

Shruti: offline speech-to-text for Hindi, English and Hinglish in 29.5 MB

UDGAM LABS VERSION 1.0.0

Abstract

Shruti is a 55M-parameter speech-to-text model for Hindi, English and Hinglish, the everyday mix of the two. It writes Hindi in Devanagari and English words in lowercase Latin script, the way people type Hinglish ("मेरा order कब आएगा"). It ships as one 29.5 MB file (4-bit weights, 8-bit activations) that runs offline in Cactus Compute's open-source needle engine, on a phone, a laptop or inside a web page. Shruti is built on Cactus Whistle [1, 2], an Apache-2.0 speech model that Cactus Compute trained on European languages. We added 714 Devanagari vocabulary pieces and a Hindi language tag, then fine-tuned all 55M weights with 4-bit quantisation-aware training on 2,732 hours of openly licensed speech. The weights are released under Apache-2.0.

We ran every model in this report ourselves, with one harness: the same test clips, the same text normaliser and the same scorers. Word error rate (WER, %) is lower-is-better.

- **Hinglish.** On the MUCS 2021 Hindi-English test set (3,136 clips), Shruti's WER is 35.21, the lowest of the open models we measured. Two Whisper models fine-tuned for Hinglish write everything in Roman script, so we also scored both sides after converting them to Roman script with the same function. Shruti stays ahead, 34.32 against 43.45 and 44.94, but with the most forgiving converter the lead is narrow: 31.37 against 32.66 and 34.22.
- **Hindi, against Whisper large-v3.** Shruti's file is about 100 times smaller than OpenAI's Whisper large-v3 (3,087 MB), and on the same Hindi clips it makes fewer errors: Monsoon 20.46 against 33.29, FLEURS 20.14 against 28.88, and Lahaja 22.87 against 31.19 (first 300 clips). Whisper large-v3 is far better on English: 7.39 against 21.35 on Indian-accented English (Svarah) and 2.25 against 5.76 on LibriSpeech (first 300 clips of each).
- **Where Shruti is behind on Hindi.** Hindi-only models about 16 times larger are clearly more accurate on pure Hindi. Koyal-hi-120M scores 11.64 on Monsoon and 9.56 on FLEURS, and IndicConformer Hindi 13.18 on Lahaja; Shruti scores 20.46, 19.92 and 26.28.
- **On a phone.** On a 2021 Samsung Galaxy F62 (Exynos 9825, 6 GB of memory), offline, Shruti transcribes a sentence of about 7 seconds in about 1 second, including loading the model, and a 60-second recording in 9.5 seconds. Peak memory is 80 MB.
- **Silence.** It wrote 0 words on 100 silent clips. Whisper large-v3 wrote 538 words on the same clips.

We also describe how the Hinglish training labels were made, measure live (streaming) transcription against whole-recording transcription and automatic language detection against a fixed language, and list where the model falls short.

1. Introduction

Speech-to-text for Hindi speakers has three requirements that pull against each other.

1. **Hinglish output.** Much everyday Hindi is mixed with English words. When people type it, they usually write the Hindi in Devanagari and the English words in English: "मेरा order कब आएगा", rather than "मेरा ऑर्डर कब आएगा" or "mera order kab aayega". Speech models tend to pick one script. Hindi models usually spell English words in Devanagari, and the Hinglish models built on Whisper that we tested write everything in Roman script.
2. **On the device.** Voice notes, dictation and voice commands often should not leave the phone, for privacy, for cost (a cloud call per request) or because there is no connection. That rules out large models: the other open models in our comparison are 139 MB to 3,087 MB.
3. **Open licences.** To be used, checked and built on by others, the model, its base and its training data must allow commercial use.

Before training, we set a bar for usefulness: about 20% word errors on Hindi, from a file of at most 50 MB, faster than real time on a low-cost Android phone, with English words in Latin script. Shruti meets the size and speed parts: a 29.5 MB file, 6.3 times faster than real time on a 2021 phone. On accuracy it meets the bar on read and general Hindi (FLEURS 19.92, Monsoon 20.46) but not on accented Hindi (Lahaja 26.28).

Section 2 describes the model and section 3 the base model. Section 4 covers how we extended the base to Devanagari, and section 5 how it was trained and on which data. Sections 6 to 8 cover the evaluation harness, the results against other models and the measurements on devices. Section 9 lists the limitations and section 10 what we would do next.

Shruti (श्रुति) is Sanskrit for "that which is heard".

2. Model overview

SHRUTI	
What it is	Speech-to-text (automatic speech recognition) for Hindi, English and Hinglish
Base model	Cactus Whistle by Cactus Compute, Apache-2.0 [1]
Parameters	55M: 8 encoder and 8 decoder layers, width 512
Vocabulary	8,914 tokens: Whistle's 8,192 text pieces, 714 added Devanagari pieces and 8 language tags (Whistle's 7 and <code>hi</code>)
Input	16 kHz mono audio of any length. One pass covers up to 30 seconds; longer audio is processed in 30-second windows
Output	Text plus a language tag (<code>hi</code> or <code>en</code>). Hindi in Devanagari and English words in lowercase Latin; numbers written as words; no punctuation or capital letters
Files	<code>shruti.cact</code> , 29.5 MB (29,546,527 bytes; 4-bit weights, 8-bit activations) for the needle engine; <code>shruti.safetensors</code> , 222 MB (full-precision weights)
Runtime	Cactus needle engine [2], Apache-2.0: Python package, command line, C library for apps, WebAssembly for browsers
Measured by us on	a Mac (Apple silicon), Linux (x86), an Android phone and WebAssembly. The engine also has Windows and iOS builds; we have not tested Shruti on them
Licence	Apache-2.0, with Cactus Compute's notice kept

The Hugging Face repository [udgamlabs/Shruti](#) holds both files, the model card and the evaluation details. The GitHub repository [udgamlabs/shruti](#) holds runnable examples (Python, the command line and a browser page) and SHA-256 checksums of both files.

