RoMantra: Optimizing Neural Machine Translation for Low-Resource Languages through Romanization

Govind Soni , **Pushpak Bhattacharyya**
Computation for Indian Langauge Technology (CFILT)
Indian Institute of Technology Bombay, Mumbai, India.
`(govindsoni, pb)@cse.iitb.ac.in`

Abstract

Neural Machine Translation (NMT) for low-resource language pairs with distinct scripts, such as Hindi-Chinese and Hindi-Japanese, poses significant challenges due to scriptural and linguistic differences. This paper investigates the efficacy of romanization as a preprocessing step to bridge these gaps. We compare baseline models trained on native scripts with models incorporating romanization in three configurations: *both-sides*, *source-side only*, and *target-side only*. Additionally, we introduce a script restoration model that converts romanized output back to native scripts, ensuring accurate evaluation. Our experiments show that romanization, particularly when applied to both sides, improves translation quality across the studied language pairs. The script restoration model further enhances the practicality of this approach by enabling evaluation in native scripts with some performance loss. This work provides insights into leveraging romanization for NMT in low-resource, cross-script settings, presenting a promising direction for under-researched language combinations.

1 Introduction

Machine Translation (MT) between languages with distinct writing systems, such as Hindi, Chinese, and Japanese, poses significant challenges due to scriptural and linguistic differences. Traditional MT methods, particularly statistical approaches, struggle with these complexities due to limited data availability, syntax, morphology variations, and script diversity (Koehn, 2009). Although NMT has brought advancements through deep learning models (Bahdanau et al., 2014; Luong et al., 2014), translating between languages like Hindi, Chinese, and Japanese remains largely underexplored. This challenge is especially acute for Hindi-Chinese translation, where no publicly available parallel datasets or NMT models exist, while Japanese-Hindi translation has more resources, though still limited.

A core challenge in translating between these languages arises from their linguistic differences. Chinese is an isolating language that relies heavily on word order (SVO) to convey meaning, while Hindi is a highly inflectional language that uses case endings and verb inflections, allowing for flexible word order (SOV). For instance:

मुझे पानी चाहिए (mujhe pani chahiye) "I need water."

The equivalent Chinese sentence is:

我需要水 (Wǒ xūyào shuǐ) "I need water."

In Chinese, strict word order is required, while in Hindi, case inflections allow for flexibility. This disparity complicates direct translation, as the rigid structure of Chinese contrasts with Hindi's morphological richness. Furthermore, while large corpora exist for Chinese and Japanese in other contexts, these languages are considered low-resource when paired with Hindi due to the lack of publicly available parallel datasets.

We hypothesize that romanization as a preprocessing step can enhance translation performance for low-resource language pairs with distinct scripts, such as Hindi-Chinese and Hindi-Japanese. Romanization converts non-Roman scripts into a unified Roman script, mitigating scriptural differences and enabling more efficient model training. Additionally, we propose a neural script restoration model to revert romanized text back to its original script, preserving crucial linguistic and semantic features.

In this paper, we introduce a NMT framework that incorporates romanization for both source and target text, along with a neural script restoration model to ensure accurate evaluation in native scripts. Our approach optimizes the handling of script diversity and improves translation quality for language pairs like Hindi-Chinese and Hindi-Japanese.

Our contributions are:

1. A NMT framework for Hindi-Chinese and Hindi-Japanese language pairs, utilizing romanization. This is the first work targeting NMT for the Hindi-Chinese language pair.

2. A neural script restoration model that converts romanized text back into native scripts, facilitating evaluation in original writing systems.

3. Systematic evaluation demonstrating significant improvements in translation quality, offering a solution for low-resource, cross-script translation tasks.

The following sections elaborate on related work, our methodology, experimental setup, and results, demonstrating the effectiveness of our approach.

2 Related Work

Machine translation (MT) has a long history, evolving from statistical approaches to the more recent neural-based methods. Early machine translation systems, such as Statistical Machine Translation (SMT), heavily relied on rule-based methods and parallel corpora to achieve reasonable performance (Koehn, 2009; Brown et al., 1990). However, SMT systems faced significant challenges, particularly for low-resource language pairs, which often suffer from data scarcity and script disparity issues (Koehn, 2005; Koehn and Knowles, 2017). The advent of Neural Machine Translation (NMT) significantly improved translation quality, with architectures like sequence-to-sequence models (Cho et al., 2014) and attention mechanisms (Bahdanau et al., 2014; Vaswani et al., 2017). These advancements allowed NMT systems to handle complex linguistic patterns more effectively, especially in high-resource languages. Yet, for low-resource language pairs with distinct scripts, NMT still struggles, particularly when translating between languages such as Chinese and Hindi, which feature vastly different writing systems and linguistic structures (Zoph et al., 2016; Lakew et al., 2019; Wang et al., 2018). Romanization has emerged as a technique to bridge the gap between languages with disparate scripts, converting non-roman scripts into Latin characters for more unified processing. Romanization is particularly effective in low-resource settings, as it allows for leveraging existing tools designed for Roman-script languages (Gheini and May, 2019; Hermjakob et al., 2018). This method facilitates transfer learning between different scripts, enabling NMT systems to share embeddings across languages (Conneau et al., 2018; Artetxe and Schwenk, 2019). However, romanization can introduce issues, including loss of tonal and phonetic information, which is critical in languages like Chinese and Japanese (Lakew et al., 2020). In the context of Chinese and Japanese, various methods have been explored to address these challenges. StrokeNet, for example, converts Chinese characters into Latinized stroke sequences, allowing subword learning in NMT systems (Li et al., 2019). Similarly, Wubi encoding breaks Chinese characters into component radicals, facilitating character-level translation in NMT models (Stahlberg, 2020). These techniques have shown promising results but require significant adaptation for different language pairs, particularly for languages with highly distinct phonological structures, such as Hindi. Our work extends these efforts by exploring romanization on both source and target languages, focusing on translation tasks between Japanese-Hindi (Ja-Hi) and Chinese-Hindi (Zh-Hi) pairs. Prior research often focused on source-side romanization (Wang et al., 2020), but our approach also incorporates *target-side romanization*, with the addition of a neural model for restoring romanized output to its original script post-translation. This ensures the retention of native linguistic characteristics and improves the overall quality of the translated output.

