Enhancing Cross-Lingual Transfer through Reversible Transliteration: A Huffman-Based Approach for Low-Resource Languages

Anonymous ACL submission

Abstract

As large language models (LLMs) are trained on increasingly diverse and extensive multilingual corpora, they demonstrate cross-lingual transfer capabilities. However, these capabilities often fail to effectively extend to low-resource languages, particularly those utilizing non-Latin scripts. While transliterating low-resource languages into Latin script presents a natural solution, there currently lacks a comprehensive framework for integrating transliteration into LLM training and deployment. Taking a pragmatic approach, this paper innovatively combines character transliteration with Huffman coding to design a complete transliteration framework. Our proposed framework offers the following advantages: 1) Compression: Reduces storage requirements for low-resource language content, achieving file size compression ratios of 0.5 and token count reductions between 60% and 80%. 2) Accuracy: Guarantees 100% lossless conversion from transliterated text back to the source language. 3) Efficiency: Eliminates the need for vocabulary expansion for low-resource languages, improving training and inference efficiency. 4) Scalability: The framework can be extended to other low-resource languages. We validate the effectiveness of our framework across multiple downstream tasks, including text classification, machine reading comprehension, and machine translation. Experimental results demonstrate that our method significantly enhances the model's capability to process low-resource languages while maintaining performance on high-resource languages. Our data and code have been made publicly available.

1 Introduction

Large language models have demonstrated remarkable multilingual transfer capabilities, enabling knowledge transfer from one language to another without additional training (Qi et al., 2023; Gao et al., 2024; Ye et al., 2023). However, this transfer ability often performs poorly in low-resource languages, primarily constrained by three factors: scarcity of training data (Costa-jussà et al., 2022), insufficient cross-lingual word embedding alignment (Deshpande et al., 2021), and writing system differences (Anastasopoulos and Neubig, 2019; Muller et al., 2021).

Common approaches to improving LLMs' adaptability to low-resource languages include continued pre-training and supervised fine-tuning (Tao et al., 2024). Due to the low representation of low-resource languages in tokenizers and frequent occurrence of UNKnown tokens (Moosa et al., 2023), vocabulary expansion becomes a primary task (Zhuang and Sun, 2025). However, ensuring high performance for multiple low-resource languages is extremely challenging, facing two key issues. The first issue is the increased training and inference costs due to vocabulary size, as vocabulary must inevitably expand with the addition of languages to ensure tokenization performance for each language (Purkayastha et al., 2023). The second issue is the curse of multilinguality, which means that using a fixed-capacity model to pretrain multiple languages can improve cross-lingual performance to some extent, but performance begins to decline beyond a certain point (Conneau, 2019).

To improve cross-lingual transfer while avoiding issues associated with extensive vocabulary expansion, transliteration of low-resource languages has emerged as a viable approach. Transliteration refers to the process of converting text from one writing system to another according to specific rules, typically converting non-Latin scripts to Latin alphabet representation (Wellisch, 1978). Previous studies have demonstrated that transliterating text into a common character set can enhance cross-lingual transfer performance for low-resource languages with non-Latin scripts (Liu et al., 2024a, 2025). This improvement is attributed to the common character set facilitating knowledge transfer

through lexical overlap (Dhamecha et al., 2021; Pires, 2019; Amrhein and Sennrich, 2020) and enabling the reuse of existing information in embedding matrices (Purkayastha et al., 2023). Appending transliterated content to prompt templates for low-resource languages has been shown to improve downstream task performance (Ma et al., 2024).

Most research on transliteration relies on existing tools like UROMAN (Hermjakob et al., 2018), which maps any UTF-8 character to Latin letters, to investigate the impact of transliteration on cross-lingual transfer or alignment. However, this transliteration process is irreversible; due to the potential loss of characteristic information from the original script during transliteration, it is impossible to accurately restore the transliterated Latin letters back to the source language script, which limits its practical applications (Amrhein and Sennrich, 2020). Furthermore, low-resource languages typically utilize extended Unicode character sets for encoding, resulting in their textual data occupying more storage space compared to languages like English. This storage overhead issue becomes more prominent when processing large-scale multilingual corpora. Therefore, this paper focuses on two key issues: how to implement a reversible transliteration mechanism to facilitate practical applications while maintaining cross-lingual transfer effectiveness, and how to achieve text compression during the transliteration process to facilitate storage and training. We observe that these two points correspond precisely to the reversibility and compression properties of Huffman coding, which provides the theoretical foundation for our Huffman coding-based transliteration scheme.

We selected three low-resource languages: Tibetan, Mongolian, and Uyghur, which are minority languages in China with a total user base exceeding 30 million speakers, along with English and Chinese as high-resource languages for our experiments. We conducted continued pre-training of open-source LLMs using corpora obtained through various transliteration methods, analyzed cross-lingual transfer performance across text classification, named entity recognition, machine reading comprehension, knowledge extraction, and machine translation tasks, while also comparing compression rates among different transliteration methods. To enable the model to directly serve low-resource language users, we developed a FastText-based automatic transliteration framework that performs language detection before and after model processing, implementing transliteration and restoration of input and output, thereby maintaining native language interaction at the user end. In summary, our contributions are as follows:

- We propose a Huffman coding-based transliteration scheme for low-resource languages, achieving reversibility in the transliteration process and addressing the limitations of traditional transliteration methods in practical applications.
- Leveraging the compression properties of Huffman coding, we effectively reduce the storage overhead of low-resource language texts, making the training of large-scale multilingual corpora more efficient.
- We develop an end-to-end framework integrating FastText language identification, enabling automatic transliteration and restoration of low-resource languages while maintaining native language interaction and improving performance across multiple downstream tasks.

2 Related Works

Cross-lingual Transfer for Low-resource Languages To enhance low-resource language performance in LLMs, one primary approach uses continued pre-training and supervised fine-tuning with low-resource corpora. However, this method faces significant challenges: it requires complex vocabulary expansion and model architecture modifications, resulting in poor scalability, and most critically, suffers from limited training data availability (Joshi et al., 2020).

