Is Multilingual LLM Watermarking Truly Multilingual? A Simple Back-Translation Solution

Anonymous ACL submission

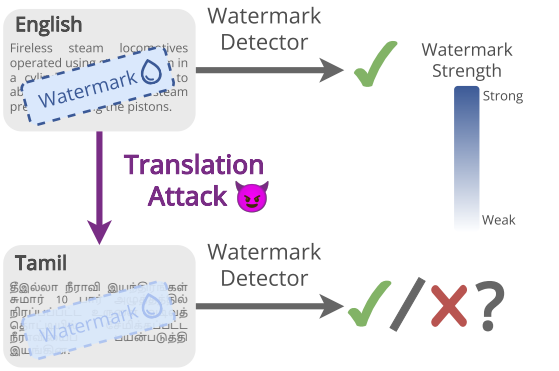
Abstract

Multilingual watermarking aims to make large language model (LLM) outputs traceable across languages, yet current methods still fall short. Despite claims of cross-lingual robustness, they are evaluated only on high-resource languages. We show that existing multilingual watermarking methods are not truly multilingual: they fail to remain robust under translation attacks in medium- and low-resource languages. We trace this failure to semantic clustering, which fails when the tokenizer vocabulary contains too few full-word tokens for a given language. To address this, we introduce STEAM, a back-translation-based detection method that restores watermark strength lost through translation. STEAM is compatible with any watermarking method, robust across different tokenizers and languages, non-invasive, and easily extendable to new languages. With average gains of +0.19 AUC and +40%p TPR@1% on 17 languages, STEAM provides a simple and robust path toward fairer watermarking across diverse languages.

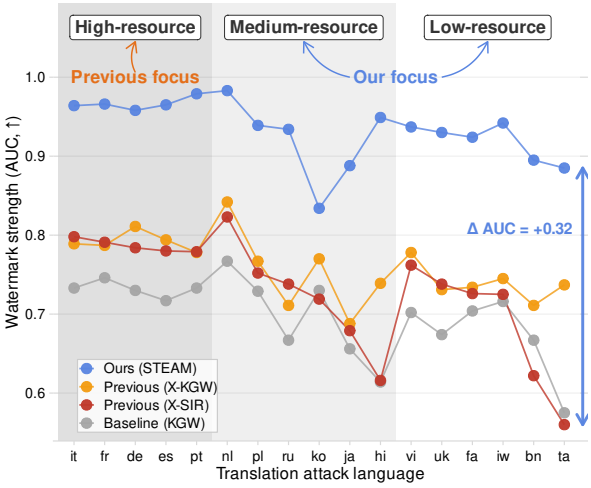
1 Introduction

Recent advances in multilingual watermarking claim to make large language model (LLM) outputs traceable across languages. Yet existing methods have been evaluated only on a small set of high-resource languages, leaving open the question of whether these techniques truly generalise to the world's linguistic diversity. In this work, we show that *current multilingual watermarking methods are not truly multilingual*. Their robustness weakens considerably for medium- and low-resource languages, revealing a major gap in current approaches to content provenance.

The limited robustness of multilingual watermarking has broad consequences. Watermarking was designed to identify LLM-generated text and to reduce the spread of misinformation on social media and synthetic content on the web. An adversary can exploit *translation attacks*, in which a model generates text in one language and the content is translated into another, effectively scrubbing the watermark and reducing its strength (He et al., 2024; Al Ghanim et al., 2025; Han et al., 2025; Luo et al., 2025; Chen et al., 2025). Figure 1a illustrates



(a) Translation attack against LLM watermarking.



(b) Watermarking robustness across languages.

Figure 1: (a) **Our goal** is to evaluate the robustness of LLM watermarks against translation attacks. (b) **Our analysis** reveals that existing multilingual watermarks fail to generalize across languages, while our approach (STEAM) performs consistently better across a wide range of languages overlooked by previous work.

a translation attack. This threat is not theoretical: large-scale deployed systems such as Google's SynthID (Dathathri et al., 2024), used in Gemini, Veo, Imagen, and others, lose detectability after translation (Han et al., 2025). This vulnerability could enable undetectable synthetic content to spread in hundreds of languages, particularly in communities where moderation tools are less effective.

Semantic clustering has been proposed as a multilingual extension of watermarking. It groups semantically equivalent tokens (for example, 'house', 'maison', 'casa') into clusters and treats all tokens in a cluster identically regarding the watermark key (for instance, all green or all red). While this approach performs adequately for high-resource languages, we observe that it performs poorly for many others. Tokenizers allocate tokens according to language frequency in their training data, meaning that only high-resource languages contain enough whole-word tokens to be properly represented in semantic clusters. For medium- and low-resource languages, most words are split into subword units not represented in any cluster, which substantially weakens the watermark. These findings suggest that semantic clustering cannot scale effectively beyond high-resource languages.

To address the limited robustness of semantic clustering, we introduce **STEAM** (*Simple Translation-Enhanced Approach for Multilingual watermarking*), a detection-time method that uses back-translation to recover the watermark strength lost during translation. STEAM is non-invasive, model-agnostic, and compatible with any existing watermarking technique. We evaluate STEAM on 17 languages covering high-, medium-, and low-resource settings. The results show large and consistent performance gains over semantic clustering, with *average improvements of +0.19 AUC and +40.0 percentage points (**%p**) in TPR@1%*. We perform an extensive robustness analysis, adaptive adversarial evaluation, and ablation study, all confirming STEAM's stability and effectiveness across diverse attack scenarios while maintaining a low false-positive rate.

Our contributions are:

1. **Extensive multilingual evaluation.** We conduct a large-scale evaluation of multilingual watermarking methods across 17 high-, medium-, and low-resource languages, uncovering weaknesses overlooked in prior work, which has focused exclusively on high-resource languages.
2. **Analysis of the limitations of semantic clustering.** We identify that the limitations of current multilingual watermarking stem from their core reliance on clusters of tokens.
3. **STEAM: a simple, robust multilingual defence.** We introduce STEAM, a back-translation-based watermark detection method that is modular, compatible with any watermarking technique and tokenizer, retroactively extensible to new languages, and non-invasive to the model output.
4. **Robustness across diverse languages.** STEAM consistently outperforms existing multilingual watermarking methods, with improvements of up to **+0.33 AUC** and **+64.5%p TPR@1%** across 17 languages.

2 Related Work

Depending on when the watermark is applied, LLM watermarking techniques are generally classified into training-time watermarking and inference-time watermarking (also known as logit-based watermarking) (Liu et al., 2024b). This work focuses exclusively on the latter.

Logit-based watermarking. Logit-based watermarking embeds a watermark by directly modifying the token probability distribution (logits) during text generation (Liu et al., 2024b). The seminal approach, KGW (Kirchenbauer et al., 2023), partitions the tokenizer vocabulary into green and red lists using a random seed derived from a fixed window of previous tokens and biases generation towards green tokens. Zhao et al. (2023) proposed Unigram Watermarking, an extension of KGW that employs a fixed green/red partition to improve robustness against text editing and paraphrasing attacks. To maintain text quality, Hu et al. (2023) introduced an unbiased watermarking approach that integrates watermarks without altering the overall probability distribution of the output. Several works (Lee et al., 2024; Lu et al., 2024a; Liu and Bu, 2024; Wu et al., 2024) further improve robustness while preserving text quality. Beyond these, ITS and EXP (Kuditipudi et al., 2024) offer model-agnostic, distortion-free watermarking schemes that remain robust to text manpulation attacks. This work analyses the multilingual capabilities of these techniques and builds upon them to develop our defence, STEAM.

Watermarking robustness. Several studies, including SIR (Liu et al., 2024a), SemaMark (Ren et al., 2024), semantic-aware watermarking (Fu et al., 2024), and SempStamp (Hou et al., 2024), incorporate semantic information to improve the robustness of watermarks against text transformation attacks. To achieve a balanced and context-aware partitioning of the green and red token lists, Guo et al. (2024) leveraged locality-sensitive hashing (LSH) (Indyk and Motwani, 1998) to generate a semantic key from contextual embeddings. Inspired by the inherent redundancy of multimedia data, WatME (Chen et al., 2024) embeds mutual exclusion rules within the lexical space for text watermarking. Furthermore, Luo et al. (2025) identified watermark collision, where multiple watermarks interact in ways that distort statistical distributions and hinder detection.

Multilingual watermarking. While much of the initial research focused on monolingual English text, a growing body of work now addresses the unique challenges of cross-lingual watermarking. A foundational contribution in this area is X-SIR (He et al., 2024), a direct extension of the SIR framework designed to defend against translation attacks. Other works have focused on evaluating the cross-lingual robustness of existing methods. For example, Han et al. (2025) assessed the robustness of SynthID-Text (Dathathri et al., 2024) to meaning-preserving transformations like back-translation. Similarly, Al Ghanim et al. (2025) conduct a comparative evaluation of four watermarking methods: KGW, Unigram, EXP, and X-SIR. Their analysis assesses robustness and text quality under various parameters and removal attacks in cross-lingual settings. Although these studies provide valuable insights, their scope is often limited to high-resource languages. Our work address this gap by providing a more comprehensive cross-lingual evaluation that includes an extensive set of low- and medium-resource languages.

