Sarvam Inter-IIT Problem Statement

Real-time Hindi → English Speech-to-Speech Translation

In one line	Build the training data, then fine-tune a model that listens to live Hindi speech and speaks the English translation while the speaker is still talking, with meaning, numbers and names intact.
Source	Hindi speech, freely mixed with English
Target	English speech
Resources	Each team gets USD 1,000 of Modal credits (H100 GPUs) and USD 1,000 of Sarvam API credits, usable on all Sarvam APIs.

1. About Sarvam

Sarvam AI builds language and speech models for India, designed for Indian languages as people speak them, often mixed with English.

2. The problem

A caller speaks Hindi. The listener on the other end understands only English and needs to hear the translation while the caller is still speaking. Four things make this hard:

- **Word order.** Hindi usually puts the verb last, while English needs it early.
    - In “मैंने कल अपना पुराना बैंक खाता बंद करवा दिया” (“Yesterday I got my old bank account closed”), the verb phrase “बंद करवा दिया” (got closed) comes last.
    - Until it is heard, the system cannot tell whether the account was opened, closed or only checked.
- **No take-backs.** Speech that has been played cannot be withdrawn, so the system must decide when it has heard enough to speak.

- **Details.** Amounts in lakh and crore, account digits, names, dates, negations and self-corrections must be translated correctly.
- **Little data.** Public Hindi–English data with speech on both sides is scarce, so you will build your own.

3. What you must build

There are two phases:

1. **Before the mid-term,** you build the data.
2. **After the mid-term,** Sarvam gives you a base speech-to-speech checkpoint with inference code. You fine-tune it on your data and submit it as a streaming service.

Part A (before the mid-term): the data engine

A speech-to-speech model takes **Hindi audio as input** and produces English audio, both mono at **24 kHz**. Build a pipeline that produces the Hindi side of the training data and its translations, and deliver the resulting dataset (the "Part A dataset").

Each record contains the following.

- **Hindi audio**
  - Mono, 16-bit, 24 kHz WAV.
  - Real recordings, synthetic speech, or both.
  - Real recordings only from datasets whose licence allows redistribution.
  - One speaker per clip.
  - Licence and consent compliance is your responsibility.
- **A verbatim Hindi transcript**
  - Exactly what was said, following IndicVoices Level 1 ("transcribed as spoken"). Disfluencies, repetitions and fillers are optional and not scored. If you leave them out, the remaining words must still be aligned correctly.
  - English words are written in Devanagari as spoken, without bracketed Latin forms.

    - Numbers are written as spoken words.
    - Every sentence ends with sentence-final punctuation.
    - Noise tags are optional.
    - Manual correction is allowed.
- **Hindi word-level alignment**
    - The start and end time of every word in the transcript, including English words written in Devanagari.
    - It may or may not be used in training.
    - It is audited at the mid-term.
- **Sentence-level English translations**
    - These become the targets your model learns to speak.
    - Think about what makes the best target for a streaming speech-to-speech model: how literal the translation is, and how sentence boundaries map.
    - Write numbers in words. Digits are also accepted.
    - Amounts keep the speaker's units (lakh, crore).
    - A sentence may carry several English translations.
    - You may keep the speaker's disfluencies in the English too: fillers as "um" or "uh", repeated words repeated, and cut-off words with a hyphen (for example "cl- close"). They are not scored.
    - How you translate is up to you.

If you use TTS-generated speech anywhere, check it and filter or regenerate bad output. TTS can mispronounce names and numbers, skip or repeat words, or add artefacts.

Requirements

- **At most 200 hours** of Hindi audio. You do not have to reach this limit. Hours are counted as follows:
    - **Real recordings:** the union of the source-file intervals you use. Overlapping or repeated cuts of the same recording count once.

    - **Synthetic audio:** every distinct generated file counts, including the same text rendered in another voice.
    - **Augmentation:** augmented copies of a clip (noise, speed, reverb and so on) are not part of the dataset. Augmentation applied during training adds no hours.
- **Clips** of at most 60 seconds.
- **Metadata** in the format of §5.2.
- **A reproducible pipeline** that rebuilds every record from the source audio plus your archived transcripts, translations and TTS outputs.
    - The rebuilt output should be byte-identical where possible.
    - Equivalent output is accepted where exact output is not possible, for example because of LLM non-determinism.

Part B (after the mid-term): fine-tune the base checkpoint

After the mid-term, every team gets the same **base checkpoint**, a streaming Hindi → English speech-to-speech model trained on synthetic data, with inference code.

