VāṇīSetu: A Human-AI Collaborative Framework for Scalable Conversational Speech Corpus Creation in Low-Resource Settings

Anonymous ACL submission

Abstract

We present **VāṇīSetu**, a human-AI collaborative framework for constructing high-quality speech corpora in low-resource languages. As a case study, we apply VāṇīSetu to create **KrishiVāṇī**, a 100-hour Hindi dataset of **unscripted**, **noisy**, and **code-mixed** agricultural speech mined from YouTube. VāṇīSetu integrates automatic speech recognition (ASR), lightweight and large language model-based post-correction, and structured, multi-stage human validation implemented though an enhanced annotation tool, **Vāgyojaka**. Experiments show that domain-specific fine-tuning improves ASR accuracy on real-world agricultural speech, and that small language models such as mT5 provide low-latency corrections that reduce annotation effort by **61%** while preserving transcript fidelity. By shifting annotators from manual transcribers to informed validators, VāṇīSetu enables scalable and linguistically rich corpus creation, and highlights practical cost-quality-latency trade-offs in integrating LMs/LLMs into human-in-the-loop dataset development.

1 Introduction

In multilingual, low-literacy settings such as rural India, speech is often the most accessible interface for information access and assistance (Deshmukh and Chalmeta, 2024). However, robust speech technology remains difficult to deploy, not only due to modeling limitations, but also because usable, domain-specific, real-world datasets are scarce (Adiga et al., 2021; Javed et al., 2024b). Despite progress in high-resource ASR (Radford et al., 2023), Hindi and other low-resource systems degrade under conditions typical of rural communication, including spontaneous noisy speech, dialectal variation, and dense code-mixing (Javed et al., 2024b).

Building ASR systems that work "in the wild" requires more than better models; it demands **better data** (Bhogale et al., 2023a; Javed et al., 2023; Kakwani et al., 2020; Gangwar et al., 2023). However, curating high-quality speech corpora remains expensive and labor-intensive: annotators must correct transcripts, resolve speaker turns, segment and align utterances, and normalize inconsistent orthography and code-mixed forms, often costing 6-8× real-time per hour of audio (Bhanushali et al., 2022b). We address this bottleneck with **VāṇīSetu** (Fig. 1), a human-AI collaborative framework for scalable creation of domain-rich speech datasets. VāṇīSetu integrates automatic transcription and diarization, LM/LLM-based post-correction (Kumar et al., 2022b; Wang et al., 2022; Kumar et al., 2024; Ma et al., 2023; Li et al., 2024b), and a layered human validation workflow supported by an enhanced Vāgyojaka interface (Kumar et al., 2022a), offloading repetitive low-confidence edits to models while reserving domain-sensitive decisions for humans.

We instantiate VāṇīSetu on **KrishiVāṇī**, a **100-hour** Hindi conversational agricultural speech corpus derived from YouTube, capturing diverse speakers/accents, real ambient noise, and dense code-mixing typical of rural and semi-urban interactions. On this testbed, AI post-correction combined with layered human validation reduces annotation effort by **61%** relative to a fully manual workflow, despite challenging phenomena such as specialized vocabulary, dialectal variation, compound words, word segmentation, and mixed Hindi/English numerals (Appendix X). Although this is 20 percentage points lower than the savings reported in (Liu et al., 2021), we attribute the gap to the higher linguistic complexity of agricultural conversational Hindi compared to clean read-speech benchmarks (e.g., Common Voice (Ardila et al., 2020)). Overall, KrishiVāṇī provides a realistic domain benchmark and shows that VāṇīSetu improves corpus construction efficiency without sacrificing linguistic fidelity or domain relevance. Our key contributions are as follows:

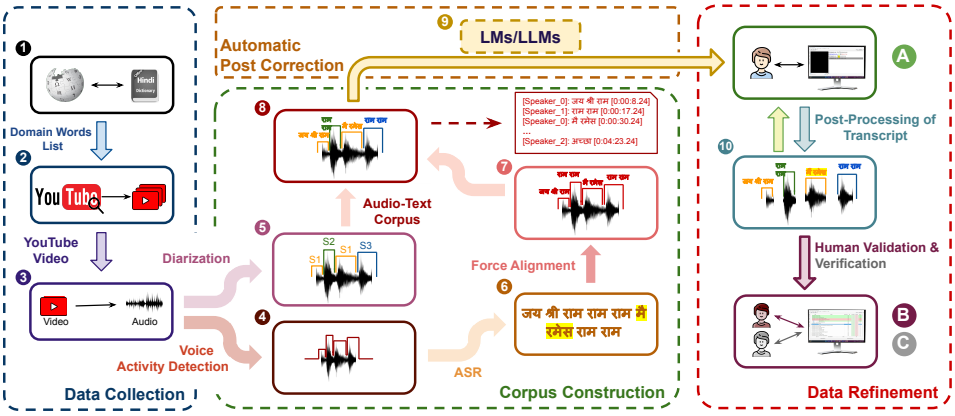

Figure 1: Overview of the VāṇīSetu pipeline for constructing KrishiVāṇī, a high-quality, multi-speaker corpus in real-world agricultural discourse combining domain-guided data collection (Steps 1-3), corpus construction (Steps 4-8), LM/LLM-based post-correction (Step 9), and layered human-in-the-loop validation (Steps A-C).

1. **VāṇīSetu**, a human-AI corpus construction framework that operationalizes *role-separated, multi-stage* annotation with explicit stage gating and audit trails, integrating ASR, automated post-correction, and layered human validation in an extensible workflow.
2. **KrishiVāṇī**, a 100-hour Hindi conversational agricultural speech dataset capturing real-world speaker and dialect variation, ambient noise, and dense code-mixing typical of rural and semi-urban interactions.
3. An **enhanced Vāgyojaka** interface that natively supports multi-stage correction, adjudication, and annotator feedback/incentive mechanisms to enable scalable annotation operations.
4. A **comparative study** of LM/LLM post-correction *in the loop*, analyzing tradeoffs across annotation effort, transcript quality, and responsiveness under agriculture-specific error modes (e.g., numerals, segmentation, code-mixing).