3 Experimental Setup

This section outlines the experimental setup used to assess the robustness of multilingual watermarking methods across different languages, models, and attack scenarios.

Dataset. We base our evaluation on the English subset of the mC4 dataset (Raffel et al., 2023), following the setup introduced by He et al. (2024). We sample a test set of 500 texts for all experiments.

Criterion	KGW & SIR	X-KGW & X-SIR	STEAM (ours)
Multilingual support	✗	✓	✓
Non-invasive	–	✗	✓
Watermark-agnostic	–	✗	✓
Tokenizer-agnostic	–	✗	✓
New language support			
Medium-resource	–	~	✓
Low-resource	–	✗	✓†
Retroactive support	–	✗	✓

Table 1: Comparison of watermarking methods and their multilingual capabilities. Criteria definitions in Appendix A.5.
✓ = Yes, ✗ = No, ~ = Limited, – = Not applicable
† Requires translator (low-quality translation sufficient)

Attacks. We evaluate watermark robustness under two translation-based attacks. Unless specified otherwise, all translations are performed with Google Translate. The first, direct translation, converts English outputs into a target language and is used in the main experiments. The second applies multi-step translation through a pivot language (He et al., 2024) and is reported in Appendix B.

Multilingual models. We use the following multilingual language models: `Aya-23-8B` (Aryabumi et al., 2024), `LLaMA-3.2-1B` (Grattafiori et al., 2024), and `LLaMAX-8B` (Lu et al., 2024b).

Watermarking methods. We analyse three watermarking schemes. We use the standard KGW (Kirchenbauer et al., 2023) as our primary non-multilingual baseline. Second, we evaluate X-SIR (He et al., 2024), a foundational work that proposes semantic clustering for cross-lingual robustness. Finally, we introduce X-KGW, a method that applies semantic clustering to KGW. This setup allows us to isolate and measure the precise impact of semantic clustering on watermark robustness (see Appendix A.3 for details about X-KGW).

Evaluation metrics. We assess the strength of the watermark using two standard binary classification metrics: *(i)* Area Under the ROC Curve (*AUC*), measuring the probability that a watermarked sample receives a higher detection score than a non-watermarked one; and *(ii)* True Positive Rate at a fixed False Positive Rate (*TPR@1%*), the proportion of correctly identified watermarked texts when the false positive rate is fixed at 1%.

4 Semantic Clustering Fails in Diverse Multilingual Settings

In this section, we show that semantic clustering is not inherently multilingual. First, it lacks robustness in unsupported languages, requiring explicit support for each language to maintain detection strength. Second, attempts to extend this support to a larger set of languages fail, especially for medium- and low-resource cases. Finally, we analyse why semantic clustering does not generalise and identify key weaknesses that hinder its ability to generalise effectively to truly multilingual watermarking.

4.1 Robustness Against Unsupported Languages

Semantic clustering has only been evaluated on the languages it explicitly supports, so its robustness in unsupported languages remains unknown. We assess semantic clustering both within its originally supported languages using a hold-out setting, and on a broader set of unsupported ones to evaluate its cross-lingual generalisation.

Hold-one-out setup. This experiment evaluates how strongly X-SIR depends on its set of supported languages to be robust. Using the same languages as He et al. (2024), we exclude one language from the semantic clustering and then test the method on that withheld language. This setup allows us to measure how much X-SIR's robustness depends on explicit language support. The full results for all languages and models are provided in Appendix B.1. We find that excluding a language from the supported set leads to only minor average changes in performance: AUC decreases by -0.0003 and TPR@1% by -0.031 for LLaMA-3.2 1B, and by +0.013 and -0.023 respectively for Aya-23 8B. In several cases, AUC even increases when a language is removed (6 out of 16 for LLaMA-3.2 and 9 out of 16 for Aya), revealing that X-SIR's behaviour is highly variable and its robustness unreliable.

New languages setup. This experiment evaluates how much X-SIR relies on explicit language support to remain robust. If there is a large enough overlap of words between languages, supporting all languages may not be necessary. To test this, we extend the evaluation to the following set of unsupported languages: Italian (it), Spanish (es), Portuguese (pt), Polish (pl), Dutch (nl), Croatian (hr), Czech (cs), Danish (da), Korean (ko), and Arabic (ar). Appendix B.2 reports the performance of X-SIR and X-KGW for Aya-23 for the other models. Overall performance remains relatively low, with average AUC and TPR@1% of 0.75 and 0.14 for Aya-23 8B, and 0.68 and 0.07 for LLaMA-3.2 1B. Similar trends are observed for X-KGW. More importantly, several languages show clearly weaker watermark strength: for X-SIR, Arabic is the most vulnerable for Aya-23 8B (AUC of 0.687, TPR@1% of 0.093), while Czech and Arabic are weakest for LLaMA-3.2 1B (AUC of 0.650, TPR@1% of 0.055). These results indicate that even a single poorly supported language can allow an attacker to bypass watermark detection, highlighting the fragility of semantic-clustering-based multilingual watermarking.

Translation Attack		X-SIR (↑)		X-KGW (↑)	
Type	Lang.	AUC	TPR@1%	AUC	TPR@1%
High-resource	fr	0.791	0.149	0.787	0.280
	de	0.784	0.163	0.811	0.312
	it	0.798	0.152	0.789	0.354
	es	0.780	0.150	0.794	0.278
	pt	0.779	0.176	0.778	0.330
Medium-resource	pl	0.752	0.146	0.767	0.312
	nl	0.823	0.213	0.842	0.332
	ru	0.738	0.122	0.711	0.246
	hi	0.616	0.056	0.739	0.194
	ko	0.719	0.115	0.770	0.318
	ja	0.679	0.103	0.688	0.160
Low-resource	bn	0.622	0.055	0.711	0.180
	fa	0.726	0.131	0.734	0.242
	vi	0.762	0.157	0.778	0.308
	iw	0.725	0.115	0.745	0.220
	uk	0.738	0.148	0.731	0.222
	ta	0.560	0.049	0.737	0.172
Minimum		0.560 (ta)	0.049 (ta)	0.688 (ja)	0.160 (ja)

Table 2: **Even when more languages are explicitly supported, the robustness of semantic clustering decreases from high- to low-resource languages.** We extend semantic clustering to 17 newly supported languages. Aya-23 8B generates a text in English, then the translation attack is applied using each of these supported languages. Minimum indicates the worst-case robustness, i.e., the best language for an attack. Other models in Appendix B.3.

4.2 Failure to Support a Broad Range of Languages

Since X-SIR and X-KGW are not robust against translation to some unsupported languages, one possible solution is to extend the set of supported languages to cover most languages. In this section, we show that even when more languages are explic-

itly included, neither method achieves consistent robustness.

To evaluate the effectiveness of semantic clustering across languages, we extend the support of X-SIR and X-KGW to 17 languages spanning high-, medium-, and low-resource settings (methodology in Appendix A.4). The high-resource group includes French, German, Italian, Spanish, and Portuguese; the medium-resource group includes Polish, Dutch, Russian, Hindi, Korean, and Japanese; and the low-resource group includes Bengali, Persian, Vietnamese, Hebrew, Ukrainian, and Tamil.

The results are reported in Table 2 and Appendix B.3. For high-resource languages, X-SIR reaches an average AUC of 0.786 and TPR@1% of 0.158, while X-KGW achieves 0.792 and 0.311 respectively. These scores drop for medium-resource languages to 0.721 and 0.126 for X-SIR, and 0.753 and 0.260 for X-KGW. The decline continues for low-resource languages, where X-SIR records an average AUC of 0.689 and TPR@1% of 0.109, and X-KGW reaches 0.739 and 0.224. This trend indicates that semantic clustering robustness depends on language resource availability.

The performance gap becomes even more pronounced for specific low-resource languages. Tamil (ta) represents the weakest case of X-SIR on Aya-23 with an AUC of 0.560 and TPR@1% of 0.049. LLaMAX-3 shares the same observation (AUC of 0.561, TPR@1% of 0.067). Such drastic degradation highlights that even with explicit support, X-SIR fails to maintain reliable watermark detection across all languages.

These findings raise a critical question: why does explicit language support fail to guarantee robustness for semantic clustering-based watermarking?

4.3 On the Fundamental Limitations of Semantic Clustering in Multilingual Watermarking

Both X-SIR and X-KGW show clear weaknesses in mid- and low-resource languages. In this section, we argue that these limitations stem from a fundamental property of semantic clustering: its inability to generalise across languages due to the uneven coverage of full-word tokens in tokenizers.

Semantic clustering assigns watermark signals using multilingual dictionaries to group semantically equivalent words across languages (He et al., 2024). However, the share of dictionary words that appear as full tokens in tokenizer vocabularies varies sharply across languages (Appendix B.4).