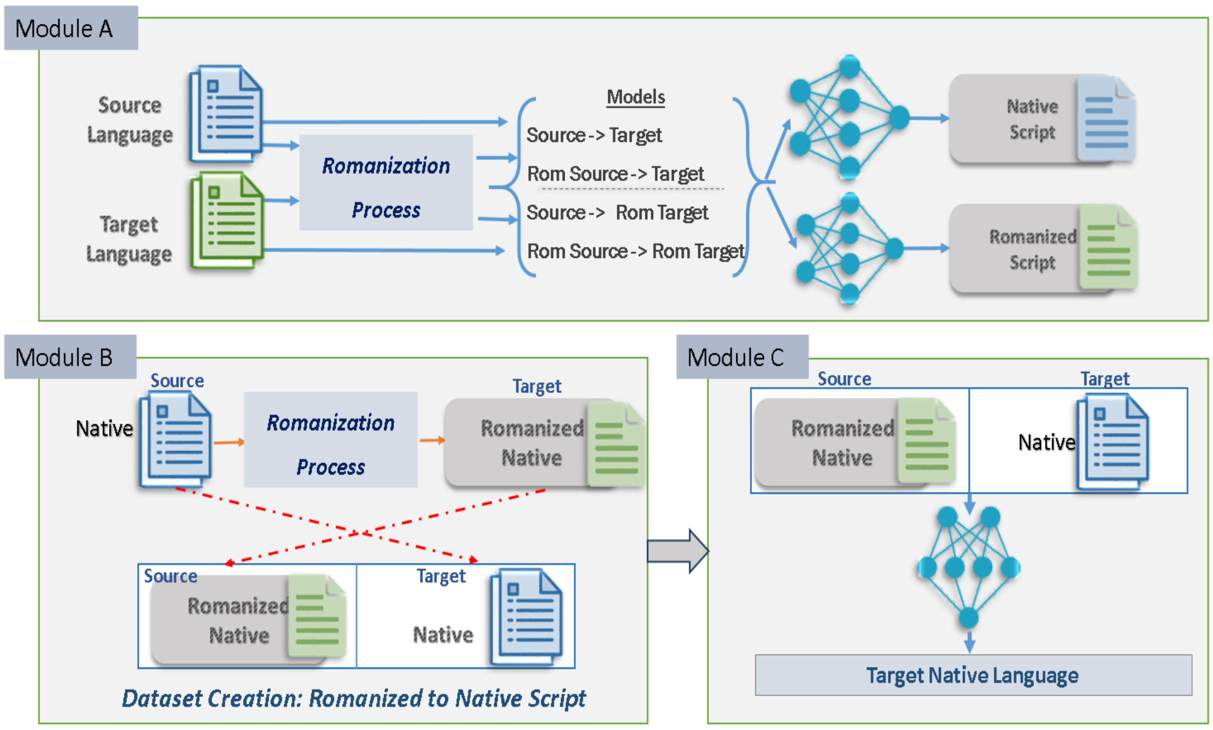

Figure 1: Overview of the proposed romanization-based NMT framework, consisting of multiple modules that handle romanization, dataset creation, and script restoration.

3 Methodology

In this section, we describe the framework of our approach for Neural Machine Translation (NMT) between low-resource languages with distinct scripts. Our proposed framework involves three key components: romanization, translation, and script restoration, as illustrated in Figure 1. Romanization is applied to simplify script processing, followed by translation using different combinations of Romanized and native scripts. The final component involves script restoration, where romanized outputs are converted back to their original native script to ensure proper evaluation and usage.

3.1 Romanization and Tool Selection

The effectiveness of romanization tools varies depending on how much linguistic detail they preserve. We employ two tools discussed below and selected based on the specific needs of the languages involved:

uroman[1] : A universal, unidirectional tool that converts text from most character sets into Latin script. It simplifies text by ignoring tonal marks and diacritics, making it ideal for languages where these features are less crucial. However, it cannot reverse the process, limiting its use in tasks requiring bidirectional script conversion(Hermjakob et al., 2018).

uconv[2] : A bidirectional tool capable of converting between Latin script and several other scripts. It retains more nuanced linguistic features, such as tonal marks in Chinese and long vowels in Japanese, making it suitable for languages where phonetic detail is critical. However, it lacks support for converting these scripts back into Latin for some languages. The romanization process, as shown in Figure 1 (Module A), is applied in different configurations: *both-source-and-target*, *source-side only*, and *target-side only* romanization.

