Niyamika at BHASHA Task 1: Word-Level Transliteration for English-Hindi Mixed Text in Grammar Correction Using MT5

Rucha Ambaliya
Department of ICT
VNSGU, India
rucha.ambaliya@gmail.com

Mahika Dugar
Department of ICT
VNSGU, India
justmahikadugar@gmail.com

Dr. Pruthwik Mishra
Department of AI
SVNIT, India
pruthwikmishra@aid.svnit.ac.in

Abstract

Grammar correction for Indian languages poses significant challenges due to complex morphology, non-standard spellings, and frequent script variations. In this work, we address grammar correction for English-mixed sentences in five Indic languages—Hindi, Bengali, Malayalam, Tamil, and Telugu—as part of the IndicGEC 2025 shared task at Bhasha Workshop. Our approach first applies word-level transliteration using IndicTrans to normalize romanized and mixed-script tokens, followed by grammar correction using the mT5-small model. Although our experiments focus on these five languages, the methodology is generalizable to other Indian languages. Our system demonstrates stable performance across the five languages in the IndicGEC 2025 shared task, which included 8–11 participating systems per language. We achieve our best performance in Telugu with a rank of 3 out of 8, while securing ranks of 7 out of 8 in both Bengali and Malayalam. For Hindi, we obtain a rank of 9 out of 11, and for Tamil, a rank of 9 out of 9. Our implementation and code are publicly available at: https://github.com/Rucha-Ambaliya/bhasha-workshop.

1 Introduction

Indian languages exhibit rich morphology and diverse scripts, which complicates grammar correction, especially when the text is code-mixed with English. Standard grammar correction models trained on monolingual text often struggle with such inputs.

To address this challenge, we propose a word-level transliteration approach: English tokens in the sentence are converted into the selected main native language script. The transliterated text is then fed into a grammar correction model based on mT5 (Xue et al., 2021), enabling accurate detection and correction of grammatical errors. This pipeline can be easily extended to other Indian languages with minimal adaptation.

2 Related Work

Grammar Error Correction (GEC) for English: Early work on GEC focused on English using statistical and neural machine translation models. The CoNLL-2014 shared task (Ng et al., 2014) evaluated GEC systems using the M^2 scorer, with the best system achieving 37.33% F1 score. Later work by Junczys-Dowmunt et al. (2018) approached neural GEC as a low-resource machine translation task, achieving competitive performance. The GLEU metric (Napoles et al., 2015) was introduced specifically for evaluating grammatical error correction systems, measuring n-gram overlap between system output and reference corrections.

GEC for Indic Languages:

Transliteration and Normalization: Bhat et al. (2015) developed rule-based and data driven systems used for standardized text processing, focusing on transliterated transliteration search tasks (Choudhury et al., 2014). Various neural approaches (Kunchukuttan et al., 2021; Madhani et al., 2023) have been proposed for Indic language transliteration, leveraging character-level and subword representations to handle script variations.

Code-Mixed Language Processing: The SAIL-2015 shared task (Patra et al., 2015) addressed sentiment analysis in Hindi-English, Bengali-English, and Tamil-English tweets, with top systems reporting 66–71% accuracy using Naive Bayes. Sharma et al. (2015) applied a lexicon-based method for Hindi-English sentiment classification using FIRE datasets. Joshi et al. (2016) used sub-word level LSTMs (Hochreiter and Schmidhuber, 1997) for Hindi-English code-mixed datasets, improving accuracy by 18%. Hassan et al. (2016) used LSTMs

for Bengali sentiment analysis, achieving 78% accuracy on binary and 55% on three-class classification tasks.

3 Corpus Details

The IndicGEC 2025 shared task at Bhasha Workshop provides a dataset for training and evaluation in five Indic languages for the grammar correction task. For each language, the dataset is distributed in three files:

- **train.csv:** Contains both input and output sentences, used to train the grammar correction models.
- **dev.csv:** Used during the development phase to evaluate the model on the organizers' system. It contains both input and output sentences and is evaluated using GLEU to identify error patterns and improvement areas.
- **test.csv:** Contains only input sentences and is used for the final evaluation during the workshop.

3.1 Original Dataset Statistics

Language	Train	Dev	Test
Hindi	599	107	236
Bangla	659	102	330
Malayalam	312	50	102
Tamil	91	16	65
Telugu	603	100	315

Table 1: Statistics of the original dataset provided by the Bhasha Workshop organizers.