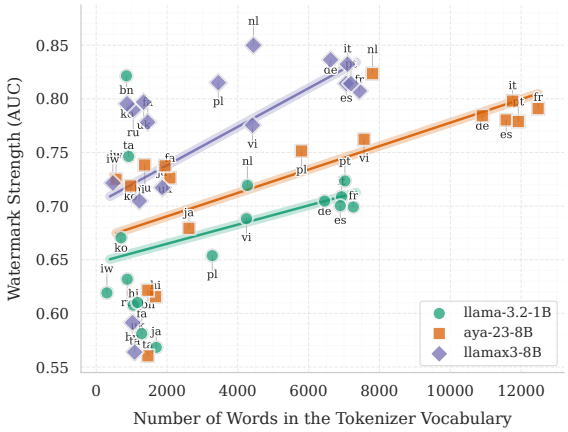


Figure 2: **Languages with larger tokenizer vocabularies have higher watermark robustness.** Average AUC per language and model across three seeds. Lines are least squared regressions.

Low-resource languages have very few full-word tokens, as low as 0.13% with Hebrew. BPE-based tokenizers allocate tokens by frequency in the training data, inherently favouring high-resource languages and fragmenting others into subword units with limited semantic meaning.

Figure 2 shows the relationship between watermark robustness (AUC) and the number of full-word tokens in the tokenizer vocabulary for each language. Across all three models, we observe a clear positive correlation: languages with higher token coverage achieve stronger watermark robustness, while those with lower coverage are far more vulnerable. This reveals a fundamental limitation of semantic clustering: *(i)* In the extreme case where a language has no full-word tokens, X-KGW collapses to KGW, as no token clusters can be formed. *(ii)* Even multilingual tokenizers cannot fully resolve this issue, since BPE allocation inherently disadvantages underrepresented languages. *(iii)* Most importantly, this vulnerability extends to monolingual watermarking: when text generated in one language (e.g., English) is translated into another, the target language may contain far fewer full-word tokens, enabling the watermark to be lost. These findings underscore that the shortcomings of semantic clustering are structural and cannot be overcome by simply expanding language support or retraining tokenizers.

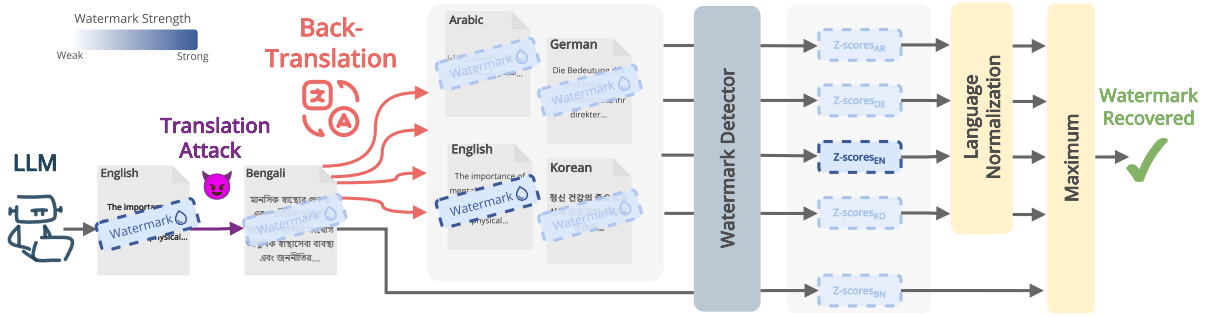


Figure 3: **Overview of STEAM**. A suspect text (Bengali) is back-translated into multiple supported languages to generate a pool of candidate texts. Each candidate, including the suspect text, is evaluated using a standard watermark detector to compute per-language z-score. STEAM outputs the highest normalized z-score.

5 STEAM: A Simple Back-Translation Defence for Many Diverse Languages

To address the limitations of semantic clustering for multilingual watermarking, we propose **Simple Translation-Enhanced Approach for Multilingual watermarking (STEAM)**, a novel, model-agnostic defence method based on back-translation. We first introduce STEAM, then evaluate its effectiveness in across different adversarial scenarios, and finally analyse its robustness.

5.1 STEAM Description

The core principle of STEAM is to amplify a potentially lost watermark signal through a multilingual back-translation and signal maximization process. To design this, a suspect text is first processed through a pipeline where it is back-translated to multiple supported languages to form a pool of candidate texts. Each candidate in this set, including the original suspect text, is then evaluated using a standard watermark detector (e.g., KGW) to compute their respective z-statistics, which measure the strength of the watermark signal (Kirchenbauer et al., 2023). From this collection of scores, STEAM selects the maximum value, which is then used as the decisive statistic for classification. An overview of the STEAM pipeline is shown in Figure 3.

Z-score language normalization. The signal maximization of STEAM is based on an assumption that is violated in low-resource languages: that a high z-score corresponds to a genuine watermark signal. Due to tokenizer limitations, single UTF-8 characters are often fragmented into high-frequency sub-character tokens (distribution of these tokens in Appendix B.6). For example, a single token represents 21.5% of all tokens in our Tamil texts for Llama 3.2, and 12.7% for Aya-23. If one of these very frequent tokens is assigned to the *green list* by the watermark key, the z-score of any text in the corresponding language will be high. This shift would cause STEAM to select the texts in this language, irrespectively of its watermark signal.

To mitigate this issue and preserve the watermark signal, we apply language-specific z-score normalization before selecting the highest-scoring back-translated text. For each language, we compute the mean z-score once using a distinct validation set of 500 human-written texts translated into that language. At test time, the corresponding language-specific mean is subtracted from each observed z-score to control for distribution shifts across languages.

5.2 STEAM Evaluation

Comparison to semantic clustering. We evaluate STEAM against semantic clustering methods to assess its robustness under translation attacks. As in §4.2, all methods are tested on the same set of 17 supported languages. STEAM achieves consistently strong results, maintaining an average AUC above **0.90** across all language categories, including medium- and low-resource ones (Table 3, Appendix B.5). Compared with semantic clustering approaches (X-SIR and X-KGW), STEAM shows large gains: on average, +0.205 AUC and +46.6%p TPR@1% relative to X-SIR, and +0.174 AUC and +33.3%p TPR@1% relative to X-KGW. The largest improvements are observed for Hindi and Portuguese, with up to +0.333 AUC and +64.6% TPR@1%, respectively. These gains are consistent across linguistically diverse languages, confirming that STEAM generalises reliably beyond high-resource settings. Unlike semantic cluster-

Translation Attack		AUC (↑)				TPR@1% (↑)			
Type	Language	KGW	X-KGW	X-SIR	STEAM	KGW	X-KGW	X-SIR	STEAM
High-resource	fr	0.746	0.787	0.791	**0.966**	0.224	0.280	0.149	**0.752**
	de	0.730	0.811	0.784	**0.958**	0.224	0.312	0.163	**0.684**
	it	0.733	0.789	0.798	**0.964**	0.202	0.354	0.152	**0.744**
	es	0.717	0.794	0.780	**0.965**	0.232	0.278	0.150	**0.712**
	pt	0.733	0.778	0.779	**0.979**	0.246	0.330	0.176	**0.822**
Medium-resource	pl	0.729	0.767	0.752	**0.939**	0.228	0.312	0.146	**0.654**
	nl	0.767	0.842	0.823	**0.983**	0.290	0.332	0.213	**0.822**
	ru	0.667	0.711	0.738	**0.934**	0.158	0.246	0.122	**0.510**
	hi	0.614	0.739	0.616	**0.949**	0.120	0.194	0.056	**0.650**
	ko	0.730	0.770	0.719	**0.834**	0.210	**0.318**	0.115	0.292
	ja	0.656	0.688	0.679	**0.888**	0.114	0.160	0.103	**0.438**
Low-resource	bn	0.667	0.711	0.622	**0.895**	0.068	0.180	0.055	**0.402**
	fa	0.704	0.734	0.726	**0.924**	0.196	0.242	0.131	**0.526**
	vi	0.702	0.778	0.762	**0.937**	0.186	0.308	0.157	**0.610**
	iw	0.716	0.745	0.725	**0.942**	0.172	0.220	0.115	**0.560**
	uk	0.674	0.731	0.738	**0.930**	0.210	0.222	0.148	**0.530**
	ta	0.575	0.737	0.560	**0.885**	0.082	0.172	0.049	**0.414**

Table 3: **STEAM is consistently better than semantic clustering by a large margin**. Watermark strength (AUC and TPR@1%) of multilingual watermarking techniques with 17 supported languages and Aya-23 8B. Red indicates that the defence reduces robustness (lower than the undefended KGW baseline). Bolded is best. Other models in Appendix B.5

ing, STEAM is robust in medium- and low-resource languages, unaffected by tokenizer limitations.

STEAM robustness to unsupported languages. Although the baseline evaluation confirms the efficacy of STEAM under ideal conditions, a more rigorous stress test is required to assess its resilience. We therefore investigate its performance when the set of supported languages is misspecified.

In our experimental design, we deliberately exclude the ground-truth source language from the back-translation pool. In this setup, STEAM performance remains comparable to X-KGW and significantly outperforms both X-SIR and the undefended baseline in terms of AUC and TPR@1% (Table 4). Unlike semantic clustering methods, which collapse in this scenario due to their dependence on dictionary coverage, STEAM robustness is bounded by the diversity of its back-translation pool. Expanding the set of candidate languages could further improve its robustness, thereby increasing the likelihood of capturing stronger watermark signals.