3.2 Why Romanization Over Script Conversion ?

We opted for Romanization over converting texts into Devanagari, Hanzi, or Kanji for two main reasons:

Cross-Language Consistency: Romanization provides a standardized script

[1]https://github.com/isi-nlp/uroman

[2]https://github.com/kevinboone/uconv

representation across linguistically diverse languages. This consistency simplifies preprocessing, facilitates uniform model training, and reduces the complexity of handling multiple native script conversions.

Resource Availability: Tools such as `uroman` and `uconv` are optimized for romanized text and readily available. In contrast, tools supporting scripts like Devanagari or Hanzi are more limited. Romanization thus offers a more practical and resource-efficient solution. Figure 1 (Module B) shows how romanization simplifies dataset creation, allowing the use of both Romanized and native text for training and evaluation.

3.3 Application to Language Pairs

Chinese and Japanese: We use `uconv` for Chinese and Japanese due to its ability to retain tonal and orthographic information, which are crucial in these languages. Example for Chinese:

Original sentence	北京的天气非常好
uroman	beijing de tianqi feichang hao
uconv	běijīng de tiānqì fēicháng hǎo
Translation	बीजिंग का मौसम बहुत अच्छा है।

`uroman` collapses tonal distinctions (e.g., `tianqi`), leading to ambiguity and loss of information. In contrast, `uconv` retains tonal marks (e.g., `tiānqì`), which are essential for accurate interpretation. Example for Japanese:

Original sentence	今日の天[F]はとてもいいです
uroman	kyou no tenki wa totemo ii desu
uconv	kyō no tenki wa totemo ii desu
Translation	आज का मौसम बहुत अच्छा है।

`uconv` captures these distinctions with diacritics, while `uroman` does not, leading to a loss of important phonetic detail. Japanese uses long vowels (`kyō` vs. `kyou`) that change word meanings.

Hindi: For Hindi, we utilize `uroman` because it efficiently converts the script without retaining tonal information, which is not used in Hindi to distinguish meanings. Example for Hindi:

Original sentence	मौसम बहुत अच्छा है।
uroman	mausam bahut accha hai
uconv	mausam bahut accā hai

`uconv` adds diacritics (`accā`), this information does not enhance translation quality for Hindi, where tonal distinctions are not relevant. Therefore, `uroman` is preferred as it provides a simpler and equally effective representation.

3.4 Script Restoration

Script restoration involves converting romanized text back to its original native script, a crucial step for preserving linguistic integrity.

Challenge: Romanization tools such as `uconv` (for Chinese and Japanese) and `uroman` (for Hindi) are commonly used to convert text into a Latin-based script. However, these tools do not support reverse conversion. To address this, we developed a specialized script restoration model trained on pairs of romanized and native script texts.

Solution: Given a romanized sentence $R = \{r_1, r_2, \ldots, r_n\}$ where r_i represents the romanized tokens, and a corresponding native script sentence $S = \{s_1, s_2, \ldots, s_n\}$, where s_i represents tokens in the original script, our goal is to learn a mapping function $f : R \to S$ as shown in Equation 1:

$$S = f(R) \tag{1}$$

This function is learned by training on a parallel corpus of Romanized and native script pairs. The training objective minimizes the loss between the predicted native script sentence $\hat{S}$ and the ground truth native script sentence S. We use a standard cross-entropy loss shown in Equation 2:

$$\mathcal{L} = -\sum_{i=1}^{n} s_i \log P(\hat{s_i} \mid r_1, r_2, \ldots, r_n) \tag{2}$$

where $P(\hat{s_i})$ is the predicted probability of the i-th token in the native script, given the romanized sequence.

Training Data: The monolingual training data used for script restoration was sourced from the same dataset mentioned in Section 4.1. We applied `uconv` for Chinese and

Japanese and uroman for Hindi to convert sentences into their Romanized forms. The model was trained on aligned pairs (R, S), where R is the romanized sentence and S is the corresponding native script sentence.

This approach allows the model to learn to reverse the Romanization process accurately, ensuring that native linguistic characteristics are preserved in the restored text. The restoration process is particularly crucial when romanization is applied to both sides of the translation, as it converts romanized outputs back into their native scripts for evaluation and further use, as seen in Figure 1 (Module C).