The table below shows the released file at work on the first clips of the Monsoon Hindi test set, in the order the set lists them, as transcribed by the needle engine (5 beams, language set to Hindi). We did not pick them for quality: only the third clip is left out, for length. Note that Monsoon's references spell English words in Devanagari, while Shruti writes them in Latin script.

REFERENCE	SHRUTI
इम्प्रूव होता है	improve होता है
हेमंत सोरेन सुने होंगे नाम	हाँ हेमंत सोरेन सुने होंगे ना
वेलकम	बैंग काम
नहीं बट वो सेफली बच गए हों तो क्या पूछेंगे	नहीं but वो safely बच रहे हों तो क्या अच्छी है
जी बिल्कुल देखिए सेकेंड टॉपिक है इसमें हाँ दुकानदारों के साथ मोल भाव करने की टिप्स साझा करना	जी बिल्कुल देखिए second topic है इसमें हाँ दुकानदारों के साथ मोल भाव करने की tips साझा करना

3. The base model: Cactus Whistle

Whistle is Cactus Compute's speech-to-text model [1, 2]. It is an encoder-decoder transformer. The encoder turns the audio into one feature vector for every 80 ms, and the decoder writes the transcript one word piece at a time, starting with a language tag. Its configuration, read from the released code and weights:

- input: 80-band log-mel features with a 25 ms window and a 10 ms hop, from 16 kHz audio;
- a three-stage convolutional stem that lowers the frame rate eightfold, to one frame per 80 ms;
- 8 encoder and 8 decoder layers of width 512, with attention that uses 8 query heads and 2 shared key-value heads;
- an input embedding shared with the output layer, and at most 320 output tokens per 30-second window.

Cactus trained Whistle on English, German, French, Spanish, Italian, Dutch and Polish [2]. It runs in needle, Cactus's C engine. Needle has builds for macOS, Linux, Windows, Android, iOS and WebAssembly, transcribes live audio in a streaming mode, can return word timestamps, and can favour given keywords.

We chose Whistle for five reasons:

1. **Size.** At 55M parameters, a 4-bit file fits well inside our 50 MB target.
2. **Licence.** Apache-2.0 lets us release the fine-tuned weights under the same licence.
3. **One file for every platform.** The same `.cact` file runs in the engine on phones, laptops and in the browser, so we ship one file and can measure that file everywhere.
4. **Quantisation built in.** Whistle's training code includes 4-bit quantisation-aware training. A fine-tuned model therefore exports without a separate compression step, and the exported file behaves like the model we trained.
5. **No invented text on silence.** The base model returns an empty transcript for silence [2], and we wanted to keep that.

What Whistle lacked was Hindi. Its tokenizer has no Devanagari pieces, so Devanagari text falls back to single UTF-8 bytes, three per character, and it has no Hindi language tag.

4. Adding Devanagari to the vocabulary

Whistle's tokenizer is a SentencePiece model [3] with 8,192 pieces and byte fallback. We appended 714 Devanagari pieces [A1, A7]: the Devanagari characters and common Devanagari subwords from our Hindi transcripts, each in a form for the start of a word and a form for inside a

word. Every new piece contains Devanagari, so English text is split exactly as before. We also added a Hindi language tag, `hi`, next to Whistle's seven tags, for a vocabulary of 8,914 tokens.

Each new token needs a row in the embedding matrix. In Whistle that row is used twice: to read the token as decoder input and, through the shared output layer, to score it as the next token.

We started each new row from what the model already knew about the same text:

- **A new piece** starts as the mean of the embedding rows of the pieces the original tokenizer splits it into. For Devanagari these are UTF-8 byte pieces, so the new piece starts where the model's byte-by-byte spelling of the same text points, rather than at random.
- The `hi` tag starts as the mean of the existing language-tag rows.

The new rows are kept as a separate block of parameters, placed so that every original token keeps its id.

5. Training (high level)

5.1 Method

- **A full fine-tune.** Shruti trains all 55M weights of Whistle, not a low-rank adapter (LoRA [4]) on top of frozen weights.
- **4-bit quantisation-aware training.** In every training forward pass, the model runs the way the engine will run it [A7]. The weights pass through needle's 4-bit quantiser, and activations and the attention key-value cache are quantised to 8 bits; the gradient passes straight through the rounding (the straight-through estimator [5, 6]). The exported `.cact` file holds the same 4-bit weights, so the file behaves like the model we trained. Every result for Shruti in sections 7 and 8 was measured on the exported file itself, in the engine.
- **Two stages.** Training started from Whistle's weights and then continued with more Indian-English speech.
- **Silence.** Training includes clips of silence and synthetic noise with empty transcripts, so that the model writes nothing when no one speaks.
- **One spelling per word.** For training targets only, Hindi words with common spelling variants (with or without a nukta or a chandrabindu) were mapped to their most frequent spelling. Test references were not changed.

5.2 Data sources

Shruti was trained on 2,732 hours of openly licensed speech [A1]. Every source allows commercial use and the publication of a trained model, and no test split was used for training. We are grateful to the creators of these datasets.