New Language	AUC (↑)			
	KGW	X-KGW	X-SIR	STEAM
it	0.733	0.772	**0.796**	0.783
es	0.717	**0.807**	0.754	0.779
pt	0.732	**0.792**	0.775	0.782
pl	0.730	0.762	0.749	**0.763**
nl	0.768	**0.808**	0.776	0.782
hr	0.706	0.757	0.726	**0.769**
cs	0.717	0.754	0.773	**0.778**
da	0.713	0.764	0.734	**0.780**
ko	0.732	**0.754**	0.729	0.749
ar	0.689	**0.765**	0.687	0.753

Table 4: **STEAM performs on par with other multilingual methods on unsupported languages.** Bolded is best. Red indicates that the defence reduces robustness (lower than the undefended KGW baseline). Full table in Appendix B.5

5.3 Robustness Analysis

Robustness to translator mismatch. The robustness of STEAM should not depend on the specific translation service used. An adversary could try to bypass our defence by using a different translation system for their attack. We examine whether the performance of STEAM remains robust when the translation service used for the attack differs from the one used for the defence. We compare a setting where both the attack and STEAM use Google Translate with a mismatch setting, where the attack uses Google Translate and STEAM uses DeepSeek-V3.2-Exp for back-translation (DeepSeek-AI et al., 2025). Table 5 and Appendix B.5 show that the average AUC remains above 0.90 in all languages, and that changing the translator actually improves detection in all but three cases. We hypothesise that DeepSeek produces higher-quality translations, which better preserve the watermark signal even

Translation Attack		Translator Mismatch Effect	
Type	Language	Δ AUC ↑	Δ TPR@1% ↑
High-resource	fr	+0.011	+0.058
	de	+0.002	+0.026
	it	+0.012	+0.096
	es	+0.010	+0.082
	pt	-0.003	+0.004
Medium-resource	pl	+0.030	+0.096
	nl	+0.002	+0.024
	ru	+0.014	+0.190
	hi	+0.023	+0.106
	ko	+0.051	+0.088
	ja	-0.022	-0.044
Low-resource	bn	+0.041	+0.144
	fa	+0.019	+0.074
	vi	+0.012	+0.058
	iw	+0.023	+0.138
	uk	+0.029	+0.154
	ta	+0.034	+0.152

Table 5: **STEAM is robust to a translator mismatch**. Positive values (green) indicates that STEAM is more reliable when using a different translator, and negative values a drop of robustness (red). Difference of AUC and TPR@1% when using Google Translate for both the translation attack and the back-translation defense and when using Google Translate for the translation attack and DeepSeek-V3.2-Exp for the back-translation defence. Full table in Appendix B.5.

though the two systems differ substantially. This suggests that our method genuinely recovers watermark strength rather than relying on translator-specific artefacts.

Adaptive evaluation: multistep translation attack. To assess the robustness of our defence under adaptive attack, we introduce a stronger multistep translation attack that adds an extra translation step beyond the single-hop setup. This design prevents STEAM from relying on direct back-translation to recover the watermark signal. In this two-step attack, the text is first translated using the full set of languages from §4.2, and the resulting output is then translated again through one of three pivot languages: German (high-resource), Korean (medium-resource), or Bengali (low-resource) (Appendix B.5). Despite this adaptive setup, STEAM remains robust, maintaining AUC values above 0.80 across all conditions, with only modest degradation when the second pivot is Korean or Bengali. While such multi-hop attacks can weaken other defences, they also tend to reduce overall translation quality, limiting their practical impact.

Ablation study. We conduct an ablation study to assess the impact of per-language z-score normalization on STEAM's performance. This component corrects the cross-lingual differences in the number of green tokens, which would otherwise increase the false positive rate. Removing normalization leads to only a modest AUC drop of 0.049. But it greatly reduces correct language identification, from 83.5% to 38.6% (Table 6). So, under adaptive multistep attacks while STEAM remains robust even when the language is misidentified, normalization improves its stability and quality.

Z-score normalization	Watermark		Language Pred.
	AUC ↑	TPR@1% ↑	Accuracy (%) ↑
No	0.902 ±0.11	0.34 ±0.13	38.6 ±16.6
Yes (STEAM)	0.951 ±0.03	0.64 ±0.17	83.5 ±09.0

Table 6: **Ablation study**. The z-score normalization in STEAM controls for differences in green token counts across languages. With normalization, STEAM selects the correct language and yields higher watermark strength. AUC, TPR@1%, and language prediction accuracy (percentage of cases where the highest z-score corresponds to the back-translation to the original language), averaged over 17 languages with and without normalization.

6 Conclusion

We showed that current multilingual watermarking methods fail to remain robust under translation attacks, especially in medium- and low-resource languages. If watermarking lacks robustness in a given language, online content in that language may be disproportionately affected by synthetic or undesirable content. This risk is especially serious for low- and medium-resource languages, which already face a shortage of high-quality digital resources and often lack effective moderation systems.

To address this, we introduced STEAM, a simple, watermark-agnostic back-translation method that restores watermark strength lost through translation. Extensive experiments on 17 languages and diverse attack scenarios show that STEAM achieves consistently stronger robustness and fairness than existing multilingual watermarking methods, particularly in medium- and low-resource settings.

Our findings highlight the need for watermarking research to treat linguistic diversity and fairness as core requirements, ensuring that the security and trust of large language models extend to all languages, not only those with abundant digital resources.

Limitations

While our proposed method, STEAM, demonstrates significant improvements in multilingual watermarking, we acknowledge several limitations that also present avenues for future research.

Our evaluation considers a set of 17 languages, chosen to represent diverse linguistic families. However, this set might not be fully representative of the linguistic diversity of the world.

STEAM demonstrates clear advantages over prior multilingual watermarking techniques on supported languages. However, its performance on unsupported languages remains comparable to existing methods. Nevertheless, a key strength of STEAM is that it can easily support additional languages. We believe that a broad coverage of languages is necessary for all multilingual watermark techniques.

The operational cost of STEAM, measured in translation API requests, scales linearly with the number of supported languages. While this presents a potential scalability concern, our empirical results show that the method's performance gains do not depend on high-cost translation services. The use of standard, widely available tools like Google Translate proved sufficient to achieve consistent improvements.

Finally, the current implementation of STEAM is specifically designed to defend against translation-based attacks. It is not designed to counter other significant text transformation attacks, such as paraphrasing attacks. This focus is a deliberate choice: STEAM is designed to be modular, allowing the translation robustness component to operate independently. This modularity ensures that other parts of the watermarking pipeline are not affected and provides a clear path for future enhancements. Future research could focus on creating and integrating new modules to build a more holistically robust watermarking system.

Ethical Considerations

This work has potential dual-use implications. On one hand, studying adversarial attacks against watermarking could inform malicious actors about possible strategies to weaken watermark defences. However, we believe the benefits outweigh these risks.

First, our contribution is not only an analysis but also a concrete defence (STEAM) that achieves a high level of robustness, substantially exceeding prior multilingual watermarking methods. Our results demonstrate that STEAM provides consistently strong robustness against translation attacks across a wide range of languages.

Second, by explicitly addressing low- and medium-resource languages, our method promotes fairness: watermarking becomes more reliable across diverse linguistic settings, rather than being limited to a handful of high-resource languages.

Robust multilingual watermarking is an important safeguard against misuse of large language models, such as the generation and dissemination of fake news or disinformation in less-resourced languages where moderation tools are often weaker. We view this work as a step toward improving the security and trustworthiness of multilingual AI systems.

References

Mansour Al Ghanim, Jiaqi Xue, Rochana Prih Hastuti, Mengxin Zheng, Yan Solihin, and Qian Lou. 2025. Evaluating the robustness and accuracy of text watermarking under real-world cross-lingual manipulations. In *Findings of the Association for Computational Linguistics: EMNLP 2025*. Association for Computational Linguistics.

Viraat Aryabumi, John Dang, Dwarak Talupuru, Saurabh Dash, David Cairuz, Hangyu Lin, Bharat Venkitesh, Madeline Smith, Kelly Marchisio, Sebastian Ruder, Acyr Locatelli, Julia Kreutzer, Nick Frosst, Phil Blunsom, Marzieh Fadaee, Ahmet Üstün, and Sara Hooker. 2024. Aya 23: Open weight releases to further multilingual progress. *Preprint*, arXiv:2405.15032.

Liang Chen, Yatao Bian, Yang Deng, Deng Cai, Shuaiyi Li, Peilin Zhao, and Kam fai Wong. 2024. Watme: Towards lossless watermarking through lexical redundancy. *Preprint*, arXiv:2311.09832.

Ruibo Chen, Yihan Wu, Junfeng Guo, and Heng Huang. 2025. Improved unbiased watermark for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20587–20601, Vienna, Austria. Association for Computational Linguistics.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, Jamie Hayes, Nidhi Vyas, Majd Al Merey, Jonah Brown-Cohen, Rudy Bunel,

Borja Balle, Taylan Cemgil, Zahra Ahmed, Kitty Stacpoole, and 5 others. 2024. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Yu Fu, Deyi Xiong, and Yue Dong. 2024. Watermarking conditional text generation for ai detection: Unveiling challenges and a semantic-aware watermark remedy. *Preprint*, arXiv:2307.13808.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Yuxuan Guo, Zhiliang Tian, Yiping Song, Tianlun Liu, Liang Ding, and Dongsheng Li. 2024. Context-aware watermark with semantic balanced green-red lists for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22633–22646, Miami, Florida, USA. Association for Computational Linguistics.

