
How LLMs Process Non-English Languages

A Deep Dive into Tokenization, Embeddings, and the Multilingual Deficit

Basanth Periyapatna Roopa Kumar

Applied Data Science | SJSU

Abstract: Large Language Models systematically disadvantage non-English languages at every level of their architecture. A Kannada sentence requires **3–5× more tokens** than its English equivalent, costs proportionally more to process, and produces lower-quality outputs. This document traces the cascade from byte-level tokenization through attention mechanisms to final generation, comparing how GPT-4o, Gemini, Claude, Llama, and specialized Indic models handle the world's most complex scripts. We examine the training data imbalance (**95% English** in Llama 3), the token fertility crisis, the curse of multilinguality in embeddings, and the frontier research closing these gaps.

1. Byte-Level Tokenization Creates a Fundamental Tax on Non-Latin Scripts

The journey of every prompt through an LLM begins at the tokenizer, and it is here that the inequality starts.

Modern LLMs including GPT-4, Claude, Llama, and Mistral use **byte-level Byte Pair Encoding (BPE)**, which operates on raw UTF-8 bytes rather than characters. The algorithm starts with a base vocabulary of 256 byte tokens, then iteratively merges the most frequent adjacent byte pairs from training data. This design guarantees universal coverage (no unknown tokens) but creates deep structural inequity across scripts.

The core problem is **byte width**. Every Latin character occupies **1 byte** in UTF-8. Each Kannada, Hindi, Tamil, or Chinese character requires **3 bytes**. Arabic and Urdu characters consume **2 bytes**. This means the BPE algorithm needs more merges just to reconstitute a single non-Latin character from its byte fragments. Because English dominates training corpora, English byte-pair merges are learned first and most frequently. Non-Latin byte sequences recur with lower probability, so many never get merged into complete characters in smaller vocabularies, leaving individual bytes as tokens.

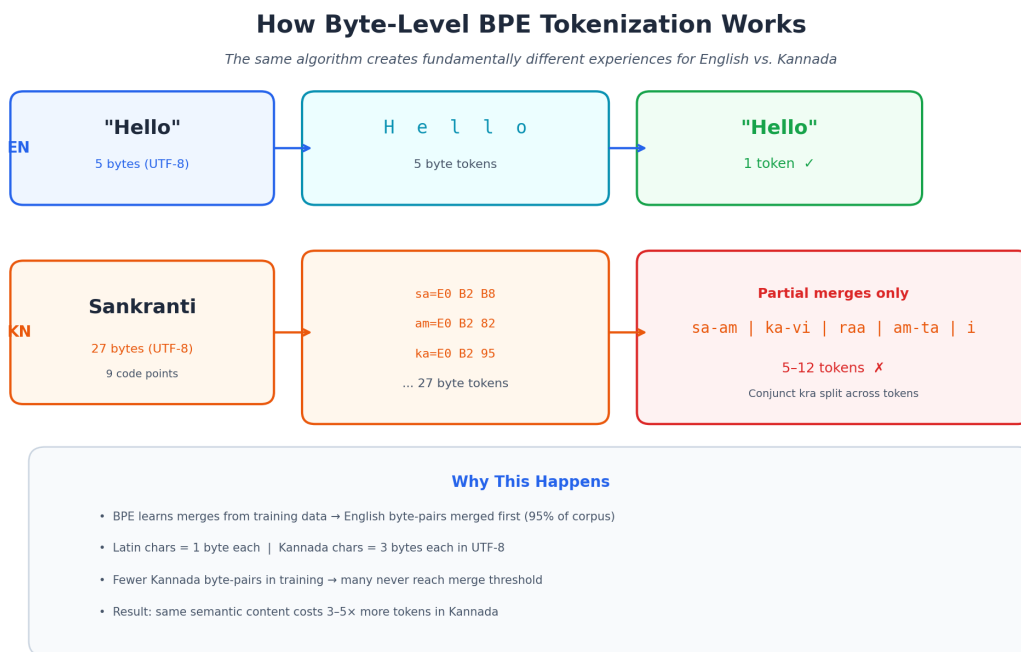


Figure 1: The BPE tokenization pipeline showing how identical semantic content diverges dramatically between English and Kannada at the byte level.

Consider how the Kannada word **Sankranti** is encoded. It contains 9 Unicode code points totaling **27 bytes** in UTF-8. The conjunct consonant **kra**, which appears as a single visual glyph (an ottakshara), is actually three separate code points spanning 9 bytes. A byte-level BPE tokenizer fragments this linguistically coherent unit across multiple tokens, destroying its semantic integrity.

SentencePiece (used by T5, mBART, and XLM-R) takes a different approach. Its unigram language model starts with a large vocabulary and iteratively removes tokens that least decrease training data likelihood. This probabilistic, top-down method can be more linguistically aware for complex scripts, and it handles text without space-based pre-tokenization — critical for languages where word boundaries are ambiguous. **WordPiece** (used by BERT)

merges pairs that maximize training data likelihood rather than simple frequency, making it reasonably effective for Chinese since individual characters are meaning-bearing units.

However, research from Digital Orientalist (2025) found BPE creates **linguistically incorrect merges** in Chinese: Llama incorrectly merges certain character pairs because those byte pairs were merged earlier during training.

2. The Fertility Crisis: Same Content, 3–15× the Tokens

Token fertility is the single most measurable manifestation of multilingual inequality in LLMs.

