation word types.

Table 19: Shared 9-language vocabulary (K=32K, GPT-2 8L). Rend./suffix is SCon, and MAS, BVR and RecB are against each language's oracle, averaged over languages.

Tokenizer	$\|V\|$	**Joint BPC↓**	**PPL$_{\text{tok}}$**	**Fertility**	**Rend./suffix↓**	**MAS↑**	**BVR↓**	**RecB↑**
BPE	32,000	1.7938	129.8	1.61	37.41	0.394	0.606	0.238
Unigram	32,000	**1.7577**	112.4	1.63	36.66	0.426	0.570	0.390
WordPiece	28,340	1.7967	113.5	1.67	32.26	0.382	0.618	0.274
MRT-B	32,000	1.7716	80.5	1.77	**10.05**	**0.467**	**0.533**	0.789
MRT-P (λ=6.5)	17,471	1.7901	33.9	2.24	10.35	0.269	0.730	**0.799**
MRT-P (λ=6.5, matched K)	32,000	1.7668	50.4	1.98	14.16	0.341	0.658	0.779

Table 20: Shared vocabulary, per-language held-out BPC (↓) of the joint GPT-2 8L. Scored on each language's full held-out split.

Tokenizer	as	bn	gu	hi	mr	ne	pa	ta	te
BPE	1.9507	1.8786	1.7372	1.9533	2.0439	1.8858	2.0409	1.7806	0.7772
Unigram	**1.8945**	**1.8374**	**1.6995**	**1.9264**	**1.9926**	**1.8507**	**2.0076**	1.7461	0.7701
WordPiece	1.9541	1.8840	1.7395	1.9536	2.0482	1.8833	2.0405	1.7852	0.7820
MRT-B	1.9095	1.8520	1.7246	1.9363	2.0207	1.8621	2.0173	1.7611	**0.7674**
MRT-P (matched K)	1.8975	1.8416	1.7132	1.9428	2.0054	1.8780	2.0129	**1.7433**	0.7700

Table 19 summarises the joint models and Table 20 gives per-language BPC. Two findings carry over from the monolingual setting and one does not. Suffix consistency survives sharing. MRT-B and MRT-P render a suffix in 10-14 distinct ways against 32-37 for the subword baselines, and for the six suffix strings listed by more than one language, both MRT variants give the same modal rendering in every language that lists them (100% agreement, against 66.7% for the baselines). Boundary recall also survives, at 0.78-0.80 for the MRT variants against 0.24-0.39 for the baselines. MRT-B has the highest MAS, and MRT-P adds splits beyond the oracle boundaries, which costs it MAS and BVR, exactly as the $\lambda \to \infty$ limit predicts. The language-model advantage does not carry over. MRT-P at matched K is second in joint BPC, 0.5% above Unigram, and has the lower BPC than Unigram in only 2 of 9 languages. Sharing also shifts the optimal λ for several languages (for example, Bengali from 7.0 to 3.0 by BPC over the grid $\{3.0, 5.5, 6.5, 7.0\}$), so one λ must serve nine languages. The joint model sees nine times the text of a monolingual one and a different number of optimisation steps, so these numbers describe the practical cost of building one model for nine languages rather than a controlled ablation of vocabulary sharing, and they come from a single budget (K=32K) and a single seed.

F.7 Full Downstream Comparison

Table 21(a) reports downstream performance across all 9 languages on the Wikipedia 100K split, evaluated on five tasks, NER F1, classification accuracy and macro-F1, and sentence similarity MRR and Recall@10. Each cell shows the best score with the best-scoring tokenizer in parentheses. MRT-P or MRT-B achieves the highest score in 37 of 45 comparisons (82.2%), including all five metrics on Telugu and Gujarati, two of the languages with the highest suffix coverage (Appendix E.1). The 8 cases where baselines prevail occur on tasks where entity spans are morphologically simple (NER in low-resource languages) or where Unigram's broader vocabulary compensates for boundary noise (Hindi similarity). Together with the per-tokenizer Telugu breakdown in Table 22, these results show that the pattern generalises across languages. Per-tokenizer Sangraha 100K results are broken down by task in Appendix F.10.

Wikipedia, 10K. Table 21(b) reports downstream task performance at the 10K Wikipedia split for all 9 languages. MRT-P or MRT-B achieves the best score in the majority of cells across NER, classification, and similarity, with the strongest gains on Tamil, Marathi, Telugu and Bengali.