Novelty: While human-in-the-loop annotation is well established, our contribution is its *agriculture-specific operationalization* through a role-separated, incentive-linked validation structure and an instrumented workflow that quantifies cost, quality, latency tradeoffs for LM/LLM assistance in real corpus creation[1].

2 The VāṇīSetu Framework

Building robust corpora in low-resource, code-mixed settings requires scalable automation with human oversight: fully manual pipelines are accurate but slow, while fully automatic pipelines degrade on noisy, spontaneous, and informal speech. We therefore introduce VāṇīSetu, a modular human-AI framework that bridges raw speech collection to quality-controlled transcripts via four stages (Fig. 1): (i) domain-guided data collection, (ii) automated corpus construction, (iii) LM/LLM-based automatic post-correction, and (iv) layered human refinement with quality control.

2.1 Data Collection Stage

In the first stage, VāṇīSetu performs domain-guided acquisition of raw speech. For KrishiVāṇī, we target Hindi agricultural content and construct a bilingual keyword inventory using domain glossaries, Wikipedia categories, and curated Hindi-English lexicons. We query YouTube with these keywords and prioritize long-form videos with unscripted, conversational speech. We then remove near-duplicates, download eligible videos under appropriate usage constraints, and convert audio to 16 kHz, mono WAV. The resulting raw audio is intentionally "in-the-wild," exhibiting spontaneous phrasing, regional accents, speaker overlaps, and ambient noise that are representative of mobile-first rural deployments but challenging for ASR.

2.2 Corpus Construction Stage

The second stage converts raw audio into a structured, navigable corpus via automated preprocessing:

Voice Activity Detection (VAD) removes silence/background-only regions and segments audio into utterances ($\leq$ 20 seconds).

[1]Code and models, KrishiVāṇī dataset: https://anonymous.4open.science/r/KrishiVaani-4C73/, modified Vāgyojaka tool: https://github.com/vagyojaka/VAgyojaka

Speaker Diarization (PyAnnote (Bredin, 2022)) assigns speaker turns, preserving conversational structure and enabling speaker-aware annotation.
ASR Transcription is produced using a strong general-purpose baseline (IndicWav2Vec (Javed et al., 2023)) and our domain-adaptive model (KVWav2Vec), with beam-search decoding for improved stability.
Forced Alignment maps transcript tokens to audio spans using a CTC-based aligner (Watanabe et al., 2017), enabling efficient navigation and targeted edits inside the annotation interface.

2.3 Automatic Post-Correction by LM/LLMs

In the third stage, VāṇīSetu applies automatic post-correction to reduce human workload before manual review. We support both lightweight LMs and larger LLMs as interchangeable components:
LM-based Correction uses fine-tuned mT5 (Xue, 2020) and ByT5 (Xue et al., 2022) on paired (ASR output, corrected transcript) examples. These models are effective for common ASR artifacts in Hindi, including lexical distortions, mistransliterations, and token boundary/segmentation errors.
LLM-based Correction evaluates larger models, including LLaMA-3-Nanda-10B (Touvron et al., 2023) via fine-tuning (Li et al., 2024a; Hu et al., 2024) and ChatGPT-4o (Ma et al., 2023) via in-context learning (Ghosh et al., 2024). While LLMs can better handle domain-specific phrases and code-switching, they may introduce unsupported edits and have higher latency, which in turn requires more careful human verification. For KrishiVāṇī, we use **mT5** as the default post-corrector and pass its outputs to the final human refinement stage.

2.4 Data Refinement Stage

The final stage performs quality-controlled human refinement over the LM/LLM-corrected transcripts. As shown in Fig. 1, we use a **role-separated workflow** with three responsibilities: **Annotators (A)** perform edits, **Validators (B)** re-check and consolidate corrections, and **Verifiers (C)** provide final approval and enforce consistent inclusion criteria across batches. We extend the open-source **Vāgyojaka** tool (Kumar et al., 2022a) to support multi-stage correction and scalable operations (Fig. 2), including (i) in-place transcript editing alongside diarization and audio metadata, (ii) transliteration assistance for embedded English terms, (iii) normalization of punctuation and spelling variants, and (iv) issue flagging for escalation. Quality control is enforced through **a reward-linked mechanism** aligned with retention/acceptance of edits during validation, which incentivizes careful corrections while maintaining throughput. This combination of automation, role separation, and incentive-linked verification is central to VāṇīSetu's efficiency gains without compromising linguistic fidelity.

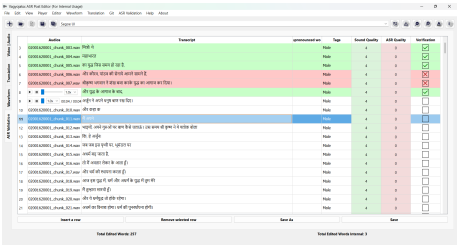

Figure 2: Performing data validation and data verification through the Vāgyojaka tool.

3 Experimentation

We design experiments to quantify how VāṇīSetu impacts (i) Hindi ASR robustness in realistic agricultural speech and (ii) the human effort required to produce high-quality transcripts. We first describe the KrishiVāṇī corpus and evaluation splits, then detail our ASR baselines and domain-adaptive model, and finally evaluate LM/LLM-based post-corrections. Section 4 reports ASR accuracy, post-correction quality gains, and the resulting reduction in human annotation effort.

3.1 KrishiVāṇī Corpus

KrishiVāṇī contains 100 hours of Hindi agricultural-domain speech curated using VāṇīSetu. We reserve a 2.5-hour test set and construct three evaluation partitions to probe deployment-relevant generalization:
KV-Known (in-domain, familiar): test speakers overlap with the training set;
KV-Unknown (in-domain, novel): test speakers are disjoint from the training set;
KV-OOD (out-of-domain): speech drawn from unseen domains with disjoint speakers.