Token fertility — the average number of tokens per word — was rigorously quantified by Petrov et al. (NeurIPS 2023) in their landmark paper "Language Model Tokenizers Introduce Unfairness Between Languages," which evaluated 17 tokenizers across dozens of languages and found tokenization length differences of **up to 15x** for equivalent text.

Token Fertility: The Hidden Tax on Non-English Languages

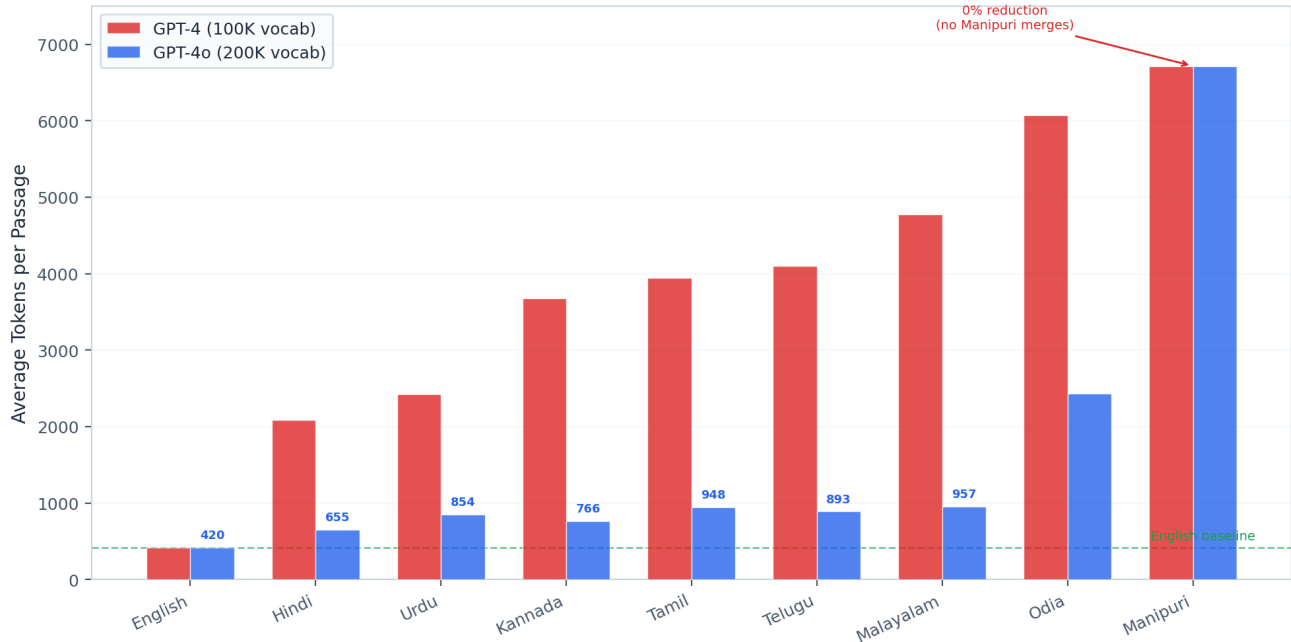


Figure 2: Token fertility comparison across Indic languages. GPT-4o's 200K vocabulary dramatically reduces tokens for most languages, but Manipuri sees zero improvement — its byte sequences were never learned.

Token Count Comparison: GPT-4 vs. GPT-4o on Indic Languages

Language	GPT-4 Tokens	GPT-4o Tokens	Reduction	Fertility vs English
Malayalam	4,775	957	79%	~4–5x
Kannada	3,681	766	79%	~3–4x
Telugu	4,097	893	77%	~3–4x
Tamil	3,949	948	74%	~3–4x
Hindi	2,090	655	64%	~2–3x
Urdu	2,428	854	62%	~2–3x
Odia	6,074	2,432	60%	~5–6x
Manipuri	6,715	6,715	0%	~8–10x

The economic consequences are severe. The paper "Do All Languages Cost the Same?" (Ahia et al., EMNLP 2023) found that non-English speakers are systematically overcharged while obtaining worse results. A company serving

Hindi users pays an estimated **\$262,000–\$365,000 more annually** than an identical English service at scale. There is a **-0.5 correlation** between a country's Human Development Index and LLM tokenization cost — the poorest countries often speak the most expensive-to-process languages.

A 128K-token context window effectively shrinks to 25,000–42,000 tokens of usable content for Hindi or Arabic text, severely limiting RAG applications and long-document processing.