4 Experimental Setup

4.1 Dataset

We utilized approximately 3.8M parallel sentences curated from proprietary sources for the Zh-Hi and Hi-Zh pairs. This dataset is the first of its kind, specifically created for Zh-Hi translation for the both directions, making it a novel resource for this language pair. The dataset spans multiple domains, including news, technical manuals, and conversational texts, ensuring diversity in linguistic structures and vocabulary. In contrast, the Ja-Hi and Hi-Ja datasets, comprising 4M sentence pairs, are openly available and were sourced from OPUS[3], including TED Talks, MultiCCAligned, and the NLLB corpus. Detailed statistics, including token counts and sentence length distributions, are provided in the Appendix. We used the FLORES(Costa-jussà et al., 2022) test set for evaluation, which contains 1,012 sentence pairs for both Zh-Hi and Ja-Hi. The validation set consisted of 2,000 sentence pairs for Zh-Hi and Ja-Hi. All data splits were carefully curated to avoid overlap between the training, validation, and test sets. Chinese text was segmented using *Jieba*, and Japanese was processed with *MeCab*. Hindi text was tokenized and normalized using the *IndicNLP toolkit*[4]. We applied *Byte Pair Encoding (BPE)* with 8K, 16K, and 32K merge operations, selecting the optimal configuration based on validation performance. Romanization was applied using the *uroman* and *uconv* tools, which convert scripts into a Latin-based representation.

4.2 Romanization Scenarios

To examine the effect of Romanization, we defined the following scenarios:

1. **Both source and target romanized**: Romanization was applied to both source and target texts.

2. **Source-side romanized only**: The source text was romanized, while the target remained in its original script.

3. **Target-side romanized only**: The source text remained in its original script, while only the target text was romanized.

For comparison, we also trained baseline models on the original, non-romanized scripts.

4.3 Model Architecture

We utilized the OpenNMT[5] framework to train Transformers architecture(Vaswani et al., 2017) with 8 encoder and 8 decoder layers. The hidden size was set to 512, with a feed-forward network size of 1024. We used 16 attention heads per layer, and dropout was applied at 0.2 for both regular and attention dropouts to prevent overfitting. The models were trained for 300K steps using the *Adam optimizer* with $\beta_1 = 0.9$, $\beta_2 = 0.998$, and a learning rate 1e-4. A *Noam learning rate schedule* with 5,000 warm-up steps was used. The maximum batch size was 4096 tokens, with gradient accumulation over 4 steps, resulting in an effective batch size of 128. Early stopping was employed after 5 consecutive validations without improvement.

4.4 Evaluation Metrics

We evaluated model performance using three metrics: BLEU, chrF, and COMET. BLEU (Papineni et al., 2002) measures n-gram precision between machine-generated and reference translations and is widely used for evaluating syntactic accuracy, particularly in distant language pairs like Ja-Hi and Zh-Hi. chrF

[3]https://opus.nlpl.eu/

[4]https://github.com/anoopkunchukuttan/indic_nlp_library

[5]https://github.com/OpenNMT/OpenNMT-py

With Romanization									
Model	both-sides (Romanized)	both-sides (Native)			Source-side			Target-side	
	BLEU	BLEU	chrF	COMET	BLEU	chrF	COMET	BLEU	chrF	COMET
Hi-Zh	22.2	12.6	21.9	91.4	13.5	21.7	90.1	12.3	20.5	89.6
Zh-Hi	17.0	15.7	42.1	69.3	15.6	43.2	67.9	13.8	42.3	66.5
Ja-Hi	16.7	16.1	49.5	71.1	14.1	40.5	58.2	16.3	48.8	73.2
Hi-Ja	26.1	22.9	29.7	95.0	22.7	29.8	93.5	22.6	29.2	94.8

Table 1: Performance of translation models with romanization for language pairs, showing BLEU, chrF, and COMET scores across *both-sides*, *source-side*, and *target-side* romanized models.

(Popović, 2016) focuses on character-level precision and recall, making it well-suited for morphologically rich languages or those with different scripts, such as Chinese and Japanese. COMET (Rei et al., 2020), a neural-based metric, evaluates semantic similarity between translations and references and correlates well with human judgment. For inference, we used a beam size of 5 and a length penalty of 1.0. Romanized outputs were restored to their original scripts using the neural script restoration model, and scores were computed using the sacreBLEU and chrF toolkits.

4.5 Baseline Comparisons

Our baseline model consists of a direct translation model trained on original, non-romanized scripts. This model provides the primary point of comparison to assess the performance gains brought by romanization approaches. In addition to comparing romanized models (*source-side*, *target-side*, and *both-sides*) with the baseline, we also integrated a script restoration model for experiments involving *both-sides* and *target-side romanization*. Since romanizing both sides or just the target side generates romanized outputs, the restoration model is applied to convert these romanized outputs back to the native script, ensuring proper evaluation against the baseline in native scripts.

5 Results & Discussions

In this section, we present the results of our experiments, comparing the performance of baseline models (*w/o romanization*) and models with romanization applied on *both sides*, *source-side* only, and *target-side* only. We also analyze the impact of different Byte Pair Encoding (BPE) merge operations and evaluate the performance of our script restoration model. These results aim to answer our core hypothesis: Can Romanization improve translation performance for distant low-resource language pairs with distinct scripts?

W/o Romanization			
Model	Metrics		
	BLEU	chrF	COMET
Hi-Zh	12.5	21.5	90.0
Zh-Hi	14.7	42.9	67.3
Ja-Hi	14.6	41.8	67.8
Hi-Ja	22.5	29.6	94.9

Table 2: Performance evaluation of translation models *w/o romanization* across language pairs, showcasing BLEU, chrF, and COMET metrics.

5.1 Effect of Romanization on Translation Performance

To evaluate the impact of Romanization, we first trained baseline models without any script modification, meaning that the original scripts of both source and target languages were kept intact. These models were then compared with models where romanization was applied to both sides, the source side only and the target side only. The BLEU, chrF, and COMET scores for these experiments are presented in Tables 1 and 2.