Xia Han, Qi Li, Jianbing Ni, and Mohammad Zulkernine. 2025. Robustness assessment and enhancement of text watermarking for google's synthid. *Preprint*, arXiv:2508.20228.

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. 2024. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4115–4129, Bangkok, Thailand. Association for Computational Linguistics.

Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. 2024. SemStamp: A semantic watermark with paraphrastic robustness for text generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4067–4082, Mexico City, Mexico. Association for Computational Linguistics.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. 2023. Unbiased watermark for large language models. *Preprint*, arXiv:2310.10669.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA. Association for Computing Machinery.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. 2024. Robust distortion-free watermarks for language models. *Preprint*, arXiv:2307.15593.

Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. 2024. Who wrote this code? watermarking for code generation. *Preprint*, arXiv:2305.15060.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. 2024a. A semantic invariant robust watermark for large language models. In *The Twelfth International Conference on Learning Representations*.

Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip S. Yu. 2024b. A survey of text watermarking in the era of large language models. *Preprint*, arXiv:2312.07913.

Yepeng Liu and Yuheng Bu. 2024. Adaptive text watermark for large language models. *Preprint*, arXiv:2401.13927.

Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. 2024a. An entropy-based text watermarking detection method. *Preprint*, arXiv:2403.13485.

Yinquan Lu, Wenhao Zhu, Lei Li, Yu Qiao, and Fei Yuan. 2024b. LLaMAX: Scaling linguistic horizons of LLM by enhancing translation capabilities beyond 100 languages. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10748–10772, Miami, Florida, USA. Association for Computational Linguistics.

Yiyang Luo, Ke Lin, Chao Gu, Jiahui Hou, Lijie Wen, and Luo Ping. 2025. Lost in overlap: Exploring logit-based watermark collision in LLMs. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 620–637, Albuquerque, New Mexico. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward

Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. Pytorch: An imperative style, high-performance deep learning library. *Preprint*, arXiv:1912.01703.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Jie Ren, Han Xu, Yiding Liu, Yingqian Cui, Shuaiqiang Wang, Dawei Yin, and Jiliang Tang. 2024. A robust semantics-based watermark for large language model against paraphrasing. *Preprint*, arXiv:2311.08721.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yihan Wu, Zhengmian Hu, Junfeng Guo, Hongyang Zhang, and Heng Huang. 2024. A resilient and accessible distribution-preserving watermark for large language models. *Preprint*, arXiv:2310.07710.

Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. 2023. Provable robust watermarking for ai-generated text. *Preprint*, arXiv:2306.17439.

Appendix

The appendices contain the following sections:

- Appendix A details the experimental settings,
- Appendix B contains additional experimental results,
- Appendix C contains the details of our usage of AI assistants.
- Appendix D contains the details of our artifacts.

For transparency and reproducibility, our code is available at this anonymized URL[1]: https://anonymous.4open.science/r/STEAM-BBFA

A Experimental Setting

A.1 Hyperparameters

To ensure reproducibility, we detail the hyperparameters used for both the neural network training and the watermark generation/detection phases of our experiments.

X-SIR neural network training. The neural network component of X-SIR, which inherits its architecture from SIR, was trained using the following hyperparameters:

- **Architecture:** The model consists of 4 layers, with an input dimension of 1024, a hidden dimension of 500, and an output dimension of 300.
- **Optimization:** We used *Stochastic Gradient Descent (SGD)* with a *learning rate* of 0.006 and a *weight decay* of 0.2. A `StepLR` scheduler with a *step size* of 200 and a gamma of 0.1 was employed to adjust the learning rate during training.
- **Training:** The model was trained for 2000 epochs with a batch size of 32.

Watermarking scheme parameters. For the watermark generation and detection phases, the following parameters were used for each scheme:

- **KGW:** We used the default parameters recommended by Kirchenbauer et al. (2023): a green list proportion (gamma) of 0.25, a logit bias (delta) of 2.0, and the `minhash` seeding scheme.
- **X-KGW:** To create a direct comparison with XSIR, we set the context width to 1. The gamma and delta values were kept consistent with KGW at 0.25 and 2.0, respectively.
- **X-SIR:** We followed the original implementation, setting the window size to 5, the chunk size to 10, and the logit bias (delta) to 1.0. The multilingual sentence embeddings were generated using the `paraphrase-multilingual-mpnet-base-v2` model.

A.2 Computational Resources & Softwares

All experiments were conducted on a Google Cloud Platform instance of type `n1-standard-4`, equipped with 4 vCPUs, 15 GB of RAM, and two NVIDIA T4 GPUs.

All translations use the Google Translate service accessed through the `deep_translator` Python library, which provides a unified interface to various translation APIs. The translator mismatch experiment in §5.3 employs the DeepSeek API for back-translation.

We used Pytorch[2] as our deep learning framework (Paszke et al., 2019), with CUDA support for GPU acceleration. In addition, we employed Hugging Face Transformers library[3] (Wolf et al., 2020) to access pretrained models and tokenizers.

[1] Anonymised repository and URL for double-blind review
[2] https://pytorch.org/
[3] https://huggingface.co/

A.3 Description of X-KGW

X-KGW (Cross-lingual KGW) is a hybrid watermarking approach we introduce to combine the hash-based mechanism of KGW with the semantic clustering strategy of X-SIR. Unlike KGW, which partitions individual tokens, X-KGW operates at the cluster level. The process consists of three distinct phases:

1. **Semantic cluster construction:** Following the X-SIR framework, we first construct a multilingual semantic graph using bilingual translation dictionaries. The Louvain community detection algorithm is then applied to partition the vocabulary $\mathcal{V}$ into $\mathcal{C}$ disjoint semantic clusters, yielding a token-to-cluster mapping $m : \mathcal{V} \to 0, 1, \ldots, \mathcal{C} - 1$.

2. **Hash-based cluster partitioning.** During text generation, at each timestep t, a context window of preceding tokens $(w_{t-h}, \ldots, w_{t-1})$ is used to compute a hash-based seed. This seed is then employed to pseudo-randomly partition the $\mathcal{C}$ clusters into green and red sets, with a fraction γ designated as green.

3. **Cluster-based logit modification.** Finally, a positive bias δ is applied to the logits of all tokens belonging to clusters assigned to the green set. The model then samples the next token from this modified probability distribution.

By combining KGW logit biasing with semantic clustering, X-KGW seeks to preserve watermark robustness under multilingual transformations while maintaining detection accuracy.

A.4 Multilingual Dictionnaries & Language Categorization

To construct our multilingual dictionary, we relied on the MUSE dictionary (Conneau et al., 2017), the same resource used by He et al. (2024) to build the semantic clusters. In addition to its role in dictionary-based clustering, we used MUSE to categorize the 17 languages included in our evaluation of §4.2, §4.3, §5.2, §5.3.

- A language was marked *high-resource* if it possesses extensive, non-English-centric dictionary mappings (i.e., bidirectional dictionaries with multiple other languages in the set).

- In contrast, languages whose resources are primarily English-centric, where MUSE provides only bidirectional dictionaries with English, were classified as either *medium-resource* or *low-resource*. The distinction between these two groups was determined by the size (i.e., the number of word pairs) of their respective English dictionaries.

A.5 Definitions of Multilingual Watermarking Comparison Criteria

For clarity, we provide the definitions of the criteria used in Table 1 to compare multilingual watermarking techniques:

- Multilingual support: Designed to resist translation attacks.

- Non-invasive: Supporting multilingual does not change the logits during generation, so the text quality is garanteed to be preserved.

- Watermark-agnostic: Can be combined with any watermarking technique without modification.

- Tokenizer-agnostic: Robustness against translation attacks does not depend on the tokenizer.

- Medium/low-resource: Robust against translation attacks to medium-/low-resource languages.

- Retroactive support: Allows adding new languages without regenerating the watermark key (red/green tokens split). Already generated texts can be detected in the new languages.