- **Fine-tuning is mandatory.** Your submitted system must be this checkpoint with weights you have fine-tuned. Any fine-tuning method is allowed, and adapters count.
- **Only your Part A dataset may be used for training.**
    - Turn its Hindi audio and translations into training pairs. How you arrange the English against the Hindi is up to you.
    - All English target audio in supervised training pairs must be synthesised with Bulbul: `bulbul:v3`, voice `shubh`, `language_code="en-IN"`, 24 kHz. Pace and temperature are free, but declare them.
    - Training pairs need word-level timings for the English audio. Bulbul does not return them, so align the English audio yourself. Like the Hindi word times, they must be ordered, non-negative and inside the English target audio.

- After the mid-term you may keep changing the dataset, including adding, removing or replacing records. The final dataset must still meet Part A, including the 200-hour cap.

- **The fine-tuned checkpoint must do the translation.** It takes the Hindi audio and generates the English speech itself.
  - **Allowed around it:**
    - audio pre-processing, such as resampling, gain, denoising and voice-activity detection;
    - decoding and sampling logic (decoding settings are free), including a draft model for speculative decoding whose proposals the fine-tuned checkpoint verifies;
    - post-processing of the output audio that does not change its words, such as loudness normalisation or silence trimming.
  - **Not allowed (a cascade):** any component outside the checkpoint that produces Hindi or English text from the input audio, or English text or speech by any other route, and passes it to the checkpoint or to the listener, or uses it to steer the output. This includes:
    - speech recognition, machine translation or LLM output fed to the checkpoint as a prompt or conditioning;
    - an outside model that transcribes or translates the input or output to choose, rerank or time the checkpoint's output;
    - an external TTS speaking a translation;
    - any other speech-translation model used in place of the checkpoint or alongside it.
- **Training code** is yours to write.
- **Submission:** the fine-tuned model as a streaming service in a container (§5.1).

4. Language and conditions

Item	Specification
Source	Hindi, with English words, phrases or clauses mixed in
Target	English speech
Audio	Mono 16-bit PCM at 24 kHz, in both directions. Lower-bandwidth recordings are upsampled.
Recordings	One speaker per session. Background noise and telephone audio may occur.
Session length	10 to 60 seconds
Content	Names, places, organisations, amounts (rupees, lakh, crore), numbers, dates, times, phone and account numbers, products, negations, conditions and self-corrections

5. Interfaces

5.1 Streaming speech-to-speech interface

Your system runs as a WebSocket server inside your container, and the organiser's harness is the client. One connection is one session.

Client → server

Message	Format	Meaning
`session_start`	JSON text frame	`{"type": "session_start", "session_id": "s-0001", "sample_rate": 24000, "encoding": "pcm_s16le", "source_language": "hi", "target_language": "en"}`
audio	Binary frames	Hindi audio as raw PCM with no WAV header, **80 ms per frame** (3,840 bytes), sent in real time. The first frame goes out at $t = 0$, right after `session_start`; frame k goes out at $t = k \times 80$ ms. The last frame is zero-padded.
`input_end`	JSON text frame	`{"type": "input_end", "session_id": "s-0001", "valid_samples": 1234560}`, sent right after the last Hindi frame. `valid_samples` is the number of real Hindi samples sent; everything after them is zero-padding.
silence	Binary frames	After `input_end`, silent 80 ms frames continue in real time until `session_complete` arrives

Server → client

Message	Format	Meaning
audio	Binary frames	English speech as raw PCM (mono, 24 kHz, `pcm_s16le`, no WAV header), at most 1 second per frame
`session_complete`	JSON text frame	`{"type": "session_complete", "session_id": "s-0001"}`, sent once, when your translation is finished

Protocol rules

1. **Playback.**
   - The harness plays your audio in the order received, as a listener would hear it.
   - Each frame starts when it arrives or when the previous one finishes, whichever is later. Gaps are heard as silence.
   - All timing is measured on this playback timeline.
2. **No take-backs.** Audio, once sent, is final.
3. **Finishing.**
   - After `input_end`, finish translating, then send `session_complete` within 30 seconds of `input_end`.
   - Speech after the speaker stops counts toward latency.
4. **Keeping up.** The harness never waits for your server. If a frame cannot be delivered within 1 second of its scheduled time because your server is not reading, the session fails.
5. **Format.** The only server-to-client text frame is one `session_complete`, and nothing may follow it.
6. **Timing.**
   - The server must accept the connection within 10 seconds.
   - Sessions run one at a time, with a 2-second gap between them.
7. **Failed sessions.** A session fails if it breaks rule 3, 4, 5 or 6, or closes early. It gets the worst value on every metric.