Sangraha, 100K. Table 21(c) reports Sangraha downstream results across all 9 languages. MRT-P or MRT-B achieves the best score in 32 of 45 comparisons (71.1%), including all five metrics on Tamil and Telugu. The pattern is consistent with Wikipedia, as MRT-P tends to lead on semantic tasks (similarity, classification) and NER for morphologically rich languages, while classification accuracy for languages with limited morphological coverage (Punjabi, Hindi) sometimes favours baselines.

Sangraha, 10K. Table 21(d) provides the Sangraha 10K downstream results across all 9 languages. At the 10K scale, MRT-P still achieves the best score in the majority of comparisons. The similarity task shows the most consistent MRT-P advantage (7/9 languages on MRR), while POS tagging shows the smallest differences, consistent with the observation that BiLSTM taggers compensate for boundary noise via context.

Table 21: Downstream performance as the best score per cell with the best-scoring tokenizer, on both corpora at the 100K and 10K splits.

Language	NER F1↑	CLF Acc↑	CLF F1↑	Sim MRR↑	Sim R@10↑
(a) Wikipedia, 100K					
Assamese	0.417 (MRT-P)	0.862 (BPE)	0.784 (MRT-P)	0.097 (MRT-P)	0.164 (MRT-P)
Bengali	0.893 (MRT-P)	0.898 (MRT-P)	0.698 (BPE)	0.259 (MRT-P)	0.418 (MRT-P)
Gujarati	0.030 (MRT-P)	0.898 (MRT-P)	0.732 (MRT-P)	0.193 (MRT-P)	0.334 (MRT-P)
Hindi	0.779 (MRT-P)	0.941 (MRT-P)	0.840 (MRT-P)	0.286 (Uni)	0.460 (Uni)
Marathi	0.735 (MRT-P)	0.890 (MRT-P)	0.769 (BPE)	0.026 (MRT-P)	0.043 (MRT-P)
Nepali	0.237 (WP)	0.889 (MRT-P)	0.626 (MRT-P)	0.227 (MRT-P)	0.405 (MRT-P)
Punjabi	0.244 (MRT-B)	0.896 (Morf)	0.770 (MRT-P)	0.143 (MRT-P)	0.236 (MRT-P)
Tamil	0.757 (MRT-P)	0.898 (MRT-P)	0.815 (Morf)	0.020 (MRT-P)	0.036 (MRT-P)
Telugu	0.495 (MRT-P)	0.909 (MRT-P)	0.788 (MRT-P)	0.109 (MRT-P)	0.203 (MRT-P)
(b) Wikipedia, 10K					
Assamese	0.364 (Morf)	0.858 (Morf)	0.794 (Morf)	0.098 (MRT-P)	0.170 (MRT-P)
Bengali	**0.902** (MRT-P)	0.878 (BPE)	0.679 (MRT-P)	0.259 (MRT-P)	0.433 (MRT-P)
Gujarati	0.022 (Morf)	**0.898** (MRT-P)	0.716 (WP)	0.188 (MRT-P)	0.312 (MRT-P)
Hindi	0.789 (MRT-P)	0.935 (Uni)	0.834 (MRT-P)	0.283 (Morf)	0.455 (Morf)
Marathi	0.728 (MRT-P)	**0.897** (MRT-P)	0.777 (MRT-P)	0.023 (MRT-P)	0.039 (MRT-P)
Nepali	0.220 (Morf)	0.891 (Morf)	0.632 (MRT-P)	0.218 (MRT-P)	0.406 (MRT-P)
Punjabi	0.262 (Morf)	0.900 (Uni)	0.783 (MRT-B)	0.142 (Uni)	0.228 (Uni)
Tamil	0.764 (MRT-P)	**0.904** (MRT-P)	**0.848** (MRT-P)	0.020 (MRT-P)	0.035 (MRT-P)
Telugu	0.516 (MRT-P)	0.902 (Uni)	0.800 (Uni)	0.117 (MRT-P)	0.205 (MRT-B)
(c) Sangraha, 100K					
Assamese	0.329 (MRT-P)	**0.880** (MRT-P)	0.817 (WP)	0.080 (MRT-P)	0.137 (MRT-P)
Bengali	0.888 (BPE)	0.883 (Uni)	0.692 (WP)	0.252 (MRT-P)	0.415 (MRT-P)
Gujarati	0.020 (Morf)	0.896 (MRT-B)	**0.713** (MRT-P)	0.197 (MRT-P)	0.338 (MRT-P)
Hindi	0.776 (BPE)	0.938 (MRT-P)	0.834 (MRT-P)	0.290 (Uni)	0.460 (Uni)
Marathi	0.720 (MRT-P)	0.884 (MRT-P)	0.786 (BPE)	0.021 (MRT-P)	0.039 (MRT-P)
Nepali	0.023 (Morf)	0.891 (MRT-B)	0.621 (MRT-B)	0.226 (MRT-P)	0.404 (MRT-P)
Punjabi	0.370 (WP)	0.899 (BPE)	0.770 (WP)	0.136 (MRT-P)	0.224 (MRT-P)
Tamil	**0.760** (MRT-P)	0.903 (MRT-P)	0.834 (MRT-P)	0.019 (MRT-P)	0.029 (MRT-P)
Telugu	0.524 (MRT-P)	0.898 (MRT-P)	**0.796** (MRT-P)	0.111 (MRT-P)	0.190 (MRT-P)
(d) Sangraha, 10K					
Assamese	0.325 (WP)	0.887 (Morf)	0.827 (Morf)	0.083 (MRT-P)	0.144 (MRT-P)
Bengali	**0.894** (MRT-P)	0.886 (Morf)	0.719 (Uni)	0.270 (MRT-P)	0.431 (MRT-P)
Gujarati	0.057 (Morf)	**0.893** (MRT-B)	0.743 (Uni)	0.192 (MRT-P)	0.324 (MRT-P)
Hindi	0.773 (WP)	0.941 (MRT-P)	0.848 (MRT-P)	0.284 (WP)	0.454 (Morf)
Marathi	**0.723** (MRT-P)	0.886 (MRT-P)	0.774 (BPE)	0.021 (MRT-P)	0.036 (MRT-P)
Nepali	0.213 (Morf)	0.897 (Uni)	0.622 (BPE)	0.227 (MRT-P)	0.397 (MRT-P)
Punjabi	0.314 (Uni)	0.900 (Uni)	0.780 (MRT-B)	0.141 (Uni)	0.221 (MRT-P)
Tamil	**0.770** (MRT-P)	**0.908** (MRT-P)	**0.833** (WP)	0.017 (MRT-P)	0.027 (MRT-P)
Telugu	**0.498** (MRT-B)	**0.902** (MRT-P)	**0.817** (MRT-B)	0.116 (MRT-P)	0.189 (MRT-P)