Alternative approaches focus on improving cross-lingual transfer capabilities through various mechanisms: concatenating multilingual input sequences to leverage shared representation spaces (Kim et al.; Tanwar et al., 2023; Cueva et al., 2024), projecting target language representations onto high-resource languages for enhanced feature extraction (Xu et al., 2023), and increasing the parallel content in multilingual training corpora (Zhuang and Sun, 2025).

While these methods aim to transfer capabilities from resource-rich to low-resource languages, a fundamental challenge remains: the substantial differences in writing systems among low-resource languages. Unifying multiple languages into a single writing system could potentially address vocabulary challenges and promote vocabulary sharing,

thereby facilitating cross-lingual knowledge transfer (Purkayastha et al., 2023).

Tokenization and Vocabulary Expansion Existing subword tokenizers (such as BPE (Sennrich, 2015) and SentencePiece (Kudo, 2018)) have been widely adopted for low-resource languages. However, due to the limited representation of these languages in pre-training corpora, they suffer from insufficient vocabulary coverage, over-segmentation, and high ratios of unknown tokens. While vocabulary expansion (Cui et al., 2023) offers a potential solution, it introduces new challenges: the need for substantial training data to adequately train new tokens, and increased model capacity requirements to mitigate the multilingual curse (Conneau, 2019).

Recent approaches have focused on more efficient solutions, such as leveraging shared linguistic information and cross-lingual word embedding alignment (Ogueji et al., 2021; Liu et al., 2021), which improve tokenization without significant vocabulary expansion. Notably, transliterating low-resource languages into a unified writing system has shown promising results (Dhamecha et al., 2021; Liu et al., 2024b), simultaneously enhancing vocabulary sharing and model transfer capabilities while avoiding the computational overhead of vocabulary expansion.

Romanization and Transliteration Romanization is the process of mapping various characters to Latin characters, though this process is typically irreversible. Its objective is to approximate the pronunciation of the original character text as closely as possible. Specialized tools like UROMAN (Hermjakob et al., 2018) can romanize almost all characters by directly mapping UTF-8 characters to Latin letters, though this process involves information loss, such as the omission of tonal information. There are also general character conversion tools like uconv that can preserve more original character information, such as adding diacritical marks, but this limits subword sharing across languages.

The Tibetan, Mongolian, Uyghur, and Chinese languages used in our experiments can all be romanized through UROMAN; however, due to the uniqueness of the Tibetan writing system, uconv currently cannot transliterate Tibetan. Romanization encoding has been studied in both natural language processing and speech processing domains, such as its application in multilingual pre-trained language models to enhance low-resource languages (Purkayastha et al., 2023), and in speech processing systems' pre-training as additional forced alignment for text labeling (Pratap et al., 2024). Moreover, phonological distinctions may be lost during romanization - for instance, Chinese characters become toneless pinyin when romanized, with a single pinyin potentially corresponding to many different characters. UROMAN also converts numbers from different writing systems into Western Arabic numerals (Ding et al., 2024), which further complicates the process of converting romanized text back to source languages, particularly when users expect LLMs to output in their native writing systems. In contrast, our proposed Huffman coding-based transliteration method is an innovative approach that balances transliteration (improving cross-lingual transfer), compression (reducing storage and training costs), and reversibility (facilitating practical restoration and interaction).

3 Methodology

3.1 Overview

We propose a three-stage processing approach for low-resource language transliteration and applications: (1) character encoding design: analyzing character frequencies and designing custom encodings. (2) transliteration and model training: training on transliterated raw corpora. (3) end-to-end language processing pipeline: comprising input language classification and processing, model inference, and output language classification and processing, as shown in Figure 1.

In this study, we focus on three low-resource languages of China: Tibetan, Uyghur, and Mongolian (see Table 1). These languages represent different writing systems, with a combined user base exceeding 30 million speakers. We select these languages because they present significant challenges for cross-lingual transfer: they use non-Latin scripts, have limited digital resources, and exhibit distinct writing systems that differ substantially from high-resource languages like Chinese and English.

Name	ISO 639-1	Writing System
Tibetan	bo	Tibetan script
Uyghur	ug	Uyghur Arabic script
Mongolian	mn	Traditional Mongolian script

Table 1: Overview of the low-resource languages studied in this work.

