

SANKET: ON-DEVICE, OPEN-VOCABULARY INDIAN SIGN LANGUAGE RECOGNITION IN THE BROWSER

21CSC601T – Case Study

Submitted by

MITHUNISH PRABHAKARAN [RA2512049015065]

Under the guidance of

Dr. S. SARAVANAN

Professor

Department of Computational Intelligence

in partial fulfillment of the requirements for the degree

of

MASTER OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE



**FACULTY OF ENGINEERING AND TECHNOLOGY
SCHOOL OF COMPUTING
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603203**

October 2026



SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603203

BONAFIDE CERTIFICATE

This is to certify that 21CSC601T – Case Study report titled “**SANKET: ON-DEVICE, OPEN-VOCABULARY INDIAN SIGN LANGUAGE RECOGNITION IN THE BROWSER**” is the bonafide work of **MITHUNISH PRABHAKARAN (RA2512049015065)**, who undertook the task of completing the report within the allotted time.

SIGNATURE

Faculty In-Charge

Dr. S. Saravanan

Professor

Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur Campus, Chennai

SIGNATURE

Head of the Department

Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur Campus, Chennai

DECLARATION

I, **Mithunish Prabhakaran** (RA2512049015065), a student of M.Tech in Artificial Intelligence, Department of Computational Intelligence, School of Computing, SRM Institute of Science and Technology, Kattankulathur, hereby declare that this case study report, titled “**Sanket: On-Device, Open-Vocabulary Indian Sign Language Recognition in the Browser,**” submitted for the course 21CSC601T – Case Study in partial fulfillment of the requirements for the degree of Master of Technology in Artificial Intelligence, is a record of original work carried out by me under the guidance of **Dr. S. Saravanan**, Professor, Department of Computational Intelligence.

I further declare that:

1. the work contained in this report is my own except where indicated, and all sources of information, data, figures and software have been acknowledged and cited;
2. all datasets were used within their published licences and terms of use (Section 3.2), and no data collected from people was used without their consent;
3. no part of this report has been submitted, in this or any other form, for the award of any other degree or diploma of this or any other university or institution;
4. software tools, including AI-based writing and coding assistants, were used as tools; every design decision, measurement and claim in this report is mine and is reproducible from the committed code (Appendix D).

Place: Kattankulathur

Date: _____

Mithunish Prabhakaran

RA2512049015065

ACKNOWLEDGEMENTS

I express my sincere gratitude to **Dr. S. Saravanan**, Professor, Department of Computational Intelligence, for guiding this case study through its four reviews, and for the questions that kept its claims tied to evidence.

I thank the Head of the Department of Computational Intelligence and the faculty of the School of Computing, SRM Institute of Science and Technology, for the structure of the 21CSC601T course, whose four reviews shaped the work, and the members of the Project Review Panel for their feedback at Reviews 0, 1 and 2. I thank the SRM IST Program Office for organising the M.Tech programme and its reviews.

This work stands on public resources created by others: the INCLUDE, CISLR, ISLRTC and RKMVERI Indian Sign Language corpora; the Google Isolated Sign Language Recognition corpus (PopSign ASL v1.0) released by the Deaf Professional Arts Network and the Georgia Institute of Technology under CC BY 4.0; and the open-source MediaPipe, PyTorch, ONNX Runtime and Typst projects. Above all, I am grateful to the Deaf signers whose recorded signing made every model in this report possible.

Mithunish Prabhakaran

ABSTRACT

Indian Sign Language (ISL) is the first language of millions of Indians with hearing disability, yet India’s directory of certified interpreters lists only 339 people, so most everyday interactions happen without one. Automatic sign recognition can help, but published systems recognise small fixed vocabularies, send camera video to servers, and are evaluated on the same signers they were trained on. This case study builds **Sanket**, a web application that recognises ISL signs from an ordinary webcam entirely on the user’s device: MediaPipe reduces every frame to 107 body and hand landmarks and the frame is discarded. A 5.2 MB int8 classifier names 260 signs with calibrated confidence, a retrieval path reaches an 8,188-word vocabulary, and a model-free matcher runs alongside. The classifier is an articulator-token transformer pretrained with ISL-JEPA, a self-supervised latent-prediction objective over a 123,340-clip landmark pool; it is then taught 250 American Sign Language signs from Google’s 94,290-clip ISLR corpus, fine-tuned on ISL with gloss-matched dictionary clips, synthetic signers and a phonological head, and twelve such models are distilled into one browser model. Accuracy is measured by one pre-stated test computed exactly as the application runs: 599 clips from a signer never seen in training and from dictionary clips never trained on. The deployed model scores 0.553 on this test and 0.514 on the unseen signer, against 0.147 and 0.332 at Review 1, meeting the 0.50 targets set there. Every claim rests on at least four training seeds, and the negative results, including ensembles and self-supervision that did not help the unseen signer, are reported. The work is aligned with SDG 10, Reduced Inequalities.

TABLE OF CONTENTS

Chapter No	Title	Page No
1	Introduction	1
1.1	Background	1
1.2	Industry and Domain Context	2
1.3	Problem Statement	2
1.4	Objectives	3
1.5	Scope and Limitations	4
1.6	Sustainable Development Goal Alignment	4
1.7	Organization of the Report	5
2	Literature Review	7
2.1	Introduction	7
2.2	Review of Existing Approaches	7
2.3	Comparative Analysis and Gap Identification	10
2.4	Novelty of This Work	12
3	Methodology	14
3.1	Approach Overview	14
3.2	Data Collection and Requirements Gathering	15
3.3	Architecture	17
3.4	Implementation Strategy	21
3.5	Justification for the Proposed System	22
4	Implementation	25
4.1	Execution and Step-by-Step Analysis	25
4.2	Core Algorithms and Configurations	26
4.3	Challenges Faced and Mitigation	34
5	Results and Discussion	36
5.1	Results	36
5.2	Discussion	44
5.3	Practical Implications	45
6	Conclusion and Future Scope	47
6.1	Conclusion	47
6.2	Future Scope	48
	References	50
Appendix A	Source Code	54
A.1	Canonical window built in the browser	54

Chapter No	Title	Page No
A.2	Ensemble distillation step	55
A.3	Temperature calibration with cross-fit	55
A.4	The resting test of sign by sign	56
Appendix B	Research Paper	57
Appendix C	Screenshots of the Working System	63
Appendix D	Reproducibility	68
D.1	Where the work lives	68
D.2	Versions of record	68
D.3	Commands	69
D.4	Evaluation discipline	69
D.5	Compute	69
Appendix E	Record of Mentor Meetings	70

LIST OF FIGURES

S.No	Description	Page No
1	Figure 1.1: Three gaps and this work’s answer	3
2	Figure 1.2: The four reviews	5
3	Figure 3.1: The approach in five steps	14
4	Figure 3.2: The system at run time	18
5	Figure 3.3: How a model is made	19
6	Figure 3.4: Transfer from American Sign Language	19
7	Figure 3.5: The accuracy test	20
8	Figure 3.6: The dataset review loop	21
9	Figure 4.1: Synthetic signers from real clips	29
10	Figure 4.2: The phonological head	30
11	Figure 4.3: Twelve models into one	31
12	Figure 4.4: Sign by sign	33
13	Figure 5.1: Model lineage on the review timeline	37
14	Figure 5.2: From jepa-rr4 to jepa-asl-b3 and the ensembles	38
15	Figure 5.3: ASL transfer: four-seed means	39
16	Figure 5.4: Shape match as a helper, measured	42
17	Figure 6.1: Future scope	48
18	Figure C.1: Studio	63
19	Figure C.2: Sign by sign	64
20	Figure C.3: Learn	64
21	Figure C.4: Practice	65
22	Figure C.5: Dictionary look-up	65
23	Figure C.6: Model Lab	66
24	Figure C.7: Eval bench	66
25	Figure C.8: Dataset tab	67
26	Figure C.9: Settings	67

LIST OF TABLES

S.No	Description	Page No
1	Table 1.1: Alignment with the Sustainable Development Goals.	5
2	Table 1.2: Course milestones and where this report answers each deliverable.	5
3	Table 2.1: Comparison of ten core works	11
4	Table 2.2: What the literature provides and what this project adds.	12
5	Table 3.1: Datasets and their roles	15
6	Table 3.2: Functional requirements.	16
7	Table 3.3: Non-functional requirements and how they were verified.	17
8	Table 3.4: Run-time artifacts.	18
9	Table 3.5: Technology stack.	21
10	Table 3.6: Time frame: proposed at Review 0 and actual.	22
11	Table 3.7: Design choices and the evidence behind them	23
12	Table 4.1: Challenges faced and their mitigation	34
13	Table 5.1: Model lineage on the accuracy test	36
14	Table 5.2: Accuracy test per seed: control versus the v4 pack	38
15	Table 5.3: ASL transfer: single models, four-seed means	39
16	Table 5.4: ASL transfer: four-model ensembles	40
17	Table 5.5: Candidate students and the promotion rule	40
18	Table 5.6: The deployed model against its predecessor	41
19	Table 5.7: Sign by sign against sliding windows	41
20	Table 5.8: Data findings	43
21	Table 5.9: Negative results	43
22	Table 5.10: Objectives and how they were met	45
23	Table 5.11: Targets set at Review 1 and their status	45
24	Table D.1: Versions of record	68
25	Table D.2: Commands	69
26	Table E.1: Review milestones	70

LIST OF ALGORITHMS

S.No	Description	Page No
1	Algorithm 4.1: Landmark canonicalisation (browser, extraction and evaluation)	27
2	Algorithm 4.2: ISL-JEPA pretraining step	28
3	Algorithm 4.3: Supervised ASL stage before the ISL fine-tune	29
4	Algorithm 4.4: Ensemble distillation	31
5	Algorithm 4.5: Temperature calibration with cross-fit	32
6	Algorithm 4.6: Mirror test-time augmentation	33
7	Algorithm 4.7: Sign-by-sign segmentation	33

LIST OF ABBREVIATIONS

Abbreviation	Expansion
ASL	American Sign Language
AURC	Area under the risk–coverage curve
CE	Cross-entropy (loss)
CISLR	Corpus for Indian Sign Language Recognition
DTW	Dynamic time warping
ECE	Expected calibration error
EMA	Exponential moving average
fp32	32-bit floating point
int8	8-bit integer (quantised)
ISL	Indian Sign Language
ISLR	Isolated Sign Language Recognition (the Google corpus, Kaggle asl-signs)
ISLRTC	Indian Sign Language Research and Training Centre
JEPA	Joint-Embedding Predictive Architecture
KD	Knowledge distillation
KL	Kullback–Leibler divergence
MoM	Minutes of meeting
NLL	Negative log-likelihood
ONNX	Open Neural Network Exchange (model format)
pp	Percentage points
RKMVU	The RKMVERI ISL dictionary signer used as the unseen-signer test (220 clips)
RTMW	Real-Time Multi-person Whole-body pose estimator
SDG	Sustainable Development Goal (United Nations)
SLR	Sign language recognition
SSL	Self-supervised learning
TTA	Test-time augmentation
WASM	WebAssembly

CHAPTER 1

Introduction

1.1 Background

A signed language is a complete natural language, with its own vocabulary, grammar and word order. Indian Sign Language (ISL) is not a manual encoding of Hindi or English: it is the language in which Deaf Indians are at home, and the language of Deaf schools, families and friendships. Each sign is built from a small set of parameters — the shape of the hand, where it is held relative to the body, how it moves, and which way the palm faces — together with non-manual signals such as facial expression. Two signs can differ in only one of these parameters, and the same sign is performed slightly differently by every signer, at a different speed, with a different body, and sometimes with the other hand.

India's 2011 census counted 5.07 million people with a hearing disability [1], and the directory of certified ISL interpreters maintained by the Indian Sign Language Research and Training Centre (ISLRTC) lists 339 interpreters for the whole country [2]. The surrounding society runs on speech and writing: a Deaf customer at a bank counter, a Deaf patient describing symptoms, or a Deaf student in a mainstream classroom depends on an interpreter being present, and for almost every such interaction there is none.

Sign language recognition is the task of mapping video of signing to the words being signed. In its *isolated* (word-level) form, studied in this report, the input is a short clip that contains one sign and the output is the sign's gloss — the word used to name it. Recognition can help in two ways that do not require a full translation system: it lets hearing people and learners practise and check their own signing, and it gives a Deaf signer a way to be understood by a device when no interpreter is present. Both uses need the recogniser to work for *the person in front of the camera*, who is, by definition, not one of the people in its training data.

Two technical developments make such a recogniser practical on ordinary hardware. Pose estimators such as MediaPipe [3], [4] reduce each video frame to a few dozen body and hand landmarks in real time on a laptop or phone, so a model can learn from the motion of the hands rather than from pixels. And modern browsers can execute quantised neural networks locally through WebAssembly [5], so a recogniser can run inside a web page with no server in the loop.

1.2 Industry and Domain Context

Accessibility technology for Deaf users in India today consists mostly of human services — interpreters, video-relay calls to an interpreter, captioning — and of learning resources: the ISLRTC dictionary of more than ten thousand recorded words, school curricula, and mobile apps that show how a sign is made. What is missing is the *reverse direction*: a tool that watches signing and says what was signed. Without it, a learner has no feedback on their own signing, and a Deaf user cannot be understood by a device.