128K Context Window: What You Actually Get Per Language

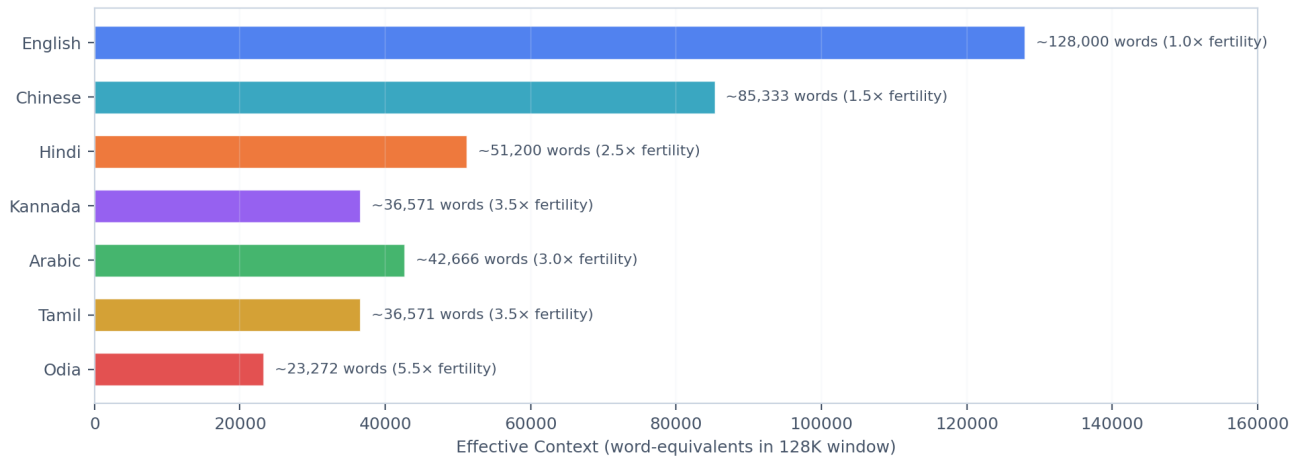


Figure 3: Effective context window per language. The same 128K token limit yields vastly different amounts of usable content depending on script complexity.

3. How Multilingual Embeddings Bridge Scripts — Imperfectly

When two words mean the same thing in different scripts, how does a model know they're related?

Multilingual embedding models create shared vector spaces where semantically similar text in different scripts clusters together. The mechanisms enabling this are deceptively simple: **shared vocabulary** and **shared parameters** force models to learn partially language-neutral representations. When Hindi and English "computer" appear in similar contexts across training data, the model converges on similar internal representations despite entirely different scripts.

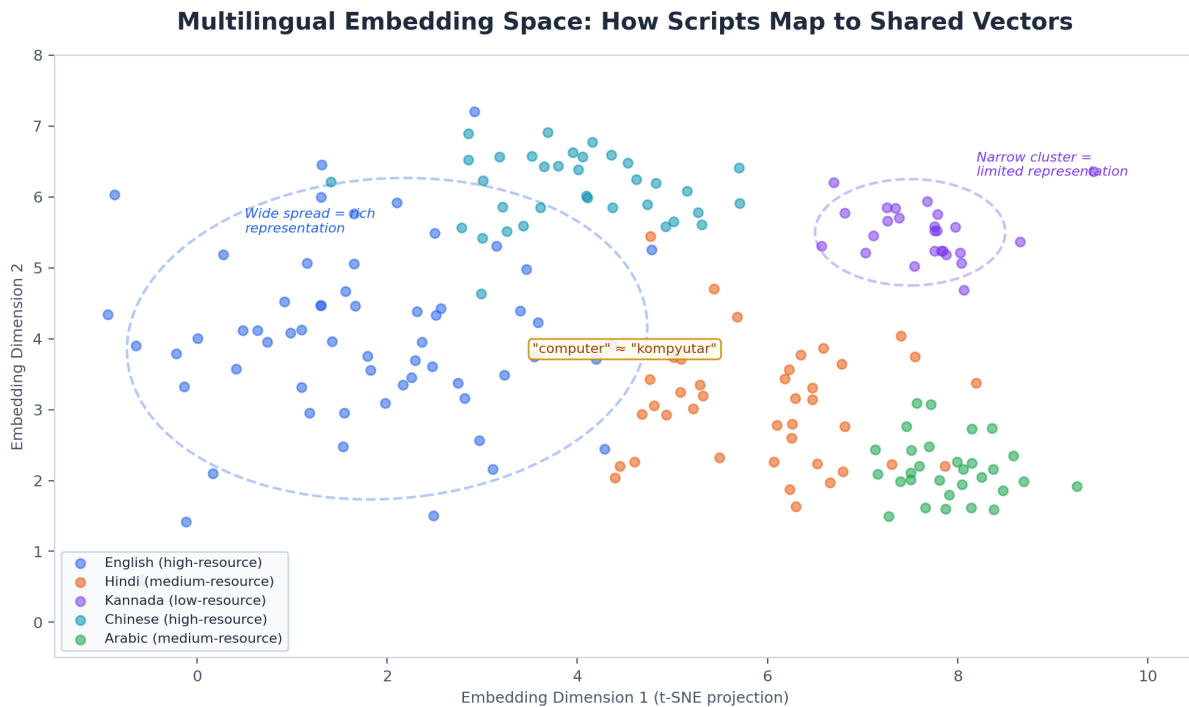


Figure 4: Visualization of multilingual embedding space. High-resource languages like English spread across a wide semantic region, while low-resource languages like Kannada cluster narrowly — indicating less expressive representations.

XLm-RoBERTa (XLM-R) represented the first major leap, training on **2.5 TB of Common Crawl data** across 100 languages with a 250K SentencePiece vocabulary. It improved over mBERT by +14.6% on XNLI and +11.4% for Urdu specifically. More recent models have pushed further: **multilingual E5 (mE5-large-instruct)**, built on XLM-R, emerged as the best publicly available model on MMTEB (2025), a benchmark spanning 500+ tasks across 250+ languages. **BGE-M3**, supporting 8,192 token inputs and 100+ languages, pioneered multi-functional retrieval (dense, sparse, and multi-vector) in a single architecture.

The fundamental tension is the **curse of multilinguality**: as more languages are added to a fixed-capacity model, per-language quality degrades. XLM-R experiments show XNLI accuracy drops from **71.8% with 7 languages to 67.7% with 100 languages**. The emerging solution is modular architectures — Mixture-of-Experts models like Nomic Embed v2 (2025) dynamically route inputs to language-appropriate parameter subsets, and adapter-based approaches like X-MOD add language-specific modules that can be extended without degrading existing languages.