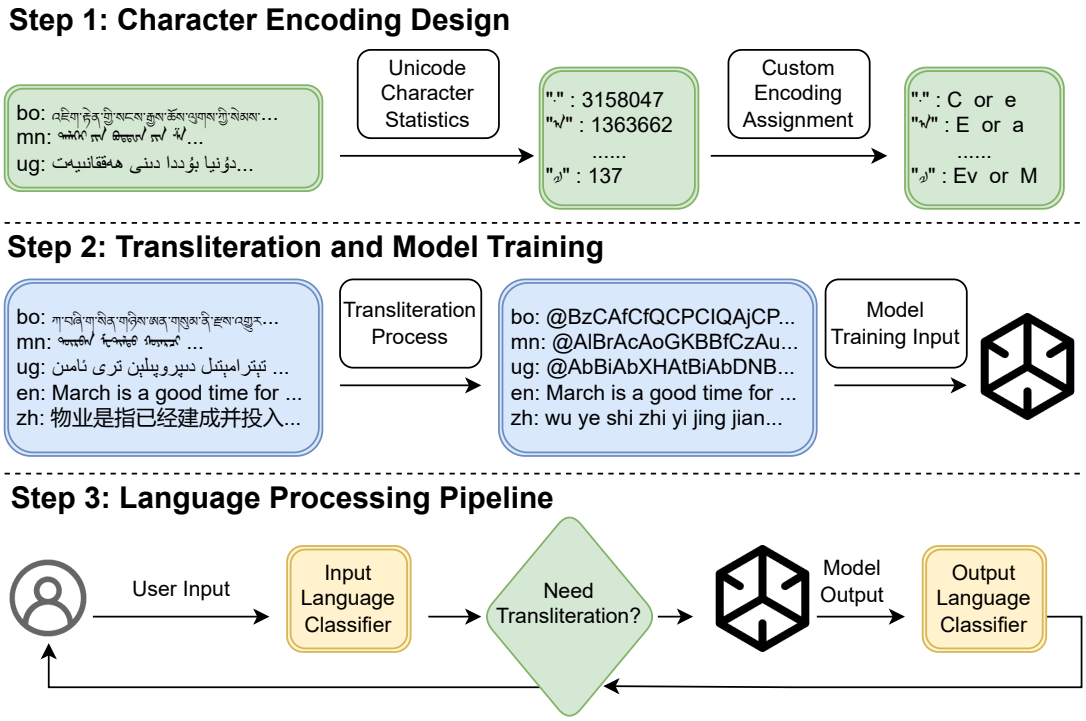

Figure 1: Overview of our three-stage approach. Step 1: Character encoding design with Unicode character statistics and custom encoding assignment. Step 2: Transliteration process for model training input. Step 3: Language processing pipeline with language classification for user interaction.

3.2 Character Frequency Analysis

Since English already uses Latin script and Chinese has well-established romanization tools for converting characters to pinyin, we focused our transliteration efforts on three low-resource languages: Uyghur, Tibetan, and Mongolian. Our character frequency analysis began with the CUTE open-source parallel dataset (Zhuang and Sun, 2025), which provided aligned text across Chinese, Uyghur, Tibetan, and English. To incorporate Mongolian, which wasn't originally included in CUTE, we followed the dataset's methodology to collect and evaluate Mongolian translations, using human evaluators to assess translation quality from Chinese to Mongolian. For comprehensive character analysis, we sampled 3,000 instances from each of the three low-resource languages and conducted a thorough examination of their Unicode characters and frequencies. Through a systematic approach combining Unicode code point ranges, character naming conventions, and expert linguistic validation, we identified the core character sets for each language: 45 characters for Mongolian, 38 for Uyghur, and 81 for Tibetan. These character sets were then consolidated and arranged in descending order of frequency, providing a foundation for our transliteration scheme.

3.3 Huffman-based Encoding Design

Properties of Huffman Coding Huffman coding, as a variable-length encoding method, possesses both variable-length allocation and prefix properties. The variable-length allocation ensures that high-frequency characters receive shorter codes while low-frequency characters receive longer codes, providing a theoretical foundation for our text compression. The prefix property guarantees that no code is a prefix of any other character's code, facilitating unambiguous decoding.

Customized Encoding Scheme Based on the principles of Huffman coding, we designed an improved encoding scheme. To accommodate more languages, our scheme, while not strictly adhering to the prefix property, ensures unambiguous decoding through structured design. Specifically, we constrain all codes to follow the pattern "First letter capitalized, subsequent letters lowercase" and employ a maximum matching strategy for decoding.

Table 2 demonstrates the possibilities of this design pattern at different lengths.

Length	Pattern	Capacity
One	A, B, ..., Z	26
Two	Aa, Ab, ..., Zz	676
Three	Aaa, Aab, ..., Zzz	17,576
Four	Aaaa, Aaab, ..., Zzzz	456,976
Total (up to four characters)		475,254

Table 2: Encoding patterns and theoretical capacity for different lengths. The pattern consists of one uppercase letter followed by zero or more lowercase letters.

This design offers three key advantages: (1) By constraining the first character to be uppercase and subsequent characters to be lowercase, combined with the maximum matching strategy, it ensures unambiguous decoding. (2) It maintains the core principle of Huffman coding, allowing for variable-length code allocation based on character frequencies. (3) It provides significant scalability, theoretically supporting encoding for up to 475,254 characters. In this study, we implemented a subset of the two-character scheme, utilizing 21 single-character codes (B-X) and 141 two-character codes (Aa-Fk), totaling 162 encoding options. This scale is sufficient to cover the character sets of Uyghur, Tibetan, and Mongolian languages.

3.4 Transliteration Strategies

To explore optimal transliteration strategies, we designed three progressive transliteration schemes, with each scheme building upon and improving its predecessor.

Basic Transliteration Strategy We designed a transliteration scheme using "First-letter-capitalized + lowercase" encoding rules. This scheme transliterates Uyghur, Tibetan, and Mongolian into Latin alphabet representations according to encoding rules, while converting Chinese characters into Pinyin and preserving English text unchanged. While this strategy achieved basic transliteration functionality and reversibility, it did not account for the tokenizer's characteristics, leaving room for further token compression.

Tokenizer-based Optimization Strategy To address the token optimization potential in the basic strategy, we analyzed the characteristics of the Llama2 tokenizer. Research showed that 66% of original characters required four tokens for representation. In response, we innovatively utilized single tokens from the Llama2 tokenizer (Touvron et al., 2023) as encoding mappings for original characters, enabling all 162 original characters to be represented by single tokens. The comparative token distribution is shown in Table 3.

Method	1-token	2-token	3-token	4-token
Original	1	45	9	107
Basic	90	72	0	0
Optimized	162	0	0	0

Table 3: Character distribution by token length after Llama2 tokenization. The Optimized method achieves single-token encoding for all characters.

Hybrid Vocabulary Strategy Building upon the second strategy, we leveraged the linguistic patterns inherent in the transliterated text to train a specialized vocabulary of 4,000 tokens. This vocabulary was merged with Llama2's original 32,000-token vocabulary to create a hybrid vocabulary of 33,738 tokens, maintaining efficient single-character encoding while capturing common character combinations. The comparison of the three strategies is presented in Table 4. For a comprehensive analysis of file size and token compression ratios across different languages and strategies, see Appendix B.

Strategy	Vocab Size	Compr.	Cost
Basic	32,000	1.63×	Low
Tokenizer	32,000	2.35×	Medium
Hybrid	33,738	3.04×	High

Table 4: Comparison of different strategies. Compr. shows average token compression ratio across Tibetan, Mongolian and Uyghur languages.