SOURCE	LICENCE	WHAT IT IS
AI4Bharat IndicVoices, Hindi [7]	CC-BY-4.0	Spontaneous and read Hindi from speakers across many districts
ARTPARK-IISc Vaani, Hindi [8]	CC-BY-4.0	Spontaneous phone speech describing images (transcribed training split)
AI4Bharat Shrutilipi, Hindi [9]	CC-BY-4.0	All India Radio news with mined transcripts
AI4Bharat Kathbath, Hindi [10]	CC-BY-4.0	Read Hindi
AI4Bharat Rural Women ASR v2, Hindi [10]	CC-BY-4.0	Rural women in Uttar Pradesh and Madhya Pradesh, extempore and read
Google FLEURS, Hindi training split [14]	CC-BY-4.0	Read Wikipedia sentences
MUCS 2021 Hindi-English [13], via dianavdavidson/MUCS-Hinglish	CC-BY-4.0 on the card; the original corpus is CC-BY-SA-4.0	Code-mixed lectures (training split)
AI4Bharat NPTEL [10]	CC-BY-4.0	Indian-English technical lectures
Mozilla Common Voice 17, English with Indian and South Asian accents [11]	CC0	Read English
LibriSpeech [12]	CC-BY-4.0	Read US English (audiobooks)

Vaani is gated on Hugging Face; we accepted its terms with the commercial-use option.

5.3 Hinglish labels

Most Hindi speech corpora write English words in Devanagari ("फ्रिज", not "fridge"). In Shruti's training data, English words in Hindi transcripts are written in Latin script using each dataset's own annotations [A1]. For example, AI4Bharat's annotators mark an English word with its English spelling in brackets ("फ्रिज [fridge]"), and Vaani's annotators use braces ("शॉप {shop}"). Sound-event tags are not turned into words, and transcripts whose annotations could not be converted safely were left out. The MUCS references already write most English words in Latin script.

To check the labels, we compared them with an existing third-party Hinglish relabelling of IndicVoices on 2,800 matched clips. That relabelling has no stated licence, so we used it only for this check and never for training. Our Latin-script words had a precision of 0.989 and a recall of 0.986 against it [A9].

5.4 Test-set hygiene

Every training clip was checked against the six test sets in section 6 (Monsoon, FLEURS, LibriSpeech, MUCS, Svarah and Lahaja) by hashes of its normalised text and of its audio: 27,710 text hashes and 17,115 audio hashes [A9]. Monsoon's and Lahaja's references are in Devanagari, so the text check also runs on the Devanagari-only form of each Hinglish label.

The test sets joined this blocklist at different times, and clips added earlier were not re-checked against test sets that joined later. The earliest Shrutilipi and LibriSpeech clips were checked against Monsoon, FLEURS and LibriSpeech only, and the MUCS training clips and the rest of Shrutilipi were not checked against Svarah and Lahaja. The FLEURS training split has no exact text overlap with its test split, and Vaani's test split was never used.

This report does not describe data proportions, training settings or compute.

6. Evaluation setup

6.1 Test sets

SET	WHAT IT IS	CLIPS
Monsoon Hindi	Read and spontaneous Hindi; the test set of the Hindi track of the Hugging Face Open ASR leaderboard [15] (public split)	753
FLEURS Hindi [14]	Read Wikipedia sentences (test split)	418
Lahaja [16]	Hindi from speakers of many Indian regions and accents	6,152
MUCS Hinglish [13]	Hindi-English code-mixed lectures (test split); references write most English words in Latin script	3,136
Svarah [17]	Indian-accented English	6,656
LibriSpeech test-clean [12]	Read US English	2,620
Silence	100 clips of 4 seconds: 50 of digital silence and 50 of very faint synthetic noise (Gaussian, standard deviation 0.003). We count the words a model writes	100

6.2 One harness for every model

All models go through the same code [A2]:

- the same clips in the same order, with pinned dataset revisions;
- the same audio decoding to 16 kHz;
- the same text normaliser: Unicode NFC, lowercase, punctuation removed;

- the same scorer: word-level edit distance over the reference words.

Monsoon uses the official OIWER scorer of the Hugging Face Open ASR leaderboard. It scores against a lattice that lists the accepted spelling variants of each word. Our Monsoon numbers are on the public split; the leaderboard's own number for a model can differ by a few points. Shruti has not been submitted, so it has no official rank.

We checked the harness before using it. Our model's numbers through the harness matched our original evaluation code on all seven test sets, to the digit. A repeat of 200 MUCS clips gave identical output for every model in the Hinglish comparison [A2].

Shruti always ran on a Mac CPU, in the engine, with its default 5 beams and the language set: `hi` for the Hindi and Hinglish sets, `en` for the English sets. The other models ran on an NVIDIA RTX 4090 or on the Mac.

6.3 How each model was run

MODEL	VERSION	LICENCE	DECODING
Shruti	shruti.cact (SHA-256 b189e954...)	Apache-2.0	needle engine on a Mac, 5 beams, language set
Koyal-hi-120M (Adalat AI)	adalat-ai/koyal-hi-120m-1.0 @ 7f2cb57	CC-BY-4.0 (gated)	model default (greedy)
IndicConformer Hindi (AI4Bharat), fp32 ONNX	OpenVoiceOS/ai4bharat-indicconformer-hi-onnx @ 8960b86	MIT	greedy CTC (onnx-asr 0.12.0)
Vosk small (Alpha Cephei)	vosk-model-small-hi-0.22 + vosk-model-small-en-in-0.4	Apache-2.0	Kaldi defaults (vosk 0.3.44)
Whisper-Hindi2Hinglish-Swift (Oriserve)	Oriserve/Whisper-Hindi2Hinglish-Swift @ cd0d4d0	Apache-2.0	beam 5, language en , as its card says
Whisper-small (OpenAI) [18]	openai/whisper-small @ 973afd2	Apache-2.0	beam 5, language forced (hi / en)
whisper-base-hinglish, fp32	abhishekgautamm/whisper-base-hinglish @ c780320	MIT	beam 5, language en ; on MUCS also with its card's repetition settings
whisper-tiny-hinglish-v2, fp32	abhishekgautamm/whisper-tiny-hinglish-v2 @ 751d719	MIT	as above
Whisper large-v3 (OpenAI) [18]	openai/whisper-large-v3 @ 06f233f	Apache-2.0	transformers 5.18.0 on the Mac GPU (MPS, fp16), beam 5, language forced (hi / en), sequential long-form decoding for clips over 30 s