As seen in Table 1, romanization consistently improves translation performance across all language pairs when compared to the baseline models trained on native scripts (see Table 2). For example, in the Hi-Zh language pair, the baseline model achieved a BLEU score of 12.5, while the model with both sides romanized achieved a BLEU score of 22.2 on romanized output and 12.6 after script restoration to native text. For Zh-Hi, the romanized model yielded 17.0 BLEU on romanized output and 15.7 after restoring to the native script, surpassing the baseline score of 14.7.

Table 2 highlights the baseline model performance, showing significantly lower BLEU scores than the Romanized models. The baseline Zh-Hi model achieved 14.7 BLEU, whereas the romanized model delivered 17.0 BLEU in the romanized script and 15.7 in the native script after restoration. This performance gap illustrates how Romanization can reduce the complexity of cross-script translations and better align the representations for low-resource language pairs. For the Ja-Hi and Hi-Ja language pairs, the results continued to demonstrate the effectiveness of romanization, as observed in other language pairs. Specifically, for Ja-Hi, the model with both-sides romanization achieved a BLEU score of 16.7 on the romanized output and 16.1 after script restoration to the native script. This shows a marked improvement over the baseline BLEU score of 14.6. Similarly, for Hi-Ja, the both-sides romanization model reached a BLEU score of 26.1 on the romanized output, which slightly dropped to 22.9 after script restoration, still significantly outperforming the baseline score of 22.5.

The consistent improvement observed with Romanized models shows that Romanization helps manage the challenges posed by diverse scripts, especially for languages with low parallel data availability. Romanization helps reduce the complexity of handling multiple scripts and enhances the ability to transfer learning between languages that are scripturally distinct, thereby improving translation quality.

5.2 Impact of BPE Merge Operations

To further improve translation performance, we experimented with different BPE merge operations (8K, 16K, and 32K) and evaluated their impact on the translation results. As shown in Table 3 and Figure 2, different BPE merge sizes produced varying results, depending on the language pair. For Hi-Zh, the best BLEU score was obtained with a 32K BPE merge operation, reaching 22.2 on the romanized output, while the Ja-Hi and Hi-Ja pairs achieved optimal results with 16K merges. These findings show that larger BPE merges better handle long and rare words, especially in script-dense languages like Chinese, whereas smaller BPE merges (16K) strike a balance between over- and under-segmentation for morphologically rich languages like Hindi.

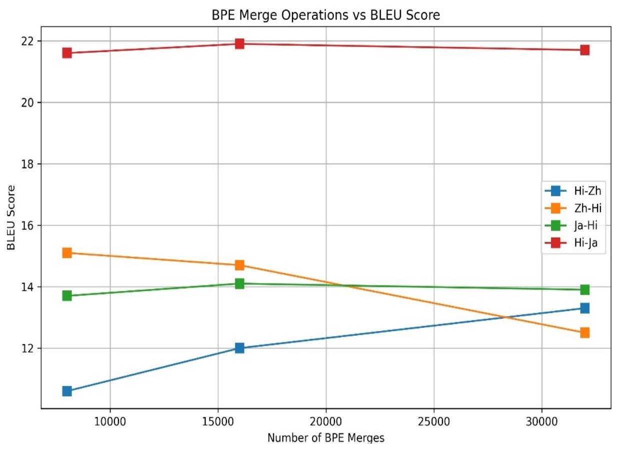

Figure 2: BLEU on the different numbers of BPE merge operations.

	BPE		
Model	8K	16K	32K
Hi-Zh	-	-	✓
Zh-Hi	✓	-	-
Ja-Hi	-	✓	-
Hi-Ja	-	✓	-

Table 3: BPE Merge Operation.

5.3 Script Restoration Module

The script restoration model trained using the same architecture as described in Section 4.3, was evaluated on its ability to convert romanized text back into the native script. This step is crucial for ensuring the final translation output retains its original linguistic integrity. The model' s performance was assessed using BLEU and chrF scores, shown in Table 4. The results indicate that the model is highly effective across languages, with Hindi achieving 85.6 BLEU and 95.8 chrF and Japanese scoring 95.9 BLEU and 97.5 chrF. While the Chinese model shows a slightly lower performance (76.1 BLEU), the overall scores demonstrate that the restoration model reliably preserves the quality of the native script.

	Trained			
	uconv		uroman	
	BLEU	chrF	BLEU	chrF
Hi	-	-	85.6	95.8
Zh	76.1	83.7	-	-
Ja	95.9	97.5	-	-

Table 4: BLEU and chrF scores for Romanized-to-Native script conversion using trained models for languages.

These results validate the model's capability to accurately restore native scripts from romanized text, ensuring high fidelity in the final output with minimal performance drop compared to the romanized results.

5.4 Qualitative Evaluation

We conducted a qualitative evaluation to compare the performance of models trained with and *w/o romanization* across three language pairs: Zh-Hi, Ja-Hi, and Hi-Zh. The evaluation focused on short and long sentences to assess how well different models captured linguistic nuances, particularly in complex sentence structures, named entities, and technical terms.