Recognition research has followed the resources. Word-level recognition matured on American Sign Language (ASL), where WLASL and MS-ASL provide thousands of glosses from hundreds of signers [6], [7]. For ISL, INCLUDE offers 4,287 studio videos of 263 words [8], CISLR about 4,700 words with one or two clips each [9], and iSign 118,000 sentence pairs without word labels [10]. The realistic regime for ISL is therefore *thousands of words with one or two examples each*, recorded by a handful of signers — the opposite of the large, balanced datasets on which recognition models are usually built.

Three constraints of the Indian context shape this case study. **Cost and connectivity**: a tool that needs a server and a fast connection is not free to run and does not work everywhere; a tool that runs in a browser on the user’s own device costs nothing per use and works offline once loaded. **Privacy**: a camera feed of a person signing, possibly about health or money, should not leave the device. **Signer variety**: the people who will use the tool are not the dozen signers who recorded the training corpora, so accuracy on a stranger is the only accuracy that matters.

1.3 Problem Statement

The literature reviewed in Chapter 2 and the measurements made in this project at Reviews 0 and 1 identify three gaps (Figure 1.1).

Closed vocabularies. Published ISL recognisers classify a fixed list of signs — 263 for INCLUDE — and force every other sign onto its nearest listed word. Real ISL has thousands of words, most with one or two reference clips.

Server-side inference. Recognisers that need the camera stream on a server impose connectivity, cost and a privacy exposure precisely on the users who most need them to be free and private.

The cross-signer cliff. At Review 1 this project measured the gap directly: a supervised model that scored 0.82 on held-out signers of its own corpus fell to 0.33 on a signer

from a different corpus, and to 0.12 when that signer signed with the left hand. Within-corpus evaluation, even when signer-disjoint, did not predict this.

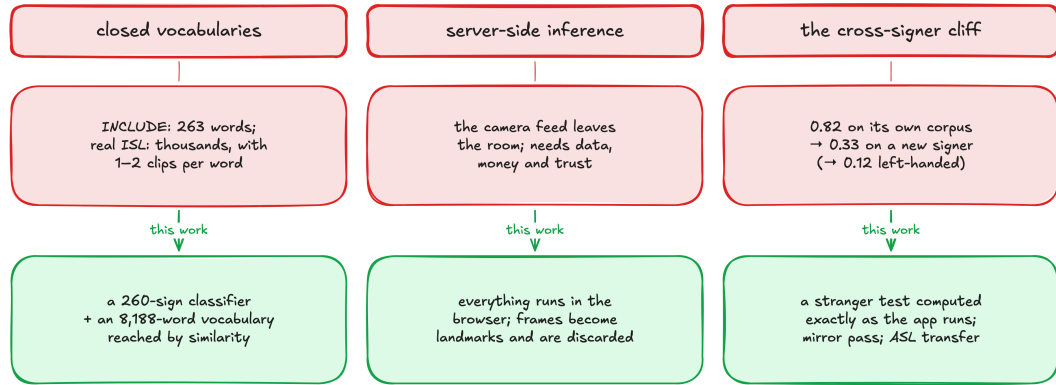


Figure 1.1. The three gaps this case study addresses. Each column shows what current systems do (red) and what Sanket does instead (green).

Problem statement

Build an Indian Sign Language recogniser that runs entirely in the user’s browser, answers over a large vocabulary, and is measured — and improved — on signers it has never seen, with the measurement computed exactly as the deployed application computes it.

1.4 Objectives

The objectives were fixed at Review 0 and refined at Review 1; Table 5.10 in Chapter 5 states how each one was met.

1. **To analyse the current state of ISL recognition** — the datasets, the recognition approaches and their evaluation practice — and to measure, on our own pipeline, where cross-signer accuracy is lost (Chapter 2, Section 1.3).
2. **To design and build an on-device recogniser** that runs entirely in a web browser: landmark extraction, recognition and the user interface in the tab, with no image, video or landmark stream leaving the device (Chapter 3).
3. **To cover the vocabulary a learner meets** with two complementary recognisers: a calibrated classifier for a 260-sign set and a retrieval path over the 8,188-word dictionary vocabulary (Chapter 3, Chapter 4).
4. **To close the cross-signer gap** with methods that fit ISL’s data regime — self-supervised pretraining on unlabelled landmarks, targeted data, transfer from a sign language with more signers, and ensembles distilled into one browser-sized model (Chapter 4).

5. **To evaluate honestly:** one pre-stated accuracy test from a signer never seen and dictionary clips never trained on, computed through the deployed numeric pipeline, with at least four training seeds behind every claim and negative results reported (Chapter 5).
6. **To ship a useful product** around the recogniser — Learn, Practice, a recognition Studio and a sign-by-sign mode — at `studio.sanket.kadal.cc`.

1.5 Scope and Limitations

In scope. Word-level (isolated) recognition of ISL signs from a webcam; the 260-sign closed set and the 8,188-word full vocabulary; on-device inference in the browser; the training pipeline that produces the models (self-supervised pretraining, transfer from ASL, fine-tuning, distillation, calibration, export); the evaluation harness; the dataset-quality system; and the learner-facing application, including a sign-by-sign mode that segments continuous signing at rest positions.

Out of scope. Sentence-level translation (the application assembles recognised glosses with rules only); fingerspelling as a dedicated modality; non-manual grammar (the face is tracked but not modelled as grammar); dialect coverage beyond the source dictionaries.

Limitations of the evaluation. The unseen-signer test is one signer (220 clips), so one clip is worth 0.45 percentage points on its own and 0.17 points on the combined test; claims therefore rest on at least four training seeds. The second test set (CISLR) shares presenters with CISLR clips used in training, so gains on it partly reflect presenter familiarity. No new signers were recorded for this project, by design: the work had to improve the stranger number without contacting people or organisations for new data.

1.6 Sustainable Development Goal Alignment

The course asks every case study to align with one of the United Nations Sustainable Development Goals (SDGs). This project aligns with **SDG 10 — Reduced Inequalities** [11], and secondarily with **SDG 4 — Quality Education** [12] (Table 1.1).

SDG 10 asks countries to reduce inequality within and among themselves; its target 10.2 is the social, economic and political inclusion of all people irrespective of attributes that include disability. Language access is a direct instance: a Deaf person’s participation in ordinary transactions depends on whether the hearing side can understand signing. A recogniser that is free, works in any browser, and never uploads the user’s video lowers the cost of that access to the cost of a webcam. SDG 4 concerns inclusive education: learning ISL is how hearing family members, teachers and classmates meet a Deaf

person halfway, and the Learn and Practice surfaces turn the recogniser into a patient practice partner that scores every attempt on the learner’s own device.

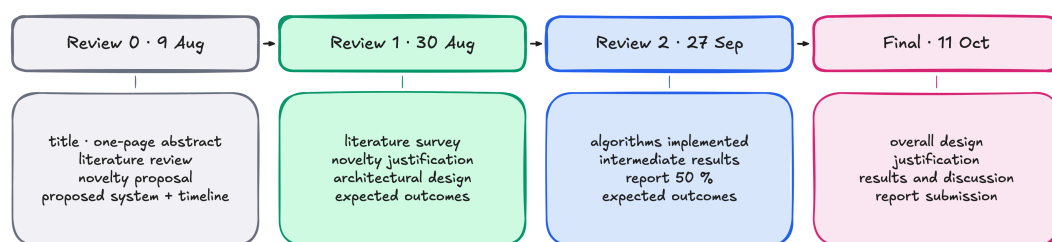
Table 1.1. Alignment with the Sustainable Development Goals.

Goal	Concern relevant here	What Sanket contributes
SDG 10 (primary)	Inclusion irrespective of disability (target 10.2)	Free, private, on-device ISL recognition; no interpreter, server or data plan needed
SDG 4 (secondary)	Inclusive education and lifelong learning (targets 4.5, 4.a)	Learn and Practice surfaces for ISL learners, scored on the learner’s own device

1.7 Organization of the Report

Chapter 2 reviews the literature and identifies the gap. Chapter 3 presents the methodology: the approach, the data and requirements, the architecture and the implementation strategy, and justifies the proposed system against its alternatives. Chapter 4 describes how the work was executed, the core algorithms and the challenges met. Chapter 5 reports and discusses the results, including the negative ones, and Chapter 6 concludes and sets out the future scope. The appendices give the key source code (A), a research paper prepared from this work (B), screenshots of the working system (C), the reproducibility record (D) and the record of mentor meetings (E).

The case study was conducted through the four reviews prescribed by the course guidelines (Figure 1.2). Table 1.2 maps each review’s deliverables to where they are answered in this report.



each review's deliverables are answered in this report; the final review adds Chapters 5 and 6

Figure 1.2. The four reviews of the 21CSC601T course and what each asked for.

Table 1.2. Course milestones and where this report answers each deliverable.

Review	Deliverable	Answered in
Review 0 · 9 Aug 2026	Title identification; one-page abstract; literature review; novelty	Title page; Abstract; Chapter 2; Section 2.4; Section 3.1 and Table 3.6

Review	Deliverable	Answered in
	proposal; proposed system with time frame	
Review 1 · 30 Aug 2026	Literature survey; novelty justification; architectural design; expected outcomes	Chapter 2; Section 2.4 and Section 3.5; Section 3.3; Table 5.10
Review 2 · 27 Sep 2026	Implementation of algorithms; intermediate results; report completion (50%); expected outcomes	Chapter 4; Chapter 5 (lineage to 27 Sep); the Review 2 report; Section 6.2
Final review · 11 Oct 2026	Overall design; justification for the proposed system; results and discussion; report submission	Chapter 3; Section 3.5; Chapter 5; this report

CHAPTER 2

Literature Review

2.1 Introduction

This chapter reviews the work on which the design rests and the work it departs from. It consolidates the surveys submitted at Review 0 and Review 1 and adds the literature that the final round of work drew on: transfer across sign languages and phonological structure. It is organised by the questions the implementation had to answer — which data exist for ISL, which input representation suits a browser, how to recognise with one or two examples per word, how to learn without labels, how to transfer from another sign language, and how to compress and calibrate a model for deployment — and closes with a structured comparison of ten core works and the gap this case study fills. Every factual statement about prior work carries a citation verified against its publishing venue.

2.2 Review of Existing Approaches

2.2.1 Word-level datasets, and where ISL stands

Word-level recognition matured on ASL corpora: WLASL provides more than 2,000 glosses in over 21,000 videos from more than 100 signers [6], and MS-ASL 1,000 classes from over 200 signers [7]. For ISL, INCLUDE offers 4,287 studio recordings of 263 words in 15 categories [8] — enough for supervised classification, but performed by few signers and far below the size of the language. CISLR inverted the trade-off: about 4,700 words, most with a single reference, framed as one-shot retrieval [9]. iSign added more than 118,000 video–sentence pairs for sentence-level ISL [10]. The ISLRTC publishes per-word dictionary videos that this project treats as first-class supervision. Google’s Isolated Sign Language Recognition corpus, released through a Kaggle competition [13], is the landmark version of PopSign ASL v1.0: 250 ASL signs recorded on smartphones by Deaf signers, extracted with the same MediaPipe pose model this project uses [14].

2.2.2 Appearance-based and pose-based recognition

Video networks such as I3D [15] set the closed-set baselines on WLASL and INCLUDE, but they consume raw RGB: heavy for in-browser inference, tied to studio appearance, and requiring the camera stream as model input. Pose-based work replaces pixels with

joints. ST-GCN introduced graph convolutions over joint sequences [16]; SAM-SLR carried multi-modal skeleton ensembles to the top of sign recognition challenges [17]; SPOTER showed that a small pose transformer, with signing-space normalisation and hand-local coordinates, is competitive and cheap [18]. OpenHands standardised pose-based datasets and checkpoints across six sign languages, including ISL [19]. MediaPipe supplies body and hand landmarks on-device at camera rate [3], [4]; RTMW is a stronger whole-body model [20], evaluated in this project as an alternative front end (Section 5.1.9).

2.2.3 The one-shot, open-vocabulary regime

With one reference per word, classification becomes matching: learn an embedding in which performances of the same sign cluster, and recognise by distance to class prototypes [21]. Additive-margin losses such as ArcFace [22] and CurricularFace [23] make such embeddings discriminative. CISLR’s baseline follows this framing and transfers features learned on ASL to one-shot ISL prediction [9]. Before deep learning, gesture recognition was alignment: dynamic time warping (DTW) finds the least costly monotone alignment between two sequences [24], and audio fingerprinting showed at industrial scale that retrieval against a reference library works when the query and the reference are performances of the same object [25].