6.4 Caveats that apply to every comparison

- **Koyal** writes punctuation and digits. Our normaliser removes punctuation but does not turn digits into words, so its digits count as errors where references spell numbers out.
- **Whisper-Hindi2Hinglish-Swift** and the two **whisper-*-hinglish** models write Roman script only. On sets with Devanagari references their standard scores mostly measure the script. We mark these cells, and for MUCS we add Roman-script views (section 7.3).
- **Vosk** writes English words in Devanagari, which counts against it on MUCS. That is a real difference in output, and we say so.

- **Whisper large-v3** ran with plain beam search, without the temperature fallback of OpenAI's own decoding. This causes some repetition loops (section 7.2).
- **Training data overlap.** Shruti was trained on the MUCS and FLEURS training splits (never their test splits), and so were some rivals: both whisper-***-hinglish** models on MUCS, and Koyal on FLEURS, according to their cards. Koyal's card says Lahaja was held out. Whisper's training data is not public.

7. Results

7.1 All models

WER %, lower is better; the lowest number in each column is in bold [A2, A3]. "Hindi only": the model does not transcribe English. n/c: not comparable, because the model writes Roman script only. –: not measured.

MODEL	FILE SIZE	MONSOON HINDI	FLEURS HINDI	LAHAJA	MUCS HINGLISH	SVARAH	LIBRISPEECH	WORDS ON 100 SILENT CLIPS
Shruti	29.5 MB	20.46	19.92	26.28	35.21	22.47	5.95	0
Koyal-hi-120M	481 MB	11.64	9.56	18.70	57.51	Hindi only	Hindi only	0
IndicConformer Hindi (fp32)	482 MB	15.31	11.31	13.18	64.48	Hindi only	Hindi only	88
Vosk small (hi + en-in)	139 MB	35.75	18.07	26.60	61.14	38.51	30.18	1
Whisper-Hindi2Hinglish-Swift	296 MB	57.16 *	n/c	n/c	n/c	20.16	12.60	1,676
Whisper-small	971 MB	128.25	59.42	117.5	–	–	–	–
whisper-base-hinglish (fp32)	290 MB	65.66 *	–	–	87.05 *	27.11 †	15.53 †	500
whisper-tiny-hinglish-v2 (fp32)	151 MB	70.60 *	–	–	87.73 *	34.50 †	23.92 †	500
Whisper large-v3	3,087 MB	33.29	28.88	31.19 ‡	79.94 ‡	7.39 ‡	2.25 ‡	538

* Roman-script output only, so this mostly measures the script; Roman-script views for MUCS are in section 7.3.

† First 1,500 Svarah and first 1,000 LibriSpeech clips; Shruti scores 21.35 and 6.36 on the same clips.

‡ First 300 Lahaja, Svarah and LibriSpeech clips and first 600 MUCS clips; Shruti on the same clips: 22.87, 21.35, 5.76 and 45.3. On Monsoon and FLEURS, Whisper large-v3 ran on every clip. In that run Shruti scored 20.46 on Monsoon and 20.14 on FLEURS with the newest engine build (section 7.5).

The bold marks show the trade-off. Shruti is the smallest file, has the lowest MUCS error and writes nothing on silence. On pure Hindi, the two Hindi-only models about 16 times larger are clearly ahead, and on English, Whisper large-v3 is.

For context, the leaderboard publishes these Monsoon numbers, measured with its own decoding on the full test set: Whisper large-v3 (1.55B parameters) 28.17, Whisper large-v3-turbo 40.79, Voxtral Mini 3B 22.9, Qwen3-ASR-0.6B 18.13 and IndicConformer-600M 8.47 [15].

7.2 Whisper large-v3 on the same clips

Whisper large-v3 has 1.55B parameters; its `model.safetensors` is 3,087 MB in fp16, about 105 times Shruti's file [A2]. Both models ran on the Mac on identical clips. The interval is a paired bootstrap 95% interval over clips [19]; the lattice scorer used for Monsoon does not support one.

SET (CLIPS)	SHRUTI	WHISPER LARGE-V3	WHISPER MINUS SHRUTI (95% INTERVAL)
Monsoon Hindi, OIWER (753)	20.46	33.29	+12.8 (no interval: lattice)
FLEURS Hindi (418)	20.14	28.88	+8.7 (+6.0 to +12.5)
Lahaja, Hindi accents (first 300)	22.87	31.19	+8.3 (+4.4 to +14.7)
MUCS Hinglish (first 600)	45.3	79.94	+34.6 (+23.5 to +48.7); partly a script difference
Svarah, Indian English (first 300)	21.35	7.39	-14.0 (-17.1 to -11.2)
LibriSpeech test-clean (first 300)	5.76	2.25	-3.5 (-4.3 to -2.8)
Silence: words written (100)	0	538	

On Hindi, Shruti is more accurate on all three sets. On English, Whisper large-v3 is far more accurate, and Shruti makes about three times as many errors on Indian-accented English.