Although the original dataset already contains realistic grammatical issues such as insertions, inconsistent punctuation, character-level errors (missing or swapped characters etc.), and word-level errors (misplaced or missing tokens etc.). Its overall size is too limited to train a large multilingual model scuh as mT5 effectively. Given the complexity of Indic languages, which involve rich morphology, spelling variations, and frequent code-mixing, the amount of erroneous data in the original set is insufficient for the model to learn diverse and robust error patterns.

3.2 Baseline Performance on Original Dataset

To assess the effectiveness of the provided dataset, preliminary experiments are conducted using the mT5-small model trained separately for each language using its respective train.csv file provided by Bhasha Workshop. The trained models are then evaluated on the corresponding dev and test datasets. We measure GLEU scores with and without applying the IndicTrans transliteration step.

Language	GLEU (Transliterated)	GLEU (Non-Transliterated)
Hindi	17.74	18.13
Bangla	17.00	17.00
Malayalam	20.05	20.05
Tamil	4.99	4.99
Telugu	12.21	12.21

Table 2: GLEU scores on dev.csv using the original dataset.

Language	GLEU (Transliterated)	GLEU (Non-Transliterated)
Hindi	15.62	15.56
Bangla	18.08	18.08
Malayalam	27.07	27.07
Tamil	0.46	0.46
Telugu	12.39	12.16

Table 3: GLEU scores on test.csv using the original dataset.

The results indicate consistently low GLEU scores across all languages, with extremely poor performance for Tamil and only marginal improvements across the remaining languages. Furthermore, the transliteration step did not yield significant gains at this stage. This can be largely attributed to the fact that, except for Hindi and a very small number of instances in Telugu, none of the other languages contained English tokens in their dev and test datasets, thereby limiting the observable impact of transliteration. Even for Hindi, only a single English word was present in the dev.csv file across the evaluated samples, while the Telugu test set contained only 2-3 English tokens in total. However, the slight improvement observed in the Hindi and Telugu test set suggests that transliteration may have contributed positively where English-mixed content was present. Overall, the poor performance indicates that the primary limitation stemmed from insufficient training data rather than script normalization, which

motivated the need to expand and augment the dataset with synthetic grammatical errors to improve model generalization and correction capability.

3.3 Data Filtering

To address this, we construct an augmented training corpus by incorporating additional sentences from the IndicCorpV2 dataset (Doddapaneni et al., 2023) for each language. The following filtering criteria are applied:

- Only sentences containing characters of the main language are retained, since mixing with other languages reduces the prediction accuracy of mT5 for the target language.
- Sentence length between 5 and 15 words is selected, as IndicCorpV2 contained long paragraphs and multiple iterations show that the model performs best with this length.

This process resulted in an initial corpus of 10000 sentences per language.

3.4 Data Augmentation

To simulate realistic grammatical and spelling errors, we apply **both character-level and word-level augmentations**:

- **Character-level:** random insertion, deletion, or swapping of characters.
  (Inserted or swapped characters are arbitrarily selected from within the same sentence and placed at a random position.)
  - **Insertion:** घर → घरर (random character र inserted)
  - **Deletion:** रमत → रत (letter म deleted)
  - **Swap:** खाना → नाखा (characters खा and ना swapped)
- **Word-level:** random insertion, deletion, or swapping of words within a sentence.
  (Inserted or swapped words are arbitrarily selected from within the same sentence and placed at a random position.)
  - **Insertion:** मैं स्कूल गया। → मैं गया स्कूल गया। (word गया inserted)
  - **Deletion:** मैं स्कूल गया। → मैं गया। (word स्कूल deleted)
  - **Swap:** मैं स्कूल गया। → स्कूल मैं गया। (words मैं and स्कूल swapped)

For each sentence, either a character-level or a word-level error is introduced randomly. The augmented sentences are paired with their original versions to form input-output pairs for training.

During early experiments, augmenting only 50% of the sentences did not provide the model with enough erroneous examples, leading it to often copy the input as-is instead of applying corrections. To ensure that the model learns error patterns effectively, we increase the augmentation ratio to 70% of the sentences. Each augmented sentence is paired with its original version to form input-output pairs for training.