2.2.4 Self-supervised learning on video, skeletons and signs

Joint-Embedding Predictive Architectures (JEPA) learn by predicting the latent representation of hidden content from visible context, rather than reconstructing pixels or coordinates. I-JEPA established the approach for images [26]; V-JEPA extended it to video [27]; V-JEPA 2 scaled it [28], and V-JEPA 2.1 added dense predictive losses [29]. S-JEPA predicts the latents of masked joints with an exponential-moving-average target encoder and target centring, with its largest gains in the low-label regime [30]. Skeleton2vec [31] and an IJCV benchmark of skeleton self-supervision [32] support the same finding: latent targets beat coordinate reconstruction. For sign language, SignBERT+ introduced hand-model-aware masked pretraining [33]; SHuBERT predicts masked multi-stream clusters over about a thousand hours of ASL [34]; SignMAE argues that masking should follow the articulators [35]; SSL-SLR cautions that contrastive negatives are ill-posed because different signs share sub-movements [36]; Uni-Sign pretrains a unified model at scale [37]; YouTube-SL-25 released identifiers for more than 3,000 hours of signing in 25 languages [38].

2.2.5 Transfer across sign languages

Sign languages differ in vocabulary but share their phonological building blocks — handshapes, locations and movements — so a model can carry what it learns on one language to another. OpenHands reported the first cross-lingual transfer for isolated recognition by pretraining on Indian sign data [19]. Cross-lingual few-shot recognition learns from a source language to recognise a target language’s signs from a few examples [39]; transfer learning from larger corpora rescues extremely low-resource languages such as Irish Sign Language [40]; cross-dataset transfer within one language helps under-resourced datasets [41]; and pretraining on a large multilingual corpus improves recognition across target languages [42]. PhonSSM makes the phonological structure explicit: separate branches predict handshape, location, movement and orientation, and a sign is recognised from their composition [43].

2.2.6 Distillation, calibration and deployment

Knowledge distillation compresses an ensemble into one deployable model by training a student on the teacher’s temperature-softened output distribution [44]. Label smoothing regularises over-confident classifiers [45]. Modern networks are often miscalibrated, and a single temperature fitted on held-out data is a simple remedy that cannot change the ranking of classes [46]. Integer quantisation makes models small and fast enough for commodity processors [47], and ONNX Runtime Web executes them in the browser in WebAssembly [5]. Evaluation practice is converging on signer independence: the MSLR workshop runs an explicit signer-independent track [48], and continuous recognition treats each signer as a domain whose identity should be removed [49].

2.2.7 Critical analysis of ten core works

INCLUDE [8] established ISL word-level recognition with a 263-word, 15-category studio corpus and a study of feature extractors, reaching 85.6% on the full set. Its strength is clean, balanced data; its limitation for this work is that the signers are few and the evaluation does not isolate an unseen signer, so its accuracy does not predict accuracy on a stranger.

CISLR [9] framed ISL at dictionary scale as one-shot retrieval and transferred ASL-learned features to it. The framing is the one this project adopts for the full vocabulary. Its limitation is inherent to the data: with one or two clips per word, signer variation is barely sampled.

iSign [10] is the largest ISL resource, with sentence-level pairs and five benchmark tasks. It supplies ISL signing for self-supervised pretraining but no isolated-word labels, so it cannot directly train or test a word recogniser.

WLASL [6] showed that pose and appearance baselines are comparable at 2,000 ASL glosses (up to 66% top-10). Its random splits share signers between training and test; later work found that such models fall sharply on unseen signers, which is the failure this project measures for ISL.

PopSign ASL v1.0 [14] is the largest isolated-sign dataset by example count, recorded by Deaf signers on smartphones, with a signer-disjoint split (84.2% test accuracy for an LSTM baseline). It is a closed 250-sign game vocabulary in ASL, not ISL, but its many signers make it a candidate source for transfer, which Section 5.1.4 tests.

SPOTER [18] is a small pose transformer aimed at handheld devices, with signing-space normalisation and hand-local coordinates (63.2% top-1 on WLASL100). It demonstrates that pose models can be cheap; its evaluation stops at 300 glosses with a closed-set head.

OpenHands [19] standardised pose data and checkpoints across six sign languages, including ISL, with self-supervised pretraining and the first cross-lingual transfer for isolated recognition. Its heads are per-dataset and closed-set, and its ISL results inherit INCLUDE’s signer limitation.

SignBERT+ [33] introduced hand-model-aware masked pretraining with strong results across recognition and translation. It is hands-only, and its pretraining data overlap its fine-tuning datasets, so generalisation to unseen data is not isolated.

SHuBERT [34] learns one multi-stream representation from about a thousand hours of ASL and transfers it to several tasks. Its strength is scale; its limitations here are the language (ASL only) and a compute budget far beyond a browser.

Uni-Sign [37] casts every sign task as translation after large-scale generative pretraining and reports strong results on WLASL and MS-ASL. At roughly 580 million parameters it illustrates the gap between the state of the art and what can run on a learner’s laptop in a browser tab.

2.3 Comparative Analysis and Gap Identification

Table 2.1 summarises the ten works. Read across the last column, the pattern is consistent: the strongest results come from large models, closed vocabularies or seen signers, and none of the works combines open-vocabulary ISL, on-device execution and an unseen-signer evaluation computed through the deployed pipeline.

Table 2.1. Comparison of ten core works. “Limitation” is judged against this project’s targets — on-device inference, an open vocabulary and unseen signers; entries marked † are our reading rather than the authors’ statement.

Work	Venue	Data and language	Input	Contribution	Limitation for this project
INCLUDE [8]	ACM MM 2020	4,287 videos, 263 signs; ISL	RGB	First public ISL word-level dataset; augmentation and feature study; 85.6% on INCLUDE	Closed 263-word set, few signers; no unseen-signer protocol†
CISLR [9]	EMNLP 2022	4,700 words, 1–2 clips each; ISL (ASL source)	RGB	Dictionary-scale ISL corpus; one-shot learner with ASL-transferred features	Signer variation barely sampled with one clip per word†
iSign [10]	ACL Findings 2024	>118K video–sentence pairs; ISL	RGB, pose	Largest ISL–English corpus; five benchmark tasks	Sentence level; no isolated-word labels†
WLASL [6]	WACV 2020	>2,000 glosses, >100 signers; ASL	RGB, pose	Large word-level ASL benchmark; Pose-TGCN; up to 66% top-10	Seen-signer splits; large drop on unseen signers reported by later work
PopSign [14]	NeurIPS 2023 D&B	175K examples, 250 signs, 47 Deaf signers; ASL	RGB (phone)	Largest isolated-sign dataset by examples; signer-disjoint split; 84.2% baseline	Closed 250-sign one-handed game vocabulary†
SPOTER [18]	WACV-W 2022	WLASL100/300, Pose LSA64; ASL, LSA	Pose	Small pose transformer for handheld devices; 63.2% top-1 WLASL100	Up to 300 glosses, closed-set head†
OpenHands [19]	ACL 2022	7 datasets, 6 languages incl. ISL	Pose	Pose checkpoints across languages; SSL; first cross-lingual ISLR transfer	Per-dataset closed-set heads†
SignBERT+ [33]	TPAMI 2023	MSASL, WLASL, SLR500; ASL, CSL	Hand pose	Hand-model-aware masked pretraining	Hands only; pretraining overlaps fine-tuning data
SHuBERT [34]	ACL 2025	1,000 h; ASL	Multi-stream	Masked cluster prediction; one representation for several tasks	ASL only; far beyond a browser budget†

Work	Venue	Data and language	Input	Contribution	Limitation for this project
Uni-Sign [37]	ICLR 2025	CSL-News 1,985 h; CSL, ASL	Pose + RGB	Unified generative pretraining; tasks cast as translation	580M parameters; not deployable on-device†

Table 2.2. What the literature provides and what this project adds.

Prior work provides	Prior work does not provide
Closed-set ISL classification at 263 classes [8]	Recognition over the dictionary-scale vocabulary, deployed on end-user devices
One-shot retrieval framing for ISL [9]	That framing running in the browser at camera rate
Latent-predictive SSL for images, video and skeletons [26], [30]	A JEPA on sign-language landmarks, or any ISL application of one
Cross-lingual transfer and phonological structure [19], [39], [43]	A measured account of what transfer from ASL does for an <i>unseen ISL signer</i> , separately from ensembling
Signer-independent evaluation ethos [48]	An unseen-signer test computed through the exact deployed numeric contract
Distillation and temperature calibration [44], [46]	A calibration that must transfer across two corpora before it ships

2.4 Novelty of This Work

The novelty proposed at Review 0 and justified at Review 1 is summarised below, as it stands after the final round of work.

1. **Open-vocabulary ISL recognition shipped on-device.** A calibrated 260-sign classifier and an 8,188-word retrieval vocabulary run together in a browser tab, with no server in the recognition path; vocabulary grows by adding references, not by retraining.
2. **A stranger test computed through the deployed pipeline.** One pre-stated accuracy test from a signer never seen and from dictionary clips never trained on, replayed through the same int8 graph, canonical tensors and mirror pass that the application executes, with at least four training seeds per claim and the unseen-signer number reported beside every aggregate.
3. **ISL-JEPA.** A joint-embedding predictive objective on articulator tokens of sign-language landmarks, pretrained on a 123,340-clip unlabelled pool, with the measured finding that only the first four layers transfer.
4. **Transfer from ASL to an unseen ISL signer, measured against ensembling.** A supervised stage on 94,290 clips of Google’s ISLR corpus lifts single models on

the unseen signer by up to 3.5 percentage points, but four-model ensembles reach the same level without it — a distinction that reported transfer gains rarely make.

5. **Data review as a measured lever.** A dataset-quality system whose reviewed contract alone lifted the accuracy test by 7.0 points on four of four seeds.

Novelty in one sentence

A deployed, private, open-vocabulary ISL recogniser whose accuracy on a stranger is measured exactly as it runs — and an account of which methods actually move that number and which only look as if they do.

CHAPTER 3

Methodology

3.1 Approach Overview

The approach has five steps (Figure 3.1). **First**, every video frame is reduced to landmarks and discarded: 107 canonical body and hand points per frame, 64 frames per window. **Second**, an encoder learns the structure of signing without labels, by self-supervised latent prediction (ISL-JEPA) on a pool of 123,340 unlabelled clips. **Third**, it is specialised with transfer and targeted data: a supervised stage on 250 American Sign Language signs from 21 Deaf signers, then the ISL fine-tune with gloss-matched dictionary clips, synthetic signers and a phonological head. **Fourth**, many such models are averaged into an ensemble and distilled back into one browser-sized model, exported to int8. **Fifth**, the model is calibrated and tested on a stranger, and it replaces the deployed model only if it beats it both on the accuracy test and on the unseen signer.

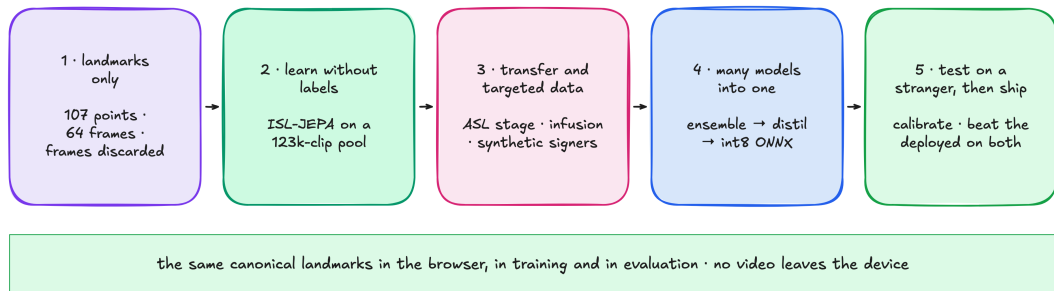


Figure 3.1. The approach in five steps. The same canonical landmarks are used in the browser, in training and in evaluation.

The system is three architectures bound by one numeric contract. The **run-time** architecture is what the user’s browser executes; the **training** architecture produces the models; the **evaluation** architecture decides whether a model may replace the one in production. A fourth loop, the **data pipeline**, keeps the training data honest. The canonicalisation — mid-shoulder anchor, shoulder-width scale, aspect correction, absent hands zeroed — is executed by the browser at inference, by the extraction jobs and by the evaluation harness, so a change reaches all three or none.

Four design principles follow from the problem statement:

1. **Nothing leaves the device.** Frames become landmarks in a worker and are discarded; models and vocabulary banks are static files cached by the browser.

2. **The bundle is the contract.** Every threshold, label list and landmark index is read from the model bundle at run time, so a retrained model is a file drop, not an application release.
3. **Measure what the user sees.** Evaluation replays the same int8 graph, the same canonical tensors and the same mirror pass that the application runs.
4. **Name things by their size.** The interface says *260-sign set, full vocabulary* and *shape match*, so that no label can be read backwards.

3.2 Data Collection and Requirements Gathering

3.2.1 Data sources

Table 3.1 lists every corpus, its size, its licence terms and its role. Two sets are never trained on and form the accuracy test: the RKMVU dictionary signer, who appears nowhere in training, and the CISLR *test role*. The INCLUDE corpus trains the 260-sign classifier; CISLR and ISLRTC clips whose gloss matches one of the 260 classes are added to training (“infusion”); the unlabelled pool feeds self-supervised pretraining; and the Google ISLR corpus provides the transfer stage. No new signers were recorded: improving the unseen-signer number without contacting people or organisations for new data was a constraint of the final round.