4. Transformers Are Script-Agnostic by Design, but Not Script-Fair

The transformer doesn't care about script direction. But that doesn't mean all scripts get equal treatment.

A common misconception is that transformers need architectural modifications for right-to-left scripts like Arabic and Urdu. They do not. Self-attention computes pairwise relationships between **all tokens regardless of position**, and even Arabic/Urdu text is stored in left-to-right logical order in Unicode — right-to-left display is purely a rendering concern. Standard positional encodings (sinusoidal or RoPE) simply mark each token's position numerically.

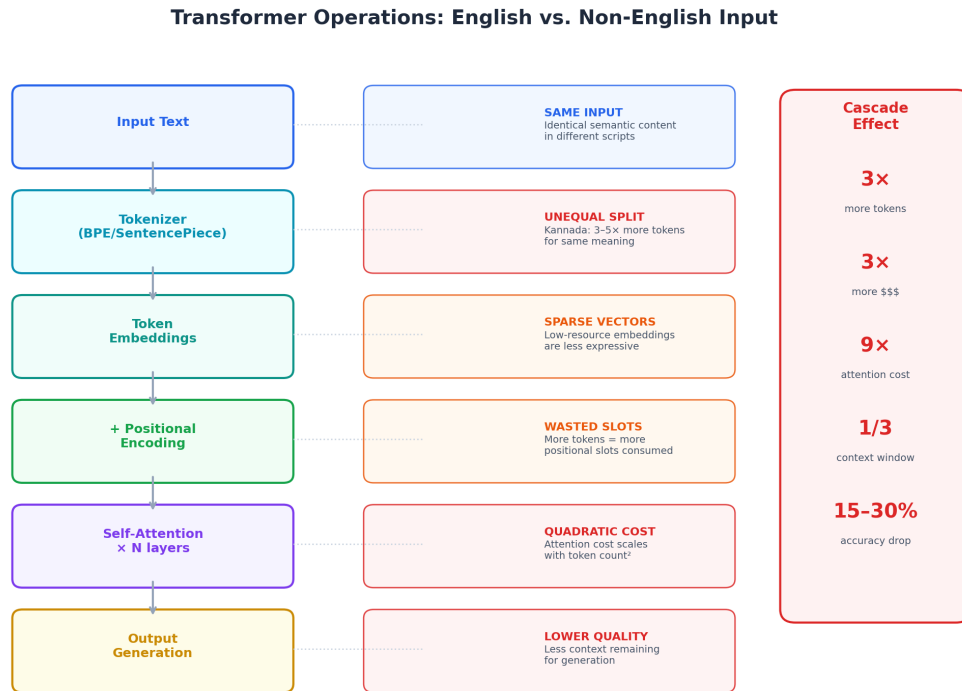


Figure 5: How tokenization inequality cascades through every transformer layer. More tokens means more positional slots consumed, quadratically higher attention cost, and less effective context for generation.

That said, research by Baniata et al. (2024) demonstrated that **Reverse Positional Encoding** — flipping the standard position assignments — improves performance on Arabic text classification. The real architectural challenge for Arabic isn't directionality but **morphology**: Arabic uses non-concatenative morphological templates where tense is marked by vowel changes inside consonant roots. BPE's sequential merging cannot capture these interleaved patterns.

The deeper architectural issue is that **high token fertility consumes positional encoding slots disproportionately**. For a model with 128K context, Hindi text at 3x fertility gets only ~42,000 word-equivalents of context compared to ~128,000 for English. This isn't a bug in the attention mechanism — it's a downstream consequence of tokenization inequality propagating through every layer of the architecture.

5. Training Data Starvation and Compounding Challenges

The root cause of multilingual inequality isn't algorithmic — it's data.