Table 22: Telugu (Wikipedia, 100K) downstream metrics per tokenizer.

Task	BPE	Uni	WP	Morf	MRT-B	MRT-P
NER F1↑	0.462	0.463	0.450	0.466	0.477	**0.495**
CLF Acc↑	0.856	0.888	0.884	0.887	0.897	**0.909**
CLF F1↑	0.754	0.755	0.749	0.732	0.761	**0.788**
Sim MRR↑	0.095	0.100	0.101	0.096	0.108	**0.109**
Sim R@10↑	0.161	0.165	0.172	0.155	0.178	**0.203**

F.8 Full Lambda Sweep

Table 23 reports language-model perplexity at a representative subset of the 18 swept λ values for Bengali Wikipedia (10K split), supplementing Figure 3 in the main text with exact numbers. PPL drops steeply from λ=0 (BPE, 1131.7) to λ=1, plateaus through the moderate range, and is minimised at λ=7.0 (194.7). At $\lambda \to \infty$ (MRT-B) PPL rises again to 432.2, illustrating the over-fragmentation cost of the hard constraint. The MRT-P rows come from the dedicated sweep run, whereas the separately trained per-language model at λ=7.0 in Table 24(a) reaches 195.7.

Table 23: Bengali per-token LM perplexity across λ (Wikipedia 10K, GPT-2 2L, K=32K). The ∞ row is MRT-B.