Container

- The image's default entrypoint starts the server on port 8000, path `/ws`.
- `GET /health` returns HTTP 200 within 300 seconds of start-up.
- All weights and dependencies must be inside the container.

5.2 Part A dataset format

Share the dataset privately with Sarvam as Hindi audio plus a JSONL manifest, one record per line:

```
{
  "id": "rec-000123",
  "audio": "audio/rec-000123.wav",
  "duration": 41.36,
  "speaker_id": "spk-0042",
  "source": {"corpus": "...", "source_file": "...", "start":
132.48, "end": 173.84, "synthetic": false},
  "sentences": [
    {"hi_text": "<verbatim Hindi>", "hi_start": 0.42, "hi_en
d": 4.10,
     "en_text": "...", "en_alt": ["..."]}
  ],
  "hi_words": [{"w": "मैंने", "start": 0.42, "end": 0.71}],
  "translation_method": "..."
}
```

- `source.start` and `source.end` locate the clip in its source file and are used to count unique hours.
- If a clip joins several source intervals, `source` may be a list of segments in playback order, each with the same fields. All segments must come from the same speaker. Hours count the union of all segments' intervals.
- For synthetic Hindi:
  - `source.corpus` names where the Hindi text came from;

    - `source.source_file` is the generated audio file;
    - `source.start` and `source.end` locate the clip in the generated file (0 and the clip duration when the file is the clip);
    - `source.model` and `source.voice` give the generating model and voice.
- `speaker_id` and `en_alt` (alternative translations) are optional.
- No `source` field may be left empty.
- All other times are in seconds from the start of the record's audio. Word times must be ordered, non-negative and inside the clip.

6. Evaluation setup

- **Submission.** You submit a container, which Sarvam runs on hidden Hindi test audio. You receive only your own score summaries.
- **Hardware.** One NVIDIA H100 80 GB GPU, 16 CPU cores, 128 GB RAM, no network access. The machine is identical for every team.
- **Hidden test.** Real Hindi recordings as in §4.
- **Container check.** Opens when the base checkpoint is released. It gives pass/fail results against §5, and submissions are unlimited.

7. KPIs

All final KPIs are measured on your system's English speech.

KPI	What it measures
Translation quality	How well the English speech matches human reference translations
Intent accuracy	Whether the speaker's intent survives: what is asked, refused, confirmed or corrected, negations, conditions, and who does what to whom
Entity accuracy	Whether names, places, organisations, amounts, numbers, dates and products come out exactly right
Latency	How far the English speech trails the speaker, and how soon it finishes after the speaker stops

KPI	What it measures
Audio quality and naturalness	How clean and natural the English speech sounds, as rated by listeners
Dataset audit (mid-term)	The quality of sampled Part A records: transcripts, Hindi word timing, translations and format

8. Gates

Mid-term. The Part A dataset earns points only if it meets §3 Part A and §5.2.

Final. If any gate fails, the final-round KPI points (70) are zero. The report and presentation are still graded.

- **G1.** The container meets §5.
- **G2.** Fine-tuning evidence:
  - your fine-tuned weights;
  - training logs;
  - training-pair manifests (any layout). Each training pair lists its Part A record ID, the target text used and the Bulbul settings;
  - scores before and after fine-tuning.
- **G3.** The final Part A dataset meets §3 Part A.

9. Evaluation matrix and deliverables

Evaluation matrix (100 points)

Stage	Component	Points
Mid-term	Dataset audit	12
	Data-pipeline reproducibility	3
	Mid-term report	5
Final	Translation quality	12
	Intent accuracy	18
	Entity accuracy	18
	Latency	16

Stage	Component	Points
	Audio quality and naturalness	6
	Final report	5
	Presentation and live demo	5
	Total	**100**

- **KPI components** are scored relative to the other teams and the unchanged base checkpoint.
- **Reports and presentations** are graded by the Sarvam judges on technical depth, rigour and clarity.

Deliverables

Milestone	What to submit	Format
Mid-term	Part A dataset; data-pipeline code; mid-term report	Dataset: `.wav` audio and a `.jsonl` manifest (§5.2) in a private Hugging Face dataset repo shared with Sarvam, with its revision (commit hash). Code: Git commit hash. Report: `midterm_report.pdf`
Final	Container; fine-tuned weights; data-pipeline, training and training-pair code; training logs and training-pair manifests; `TRAINING.md`; `model_card.md` and `data_card.md`; final Part A dataset; final report (including scores before and after fine-tuning); slides	Container: `.tar` Docker image or registry digest. Weights and dataset: private Hugging Face repos (dataset as in §5.2), each with its revision (commit hash). The dataset revision must be the one used for training. Code, logs, manifests and `.md` files: in the repository, Git commit hash. Report and slides: `final_report.pdf`, `slides.pdf`

Milestone	What to submit	Format
Presentation	A 10-minute talk and 5-minute Q&A with a live demo: Hindi speech into your deployment, English speech back. A recorded fallback (≤ 3 min) may be used only if the live demo fails.	`slides.pdf`, at most 12 slides

Page limits. Pages beyond the limit are not read. The title page and table of contents do not count.