3.2 Baseline ASR Models

To study domain adaptation, we build **KVWav2Vec** by fine-tuning IndicWav2Vec on 75 hours of Hindi speech (65h IndicVoice + 10h KrishiVāṇī-train), ensuring no overlap with KrishiVāṇī test speakers. We compare against widely used Hindi ASR baselines: **IndicWav2Vec** (Javed et al., 2023),

Model	IW2V	IC	KVWav2Vec
KV-Known	23.7	24.2	**22.38**
KV-UnKnown	28.57	26.84	**26.04**
KV-OOD	**22.41**	28.56	24.61

Table 1: Models comparison WER (%) scores across different KrishiVaani datasets (**KV = KrishiVaani, IC = IndicConformer, IW2V = IndicWav2Vec**)

Model	KVW2V	ByT5	mT5	Llama	ChatGPT
KV-Known	22.38	24.33	**22.29**	26.37	23.59
KV-UnKnown	26.04	26.77	**25.70**	37.75	26.35
KV-OOD	24.61	25.12	24.35	30.08	**22.62**

Table 2: Comparison of different WER (%) scores of KVWav2Vec with other fine-tuned LMs/LLMs (**KV = KrishiVaani, KVW2V = KVWav2Vec**).

a wav2vec 2.0 CTC model fine-tuned for Hindi, and **IndicConformer** (Javed et al., 2024a), a Conformer-based Hindi ASR model trained with subword/character tokenization.[2]

3.3 Post-Correction using LMs and LLMs

We evaluate automatic post-correction of ASR hypotheses using (i) fine-tuned sequence-to-sequence LMs and (ii) in-context learning (ICL) with LLMs. Our LMs are **mT5-small** (Xue, 2020) and **ByT5-small** (Xue et al., 2022), which address common Hindi ASR artifacts such as lexical distortions, mistransliterations, and segmentation errors. For LLMs, we evaluate **LLaMA-3-Nanda-10B** (Touvron et al., 2023) though supervised fine-tuning and **ChatGPT-4o mini** (Ma et al., 2023) though zero/one/few-shot ICL (Table 4).

4 Results and Discussions

As shown in Table 1, our domain-adapted KVWav2Vec model consistently outperforms strong Hindi ASR baselines on the KrishiVāṇī-Known and KrishiVāṇī-Unknown splits. This indicates that incorporating even a modest amount of in-domain conversational data (10h) can improve recognition in noisy, accented, and code-mixed agricultural speech. On the OOD split, IndicWav2Vec slightly outperforms KVWav2Vec, suggesting that generalist models may retain an advantage under topic drift and domain mismatch. Overall, these results validate blending open-domain and domain-specific data during fine-tuning and retaining a generalist baseline for robustness under distribution shift.

[2]The main paper reports two primary baselines; additional baselines and results appear in Table 3.

To reduce annotation overhead, we evaluate automatic post-correction as a first-pass refinement step before human editing. Table 2 shows that smaller models, particularly mT5-small, outperform larger LLMs in accuracy on KV-Known and KV-Unknown. In KV-OOD, ChatGPT ICL performs best, consistent with its broader exposure to diverse, out-of-domain textual contexts. Latency comparisons are reported in Appendix F. Consistent with these trends, mT5-based post-correction yields the largest reduction in annotator effort within VāṇīSetu (Section 2), demonstrating that smaller, fine-tuned LMs can be a better default than larger LLMs for scalable corpus construction. The reduction in annotation time across stages is also evident, as illustrated in Figure 3.

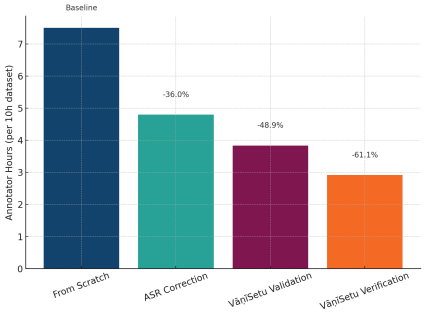

Figure 3: VāṇīSetu achieves a 61.1% reduction in annotation effort over full manual transcription.

5 Conclusion

In this work, we presented VāṇīSetu, a human-AI collaborative framework for constructing high-quality speech corpora in low-resource, code-mixed settings, and validated it through KrishiVāṇī, a 100-hour Hindi conversational agricultural dataset. VāṇīSetu combines ASR, automatic post-correction, and structured, multi-stage human validation via an enhanced Vāgyojaka interface, shifting annotators from full manual transcription to targeted verification of model outputs. This design reduces annotation effort by 61% while preserving linguistic fidelity under realistic noise, accent variation, and dense code-mixing. We further show that mT5 offers a strong accuracy-responsiveness trade-off and is often a better default than larger LLMs for in-domain correction, while ChatGPT ICL performs best under out-of-domain drift. Overall, VāṇīSetu contributes a reusable, domain-adaptive blueprint for scalable speech dataset creation in underrepresented languages and high-variance real-world domains.

Limitations

VāṇīSetu demonstrates promising results in enabling human-AI collaborative corpus creation for Hindi; several limitations constrain its broader applicability and generalizability.

- **Language Resource Dependency**: The framework currently depends on pre-existing Hindi ASR and language models such as IndicWav2Vec, ByT5, and LLaMA-3-Nanda, which benefit from extensive pretraining on Indic language corpora. For truly low-resource languages lacking such foundational models, VāṇīSetu's post-correction and alignment mechanisms may underperform unless comparable models or datasets are first developed. This limits the system's plug-and-play portability across the broader spectrum of underrepresented languages.

- **Code-Switching Complexity**: Although KrishiVāṇī includes code-mixed utterances, the current system is not optimized for heavily code-switched conversations (e.g., rapid alternation between Hindi and English within a single utterance). The underlying ASR and correction models often misrecognize such segments or apply inconsistent transliteration rules. Extending the framework to support multi-ASR fusion or language-aware post-editing remains an open challenge.

References

Devaraja Adiga, Rishabh Kumar, Amrith Krishna, Preethi Jyothi, Ganesh Ramakrishnan, and Pawan Goyal. 2021. Automatic speech recognition in sanskrit: A new speech corpus and modelling insights. *arXiv preprint arXiv:2106.05852*.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Proceedings of the twelfth language resources and evaluation conference*, pages 4218–4222.