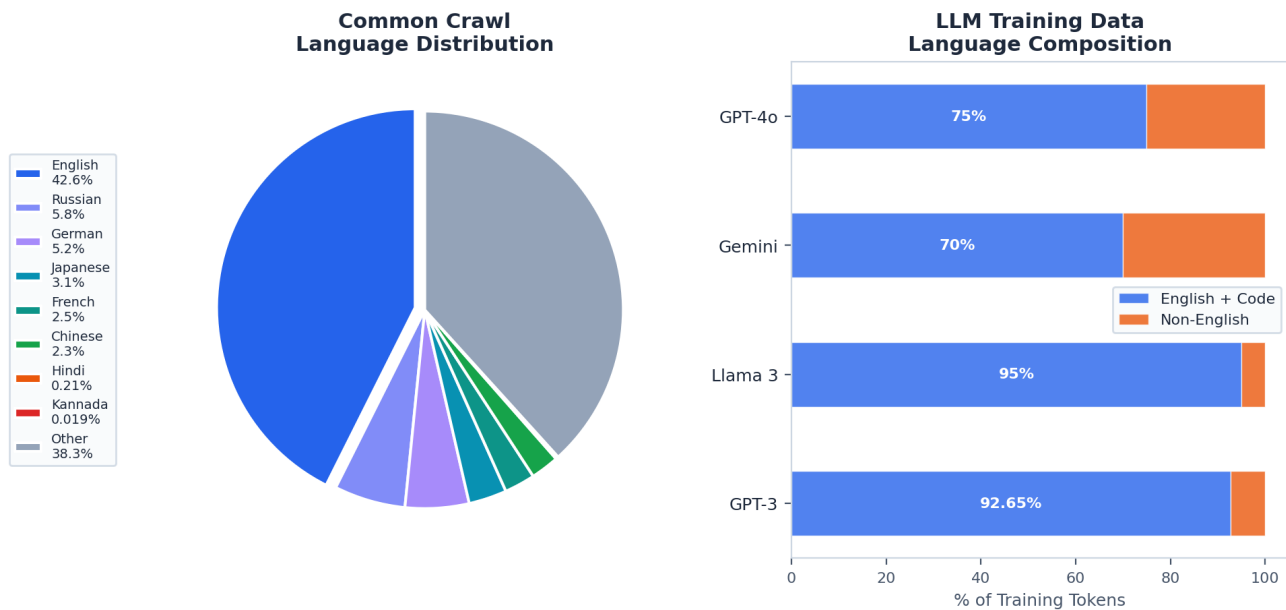


Figure 6: Language distribution in Common Crawl (left) and major LLM training corpora (right). English dominates at 42–95%, while Kannada represents just 0.019% of Common Crawl.

GPT-3 used **92.65% English** training tokens. Llama 3 used approximately 95% English and code. Common Crawl, the backbone of most training corpora, contains 42.6% English but only 0.21% Hindi and a mere **0.019% Kannada**. This creates a vicious cycle: languages with more web presence get better models, attracting more users who generate more content.

Script complexity interacts destructively with BPE tokenization. Kannada's abugida script permits up to **18,511 distinct character combinations** from 49 base phonemic characters. Conjunct consonants (ottaksharas) combine via virama characters into single visual glyphs that span 3–6 Unicode code points each. Dependent vowel signs appear in various positions — above, below, to the left, or to the right of consonant clusters — meaning what humans perceive as a single orthographic unit can span code points scattered across the sequence.

Morphological richness explodes vocabulary size for agglutinative languages. A single Turkish verb can take thousands of surface forms. Research evaluating morphological compositional generalization (2024) found that as morphological complexity increases, **LLM performance drops sharply to near zero**, while human performance remains unaffected — suggesting a fundamental limitation rather than a scaling issue.

Code-switching — mixing languages within a sentence, as in Hinglish (Hindi-English) — remains an area where LLMs significantly underperform fine-tuned smaller models. Zhang et al. (EMNLP 2023) demonstrated that even GPT-4 loses to task-specific fine-tuned models on code-switched sentiment analysis. The **English pivot phenomenon** compounds this further: research by Wendler et al. (2024) showed that decoder-only LLMs internally convert non-English inputs into English-like latent representations for reasoning, then generate in the target language — creating an information bottleneck where nuances that don't map cleanly to English concepts are lost.

6. Model Performance Comparison: Who Handles Non-English Best?

Benchmarking multilingual LLMs has matured dramatically since 2023, with purpose-built evaluations replacing adapted English benchmarks.

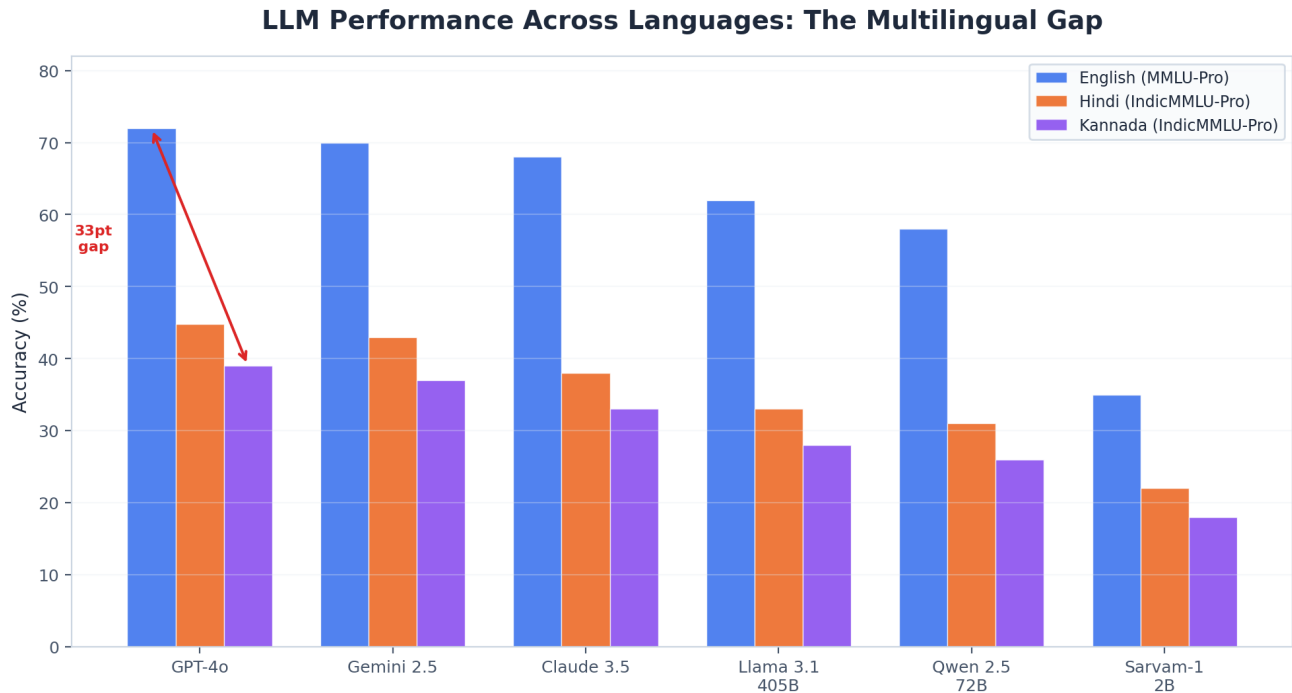


Figure 7: LLM accuracy across English, Hindi, and Kannada. Even the best models show 25–33 percentage point drops from English to Kannada on knowledge-intensive tasks.