Table 3.1. Datasets, their licence terms and their roles. RKMVU and the CISLR test role are never trained on.

Corpus	Size	Terms	Role in this project
INCLUDE [8]	4,287 videos, 263 words, 15 categories; 4,284 extracted	Research use under the release terms	Trains the 260-sign classifier (260 classes)
CISLR [9]	7,050 videos, 4,700 words	Research use under the release terms	Infusion clips (training role); 379-clip test role, never trained on
ISLRTC dictionary	4,804 of 11,993 entries extracted (download quota)	Official public reference videos; only landmarks kept	Infusion clips; full-vocabulary references
RKMVU (RKMVERI)	220 clips, one signer	Public dictionary videos	The unseen-signer test; never trained on
Unlabelled pool	123,340 clips from five ISL sources	As each source	ISL-JEPA pretraining only (no labels)
Google ISLR [13], [14]	94,477 clips, 250 ASL signs, 21 Deaf signers; 94,290 canonicalised	CC BY 4.0, any use; raw data not redistributed	Supervised transfer stage before the ISL fine-tune

3.2.2 Data collection pipeline

Video corpora were converted to landmarks once, through the same contract the browser uses, by extraction jobs on cloud CPU task queues; only landmarks are stored. The Google ISLR corpus already consists of MediaPipe landmarks, so it needed no video processing: a CPU task on Beam read the 40 GB competition archive by HTTP byte ranges, never writing it to disk, and kept the pose and hand points of all 94,477 clips in 77 seconds at a cost of about two US cents. Because the corpus stores coordinates normalised by the video’s width and height without recording the video size, the height-to-width ratio was estimated from five independent cues against our own pixel-true corpora (median 1.38) and fixed at 4:3, PopSign’s capture format; 187 clips shorter than four frames were dropped, leaving 94,290.

3.2.3 Hardware and software prerequisites

The application needs a computer or phone with a webcam and a current browser (Chrome, Edge, Firefox or Safari) with WebAssembly; no installation, account or GPU. Development and training used a MacBook Pro (Apple M2 Pro, 16 GB), cloud GPUs (Modal A10G until 27 September; Google Colab Pro A100 80 GB; Beam), and cloud CPU task queues for extraction.

3.2.4 Functional requirements

Table 3.2. Functional requirements.

ID	Requirement	Where met
F1	Recognise an ISL sign from a webcam window and show the top answers with calibrated confidence	Studio, Practice
F2	Reach the full 8,188-word vocabulary by similarity to stored examples	Full-vocabulary path
F3	Recognise continuous signing sign by sign, splitting the stream at rest positions	Sign-by-sign mode
F4	Teach a word: show it signed by a figure and by a filmed reference	Learn
F5	Score a learner’s attempt at a chosen word and credit it above a calibrated bar	Practice
F6	Let a developer swap models, inspect model cards and run the accuracy test in the browser	Lab, Eval bench
F7	Let a developer review and correct the training data at clip, word and corpus level	Dataset, Review tabs

3.2.5 Non-functional requirements

Table 3.3. Non-functional requirements and how they were verified.

ID	Requirement	Verification
N1	Privacy: no image, video or landmark stream leaves the device	By construction: no upload path exists; the server serves static files only
N2	Footprint: the 260-sign model is a few megabytes	5.2 MB int8 ONNX (jepa-asl-b3)
N3	Speed: recognition keeps up with the camera on a laptop	Pose at camera rate; classifier tens of milliseconds per window
N4	Honesty: accuracy is reported on a signer never seen, through the deployed pipeline	The accuracy test (Section 3.3.4), at least four seeds per claim
N5	Calibration: a stated confidence means what it says	Cross-fitted temperature; expected calibration error 0.039
N6	Cost: free to use; no per-use server cost	Static hosting on Cloudflare
N7	Licensing: every dataset used within its terms; attribution where required	Table 3.1; CC BY credit in the application and documentation

3.3 Architecture

3.3.1 Run-time architecture

Figure 3.2 shows the application at run time. A webcam frame enters a pose worker, where MediaPipe extracts body and hand landmarks; these are canonicalised into 107 points and pushed into a 64-frame ring. Each window is scored by three recognisers:

- **260-sign set.** The deployed classifier, jepa-asl-b3 (int8 ONNX, 5.2 MB), runs on the window and on its mirror image; the more confident side answers, and a fitted temperature turns its output into a calibrated probability.
- **Full vocabulary.** A supervised encoder maps the window to a 256-dimensional embedding, compared by cosine similarity against a bank of 8,497 example rows that reaches 8,165 of the 8,188 words.
- **Shape match.** A model-free dynamic-time-warping matcher over hand-crafted kinematics runs beside them, with no learned weights.

The Learn, Practice and Dictionary surfaces and the Studio read from these recognisers; the sign-by-sign mode (Section 4.2.10) splits continuous signing into signs before the 260-sign classifier. The application is served as static files from Cloudflare at `studio.sanket.kadal.cc`; the server never sees a frame.

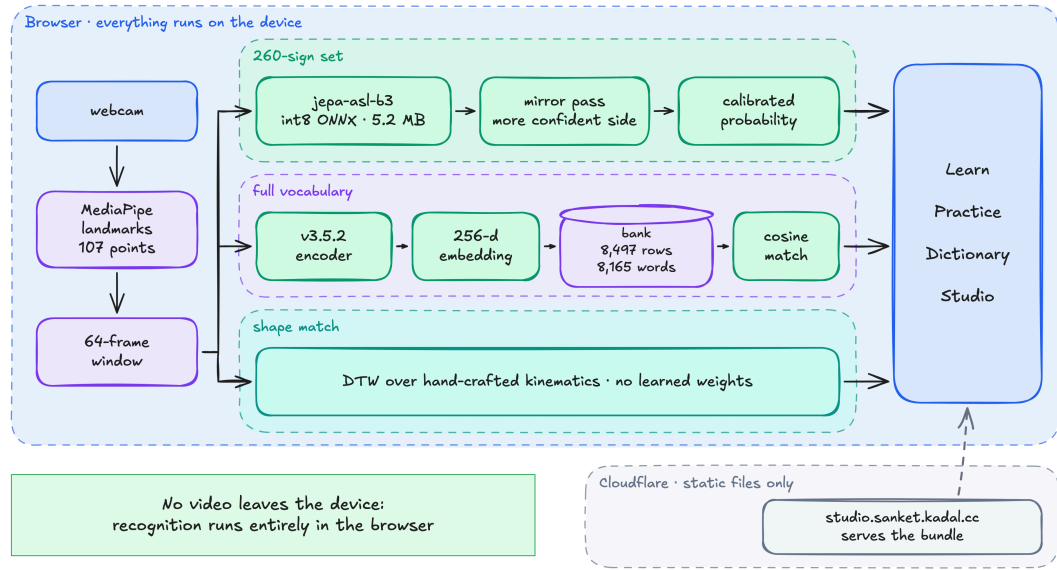


Figure 3.2. The system at run time. Everything inside the browser region runs on the user’s device; the only network traffic is the static download of the application and the model files.

Table 3.4. Run-time artifacts.

Artifact	Role	Size
jepa-asl-b3 classifier (int8 ONNX)	260-sign set: calibrated probabilities with the mirror pass	5.2 MB
Supervised encoder v3.5.2 (int8 ONNX)	Full vocabulary: 256-dimensional embedding	9.9 MB
Example bank	Full vocabulary: 8,497 rows, cosine similarity	8,165 of 8,188 words reachable
Shape-match references	DTW over kinematics	No learned weights

3.3.2 Training architecture

Figure 3.3 shows how a model is made. Corpora are extracted once into a canonical pack of 107 points $\times$ 64 frames. An articulator-token transformer is pretrained with the ISL-JEPA objective on the unlabelled pool and cut at layer 4, the depth beyond which its features stopped transferring. The cut encoder is taught 250 ASL signs, then fine-tuned on INCLUDE with infusion clips, synthetic signers, recover-and-resample augmentation and a phonological head. The fine-tunes of several recipes and seeds form an ensemble that is distilled into one student; the student is exported to int8 ONNX with the softmax inside the graph, calibrated, packaged as a bundle and promoted only through the accuracy test.

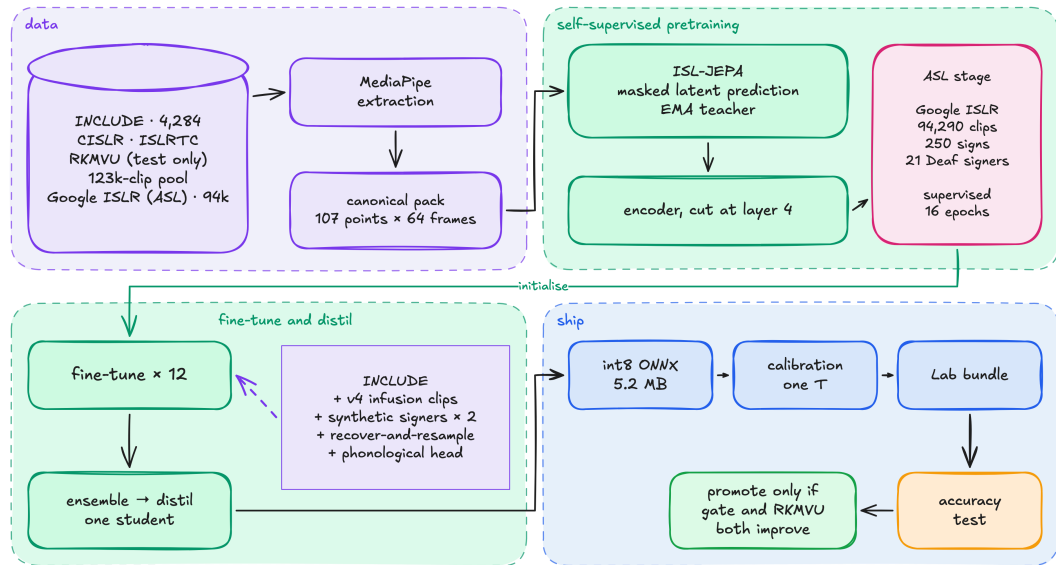


Figure 3.3. How a model is made: data, pretraining with the ASL stage, fine-tuning and distillation, and shipping. Each stage writes a versioned artifact that the next stage reads.

3.3.3 Transfer from American Sign Language

Figure 3.4 details the transfer stage added in the final round. The Google ISLR corpus is streamed and canonicalised with the same rules as the Indian corpora; an encoder initialised from ISL-JEPA learns its 250 signs with a supervised head, and the ISL fine-tune then starts from that encoder, optionally keeping the ASL head as an auxiliary task. Four recipes, four seeds each, were measured separately and as ensembles before twelve of the resulting models were distilled into the deployed student.

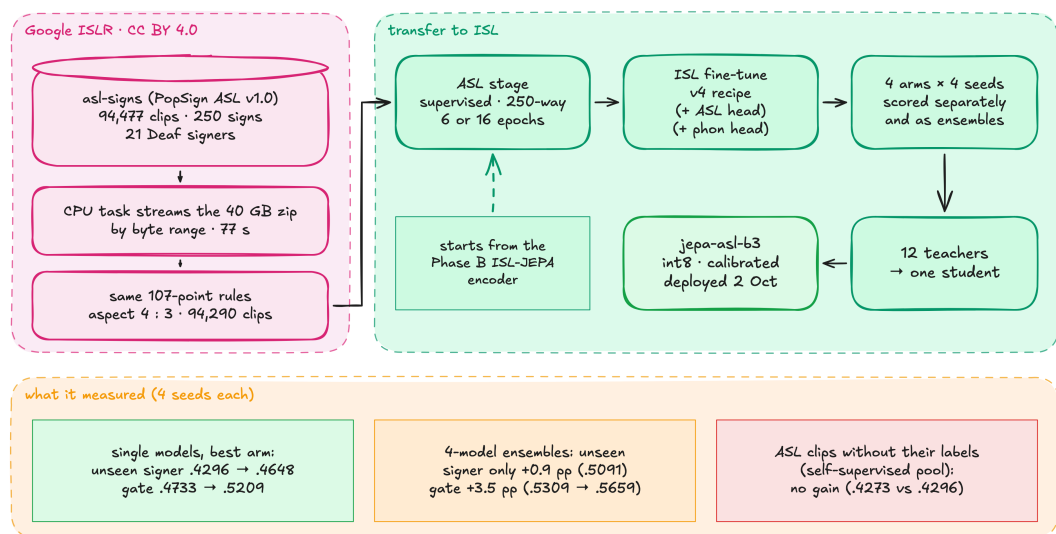


Figure 3.4. Transfer from American Sign Language: ingest and canonicalisation (left), the transfer recipes and distillation (right), and what was measured (bottom).

3.3.4 Evaluation architecture

Figure 3.5 shows the accuracy test. Two sets feed one number. RKMVU-220 is 220 dictionary clips from one signer who appears nowhere in training — *the stranger number*. The CISLR test role is 379 clips that were never trained on, but whose presenters also appear in CISLR training clips. Their clip-weighted top-1 accuracy over 599 clips is the accuracy test (*the gate* in the training ledgers); one clip is worth 0.17 percentage points. Four side checks are fixed in advance: at least four seeds per claim, a leak check against training rows, calibration cross-fitted across the two corpora, and in-app parity through the int8 graph with the mirror pass. The promotion rule is that a candidate must beat the deployed model on the accuracy test *and* on the unseen signer.