λ	PPL ↓
0.0 (BPE)	1131.7
0.50	335.3
1.00	243.6
1.50	210.2
2.00	212.9
3.00	203.2
5.00	198.0
7.00	**194.7**
9.00	196.1
∞ (MRT-B)	432.2

The minimum is at λ^*=7.0 with PPL 194.7, and the curve is flat (within ±2 PPL) for $\lambda \in [5, 9]$. The corresponding standard deviation $\sigma_\lambda = \text{std}(\text{PPL}(0.5), \ldots, \text{PPL}(9.0)) = 44.5$ over the eight λ values $\{0.5, 1.0, 1.5, 2.0, 3.0, 5.0, 7.0, 9.0\}$ in Table 23 is dominated by the steep λ<1.5 region, and restricting to $\lambda \in [1.5, 9.0]$ yields $\sigma_\lambda = 7.0$ (population standard deviation in both cases), indicating the optimum is broad and robust. At every λ in the sweep the exported vocabulary is about 16K (16,020-16,787), below the K=32,000 target, because MRT-P's SentencePiece export caps the vocabulary at 55% of the distinct pieces (Table 11), so the λ comparison is therefore at near-constant vocabulary size.

F.9 Per-Token Perplexity of Small-Scale Language Models

Table 24(a) and (b) report held-out per-token perplexity of the GPT-2 2L configuration for all 9 languages on Wikipedia and Sangraha (10K split, 15 epochs), the lowest rung of our scale ladder. MRT-P has the lowest per-token perplexity in every language on both corpora. Per-token perplexity divides by the tokenizer's own token count and MRT-P is the most fragmenting tokenizer, so these tables show that the ordering is stable across languages, corpora and split sizes, not the size of a tokenizer-independent gain. The tokenizer-invariant comparison is in Table 1 and Appendix F.1. Excluding Telugu Wikipedia (see the table note), the absolute reduction over the strongest baseline ranges from 66.6 (Punjabi, Sangraha) to 885.2 (Telugu, Sangraha).

Per-token perplexity by split size. To check that the per-token ordering of the GPT-2 2L configuration is not an artifact of a particular data scale, Table 24(c) reports Bengali at the 100-sentence (extreme low-resource) and 100K Wikipedia splits and at the 100K

Sangraha split, since Bengali is the only language run at these sizes. MRT-P achieves the lowest per-token PPL in all three settings, 63% below the strongest baseline (Unigram) on Wikipedia 100K and 71% below it on Sangraha 100K, so the ordering is stable across data regimes and corpora. As elsewhere, per-token perplexity is not a tokenizer-invariant measure of the size of the gain.

Table 24: Per-token LM perplexity of GPT-2 2L. Δ is best baseline PPL − best MRT PPL. †Unfiltered held-out split.

Language	BPE	Uni	WP	Morf	MRT-B	**MRT-P**	Δ
(a) Wikipedia, 10K							
Assamese	1191.3	707.7	980.8	742.2	397.5	**206.9** (λ=6.5)	+500.9
Bengali	1131.7	393.0	964.4	686.1	432.2	**195.7** (λ=7.0)	+197.4
Gujarati	349.8	175.7	299.3	237.7	151.6	**68.7** (λ=6.0)	+107.0
Hindi	479.1	207.2	437.6	377.7	304.7	**140.3** (λ=6.5)	+66.8
Marathi	1661.1	836.9	1323.5	850.3	788.8	**296.3** (λ=7.0)	+540.5
Nepali	784.7	391.2	664.4	466.8	389.5	**153.8** (λ=6.0)	+237.4
Punjabi	534.9	245.4	477.1	418.9	368.2	**169.9** (λ=5.5)	+75.5
Tamil	2628.6	1079.0	1945.2	857.4	1408.5	**389.8** (λ=6.0)	+467.6
Telugu†	19.3	12.1	18.2	16.7	12.4	**6.3** (λ=6.0)	+5.9
(b) Sangraha, 10K							
Assamese	2112.4	1230.6	1780.7	1237.4	689.8	**349.8** (λ=5.5)	+880.8
Bengali	2134.4	1220.8	1799.7	1243.1	707.2	**351.1** (λ=7.0)	+869.7
Gujarati	1504.9	818.0	1285.6	887.6	644.8	**299.7** (λ=6.0)	+518.3
Hindi	424.9	200.0	389.9	340.7	278.4	**126.6** (λ=7.0)	+73.3
Marathi	1604.4	853.1	1331.7	972.2	710.3	**265.8** (λ=7.0)	+587.3
Nepali	1114.4	314.8	921.7	512.8	428.8	**168.0** (λ=6.0)	+146.8
Punjabi	495.3	222.3	444.1	394.4	333.2	**155.7** (λ=6.5)	+66.6
Tamil	3099.5	1265.8	2409.3	957.6	1639.3	**463.8** (λ=7.0)	+493.8
Telugu	2282.7	1105.5	1878.1	1567.1	890.1	**220.4** (λ=5.5)	+885.2
(c) Bengali at further split sizes							
Bengali (Wikipedia, 100)	559.4	418.1	441.3	480.9	449.5	**180.7** (λ=6.0)	+237.4
Bengali (Wikipedia, 100K)	279.8	220.2	263.8	273.6	126.6	**80.7** (λ=7.0)	+139.5
Bengali (Sangraha, 100K)	554.5	461.6	511.9	531.1	227.0	**131.8** (λ=7.0)	+329.8