Loïc Barrault, Yu-An Chung, Mariano Coria Meglioli, David Dale, Ning Dong, Mark Duppenthaler, Paul-Ambroise Duquenne, Brian Ellis, Hady Elsahar, Justin Haaheim, et al. 2023. Seamless: Multilingual expressive and streaming speech translation. *arXiv preprint arXiv:2312.05187*.

Anish Bhanushali, Grant Bridgman, Deekshitha G, Prasanta Ghosh, Pratik Kumar, Saurabh Kumar, Adithya Raj Kolladath, Nithya Ravi, Aaditeshwar Seth, Ashish Seth, Abhayjeet Singh, Vrunda Sukhadia, Umesh S, Sathvik Udupa, and Lodagala V. S. V. Durga Prasad. 2022a. Gram vaani asr challenge on spontaneous telephone speech recordings in regional variations of hindi. In *Proc. Interspeech 2022*, pages 3548–3552.

Anish Bhanushali, Grant Bridgman, Prasanta Ghosh, Pratik Kumar, Saurabh Kumar, Adithya Raj Kolladath, Nithya Ravi, Aaditeshwar Seth, Ashish Seth, Abhayjeet Singh, et al. 2022b. Gram vaani asr challenge on spontaneous telephone speech recordings in regional variations of hindi. In *Proc. Interspeech 2022*, pages 3548–3552.

Kaushal Bhogale, Abhigyan Raman, Tahir Javed, Sumanth Doddapaneni, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh M Khapra. 2023a. Effectiveness of mining audio and text pairs from public data for improving asr systems for low-resource languages. In *Icassp 2023-2023 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 1–5. IEEE.

Kaushal Santosh Bhogale, Sai Sundaresan, Abhigyan Raman, Tahir Javed, Mitesh M Khapra, and Pratyush Kumar. 2023b. Vistaar: Diverse benchmarks and training sets for indian language asr. *arXiv preprint arXiv:2305.15386*.

Hervé Bredin. 2022. pyannote.audio speaker diarization pipeline at voxsrc 2022.

Hervé Bredin, Ruiqing Yin, Juan Manuel Coria, Gregory Gelly, Pavel Korshunov, Marvin Lavechin, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, and Marie-Philippe Gill. 2020. Pyannote. audio: neural building blocks for speaker diarization. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7124–7128. IEEE.

Alexis Conneau, Min Ma, Simran Khanuja, Yu Zhang, Vera Axelrod, Siddharth Dalmia, Jason Riesa, Clara Rivera, and Ankur Bapna. 2023. Fleurs: Few-shot learning evaluation of universal representations of speech. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 798–805. IEEE.

Akshay Madhav Deshmukh and Ricardo Chalmeta. 2024. User experience and usability of voice user interfaces: A systematic literature review. *Information*, 15(9):579.

Arjun Gangwar, S Umesh, Rithik Sarab, Akhilesh Kumar Dubey, Govind Divakaran, Suryakanth V Gangashetty, et al. 2023. Spring-inx: A multilingual indian language speech corpus by spring lab, iit madras. *arXiv preprint arXiv:2310.14654*.

Elodie Gauthier, Aminata Ndiaye, and Abdoulaye Guissé. 2024. Kallaama: A transcribed speech dataset about agriculture in the three most

widely spoken languages in senegal. *Preprint*, arXiv:2404.01991. To appear in RAIL @ LREC-COLING 2024.

Sreyan Ghosh, Mohammad Sadegh Rasooli, Michael Levit, Peidong Wang, Jian Xue, Dinesh Manocha, and Jinyu Li. 2024. Failing forward: Improving generative error correction for asr with synthetic data and retrieval augmentation. *arXiv preprint arXiv:2410.13198*.

Yuchen Hu, Chen Chen, Chao-Han Huck Yang, Ruizhe Li, Chao Zhang, Pin-Yu Chen, and EnSiong Chng. 2024. Large language models are efficient learners of noise-robust speech recognition. *arXiv preprint arXiv:2401.10446*.

Tahir Javed, Kaushal Bhogale, Abhigyan Raman, Pratyush Kumar, Anoop Kunchukuttan, and Mitesh M Khapra. 2023. Indicsuperb: A speech processing universal performance benchmark for indian languages. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12942–12950.

Tahir Javed, Janki Nawale, Sakshi Joshi, Eldho George, Kaushal Bhogale, Deovrat Mehendale, and Mitesh M Khapra. 2024a. Lahaja: A robust multi-accent benchmark for evaluating hindi asr systems. *arXiv preprint arXiv:2408.11440*.

Tahir Javed, Janki Atul Nawale, Eldho Ittan George, Sakshi Joshi, Kaushal Santosh Bhogale, Deovrat Mehendale, Ishvinder Virender Sethi, Aparna Ananthanarayanan, Hafsah Faquih, Pratiti Palit, et al. 2024b. Indicvoices: Towards building an inclusive multilingual speech dataset for indian languages. *arXiv preprint arXiv:2403.01926*.

Ananya Joshi, Aditi Kajale, Janhavi Gadre, Samruddhi Deode, and Raviraj Joshi. 2023. L3cube-mahasbert and hindsbert: Sentence bert models and benchmarking bert sentence representations for hindi and marathi. In *Science and Information Conference*, pages 1184–1199. Springer.

Sakshi Joshi, Eldho Ittan George, Tahir Javed, Kaushal Bhogale, Nikhil Narasimhan, and Mitesh M. Khapra. 2025. Recognizing every voice: Towards inclusive asr for rural bhojpuri women. *Preprint*, arXiv:2506.09653. Accepted at Interspeech 2025.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, NC Gokul, Avik Bhattacharyya, Mitesh M Khapra, and Pratyush Kumar. 2020. Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961.

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

Rishabh Kumar, Devaraja Adiga, Mayank Kothyari, Jatin Dalal, Ganesh Ramakrishnan, and Preethi Jyothi. 2022a. Vagyojaka: An annotating and post-editing tool for automatic speech recognition. In *INTERSPEECH*, pages 857–858.

Rishabh Kumar, Devaraja Adiga, Rishav Ranjan, Amrith Krishna, Ganesh Ramakrishnan, Pawan Goyal, and Preethi Jyothi. 2022b. Linguistically informed post-processing for asr error correction in sanskrit. In *INTERSPEECH*, pages 2293–2297.