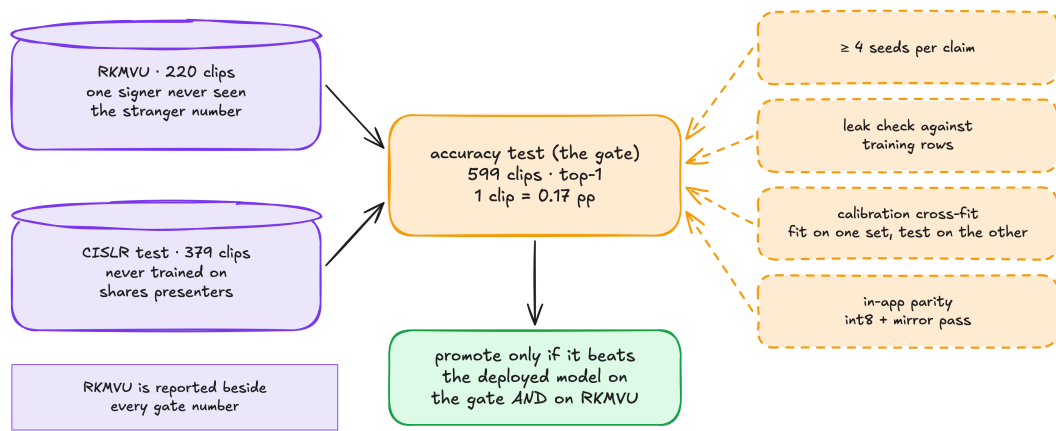


Figure 3.5. The accuracy test: two sets, one number, four side checks, and the promotion rule.

3.3.5 Data pipeline and review loop

Figure 3.6 shows the loop that turned dataset quality into accuracy. The Dataset tab shows every clip of four corpora with its video, its landmarks and a figure rendering. Marks — at clip, word or corpus-rule level, from an agent or a person — are compiled by one command into the contract the training packs are built from. Ambiguous cases go to a vision-model review of stick-figure packets; what it cannot settle is escalated to a blind review of the original video with hidden controls; and what remains goes to a person in the Review tab, whose decisions flow back into the marks.

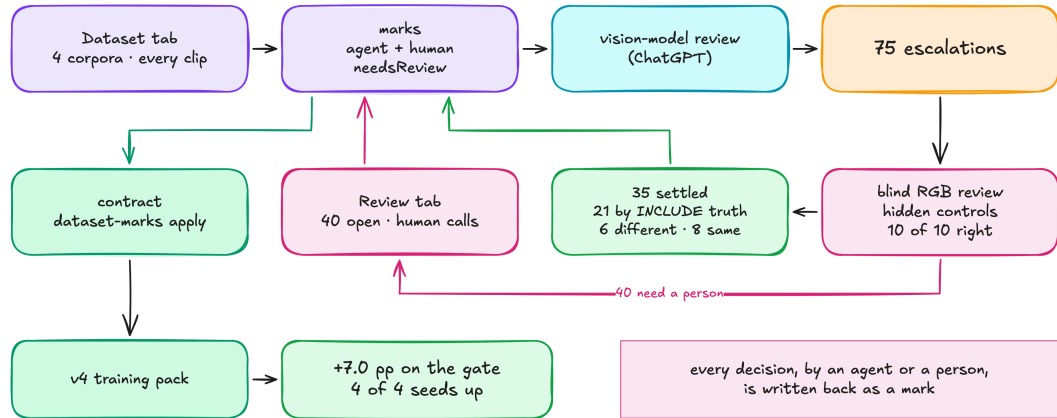


Figure 3.6. The dataset review loop, from marks to a rebuilt training pack. The arrow from the Review tab back to the marks closes the loop.

3.4 Implementation Strategy

Table 3.5 lists the technology used at each layer. The application is written in TypeScript and React and runs its recognisers in Web Workers with ONNX Runtime Web; training is in Python and PyTorch; models move between the two as ONNX files with the pre- and post-processing inside the graph, so the browser and the training harness share one definition of the model.

Table 3.5. Technology stack.

Layer	Technology	Used for
Application	TypeScript, React, Vite, TanStack Router	The studio single-page application and its surfaces
In-browser inference	ONNX Runtime Web (WebAssembly, SIMD, threads) [5]	int8 classifier and encoder, one worker per recogniser
Landmarks	MediaPipe pose and hands [3], [4]	107 canonical points per frame, on-device
Training	Python, PyTorch, NumPy	ISL-JEPA, ASL stage, fine-tuning, distillation
Export	ONNX (opset 17), int8 dynamic quantisation [47]	Browser-shippable graphs with the softmax inside
Compute	Modal A10G; Colab Pro A100; Beam; Apple M2 Pro (MPS)	Fine-tune and distillation gauntlet; ASL ingest; extraction
Hosting	Cloudflare Workers (static assets)	sanket.kadal.cc (documentation), studio.sanket.kadal.cc (application)

Layer	Technology	Used for
Evaluation	Python harnesses; Playwright-driven browser Eval bench	The accuracy test offline and in the browser
Documents	Typst; Excalidraw (diagrams drawn in code)	This report, the paper and the review documents

The engineering process had three rules. Every candidate change is an *arm* in a harness that runs the unmodified training scripts on any of the compute back-ends, with at least four seeds and an in-harness control. Every model is scored by the same accuracy test before and after export. And a model replaces the deployed one only through the promotion rule, so the deployed number can only move up on the unseen signer. Table 3.6 compares the time frame proposed at Review 0 with what was done.

Table 3.6. Time frame: proposed at Review 0 and actual.

Period	Planned (Review 0)	Done
9–30 Aug	Reference expansion, hygiene audit, harnesses frozen	Harness and canonical contract; cross-corpus gap measured (0.82 → 0.33); mirror pass
31 Aug–4 Sep	Encoder retrain on cleaned references	ISL-JEPA Phase B (123,340 clips); infusion, synthetic signers, rr; jepa-rr4 deployed
5–27 Sep	Calibration, signer study, Learn and Practice polish	Dataset review loop and v4 pack (+7.0 points); calibration; 12-teacher distillation (jepa-x12); Review 2
28 Sep–3 Oct	Evaluation freeze, report skeleton	Phonological head; sign-by-sign mode; ASL transfer (jepa-asl-b3 deployed 2 Oct); two negative levers closed
4–11 Oct	Final report, demo hardening	Final report, paper, presentation; final review on 11 Oct

3.5 Justification for the Proposed System

Each major choice was made against an alternative, and most were settled by measurement rather than by argument (Table 3.7).

Landmarks rather than pixels. Landmarks remove appearance — skin, clothing, background, lighting — which a small training set would otherwise memorise, and they are the only representation the privacy requirement allows to persist. They also make the browser feasible: a landmark window is $64 \times 107 \times 2$ numbers, against millions of pixels.

The browser rather than a server. A server-side recogniser would need the camera stream, a connection and a running cost per user. In the browser the cost per use is zero and privacy is a property of the construction.

MediaPipe rather than a stronger tracker. RTMW is a stronger whole-body estimator, and on INCLUDE’s own signers it lifted accuracy by 13 points; on the accuracy test, under the same trainer and five seeds, MediaPipe scored 0.257 and RTMW 0.238 (Section 5.1.9). The browser must run MediaPipe anyway, so the stronger tracker bought nothing where it matters.

Self-supervision rather than supervised training alone. ISL has thousands of words but only one or two labelled clips for most of them, and few signers; unlabelled ISL signing is plentiful. ISL-JEPA used 123,340 unlabelled clips and gave the first browser model that tied the deployed supervised model at half its size (Section 5.1.2).

Transfer from ASL rather than more ISL data. The constraint of the final round was to improve the unseen-signer number without new signers. Google’s ISLR corpus brings 21 Deaf signers under a licence that allows a shipped model, with the same landmark format. Its labels, not its clips, carry the gain: as unlabelled data it did nothing (Section 5.1.4).

One distilled model rather than an ensemble. Ensembles of four to sixteen models scored higher than any single model, but a browser can run one; distillation keeps most of an ensemble’s accuracy in one 5.2 MB model, and running two or three models in the browser was measured not to help the unseen signer (Section 5.1.9).

A calibrated probability rather than a raw score. Practice credits a sign above a confidence bar; an uncalibrated model refused learners who had signed correctly. One cross-fitted temperature fixes the meaning of the number without changing which word ranks first.

Table 3.7. Design choices, the alternative each was weighed against, and the evidence that settled it.

Choice	Alternative	Evidence
Landmarks	RGB video	Privacy requirement; browser footprint; appearance is what small data memorises
Browser inference	Server inference	Zero cost per use; nothing uploaded; works offline once loaded
MediaPipe	RTMW	Accuracy test 0.257 vs 0.238, five seeds; RTMW invents legs on waist-framed video
ISL-JEPA pretraining	Supervised only	123,340 unlabelled clips; tied the supervised model at half the size

Choice	Alternative	Evidence
Supervised ASL stage	ASL as unlabelled data	Unseen signer 0.4648 vs 0.4273 (four seeds each)
One distilled model	Browser ensemble	Pre-registered 2–3 model ensembles: 0.5091 at best vs 0.5136 alone
Cross-fitted temperature	Raw scores	Calibration error 0.160 → 0.039; Practice credits correct signs it used to refuse

CHAPTER 4

Implementation

4.1 Execution and Step-by-Step Analysis

The work was executed in nine steps. Each step produced a versioned artifact — a pack, an encoder, a set of fine-tunes, a bundle — and each was measured before the next one built on it.

Step 1 — Landmark extraction and the canonical pack. Every corpus was converted to MediaPipe landmarks through the browser’s own contract and stored as canonical windows of 64 frames $\times$ 107 points. The extraction jobs ran on cloud CPU task queues and are resumable: ISLRTC’s download quota stopped the first pass at 4,804 of 11,993 dictionary entries, and the job resumes from where it stopped.

Step 2 — Self-supervised pretraining (ISL-JEPA). An eight-layer articulator-token transformer (Section 4.2.2) was pretrained without labels. The pilot ran on 4,276 INCLUDE clips; Phase B ran 100 epochs on 123,340 clips in 11.1 hours on an A100. Its last-layer target collapsed late in training while the features of layers up to four kept transferring, so every shipped encoder is cut at layer four. Phase C, with an all-layer teacher, was a wash at full budget and is used only as a source of ensemble teachers.

Step 3 — The fine-tuning recipe. The cut encoder was fine-tuned on INCLUDE with an attentive head. Three additions were each measured over several seeds before being kept: gloss-matched infusion clips from CISLR and ISLRTC, geometric synthetic signers, and recover-and-resample augmentation (Section 4.2.3). A harness (the *gauntlet*) runs the unmodified training scripts as arms with seeds on any of the compute back-ends.

Step 4 — The dataset review. The Dataset tab, the marks system and the vision and blind reviews (Figure 3.6) produced a reviewed contract from which the v4 training pack was rebuilt. The same recipe on the v4 pack gained 7.0 points on the accuracy test, on four of four seeds.

Step 5 — Ensemble, distillation, export and calibration. Fine-tunes of several seeds and encoders were averaged into a teacher and distilled into one student (Section 4.2.6); the student was exported to int8 ONNX with the softmax inside the graph, checked for parity against PyTorch, calibrated with a cross-fitted temperature (Section 4.2.8) and packaged as a bundle with its model card. *jepa-x12*, the Review 2 model, came from this step.

Step 6 — The phonological head. Following PhonSSM [43], the linear head was replaced by four branches for handshape, location, movement and orientation (Section 4.2.5). Over four seeds it lifted the unseen signer from 0.4284 to 0.4579 and the accuracy test from 0.4817 to 0.5067.

Step 7 — Transfer from ASL. The Google ISLR corpus was ingested and canonicalised (Section 3.2), and a supervised ASL stage was added before the ISL fine-tune (Section 4.2.4). Four recipes were run with four seeds each on a Colab A100, scored singly and as ensembles, and twelve of the models were distilled into three candidate students; one of them, *jepa-asl-b3*, passed the promotion rule and was deployed on 2 October 2026.

Step 8 — Closing levers. Two further levers were run to completion and measured as negative: a browser ensemble of two or three models, and ASL clips used as unlabelled pretraining data (Section 5.1.9).

Step 9 — The application. Around the recogniser, the application provides Learn, Practice, a Dictionary look-up, the Studio, a sign-by-sign mode (Section 4.2.10), and developer surfaces — the Model Lab, the Eval bench, the Dataset and Review tabs — that keep the models and the data honest (Appendix C).

4.2 Core Algorithms and Configurations

4.2.1 Landmark canonicalisation

MediaPipe returns 33 pose points and 21 points for each hand, in coordinates normalised to the image. These are mapped to 107 canonical points per frame: 23 body points in the COCO layout (the legs are zeroed), 42 hand points in the body frame, and 42 hand points in a hand-local frame (each hand’s points relative to its wrist, divided by the hand’s span). Body and body-frame hand points are translated to the mid-shoulder origin and divided by the shoulder width. The same function runs in the browser, in the extraction jobs and in evaluation, and unit tests hold the three to one another.