Three diagnostics qualify the Hindi result:

- **Repetition loops.** Without the temperature fallback, Whisper's beam search sometimes repeats itself. Removing every clip where any model's output is more than 3 times the length of the reference, Shruti still leads: FLEURS 19.9 against 26.43 (2 clips removed), Lahaja 22.87 against 28.66 (1 clip), MUCS 44.94 against 58.79 (21 clips). The leaderboard's own Whisper large-v3 Monsoon number, 28.17 with its own decoding, is also behind Shruti's 20.46.
- **Digits.** Whisper writes numbers as digits (83 output words with digits on FLEURS and 45 on Lahaja, against 1 and 0 for Shruti). The references spell numbers out, so these count as errors for Whisper.
- **Script on MUCS.** With language `hi`, Whisper writes many English words in Devanagari, which the MUCS references write in Latin script, so part of its MUCS gap is script rather than recognition.

7.3 Hinglish and the script question

The two whisper-*hinglish models write everything in Roman script, so their standard MUCS scores (87.05 and 87.73) mostly measure script. For a fairer view we converted the reference and every model's output to Roman script with the same function and scored again [A1, A2, A9]. We chose the Hinglish transliteration table as the main view before seeing any scores. ITRANS and a tolerant matching key are alternatives.

MODEL	STANDARD	ROMANISED: HINGLISH TABLE	ROMANISED: ITRANS	ROMANISED: TOLERANT KEY	CHARACTER ERROR RATE, HINGLISH TABLE
Shruti	35.22	34.32	34.79	31.37	22.86
whisper-base-hinglish, card decoding	87.05	43.45	68.10	32.66	25.12
whisper-tiny-hinglish-v2, card decoding	87.73	44.94	69.17	34.22	25.82
whisper-base-hinglish, beam 5 only	91.14	46.74	71.97	35.85	28.52
whisper-tiny-hinglish-v2, beam 5 only	101.40	58.04	82.70	47.17	39.63

MUCS, all 3,136 clips. Shruti's standard score here, 35.22, is from the newest engine build (section 7.5).

Shruti is ahead under all three converters, but by different margins. Paired bootstrap 95% intervals for whisper-base-hinglish minus Shruti:

- Hinglish table: +8.2 to +10.0;
- ITRANS: +32.3 to +34.3;
- tolerant key: +0.4 to +2.2, a narrow lead.

Any neutral converter slightly favours Shruti. Where the references spell an English word in Devanagari ("माउस"), its romanised form ("maus") does not match the correct English spelling the rivals write. The models' author reports 28.4 (base) and 32.0 (tiny) on a 250-clip sample, with their own converter. We did not run that converter, because it is code we have not reviewed, and so we make no comparison with those numbers. Measured as released, these two models are 290 MB and 151 MB (fp32); their int8 builds (76 MB and 41 MB) were not measured.

Our claim is therefore limited to this: Shruti has the lowest Hinglish word error rate of the open models we measured (MUCS 2021, same harness), and stays ahead when both sides are written in Roman script.

7.4 Silence

On the 100 silent clips Shruti and Koyal wrote 0 words and Vosk wrote 1. The others wrote more: IndicConformer 88, both whisper-*-hinglish models 500 (the same five-word sign-off on every clip), Whisper large-v3 538 and Whisper-Hindi2Hinglish-Swift 1,676. This tests silence and very faint noise only. We have not tested real background noise or music.

7.5 Engine builds

The numbers in the main table (section 7.1) come from the needle build we used on 18 October 2026. With the newest build, used for the Whisper comparison, Shruti's numbers move by at most 0.22 points [A3, A9]:

	MONSOON	FLEURS	LAHAJA	MUCS	SVARAH	LIBRISPEECH	SILENCE
Build of 18 October	20.46	19.92	26.28	35.21	22.47	5.95	0
Newest build	20.46	20.14	26.33	35.22	22.47	6.00	0

8. On the device

8.1 A budget phone

We ran the released file on a Samsung Galaxy F62 (Exynos 9825, 6 GB of memory, Android 13), a 2021 phone that sold for about ₹18,000. We used the engine's android-arm64 build, offline, on 51 test clips [A4]:

THREADS	MEDIAN TIME FOR A MONSOON SENTENCE (~7 S), MODEL LOAD INCLUDED	A 60 S RECORDING	ALL CLIPS, TIMES FASTER THAN REAL TIME	PEAK MEMORY	TRANSCRIPTS IDENTICAL TO THE MAC'S (OF 51)
2	1,010 ms	9.5 s (6.3× real time)	6.6×	80 MB	42
4	1,301 ms	12.6 s (4.8×	4.6×	80 MB	42
8	3,977 ms	65.8 s (0.9×	1.5×	80 MB	42

Two threads is the best setting on this phone. More threads are slower, presumably because the phone has only two fast cores. The 9 transcripts that differ from the Mac's differ by a letter or two.

8.2 The browser and the Hugging Face Space

- **Browser (WebAssembly).** The engine's WebAssembly build, single-threaded and run in Node on a Mac, transcribes a 9.6-second clip in 0.45 seconds, with transcripts identical to the native engine [A1]. The model page on udgamlabs.com runs this build in a Web Worker: the browser downloads the model once and then transcribes on the device, also offline.
- **Hugging Face Space.** On a shared 2-core CPU, the demo Space transcribes a 2.8-second clip in 0.35 seconds [A1].
- **Browsers checked on the live page.** On 19 October 2026 the model page worked in Chrome on the Galaxy F62 (microphone and files; all 12 test clips given as files matched the Mac's transcripts), in Chrome on a Mac, and in Safari on an iPhone, in the iOS simulator: the model loads and a file transcribes correctly. We did not test the iPhone's microphone [A9]. In Chrome on the Galaxy F62, the final pass after recording stops takes about a third as long as the speech [A6].