3.5 Reversibility Mechanism

Our transliteration system achieves perfect reversibility through a carefully designed mapping mechanism, maintaining bidirectional mappings between original characters and Latin codes. For characters not in the mapping tables (e.g., emojis, rare characters, or special symbols), the system preserves them using '@' markers with proper escape sequences (e.g., '@@' for the '@' character itself), ensuring no information loss during transliteration. The detailed process is shown in Algorithm 1.

Algorithm 1 Reversible Transliteration System

Input: M_{c2l}, M_{l2c}: Bidirectional mappings, $text$: Input text
Output: Transliterated or restored text

Function ToLatin($text$):
  $result \leftarrow []$
  for each c in $text$ **do**
    Append $M_{c2l}[c]$ if exists, else preserve as @...@
  end for
  return joined result

Function FromLatin($latin\_text$):
  $result \leftarrow [], i \leftarrow 0$
  while $i <$ length($latin\_text$) **do**
    Process @ markers or find longest matching code
    Advance i accordingly
  end while
  return joined result

The system employs a greedy matching strategy during restoration, where it attempts to match the longest possible Latin code sequence for mapped characters while correctly handling preserved sequences between '@' markers. This dual mechanism ensures 100% restoration accuracy by either mapping characters through the bidirectional tables or preserving them in their original form.

3.6 Auxiliary Models for Practical Deployment

To achieve end-to-end system deployment, we developed three auxiliary models. At the input stage, we trained a FastText-based classifier specifically for identifying Mongolian, Tibetan, Uyghur, Chinese, and other languages. At the output stage, we trained a FastText language classifier tailored to the characteristics of transliterated text to guide language restoration. Additionally, to accurately handle the conversion from Chinese pinyin to characters, we fine-tuned a specialized model based on Qwen2.5-0.5B (Yang et al., 2024). The detailed training processes and evaluation results of these models are presented in Appendix A.

4 Experiments and Analysis

To evaluate the effectiveness of different transliteration schemes, we conducted a series of experiments examining the cross-lingual transfer performance of models trained with various transliteration strategies. We specifically focused on whether the models could effectively transfer knowledge to low-resource languages (Tibetan, Mongolian, and Uyghur) while maintaining performance in high-resource languages (Chinese and English).

4.1 Experimental Setup

We adopt the following experimental procedure: First, we process the pre-training corpus using different transliteration methods, followed by continued pre-training of the model. The choice of continued pre-training over training from scratch is motivated by the common challenge of insufficient training data faced by low-resource languages, which makes it difficult to support a complete pre-training process. After pre-training, we perform supervised fine-tuning using downstream task data from high-resource languages, and then directly conduct zero-shot evaluation on low-resource languages to verify the model's cross-lingual transfer capability.

For Tibetan, Uyghur, and Mongolian languages, we identified a limited number of available datasets. Our experiments encompassed three primary tasks: text classification, machine reading comprehension, and translation. These tasks evaluated the model's capabilities across different levels of language processing, thereby enabling a comprehensive assessment of the transliteration scheme's effectiveness. Details regarding the pre-training data and parameter settings can be found in Appendix C.

We designed the following comparative experiments:

- Direct Continued Pre-training: Continuing pre-training on the original model using raw corpora.
- Vocabulary Expansion: Augmenting the original model's vocabulary with dedicated lexicons for each low-resource language.
- UROMAN Transliteration: Applying universal romanization tools for transliteration.
- Three Progressive Transliteration Strategies: As detailed in Section 3.4.

4.2 Text Classification

We first evaluated the effectiveness of various transliteration schemes on the text classification task. The experiments utilized the WCM-v2 dataset

Model	Low-resource Languages (Acc / F1)			Chinese		Average	
	bo	mn	ug	Acc	F1	Minorities	All
Base Llama2	28.65 / 21.23	1.78 / 1.65	73.33 / 74.69	86.12	85.91	13.48 / 9.01	48.15 / 48.78
Expanded Vocab	53.96 / 51.69	64.45 / 67.95	76.00 / 82.91	89.95	89.91	62.58 / 64.56	75.64 / 76.34
UROMAN	49.37 / 48.88	64.92 / 67.78	**81.33 / 86.18**	89.82	89.75	62.10 / 63.47	75.33 / 75.84
Basic Trans.	52.16 / 52.18	**66.46 / 69.55**	74.67 / 81.51	89.92	89.81	**63.40 / 65.17**	**76.06 / 76.60**
Token-Opt Trans.	**54.14 / 54.94**	61.25 / 64.61	81.00 / 85.31	**90.15**	**90.07**	60.80 / 63.54	74.81 / 75.86
Hybrid Trans.	50.45 / 52.58	61.25 / 65.15	65.33 / 75.59	90.00	89.95	58.80 / 62.15	73.68 / 75.02

Table 5: Performance comparison on the WCM-v2 dataset. The best scores are in **bold**, with the second best underlined. Base Llama2: directly fine-tuned on original texts; Expanded Vocab: vocabulary expansion for each low-resource language; Basic/Token-Opt/Hybrid Trans.: three progressive transliteration strategies. Minorities average is calculated as the mean of scores for low-resource languages.

Model	CMRC-Trained				SQuAD-Trained			
	Chinese		Tibetan		English		Tibetan	
	EM	F1	EM	F1	EM	F1	EM	F1
Base Llama2	77.2	89.5	7.9	45.8	89.5	95.3	6.5	50.8
Expanded Vocab	81.3	91.1	11.5	50.6	**89.9**	**95.7**	10.3	58.9
UROMAN	79.6	88.3	12.0	53.4	84.2	88.7	11.0	61.1
Basic Trans.	87.7	92.7	15.5	59.5	87.7	89.1	12.7	65.1
Token-Opt Trans.	**88.4**	**93.6**	**16.0**	**60.2**	88.0	89.3	**13.5**	**66.5**
Hybrid Trans.	83.1	90.2	14.8	58.8	87.2	89.0	12.3	64.9

Table 6: Machine Reading Comprehension performance comparison. The best scores are in **bold**, with the second best underlined. Results show both source language (Chinese/English) and target language (Tibetan) performance under different training settings. EM: Exact Match score; F1: F1 score.