Algorithm 4.1: Landmark canonicalisation (browser, extraction and evaluation)

Input: frames of pose landmarks P_t (33 points) and hand landmarks H_t (21 points each), camera aspect W/H

Output: one canonical window, identical in the browser, in training and in evaluation

- 1: Correct the aspect: $x \leftarrow x \cdot W/H$, so that horizontal and vertical distances share one unit.
 - 2: Shoulder width $s_t \leftarrow \|P_t^{\text{ls}} - P_t^{\text{rs}}\|$; a degenerate width ($< 10^{-3}$) is replaced by the window’s median width.
 - 3: Anchor $a_t \leftarrow (P_t^{\text{ls}} + P_t^{\text{rs}})/2$.
 - 4: Body: map the pose points to 23 COCO slots (the lower body stays zero); $b_t \leftarrow (P_t - a_t)/s_t$.
 - 5: Body-frame hands: $(H_t - a_t)/s_t$; hand-local: $(H_t - H_t^{\text{wrist}})/\max_j \|H_t^j - H_t^{\text{wrist}}\|$.
 - 6: A frame without a pose stays all zero; an absent hand zeroes both of its blocks.
 - 7: A window is $T = 64$ frames (live: the last 64 frames; training: the clip resampled to 64), so the input is $64 \times 107 \times 2$.
-

4.2.2 ISL-JEPA self-supervised pretraining

Each frame contributes three articulator tokens — body, left hand, right hand — giving 192 tokens for a 64-frame window, encoded by an eight-layer transformer of width 256. A context encoder sees a masked or corrupted view, and a predictor must produce the latents that an exponential-moving-average (EMA) target encoder assigns to the clean clip. Three masks are used: M1 hides one articulator stream, M2 hides temporal spans, and M3 feeds a deliberately corrupted view (mirrored, tempo-warped, morphed, jittered, a hand dropped) whose latents must match those of the clean clip. Targets are layer-normalised and centred; a hinge on the per-dimension standard deviation of the latents guards against collapse. The loss on the masked set $\mathcal{M}$ is

$$\begin{aligned} \mathcal{L}_{\text{JEPA}} = & \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \text{SmoothL1}(\hat{z}_i, \text{sg}(\bar{z}_i - c)) \\ & + \lambda_v \mathcal{L}_{\text{view}} + \frac{\lambda_\sigma}{D} \sum_{d=1}^D \max(0, \gamma - \sigma_d(h)) \end{aligned} \quad (4.1)$$

where $\hat{z}_i$ is the predictor output, $\bar{z}_i$ the normalised target latent, c the running centre, “sg” the stop-gradient, $\mathcal{L}_{\text{view}}$ the M3 term and $\sigma_d(h)$ the standard deviation of dimension d of the context latents, with $\gamma = 0.5$ and $\lambda_\sigma = 0.5$.

Algorithm 4.2: ISL-JEPA pretraining step

Input: a batch of canonical windows X ; context encoder f_θ ; predictor g_φ ; EMA target encoder f_ξ ; centre c ; momentum τ

Output: updated θ, φ, ξ, c ; after training the encoder is cut at layer 4

- 1: Tokenise each frame into three articulator tokens: 192 tokens per window.
 - 2: Sample a mask m : M1 hides one articulator stream, or M2 hides temporal spans.
 - 3: Build a nuisance view $\tilde{X}$ (mirror, tempo warp, morph, jitter, hand dropout) for M3.
 - 4: Targets without gradient: $\bar{Z} \leftarrow \text{LN}(f_\xi(X)) - c$.
 - 5: Context $H \leftarrow f_\theta(X \odot \neg m)$; prediction $\hat{Z}_m \leftarrow g_\varphi(H, m)$.
 - 6: $\mathcal{L} \leftarrow \text{SmoothL1}(\hat{Z}_m, \bar{Z}_m) + \lambda_v \text{SmoothL1}(q(f_\theta(\tilde{X})), \bar{Z}) + \lambda_\sigma \text{mean}_d \max(0, \gamma - \sigma_d)$.
 - 7: One optimiser step on θ, φ ; EMA update $\xi \leftarrow \tau\xi + (1 - \tau)\theta$; centre $c \leftarrow 0.99c + 0.01 \cdot \text{mean}(\bar{Z})$.
-

4.2.3 Fine-tuning recipe

The recipe, named `all+synth:full@2+rr` in the ledgers, fine-tunes the cut encoder with an attentive pooling head on the 260 classes, with label smoothing 0.1, AdamW (learning rate 10^{-4} for the encoder and 10^{-3} for the head, weight decay 0.05) and a cosine schedule over 40 epochs.

- **Gloss infusion.** Clips from CISLR and ISLRTC whose gloss matches one of the 260 classes are added to the training set; the CISLR test role is never admitted.
- **Geometric synthetic signers** (Figure 4.1). Each real clip’s landmarks, including MediaPipe’s depth coordinate, are lifted to 3-D, rotated in yaw and pitch, retargeted to different limb proportions and projected back — two variants per clip. Alone they add almost nothing; stacked on the real infusion clips they lifted five-seed cross-validation from 0.3018 to 0.3927.
- **Recover-and-resample (rr).** With probability 0.5 per batch, up to 25% of the frames are dropped from each end of the clip and the rest is stretched back to 64 frames, so that the model sees signs that start late or end early — the way dictionary clips differ from INCLUDE takes.

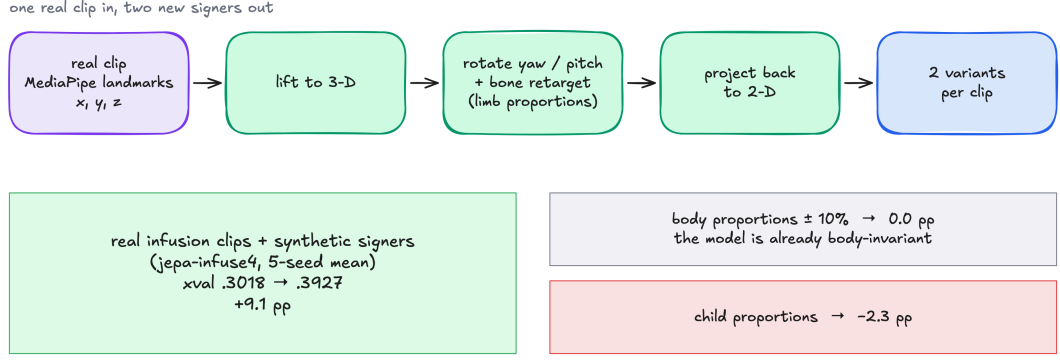


Figure 4.1. Synthetic signers from real clips: lift to 3-D, rotate and retarget, project back to 2-D.

4.2.4 The ASL transfer stage

The ASL stage trains the cut encoder, with its own attentive head, to classify the 250 ISLR signs before the ISL fine-tune begins. The two ISLR participants with the fewest clips are held out to report ASL accuracy on unseen ASL signers. The stage runs for six or sixteen epochs (batch 128, learning rate 2×10^{-4} with a one-cycle schedule). In the `asl` variant the ASL head is kept during the ISL fine-tune as an auxiliary loss on one ASL batch per ISL batch, weighted 0.5. Augmentation runs on the GPU, with the same transforms as the ISL batches.

Algorithm 4.3: Supervised ASL stage before the ISL fine-tune

Input: cut encoder f ; ISLR windows A with labels y^A (250 signs); ISL training set D ; epochs E_A

Output: an ISL classifier whose encoder has learned 250 ASL signs from 19 signers first

- 1: Attach an ASL head h_A (attentive pooling + linear, 250 outputs).
 - 2: For E_A epochs over A : minimise $\text{CE}_{0.1}(y^A, h_A(f(a)))$ on augmented ASL batches.
 - 3: Attach the ISL head h_I (linear or phonological, 260 outputs).
 - 4: For each ISL batch $B \subset D$: $\mathcal{L} \leftarrow \text{CE}_{0.1}(y, h_I(f(B)))$;
 - 5: `asl` variant: draw an ASL batch B_A , add $0.5 \cdot \text{CE}_{0.1}(y^A, h_A(f(B_A)))$;
 - 6: one optimiser step on f , h_I (and h_A).
-

4.2.5 The phonological head

The phonological head (Figure 4.2) replaces the linear classifier. Four branches — handshape, location, movement and orientation — each map the encoder’s tokens to a 32-dimensional code and score it against their own learned prototypes (30, 15, 10 and

8). The branch codes and prototype scores are fused with the pooled clip embedding, and the sign is predicted by cosine similarity to one learned prototype per sign:

$$\text{logit}_k = s \cdot \cos(e, w_k), \quad e = \text{Fuse}(o, r_{\text{hs}}, r_{\text{loc}}, r_{\text{mov}}, r_{\text{ori}}) \quad (4.2)$$

where o is the pooled embedding, r the branch outputs, w_k the prototype of sign k and s a learned scale. Two auxiliary terms keep the branches apart (orthogonality, weight 0.1) and the prototypes diverse (weight 0.01).

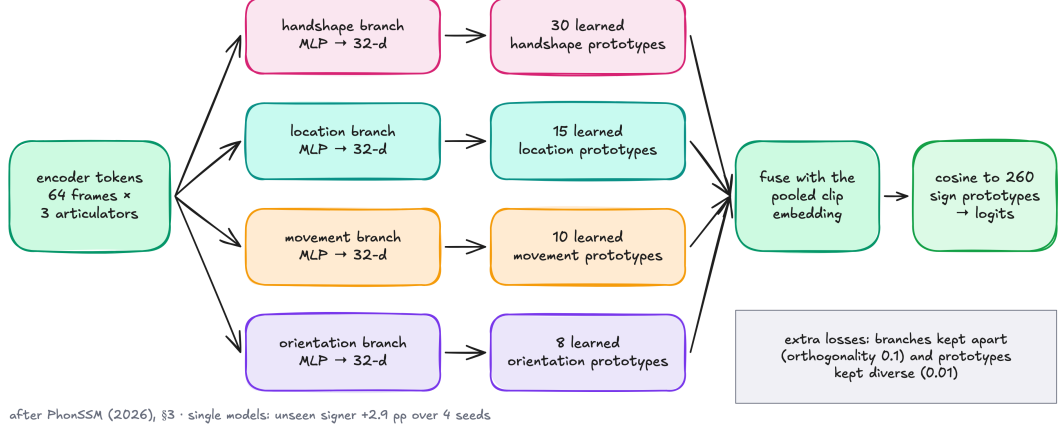


Figure 4.2. The phonological head: four branches for the four parameters of a sign, each with its own prototypes, fused and compared with one prototype per sign.

4.2.6 Ensemble distillation

Averaging fine-tunes scores higher than any single one, but a browser can run one model. The teacher is the mean softmax $\bar{p}$ of N fine-tunes, tempered at temperature T by renormalising its powers; the student is trained on

$$\mathcal{L}_{\text{KD}} = (1 - \alpha) \cdot \text{CE}_\varepsilon(y, \text{softmax}(z_s)) + \alpha T^2 \text{KL}(\tilde{p}_T \parallel \text{softmax}(z_s/T)) \quad (4.3)$$

where the tempered teacher target is

$$\tilde{p}_T(k) = \frac{\bar{p}(k)^{1/T}}{\sum_j \bar{p}(j)^{1/T}}, \quad (4.4)$$

with label smoothing $\varepsilon = 0.1$, $\alpha = 0.7$, $T = 2$ and 40 epochs [44]. The T^2 factor keeps the soft-target gradients on the scale of the hard-label term. The deployed student is distilled from twelve ASL-pretrained fine-tunes (Figure 4.3).

Algorithm 4.4: Ensemble distillation

Input: fine-tuned teachers $M_1 \dots M_N$; student S ; $\alpha = 0.7$, $T = 2, 40$ epochs

Output: one student, exported to int8 ONNX

- 1: For each epoch and augmented mini-batch B :
 - 2: teacher $\bar{p} \leftarrow \frac{1}{N} \sum_n \text{softmax}(M_n(B))$ (no gradient); tempered target $\tilde{p}_T$;
 - 3: student logits $z_s \leftarrow S(B)$;
 - 4: $\mathcal{L} \leftarrow (1 - \alpha)\text{CE}_{0.1}(y, z_s) + \alpha T^2 \text{KL}(\tilde{p}_T \parallel \text{softmax}(z_s/T))$ (+ the phonological terms if S has that head);
 - 5: one optimiser step on S .
-

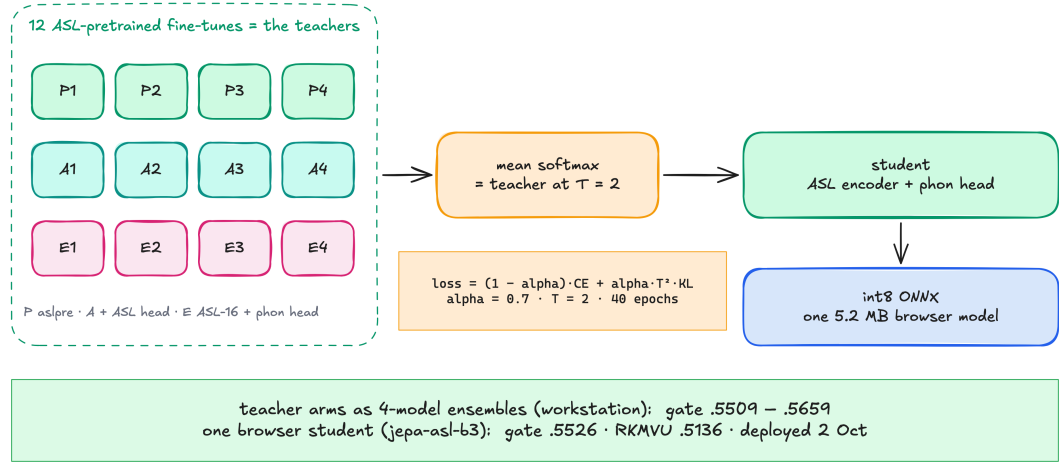


Figure 4.3. Twelve ASL-pretrained fine-tunes teach one browser-sized student.