8.3 Live text and whole recordings

The engine can transcribe while someone speaks. It takes about a second of audio at a time and commits words once they are stable [A5]. We compared that with transcribing the whole recording at once, which is what the Space does and what the website does as its final pass. Same engine, same file, first 150 clips of each set [A9]:

SET	WHOLE RECORDING	LIVE, 1 S CHUNKS (COMMITTED WORDS)	IDENTICAL TEXT
Monsoon Hindi	21.43	22.60	81 / 150
FLEURS Hindi	19.48	21.47	51 / 150
MUCS Hinglish	40.48	42.69	104 / 150

The committed live text is 1 to 2 points worse. The pending words shown while someone is still speaking are guesses and change more.

Two checks of the capture chain found no loss:

- On the same sentences, audio captured by the website scored 12.5% WER and the phone's screen-recorder audio 12.0%.
- The website's resampler gives the same transcripts as soxr.

8.4 Detecting the language

The engine can detect the language or take it as a setting. We scored each set three ways [A9]:

SET (CLIPS)	LANGUAGE HI	LANGUAGE EN	DETECTED	DETECTED AS
Monsoon Hindi (753)	20.46	31.38	20.47	hi 751, en 2
MUCS Hinglish (first 400)	49.89	68.84	49.89	hi 400
Lahaja (first 300)	22.87	34.46	22.89	hi 298, en 2
Svarah, Indian English (first 400)	27.52	21.13	25.75	hi 257, en 142, none 1
LibriSpeech (first 400)	6.59	6.04	6.04	en 400

Detection does as well as the right setting on Hindi, Hinglish, accented Hindi and US English. Indian-accented English is the exception: detection takes about two thirds of those clips for Hindi. The demos therefore show the model as it is, with detection on. The model card tells developers to set `en` for speech that is entirely English.

9. Limitations

- **Pure Hindi.** Hindi-only models about 16 times larger are clearly more accurate. Koyal-hi-120M makes roughly half as many errors on Monsoon and FLEURS, and IndicConformer Hindi half as many on accented Hindi (Lahaja). For Hindi only, with memory to spare, they are the better choice.
- **English.** Whisper large-v3 is far more accurate on English: 7.39 against 21.35 on Indian-accented English and 2.25 against 5.76 on LibriSpeech, on the same clips. On Indian-accented English, Shruti (22.47) is also behind Whisper-Hindi2Hinglish-Swift (20.16), which is 10 times larger and writes Hindi in Roman script only.
- **Accented Hindi.** At 26.28 on Lahaja, Shruti misses our own usefulness bar of about 20%.
- **Names it has never heard** are spelt by sound. In our phone test it wrote its own name, Shruti, as "सूती", "शूती" and "shooty", and Udgam Labs as "outgum labs" and "oathgum labs". The engine's `keywords` option can favour such names; we have not measured how much it helps.

- **Spelling and grammar look-alikes** are common errors: है/हैं, मैं/में, की/के, and spoken against formal forms (पे/पर, वो/वह).
- **Speak close to the device.** When speech was played from a small computer speaker across a desk, the model often detected it as English, while the same clips given as files were transcribed correctly [A9].
- **Noise and music** were not tested. The silence set has only digital silence and very faint synthetic noise.
- **Output format.** Numbers are written as words, the way they are spoken. There is no punctuation and there are no capital letters.
- **Platforms.** We measured the model on a Mac, Linux (x86), one Android phone and WebAssembly. The engine has Windows and iOS builds, but we have not tested Shruti on them.
- **Other languages.** Whistle's other languages (German, French, Spanish, Italian, Dutch, Polish) were not evaluated, and are probably weaker after our training.
- **Test overlap.** Shruti saw the MUCS and FLEURS training splits, so those two test sets are in-domain for it. Some training sources were not re-checked against the test sets added to the blacklist later (section 5.4).
- **Not for high-stakes use.** It should not be used in medical, legal or similar settings without a person checking the output.

10. What we would do next

- **Pure Hindi.** Add more conversational Hindi under open licences, human-corrected labels for existing corpora, and labels for untranscribed audio from a permissively licensed Hindi model.
- **Indian-accented English.** Add more Indian-English conversational speech, the area where Shruti is furthest behind other models.
- **Noise.** Measure accuracy and invented words on real background noise and music, not only silence.
- **Names.** Measure how much keyword biasing helps on rare names and brand names.
- **Smaller still.** Test 2-bit quantisation-aware training, which by our estimate would bring the file to roughly 17 MB, and check that accuracy holds.
- **An official number.** Submit to the Hugging Face Open ASR leaderboard, so that Monsoon is scored on its full test set by the leaderboard itself.

- **Text-to-speech.** Shruti is the first half of an offline voice stack for Hindi, English and Hinglish; a small text-to-speech model is the other half.

