# 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 conversational dataset derived from agricultural discourse on YouTube. The framework combines automatic speech recognition (ASR), lightweight and large-scale language model-based post-correction, and structured human validation through an enhanced annotation tool, Vāgyojaka. Our experiments show that domain-specific fine-tuning improves ASR accuracy on real-world speech, and that small language models like mT5 provide low-latency corrections that meaningfully reduce annotator burden. By transforming annotators from manual transcribers to informed validators, VāṇīSetu enables scalable, efficient, and linguistically rich speech corpus creation. We discuss the broader implications for speech dataset development, choices in model integration, and human-in-the-loop annotation workflows.

## 1 Introduction

In multilingual, low-literacy contexts like rural India, speech remains the most accessible and natural mode of interaction. However, designing robust speech technologies (Deshmukh and Chalmeta, 2024) for these settings remains a challenge, not because of algorithmic limitations alone, but due to the absence of usable, domain-specific, real-world datasets (Adiga et al., 2021; Javed et al., 2024b). Although advances in ASR have brought significant gains in high-resource languages, ASR systems for Hindi and other Indic languages continue to underperform when deployed in noisy, spontaneous, and code-mixed environments, such as farm advisories, government broadcasts, or peer-to-peer dialogues among rural speakers (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 speech corpora remains an expensive, labor-intensive process, requiring human annotators to correct transcripts, resolve speaker turns, align utterances, and normalize inconsistent language, tasks that can consume 6-8× the real-time length of the audio. These high annotation costs constrain the growth of ASR in precisely those domains where it could have the greatest impact. In this paper, we present VāṇīSetu (Fig. **??**), a human-AI collaborative framework designed to support the scalable creation of domain-rich speech datasets in Indic languages. Rather than relying solely on human labor or fully automatic methods, VāṇīSetu treats corpus creation as a cooperative workflow between AI components and human annotators. It combines automatic speech transcription and diarization, post-correction using language models (LMs (Kumar et al., 2022b; Wang et al., 2022) and LLMs (Kumar et al., 2024; Ma et al., 2023; Li et al., 2024b)), and a layered human validation process through our annotation interface, Vāgyojaka (Kumar et al., 2022a). Together, these components enable annotators to focus on nuanced corrections while offloading repetitive and lower-confidence tasks to AI models.

Although designed for multilingual corpus creation, we demonstrate the framework on KrishiVāṇī, a 100-hour Hindi agricultural speech corpus capturing diverse speakers, accents, noise, and code-mixed speech. VāṇīSetu's integration of AI post-correction with human validation reduced the annotation effort by over 50%, while preserving linguistic fidelity and domain relevance[1]. Our key contributions are as follows:

- VāṇīSetu, a human-AI collaborative frame-

[1]Code and models, KrishiVāṇī dataset: https://anonymous.4open.science/r/KrishiVaani-2463, modified Vāgyojaka tool: https://github.com/vagyojaka/VAgyojaka

work for efficient speech corpus construction, which integrates ASR, automatic post-correction, and layered human validation in a unified, extensible workflow.

- KrishiVāṇī, a 100-hour Hindi conversational speech dataset from the agricultural domain, capturing real-world speaker variation, ambient noise, and dense code-mixing across rural and semi-urban contexts.
- An enhanced version of the Vāgyojaka annotation tool, redesigned to support multi-stage correction workflows, annotator feedback and incentive mechanisms, and scalable annotation operations across diverse speech tasks.
- A comparative analysis of LM/LLM-based post-correction strategies, highlighting how small LMs (ByT5, mT5) and large LLMs (ChatGPT, LLaMA) impact annotation effort, transcript quality, and system responsiveness.

By reimagining data annotation as a site of collaborative intelligence, this work contributes new design insights for building speech systems that are not only scalable but also inclusive, situated, and culturally grounded.

## 2 The VāṇīSetu Framework

Building robust corpora in low-resource, code-mixed settings requires scalable processing with human oversight, as manual pipelines are accurate but slow, while automated ones are efficient but fail to generalize to noisy or informal speech. To balance these trade-offs, we introduce **VāṇīSetu**, a four-stage human-AI framework that integrates ASR, automatic correction, and human validation (Fig. 1).

### 2.1 Data Collection Stage

The first stage involves domain-guided collection of raw data. VāṇīSetu begins by identifying the target speech domain, in our case, Hindi-language content relevant to agriculture, and constructs a bilingual keyword list drawn from domain glossaries, Wikipedia, and curated Hindi-English lexicons. These keywords are used to query YouTube, prioritizing long-form, unscripted videos with conversational speech. The retrieved videos are filtered for duplicates, downloaded using open-source scraping tools, and converted to 16 kHz mono-channel WAV format. This process yields raw, unsegmented audio, characterized by real-world variation, including spontaneous phrasing, regional accents, speaker overlaps, and ambient noise. These characteristics make the corpus both challenging and representative of deployment conditions in low-literacy, mobile-first settings.