(see Appendix D.1), a classification dataset encompassing multiple ethnic minority languages of China (Yang et al., 2022). This dataset maintains balanced distributions across both categories and languages, containing texts from 10 domains including arts, geography, and history. The experimental results are shown in Table 5. To comprehensively evaluate the effectiveness of each approach, we focus not only on the overall performance but also specifically on the average performance across low-resource languages.

4.3 Machine Reading Comprehension

For the machine reading comprehension task, we evaluate the models' performance by fine-tuning them on the Chinese CMRC dataset (Cui et al., 2019) and English SQuAD dataset (Rajpurkar, 2016), followed by zero-shot testing on the TibetanQA dataset (Sun et al., 2021) to assess their cross-lingual transfer capabilities. We also report the performance on the source languages to verify that our approaches maintain strong performance on high-resource languages while enabling effective cross-lingual transfer. The results are shown in Table 6. For detailed information about the datasets, please refer to Appendix D.2.

4.4 Machine Translation

To evaluate the models' machine translation capabilities for low-resource languages, we conduct experiments on Chinese-to-Tibetan (zh-bo) and Chinese-to-Uyghur (zh-ug) translation tasks using the Flores-200 dataset (Costa-jussà et al., 2022). We employ few-shot prompting with three carefully selected examples for each language pair, ensuring the examples cover diverse linguistic patterns. The evaluation uses three standard metrics: BLEU score for overall translation quality, chrF for character-level accuracy, and Translation Edit Rate (TER) for measuring the amount of editing required to match the reference translation. Table 7 presents the results of our comparative evaluation. For detailed information about the dataset and prompts used, please refer to Appendix E.

4.5 Overall Analysis

Cross-task Performance Analysis Through a comparative analysis of experimental results across text classification, machine reading comprehension, and machine translation tasks, our proposed transliteration approach demonstrated excellent cross-lingual transfer capabilities. In text classification tasks, the basic transliteration strategy

Model	Chinese-to-Tibetan (zh-bo)			Chinese-to-Uyghur (zh-ug)		
	BLEU↑	chrF↑	TER↓	BLEU↑	chrF↑	TER↓
Base Llama2	3.5	0.28	0.92	4.2	0.31	0.89
Expanded Vocab	5.0	0.35	0.86	5.7	0.38	0.83
UROMAN	4.5	0.33	0.88	5.2	0.36	0.85
Basic Trans.	5.7	0.37	0.84	6.4	0.40	0.81
Token-Opt Trans.	**6.3**	**0.39**	**0.82**	**7.0**	**0.42**	**0.79**
Hybrid Trans.	3.8	0.30	0.90	4.5	0.33	0.87

Table 7: Machine Translation performance comparison on Flores-200 dataset using few-shot prompting (3 examples). ↑: higher is better, ↓: lower is better. The best scores are in **bold**, with the second best underlined. TER: Translation Edit Rate.

achieved an average accuracy of 63.40% on low-resource languages, showing an improvement of 0.82% compared to the vocabulary expansion approach. For machine reading comprehension tasks, the tokenizer-optimized transliteration strategy achieved an exact match score of 16.0% on Chinese-to-Tibetan transfer, outperforming the vocabulary expansion approach by 4.5%. This strategy also exhibited superior performance in translation tasks, achieving a BLEU score of 6.3 in Chinese-to-Tibetan translation. Notably, these improvements were achieved while maintaining high performance on resource-rich languages, as exemplified by our approach achieving 90.15% accuracy on Chinese text classification tasks.

Key Findings Our experimental results yield three significant findings:

- **Performance and Efficiency:** Our transliteration approaches consistently outperformed traditional vocabulary expansion methods across tasks, with both basic and tokenizer-optimized strategies showing exceptional results. By leveraging existing tokenizer characteristics, these approaches significantly improved low-resource language processing without vocabulary expansion.
- **Untapped Potential:** Despite using simple frequency-based encoding schemes (B-X, Aa-Fk) and random token assignments, our methods demonstrated remarkable effectiveness. This suggests substantial room for improvement through the incorporation of linguistic features and more sophisticated encoding strategies.
- **Scalable Framework:** Our findings establish a new paradigm for low-resource language processing, offering a more promising direction than vocabulary expansion. The success of this relatively simple implementation particularly demonstrates its potential for scaling to multiple low-resource languages.

These results not only validate our approach but also indicate that more sophisticated versions of these strategies could yield even more significant improvements in low-resource language processing.

5 Conclusion

In this paper, we introduce a novel Huffman-based transliteration framework that addresses three critical challenges in low-resource language processing: cross-lingual transfer, storage efficiency, and practical deployment. Our framework demonstrates superior performance across diverse tasks while maintaining a lightweight implementation. The basic and tokenizer-optimized strategies consistently outperform traditional approaches, achieving up to 4.5% improvement in cross-lingual machine reading comprehension and significant gains in translation tasks, all while preserving performance on high-resource languages. Beyond performance gains, our approach offers unique advantages in compression efficiency, reducing both file size and token count by 2-3 times without sacrificing reversibility. Most importantly, our framework's success with simple frequency-based encoding suggests substantial potential for improvement through the incorporation of linguistic features and more sophisticated encoding strategies. These findings establish a promising direction for scaling language technologies to the world's many low-resource languages, offering a more practical alternative to the traditional vocabulary expansion paradigm.

Limitations

While our approach demonstrates promising results, there are several important limitations to consider. Our current evaluation scope is restricted to three low-resource languages with non-Latin scripts. Although the framework is theoretically extensible to other writing systems, specific adaptations may be necessary to accommodate their unique characteristics. The limited availability of evaluation datasets for low-resource languages also poses a challenge, particularly in tasks like machine reading comprehension, where we could only assess performance on a subset of languages.