F.10 Per-Task Detailed Results (Sangraha, 100K)

Table 25 breaks down the Sangraha 100K downstream results by task, showing all 6 tokenizers for each available language. These complement the summary in Table 21(c) by exposing the full competitive landscape.

POS Tagging. In Table 25(a), MRT-P achieves the highest accuracy on 3 of 6 languages (Bengali, Tamil, Telugu) and MRT-B on Hindi, with the largest improvement on Telugu (+2.2 percentage points over Unigram, the strongest baseline). Gujarati yields low accuracy across *all* tokenizers, likely due to the small UD treebank size (503 sentences).

Named Entity Recognition. Table 25(b) reports per-tokenizer NER F1 on Sangraha (100K). MRT-P achieves the highest NER F1 on 4 of 9 languages (Assamese, Marathi, Tamil, Telugu), and is 0.4 points below BPE on Bengali. On Telugu (+5.9 percentage points) and Tamil (+1.7 points) the improvements over the strongest baseline are noteworthy. Punjabi represents the largest negative result (−9.6 points below WordPiece), attributable to limited NER training data and short entity spans in that language.

Sentence Similarity. In Table 25(c), MRT-P achieves the highest MRR on 8 of 9 languages (all except Hindi, and on Punjabi by 0.001 over Unigram). The largest improvements appear for Bengali (+5.0 points MRR), Gujarati (+4.6 points), and Nepali (+4.5 points). Hindi is the only language where Unigram's broader vocabulary yields higher similarity retrieval, plausibly due to Hindi's relatively simple suffixation and strong baseline Unigram performance.

Table 25: Per-tokenizer downstream results on Sangraha (100K). "—" means the NER model predicted only O tags.

Language	BPE	Uni	WP	Morf	MRT-B	**MRT-P**
(a) POS tagging accuracy						
Bengali	0.605	0.616	0.610	0.616	0.620	**0.648** (λ=7.0)
Gujarati	0.470	0.487	**0.503**	0.487	0.477	0.490 (λ=6.0)
Hindi	0.954	0.954	0.953	0.953	**0.955**	0.954 (λ=7.0)
Marathi	0.724	0.730	0.737	**0.738**	0.725	0.729 (λ=7.0)
Tamil	0.626	0.611	0.620	0.616	0.632	**0.639** (λ=7.0)
Telugu	0.830	0.833	0.825	0.819	0.842	**0.855** (λ=5.5)
(b) NER F1						
Assamese	0.257	0.279	0.308	0.252	0.239	**0.329** (λ=7.0)
Bengali	**0.888**	0.888	0.888	0.884	0.884	0.884 (λ=7.0)
Gujarati	—	—	—	**0.020**	—	—
Hindi	**0.776**	0.770	0.768	0.771	0.761	0.774 (λ=7.0)
Marathi	0.702	0.707	0.701	0.702	0.714	**0.720** (λ=7.0)
Nepali	—	—	—	**0.023**	0.011	—
Punjabi	0.155	0.205	**0.370**	0.259	0.178	0.274 (λ=7.0)
Tamil	0.734	0.737	0.743	0.734	0.746	**0.760** (λ=7.0)
Telugu	0.449	0.465	0.450	0.462	0.482	**0.524** (λ=5.5)
(c) Sentence-similarity MRR						
Assamese	0.046	0.055	0.048	0.046	0.063	**0.080** (λ=7.0)
Bengali	0.174	0.202	0.160	0.158	0.219	**0.252** (λ=7.0)
Gujarati	0.125	0.151	0.131	0.129	0.161	**0.197** (λ=6.0)
Hindi	0.278	**0.290**	0.279	0.286	0.286	0.280 (λ=7.0)
Marathi	0.018	0.018	0.018	0.017	0.017	**0.021** (λ=7.0)
Nepali	0.147	0.181	0.135	0.139	0.178	**0.226** (λ=6.0)
Punjabi	0.121	0.135	0.109	0.119	0.133	**0.136** (λ=7.0)
Tamil	0.012	0.014	0.013	0.014	0.012	**0.019** (λ=7.0)
Telugu	0.087	0.081	0.086	0.081	0.101	**0.111** (λ=5.5)