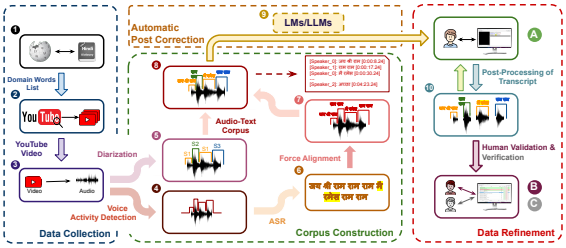


Figure 1: The VāṇīSetu pipeline for constructing the KrishiVāṇī conversational Hindi speech corpus. The system combines domain-guided data collection (Steps 1-3), corpus construction (Steps 4-8), LM/LLM-based post-correction (Step 9), and layered human-in-the-loop validation (Steps A-C) to create a high-quality,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

### 2.2 Corpus Construction Stage

This stage converts raw audio into a structured corpus through the following automated processing steps:

**Voice Activity Detection (VAD)** is applied to remove silence and background-only regions, segmenting the audio into utterances of $\leq 20$ seconds.

**Speaker Diarization** is performed using pretrained models (e.g., PyAnnote (Bredin, 2022)) to label speaker turns. This preserves conversational structure and enables speaker-wise annotation.

**ASR Transcription** is performed using both general purpose (IndicWav2Vec (Javed et al., 2023)) and our fine-tuned (KVWav2Vec) ASR models. Transcripts are generated through beam search decoding to improve fluency and coverage.

**Forced Alignment** is used to map the transcript tokens to their audio spans using a CTC-based aligner (Watanabe et al., 2017). This alignment supports efficient audio navigation and post-editing within the annotation interface.

### 2.3 Automatic Post-Correction by LM/LLMs

This stage introduces automatic post-correction, where raw ASR output is refined using lightweight or large language models before human review, reducing annotator effort by fixing common errors and improving fluency.

**LM-based Correction:** We fine-tune small models such as mT5 (Xue, 2020) and ByT5 (Xue et al., 2022) in Hindi ASR error-correction pairs. These subword- and byte-level models are effective in correcting lexical distortions, mis-transliterated terms, and token boundary errors.

**LLM-based Correction:** We also evaluate larger models such as LLaMA-3-Nanda-10B (Touvron et al., 2023)and ChatGPT-4o (Ma et al., 2023) using fine-tuned (Li et al., 2024a; Hu et al., 2024) and in-context learning (Ghosh et al., 2024) respectively. These LLMs handle domain-specific or code-switched utterances but exhibit slower inference and occasional hallucinations.

The framework allows choosing any LM or LLM for post-correction. For KrishiVāṇī, we use mT5, and its corrected transcripts are then passed to the final human-in-the-loop refinement stage.

### 2.4 Data Refinement Stage

The final stage covers post-processing and proofreading of the LM- or LLM-corrected corpus. As shown in Figure 1, three roles are involved: annotators (A) for manual corrections, validators (B), and verifiers (C). We extend the open-source **Vāgyojaka** tool with online and offline features, allowing annotators to review ASR and corrected outputs, listen to audio, and apply corrections (Fig. 2). The interface provides the following support for the annotators:

- In-place editing of transcripts alongside diarization and audio metadata.
- Transliteration of embedded English words to Hindi, with in-tool suggestions.
- Normalization of punctuation, spelling variants, and stylistic inconsistencies.
- Flagging of problematic segments for additional review.

**Quality control** is layered through a reward-linked, role-based model:

**Validators** recheck corrections and earn incentives for retained edits.

**Verifiers** perform final approval, enabling consistent inclusion criteria across batches.

Combining automation with human discretion ensures that both linguistic precision and annotation efficiency are maintained.

## 3 Experimentation

We design experiments to study how VāṇīSetu affects ASR performance and human effort. We begin with a description of the KrishiVāṇī dataset and then present our experimental setup, covering (i) its impact on Hindi ASR, (ii) the effectiveness of LM- and LLM-based post-correction, and (iii) the resulting reduction in human effort. Section 4 presents and discusses the results.