The Artificial Analysis Multilingual Index (2025), testing across 16 languages, places **Google Gemini at the top in 13 of 16 languages**. For Indic languages specifically, **GPT-4o achieves the highest scores** on MILU (Multi-task Indic Language Understanding, AI4Bharat/NAACL 2025) at 72–74% average across 11 Indic languages and 85K questions, and on IndicMMLU-Pro at 44.8% for Hindi and 39.0% for Kannada.

Specialized Indic models dramatically underperform general-purpose LLMs — that is the most striking finding on reasoning and knowledge tasks. On IndicMMLU-Pro, models like IndicBERT and MuRIL score near random baseline levels (~11%). The exception is translation, where specialized models like IndicTrans2 convincingly outperform general LLMs.

Performance gaps between English and non-English remain substantial: **15–30% absolute accuracy drops** from English to Hindi/Kannada on knowledge-intensive tasks, reaching 38 percentage points for low-resource languages on MMLU-ProX (2025). STEM subjects show the smallest gaps (most transferable), while culturally specific knowledge shows the largest — up to 25–35% worse than STEM performance.

7. The Gemini Moment: When an LLM Wrote a Kannada Anchoring Speech

This entire research journey began with a single prompt to Gemini 3.

When KKNC (Kannada Koota Northern California) invited me to anchor their Sankranti celebration, I needed to craft a welcome speech that connected the cultural significance of Sankranti with the community's identity — in Kannada. Having anchored Kannada Rajyotsava the previous semester, I knew the flow I wanted. But for the segue between welcoming the audience and connecting KKNC's mission to Sankranti, I turned to Gemini 3.

The response was extraordinary. Gemini produced a Kannada anchoring script that wove cultural idioms like "Ellu-Bella" (the traditional sesame-jaggery exchange), references to village festivals, and the metaphor of Sankranti as a "home away from home" — all in grammatically coherent Kannada with appropriate honorifics and emotional cadence. **This is what the model produced:**

Gemini 3 Generated Script: KKNC Sankranti Anchoring Speech

Anchoring Speech for KKNC Sankranti

Paragraph 1:

"ಸಂಕ್ರಾಂತಿ ಎಂದರೆ 'ಸಂಕ್ರಮಣ' - ಅಂದರೆ ಒಂದು ಸ್ಥಿತಿಯಿಂದ ಇನ್ನೊಂದಕ್ಕೆ ಸೇರುವುದು ಅಥವಾ ಬೆರೆಯುವುದು ಎಂದರ್ಥ. ನಮ್ಮ KKNC (Kannada Koota Northern California) ಕೂಡ ಕಳೆದ ಹಲವಾರು ದಶಕಗಳಿಂದ ಇದೇ ಕೆಲಸವನ್ನು ಮಾಡುತ್ತಾ ಬಂದಿದೆ."

English Translation: "Sankranti means 'Sankramana'—which signifies transitioning or blending from one state to another. Our KKNC (Kannada Koota Northern California) has been doing exactly this for the past several decades."

Paragraph 2:

"ಸಾವಿರಾರು ಮೈಲಿ ದೂರದ ಅಮೇರಿಕಾದಲ್ಲಿರುವ ನಮ್ಮನ್ನು, ನಮ್ಮ ಮಣ್ಣಿನ ಸಂಸ್ಕೃತಿ, ಕಲೆ ಮತ್ತು ಭಾಷೆಯೊಂದಿಗೆ ಬೆಸೆಯುವ ಒಂದು ಸುಂದರ ಸೇತುವೆ ಈ 'ಕನ್ನಡ ಕೂಟ'. ಸಂಕ್ರಾಂತಿಯಂದು ನಾವು ಹೇಗೆ ಎಳ್ಳು-ಬೆಲ್ಲವನ್ನು ಬೆರೆಸಿ ಹಂಚುತ್ತೇವೆಯೋ, ಹಾಗೆಯೇ KKNC ಇಲ್ಲಿರುವ ಪ್ರತಿಯೊಬ್ಬ ಕನ್ನಡಿಗರನ್ನು ಪ್ರೀತಿ ಮತ್ತು ಸೌಹಾರ್ದತೆಯಿಂದ ಒಂದುಗೂಡಿಸಿದೆ. ಈ ವೇದಿಕೆ ಕೇವಲ ಕಾರ್ಯಕ್ರಮವಲ್ಲ, ಇದು ನಮ್ಮ 'ಪರಬೆಲಗಿನ' ಮನೆ!"

English Translation: "This 'Kannada Koota' is a beautiful bridge that connects us, who are thousands of miles away in America, with the culture, art, and language of our homeland. Just as we mix and share 'Ellu-Bella' (sesame seeds and jaggery) during Sankranti, KKNC has united every Kannadiga here with love and harmony. This stage is not just an event; it is our home away from home!"