A.6 Prompt for DeepSeek-V3.2-Exp Translation

To use DeepSeek-V3.2-Exp as a translation engine, we designed a structured prompt format. We define:

```
Source language: {src_lang}
Target language: {tgt_lang}
Input text: {response}
```

`src_lang` and `tgt_lang` indicate the source and target language codes. We convert these codes into their full language names using the `Language` class from the `langcodes` [4] library:

```
Language.make(language=src_lang).display_name()
Language.make(language=tgt_lang).display_name()
```

The final prompt provided to DeepSeek-V3.2-Exp:

```
Translate the following {Language.make(language=src_lang).display_name()}
text to {Language.make(language=tgt_lang).display_name()}:

{response}
```

[4] https://pypi.org/project/langcodes/

B Additional Results

B.1 Hold-out languages for X-SIR

Languages		AUC (↑)			TPR@1% (↑)		
Held-out	Prompt	Held-Out	Supported	Δ	Held-Out	Supported	Δ
en	fr	0.795 ±0.045	0.816 ±0.014	+0.021	0.198 ±0.049	0.149 ±0.042	-0.049
	de	0.780 ±0.054	0.811 ±0.018	+0.031	0.172 ±0.047	0.168 ±0.039	-0.004
	zh	0.731 ±0.020	0.669 ±0.042	-0.062	0.141 ±0.011	0.083 ±0.014	-0.058
fr	en	0.757 ±0.022	0.799 ±0.027	+0.042	0.157 ±0.016	0.139 ±0.037	-0.018
	de	0.723 ±0.020	0.781 ±0.025	+0.058	0.101 ±0.029	0.156 ±0.061	+0.055
	zh	0.651 ±0.026	0.638 ±0.037	-0.013	0.076 ±0.021	0.052 ±0.020	-0.024
de	en	0.736 ±0.011	0.802 ±0.020	+0.067	0.153 ±0.050	0.214 ±0.035	+0.061
	fr	0.765 ±0.004	0.784 ±0.020	+0.019	0.139 ±0.068	0.118 ±0.039	-0.021
	zh	0.667 ±0.041	0.642 ±0.014	-0.025	0.073 ±0.032	0.065 ±0.016	-0.008
zh	en	0.644 ±0.052	0.692 ±0.041	+0.048	0.120 ±0.046	0.111 ±0.011	-0.009
	fr	0.671 ±0.072	0.714 ±0.045	+0.043	0.112 ±0.053	0.069 ±0.025	-0.043
	de	0.675 ±0.048	0.701 ±0.026	+0.026	0.105 ±0.053	0.107 ±0.027	+0.002
ja	en	0.685 ±0.059	0.656 ±0.017	-0.029	0.113 ±0.024	0.070 ±0.008	-0.043
	fr	0.698 ±0.038	0.670 ±0.018	-0.028	0.101 ±0.033	0.089 ±0.023	-0.012
	de	0.688 ±0.037	0.669 ±0.027	-0.019	0.138 ±0.031	0.079 ±0.005	-0.059

Table 7: **Semantic clustering (XSIR) is weak for hold-out unsupported languages**. Δ measures the robustness gains against a translation attack on a language after it has been supported by XSIR. The semantic clustering of tokens is applied on all the original five languages of XSIR (en, fr, de, zh, ja) for *supported*, and on all but the held-out language for *held-out*. Aya-23 8B generates a text in the *Prompt* language, then the translation attack is applied on the held-out language. Red indicates that XSIR performs worst after supporting the held-out language. Other models in Appendix B.1.

Languages		AUC (↑)			TPR@1% (↑)		
Held-out	**Prompt**	**Supported**	**Held-Out**	**Δ**	**Supported**	**Held-Out**	**Δ**
en	fr	$0.901_{0.017}$	$0.907_{0.015}$	+0.006	$0.363_{0.047}$	$0.275_{0.007}$	-0.088
	de	$0.884_{0.043}$	$0.894_{0.041}$	+0.010	$0.379_{0.131}$	$0.331_{0.043}$	-0.048
	zh	$0.795_{0.043}$	$0.827_{0.013}$	+0.032	$0.450_{0.040}$	$0.407_{0.011}$	-0.043
fr	en	$0.743_{0.050}$	$0.682_{0.026}$	-0.061	$0.113_{0.056}$	$0.068_{0.012}$	-0.045
	de	$0.787_{0.050}$	$0.687_{0.011}$	-0.100	$0.161_{0.038}$	$0.093_{0.007}$	-0.068
	zh	$0.678_{0.006}$	$0.681_{0.007}$	+0.003	$0.086_{0.027}$	$0.083_{0.005}$	-0.003
de	en	$0.693_{0.017}$	$0.692_{0.043}$	-0.001	$0.069_{0.022}$	$0.068_{0.025}$	-0.001
	fr	$0.724_{0.023}$	$0.725_{0.044}$	+0.001	$0.098_{0.011}$	$0.071_{0.009}$	-0.027
	zh	$0.665_{0.018}$	$0.693_{0.046}$	+0.028	$0.065_{0.009}$	$0.073_{0.014}$	+0.008
zh	en	$0.666_{0.035}$	$0.605_{0.012}$	-0.061	$0.082_{0.052}$	$0.026_{0.006}$	-0.056
	fr	$0.704_{0.011}$	$0.609_{0.024}$	-0.095	$0.085_{0.041}$	$0.036_{0.012}$	-0.049
	de	$0.703_{0.022}$	$0.636_{0.017}$	-0.067	$0.088_{0.010}$	$0.050_{0.008}$	-0.038
ja	en	$0.576_{0.052}$	$0.573_{0.024}$	-0.003	$0.039_{0.019}$	$0.033_{0.005}$	-0.006
	fr	$0.630_{0.066}$	$0.581_{0.041}$	-0.049	$0.067_{0.016}$	$0.025_{0.014}$	-0.042
	de	$0.624_{0.055}$	$0.589_{0.039}$	-0.035	$0.074_{0.023}$	$0.037_{0.018}$	-0.037
	zh	$0.663_{0.059}$	$0.650_{0.026}$	-0.013	$0.121_{0.065}$	$0.075_{0.040}$	-0.046

Table 8: **Semantic clustering (XSIR) performs poorly on hold-out unsupported languages**. Δ measures the robustness gains against a translation attack on a language after it has been supported by XSIR. The semantic clustering of tokens is applied on all the original five languages of XSIR (en, fr, de, zh, ja) for *supported*, and on all but the held-out language for *held-out*. LLaMA-3.2 1B generates a text in the *Prompt* language, then the translation attack is applied on the held-out language. Red indicates that XSIR performs worst after supporting the held-out language. Other models in Appendix B.1.

New Lang.	X-SIR (↑)		X-KGW (↑)	
	AUC	TPR@1%	AUC	TPR@1%
it	0.796	0.177	0.772	0.238
es	0.754	0.155	0.807	0.230
pt	0.775	0.133	0.792	0.286
pl	0.749	0.127	0.762	0.236
nl	0.776	0.164	0.808	0.314
hr	0.726	0.124	0.757	0.210
cs	0.773	0.111	0.754	0.254
da	0.734	0.161	0.764	0.266
ko	0.729	0.136	0.754	0.226
ar	0.687	0.093	0.765	0.168
Min.	0.687 (ar)	0.093 (ar)	0.754 (ko)	0.168 (ar)

Table 9: **Semantic clustering is weak for unsupported languages**. Watermark strength (AUC and TPR@1%) of X-SIR and X-KGW, limited to the five originally supported languages (en, fr, de, zh, ja). Aya-23 8B generates English text, which is then translated into a new unsupported language for evaluation. Minimum marks the weakest robustness (best attack case).

B.2 Unsupported languages for X-SIR & X-KGW

New Lang.	XSIR		XKGW	
	AUC ↑	TPR@1% ↑	AUC ↑	TPR@1% ↑
it	0.699	0.069	0.760	0.212
es	0.665	0.076	0.744	0.222
pt	0.641	0.059	0.722	0.152
pl	0.679	0.069	0.677	0.144
nl	0.754	0.095	0.781	0.244
hr	0.660	0.066	0.733	0.162
cs	0.650	0.064	0.759	0.190
da	0.675	0.093	0.765	0.196
ko	0.673	0.062	0.672	0.124
ar	0.655	0.055	0.704	0.168
Baseline	0.653	0.050	0.693	0.149
Min.	0.650 (cs)	0.055 (ar)	0.672 (ko)	0.124 (ko)

Table 10: **Semantic clustering performs poorly on new unsupported languages**. The semantic clustering of tokens is applied using the original five supported languages (en, fr, de, zh, ja). LLaMA-3.2 1B generates a text in English, then the translation attack is applied on the *New Language*. Baseline is the average on the original supported languages. Higher values indicate better robustness. Minimum indicates the worst-case robustness, i.e., the best language for an attack.

New Lang.	XSIR		XKGW	
	AUC ↑	TPR@1% ↑	AUC ↑	TPR@1% ↑
it	0.829	0.335	0.860	0.510
es	0.810	0.314	0.864	0.490
pt	0.812	0.337	0.861	0.498
pl	0.812	0.308	0.817	0.420
nl	0.845	0.351	0.896	0.584
hr	0.804	0.279	0.822	0.344
cs	0.798	0.305	0.838	0.444
da	0.832	0.357	0.871	0.436
ko	0.792	0.276	0.800	0.390
ar	0.765	0.251	0.815	0.358
Baseline	0.782	0.280	0.803	0.387
Min.	0.765 (ar)	0.251 (ar)	0.800 (ko)	0.344 (hr)

Table 11: **Semantic clustering performs poorly on new unsupported languages**. The semantic clustering of tokens is applied using the original five supported languages (en, fr, de, zh, ja). LLaMAX-3 8B generates a text in English, then the translation attack is applied on the *New Language*. Baseline is the average on the original supported languages. Higher values indicate better robustness. Minimum indicates the worst-case robustness, i.e., the best language for an attack.

