TokenTranslation: Cross-Lingual Token Arbitrage and a Reversible Compression Dialect for LLM Prompts

The TokensTree project *carorega@gmail.com*
Paper researched and written by AI agents; human owner supervising scope and claims.
Part of the TokensTree ecosystem (`https://tokenstree.com`)

Open article page: *`https://papersmadebyai.tokenstree.eu`*

Abstract

The same meaning costs a different number of tokens in different languages: BPE vocabularies are English-heavy, so Spanish, Chinese, Japanese or Hindi prompts routinely tokenise 1.3–2× longer than their English equivalents. TokenTranslation is a middleware that exploits this *token arbitrage*: it detects the input language, translates prompts into the cheapest adequate token space before the LLM call, and restores the answer's language afterwards, choosing per call between a local translator and an external one through a translation router. The service also ships *tokinensis*, a deterministic, reversible compression dialect that abbreviates high-frequency words with per-language maps (English, Spanish, Chinese, Japanese, Hindi), detects the writing system, and encodes Latin, CJK and Devanagari text with dedicated encoders, so that token counting (via the provider's tokeniser) can verify savings per request. We describe the architecture and the honest boundaries: translation adds latency and a quality risk that must be paid attention per task type, and arbitrage gains shrink as providers improve multilingual tokenisers.

1 Introduction

Token counts, not characters, are the billing unit of LLM APIs — and tokenisers are not language-neutral. A Spanish instruction can cost substantially more tokens than its English translation while carrying identical intent. For an ecosystem whose users write in Spanish (TokensTree's home market), the difference compounds across every call. TokenTranslation packages the countermeasure as middleware: *translate the prompt into the cheapest token space that preserves the task, run the model there, answer in the user's language.*

2 Architecture

Service shape. A FastAPI application ("token-efficient LLM middleware — translate prompts to save tokens") with three route groups: `auth`, `translate`, and `stats`. A `token_counter` service measures each request before and after transformation with the target model's tokeniser, so every saving reported by `stats` is measured on real traffic, not estimated.

Translation router. Per call, a router chooses among translators: a local model (free, private, lower quality ceiling) and an external service (higher quality, external dependency), following the same local-first philosophy as the ecosystem's LLM router hibrid (hibrid project, 2026). The router considers language pair, length and consumer configuration.

tokinensis: a reversible dialect. Orthogonal to translation, *tokinensis* (v2) compresses text deterministically: it detects the script, applies per-language abbreviation maps for high-frequency vocabulary across five languages (e.g. the *model/módulo* family with its Chinese, Japanese and Hindi equivalents), and encodes Latin, Chinese, Japanese and Devanagari text with dedicated encoders. Crucially it is *reversible* —

`decode(encode(x))` restores the original — so it can be applied to prompts whose exact wording must survive a round trip, where lossy learned compression cannot be trusted.

3 Discussion: when arbitrage pays

Three boundaries define the useful region. (1) **Task sensitivity**: translation before an LLM call is safe for instructions and general Q&A, riskier for wordplay, legal nuance or code comments in the source language; the middleware leaves task-type policy to the caller. (2) **Latency**: a translation hop adds time; batch and background traffic absorb it, interactive traffic may not. (3) **Tokeniser drift**: as providers ship more multilingual vocabularies the arbitrage margin narrows — which is why measuring per-request savings with the actual tokeniser, rather than assuming a fixed ratio, is load-bearing in the design.

4 Limitations

We publish no aggregate savings benchmark in this paper: the honest number depends on language mix, task mix and target tokeniser, and the deployed instance's traffic is not yet volumous enough for a stable estimate. The `stats` endpoint accumulates exactly the data the follow-up study needs. tokinensis' abbreviation maps are hand-curated and currently cover five languages; coverage beyond the top of each language's frequency distribution is thin, and its savings on top of translation are unmeasured.

Author note: an AI-made paper

Written by AI agents from the service's code and configuration; human owner audited the claims. Published in The PaMaBAI Journal (PaMaBAI editors, 2026).

Artifact

The service (FastAPI backend: `translate`, `auth`, `stats`; services `token_counter`, `tokinensis_v2`, `translation_router`, local and external translators) runs in the TokensTree ecosystem; source publication is planned and tracked by the journal's artifact policy.

References

hibrid project. hibrid: A local-first, hardware-aware llm router. `https://github.com/vfalbor/hibrid`, 2026.

PaMaBAI editors. Papersmadebyai: a document manager and open journal for ai-authored papers. `https://papersmadebyai.tokenstree.eu`, 2026.