F.11 Fidelity vs. Word Length

A central concern with morphological regularisation is that forcing splits at morpheme boundaries may excessively fragment long words, inflating sequence length and thus transformer FLOPs. To quantify this trade-off, we measure *fertility* (tokens per word) as a function of word length in characters for all six tokenizers on Bengali Wikipedia (10K split, vocab 32K).

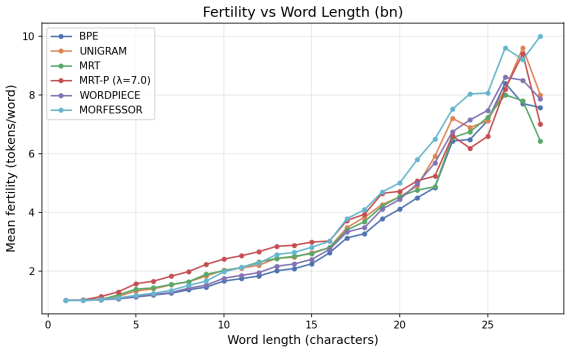


Figure 5: Fertility vs. word length (Bengali, Wikipedia 10K, vocabulary 32K).

Figure 5 reveals three regimes. For **short words (1-10 chars)**, all tokenizers produce ∼1-2 tokens (mono-morphemic, no boundaries to violate). For **mid-length words (10-20 chars)**, MRT-P adds 0.5-1.0 tokens relative to BPE, localised precisely at suffix boundaries.

For **long words (> 25 chars)**, all tokenizers converge to high fertility (> 6 tokens). Overall, on Bengali MRT-P renders suffixes in 4.1× fewer distinct forms than BPE (Section 4.1) at a fertility of 1.62 against 1.23 tokens per word (+32%, Table 11). Because per-token perplexity rewards fragmentation, it cannot tell whether these extra splits pay for themselves. Bits per character can. In Table 1, MRT-P's held-out loss per character is lower than every baseline's in 7 of 9 languages despite its higher fertility, so the added tokens are not pure overhead.

G Qualitative Analysis

This section provides concrete examples of how different tokenizers segment individual words, quantifies vocabulary-level differences between methods, and measures the predictive validity of intrinsic metrics for downstream performance. Together, these analyses give direct insight into *why* MRT works, not just *that* it works.

G.1 Tokenization Examples

Table 26 contrasts how each method segments four representative Bengali words, exposing where boundary-aware merges differ from purely frequency-driven ones.

Table 26: Tokenization of example Bengali words under each method (Wikipedia 10K split, MRT-P at λ=7.0). | denotes token boundary.

Word	Gloss	BPE	Unigram	MRT-B	MRT-P
বইগুলোতে	in the books	বই|গুলোতে	বইগুলো|তে	বই|গুলো|ত|ে	বই|গুলো|তে
করতেছিলাম	I was doing	করতে|ছিল|াম	করতে|ছিল|াম	করত|েছ|িলা|ম	করত|েছিল|াম
আন্তর্জাতিক	international	আন্তর্জাতিক	আন্তর্জাতিক	আন্তর্জাতিক	আন্তর্জা|তিক
বিশ্ববিদ্যালয়ে	in the university	বিশ্ববিদ্যালয়ে	বিশ্ববিদ্যালয়ে	বিশ্ববিদ্যালয়ে	বিশ্ববিদ্যালয়|ে

Observed pattern. For বইগুলোতে, BPE absorbs the case marker "ে" into "গুলোতে" and Unigram drops the boundary between root and plural marker, and only MRT-P recovers the canonical 3-morpheme split বই|গুলো|তে. MRT-B trained at the 10K split over-fragments, splitting the case marker into ত|ে, because the segmented training corpus lacks the merge তে as a frequent atomic token. MRT-P's normalised penalty avoids this brittleness by allowing high-frequency within-suffix merges through the $\max(0, 1 - \lambda r)$ clamp. For low-frequency words (আন্তর্জাতিক, বিশ্ববিদ্যালয়ে) BPE/Unigram/MRT-B all keep the surface form whole, whereas MRT-P still introduces a split because the case marker "ে" is a high-crossing-rate edge. Section 5 discusses analogous Gujarati cases, and Section 5 quantifies these error types on Telugu.