CWRA Attack	Aya-23 8B		LLaMA-3.2 1B		LLaMAX-3 8B	
New Language	AUC ↑	TPR@1% ↑	AUC ↑	TPR@1% ↑	AUC ↑	TPR@1% ↑
it	0.746	0.194	0.855	0.245	0.826	0.313
es	0.751	0.147	0.830	0.217	0.816	0.319
pt	0.781	0.179	0.854	0.244	0.827	0.336
pl	0.793	0.195	0.859	0.289	0.815	0.299
nl	0.836	0.252	0.900	0.375	0.835	0.360
hr	0.810	0.236	0.853	0.269	0.790	0.291
cs	0.785	0.180	0.835	0.201	0.787	0.309
da	0.857	0.247	0.864	0.243	0.830	0.331
ko	0.750	0.157	0.852	0.255	0.809	0.309
ar	0.704	0.209	0.822	0.222	0.771	0.286
Baseline	0.759	0.223	0.866	0.448	0.794	0.351
Min.	0.704 (ar)	0.147 (es)	0.822 (ar)	0.201 (cs)	0.771 (ar)	0.286 (ar)

Table 12: **Semantic clustering (XSIR) performs inconsistently on an expanded set of supported languages**. The semantic clustering is applied using an expanded set of 17 newly supported languages. A prompt in English is first translated into each target language. Aya-23 8B, LLaMA-3.2 1B, and LLaMAX-3 8B are then prompted with the translated input to generate text in the target language. Finally, the CWRA attack is applied by translating the generated text back into English. Baseline is the average on the original supported languages. Higher values indicate better robustness. Minimum indicates the worst-case robustness, i.e., the best language for an attack.

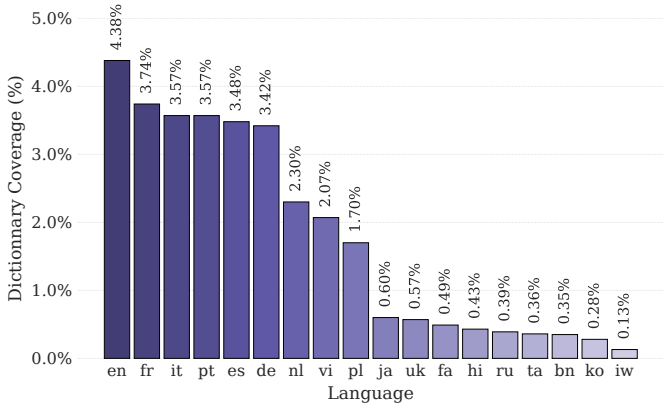


Figure 4: **Tokenizer vocabulary favours high-resource languages**. Percentage of words in multilingual dictionaries that appear in the tokenizer vocabulary.

B.3 Supported languages for X-SIR & X-KGW

Translation Attack		XSIR		XKGW	
Type	Language	AUC ↑	TPR@1% ↑	AUC ↑	TPR@1% ↑
High-resource	fr	0.702	0.085	0.719	0.166
	de	0.708	0.067	0.752	0.186
	it	0.712	0.111	0.750	0.230
	es	0.703	0.089	0.724	0.222
	pt	0.726	0.102	0.747	0.206
Medium-resource	pl	0.657	0.065	0.703	0.188
	nl	0.722	0.091	0.787	0.252
	ru	0.635	0.075	0.656	0.100
	hi	0.611	0.037	0.620	0.084
	ko	0.673	0.055	0.701	0.154
	ja	0.571	0.042	0.598	0.110
Low-resource	bn	0.825	0.509	0.701	0.078
	fa	0.584	0.055	0.673	0.086
	vi	0.691	0.084	0.722	0.186
	iw	0.622	0.026	0.702	0.134
	uk	0.613	0.064	0.725	0.118
	ta	0.749	0.095	0.672	0.108
	Minimum	0.584 (fa)	0.026 (iw)	0.598 (ja)	0.084 (hi)

Table 13: **Semantic clustering performs poorly on an expanded set of supported languages**. The semantic clustering is applied using an expanded set of 17 newly supported languages. LLaMA-3.2 1B generates a text in English, then the translation attack is applied using each of these supported languages as target language. Higher values indicate better robustness. Minimum indicates the worst-case robustness, i.e., the best language for an attack.

B.4 Tokenizer vocabulary analysis

Translation Attack		XSIR		XKGW	
Type	Language	AUC ↑	TPR@1% ↑	AUC ↑	TPR@1% ↑
High-resource	fr	0.804	0.249	0.852	0.466
	de	0.833	0.399	0.850	0.484
	it	0.829	0.336	0.870	0.478
	es	0.811	0.319	0.869	0.506
	pt	0.726	0.338	0.863	0.454
Medium-resource	pl	0.812	0.308	0.847	0.410
	nl	0.847	0.355	0.882	0.592
	ru	0.787	0.256	0.821	0.368
	hi	0.702	0.215	0.714	0.228
	ko	0.792	0.276	0.822	0.422
	ja	0.714	0.187	0.705	0.206
Low-resource	bn	0.588	0.086	0.765	0.244
	fa	0.755	0.268	0.829	0.398
	vi	0.772	0.238	0.802	0.328
	iw	0.719	0.196	0.808	0.444
	uk	0.794	0.309	0.817	0.118
	ta	0.561	0.067	0.789	0.316
	Minimum	0.561 (ta)	0.067 (ta)	0.705 (ja)	0.118 (uk)

Table 14: **Semantic clustering performs poorly on an expanded set of supported languages**. The semantic clustering is applied using an expanded set of 17 newly supported languages. LLaMAX-3 8B generates a text in English, then the translation attack is applied using each of these supported languages as target language. Higher values indicate better robustness. Minimum indicates the worst-case robustness, i.e., the best language for an attack.

CWRA Attack		Aya-23 8B		LLaMA-3.2 1B		LLaMAX-3 8B	
Type	Language	AUC ↑	TPR@1% ↑	AUC ↑	TPR@1% ↑	AUC ↑	TPR@1% ↑
High-resource	fr	0.831	0.201	0.898	0.421	0.845	0.345
	de	0.820	0.198	0.902	0.394	0.841	0.361
	it	0.817	0.196	0.872	0.331	0.825	0.315
	es	0.819	0.200	0.859	0.281	0.816	0.320
	pt	0.801	0.191	0.876	0.348	0.827	0.335
Medium-resource	pl	0.804	0.202	0.881	0.381	0.815	0.291
	nl	0.859	0.265	0.899	0.434	0.834	0.367
	ru	0.769	0.175	0.824	0.226	0.771	0.233
	hi	0.710	0.147	0.771	0.304	0.744	0.263
	ko	0.769	0.178	0.854	0.340	0.805	0.299
	ja	0.787	0.278	0.904	0.644	0.720	0.314
Low-resource	bn	0.721	0.255	0.934	0.628	0.784	0.354
	fa	0.691	0.113	0.796	0.260	0.733	0.246
	vi	0.781	0.180	0.865	0.325	0.805	0.301
	iw	0.726	0.124	0.815	0.207	0.729	0.245
	uk	0.769	0.157	0.849	0.253	0.750	0.225
	ta	0.819	0.405	0.917	0.516	0.740	0.391
	Minimum	0.691 (fa)	0.113 (fa)	0.771 (hi)	0.207 (iw)	0.720 (ja)	0.225 (uk)

Table 15: **Semantic clustering (XSIR) performs inconsistently on an expanded set of supported languages**. The semantic clustering is applied using an expanded set of 17 newly supported languages. A prompt in English is first translated into each target language. Aya-23 8B, LLaMA-3.2 1B, and LLaMAX-3 8B are then prompted with the translated input to generate text in the target language. Finally, the CWRA attack is applied by translating the generated text back into English. Higher values indicate better robustness. Minimum indicates the worst-case robustness, i.e., the best language for an attack.

B.5 STEAM

Translation Attack		AUC (↑)			TPR@1% (↑)		
Type	Language	KGW	XSIR	STEAM	KGW	XSIR	STEAM
High-resource	fr	0.655	0.702	**0.973**	0.052	0.085	**0.770**
	de	0.649	0.708	**0.963**	0.090	0.067	**0.722**
	it	0.617	0.712	**0.972**	0.118	0.111	**0.770**
	es	0.616	0.703	**0.973**	0.122	0.089	**0.774**
	pt	0.651	0.726	**0.981**	0.096	0.102	**0.880**
Medium-resource	pl	0.622	0.657	**0.964**	0.088	0.065	**0.704**
	nl	0.719	0.722	**0.988**	0.126	0.091	**0.850**
	ru	0.629	0.635	**0.934**	0.052	0.075	**0.536**
	hi	0.568	0.611	**0.969**	0.048	0.037	**0.710**
	ko	0.625	0.673	**0.878**	0.068	0.055	**0.330**
	ja	0.578	0.571	**0.938**	0.048	0.042	**0.580**
Low-resource	bn	0.574	0.825	**0.921**	0.020	**0.509**	0.438
	fa	0.586	0.584	**0.940**	0.082	0.055	**0.528**
	vi	0.658	0.691	**0.948**	0.082	0.084	**0.576**
	iw	0.495	0.622	**0.942**	0.042	0.026	**0.688**
	uk	0.629	0.613	**0.953**	0.084	0.064	**0.672**
	ta	0.877	0.749	**0.913**	0.272	0.095	**0.302**

Table 16: **STEAM is consistently better than semantic clustering by a large margin**. Bolded is best. Red indicates that the defence reduces robustness (lower than the undefended KGW baseline).