11. Release details

ITEM	WHERE
Weights, model card and evaluation details	huggingface.co/udgamlabs/Shruti
Demo (Hugging Face Space)	huggingface.co/spaces/udgamlabs/Shruti
Examples and checksums (v1.0.0)	github.com/udgamlabs/shruti
Model page, with an in-browser demo	udgamlabs.com/models/shruti

- **Release:** 19 October 2026, version 1.0.0.
- **Files:** `shruti.cact` (29.5 MB, SHA-256 b189e954...) for the needle engine; `shruti.safetensors` (222 MB, full-precision weights). Checksums are in `SHA256SUMS` in the GitHub repository.
- **Runtime:** the Cactus needle engine (Apache-2.0): the `cactus-needle` Python package, the `needle` command-line binaries in `Cactus-Compute/needle3`, the C library (`needle.h`), and the WebAssembly build. The Python package sends anonymous usage counts by default; `NEEDLE_TELEMETRY=0` turns them off.
- **Licence:** Apache-2.0. Shruti is a modified version of Cactus Whistle by Cactus Compute (Apache-2.0), and the `NOTICE` file says so. The training data licences are in section 5.2.
- **Developer:** Udgam Labs, udgamlabs.com.
- **Contact:** the Community tab of the Hugging Face model page, or issues on the GitHub repository.

```
@techreport{udgamlabs2026shrutireport,
  title      = {Shruti: offline speech-to-text for Hindi, English and Hinglish
in 29.5 MB},
  author     = {{Udgam Labs}},
  institution = {Udgam Labs},
  type       = {Technical report},
  year       = {2026},
  url        = {https://udgamlabs.com/research/shruti}
}
```

12. Conclusion

A 55M-parameter speech model built on Cactus Whistle runs offline from one 29.5 MB file and writes Hinglish the way people type it. Three things made it work:

- a Devanagari vocabulary whose new rows start from the model's existing byte-level spelling;
- a full fine-tune with 4-bit quantisation-aware training;
- 2,732 hours of openly licensed speech, with English words labelled in Latin script from the datasets' own annotations.

Against every open model we measured with one harness, Shruti has the lowest Hinglish error rate on MUCS, and stays ahead when both sides are written in Roman script. It is about 100 times smaller than Whisper large-v3 and more accurate on Hindi on the same clips (Monsoon, FLEURS and Lahaja). It writes nothing on silence, and it transcribes a sentence in about a second on a 2021 phone.

The same harness shows its limits just as clearly:

- Hindi-only models about 16 times larger are far more accurate on pure Hindi.
- Whisper large-v3 is far more accurate on English.
- Accented Hindi, rare names, noise and music remain open problems.

The weights, examples and evaluation details are open under Apache-2.0, so the results can be checked and the model can be fine-tuned further.

13. References

1. Cactus Compute. *Whistle* model card and weights. huggingface.co/Cactus-Compute/whistle (accessed 19 October 2026).
2. Cactus Compute. *needle*: engine, Python package and Whistle training code. github.com/cactus-compute/needle (commit 482361d for training); engine builds at huggingface.co/Cactus-Compute/needle3.
3. Kudo, T. and Richardson, J. (2018). *SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing*. [arXiv:1808.06226](https://arxiv.org/abs/1808.06226).
4. Hu, E. J., Shen, Y. et al. (2021). *LoRA: Low-Rank Adaptation of Large Language Models*. [arXiv:2106.09685](https://arxiv.org/abs/2106.09685).
5. Bengio, Y., Léonard, N. and Courville, A. (2013). *Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation*. [arXiv:1308.3432](https://arxiv.org/abs/1308.3432).

6. Jacob, B., Kligys, S. et al. (2018). *Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference*. [arXiv:1712.05877](https://arxiv.org/abs/1712.05877).
7. Javed, T., Nawale, J. A. et al. (2024). *IndicVoices: Towards building an Inclusive Multilingual Speech Dataset for Indian Languages*. [arXiv:2403.01926](https://arxiv.org/abs/2403.01926). Dataset: [ai4bharat/IndicVoices](https://ai4bharat.org/IndicVoices).
8. ARTPARK and IISc. *Vaani* transcribed speech. Dataset: [ARTPARK-IISc/Vaani-transcription-part](https://ai4bharat.org/ARTPARK-IISc-Vaani-transcription-part).
9. Bhogale, K. S., Raman, A. et al. (2022). *Effectiveness of Mining Audio and Text Pairs from Public Data for Improving ASR Systems for Low-Resource Languages* (Shrutilipi). [arXiv:2208.12666](https://arxiv.org/abs/2208.12666). Dataset: [ai4bharat/Shrutilipi](https://ai4bharat.org/Shrutilipi).
10. Javed, T., Bhogale, K. S. et al. (2022). *IndicSUPERB: A Speech Processing Universal Performance Benchmark for Indian languages* (Kathbath). [arXiv:2208.11761](https://arxiv.org/abs/2208.11761). Other AI4Bharat datasets used: [ai4bharat/Kathbath](https://ai4bharat.org/Kathbath), [ai4bharat/NPTEL](https://ai4bharat.org/NPTEL), [ai4bharat/Rural_Women_ASR_v2](https://ai4bharat.org/Rural_Women_ASR_v2).
11. Ardila, R., Branson, M. et al. (2019). *Common Voice: A Massively-Multilingual Speech Corpus*. [arXiv:1912.06670](https://arxiv.org/abs/1912.06670). Version 17 via [fixie-ai/common_voice_17_0](https://fixie-ai.com/common_voice_17_0).
12. Panayotov, V., Chen, G., Povey, D. and Khudanpur, S. (2015). *LibriSpeech: an ASR corpus based on public domain audio books*. ICASSP 2015. Dataset: [openslr/librispeech_asr](https://openslr.org/librispeech_asr).
13. Diwan, A., Vaideeswaran, R. et al. (2021). *Multilingual and code-switching ASR challenges for low resource Indian languages* (MUCS 2021). [arXiv:2104.00235](https://arxiv.org/abs/2104.00235). Copy used: [dianavdavidson/MUCS-Hinglish](https://dianavdavidson.github.io/MUCS-Hinglish).
14. Conneau, A., Ma, M. et al. (2022). *FLEURS: Few-shot Learning Evaluation of Universal Representations of Speech*. [arXiv:2205.12446](https://arxiv.org/abs/2205.12446). Dataset: [google/fleurs](https://google.github.io/fleurs).
15. Srivastav, V., Zheng, S., Bezzam, E. et al. (2025). *Open ASR Leaderboard: Towards Reproducible and Transparent Multilingual and Long-Form Speech Recognition Evaluation*. [arXiv:2510.06961](https://arxiv.org/abs/2510.06961). Leaderboard: huggingface.co/spaces/hf-audio/open_asr_leaderboard (Hindi numbers accessed 18 October 2026). Test set: [VoiceArena/Monsoon_hi_test](https://voicearena.org/monsoon_hi_test).
16. Javed, T., Nawale, J. et al. (2024). *LAHAJA: A Robust Multi-accent Benchmark for Evaluating Hindi ASR Systems*. [arXiv:2408.11440](https://arxiv.org/abs/2408.11440).
17. Javed, T., Joshi, S. et al. (2023). *Svarah: Evaluating English ASR Systems on Indian Accents*. [arXiv:2305.15760](https://arxiv.org/abs/2305.15760).
18. Radford, A., Kim, J. W. et al. (2022). *Robust Speech Recognition via Large-Scale Weak Supervision* (Whisper). [arXiv:2212.04356](https://arxiv.org/abs/2212.04356). Models: [openai/whisper-large-v3](https://openai.com/whisper-large-v3), [openai/whisper-small](https://openai.com/whisper-small).
19. Bisani, M. and Ney, H. (2004). *Bootstrap estimates for confidence intervals in ASR performance evaluation*. ICASSP 2004.