The results show that the *both-sides* romanization model consistently outperforms the *w/o romanization*, *source-side*, and *target-side* models. In particular, the *both-sides* model better preserves context, formal communication, and key entities, delivering more accurate translations in cases involving technical terminology (e.g., "WiFi Doorbell"). The *w/o romanization* model frequently misinterprets or loses important details, while the *source-side* and *target-side* models perform relatively well but still introduce subtle errors in specific contexts.

Detailed examples and further analysis can be found in the Appendix.

5.5 Challenges and Limitations

While our approach demonstrates improvements in translation quality through Romanization and script restoration, several challenges and limitations must be acknowledged:

Model Constraints: Our proposed NMT framework involves additional processing steps, such as romanization and script restoration, which can increase the computational complexity. These additional steps may introduce overhead during training and inference, especially when applied to large-scale datasets. Furthermore, the dataset size for certain language pairs remains a limitation, particularly for the Zh-Hi pair, where parallel data is scarce.

Generalization to Other Language Pairs: While our model demonstrates success with the specific language pairs studied, it remains uncertain how well the approach generalizes to other low-resource languages with different linguistic structures or scriptural complexities.

6 Conclusion and Future Work

This paper introduced an approach to Neural Machine Translation (NMT) for low-resource language pairs like Hindi-Chinese (Hi-Zh) and Hindi-Japanese (Hi-Ja) by employing romanization as a preprocessing step, simplifying script processing and enhancing translation quality. Integrating a neural script restoration model ensured accurate conversion of romanized outputs back to their original scripts, preserving semantic integrity. Our experiments showed significant improvements in BLEU and chrF scores, demonstrating the effectiveness of this approach for languages with distinct scripts and limited parallel resources. Future work will focus on refining romanization to handle script-specific ambiguities, improving the script restoration model, expanding the framework to more language pairs, and incorporating advanced pre-trained language models to enhance translation accuracy and scalability.

References

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R. Bowman, Holger

Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations.

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*.

Mozhdeh Gheini and Jonathan May. 2019. A universal parent model for low-resource neural machine translation transfer. *arXiv preprint arXiv:1909.06516*.

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, Melbourne, Australia. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Surafel M. Lakew, Alina Karakanta, Marcello Federico, Matteo Negri, and Marco Turchi. 2019. Adapting multilingual neural machine translation to unseen languages. In *Proceedings of the 16th International Conference on Spoken Language Translation*, Hong Kong. Association for Computational Linguistics.

Surafel M Lakew, Matteo Negri, and Marco Turchi. 2020. Low resource neural machine translation: A benchmark for five african languages. *arXiv preprint arXiv:2003.14402*.

Bei Li, Yinqiao Li, Chen Xu, Ye Lin, Jiqiang Liu, Hui Liu, Ziyang Wang, Yuhao Zhang, Nuo Xu, Zeyang Wang, et al. 2019. The niutrans machine translation systems for wmt19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 257–266.

Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2014. Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Maja Popović. 2016. chrf deconstructed: beta parameters and n-gram weights. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 499–504.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Felix Stahlberg. 2020. Neural machine translation: A review. *Journal of Artificial Intelligence Research*, 69:343–418.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Na Wang, Haoliang Wang, Stefano Petrangeli, Viswanathan Swaminathan, Fei Li, and Songqing Chen. 2020. Towards field-of-view prediction for augmented reality applications on mobile devices. In *Proceedings of the 12th ACM International Workshop on Immersive Mixed and Virtual Environment Systems*, pages 13–18.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 856–861, Brussels, Belgium. Association for Computational Linguistics.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

Appendix

A Appendix

A.1 Dataset Statistics

The datasets show significant variation in sentence lengths and structures. For example, Chinese sentences are generally longer, with an average length of 25.3 tokens in the Hi-Zh dataset. Hindi sentences, on the other hand, are even longer, averaging 31.71 tokens, due to their rich morphology and grammatical complexity. Japanese sentences are shorter, with an average length of 17.09 tokens, reflecting their compact, agglutinative structure. These differences highlight the diverse linguistic challenges posed by the datasets.

Table 1 provides detailed statistics for the datasets, including total token counts, average and median sentence lengths, and the distribution of sentences across predefined length buckets (1-10 tokens, 11-20 tokens, 21-30 tokens, and >30 tokens). These insights demonstrate the structural diversity and complexity of the datasets.

A.2 Computational Overhead Analysis

We evaluate the computational overhead introduced by Romanization in Neural Machine Translation (NMT) by comparing the training and inference times of Romanized models (using both Romanized and native script outputs) with a baseline model (using native script only). Our analysis, based on the FLORES test set, reveals that training with romanized data increases computational time for Hi-Zh and Zh-Hi by +89% and +247%, respectively, while the overhead is minimal for Hi-Ja (+1.3%) and slightly reduced for Ja-Hi (-0.9%).

Training time is influenced by various hyperparameter settings, including batch size, learning rate, and the number of training steps. However, a longer training duration does not necessarily correlate with better model performance, and vice versa. These results are summarized in Table 2, which presents the training and inference time differences between the romanized and baseline models. Despite these computational costs, the romanized models yield significant improvements in translation quality, with BLEU scores increasing from 12.5 to 22.2 for Hi-Zh, 14.7 to 17.0 for Zh-Hi, and 22.5 to 26.1 for Hi-Ja. These results show that the trade-off in computational cost is justified by the gains in translation quality. Future work will focus on optimizing the script restoration process to further reduce inference time.