Screenshot 1: The opening paragraphs of the Gemini-generated Kannada anchoring script, with English translations showing the cultural depth and linguistic coherence.

Anchoring Speech for KKNC Sankranti

The Spirit of the Festival (1:15 - 1:45)

"ಹೊಸ ಅಕ್ಕಿ, ಹೊಸ ಬೆಲ್ಲ, ರೈತರ ಮುಖದಲ್ಲಿ ಮಂದಹಾಸ... ಹಳ್ಳಿಗಳಲ್ಲಿ ಎತ್ತುಗಳನ್ನು ಕಿಚ್ಚು ಹಾಯಿಸುವುದು, ರಂಗೋಲಿ ಹಾಕುವುದು - ಇವೆಲ್ಲವನ್ನೂ ನಾವು ಇಲ್ಲಿ ಮಿಸ್ ಮಾಡಿಕೊಳ್ಳಬಹುದು. ಆದರೆ ಆ ಸಂಭ್ರಮದ ಕಿಚ್ಚು ನಮ್ಮ ಮನಸ್ಸಿನಲ್ಲಿ ಹಾಗೆಯೇ ಇದೆ. ಇಂದು ಈ ವೇದಿಕೆಯ ಮೇಲೆ ನಮ್ಮ ಮಕ್ಕಳು ಮತ್ತು ಕಲಾವಿದರು ಪ್ರಸ್ತುತಪಡಿಸುವ ಕಲಾಪ್ರತಿಭೆಗಳು ಆ ಹಳ್ಳಿಯ ಸೊಗಡನ್ನು ಸಾಂತಾ ಕ್ಲಾರಾದ ಈ ನೆಲಕ್ಕೆ ತರಲಿವೆ ಎಂಬ ನಂಬಿಕೆ ನನಗಿದೆ."

English Translation:

"Fresh rice, fresh jaggery, the smiles on the faces of farmers... making the bulls jump over the fire in the villages, drawing beautiful rangolis—we might miss experiencing all these things physically here. But the spark of that celebration remains burning brightly in our hearts. I truly believe that the artistic talents presented by our children and artists on this stage today will bring the authentic charm of our villages right here to the soil of Santa Clara."

Closing/Call to Action (1:45 - 2:00)

"ಬನ್ನಿ, ಎಳ್ಳು ಬೆಲ್ಲ ಸವಿಯುತ್ತಾ, ಒಳ್ಳೆಯ ಮಾತುಗಳನ್ನಾಡುತ್ತಾ, ಈ 'ಸಂಕ್ರಾಂತಿ ಸಂಭ್ರಮ'ವನ್ನು ಅದ್ದೂರಿಯಾಗಿ ಆಚರಿಸೋಣ."

'ಎಳ್ಳು ಬೆಲ್ಲ ತಿಂದು ಒಳ್ಳೆಯ ಮಾತಾಡಿ' ಎಂಬ ಮಾತಿನಂತೆ, ಇಂದಿನ ಕಾರ್ಯಕ್ರಮದ ಉದ್ದಕ್ಕೂ ನಿಮ್ಮ ಚಪ್ಪಾಳೆ ಮತ್ತು ಪ್ರೋತ್ಸಾಹ ಹೀಗೆಯೇ ಇರಲಿ. Now, let's kick off the performances!"

English Translation:

"Come, as we savor the Ellu-Bella and exchange pleasantries, let us celebrate this 'Sankranti Sambhrama' (Sankranti celebration) in a grand way."

Living up to the proverb, 'Eat Ellu-Bella and speak good words,' let your applause and encouragement continue to flow throughout today's program. Now, let's kick off the performances!"

Screenshot 2: The closing section with the call-to-action, incorporating the traditional proverb "Eat Ellu-Bella and speak good words" as a segue into the performances.

What shook me wasn't just the quality — it was the question it raised. Kannada represents **0.019% of Common Crawl**. Its script has 18,511 possible character combinations. Each character requires 3 bytes in UTF-8, meaning every word costs 3–5x more tokens than English. And yet, Gemini produced culturally nuanced, grammatically sound Kannada prose. **How?** That question led me down the rabbit hole that became this research document.

8. The Frontier of Solutions: From MAGNET to Sovereign Indic Models

The gap is real, but it's closing. Here's the research and engineering making it happen.