G.2 Vocabulary Overlap and Coverage Analysis

Beyond individual word segmentations, we ask how different the *vocabularies* produced by MRT and standard BPE are. A low overlap would indicate that MRT fundamentally restructures the token inventory rather than making marginal adjustments. We also measure *morpheme coverage*, what fraction of gold morphemes appear as whole tokens in the vocabulary, and the cross-boundary token rate to quantify vocabulary-level alignment.

Let V_{BPE} and V_{MRT} denote the vocabularies of BPE and MRT-P respectively, each of size K. We define

$$\text{Overlap}(V_1, V_2) = \frac{|V_1 \cap V_2|}{K}, \qquad \text{MorphCoverage}(V) = \frac{|\{m \in M : m \in V\}|}{|M|}$$

where $M = \bigcup_{w \in \mathcal{W}} \mu(w)$ is the set of all distinct morphemes observed in the corpus. Table 27 reports these vocabulary-level statistics for BPE, Unigram, MRT-B, and MRT-P on the Bengali Wikipedia 100K split.

Table 27: Vocabulary statistics (Bengali Wikipedia 100K, target K=32,000, MRT-P at λ=7.0).

Metric	BPE	Unigram	MRT-B	MRT-P
Effective vocabulary size $\|V\|$	29,730	29,830	29,299	15,196
Overlap with BPE	1.00	0.62	0.68	0.36
Morpheme coverage	0.079	0.105	**0.108**	0.056
Mean token length (chars)	5.91	6.42	5.70	5.14
Tokens that are full morphemes	0.564	0.749	0.784	**0.788**
Tokens that cross boundaries	0.082	0.034	**0.005**	**0.005**

MRT-B and MRT-P both reduce the cross-boundary token rate to 0.5%, an order of magnitude below BPE (8.2%). MRT-P attains the highest fraction of full-morpheme tokens (0.788) despite having half the effective vocabulary (15,196 pieces, because its SentencePiece export caps the vocabulary at 55% of the distinct pieces), because the normalised penalty front-loads merges onto morpheme-aligned substrings, which the capped export retains. Its lower absolute morpheme-coverage figure (0.056) is a function of the smaller $|V|$, not of inferior alignment. Per token, 78.8% of MRT-P's vocabulary is a valid morpheme, against 56.4% for BPE. The 0.36 overlap with BPE quantifies how much vocabulary mass MRT-P reallocates away from BPE-style boundary-crossing merges.

G.3 Statistical Significance and Downstream Performance

A key scientific question for the community is whether cheap-to-compute intrinsic metrics can serve as reliable proxies for expensive downstream evaluation. If so, practitioners could compare tokenizers without training downstream models. We compute Spearman rank correlations (ρ) between the token-level alignment metrics (MAS and BVR) and downstream task scores, pooled over all (tokenizer, language, split) combinations on Wikipedia (Table 28). The number of paired observations varies per task because not all languages have annotations for all tasks (e.g., POS requires UD treebanks). The sign convention is set so that a more negative ρ means the intrinsic metric is a better predictor, and SC and SCon are not included in this pooled analysis.

Table 28: Spearman ρ between intrinsic metrics and downstream scores, pooled over (tokenizer, language, split) on Wikipedia, with the number of pairs in parentheses.