From a practical perspective, our approach faces a trade-off between storage efficiency and computational overhead. While we achieve significant reductions in storage requirements, the transliteration and restoration processes introduce additional computational steps that could impact real-time performance, especially in scenarios requiring frequent language switching. Furthermore, our current encoding scheme relies primarily on character frequency, leaving room for potential improvements through the incorporation of linguistic features such as phonemes and morphological information.

References

Chantal Amrhein and Rico Sennrich. 2020. On romanization for model transfer between scripts in neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2461–2469.

Antonios Anastasopoulos and Graham Neubig. 2019. Pushing the limits of low-resource morphological inflection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 984–996.

A Conneau. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Emilio Cueva, Adrian López-Monroy, Fernando Sánchez Vega, and Thamar Solorio. 2024. Adaptive cross-lingual text classification through in-context one-shot demonstrations. 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 8317–8335.

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2019. A span-extraction dataset for chinese machine reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5883–5889.

Yiming Cui, Ziqing Yang, and Xin Yao. 2023. Efficient and effective text encoding for chinese llama and alpaca. *arXiv preprint arXiv:2304.08177*.

Ameet Deshpande, Partha Talukdar, and Karthik Narasimhan. 2021. When is bert multilingual? isolating crucial ingredients for cross-lingual transfer. *arXiv preprint arXiv:2110.14782*.

Tejas Dhamecha, Rudra Murthy, Samarth Bharadwaj, Karthik Sankaranarayanan, and Pushpak Bhattacharyya. 2021. Role of language relatedness in multilingual fine-tuning of language models: A case study in indo-aryan languages. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8584–8595.

Wen Ding, Fei Jia, Hainan Xu, Yu Xi, Junjie Lai, and Boris Ginsburg. 2024. Romanization encoding for multilingual asr. *arXiv preprint arXiv:2407.04368*.

Changjiang Gao, Hongda Hu, Peng Hu, Jiajun Chen, Jixing Li, and Shujian Huang. 2024. Multilingual pretraining and instruction tuning improve cross-lingual knowledge alignment, but only shallowly. 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 6101–6117.

Ulf Hermjakob, Jonathan May, and Kevin Knight. 2018. Out-of-the-box universal romanization tool uroman. In *Proceedings of ACL 2018, system demonstrations*, pages 13–18.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the nlp world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293.

Sunkyoung Kim, Dayeon Ki, Yireun Kim, and Jinsik Lee. Translating qa is enough: A key to unlocking in-context cross-lingual performance. In *ICML 2024 Workshop on In-Context Learning*.

T Kudo. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Linlin Liu, Bosheng Ding, Lidong Bing, Shafiq Joty, Luo Si, and Chunyan Miao. 2021. Mulda: A multilingual data augmentation framework for low-resource cross-lingual ner. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5834–5846.

Yihong Liu, Chunlan Ma, Haotian Ye, and Hinrich Schütze. 2024a. Translico: A contrastive learning framework to address the script barrier in multilingual pretrained language models. *arXiv preprint arXiv:2401.06620*.

Yihong Liu, Chunlan Ma, Haotian Ye, and Hinrich Schütze. 2024b. Transmi: A framework to create strong baselines from multilingual pretrained language models for transliterated data. *arXiv preprint arXiv:2405.09913*.

Yihong Liu, Mingyang Wang, Amir Hossein Kargaran, Ayyoob ImaniGooghari, Orgest Xhelili, Haotian Ye, Chunlan Ma, François Yvon, and Hinrich Schütze. 2025. How transliterations improve crosslingual alignment. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2417–2433.

Chunlan Ma, Yihong Liu, Haotian Ye, and Hinrich Schütze. 2024. Exploring the role of transliteration in in-context learning for low-resource languages written in non-latin scripts. *arXiv preprint arXiv:2407.02320*.

Ibraheem Muhammad Moosa, Mahmud Elahi Akhter, and Ashfia Binte Habib. 2023. Does transliteration help multilingual language modeling? In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 670–685.

Benjamin Muller, Antonios Anastasopoulos, Benoît Sagot, and Djamé Seddah. 2021. When being unseen from mbert is just the beginning: Handling new languages with multilingual language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 448–462.

Kelechi Ogueji, Yuxin Zhu, and Jimmy Lin. 2021. Small data? no problem! exploring the viability of pretrained multilingual language models for low-resourced languages. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 116–126.

T Pires. 2019. How multilingual is multilingual bert. *arXiv preprint arXiv:1906.01502*.

Vineel Pratap, Andros Tjandra, Bowen Shi, Paden Tomasello, Arun Babu, Sayani Kundu, Ali Elkahky, Zhaoheng Ni, Apoorv Vyas, Maryam Fazel-Zarandi, et al. 2024. Scaling speech technology to 1,000+ languages. *Journal of Machine Learning Research*, 25(97):1–52.

Sukannya Purkayastha, Sebastian Ruder, Jonas Pfeiffer, Iryna Gurevych, and Ivan Vulić. 2023. Romanization-based large-scale adaptation of multilingual language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7996–8005.

Jirui Qi, Raquel Fernández, and Arianna Bisazza. 2023. Cross-lingual consistency of factual knowledge in multilingual language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10650–10666.

P Rajpurkar. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Rico Sennrich. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Y Sun, S Liu, C Chen, Z Dan, and X Zhao. 2021. Construction of high-quality tibetan dataset for machine reading comprehension. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 208–218.

Eshaan Tanwar, Subhabrata Dutta, Manish Borthakur, and Tanmoy Chakraborty. 2023. Multilingual llms are better cross-lingual in-context learners with alignment. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6292–6307.

Mingxu Tao, Chen Zhang, Quzhe Huang, Tianyao Ma, Songfang Huang, Dongyan Zhao, and Yansong Feng. 2024. Unlocking the potential of model merging for low-resource languages. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8705–8720.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Hans H Wellisch. 1978. The conversion of scripts-its nature, history, and utilization.