3.5 Final Dataset Statistics

Language	Correct Original Pairs	Augmented Pairs	Final Training Pairs
Hindi	10000	7000	10000
Bangla	10000	7000	10000
Malayalam	10000	7000	10000
Tamil	10000	7000	10000
Telugu	10000	7000	10000

Table 4: Corpus augmentation statistics after filtering and applying character- and word-level perturbations.

7,000 sentences were randomly selected for augmentation while preserving 10,000 input-output pairs per language. Although error injection was performed randomly, the resulting distribution of augmentation types was approximately uniform. Since the final model used in our experiments was trained only on the Hindi corpus, we report detailed augmentation statistics for Hindi to illustrate the distribution of error types. The augmented Hindi samples were evenly spread across different perturbation categories: word deletion in 596 sentences (17.03%), word insertion in 598 sentences (17.09%), word swapping in 575 sentences (16.43%), character insertion in 564 sentences (16.11%), character deletion in 570 sentences (16.29%), and character swapping in 597 sentences (17.06%). This balanced distribution ensured exposure to a diverse range of grammatical and spelling error patterns without bias toward any single error type.

This augmentation strategy provides a balanced distribution of correct and erroneous sentences, significantly improving the model's ability to learn grammar correction patterns and handle real-world noise such as spelling errors, informal usages, and

code-mixed constructions common in Indic language text.

4 Approach

4.1 Features

4.1.1 Token-level embeddings:

Sentences are first tokenized using the mT5 tokenizer. Tokens belonging to the main language (e.g., Hindi, Bangla) are kept unchanged, while tokens in other scripts or Romanized form are transliterated into their canonical native script using IndicTrans (Bhat et al., 2015) transliteration toolkit.

4.1.2 Subword-level encoding:

The SentencePiece (Kudo and Richardson, 2018) tokenization of mT5 helps effectively handle out-of-vocabulary (OOV) words that frequently occur in Romanized and code-mixed Indic language text.

4.1.3 Cross-lingual Generalization:

Although the mT5 model is trained only on Hindi data, we directly use for inference on the other four languages (Bangla, Malayalam, Tamil, and Telugu) without additional fine-tuning under zero-shot settings. This is possible because mT5 is a multilingual model with shared subword representations across Indic languages. The transliteration step ensures that the input text for all languages is standardized to native scripts, allowing the model to generalize effectively across languages.

4.2 Models

4.2.1 Transliteration with IndicTrans:

Since the grammar correction model is trained exclusively on sentences in the main language, it is unable to handle words in other scripts (e.g., English or Romanized Hindi). To address this, we use IndicTrans (Bhat et al., 2015) to transliterate all non-main language tokens into the canonical script at the word level. Tokens already in the main language are left unchanged. This ensures that the grammar correction model is provided with inputs in a consistent script.

4.2.2 Grammar Correction with mT5:

Once standardized, the transliterated sentences are passed to the mT5 encoder, which predicts grammatically corrected sequences in the decoding stage. This step improves sentence structure, morphology, spelling, and word order, producing clean and standardized output sentences.

4.3 Inference Pipeline

The complete inference pipeline follows these steps:

1. The input sentence is tokenized using the mT5 tokenizer.
2. Non-main language tokens (e.g., English words in a Hindi sentence) are transliterated into the main language script using IndicTrans.
3. The standardized (transliterated) sentence is fed into the mT5 grammar correction model.
4. The output sentence contains corrected grammar and transliterated tokens in the native script, while the original main language tokens are preserved.

Hyperparameters:

The hyperparameters used in fine-tuning the mT5 model are detailed in Table 5.

Hyperparameter	Value
Model	mT5-small
Learning Rate	2e-4
Batch Size	2
Epochs	21
Max Seq Length	128
Gradient Accumulation	4

Table 5: Hyperparameters for mT5-based transliteration and grammar correction.

5 Evaluation

We evaluate our model using the GLEU score (Napoles et al., 2015), following the official evaluation script used by the IndicGEC 2025 shared task (available at https://github.com/BHASHA-Workshop/IndicGEC2025/blob/main/score.py). It measures the grammatical accuracy of predicted sentences by comparing them to reference sentences using n-gram precision and recall. Higher scores indicate better grammatical quality.

5.1 Performance on Augmented Data

We evaluate the model trained on the augmented dataset under two configurations: with and without applying the IndicTrans transliteration step, on both dev.csv and test.csv.