Rishabh Kumar, Sabyasachi Ghosh, and Ganesh Ramakrishnan. 2024. Beyond common words: Enhancing asr cross-lingual proper noun recognition using large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6821–6828.

Sheng Li, Chen Chen, Chin Yuen Kwok, Chenhui Chu, Eng Siong Chng, and Hisashi Kawai. 2024a. Investigating asr error correction with large language model and multilingual 1-best hypotheses. In *Proc. Interspeech*, pages 1315–1319.

Yuang Li, Jiawei Yu, Min Zhang, Mengxin Ren, Yanqing Zhao, Xiaofeng Zhao, Shimin Tao, Jinsong Su, and Hao Yang. 2024b. Using large language model for end-to-end chinese asr and ner. *arXiv preprint arXiv:2401.11382*.

Mingkuan Liu, Chi Zhang, Hua Xing, Chao Feng, Monchu Chen, Judith Bishop, and Grace Ngapo. 2021. Scalable data annotation pipeline for high-quality large speech datasets development. *arXiv preprint arXiv:2109.01164*.

Rao Ma, Mengjie Qian, Potsawee Manakul, Mark Gales, and Kate Knill. 2023. Can generative large language models perform asr error correction? *arXiv preprint arXiv:2307.04172*.

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.

Ashish Mittal, Darshan Prabhu, Sunita Sarawagi, and Preethi Jyothi. 2024. Salsa: Speedy asr-llm synchronous aggregation. *arXiv preprint arXiv:2408.16542*.

Aanchan Mohan, Richard C. Rose, Sina Hamidi Ghalehjegh, and Srinivasan Umesh. 2014. Acoustic modelling for speech recognition in indian languages in an agricultural commodities task domain. *Speech Communication*, 56:167–180.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ehsan Variani, Tom Bagby, Kamel Lahouel, Erik McDermott, and Michiel Bacchiani. 2018. Sampled connectionist temporal classification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4959–4963. IEEE.

Chengyu Wang, Suyang Dai, Yipeng Wang, Fei Yang, Minghui Qiu, Kehan Chen, Wei Zhou, and Jun Huang. 2022. Arobert: An asr robust pre-trained language model for spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1207–1218.

Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R. Hershey, and Tomoki Hayashi. 2017. Hybrid ctc/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11:1240–1253.

L Xue. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Bo Yang, Lanfei Feng, Yunkui Chen, Yu Zhang, Jianyu Zhang, Xiao Xu, Nueraili Aierken, and Shijian Li. 2025. Agrigpt-omni: A unified speech-vision-text framework for multilingual agricultural intelligence. *Preprint*, arXiv:2512.10624.

A Appendix

In the Appendix, we provide:

1. Section B: Detailed Pipeline for VāṇīSetu
2. Section C: Baseline Models Details
3. Section D: VāṇīSetu Human-AI Pipeline
4. Section E: LM/LLM Models Details
5. Section F: Additional Results
6. Section G: LLM-based Post-Correction
7. Section H: Data Annotators
8. Section K: System Adaptability and Reuse
9. Section I: Related Work
10. Section J: Additional Languages
11. Section L: Compute Infrastructure
12. Section M: Prompts

B Detailed pipeline for VāṇīSetu

This section describes the generation of the KrishiVaani speech corpus through our VāṇīSetu pipeline. Figure 1 illustrates the design of this multi-stage pipeline, which involves these stages: data collection, speech corpus construction, and data refinement. The following subsections give a detailed overview of each stage:

B.1 Data Collection

This stage gathers the required data to be processed further. It is sequentially broken down into the following steps:

Domain Words List (1): We generate different keyword lists (e.g., "khetī" means farming) containing Hindi and English terms from different dictionaries and Wikipedia articles on these domains. We refine it using a keyword filtering module relevant to YouTube Hindi videos[3]. YouTube Hindi videos have English keywords associated with them. Therefore, it is essential to have both Hindi and English keyword lists.

YouTube Search Engine (2): We use the YouTube search engine to find the corresponding videos on these domains from the respective keyword lists. We de-duplicate the retrieved videos. Our analysis of these videos revealed that many of them contained missing or inaccurate subtitles. Therefore, we generated the transcripts for these videos.

Converting Video to Audio (3): We download the video and convert it into audio using single single-channel *wav* format and 16 kHz sampling rate.

B.2 Speech Corpus Construction

Once the data collection stage is complete, the speech corpus construction process begins, involving the following steps:

Voice Activity Detection (4): We conduct voice activity detection (VAD) with a limited maximum

[3]We follow YCC licensing.

duration of 20 sec and average silence detection using mp3split[4]. We also use speaker segmentation from Pyannote to separate the speaker's audio and ensure the integrity of speech content as much as possible.

Diarization (5): For speaker diarization, we segment and label an audio recording using Pyannote (Bredin et al., 2020). The goal of this step is to determine *"who spoke when"* within a multi-speaker audio stream. This enables cleaner and more reliable training data for ASR models.

ASR (6): In this step, we use Hindi ASR to generate the transcripts of the converted audio files. We considered multiple ASR models such as IndicWhisper (Bhogale et al., 2023b), IndicWav2Vec (Javed et al., 2023), etc., to maintain flexibility. Though IndicWhisper showcases a lower Word Error Rate (WER), we prefer IndicWav2Vec owing to its lower Character Error Rate (CER). Since IndicWav2Vec captures fine-grained details at the character level, it also helps to reduce further annotation effort.

Forced Alignment (7): This step maps each word in the utterance to its corresponding audio timestamp. We use a pre-trained IndicWav2Vec-Hindi model based on the Connectionist Temporal Classification (CTC) criterion (Variani et al., 2018). This step also benefits the upcoming data curation stage.

Audio-Text Corpus (8): After performing diarization and forced alignment, we combine the timestamps of the speaker(s) and the transcripts to create the final speaker-wise utterances.