A.3 Qualitative analysis

In Table 3, we present several examples from our test dataset that encompass short and long sentences across three language pairs. These examples are drawn from various models: *w/o romanization*, *both-sides romanization*, *source-side romanization*, and *target-side romanization*. The focus of this qualitative analysis is to highlight the performance differences among these models.

Zh-Hi Translation Example In Example 1, the source sentence describes a message exchange with a friendly response. The *w/o romanization* and *source-side* models produce sentences that deviate from the reference. The *w/o romanization* translation incorrectly translates "we are not doing anything" into "we are keeping silent," which changes the tone of the sentence. The *source-side* model misinterprets "currently" as "under the ice," further distorting the meaning. In contrast, the *both-sides romanization* model generates a translation much closer to the reference. The sentence "मैंने अपने साथ काम करने वाले लोगों को ईमेल भेजे हैं, और एक दोस्ताना जवाब मिला है" directly matches the reference in both content and tone. This shows that *both-sides romanization* captures nuance better than the other models, preserving context and meaning, making it a stronger translation system for this example.

Ja-Hi Translation Example In Example 2, the *w/o romanization* model struggles with word choices and structure, producing "हमारे पास चार महीने का माउस एक बार मधुमेह था" which reads unnaturally. It also omits some key details like "presently" (वर्तमान में). On the other hand, the *source-side romanization* model slightly improves by introducing more accurate verb forms and a clearer structure, but it still reads stiffly. The *both-sides romanization* model, however, produces a more refined translation that preserves context and key details: "हमारे पास पहले से मौजूद चार महीने के चूहों में मधुमेह था, लेकिन वर्तमान में मधुमेह नहीं है." This output more closely mirrors the reference, suggesting that *both-sides romanization*

	Ja-Hi		Zh-Hi	
Language	**Japanese**	**Hindi**	**Chinese**	**Hindi**
Total Tokens	69140954	50146691	100041611	125407098
Mean Length	17.09	12.39	25.3	31.71
Median Length	11	9	26	33
1-10 Tokens (%)	45.69	60.56	9.25	5.25
11-20 Tokens (%)	30.38	24.84	18.1	11.09
21-30 Tokens (%)	11.12	6.46	44.77	22.95
>30 Tokens (%)	12.81	8.14	27.89	60.71

Table 1: Sentence length statistics for Ja-Hi and Zh-Hi datasets, including token counts and length distributions.

Language Pair	Training Time		Inference Time		
	With Romanization	**W/o Romanization**	**Roman+Native**	**Native-only**	**Training+Inference Overhead**
Hi-Zh	357,555 sec	189,235 sec	4m12.583s	1m43.924s	+89% for training and +85% for inference
Zh-Hi	258,213 sec	74,280 sec	5m47.575s	2m58.649s	+247% for training and +94% for inference
Hi-Ja	865,910 sec	855,165 sec	5m23.575s	4m14.752s	+1.3% for training and +27% for inference
Ja-Hi	750,229 sec	756,897 sec	4m40.228s	2m10.028s	-0.9% for training and +116% for inference

Table 2: Comparison of training and inference times with and without romanization across four language pairs, where Roman+Native refers to results with romanization applied and Native-only represents the baseline model trained on native scripts.

handles complex sentence structures and medical terminology more effectively than the other models.

Hi-Zh Translation Example In the Hi-Zh example, which revolves around the creation of a "WiFi Doorbell," the *w/o romanization* model mistranslates "WiFi Doorbell" as "柳条 钟" (willow clock), which is entirely incorrect. It also inaccurately transliterates the name **"जेमी सिमिनॉफ़"** as "杰米 · 西米夫." The *target-side romanization* model suffers from a similar issue, translating "Wifi Doorbell" as "无线电 门铃" (radio doorbell), which introduces a subtle but significant error. Both the *both-sides* and *source-side romanization* models, however, generate the accurate translation "WiFi 门铃," maintaining both the meaning and romanization of key entities like **"जेमी सिमिनॉफ़"**.

Across these examples, the *both-sides romanization* model consistently outperforms the *w/o romanization*, *source-side*, and *target-side* models, especially when it comes to preserving named entities, complex sentence structures, and nuanced contexts. In particular:

both-sides Romanization delivers more accurate translations in cases involving formal communication and technical terminology (e.g., WiFi Doorbell, medical conditions).

Without Romanization model often struggles with nuance, mistranslating key phrases and changing the intended meaning.

Source-Side Romanization and Target-Side Romanization perform relatively well but tend to either miss small details or introduce subtle errors in specific contexts. These findings suggest that *both-sides romanization* is a more reliable approach, especially for handling complex or contextually rich sentences, making it a valuable model in both short and long-sentence translation tasks.