5.1.1 Development Set Results (Augmented Training)

Language	GLEU (Trans-literated)	GLEU (Non-Trans-literated)
Hindi	83.25	83.25
Bangla	86.94	86.94
Malayalam	89.79	89.79
Tamil	73.07	73.07
Telugu	85.18	85.18

Table 6: GLEU scores on dev.csv using the augmented dataset.

5.1.2 Test Set Results (Augmented Training)

Language	GLEU (Trans-literated)	GLEU (Non-Trans-literated)
Hindi	79.47	78.98
Bangla	81.83	81.83
Malayalam	89.77	89.77
Tamil	84.48	84.48
Telugu	85.03	85.03

Table 7: GLEU scores on test.csv using the augmented dataset.

5.1.3 Observations

The augmented data yielded substantially improved results after training on the expanded corpus, as evidenced by the significant increase in GLEU scores when compared with the baseline results reported in Tables 2 and 3.

Notably, the transliteration step showed a slight but consistent positive effect for Hindi in the test set. This can be attributed to the presence of a small number of English tokens in the Hindi data, whereas the other languages contained no English words in both `dev.csv` and `test.csv`. Even for Hindi, only a single English word was observed in the development set. Despite this scarcity, the marginal improvement in the Hindi test results suggests that transliteration contributed positively in scenarios involving code-mixed content, indicating its potential effectiveness when such inputs are more prevalent.

5.2 Final Submission Results

For the final shared task submission, our model was trained on 10,000 augmented sentence-pairs of Hindi and infered for others, incorporating the IndicTrans transliteration step. The trained model was evaluated on the official test.csv file. Table 8 presents the official GLEU scores and corresponding ranks obtained on the leaderboard.

Language	GLEU	Rank
Hindi	79.47	9
Bangla	81.83	7
Malayalam	89.77	7
Tamil	84.48	9
Telugu	85.03	3

Table 8: Final leaderboard performance of our system on the IndicGEC 2025 test set.

These results demonstrate that training on augmented data substantially enhanced the model's grammar correction capability, leading to stable and competitive performance across languages—especially in Telugu where we secured the 3rd rank. While transliteration's benefit was limited to Hindi due to the scarcity of English tokens in the other languages, the overall trend confirms that our augmentation strategy and pipeline were effective.

6 Error Analysis & Observations

Analysis of the model outputs reveals distinct error patterns for Hindi and non-Hindi languages due to differences in training exposure and linguistic structure.

6.1 Errors in Hindi Outputs

Since the model was fine-tuned on Hindi input–output pairs, most Hindi errors are surface-level formatting issues:

- **Spacing and punctuation inconsistencies:** Extra or missing spaces around commas, full stops, colons, hyphens, quotes, digits, and measurement units, reducing textual readability.
- **Incorrect hyphen and quote formatting:** Improper spacing in compound words and misaligned quotation marks, especially around acronyms such as "आई.आई.टी" already given in Hindi.
- **Minor transliteration and tokenisation errors:** Occasional incorrect mapping of Roman words into Devanagari script, e.g., "CHATGPT" → "चतप्त" instead of "चैट जीपीटी".

6.2 Errors in Other Languages (Bangla, Malayalam, Tamil, Telugu)

For non-Hindi languages, the model exhibits more severe linguistic issues arising from poor cross-lingual generalisation:

- **Cross-script contamination:** Output sentences occasionally include Devanagari characters due to Hindi-centric training, especially when the model is uncertain about the target language context.
- **Semantic drift instead of correction:** Rather than performing precise grammatical correction, the model occasionally produces partially translated or semantically altered sentences, deviating from the original meaning.
- **Poor linguistic adaptation:** Hindi-centric training leads to incorrect grammar, misplaced punctuation, and structurally invalid sentence formations when applied to other Indic languages.

Summary: Hindi outputs primarily suffer from formatting and minor transliteration inconsistencies, whereas non-Hindi languages demonstrate deeper structural problems such as script mixing, semantic deviation, and poor linguistic coherence. These differences highlight the limitations of applying a Hindi-trained mT5-small model to multilingual grammatical correction tasks without language-specific fine-tuning or adaptation strategies.