Intrinsic metric	LM PPL ↓	POS Acc ↑	NER F1 ↑	CLF F1 ↑	SIM MRR ↑
MAS ↑	−0.28 (66)	+0.02 (132)	−0.08 (169)	−0.04 (195)	+0.23 (177)
BVR ↓	+0.29 (66)	−0.02 (132)	+0.09 (169)	+0.05 (195)	−0.23 (177)
SCon ↓	n/a	n/a	n/a	n/a	n/a

The strongest signals are between MAS/BVR and the two information-bearing tasks, *LM PPL* ($|\rho| \approx 0.29$) and *sentence similarity MRR* ($|\rho| \approx 0.23$), with the expected directionsbetter alignment correlates with lower perplexity and higher MRR. Token-classification tasks (POS, NER) show essentially zero correlation, consistent with the explanation that BiLSTM taggers can compensate for boundary noise via context once the word is presented in pieces. These results support BVR/MAS as cheap proxies for LM and similarity quality, although *not* as a substitute for running tagger-based downstream evaluations.

Statistical testing and its scope. All runs use a single seed (42), so we have no estimate of variance over random initialisation. To check that differences are not artefacts of one particular data condition, we use bootstrap confidence intervals (B = 10,000 resamples) and Wilcoxon signed-rank tests that treat the corpus-size splits as paired observations, with significance thresholds at $p < 0.05$, $p < 0.01$ and $p < 0.001$. These quantify variation across data conditions, not across initialisations, and should not be read as seed variance. Where a margin is small relative to this variation, as for Bengali, Hindi, Punjabi and filtered Telugu Wikipedia in Table 1, we describe it as a tie rather than an improvement.

H Limitations

MRT targets predominantly suffixing languages and relies on a short curated suffix list for each language, so its benefits depend in part on the quality of that list, and extending it to other morphological types would require adapting the oracle. Our language models are trained from scratch at moderate scale so that the tokenizer is the only variable, and how the gains carry over to larger models and to the pretrain-then-finetune setting remains to be studied. As in any cross-tokenizer comparison, tokenizers differ in their vocabularies and output spaces, so even tokenizer-invariant metrics cannot make the runs fully identical, and small margins are best read as ties. Some intrinsic metrics are defined relative to our oracle, which we complement with external annotation where it is available, and our experiments use a single training seed. Addressing these points is a natural direction for future work.

I Broader Impacts

Positive societal impacts: MRT directly benefits speakers of low-resource Indic languages, collectively over 1.5 billion people by improving the quality of NLP tools (language models, POS taggers, NER systems, search engines) built on top of subword tokenizers. By requiring only compact, manually curated suffix lists (11-44 entries) rather than expensive annotated corpora or trained morphological analysers, MRT lowers the barrier to building linguistically informed NLP pipelines for under-served languages. The framework is language-agnostic in design, so any agglutinative or suffixing language can benefit by supplying a short suffix list, enabling community-driven extension without access to large-scale compute or linguistic expertise beyond native-speaker knowledge. Improved tokenization quality also reduces the data requirements for acceptable model performance, supporting more equitable access to language technology in regions with limited digital text resources.

Negative societal impacts: As a tokenization method, MRT operates at the preprocessing layer and does not directly generate text or make decisions about individuals. However, improved language modelling for low-resource languages could lower the cost of generating fluent disinformation or spam in those languages, where automated detection tools are less mature. Additionally, the curated suffix lists encode linguistic norms from standard grammars, which may not fully represent dialectal variation or non-standard usage, and deploying MRT-based systems could inadvertently reinforce linguistic standardisation at the expense of dialectal diversity. We note that these risks are shared broadly across all NLP improvements for low-resource languages and are not unique to MRT. We encourage practitioners deploying MRT-based systems to consider dialect coverage when curating suffix lists and to pair improved generation capabilities with appropriate content moderation.

J Statements

AI use statement. Generative AI tools were used in two limited roles. For the manuscript, they helped edit and polish the text and cross-check the reported numbers against the released configurations and experiment outputs. For the released code, they assisted with auxiliary components such as plot generation and formatting, and every AI-assisted piece of code was reviewed, tested and verified by the authors before use. No generative AI tool was used to develop research ideas, to design the method, or to produce experimental results, and every reported number comes from actually training and evaluating the models described in this paper.

Ethics statement. This work proposes a tokenization method and involves no human-subject research. All corpora and task datasets, namely Wikipedia, Sangraha-Verified, WikiANN, XL-Sum, the Universal Dependencies treebanks and UniMorph, are existing public resources used under their original licenses and cited in Appendix D.2, and we collected no new annotations. The only new linguistic resource is the set of short curated suffix lists, compiled from standard grammars. The potential benefits and risks of better tokenization for low-resource Indic languages are discussed in Appendix I.