- Mid-term report: at most 4 pages, plus 1 page of references.
- Final report: at most 8 pages, plus 2 pages of references and 4 pages of appendix.

Extra files.

- Only the items above may be submitted, plus one optional, ungraded demo video (`.mp4`, at most 3 minutes).
- Source, configuration and dependency files inside your repository or container are expected. A Hugging Face dataset holding your source audio (§11) is also allowed.
- Any other file is not permitted and attracts the standard penalty deductions enforced by the organising team.

The reports must cover:

- **Mid-term:** the data pipeline, its quality checks and failure modes, and the dataset's statistics.

- **Final:** how you built training pairs, the fine-tuning method and its effect against the base checkpoint, the latency–accuracy trade-off, and an analysis of entity and number errors.

10. Rules

1. **Resources.** Any open-weight model may be used to build the Part A dataset. Real recordings must come from datasets whose licence allows redistribution. Declare everything in `model_card.md` and `data_card.md`. Licence and consent compliance is your responsibility.
2. **Closed or paid APIs**, including Sarvam's, may be used to build data, if declared. They must not be part of the submitted system.
3. **Sarvam API outputs.**
   - Sarvam permits outputs generated with its APIs using the competition credits to be used to build your data, to train and evaluate models for this competition, and to be shared with Sarvam.
   - Do not publish or redistribute them.
4. **Base checkpoint.** Use it, and models derived from it, only for this competition. Do not redistribute it.
5. **Intellectual property.** Sarvam holds all intellectual property in the code you submit, including your training code, and may use it as it sees fit.
6. **Causality.** The system may use only audio it has already received. Reading future audio, fingerprinting test recordings or accessing the network leads to disqualification.
7. **Reproducibility.** Sarvam may ask any top-five team to re-run part of its training or data pipeline from the frozen commit. A result that cannot be reproduced leads to disqualification.
8. **Scorer integrity.** Output designed to game the evaluation rather than translate can lead to disqualification. Examples: filler speech, padding, or reading out alternatives the speaker did not offer. Disfluencies that mirror the speaker's own are not filler speech.

9. **Communication.** Send questions through the doubt form or Discord. Sarvam answers daily.

11. FAQ

Will there be sample data?
Yes. Sarvam shares sample Part A records and a format validator before the mid-term.

Do we need to host the source audio?
You can put the source audio you use in a Hugging Face dataset and reference it, with the part you used, in `source.corpus` and `source.source_file`.

When do the credits arrive?
Part of the Modal credits comes about two weeks after the PS release, and the rest after the mid-term. Sarvam API credits are shared before the mid-term.

Are the credits enough for full fine-tuning?
How you use them, including whether to fine-tune fully, is up to you.

What do we learn about the base checkpoint before the mid-term?
Only what §3 Part B says. Its architecture, size, framework and weight format, its inference code and the reference WebSocket client are shared after the mid-term.

May we quantise or prune the model, add layers or heads, or decode with a draft model?
Yes, as long as the fine-tuned checkpoint still does the translation (§3 Part B). A draft model may only propose tokens that the checkpoint verifies, as in speculative decoding.

May we use external models during training?
Yes, for example to align the English audio, as reward models or as teachers. The cascade rules in §3 Part B apply only to the submitted system. Training data must still come from your Part A dataset, and English target audio in supervised training pairs must still be Bulbul `shubh`.

Do I need English word timestamps?
Not in the Part A dataset. Your training pairs need them, and Bulbul does not return them, so align the English audio yourself with any ASR or aligner.

How are numbers judged?
Equivalent written forms of the same number, such as 25,000 and 25k, both count as correct.

What if the speaker corrects themselves?
The references encode the corrected meaning. Speaking the abandoned value as fact is wrong.

Do English words in the Hindi speech need translating?
They must come out correctly in the English speech.

Is there an image-size limit, a required CUDA version or any other container rule?
No. The only container requirements are those in §5.1 and §6.

How do we test our container?
Sarvam releases a reference WebSocket client with the base checkpoint after the mid-term. Use it with the container check (§6).