7 Limitations

- **Limited fine-tuning across languages:** Although evaluation was conducted for multiple Indic languages, the model was only fine-tuned on the Hindi augmented corpus. For the remaining languages, the model was used in an inference-only setup, without language-specific fine-tuning on their respective augmented datasets, which may have constrained performance and generalization.
- **Numerical normalization:** English numerals were not converted into their corresponding Indic script representations (e.g., 123 → १२३), which could affect readability and grammatical correctness in certain contexts.
- **Transliteration of unseen tokens:** The transliteration module occasionally produced incorrect outputs for unknown or rare tokens such as brand names and technical terms (e.g., "CHATGPT" → "चतप्त"), highlighting limitations in handling out-of-vocabulary words.

8 Conclusion & Future Work

We present a word-level transliteration approach using IndicTrans for English-Hindi code-mixed text, followed by grammar correction by mT5. The approach improves the performance of grammar correction systems on code-mixed inputs. Future directions include:

- **Contextual Understanding:** Better handle long and complex sentences using syntactic or semantic features using larger models such as mT5-base and mT5-large.
- **Multilingual Datasets:** Explore multilingual GEC datasets to enhance grammar correction for code-mixed text.
- **Punctuation:** Incorporate explicit punctuation correction modules or multi-task learning.
- **Evaluation:** Complement GLEU with contextual embedding based metrics such as BERTScore (Zhang et al., 2020), LaBSE (Feng et al., 2022), and human-in-the-loop evaluation.

References

Irshad Ahmad Bhat, Vandan Mujadia, Aniruddha Tammewar, Riyaz Ahmad Bhat, and Manish Shrivastava. 2015. IIIT-H System Submission for FIRE2014 Shared Task on Transliterated Search. In *Proceedings of the Forum for Information Retrieval Evaluation (FIRE '14)*, pages 48–53.

Monojit Choudhury, Gokul Chittaranjan, Parth Gupta, and Amitava Das. 2014. Overview of fire 2014 track on transliterated search. *FIRE 2014*, pages 68–89.

Sumanth Doddapaneni, Rahul Aralikatte, Gowtham Ramesh, Shreya Goyal, Mitesh M. Khapra, Anoop Kunchukuttan, and Pratyush Kumar. 2023. Towards Leaving No Indic Language Behind: Building Monolingual Corpora, Benchmark and Models for Indic Languages. In *Proceedings of ACL 2023*, pages 12402–12426.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Md. Akmal Hassan, Md. Saiful Islam, and Mumit Khan. 2016. Sentiment Analysis on Bangla and Romanized Bangla Text Using Long Short-Term Memory Recurrent Neural Network. *arXiv preprint arXiv:1610.00369*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Aditya Joshi, Ameya Prabhu, Manish Shrivastava, and Vasudeva Varma. 2016. Towards Sub-Word Level Compositions for Sentiment Analysis of Hindi-English Code Mixed Text. In *Proceedings of COLING 2016*, pages 2482–2491.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching Neural Grammatical Error Correction as a Low-Resource Machine Translation Task. In *Proceedings of NAACL-HLT 2018*, pages 595–606.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Anoop Kunchukuttan, Siddharth Jain, and Rahul Kejriwal. 2021. A large-scale evaluation of neural machine transliteration for indic languages. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3469–3475.

Yash Madhani, Sushane Parthan, Priyanka Bedekar, Gokul Nc, Ruchi Khapra, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh M Khapra. 2023. Aksharantar: Open indic-language transliteration datasets and models for the next billion users. In *Findings of the association for computational linguistics: Emnlp 2023*, pages 40–57.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground Truth for Grammatical Error Correction Metrics. In *Proceedings of ACL-IJCNLP 2015*, pages 588–593.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 Shared Task on Grammatical Error Correction. In *Proceedings of CoNLL-2014*, pages 1–14.

Braja Gopal Patra, Dipankar Das, and Amitava Das. 2015. Shared Task on Sentiment Analysis in Indian Languages (SAIL) at MIKE 2015. In *Proceedings of the International Conference on Mining Intelligence and Knowledge Exploration (MIKE)*, pages 650–655.

Shashank Sharma, Srinivas Pykl, and Chandra Rakesh. 2015. Text normalization of code mix and sentiment analysis. In *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1468–1473.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. In *Proceedings of NAACL-HLT 2021*, pages 483–498.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *Proceedings of the International Conference on Learning Representations*.