Automatic ASR Correction (9): We use the best one of different fine-tuned LMs (ByT5 (Xue et al., 2022), mT5 (Xue, 2020)) and LLMs (Llama (Touvron et al., 2023), ChatGPT) to perform ASR post-correction. This step enhances both transcript accuracy and annotation efficiency. The output of this step is provided to the data refinement stage.

B.3 Data Refinement

For the entire stage of data refinement, we use the VAgyojaka tool (Kumar et al., 2022a) for each of the data curation, data validation, and data verification steps. Before VAgyojaka, ASR curation was a time-intensive process, where X hours of speech data required 8X hours for curation. This time was spent correcting ASR transcripts, validating forced alignments, and checking speaker diarization. It has proven to reduce this workload to 4-6X hours.

[4]http://mp3splt.sourceforge.net/mp3splt_page/home.php

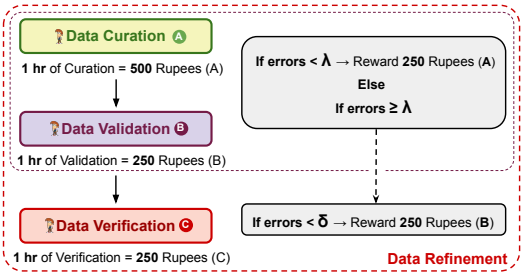

Figure 4: The reward system adopted to encourage data curators and validators. Alongside standard compensation, we used λ as 5% and δ as 2% of the total number of words for enforcing the reward system while curating the KrishiVaani dataset.

Figure 2 displays a snapshot of the VAgyojaka tool.

Data Curation (A): In this step, we collectively perform transcription correction, utterance alignment, and speaker diarization. Due to the absence of open-source guidelines for data curation in Indian languages, we create and use our own set of guidelines for this purpose. We have used fifteen annotators (experts in the Hindi language) for data curation.

Post-Processing of Transcript (10): This step involves normalizing the text by removing punctuation and tags. We also back-transliterate English words like "*Local*" as "*lokl*" using the transliteration model (IndicTXlit (Madhani et al., 2023)), which linguists further verify. We also check if any edits are made in the transcripts; depending on which, they can be sent back to the data curator. Eventually, we split the audio based on utterance alignment, which will be used for further validation and verification.

Data Validation (B): Data validation is a crucial step in ensuring the quality of the curated data. The same fifteen annotators carried out the validation, ensuring that no one validated their own curated data. The data validator corrects the transcript of the split audio. We also introduce a reward system so that annotators can work better as data curators and validators. As seen in Figure 4, whenever any data validator makes edits less than a certain threshold (λ), the data curator will be rewarded, or else, the data validator will perform the subsequent post-correction. If this step achieves less than a certain threshold (δ), the data validator will be rewarded.

Data Verification (C): The Data verifier examines the ASR transcript for the split audio and determines whether to include it in the final dataset. These verifiers are Hindi language experts with

extensive experience. The verifier also helps in complying with the reward system.

C Baseline Models Details

1. IndicWav2Vec (Javed et al., 2023), a multilingual ASR model based on wav2vec 2.0, pretrained on 17,314 hours of Indian language audio using self-supervised learning and fine-tuned with CTC, achieving reduced WER for low-resource languages like Hindi.
2. IndicWhisper (Bhogale et al., 2023b), a fine-tuned version of OpenAI's Whisper trained on 2,150 hours of Hindi audio using multilingual byte-level BPE tokenization, balancing weak supervision with competitive WER.
3. SALSA (Mittal et al., 2024), a hybrid ASR approach integrating Whisper's encoder-decoder with a decoder-only LLM (Llama2-7B) (Touvron et al., 2023) to improve recognition in low-resource settings through synchronous decoding. It is fine-tuned on 10 hrs of FLEURS (Conneau et al., 2023) Hindi dataset.
4. IndicConformer (Javed et al., 2024a), a 130M parameter conformer-based ASR Hindi model combining convolution and transformer layers to support Indian languages with a unified subword-character tokenization strategy.
5. SeamlessM4T (Barrault et al., 2023), a multilingual ASR and translation model covering 100 languages, leveraging 4.5 million hours of pretraining but with unexplored effectiveness in real-world low-resource speech settings.

D VāṇīSetu Human-AI Pipeline

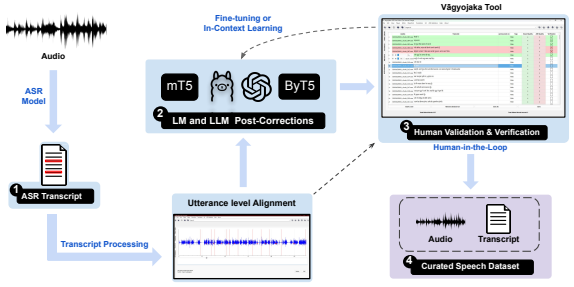

Figure 5: The VāṇīSetu human-AI pipeline. (1) ASR transcripts are created from the audio by using the baseline/fine-tuned ASR models. (2) Lightweight and large-scale language models (mT5, ByT5, ChatGPT, LLaMA) generate post-corrections. (3) Annotators validate and verify transcripts using the Vāgyojaka interface. (4) The curated outputs form the KrishiVāṇī Hindi speech corpus.

E LM/LLM Models Details

1. mT5: mT5 (Xue, 2020) belongs to the T5 family of models, which is an encoder-decoder based LM. It is a multilingual version of the T5 (Raffel et al., 2020) model. The mT5 model uses a subword-based tokenization strategy. It follows the SentencePiece tokenizer (Kudo, 2018), which is a variant of the BPE tokenizer. We selected the *mT5-small* variant as the correction model.
2. ByT5: ByT5 (Xue et al., 2022) model shares the same architecture as the mT5 model. It is a tokenizer-free variant of the mT5 model. Although mT5 has a standard tokenizer, ByT5 processes single-character tokens in UTF-8 encoded bytes. We select the *ByT5-small* variant as the correction model.
3. Llama: Llama-3-Nanda-10B-Chat (Touvron et al., 2023) (Nanda) is a Hindi-focused, instruction-tuned LLM with 10 billion parameters. Built on the Llama-3 model, it has been extensively trained on 65 billion Hindi tokens. Unlike general multilingual models, Nanda prioritizes Hindi, using a balanced 1:1 Hindi-English dataset during training to enhance both languages' capabilities and make it well-suited for fine-tuned text correction tasks.
4. ChatGPT: Powered by OpenAI, ChatGPT (Ma et al., 2023) is an advanced LLM that is used for multiple downstream tasks, including text correction. Unlike fine-tuning for the above LMs/LLMs, we use ChatGPT-4o mini for in-context learning and transcript correction using zero-shot, 1-shot, and few-shot learning. We use SBERT (Joshi et al., 2023) to create sentence embedding for in-context learning.