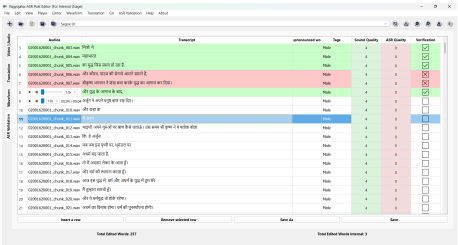

Figure 2: Performing data validation and data verification through the Vāgyojaka tool.

### 3.1 KrishiVāṇī Corpus

The **KrishiVāṇī** corpus comprises 100 hours of Hindi agricultural-domain speech, curated using the VāṇīSetu framework. Its 2.5-hour test set is partitioned to support robust evaluation across diverse deployment conditions into:

**KrishiVāṇī-Known**: Test speakers also appear in the training set (in-domain, familiar).

**KrishiVāṇī-Unknown**: Test speakers are disjoint from the training set (in-domain, novel).

**KrishiVāṇī-OOD**: Contains speech from entirely unseen domains and speakers (out-of-domain).

### 3.2 Baseline ASR Models

We build a domain-adaptive ASR model, **KVWav2Vec**, by fine-tuning IndicWav2Vec on 75 hours of Hindi speech (65h IndicVoice, 10h KrishiVāṇī) and compare it against strong Hindi ASR baselines:

**IndicWav2Vec** (Javed et al., 2023): Multilingual wav2vec 2.0 model fine-tuned on Hindi dataset with CTC loss.

**IndicWhisper** (Bhogale et al., 2023b): Whisper-based ASR model fine-tuned on Hindi dataset with byte-level BPE.

**SALSA** (Mittal et al., 2024): Hybrid model with Whisper encoder and Llama2-7B(Touvron et al., 2023) decoder for low-resource Hindi ASR.

**IndicConformer** (Javed et al., 2024a): The Conformer-based model fine-tuned on Hindi using subword-character tokenization.

**SeamlessM4T** (Barrault et al., 2023): A large-scale multilingual ASR and translation model

| 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 1: Models Comparison across different KrishiVaani datasets

| Model | KVWav2Vec | ByT5 | mT5 | Llama |
|---|---|---|---|---|
| **KrishiVaani-Known** | 22.38 | 24.33 | **22.29** | 26.37 |
| **KrishiVaani-UnKnown** | 26.04 | 26.77 | **25.70** | 37.75 |
| **KrishiVaani-OOD** | 24.61 | 25.12 | **24.35** | 30.08 |

Table 2: Comparison of different WER (%) scores of KVWav2Vec with other fine-tuned LMs/LLMs.

trained on 4.5M hours of speech.

### 3.3 Post-Correction using LMs and LLMs

We describe the use of fine-tuning and in-context learning for ASR post-correction using LM or an LLM. The candidate models considered are:
**mT5-small** (Xue, 2020): Multilingual encoder-decoder with subword tokenization.
**ByT5-small** (Xue et al., 2022): Tokenizer-free variant of mT5 using UTF-8 bytes.
**Llama-3-Nanda-10B** (Touvron et al., 2023): **(Llama)** Hindi-focused LLM trained on 65B Hindi tokens.
**ChatGPT-4o mini** (Ma et al., 2023): Used for zero-, one-, and few-shot in-context correction; SBERT embeddings (Joshi et al., 2023) guide selection of examples.

## 4 Results and Discussions

We now analyze how the VāṇīSetu framework affects system accuracy and human experience across multiple layers of the corpus creation process. Our findings highlight how model selection, post-correction strategy, and workflow design collectively shape the quality of the pipeline.

As shown in Table 1, our domain-adapted ASR model **KVWav2Vec** consistently outperforms state-of-the-art baselines on the **KrishiVāṇī-Unknown** and **KrishiVāṇī-Known** test splits. These gains suggest that even a modest amount of domain-specific conversational data (10h) can significantly improve recognition quality, especially for informal, code-switched, or accented Hindi speech. On the out-of-domain (OOD) split, IndicWav2Vec slightly outperforms KVWav2Vec, indicating that generalist models retain value when faced with topic drift or unseen speakers. However, the performance gap is small (CER: 8.51 vs. 9.51), and the domain-tuned model remains highly competitive. These results validate our design choice to blend open-domain and agricultural speech data during fine-tuning, enabling broader generalization while preserving domain fluency.

To reduce annotation overhead, we evaluated language model-based post-correction strategies that act as a first-pass filter before human intervention. As seen in Table 2, surprisingly, smaller models such as **mT5-small** and **ByT5-small** outperformed larger LLMs in both accuracy and latency. mT5, trained on Hindi ASR correction pairs, captured common morphological patterns and repaired character-level hallucinations with minimal delay. ByT5 showed particular strength in handling token boundaries and spelling variants, which are frequent in noisy ASR outputs. For additional analysis, see Appendix D.