Shaoyang Xu, Junzhuo Li, and Deyi Xiong. 2023. Language representation projection: Can we transfer factual knowledge across languages in multilingual language models? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3692–3702.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Ziqing Yang, Zihang Xu, Yiming Cui, Baoxin Wang, Min Lin, Dayong Wu, and Zhigang Chen. 2022. Cino: A chinese minority pre-trained language model.

In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3937–3949.

Jiacheng Ye, Xijia Tao, and Lingpeng Kong. 2023. Language versatilists vs. specialists: An empirical revisiting on multilingual transfer ability. *arXiv preprint arXiv:2306.06688*.

Wenhao Zhuang and Yuan Sun. 2025. Cute: A multilingual dataset for enhancing cross-lingual knowledge transfer in low-resource languages. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10037–10046.

A Auxiliary Models

To achieve a comprehensive end-to-end system, we developed three auxiliary models for input language identification, output language identification, and Chinese pinyin conversion. These models collectively form a complete language processing pipeline, ensuring that the system can accurately process multilingual inputs and generate appropriate outputs.

A.1 Input Language Classifier

We trained specialized language identification models based on the FastText framework to accurately identify the language type of input text. The classifier supports five language categories: Tibetan (bo), Mongolian (mn), Uyghur (ug), Chinese (zh), and other languages (other). The training data was sourced from multiple datasets, including CUTE (Zhuang and Sun, 2025), WCM-v2 (Yang et al., 2022), and other open-source datasets, to ensure the model can process text from various domains. The training parameters for the input language classifier are shown in Table 9.

The evaluation results on the test set of 5,000 entries show that the classifier achieved a high level of classification across all languages and can be used for actual classification needs. The evaluation results are shown in Table 8.

Language	Precision	Recall	F1
Tibetan	0.992	0.989	0.991
Mongolian	0.987	0.985	0.986
Uyghur	0.995	0.993	0.994
Chinese	0.998	0.997	0.998
Other	0.981	0.978	0.980

Table 8: Performance of the input language classifier on various languages.

A.2 Transliterated Text Classifier

To accurately identify transliterated text in model outputs and guide proper language restoration, we trained a specialized FastText classifier. The distinguishing feature of this classifier lies in its need to process transliterated text; therefore, we utilized parallel corpora of transliterated text for training, ensuring the model could recognize textual features under different transliteration strategies. The training parameters for the output language classifier are shown in Table 9.

The classification performance on the transliterated text test set is shown in Table 10.

Language	Precision	Recall	F1
Tibetan	0.988	0.985	0.987
Mongolian	0.983	0.981	0.982
Uyghur	0.991	0.989	0.990
Chinese	0.995	0.994	0.995
Other	0.992	0.990	0.991

Table 10: Performance of the transliteration text classifier in various languages.

A.3 Pinyin-to-Chinese Converter

For Chinese pinyin conversion, we performed task-specific fine-tuning based on the Qwen2.5-0.5B model (Yang et al., 2024). This model handles the conversion from pinyin sequences to Chinese characters, which is a typical sequence-to-sequence conversion task. We used approximately 1 million pinyin-character pairs for training, with data sourced from news texts, Wikipedia, and general domain texts. The parameters used for fine-tuning are shown in Table 11.

Parameter	Value
Batch Size	128
Learning Rate	2e-5
Max Length	2048
Epochs	3
Warmup Steps	1000
Weight Decay	0.01

Table 11: Qwen2.5-0.5B fine-tuning parameter settings.

The performance evaluation of the model on the test set is shown in Table 12.

These three auxiliary models collectively form a complete language processing pipeline, capable of accurately identifying input languages, processing transliterated text, and converting pinyin to Chinese characters when needed. In practical applications, these models have demonstrated stable performance and high accuracy.

B Compression Analysis

To comprehensively evaluate different transliteration approaches, we compare our three strategies with two baselines: vocabulary expansion (adding 6,000 tokens for each low-resource language) and UROMAN (a widely-used romaniza-

Parameter	Input Classifier	Output Classifier	Note
Learning Rate	0.1	0.05	Initial learning rate
Epochs	25	30	Training epochs
Word n-grams	2	3	Maximum length of word n-gram
Vector Dimension	100	150	Embedding dimension
Context Window	5	7	Size of context window
Min Word Count	5	3	Minimum word frequency

Table 9: Training parameters for FastText language classifiers. The input classifier is optimized for raw text classification, while the output classifier is specifically tuned for transliterated text patterns with slightly different hyperparameters.

Metric	Value	Note
Character Accuracy	0.975	Single character accuracy
Sentence Accuracy	0.892	Complete sentence accuracy
BLEU Score	96.8	Overall translation quality
Inference Speed	125ms/sent	Average processing time

Table 12: Evaluation of Pinyin Conversion Model Performance. The model shows strong performance in character-level accuracy and complete sentence conversion, with reasonable inference speed suitable for real-time applications.

tion tool). Table 13 presents the compression performance across different approaches and languages.

Method	Lang	File Compr.	Token Compr.
Vocab Expansion	bo	1.00×	6.92×
	mn	1.00×	9.66×
	ug	1.00×	4.49×
	zh	1.00×	1.75×
UROMAN	bo	2.07×	1.88×
	mn	2.41×	5.16×
	ug	1.78×	2.46×
	zh	1.00×	1.12×
Basic	bo	1.98×	1.33×
	mn	1.95×	2.57×
	ug	1.25×	1.00×
	zh	0.73×	0.90×
Tokenizer	bo	2.61×	1.80×
	mn	2.07×	3.85×
	ug	1.27×	1.39×
	zh	0.73×	0.90×
Hybrid	bo	2.61×	2.23×
	mn	2.07×	4.98×
	ug	1.27×	1.92×
	zh	0.73×	1.41×

Table 13: Compression performance across different approaches and languages. File Compr. shows the ratio of original file size to transliterated file size. Token Compr. indicates the ratio of original token count to transliterated token count using Llama2 tokenizer. Language codes: bo (Tibetan), mn (Mongolian), ug (Uyghur), zh (Chinese).