F Additional Results

In contrast, larger models like **Llama** and **ChatGPT-4o mini** required careful few-shot prompting and SBERT-guided example selection to achieve competitive results. As seen in Table 4, ChatGPT-4o performed poorly in 0-shot conditions but showed significant improvement in 5-shot mode, recovering many missing or substituted tokens. This supports the potential of retrieval-augmented in-context learning, but also underscores the cost in latency and instability, especially in real-time annotation workflows.

For this reason, we integrated **mT5-small** as the default correction engine in Vāgyojaka, prioritizing

Model	IndicWav2Vec	IndicWhisper	IndicConformer	SeamlessM4T	SALSA	KVWav2Vec
KrishiVaani-Known (WER)	23.7	23.42	24.2	49.94	87.59	**22.38**
KrishiVaani-Known (CER)	**8.50**	10.65	10.3	31.23	67.35	8.58
KrishiVaani-UnKnown (WER)	28.57	30.08	26.84	41.5	95.50	**26.04**
KrishiVaani-UnKnown (CER)	12.69	16.62	12.92	25	75.20	**11.98**
KrishiVaani-OOD (WER)	**22.41**	49.78	28.56	42.73	79.09	24.61
KrishiVaani-OOD (CER)	**8.51**	35.96	17.34	27.73	61.24	9.51

Table 3: Models Comparison across different KrishiVaani datasets

Experiment	Shots	KrishiVaani-Known	KrishiVaani-UnKnown	KrishiVaani-OOD
KVWav2Vec	-	22.38	26.04	24.61
ChatGPT-4o mini	0-Shot	29.33	31.89	27.65
With Random	1-Shot	28.36	30.26	26.43
	3-Shot	27.64	30.14	25.65
	5-Shot	26.37	28.95	25.27
With SE Similarity	1-Shot	27.64	30.26	26.16
	3-Shot	26.81	29.90	25.74
	5-Shot	23.59	26.35	**22.62**

Table 4: Comparative overview of different WER (%) scores for different settings using in-context learning through ChatGPT (SE = Sentence Embedding)

ByT5-small	mT5-small	Llama	ChatGPT-4o mini
2.29	**0.97**	10.17	2.03

Table 5: Latency (in seconds) of different models for ASR post-correction.

low-latency, high-consistency performance over marginal accuracy gains from LLMs (see Table 5 for comparative latency results).

F.1 KVWav2Vec Model

We acknowledge the potential bias in evaluating KVWav2Vec solely on the KrishiVaani dataset. The primary goal of our study is to investigate whether fine-tuning the existing IndicWav2Vec model with a relatively small domain-specific dataset (KrishiVaani) could effectively support rapid dataset curation. To demonstrate this, we selected the best-performing baseline model (IndicWav2Vec) and fine-tuned it with KrishiVaani data, thereby creating distinct test scenarios - Known, Unknown, and Out-Of-Domain (OOD). The overarching aim is to minimize human annotation effort and accelerate data creation.

KVWav2Vec primarily serves to evaluate whether a domain-specific fine-tuned model can expedite and enhance the data curation process effectively, particularly in conversational and unknown-speaker scenarios. While not claiming state-of-the-art performance, we demonstrate its utility in reducing human annotation effort significantly as shown in Table 3

Model	KVWav2Vec	Wav2Vec-IV
KrishiVaani-Known	22.38	23.4
KrishiVaani-UnKnown	26.04	27.69
KrishiVaani-OOD	24.61	22.38

Table 6: Comparison of different WER and CER (%) scores of KVWav2Vec with IndicVoice (IV) + KrishiVaani (KV) and IndicVoice (IV) alone

We experimented on the IndicWav2Vec model fine-tuned with IndicVoice (IV) + KrishiVaani (KV) vs IndicVoice alone, which showed modest but consistent improvements in both WER and CER across Known and Unknown test sets.

F.2 Implications for human-AI Collaboration

From a system design perspective, these results demonstrate that post-correction not only reduces the number of human edits but also changes the annotator's role. Rather than fixing every error manually, annotators now operate as *validators*, reviewing pre-corrected hypotheses and resolving only high-ambiguity cases. This shift was particularly beneficial in low-confidence regions, such as domain-specific terminology and Hindi-English code-switching, as shown in Fig. 2. The impact of this human-AI collaboration is further quantified in Fig. 6, which shows a 61.1% reduction in annotation time when using VāṇīSetu's verification pipeline compared to fully manual transcription. Intermediate configurations, such as ASR-only correction or single-pass validation, also yield substantial efficiency gains, suggesting that even partial automation has measurable benefits.

Early annotator feedback indicated that post-correction reduced fatigue and increased trust in the system's outputs. This suggests that model-aided editing, when appropriately calibrated, can improve both speed and engagement without sacrificing quality. Moreover, the use of layered quality control (via validators and verifiers) enabled distributed annotation with minimal coordination overhead, further enhancing scalability.