20. Model cards of the other models compared: [adalat-ai/koyal-hi-120m-1.0](#), [OpenVoiceOS/ai4bharat-indicconformer-hi-onnx](#), [Vosk models](#), [Oriserve/Whisper-Hindi2Hinglish-Swift](#), [abhishekgautamm/whisper-base-hinglish](#), [abhishekgautamm/whisper-tiny-hinglish-v2](#) (accessed 17 to 19 October 2026).
21. Udgam Labs. Shruti model card, evaluation details ([eval/comparison.md](#) , [eval/shruti_on_device.json](#) , [eval/phone_benchmark.json](#)), GitHub repository and model page, October 2026.

Appendix A. Where every number comes from

Every number in this report can be checked against one of the public sources below, or is marked as measured by Udgam Labs.

- **A1.** Shruti model card, [huggingface.co/udgamlabs/Shruti](#): file size, parameters and layers; the 714 added pieces; 2,732 hours of training speech, the data sources and their licences; test sentences removed by text and audio hash; the Hinglish labelling convention; the phone, browser (0.45 s for a 9.6 s clip) and Space (0.35 s for a 2.8 s clip) timings; the romanised MUCS view with the most forgiving converter (31.37 against 32.66 and 34.22); the names misheard in the phone test; the platforms we measured.
- **A2.** [eval/comparison.md](#) in the Hugging Face repository ([huggingface.co/udgamlabs/Shruti/tree/main/eval](#)), also [docs/comparison.md](#) on GitHub: the test sets and clip counts, including the silence set; every model's results, versions, licences and decoding; the Whisper large-v3 same-clip table with its intervals and its repetition-loop and digit diagnostics; the romanised MUCS scores 34.32, 43.45 and 44.94; the 200-clip repeat check.
- **A3.** [eval/shruti_on_device.json](#) (same folder; also in [docs/](#) on GitHub): Shruti's on-device results with the engine build of 18 October 2026.
- **A4.** [eval/phone_benchmark.json](#) (same folder; also in [docs/](#) on GitHub): phone timings, memory and identical transcripts by thread count.
- **A5.** GitHub repository, [github.com/udgamlabs/shruti](#): how the model works (80 ms frames, the language tag, 30-second windows, streaming), runnable examples and [SHA256SUMS](#) .
- **A6.** Model page, [udgamlabs.com/models/shruti](#): the phone's price; the final-pass time in the phone's browser; the language tip (21.13 with English set against 25.75 detected, on a Svarah subset).
- **A7.** Cactus Whistle and needle [1, 2]: Whistle's architecture and 8,192-piece tokenizer; needle's 4-bit quantiser and its 8-bit activation and key-value cache settings; the initialisation of new embedding rows in needle's Whistle fine-tuning code; the base model's empty output on silence; the engine's platforms.

- **A8. Public dataset cards, model cards and the leaderboard [7–18, 20]:** dataset licences and splits; other models' sizes, licences and training data as their cards state them; the leaderboard's published Monsoon numbers.
- **A9. Measured by Udgam Labs, 18–19 October 2026, and not published as separate files:**
 - the example transcripts (section 2);
 - the label-quality check (0.989 and 0.986) and the blocklist sizes (section 5);
 - the check of our harness against our original evaluation code (section 6.2);
 - the romanised MUCS views not in A1 or A2: ITRANS, character error rates, the beam-5-only rows and the bootstrap intervals (section 7.3);
 - Shruti's numbers with the newest engine build (section 7.5), except FLEURS 20.14, which is in A2;
 - live against whole-recording text (section 8.3) and the language-detection table (section 8.4), beyond the subset in A6;
 - the browser checks on the live page (section 8.2) and the speaker-played audio (section 9).

About Udgam Labs. Udgam Labs is an independent AI research lab. We work from first principles: before we build, we understand why things work, whether we are training a model, fine-tuning one or applying AI to a new problem. Udgam (उद्गम) is Sanskrit for origin. udgamlabs.com