4.2.7 Export, quantisation and parity

The student is exported to ONNX (opset 17) with velocity and bone channels, normalisation and the softmax inside the graph, so the browser feeds positions and reads probabilities. Dynamic int8 quantisation [47] reduces it to 5.2 MB. A parity check compares the PyTorch and ONNX outputs on real windows (maximum difference 2.3×10^{-6} for jepa-asl-b3), and the int8 graph is scored on the accuracy test like any other candidate.

4.2.8 Temperature calibration with cross-fitting

One scalar temperature per bundle turns the classifier's output into a probability that means what it says [46]. It acts on exactly the posterior the user sees, after the mirror pass:

$$q_i = \frac{p_i^{1/T}}{\sum_j p_j^{1/T}}, \quad T^* = \arg \min_{T \in \mathcal{G}} -\frac{1}{n} \sum_{k=1}^n \log q_{y_k}^{(k)}(T) \quad (4.5)$$

where $\mathcal{G}$ is a grid of 400 log-spaced values in $[0.2, 3.0]$. The only labelled cross-signer data are the accuracy-test clips, so the fit is cross-checked: the temperature fitted on RKMVU is scored on CISLR and vice versa, by the expected calibration error over ten confidence bins,

$$\text{ECE} = \sum_{b=1}^{10} \frac{|B_b|}{n} |\text{acc}(B_b) - \text{conf}(B_b)|, \quad (4.6)$$

and the pooled temperature ships only if each one-corpus fit lowers the error on the other corpus. A temperature cannot change which class ranks first, so it cannot move accuracy on any benchmark.

Algorithm 4.5: Temperature calibration with cross-fit

Input: classifier M ; accuracy-test clips by corpus, R (RKMVU) and C (CISLR); grid $\mathcal{G}$

Output: a temperature T^* , or a documented refusal

- 1: For every clip: $p \leftarrow M(x)$, $p' \leftarrow M(\text{mirror}(x))$; keep the side with the larger maximum.
 - 2: $T_R \leftarrow \arg \min_{T \in \mathcal{G}} \text{NLL}_R(T)$; $T_C \leftarrow \arg \min_{T \in \mathcal{G}} \text{NLL}_C(T)$.
 - 3: Transfer holds if $\text{ECE}_C(T_R) < \text{ECE}_C(1)$ and $\text{ECE}_R(T_C) < \text{ECE}_R(1)$.
 - 4: If it holds: $T^* \leftarrow \arg \min_{T \in \mathcal{G}} \text{NLL}_{RUC}(T)$, written into the bundle; otherwise ship uncalibrated and report the disagreement.
-

4.2.9 Mirror test-time augmentation

A left-handed signer produces the mirror image of a right-handed performance; at Review 1 such a signer scored 0.12 against 0.33. Every window is therefore scored twice — as captured and mirrored — and the more confident side answers. Because the canonical frame is symmetric about the body midline, mirroring is exact: negate the horizontal coordinate and swap every left–right pair, including the two hands.

Algorithm 4.6: Mirror test-time augmentation

Input: canonical window x ; classifier M with temperature T^*

Output: a word and a calibrated confidence

- 1: $x' \leftarrow x$ with the horizontal coordinate negated and left–right pairs swapped.
 - 2: $p \leftarrow M(x)$, $p' \leftarrow M(x')$; if $\max_k p'_k > \max_k p_k$ then $\hat{p} \leftarrow p'$ else $\hat{p} \leftarrow p$.
 - 3: $q \leftarrow \text{temper}(\hat{p}, T^*)$ (Eq. (4.5)); answer $\arg \max_k q_k$ with confidence $\max_k q_k$.
 - 4: Practice credits the target word when $q_{\text{target}} \geq 0.5$.
-

4.2.10 Sign by sign

Sliding windows over continuous signing mix the end of one sign with the start of the next. The sign-by-sign mode instead splits the landmark stream where the signer’s hands come to rest (Figure 4.4) and classifies each span once. The segmenter is geometric, with no learned weights.

Algorithm 4.7: Sign-by-sign segmentation

Input: a stream of canonical frames with pose and hand presence

Output: one word per sign, with its calibrated confidence

- 1: A frame is *at rest* if no present wrist is above the hip line (a fallback height when the hips are not visible).
 - 2: A span opens at the first frame that is not at rest, and closes after at least 4 consecutive rest frames.
 - 3: Drop spans shorter than 10 frames; pad each span by 2 frames on both sides.
 - 4: Resample the span to 64 frames and classify it once with the 260-sign classifier and the mirror pass.
-

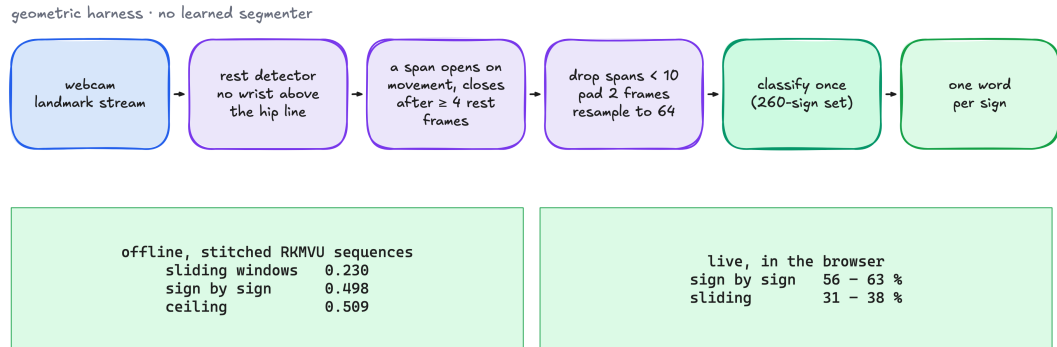


Figure 4.4. Sign by sign: the stream is split where the hands rest, and each sign is classified once.

4.2.11 Full-vocabulary retrieval and shape match

The full vocabulary is reached by similarity: a supervised encoder trained with an additive-margin loss [22] maps a window to a 256-dimensional unit vector, and the answer is the word of the most similar example row, emitted only above a confidence bar. Because 833 words are signed in more than one way, the bank keeps one variant per clip rather than one averaged prototype per word, which gained 10.7 points. The shape match aligns hand-crafted kinematics — hand shape, orientation, position relative to the body and motion — against reference performances by dwell-weighted dynamic time warping [24]; it needs no training and serves as a fallback and as a diagnostic instrument (Section 5.1.7).

4.3 Challenges Faced and Mitigation

Table 4.1 lists the practical hurdles met during implementation and how each was resolved. Three of them shaped the method itself. **Collapse at depth:** the Phase B encoder’s last layers over-specialised to the pretext task while its early layers kept transferring; cutting the encoder at layer four turned a failed run into the project’s best encoder. **Label noise:** ISL dictionary corpora contain lessons rather than words (64% of ISLRTC files run longer than eight seconds), sense-conflated labels and words signed in several ways; the dataset review loop converted this into the largest single gain. **Compute availability:** the main GPU provider disabled the workspace during the night before Review 2, and later providers had small credit limits or idle timeouts; the harness was kept provider-independent, so the unmodified scripts moved between Modal, Beam, Colab and the local GPU.

Table 4.1. Challenges faced during implementation and how each was mitigated.

Challenge	Effect	Mitigation
Few labelled clips per word; few signers	Supervised models memorise signers (0.82 $\rightarrow$ 0.33 cross-corpus)	Self-supervised pretraining on 123k clips; infusion; synthetic signers; ASL transfer
Label noise in dictionary corpora	Lessons filed as words; sense conflation; variants averaged together	Dataset marks, vision and blind reviews, a reviewed contract (+7.0 points); one bank row per clip (+10.7)
Left-handed signers	0.12 instead of 0.33	Exact mirror pass at test time; mirror augmentation in training
Late collapse of the JEPA target	Last-layer features stopped transferring	Encoder cut at layer 4; history checkpoints to find the peak
Seed noise on a 599-clip test	One clip is 0.17 points	At least four seeds per claim; seed-level tables; promotion rule on both numbers

Challenge	Effect	Mitigation
Hand-depth normalisation bug in the browser	Every window of the full vocabulary embedded to one point	Browser anchored x and y only, as training does; a parity test across the three code paths
GPU workspace disabled before Review 2	A 16-teacher distillation could not finish	Weights pulled from the volume; scripts made device-independent (CUDA, MPS)
Small cloud credits; Colab idle timeout	Runs at risk of being cut off	CPU-only tasks for ingest (US \$0.02); a running notebook cell as keep-alive; artefacts pulled off the VM promptly
40 GB ASL archive, 17 GB of free disk	Could not download and unpack locally	HTTP byte-range streaming of the archive's members on a CPU task (77 s)
Tunnel body limit and a broken transfer	Uploads over 100 MB refused; one truncated file	Split uploads; byte-size checks after every transfer
Asset size limit of the host (25 MiB per file)	The shape-match pack (25.4 MiB) blocked deployment	Re-quantised to 8 bits in the deploy step only
Phonological gains that did not survive distillation	Students tied the deployed model	Multi-teacher distillation from ASL-pretrained teachers; the promotion rule kept only a student that beat both numbers

CHAPTER 5

Results and Discussion

Every number in this chapter is measured; targets are labelled as targets. Accuracy is top-1 unless stated. The unseen-signer number (RKMVU) is reported beside every accuracy-test number, and a claim about a training recipe rests on at least four seeds.

5.1 Results

5.1.1 The accuracy test

The accuracy test (the *gate* in the training ledgers) is clip-weighted top-1 over 599 clips that no model ever trained on: the RKMVU set, 220 clips from one signer whom no training corpus contains, and the CISLR test role, 379 dictionary clips (Section 3.3.4). One clip is worth 0.17 points of the accuracy test and 0.45 points of RKMVU, which is why recipes are compared over seeds. RKMVU is the honest stranger number: the CISLR test clips share presenters with CISLR clips that are admitted to training through infusion, so part of any CISLR gain is familiarity with the presenter. The promotion rule therefore asks a candidate to beat the deployed model on **both** the accuracy test and RKMVU.

5.1.2 Model lineage, Review 0 to the final review

Table 5.1 lists every model that reached the accuracy test, and Figure 5.1 places them on the review timeline. Review 0 reported in-corpus accuracy only (INCLUDE top-1 0.958 on its own signer split; CISLR open-vocabulary top-1 0.2429 and top-20 0.4105 over 8,165 words). Review 1 measured the cross-corpus gap — 0.82 in-corpus against 0.33 cross-corpus — and the mirror fix for a left-dominant signer (0.12 to 0.33). From Review 1 onwards every model is scored on the same 599 clips.

Table 5.1. Model lineage on the accuracy test (fp32 unless stated). Bold rows are the models deployed at each review; the last row is the model deployed for the final review.