F.3 Annotator Feedback and Observations

To understand how the annotators interacted with the VāṇīSetu pipeline in practice, we collected

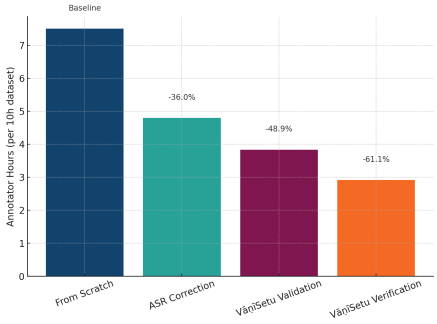

Figure 6: Annotation time (in hours per 10-hour dataset) across different stages of the pipeline. VāṇīSetu's layered design yields a 61.1% reduction in total annotation effort compared to fully manual transcription.

informal feedback from a group of eight annotators over 31 sessions. Their reflections highlight key human-AI interaction dynamics that shaped their annotation experience. Annotators widely appreciated the pre-correction capability using mT5 and LLMs, especially in reducing repetitive edits. One annotator remarked: *"mT5 has been a game-changer. Instead of fixing basic errors, I can now focus only on the more nuanced mistakes."* Another reported *"....it felt like editing a smart draft instead of transcribing from scratch."* These responses suggest a shift in role from manual transcription to semantic review, which is aligned with our design goals.

G LLM-based Post-Correction

We explore both small LMs (ByT5, mT5) and larger LLMs (Llama, ChatGPT) for automatic ASR post-correction. Our experiments indicate that smaller models (e.g., mT5) performed better than larger LLMs like ChatGPT, guiding our choice to integrate mT5 into our automatic correction pipeline. Moreover, the central objective of our work is to minimize human annotation effort and accelerate high-quality data creation for conversational ASR in Hindi.

H Data Annotators

The annotation guideline is mentioned in the KrishiVaani Github[5].

We have developed our in-house team and volunteers who help us in data curation, data validation, and data verification tasks for the KrishiVaani Dataset. We have trained them for the task and made them familiar with the VAgyojaka tool, then we ask them to do the following tasks. We have paid 5$ per hour for the data curation task and 2.5$ for both data validation and data verification tasks. All the annotators were from India.

[5]https://anonymous.4open.science/r/KrishiVaani-4C73/

I Related Work

Early work on agricultural speech technology in India emphasized voice-first access to time-sensitive information (notably commodity prices), exemplified by the Mandi Information System developed under government initiatives for multilingual spoken dialog access in rural settings (Mohan et al., 2014). Subsequent research and shared tasks highlighted that the core bottleneck is robust ASR under real rural conditions (spontaneous speech, telephony channels, background noise, and strong dialectal variation) where even strong baselines can remain around 30% WER, as demonstrated by the Gram Vaani ASR Challenge on regional Hindi (Bhanushali et al., 2022a). To address persistent data scarcity, recent efforts have pursued scale through synthetic and multimodal pipelines (e.g., AgriGPT-Omni's speech-vision-text framework with large synthetic agricultural speech resources) while simultaneously underscoring the limited availability of representative real agricultural speech (Yang et al., 2025). Complementary global datasets such as Kallaama (field-collected agricultural speech in Senegalese languages) reinforce that agriculture remains a low-resource domain internationally, where ecological validity depends on in-field capture and careful annotation (Gauthier et al., 2024). Finally, inclusion-focused studies (e.g., rural Bhojpuri women) show that transcription is itself a fundamental challenge due to missing standard guidelines, scarcity of domain-aware transcribers, and substantial regional variation, motivating agriculture-specific corpora with rigorous, standardized transcription and quality-control practices (Joshi et al., 2025). Collectively, these efforts motivate agricultural speech datasets and pipelines that prioritize (i) real, noisy speech; (ii) systematic transcription guidelines and quality control; and (iii) domain phenomena (terminology, entities, code-mixing, numerals, and dialectal variation).

J Additional Languages

Although our VāṇīSetu configuration is developed with Hindi in mind, we also evaluate the transferability of its LM-based post-correction to other

Language	IndicConformer	ByT5	mT5
Marathi	**25.55**	26.02	**25.12**
Telugu	**23.28**	24.72	**22.05**

Table 7: LM-based correction results on Marathi and Telugu subsets of IndicTTS.

Indic languages. Table 7 reports WER on Marathi and Telugu subsets of IndicTTS, comparing the raw ASR hypothesis to outputs corrected by ByT5 and mT5. Across both languages, **mT5** achieves the lowest WER, suggesting that LM-assisted post-correction in VāṇīSetu generalizes beyond Hindi to other low-resource Indic settings. We use IndicTTS for this analysis because its linguistic characteristics are closer to KrishiVāṇī-style conversational speech, enabling a consistent evaluation under comparable conditions. Motivated by these results, we are also applying VāṇīSetu to build datasets in Marathi and Telugu.

K System Adaptability and Reuse

VāṇīSetu is designed to be modular: each stage, from ASR to correction to validation, can be reconfigured for other domains or languages. Additionally, correction logs and verification patterns can be used to fine-tune future models, enabling a feedback loop between human annotators and system improvement. This framework enables speech corpus construction to move from isolated, static annotation efforts toward a scalable, adaptive, and co-created process grounded in real deployment contexts.

L Compute Infrastructure

Compute details: For all our pre-training and fine-tuning experiments, we used two NVIDIA A100-SXM4-80GB GPUs. Each training requires 4-48 hours.
Software and Packages details: We implement all our models in PyTorch[6]
VAgyojaka Tool details: We developed the data validation and data verification task in VAgyojaka tool using the QT 6 framework[7]. **Models**
mT5: mT5-small (300M parameters), mT5-base (580M parameters)
ByT5: ByT5-small (300M parameters), ByT5-base (580M parameters)
Nanda: LLaMA3-10B
GPT-4o mini: 8B parameter

[6]https://pytorch.org/
[7]https://www.qt.io/product/framework

M Prompt Used

ChatGPT Prompt

Example 1:
You are given an ASR hypothesis of a spoken utterance. The hypothesis may contain misrecognized words, incorrect word segments, or code-switching mistakes. Your job is to produce the best possible corrected text, relying on your knowledge of grammar and typical usage
Please correct any errors in
1. Incorrect transliteration of English words
2. Incorrect transliteration of English numbers
3. Incorrect transcription of native Hindi numbers
4. Misrecognition of underrepresented characters
5. Splitting of compound words
6. Incorrect word segmentation

There may be more than two errors in the ASR hypothesis. Output only the final corrected output (no extra commentary)
Hypothesis: ratha yātrā ke lie jānabūjhakara vāna ṭyūreṣṭa dvārā taitālīsa minaṭa kī derī kī gaīhai

Predicted Output: