Lekhani: in-browser transliteration in under 1 MB per language

UDGAM LABS VERSION 1.0

Jitesh K Prasad, Udgam Labs (udgamlabs.com), 14 November 2026. Technical report for Lekhani, runner 1.3.0, twelve languages, version 1.0.

Evidence labels: **[MEASURED]** our own runs; **[EXTERNAL]** other people's cards, papers or pages, not reproduced by us; **[ESTIMATE]** our projections. A label covers the sentence, list or table it closes or heads.

Summary

Lekhani by Udgam Labs turns words typed in Latin letters into their own script: `kaise ho` becomes कैसे हो, `vanakkam` becomes வணக்கம். It covers twelve languages: Hindi, Tamil, Marathi, Bengali, Urdu, Kannada, Telugu, Malayalam, Gujarati, Punjabi, Odia and Assamese. Each language is one small model and a list of common words, and everything a web page loads for one language, runner included, is under 1 MB. It runs in the page, on the user's own device, so nothing typed is sent anywhere.

- **What it is.** One character-level encoder-decoder transformer per language, 0.81M or 1.21M parameters, stored at 4 bits, plus a list of 125,000 to 300,000 common words and a 6.6 KB JavaScript runner with no dependencies. Released under Apache-2.0 (weights and runner) and CC0 (word lists): Hindi on 30 October 2026, then Telugu, Tamil and Kannada, and the other eight on 14 November. [MEASURED]
- **Main result.** On the Aksharantar test (AI4Bharat), Lekhani's first guess is right for 49.9% (Odia) to 81.2% (Telugu) of words. It is ahead of IndicXlit's model on its own, the usual open tool, in all twelve languages, by 4.2 to 14.8 points, and every paired 95% interval is clear of zero. [MEASURED]
- **Where it falls short.** IndicXlit's full Python package, which re-ranks with word lists of millions of words held in memory (358 MB for Hindi), is level with Lekhani or ahead in ten of the twelve languages on this test, and ahead in six of the nine languages we could test on real typing. Names are weak in eleven languages, and Odia, Punjabi, Urdu and Assamese are weak in absolute terms (49.9% to 58.6%). [MEASURED]
- **Same answers in the browser.** The runner gives the same first guess as our Python scorer on 99.9% to 100% of test words in every language. It takes about 2 to 4 ms a word in Node on a Mac mini. [MEASURED]

---

1. Why a transliterator under 1 MB

Most people who write Hindi, Tamil or Bengali on a phone or laptop type it in Latin letters ("romanised"), because that is the keyboard they have. Turning those words into the language's own script is transliteration. The open tools that do it well are large: IndicXlit's model has 11.5M parameters (46 MB as fp32 weights), and its Python package adds a word list of millions of

words per language that it holds in memory. Both run on a server, so whatever a person types is sent to it.

Lekhani asks whether a model small enough to download into a web page, and run there, can be more accurate than IndicXlit's model on its own. Under 1 MB is the budget: a page can fetch that once, quickly, and keep it.

---

2. What we release

ITEM	WHERE	LICENCE
Twelve model files (`model.q4.gz` , `model.json.gz`) and word lists (`words.bin.gz` , `rerank.json`), one folder per language	huggingface.co/udgamlabs/Lekhani	Apache-2.0 (models), CC0 (word lists)
The runner, `translit.js` (minified, 6,625 bytes gzip) and `translit.src.js` (readable)	the same repository, and github.com/udgamlabs/lekhani	Apache-2.0
A demo that runs on the visitor's device	huggingface.co/spaces/udgamlabs/Lekhani and udgamlabs.com/models/lekhani	

Everything a page loads for one language, as gzip bytes [MEASURED, runner 1.3.0]:

LANGUAGE	SCRIPT	PARAMETERS	DOWNLOAD (GZIP BYTES)
Hindi	Devanagari	0.81M	872,565
Tamil	Tamil	0.81M	708,999
Marathi	Devanagari	1.21M	958,520
Bengali	Bengali	1.21M	999,891
Urdu	Urdu (Arabic script)	1.22M	908,841
Kannada	Kannada	1.21M	981,214
Telugu	Telugu	1.21M	964,906
Malayalam	Malayalam	1.21M	956,061
Gujarati	Gujarati	1.21M	983,327
Punjabi	Gurmukhi	0.81M	932,530
Odia	Odia	1.21M	978,975
Assamese	Bengali script, Assamese letters (ৰ, ৱ)	0.81M	982,294

3. Data

All training data is openly licensed. Each source's licence was read at its source and pinned to a Hugging Face revision (Aksharantar `e418c1f`, IndicXlit `e7049fe`, IndicCorp v2 `2d7285e`). [EXTERNAL]

- **Aksharantar** (AI4Bharat) word pairs of romanised and native words: its mined parts (CC0) and its hand-collected AK-Freq rows (CC-BY). We did not use its Dakshina rows (CC BY-SA 4.0, share-alike) or its "existing sources" rows (mixed licences).
- **IndicCorp v2** (AI4Bharat, CC0): word counts in each language. The common words become each language's shipped word list, which is why the lists are CC0.
- **IndicXlit** (AI4Bharat, MIT) as a teacher: it wrote the ways people type common words from IndicCorp v2, and each language's spellings of English words.
- **CMU Pronouncing Dictionary** (Carnegie Mellon University, BSD-style): the English words given to the teacher. Not shipped.
- **Not used:** IndicXlit's own word lists, built from IndicCorp v1 (CC BY-NC 4.0). We ran them only to measure IndicXlit's package.

Every test, validation and real-typing input and answer was removed from the training pairs before training. [MEASURED]

---

4. Method

The model. Each language has its own small encoder-decoder transformer that reads a word's Latin letters and writes the native script letter by letter. Hindi's has 2 encoder and 2 decoder layers, width 128, 4 attention heads and a feed-forward width of 384: 811,904 parameters. The larger shape has 3 + 3 layers (about 1.21M parameters). For each language both shapes were trained, and the one used was chosen on the Aksharantar validation split. The models were trained from random initialisation, with IndicXlit as the teacher for most training pairs, and are stored at 4 bits. [MEASURED]

The word list. For each word the model proposes its 8 best spellings (beam search, beam 8). A list of 125,000 to 300,000 common words, built from IndicCorp v2 counts, then moves a common real word up when the model is unsure. Words that are not in the list still work from the model alone. The list's size and how strongly it counts were chosen on validation, per language. [MEASURED]

What the list adds. The same models without their lists score 3.5 to 15 points lower on the test (Hindi 56.2% without, 65.9% with; Urdu 42.3% without, 57.6% with). [MEASURED]

5. The runner

`translit.js` is one JavaScript file with no dependencies that serves every language. A page passes it a folder's four files; it returns the best spelling and the 8 candidates for each word, and leaves digits, punctuation and text already in the script as they are.

- **Same answers as Python.** Runner 1.3.0's first guess matches our Python scorer's on every test word in eight languages (Urdu, Kannada, Telugu, Malayalam, Gujarati, Punjabi, and Odia and Assamese checked at release on 13 November), on 99.909% in Marathi (11 words changed by the 1.2.0 fix for words cut short) and on 99.993% in Bengali (one empty input) [MEASURED, 11 and 13 Nov]. Two languages were checked through a chain. Tamil: 1.3.0 gives the same first guess as Tamil's release-bundle runner on 11,405 of 11,405 test words, and that runner matched Python. Hindi: the launch runner matched Python on 9,991 of 9,991 test words; the 1.2.0 fix for words cut short then changed 13 of them on purpose (one more right, one fewer, the score unchanged at 65.89%), checked against its own reference script rather than Python; and 1.3.0's Hindi output is byte-identical to 1.2.0's. On the 4,000 real-typing words of each of the nine languages that have them, the runner each language shipped with matched Python on every word; these were not re-run on 1.3.0.
- **Speed.** About 2 to 4 ms a word in Node with WebAssembly SIMD on a Mac mini (Hindi 1.84 ms, Tamil 2.41 ms) [MEASURED]. Runner 1.1.0 (2 November) made it about 10 times faster with batched beam search, WebAssembly SIMD kernels with a plain-JavaScript fallback, and a word cache; this was contributed by Manit Roy (GitHub #1). The first release took 18 ms a word in a Chromium browser on the same Mac.
- **One runner for twelve scripts.** Each language's `rerank.json` names the first letter of its script's Unicode block, so the word list can pack any script one byte per letter (runner 1.3.0, 11 November).
- **Under 1 MB.** The shipped runner is minified (6,625 bytes gzip). With the readable runner, Bengali's files would come to 1,004,472 bytes, over the budget; minified, they are 999,891.

6. How we test

- **The test.** The Aksharantar test split (AI4Bharat, revision `e418c1f`), one language at a time, every unique romanised input (4,201 for Odia to 17,402 for Gujarati). Rows are grouped by romanised input (trimmed and lower-cased). A prediction counts if it matches any reference spelling for that input, both trimmed, in Unicode NFC and with the zero-width joiners (U+200C, U+200D) removed. It is a hard test: many rare words and names.
- **Choices on validation only.** The model shape, list size and list settings were chosen on the Aksharantar validation split; each language's test was scored once.

- **Intervals.** 95% intervals from 2,000 bootstrap resamples of the test words; a gap between two systems is paired (both scored on the same resampled words).
- **The bar, set before training.** For every language the bar was IndicXlit's model on its own (11.5M parameters, beam 4, its default), run by us on the same words with the same scorer: a plain-PyTorch port of its network loading the official checkpoint, with its official decoding. The port reproduces the IndicXlit paper's own scores within 0.3 to 0.7 points on Hindi. [MEASURED vs EXTERNAL]
- **IndicXlit's full package** (its Python package's default: beam 4 and its own word list of 0.62M to 16M words per language) was also run, as a comparison, never as the bar.
- **Real typing.** For the nine languages that have it, 4,000 words from Dakshina's romanised Wikipedia sentences, typed by native speakers, aligned word by word (Dakshina has no Odia or Assamese). Words that also appear in Aksharantar's Dakshina rows were removed.
- **Hindi chat set.** 122 words people type in chat (chat words, English words used in Hindi, brands, names), written by us and checked word by word by a native Hindi speaker. Too small to separate systems a few words apart.

---

7. Results

7.1 Aksharantar test, all twelve languages

First guess right, % of unique test inputs [MEASURED, scored 28 October to 13 November 2026; re-scored with runner 1.3.0 on 11 and 13 November]:

LANGUAGE	TEST WORDS	LEKHANI (95% INTERVAL)	INDICXLIT'S MODEL	GAP (PAIRED 95% INTERVAL)	INDICXLIT'S PACKAGE
Hindi	9,991	**65.9** (64.9 to 66.8)	58.9	**+7.0** (+6.1 to +8.0)	68.7
Tamil	11,405	**74.2** (73.4 to 75.1)	64.2	**+10.0** (+9.3 to +10.7)	73.9
Marathi	12,059	**70.9** (70.2 to 71.8)	63.2	**+7.8** (+7.0 to +8.6)	72.7
Bengali	13,367	**70.2** (69.4 to 71.0)	55.4	**+14.8** (+14.0 to +15.6)	67.4
Urdu	14,617	**57.6** (56.8 to 58.4)	44.4	**+13.2** (+12.4 to +14.0)	48.2
Kannada	11,165	**79.4** (78.7 to 80.2)	72.7	**+6.7** (+6.0 to +7.4)	80.9
Telugu	9,936	**81.2** (80.5 to 81.9)	72.6	**+8.5** (+7.7 to +9.3)	81.1
Malayalam	12,343	**67.1** (66.3 to 68.0)	59.5	**+7.6** (+6.9 to +8.4)	69.2
Gujarati	17,402	**72.8** (72.1 to 73.5)	61.2	**+11.6** (+10.9 to +12.3)	72.7
Punjabi	10,949	**55.7** (54.8 to 56.6)	44.5	**+11.2** (+10.3 to +12.0)	55.2
Odia	4,201	**49.9** (48.4 to 51.4)	45.6	**+4.2** (+3.0 to +5.5)	52.0
Assamese	5,476	**58.6** (57.3 to 59.9)	53.1	**+5.5** (+4.4 to +6.7)	60.4

- **Against IndicXlit's model** Lekhani is ahead in all twelve, and every paired interval is clear of zero.
- **Against IndicXlit's package** Lekhani is ahead in Bengali (+2.8, interval +2.2 to +3.4) and Urdu (+9.4, +8.6 to +10.2); level in Tamil, Telugu, Gujarati and Punjabi (intervals that include zero); and behind in Hindi (−2.8), Marathi (−1.8), Kannada (−1.4), Malayalam (−2.1), Odia (−2.1) and Assamese (−1.8). The package holds 4.2 million Hindi words in 358 MB of memory, and about 1 GB for Tamil's 9.5 million.
- **By part,** the gain is on common words and typed words. In every language except Hindi (whose Indian-name rows get 66.6%), the name rows (AK-NEF, AK-NEI) get 25% to 60% right; in the nine languages that have typed-word (Dakshina) rows, those get 58% to 88%. In Odia and Assamese the whole gain is on common words (9.0 points for Assamese, 10.4 for Odia); on names Lekhani is level with IndicXlit's model to 4 points below it.

7.2 Real typing (nine languages)

4,000 words from Dakshina's romanised sentences per language, first guess right [MEASURED]:

LANGUAGE	LEKHANI	VS INDICXLIT'S MODEL	VS INDICXLIT'S PACKAGE
Tamil	64.0	+6.3 (+5.1 to +7.6)	−2.8 (−3.9 to −1.7)
Marathi	72.4	+7.0	−1.9
Bengali	65.6	+10.3	+1.8 (+0.5 to +3.1)
Urdu	58.6	+8.6	+2.9 (+1.4 to +4.3)
Kannada	76.2	+1.5 (+0.5 to +2.6)	−3.8 (−4.6 to −2.9)
Telugu	72.4	+3.2 (+2.0 to +4.5)	−3.3 (−4.3 to −2.3)
Malayalam	59.8	+6.3 (+5.0 to +7.6)	−2.6 (−3.8 to −1.3)
Gujarati	65.4	+6.9 (+5.5 to +8.3)	−2.8 (−4.0 to −1.6)
Punjabi	55.4	+9.6 (+8.1 to +11.1)	+0.7 (−0.6 to +2.0), level

On real typing Lekhani is ahead of IndicXlit's model in all nine, and IndicXlit's package is ahead in six.

7.3 Other open transliterators

We searched Hugging Face for new or missed transliterators before the launches (29 October for Hindi, 11 November for ten languages, 12 November for Tamil) and measured the ones in reach on the same words with the same scorer, word by word [MEASURED]:

SYSTEM	SIZE	AKSHARANTAR TEST	NOTE
IB-Emper/hindi-translit-v3	15.1M parameters	Hindi 48.6 (Lekhani +17.3, +16.3 to +18.4)	its word re-ranker is not published; its training data includes Dakshina and part of Aksharantar, so test overlap is possible and could only help it
UnimeType Transliteration 2.0, 4B	4.2B parameters	Hindi 20.0 on a fixed 2,000-word sample (Lekhani 65.9 on the same words)	built for whole sentences, so word by word is not its use; Q8_0 GGUF; 14% of outputs stay in Latin
UnimeType Transliteration 2.0, 0.8B	0.75B parameters	Hindi 8.0	same protocol
ByT5 `tenglish-to-telugu`	about 300M parameters	Telugu 78.0 on a fixed 1,000-input sample (Lekhani 81.4 on the same inputs; typed: 72.2 vs 74.9; not paired)	
`aksharakuppy` (Malayalam)	16.6M parameters + a 41 MB lexicon	real typing 53.6 (Lekhani 59.8)	its Aksharantar test score is not used: its training data overlaps the test
Smaller open Hindi models (indic-xlit-270M and -50M, a 60M MIT model)	52M to 103M	Hindi 9.1 to 21.7	

UnimeType 2.0 was measured on Hindi only; we make no claim about it in the other languages. The 11 November scan found nothing new under 1 MB for any of the ten languages it covered; Odia and Assamese were not scanned. Closed systems (Gboard, Google Input Tools, SwiftKey, paid APIs) were not measured and are not compared.

7.4 Hindi chat words

On the 122-word chat set, Lekhani (runner 1.2.0 and later) gets 113 right, IndicXlit's package 112 and IndicXlit's model 102. By group: chat words 42 of 44, English words 35 of 36, brands 17 of 20, names 19 of 22 (the launch build got 112: runner 1.2.0 fixed `acha` → अच्छा, a chat word). The set is too small to call a few words a difference, so we read this as level with the package. [MEASURED]

8. Where it falls short

- **IndicXlit's full package is level or ahead in ten of the twelve languages** on the Aksharantar test, and ahead in six of nine on real typing. If you can hold its word lists in memory (hundreds of MB), it is the more accurate choice there.
- **Names are weak, except in Hindi.** In the other eleven languages the name rows get 25% to 60% right; Odia names are the weakest (25% and 30%).
- **Odia, Punjabi, Urdu and Assamese are weak in absolute terms:** 50%, 56%, 58% and 59% of test words right on the first guess. Odia's test is about half names.
- **Odia and Assamese have no real-typing test** (Dakshina has neither), so their scores cover dictionary-style romanisation only.
- **No native-speaker review of the misses,** except Hindi's chat set. Some misses may be acceptable spellings missing from the test's answer lists; some may be errors a native reader would catch at once.
- **No sentence context.** It writes one word at a time, so in Hindi `main` ("I") and `mein` ("in") both become में.
- **Common words it gets wrong,** for example Hindi `kal` → काल (कल is its second choice) and `delhi` → देल्ही (should be दिल्ली).
- **One direction only:** Latin letters to each script. It does not translate, and it does not write Latin from the scripts.
- **Speed is measured on one Mac mini only.** Phones and older laptops are not measured.
- **One training seed per language.**

9. How to reproduce

Run it. The model card's quick start loads any language in a web page or in Node 18 or newer: download `translit.js` and the language's folder from huggingface.co/udgamlabs/Lekhani, then call `Transliterator.load` with the four files. `tr.text(s)` transliterates a sentence; `tr.word(w)` returns the best spelling and the 8 candidates.

Check the scores.

1. Fetch the Aksharantar test split for the language from `ai4bharat/Aksharantar` at revision `e418c1f`, and group its rows by romanised input, trimmed and lower-cased (each input can have several correct spellings).
2. Run each unique input through `tr.word(input).best` with the released files and runner 1.3.0.

3. Count a word as right if the output equals any reference, both trimmed, in Unicode NFC and with the zero-width joiners (U+200C, U+200D) removed.
4. For intervals, resample the test words 2,000 times with replacement. For a gap, score both systems on the same resampled words.

Or run `node tools/score.mjs <language folder> <xxx_test.json>` from the GitHub repository (Node 18 or newer, no dependencies), which does steps 2 to 4 and prints the score and its interval. From the released files it gives Hindi 6,583 of 9,991 (65.89%) and Tamil 8,465 of 11,405 (74.22%), the numbers above. Its interval uses JavaScript's random numbers, so its ends can differ from ours by a few hundredths.

IndicXlit is at github.com/AI4Bharat/IndicXlit (MIT); we ran its model with beam 4 and its language tag, and its package with its defaults.

---

10. Licences

Model weights and the runner (`translit.js` , `translit.src.js`): Apache-2.0. Word lists (`words.bin.gz`): CC0 1.0. The `NOTICE` file credits Aksharantar (AI4Bharat; CC-BY hand-collected part, CC0 mined parts), IndicXlit (AI4Bharat, MIT) as the teacher, IndicCorp v2 (AI4Bharat, CC0) for the word counts, and the CMU Pronouncing Dictionary (BSD-style) for English words. Each model was trained from random initialisation by Udgam Labs; no third-party model weights are included.

---

11. Citation

```
@misc{udgamlabs2026lekhani,
  title  = {Lekhani: in-browser transliteration, under 1 MB per language},
  author = {Prasad, Jitesh K},
  year   = {2026},
  howpublished = {Udgam Labs},
  url    = {https://huggingface.co/udgamlabs/Lekhani}
}
```

---

12. References

1. Madhani, Y. et al. Aksharantar: Towards Building Open Transliteration Tools for the Next Billion Users. 2022. arXiv:2205.03018.
2. AI4Bharat. IndicXlit. github.com/AI4Bharat/IndicXlit (MIT).

3. Doddapaneni, S. et al. Towards Leaving No Indic Language Behind: Building Monolingual Corpora, Benchmark and Models for Indic Languages (IndicCorp v2). ACL 2023.
4. Roark, B. et al. Processing South Asian Languages Written in the Latin Script: the Dakshina Dataset. LREC 2020.
5. Carnegie Mellon University. The CMU Pronouncing Dictionary. github.com/cmusphinx/cmudict.

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