New Lang.	AUC (↑)			TPR@1% (↑)		
	KGW	XSIR	STEAM	KGW	XSIR	STEAM
it	0.620	0.699	**0.708**	0.108	0.069	0.096
es	0.616	0.665	**0.693**	0.122	0.076	0.082
pt	0.652	0.641	**0.693**	0.096	0.059	**0.096**
pl	0.617	0.679	**0.682**	0.088	0.069	**0.112**
nl	0.714	**0.754**	0.693	0.112	0.095	0.110
hr	0.611	0.660	**0.690**	0.078	0.066	**0.100**
cs	0.655	0.650	**0.681**	0.072	0.064	0.068
da	0.655	0.675	**0.721**	0.080	**0.093**	0.068
ko	0.623	**0.673**	0.670	0.066	0.062	0.064
ar	0.635	0.655	**0.670**	0.110	0.055	0.078

Table 17: **STEAM performs better than XSIR on unsupported languages**. Bolded is best. Red indicates that the defence reduces robustness (lower than the undefended KGW baseline).

Translation Attack		AUC (↑)			TPR@1% (↑)		
Type	Language	Same	Different	Δ	Same	Different	Δ
High-resource	fr	0.966	0.977	+0.011	0.752	0.810	+0.058
	de	0.958	0.960	+0.002	0.684	0.710	+0.026
	it	0.964	0.976	+0.012	0.744	0.840	+0.096
	es	0.965	0.975	+0.010	0.712	0.794	+0.082
	pt	0.979	0.976	-0.003	0.822	0.826	+0.004
Medium-resource	pl	0.939	0.969	+0.030	0.654	0.750	+0.096
	nl	0.983	0.985	+0.002	0.822	0.846	+0.024
	ru	0.934	0.948	+0.014	0.510	0.700	+0.190
	hi	0.949	0.972	+0.023	0.650	0.756	+0.106
	ko	0.834	0.885	+0.051	0.292	0.380	+0.088
	ja	0.888	0.866	-0.022	0.438	0.394	-0.044
Low-resource	bn	0.895	0.936	+0.041	0.402	0.546	+0.144
	fa	0.924	0.943	+0.019	0.526	0.600	+0.074
	vi	0.937	0.949	+0.012	0.610	0.668	+0.058
	iw	0.942	0.965	+0.023	0.560	0.698	+0.138
	uk	0.930	0.959	+0.029	0.530	0.684	+0.154
	ta	0.885	0.919	+0.034	0.414	0.566	+0.152

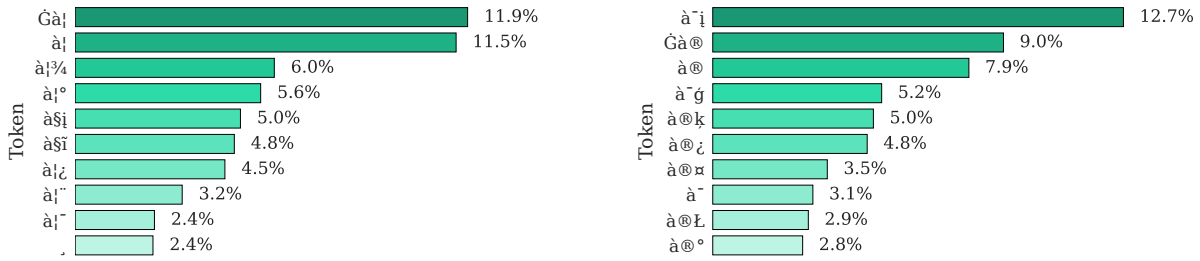
Table 18: **STEAM is robust to a translator mismatch**. AUC and TPR@1% when using Google Translate for both the translation attack and the back-translation defense (*same*) and when using Google Translate for the translation attack and DeepSeek-V3.2-Exp for the back-translation defence (*different*).

Two-Step Translation Attack		STEAM	
Language 1	Language 2	AUC ↑	TPR@1% ↑
High-resource	None	**0.966**	**0.743**
	de	0.909	0.500
	ko	0.807	0.227
	bn	0.847	0.332
Medium-resource	None	**0.936**	**0.561**
	de	0.865	0.357
	ko	0.801	0.228
	bn	0.815	0.278
Low-resource	None	**0.922**	**0.502**
	de	0.844	0.319
	ko	0.794	0.212
	bn	0.813	0.265

Table 19: **STEAM remains robust under multi-step attacks.** Aya-23 8B generates text in English that is translated to the 17 supported languages (*Language 1*). A second translation step is then applied using *Language 2* to compute the AUC and TPR@1%. None indicates the single-step translation baseline.

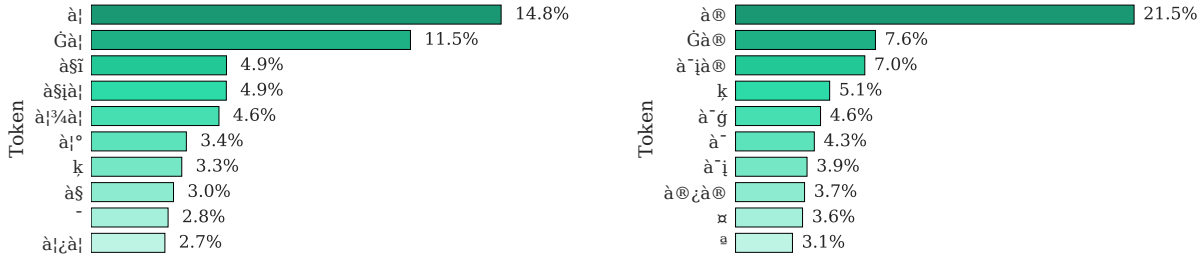
B.6 Sub-character Token Distributions

As discussed in §5.1, the z-score normalization component is designed to calibrate STEAM's detection mechanism against statistical noise introduced by tokenizer limitations. Figures 5 and 6 show the token distribution for two severely affected low-resource languages, Bengali and Tamil.



(a) **bn** (b) **ta**

Figure 5: **Tokenization of low-resource languages creates highly concentrated sub-character tokens.** Percentage of top 10 tokens for Bengali (a) and for Tamil (b) using Aya-23 8B.



(a) **bn** (b) **ta**

Figure 6: **Tokenization of low-resource languages creates highly concentrated sub-character tokens.** Percentage of top 10 tokens for Bengali (a) and for Tamil (b) using LLaMA-3.2 1B.

C Usage of AI Assistants

Throughout this study, we made use of several AI assistants to support different aspects of the research workflow. For idea generation and querying information, we relied on ChatGPT and Gemini Pro. For coding-related tasks, we relied on Claude 4.5 Sonnet alongside GitHub Copilot. We use GPT-5 and Claude for light editing (re-wording, grammar, proof-checking) to help writing the paper. For translation tasks in the experimental setting of §5.3, we use DeepSeek-V3.2-Exp as the translation model.

D Artifacts

D.1 Artifacts License

All datasets, models, and code used in this work comply with their original licenses.

- MUSE Dictionary[5] (Conneau et al., 2017): Released under the Creative Commons Attribution–NonCommercial 4.0 International (CC BY-NC 4.0) license. Use is restricted to non-commercial research and requires attribution to the original authors.
- Aya-23 8B[6]: Released under (CC BY-NC 4.0) license.
- LLaMA-3.2 1B[7]: Released under the LLaMA 3.2 Community License Agreement. This license allows research and educational use but restricts commercial deployment without explicit permission from Meta.
- LLaMAX3 8B[8]: Released under the MIT License, which permits reuse, modification, and redistribution for both commercial and non-commercial purposes, provided that attribution and the original license terms are preserved.
- DeepSeek-V3.2-Exp[9]: Released under the MIT License.
- mC4 Dataset[10] (Raffel et al., 2023): Licensed under the Open Data Commons Attribution License (ODC-BY). This allows redistribution, reuse, and adaptation of the dataset, provided that appropriate credit is given.
- `deep_translator`[11] python package: Released under the MIT License.
- `openai`[12] python package: Released under the Apache License 2.0. This license permits use, modification, and redistribution for both commercial and non-commercial purposes

D.2 Artifact Use Consistent With Intended Use

All datasets and models were used in line with their intended research purposes and licences. We used the mC4 dataset (Raffel et al., 2023) and open multilingual models (Aya-23-8B, LLaMA-3.2-1B, LLaMAX-8B) strictly for evaluation within academic settings. No data or model outputs were used for deployment or commercial applications. Our method STEAM is released only for research use and is compatible with the original access conditions of all components. No personal data were processed.

[5] https://github.com/facebookresearch/MUSE?tab=License-1-ov-file
[6] https://huggingface.co/CohereLabs/aya-23-8B
[7] https://huggingface.co/meta-llama/Llama-3.2-1B/blob/main/LICENSE.txt
[8] https://huggingface.co/LLaMAX/LLaMAX3-8B
[9] https://huggingface.co/deepseek-ai/DeepSeek-V3.2-Exp
[10] https://huggingface.co/datasets/allenai/c4
[11] https://deep-translator.readthedocs.io/en/latest/README.html
[12] https://pypi.org/project/openai/?utm_source=chatgpt.com