## 5 Conclusion

In this work, we presented VāṇīSetu, a human-AI collaborative framework for constructing high-quality speech corpora in low-resource settings, and demonstrated its effectiveness through the creation of KrishiVāṇī, a Hindi conversational dataset in the agricultural domain. By combining ASR, language model-based post-correction, and layered human validation via the Vāgyojaka interface, VāṇīSetu reduces annotation effort while preserving linguistic fidelity. Our results show that lightweight models like mT5 offer a strong balance of accuracy and responsiveness, while the pipeline as a whole enables annotators to shift from manual correction to targeted validation, enhancing both speed and user experience. More broadly, this work contributes a reusable, domain-adaptive design pattern for speech data creation in underrepresented languages, emphasizing modularity, scalability, and human agency as central to building inclusive AI systems.

## Limitations

While 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 under-represented 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*.

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*.

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*.

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.

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*.

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*.

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.

## A Detailed Pipeline for KrishiVaani

This section describes the generation of the KrishiVaani speech corpus through our custom pipeline. Figure 1 highlights the design of this multi-stage pipeline involving three stages of data collection, speech corpus construction, and data refinement, respectively. The following subsections give a detailed overview of each stage:

### A.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[2]. 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 these videos 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 channel *wav* format and 16 kHz sampling rate.

### A.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 duration of 20 sec and average silence detection using mp3split[3]. 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.

[2]We follow YCC licensing.

[3]http://mp3splt.sourceforge.net/mp3splt$_{page/home.php}$

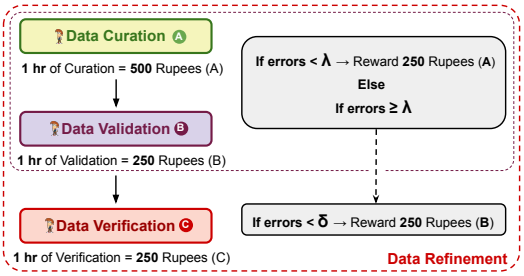


Figure 3: The reward system adopted to encourage data curators and validators. Alongside standard compensation, we used $\lambda$ as 5% and $\delta$ as 2% of the total number of words for enforcing the reward system while curating the KrishiVaani dataset.

**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.

### A.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. 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 3, whenever any data validator makes edits less than a certain threshold ($\lambda$), 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 ($\delta$), 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.

## B 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.

## C LM/LLM Models Details

**1. mT5**: mT5 (Xue, 2020) belongs to the family T5 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.

| 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 3: Comparative overview of different WER (%) scores for different settings using in-context learning through ChatGPT

# D Additional Result

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 3, 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.

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

Table 4: Latency (in seconds) of different models for ASR post-correction.

For this reason, we integrated **mT5-small** as the default correction engine in Vāgyojaka, prioritizing low-latency, high-consistency performance over marginal accuracy gains from LLMs (see Table 4 for comparative latency results).

## D.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.

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

Table 5: Comparison of different WER and CER (%) scores of KVWav2Vec with IndicVoice (IV) + KrishiVaani (KV) and IndicVoice (IV) alone

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 1

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.

## D.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. 4, 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.

## D.3 Annotator Feedback and Observations

To understand how the annotators interacted with the VāṇīSetu pipeline in practice, we collected informal feedback from a group of eight annota-

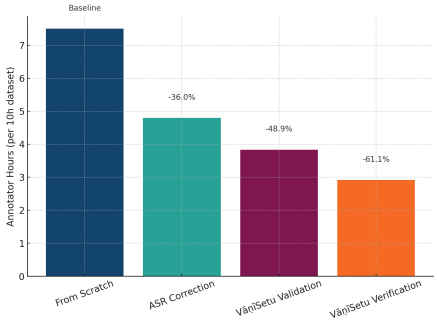


Figure 4: Annotation time (in hours per 10-hour dataset) across different stages of the pipeline. VāṇīSetu's layered design yields a 61.1% reduction in total annotation effort compared to fully manual transcription.

tors 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.

## E 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.

## F Data Annotators

The annotation guideline is mentioned in the KrishiVaani Github[4].

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.

## G 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.

## H 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[5]
**VAgyojaka Tool details**: We developed the data validation and data verification task in VAgyojaka tool using the QT 6 framework[6].

## I Prompts

[4]https://anonymous.4open.science/r/KrishiVaani-2463/

[5]https://pytorch.org/

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

**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 ṭyūresṭa dvārā taitālīsa minaṭa kī derī kī gaīhai

**Predicted Output:**