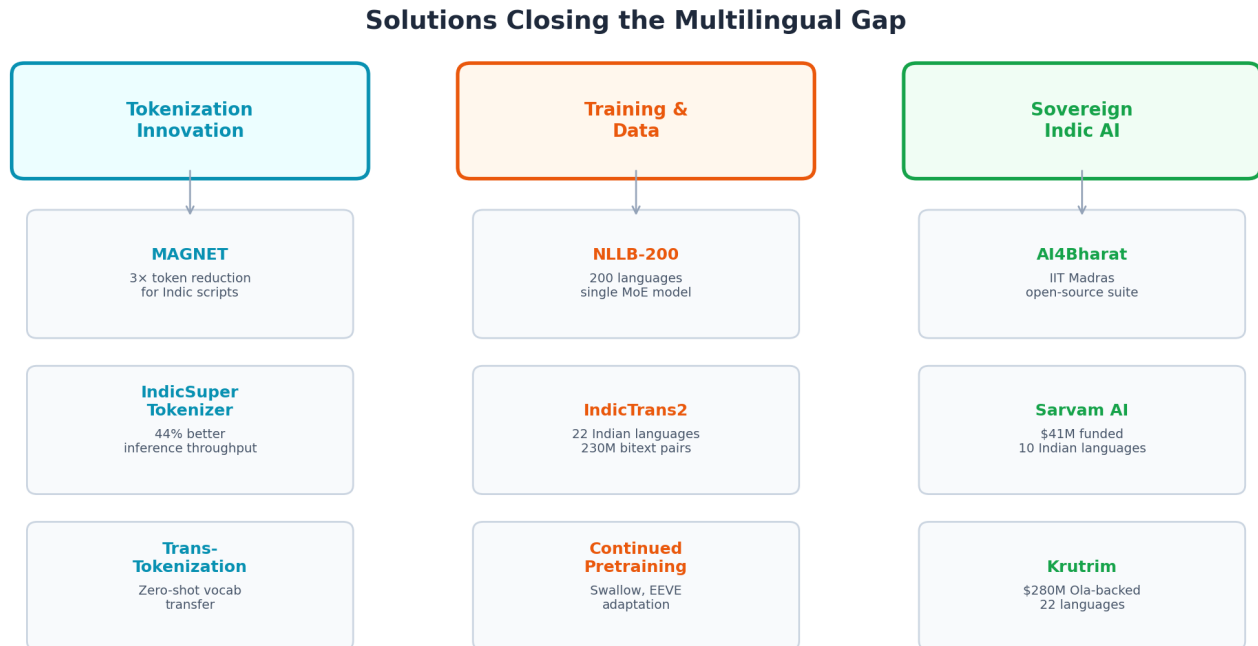


Figure 8: Three pillars of the multilingual solutions landscape — tokenization innovation, training data expansion, and India's sovereign AI ecosystem.

MAGNET (NeurIPS 2024) replaces fixed tokenizers with language-script-specific boundary predictors that dynamically segment byte sequences, achieving **3x token reduction** for Indic languages without affecting Latin-script performance. **IndicSuperTokenizer** (2025) combines subword and multi-word tokenization with language-specific pre-tokenization, delivering **44% improved inference throughput** over Llama-4 for Indic languages — its key insight being that careful pre-tokenization outweighs naive vocabulary scaling.

Vocabulary expansion has emerged as the most practical adaptation technique. **EEVE** (2024) demonstrated that parameter freezing with subword initialization can boost non-English proficiency within just 2 billion tokens of target language data. **Trans-Tokenization** (2024) enables zero-shot cross-lingual vocabulary transfer by initializing new token embeddings as weighted averages of aligned high-resource embeddings.

India has become a global hub for non-English AI development. **AI4Bharat** (IIT Madras) produced IndicTrans2, the first open-source translation model covering all 22 scheduled Indian languages, trained on 230 million bitext pairs. **Sarvam AI** (funded at \$41M) built Sarvam-1, a 2B-parameter model optimized for 10 Indian languages. **Krutrim** (by Ola, \$280M funding) launched models natively supporting 22 Indian languages with custom tokenizers. Meta's **NLLB-200** covers 200 languages with a single MoE model, achieving 44% improvement over previous state-of-the-art.

Conclusion: Tokenization Is the Original Sin, but Solutions Are Accelerating

The multilingual deficit in LLMs is not a single problem but a **cascading failure** that begins at tokenization and amplifies through every subsequent processing stage. A Kannada character that should be one token becomes three to five byte fragments, consuming context window capacity, inflating costs, and degrading output quality before the model's actual intelligence even engages.

The most actionable insight from this research is that **general-purpose frontier models (GPT-4o, Gemini 2.5) dramatically outperform specialized Indic models** on reasoning and knowledge tasks, while specialized models dominate only for translation. This suggests the path forward is not building separate models per language but improving how universal models handle non-English input — through better tokenizers, expanded vocabularies, and more balanced training data.

The explosive growth of India's AI ecosystem and government-backed initiatives signal that the resources and will to close this gap now exist. The technical solutions — adaptive tokenization, vocabulary expansion, cross-lingual transfer learning — are proven. What remains is the engineering and data-curation effort to deploy them at scale.

Key References

- [1] Petrov et al. (2023). "Language Model Tokenizers Introduce Unfairness Between Languages." NeurIPS 2023.
- [2] Ahia et al. (2023). "Do All Languages Cost the Same? Tokenization in Commercial LLMs." EMNLP 2023.
- [3] Hada et al. (2024). "MILU: Multi-task Indic Language Understanding Benchmark." NAACL 2025.
- [4] Verma et al. (2025). "IndicMMLU-Pro: Benchmarking Indic Large Language Models." arXiv:2501.15747.
- [5] Limisiewicz et al. (2024). "MAGNET: Adaptive Gradient-Based Tokenization." NeurIPS 2024.
- [6] Peri et al. (2025). "IndicSuperTokenizer: Optimized Tokenizer for Indic LLMs." arXiv:2511.03237.
- [7] Gala et al. (2023). "IndicTrans2: Translation Models for 22 Indian Languages." TMLR 2023.
- [8] Conneau et al. (2020). "Unsupervised Cross-lingual Representation Learning at Scale." ACL 2020.
- [9] NLLB Team (2022). "No Language Left Behind." Meta AI Research.
- [10] Muennighoff et al. (2025). "MMTEB: Massive Multilingual Text Embedding Benchmark." arXiv:2502.13595.
- [11] Zhang et al. (2025). "MMLU-ProX: Multilingual Benchmark for Advanced LLM Evaluation." arXiv:2503.10497.