Building a Sinhala Voice Pipeline for Phone Calls: From 3 to 8 Calls per GPU

Paper | 2026-09-28 | Seniru Epasinghe, Tharindu Sathsara, Dilanka Kasun, Sahan Dileepa, Bhawanthi Weerasinghe

How we connected our own Sinhala speech-to-text, a language model and our own text-to-speech into a phone assistant that answers in about five seconds, and how many calls it can carry.

Abstract

A phone call is a chain: hear, understand, think, speak. We built that chain for Sinhala from our own parts: speech-to-text, a language model, and text-to-speech, all served from shared GPU servers rather than one copy per call. This report shows the design, the speed-ups that mattered (a 3x faster speech-to-text engine, and splitting the two speech models onto two cards), and measured results: about 8 simultaneous calls comfortably, a typical wait of 5 seconds or less before the reply starts, and zero errors in tests.

Key findings

- Loading speech models inside every call process would take about 2.5 GB of memory per call. Serving them once from a shared server removes that limit
- Switching the speech-to-text engine to an optimised runtime made it about 3x faster with identical transcripts: a 6 second utterance went from 1.92 s to 0.67 s
- One card carried about 3 calls comfortably. Two cards, one per model, carried about 8
- At 8 simultaneous calls: typical wait 4.9 s, slowest 1 in 20 waited 8.3 s, no errors, 2% of replies had a short pause
- At 12 simultaneous calls the line was overloaded (28% of replies paused)
- The text-to-speech card fills up first (59% busy against 16% for speech-to-text)
- Last month's busiest moment, 9 calls at once, is at the limit of one pair of cards. More capacity is added by adding cards

1. The call as a chain

When a farmer speaks, four things happen: voice activity detection decides they have finished; speech-to-text writes down what they said; a language model decides the answer and which tools to use; text-to-speech turns the answer into a voice. The caller feels only one number: the time from the end of their sentence to the first sound of the reply.

2. Design: load the models once

The voice agent platform runs one operating-system process per call. If each process loaded its own speech models, twenty calls would mean twenty copies of each model and twenty graphics-card contexts, and a 16 GB card would hold about five calls before memory ran out.

So we split the system in two:

- **Speech servers** hold one copy of each model on the GPU.
- **Call workers** run on ordinary servers and hold no models, so we add workers as calls grow.

Every client reaches the speech servers through a single gateway that checks licence keys, limits lines per package and meters usage.

3. A faster speech-to-text engine

Moving speech-to-text to an optimised inference runtime gave identical transcripts and about 3x lower latency:

Caller spoke	Before	After
3 s	0.86 s	0.35 s
6 s	1.92 s	0.67 s
10-30 s	4.6-6.4 s	1.5-2.0 s

4. Two cards beat one

On a single card the two models wait for each other. Giving each its own card nearly tripled capacity.

Calls at once	Typical wait	Slowest 1 in 20	Replies with a pause	Verdict
2	3.7 s	6.0 s	0%	Comfortable
6	4.4 s	7.8 s	0%	Comfortable
8	4.9 s	8.3 s	2%	Comfortable
9	6.1 s	9.7 s	2%	Stretched
12	8.7 s	11.7 s	28%	Overloaded

Tests used real Sinhala caller clips played in real time and real agent replies from live calls, 5 minutes per level, with zero errors at every level.

5. Bugs only measurement finds

- A fixed output limit silently cut Sinhala callers off after about 7 seconds. Scaling it to each recording's length fixed it.
- Pieces of speech under about 40 characters could lose their first word.
- A failed request could still return a normal-looking empty answer, so the client treats empty text or very short audio as a failure.

6. Clarity through the full path

The voice scored 6.5% character error rate when our own speech-to-text read it back, against

11.8% for real callers.

7. What is next

More capacity by adding cards, a Tamil pipeline, and faster speech-to-text decoding, since turning speech into text now takes 2 to 3 seconds of each wait.