The vocabulary expansion approach achieves the highest token compression ratios but maintains original file sizes. UROMAN demonstrates good compression performance in both file size and token count. Our proposed methods show progressive improvements from Basic to Hybrid strategies, with the Hybrid approach achieving competitive token compression while maintaining strong file size reduction. Note that Chinese (zh) shows different patterns due to its unique characteristics in tokenization and encoding.

C Training Details

C.1 Pre-training Data

The statistics of the raw corpora used for pre-training are shown in Table 14. The data is primarily sourced from the CUTE parallel corpus (Zhuang and Sun, 2025), which provides high-quality aligned multilingual content across Chinese, Uyghur, Tibetan, and English languages. For Mongolian, we follow the data collection and quality assessment methodology described in the CUTE paper to ensure comparable data quality and distribution.

Language	Lines	Size (GB)
Tibetan (bo)	934,140	11.22
Mongolian (mn)	933,941	11.48
Uyghur (ug)	934,002	7.37
Chinese (zh)	933,946	2.54
English (en)	933,989	3.60

Table 14: Pre-training Corpora Statistics. The data is primarily sourced from the CUTE parallel corpus, with additional Mongolian data collected following similar quality standards.

C.2 Training Parameter Settings

Table 15 lists the main parameter settings for the pre-training and supervised fine-tuning phases.

D Dataset Details

D.1 WCM-v2 Dataset

WCM-v2 is a multilingual text classification dataset covering 10 domains including arts, geography, and history (Yang et al., 2022). The dataset is characterized by its balanced distribution across both categories and languages, containing Chinese training sets and test sets in multiple languages. Table 16 shows the sample distribution of each language across different categories.

D.2 Machine Reading Comprehension Datasets

We utilize three machine reading comprehension (MRC) datasets for evaluation. Table 17 shows the key statistics of these datasets.

Dataset	Train	Dev	Test
SQuAD v1.1	87,599	10,570	-
CMRC 2018	10,142	3,219	1,002
TibetanQA	-	-	2,007

Table 17: Statistics of machine reading comprehension datasets used in our experiments. TibetanQA is used only for testing cross-lingual transfer capability.

SQuAD The Stanford Question Answering Dataset (SQuAD) v1.1 (Rajpurkar, 2016) is a widely used English reading comprehension dataset containing over 100,000 question-answer pairs. The questions and answers were created by crowdworkers based on Wikipedia articles, with answers being continuous spans from the corresponding reading passages.

CMRC The Chinese Machine Reading Comprehension (CMRC) 2018 dataset (Cui et al., 2019) follows a similar format to SQuAD, featuring span-extraction questions in Chinese. The dataset covers various domains, making it suitable for evaluating Chinese reading comprehension capabilities.

TibetanQA TibetanQA (Sun et al., 2021) is a Tibetan machine reading comprehension dataset, with 2,007 publicly released question-answer pairs for evaluation. While the full dataset contains 20,000 question-answer pairs annotated from articles on Tibetan web resources, only a portion is publicly available and used in our experiments for zero-shot cross-lingual evaluation.

Note on Language Coverage While our study aims to evaluate cross-lingual transfer across multiple low-resource languages, we were unable to identify suitable machine reading comprehension datasets for Mongolian and Uyghur languages at the time of our research. This limitation highlights the scarcity of evaluation resources for these languages in certain NLP tasks.

E Translation Details

Flores-200 Dataset The Flores-200 dataset is a multilingual benchmark for evaluating machine translation systems, encompassing 200 languages (Costa-jussà et al., 2022). The sentences in the dataset are derived from English Wikipedia articles and professionally translated into other languages. We utilize both the development and test sets, which contain 997 and 1,012 samples per language pair, respectively. To ensure fair evaluation, we conduct our experiments exclusively on the test set.

Translation Prompts We employ English as the unified instruction language for translation tasks. Each prompt contains three carefully selected translation examples (3-shot), with low-resource language text appearing only in the source-target translation pairs. Specifically, the prompt template follows this structure:

(1) An instruction header:

```
Translate the following Chinese text
        to {target_language}
```

(2) Three example translation pairs, each formatted as:

Hyperparameter	Pre-training	Fine-tuning
Learning Rate	1.0e-4	2.0e-5
Training Epochs	1.0	3.0
Global Batch Size	1024	256
Max Sequence Length	4096	4096
Warmup Ratio	0.05	0.05
Data Type	BF16	BF16
LR Scheduler	Cosine	Cosine

Table 15: Hyperparameter settings for pre-training and supervised fine-tuning phases. During the pre-training phase, except for vocabulary expansion, we observed frequent loss spike phenomena when the learning rate was set to 2.0e-4. After reducing it to 1.0e-4, the training process became more stable.

Category	mn	bo	ug	zh-train	zh-test
Arts	135	141	3	2,657	335
Geography	76	339	256	12,854	1,644
History	66	111	0	1,771	248
Nature	7	0	7	1,105	110
Natural Science	779	133	20	2,314	287
People	1,402	111	0	7,706	924
Technology	191	163	8	1,184	152
Education	6	1	0	936	118
Economy	205	0	0	922	109
Health	106	111	6	551	73
Total	2,973	1,110	300	32,000	3,995

Table 16: Sample distribution across categories and languages in the WCM-v2 dataset. The dataset contains training and test sets for Chinese (zh), and test sets for ethnic minority languages (mn: Mongolian, bo: Tibetan, ug: Uyghur). Additional test sets for Korean, Kazakh, and Kyrgyz are also available in the dataset but not used in our experiments.

```
Chinese: [source text]
{target_language}: [translation]
```

(3) The translation request:

```
Now translate this:
Chinese: [input text]
```

This design is motivated by two key considerations: First, utilizing English as the instruction language leverages the model's strong capabilities in English; Second, by minimizing the presence of low-resource languages in the prompt, we can better evaluate the model's genuine translation capabilities rather than simple pattern matching. The three examples are selected to cover diverse sentence structures and vocabulary complexity, helping the model understand the requirements of the translation task.