In Example 4, the Romanized model struggles to preserve certain important elements in the text, replacing meaningful terms like "802.11n" and "2.4 GHz" with irrelevant or nonsensical phrases such as **"दक्षिणमानकों"** This issue arises from ambiguities introduced during romanization, where converting scripts simplifies inputs but may obscure key details or alter the intended meaning. Furthermore, romanization can sometimes fail to capture the context accurately, especially in sentences requiring precise interpretation. This highlights a trade-off in the approach: while romanization helps address script diversity, it can occasionally compromise the clarity and accuracy of specific information.

Translation Type		Example 1
Zh-Hi	Source Sentence	达尼 厄斯 说道:“目前 我们 保持 按兵不动 不 动。 我 给 他 关系 最好 的 合作 合作者 作者 打 过 过电 电话 并发 发送 了 电子 电子邮件 邮件,而且 收到 了 对方 非常 友好 的 回复。就 目前 而言,这 足够 了。”
	Reference	डेनियस ने कहा , ’फ़िलहाल हम कुछ नहीं कर रहे हैं . मैंने अपने साथ काम करने वाले लोगों को ईमेल भेजे हैं और उसने उनके फ्रेंडली जवाब भेजे हैं . अभी के लिए , बस इतना ही बताना चाहूँगा .
	w/o Romanization	डेनिहर ने कहा, ’इस समय हम चुप रहते हैं... मैंने उनके रिश्ते के सबसे अच्छे सहयोगी लेखक को फोन किया है और इलेक्ट्रॉनिक ईमेल भेजे हैं और दूसरे पक्ष से बहुत दोस्ताना प्रतिक्रिया प्राप्त की है, जो इस समय पर्याप्त है।
	Both_side	डेनियस ने कहा , ’फ़िलहाल हम कुछ नहीं कर रहे हैं... मैंने अपने साथ काम करने वाले लोगों को ईमेल भेजे हैं, और एक दोस्ताना जवाब मिला है, जो फिलहाल पर्याप्त है।
	Source_Side	डेनियस ने कहा, ’इस समय हम बर्फ के नीचे स्थिर रहते हैं... मैंने उनके रिश्ते के सर्वश्रेष्ठ सहयोगी लेखक को फोन किया और ईमेल भेजे और एक दूसरे से बहुत अनुकूल प्रतिक्रिया प्राप्त की, जो वर्तमान में पर्याप्त है।
	Target_Side	डार्नोल्ड ने कहा, ’इस समय हम बिना किसी हिचकिचाहट के रहते हैं... मैंने उनके रिश्ते के सबसे अच्छे सहयोगी को फोन किया है और इलेक्ट्रॉनिक ईमेल भेजे हैं और एक दूसरे से बहुत मैत्रीपूर्ण प्रतिक्रिया प्राप्त की है, जो वर्तमान में पर्याप्त है।
Translation Type		Example 2
Ja-Hi	Source Sentence	我々 が 飼っ て いる 生後 4 か月 の マウス は かつて 糖尿 病 でし た が 現在 は 糖尿 病 で は ない、」 と 彼 は 付け加え まし た。
	Reference	उन्होंने कहा ” कि अब हमारे पास 4 महीने उम्र वाले चूहे हैं जिन्हें मधुमेह नहीं है जो मधुमेह के रोगी थे ।
	w/o Romanization	हमारे पास चार महीने का माउस एक बार मधुमेह था, लेकिन अब यह मधुमेह नहीं है,” उन्होंने कहा।
	Both_side	उन्होंने कहा , ” हमारे पास पहले से मौजूद चार महीने के चूहों में मधुमेह था , लेकिन वर्तमान में मधुमेह नहीं है ।
	Source_Side	उन्होंने कहा, ”हमारे पास 4 महीने का जीवन माउस एक बार मधुमेह था, लेकिन वर्तमान में यह मधुमेह नहीं है।
	Target_Side	उन्होंने कहा, ”हमारे पास चार महीने का बच्चा एक बार मधुमेह था, लेकिन अब यह मधुमेह नहीं है,” उन्होंने कहा।
Translation Type		Example 3
Hi-Zh	Source Sentence	उन्होंने (जेमी सिमिनॉफ़) कहा , कि उन्होंने एक वाईफ़ाई डोर बेल बनाई है .
	Reference	他 称,他 制作 了 一个 WiFi 门铃。
	w/o Romanization	他 说:“杰米 · 西米夫 , 他 创造 了 一个 柳条 钟。 ”
	Both_side	他 说,他 制作 了 一个 WiFi 门铃。
	Source_Side	他 说,他 制作 了 一个 WiFi 门铃。
	Target_Side	他 说:“杰米 · 西米诺夫,他 创造 了 一个无线电 门铃。 ”
Translation Type		Example 4
Zh-Hi	Source Sentence	802.11 n 标准在 2.4 Ghz 和 5.0 Ghz 频率上都可运行
	Reference	802.11 n मानक दोनों 2.4 गीगाहर्ट्ज़ और 5.0 गीगाहर्ट्ज़ आवृत्तियों पर काम करता है।
	w/o Romanization	दक्षिणमानकों को जी . बी . एच . डी . और आर . जी . बी . दर दोनों पर लागू किया जा सकता है ।
	Both_side	802.11 मानकों को 2.4 गीगाहर्ट्ज और 5.0 गीगाहर्ट्ज आवृत्ति पर संचालित किया जा सकता है।

Table 3: Qualitative analysis of translation outputs across different romanization settings language pairs.