BHASHA Task 1 (IndicGEC): Minimal-Edit Instruction Tuning for Low-Resource Indic GEC

Akhil Rajeev P

C-DAC Bengaluru

Background

Grammatical Error Correction (GEC) for Indic languages remains challenging due to **scarce supervision**, **rich morphology**, and **script-specific orthography** (e.g., Devanagari and Malayalam ligatures and punctuation). Under low-resource settings, aggressive rewriting or paraphrasing can reduce reference overlap and harm shared-task metrics based on n-gram similarity.

We target an **augmentation-free, reproducible** GEC setup for **Hindi and Malayalam** that encourages **minimal, meaning-preserving edits**. Our key idea is to combine **instruction tuning** with **deterministic decoding** and a **deterministic error analysis** that directly informs prompt design.

Dataset

We adopt the **official IndicGEC shared-task splits** (sentence-level correction with paired noisy → corrected references).

Language	Train	Dev	Test
Hindi	600	107	236
Malayalam	300	50	102

Table 1. Official IndicGEC splits used in our experiments.

Pre-processing (no synthetic augmentation):

- Unicode cleanup (remove invisible artifacts), whitespace normalization
- Punctuation normalization (script-aware)
- Remove null/duplicate entries where applicable

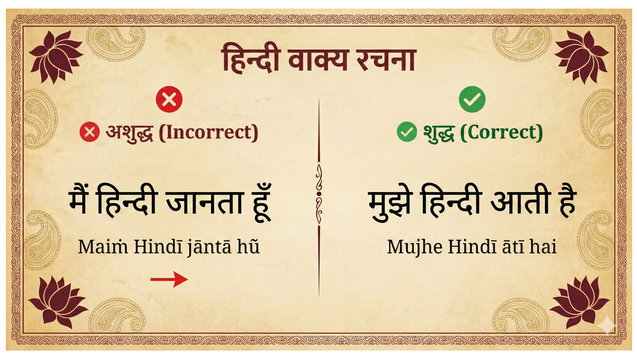

Figure 1. Hindi example: Input → System Output → Reference.

Figure 2. Malayalam example: Input → System Output → Reference.

Problem Statement

Given a noisy Hindi/Malayalam sentence, generate a corrected sentence that fixes grammatical, orthographic, and punctuation errors while *preserving meaning* and making the *fewest necessary edits*. The system should avoid paraphrasing and uncontrolled rewriting, since the shared-task evaluation uses untuned GLEU (n-gram overlap with references).

Proposed Approach

Our system is a **two-stage pipeline**:

Stage 1: Instruction Fine-Tuning (IFT)

- Backbone: **Gemma 3 12B Instruct**
- Training: **bnb 4-bit** checkpoint with **PEFT/LoRA** adapters via **Unsloth**
- Supervision: **Alpaca-style** (Instruction / Input / Response)
- Objective: enforce **minimal-edit** behavior (no paraphrase, keep entities/numbers)

Stage 2: Deterministic Inference + Light Normalizer

- **Greedy decoding** (no sampling) for stable, locality-preserving edits
- Normalizer: surface-only cleanup (whitespace, punctuation spacing, terminal marks, remove prompt echo) — **no rewriting**

Deterministic Error Analysis → Prompt Design

A deterministic classifier labels each sentence pair with one of **nine categories**:

- Null/Empty, No Error, Punctuation/Whitespace, Word Order,
- Missing/Extra Word, Syntax/Agreement, Morphology, Spelling/Orthography, General Grammar

We compute category distributions and use them to build **fixed, language-specific prompts** that **prioritize safe edits** (punctuation/morphology) and explicitly **de-prioritize risky operations** (large reordering/deletions) unless necessary.

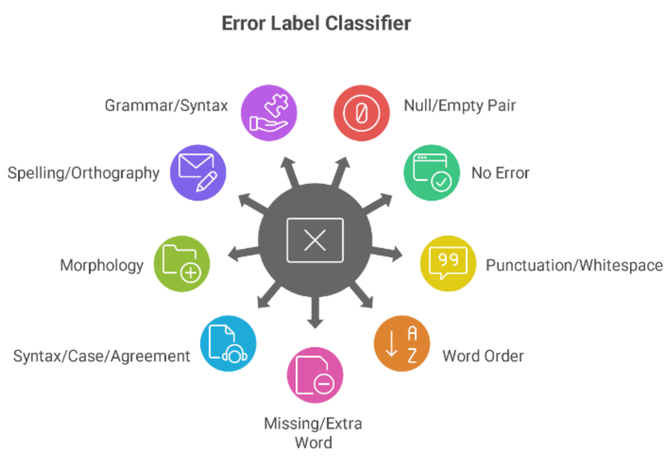

Figure 3. Error Labels identified by the deterministic error classifier

Evaluation Metric

Official shared-task metric: untuned GLEU. GLEU measures n-gram overlap between system output and reference (no parameter tuning). This incentivizes **conservative, minimal edits** and penalizes paraphrases even if fluent.

Design implications:

- Prompt constraints to discourage rewriting
- Surface normalizer to reduce trivial mismatch (spacing/punctuation)

Results

Language	Test GLEU	Rank
Malayalam	**92.41**	6th
Hindi	**81.44**	3rd

Table 2. Official IndicGEC test results under untuned GLEU.

Takeaway: Minimal-edit IFT + deterministic inference provides a strong, augmentation-free baseline under strict low-resource constraints.

(a) Limitations of the Approach

For Further Information

(b) Paper (QR)