Model	Date	Gate	RKMVU	CISLR	Notes
Supervised v3.5.2 (Review 1 model)	Aug	.1469 ^b	.3318	.0396	9.90 MB; ^b in the browser
jepa-ship4 (Phase B JEPA, cut at layer 4)	1 Sep	—	.3273	—	4.57 MB

Model	Date	Gate	RKMVU	CISLR	Notes
jepa-infuse4 (+ infusion + synthetic signers)	2 Sep	.4124	.4318	—	5-seed cross-val .3018 → .3927
jepa-distil4 (4 teachers → student)	3 Sep	.4424	.4545	.4354	
jepa-rr4 (+ recover-and-resample)	3 Sep	.4591	.4818	.4459	deployed 4–26 Sep
jepa-v4 (reviewed v4 pack)	26 Sep	.5359	.4818	.5673	in-app .5412
jepa-x12 (12 teachers → student)	26 Sep	.5509	.5000	.5805	Review 2 model; in-app .5546
16-model ensemble (workstation only)	27 Sep	.5693	.5136	.6016	Review 2 ceiling
Phonological-head fine-tunes (4-seed mean)	29 Sep	.5067	.4579	—	vs .4817 / .4284, linear head
34-model ensemble (workstation only)	2 Oct	.5793	.5318	.6069	strongest configuration measured
jepa-asl-b3 (12 ASL-pretrained teachers → student)	2 Oct	.5526	.5136 (int8 .5091)	.5752	deployed 2 Oct; in the browser .5559; 5.23 MB

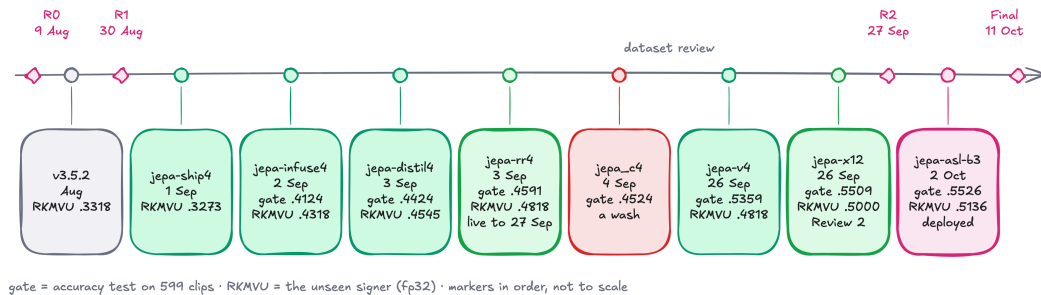


Figure 5.1. Model lineage from Review 0 to the final review, on the review timeline. Each model is labelled with its accuracy test and unseen-signer (RKMVU) numbers.

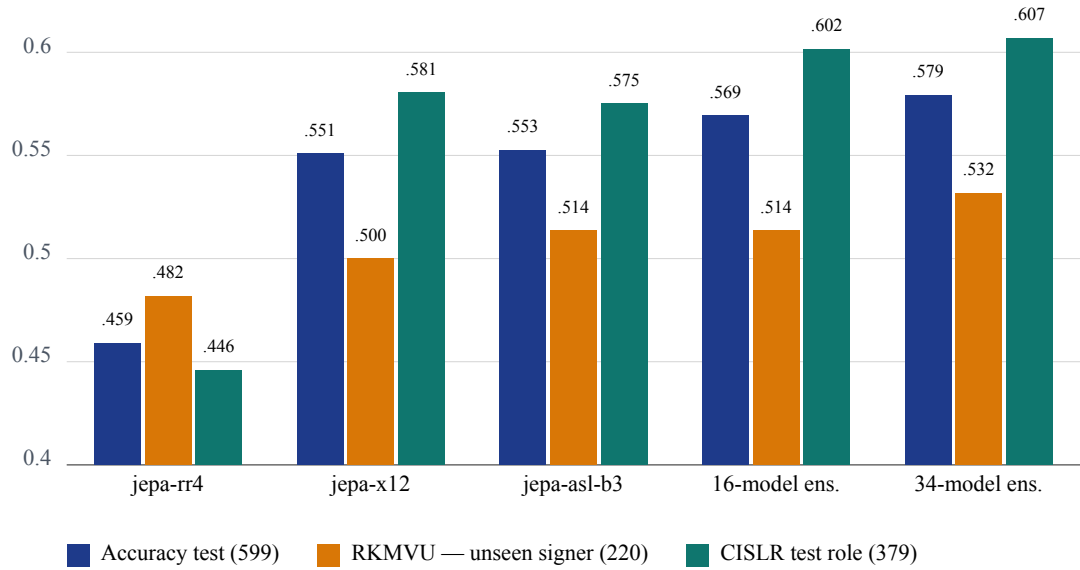


Figure 5.2. Accuracy test, unseen signer and CISLR test role (fp32) from jepa-rr4, deployed until Review 2, to jepa-asl-b3 and the two largest ensembles. The y-axis starts at 0.40. The ensembles run on a workstation and are not browser-shippable; they mark the ceiling that a distilled student aims at.

5.1.3 Seed-level evidence for the dataset contract

The v4 pack is the training pack rebuilt from the reviewed dataset contract. To separate its effect from seed noise, the recipe was trained four times on the previous pack and four times on the v4 pack, in the same run (Table 5.2). Every seed improved, and the mean rose by 7.0 points; the same comparison between four-model ensembles gives .4574 → .5376 on the accuracy test and .4727 → .4909 on RKMVU.

Table 5.2. Accuracy test per seed: the same recipe on the previous pack (control) and on the reviewed v4 pack.

Pack	Seed 1	Seed 2	Seed 3	Seed 4	Mean
Control	.4057	.4174	.4124	.4107	.4116
v4 pack	.4825	.4942	.4691	.4808	.4817

5.1.4 Transfer from ASL

Nine recipes were fine-tuned with four seeds each on the v4 pack, all starting from the Phase B encoder (Table 5.3). The control is the v4 recipe; asl keeps an ASL head as an auxiliary loss during the ISL fine-tune; aslpre runs the six-epoch supervised ASL stage before it; e16 runs a sixteen-epoch ASL stage; phon replaces the linear head with the phonological head.

Table 5.3. Transfer from ASL: four-seed means of single fine-tuned models (fp32). Green marks the best value in each column.

Recipe	Seeds	RKMVU	CISLR	Gate
Control (v4 recipe)	4	.4296	.4987	.4733
asl — ASL auxiliary loss	4	.4375	.5257	.4934
aslpre — ASL stage (6 epochs) first	4	.4523	.5449	.5109
asl+aslpre	4	.4648	.5284	.5050
e16 — ASL stage (16 epochs) first	4	.4239	.5495	.5034
e16+phon — and the phonological head	4	.4568	.5581	.5209
asl+aslpre+phon	4	.4511	.5561	.5175
both3 — three-epoch variant	4	.4409	.5383	.5025
e16asl	3	.4212	.5347	.4930

Every ASL recipe beat the control on the accuracy test, by 2.0 to 4.8 points. On the unseen signer the two best recipes gained 3.5 (asl+aslpre) and 2.7 (e16+phon) points; the sixteen-epoch stage alone did not help the stranger at all (.4239), so a longer ASL stage moves the model towards the presenters of CISLR rather than towards new signers unless the phonological head is added.

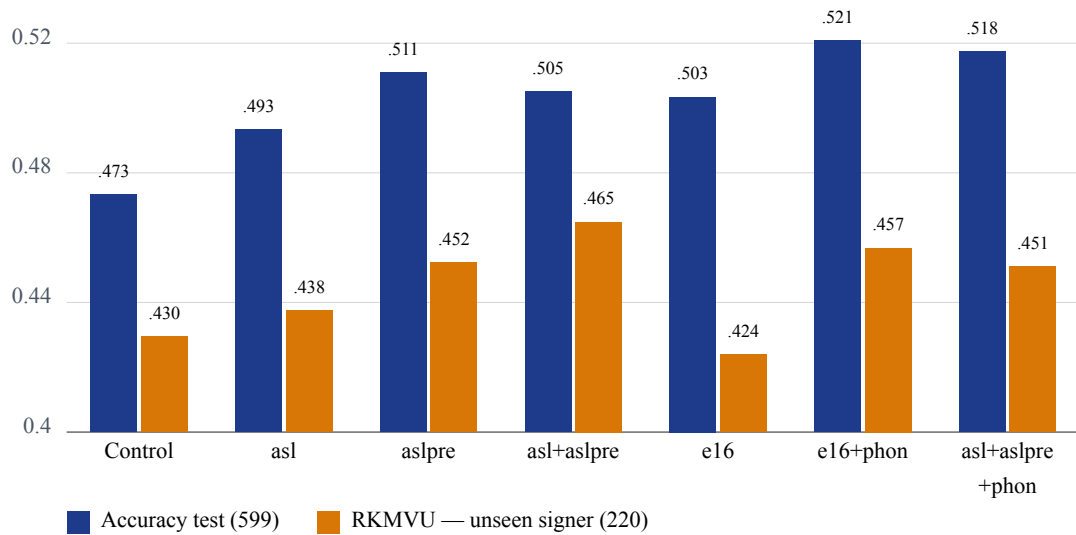


Figure 5.3. Transfer from ASL, four-seed means of single models. The y-axis starts at 0.40.

Single models are not what the browser runs; a student distilled from an ensemble is. Table 5.4 therefore scores each recipe's four seeds as an ensemble.

Table 5.4. Transfer from ASL, scored as ensembles of each recipe’s four seeds (fp32).

Ensemble	RKMVU	CISLR	Gate
Control	.5000	.5488	.5309
asl	.4773	.5646	.5326
aslpre	.5000	.6042	.5659
asl+aslpre	.5091	.5752	.5509
both3	.5091	.5699	.5476
e16+phon	.4818	.5910	.5509
e16	.4727	.5699	.5342
All 34 fine-tunes	.5318	.6069	.5793

Two observations decided the distillation. First, the recipes that were best as single models were not the best ensembles: the phonological recipes, strongest alone, gained least from averaging, so their seeds make similar mistakes. Second, the 34-model ensemble reached .5318 on the unseen signer, three points above any four-model ensemble, which says that the ceiling is set by the diversity of the teachers. Three students were distilled (Table 5.5): A from the aslpre and asl+aslpre seeds, B from those and the e16+phon seeds (twelve teachers), and C from twelve phonological teachers.

Table 5.5. Students distilled from ASL-pretrained teachers (fp32), against the deployed jepa-x12. The promotion rule asks for a gain on both the accuracy test and RKMVU.

Student	Gate	RKMVU	CISLR	Rule
jepa-x12 (deployed)	.5509	.5000	.5805	—
A — 8 teachers	.5426	.4864	.5752	fails
B, seed 1 — 12 teachers	.5543	.4955	.5884	fails (RKMVU)
B, seed 2	.5526	.4955	.5858	fails (RKMVU)
B, seed 3 — jepa-asl-b3	.5526	.5136	.5752	passes
C — 12 phonological teachers	.5376	.4727	.5752	fails

5.1.5 The deployed model: jepa-asl-b3

Student B, seed 3 was exported, quantised and calibrated as jepa-asl-b3 and deployed on 2 October 2026 (Table 5.6). It is the first deployed model to pass 0.50 on the unseen signer in the int8 graph the browser runs, and the only one of the five candidate students to beat jepa-x12 on both numbers. The margin is small — three RKMVU clips — and is stated as such in the application’s model card.

Table 5.6. The deployed model against its predecessor: int8 graphs as shipped, the application’s mirror rule, and calibration.

Measurement	jepa-x12	jepa-asl-b3
Accuracy test, int8	.5509	.5526
RKMVU, int8	.4955	.5091
CISLR test role, int8	.5831	.5778
RKMVU, int8 with the application’s mirror rule	.5000	.5136
Accuracy test in the browser (Eval bench, 6 Oct)	.5492	.5559
RKMVU in the browser: top-1 / top-5	.4864 / .6818	.5136 / .6955
Inference per window (headless browser, CPU)	32.8 ms	41.3 ms
Temperature T^* (cross-fit T_R / T_C)	0.788	0.804 (0.731 / 0.843)
Expected calibration error, before $\rightarrow$ after	.216 $\rightarrow$.056	.160 $\rightarrow$.039
Area under the risk–coverage curve (lower is better)	.208	.177
Bundle size	4.57 MB	5.23 MB

Calibration transferred in both directions — the temperature fitted on RKMVU lowered the error on CISLR and the other way round — so the pooled temperature was shipped. After calibration the model’s stated confidence is within four points of its accuracy in every populated bin, and it ranks its own answers better than jepa-x12 did (a lower area under the risk–coverage curve), which matters for the Practice credit and for the confidence bar of the Studio.

5.1.6 Sign by sign

The sign-by-sign mode (Algorithm 4.7) was measured offline on four-sign sequences built by joining unseen-signer clips end to end through the browser’s tensor path, decoded with the deployed int8 model and the mirror pass (Table 5.7), and live in Chrome on a sixteen-clip unseen-signer playlist (four runs with jepa-x12, one with jepa-asl-b3).

Table 5.7. Word accuracy on continuous four-sign sequences of unseen-signer clips (offline), and on a sixteen-clip playlist played into the live application (four runs).

Decoder	Word accuracy
Sliding windows with a persistence rule (previous behaviour)	.230
Sliding windows, best setting tuned on the same data	.376
Sign by sign: split at rest, classify each span once	.498
True sign boundaries (ceiling)	.509
Live playlist, jepa-x12: live panel	31–38%

Decoder	Word accuracy
Live playlist, jepa-x12: sign by sign	56–63%
Live playlist, jepa-asl-b3 (6 Oct): live panel / sign by sign	43.8% / 62.5%

The segmenter found exactly four signs in 98% of the sequences (mean 4.01) and came within 1.1 points of perfect boundaries. The gain is a decoding gain — the classifier is unchanged — and it depends on the signer returning the hands to rest between signs, as signers do in citation form.

5.1.7 Shape match and the full vocabulary

The model-free shape match was measured both as a recogniser and as a helper to the model (Figure 5.4). Across the 260 classes it names 25.0% of the clips; inside the model’s top-3 shortlist it names 34.7%, but the model alone reaches 45.7% on the same bench, so it never helps as a re-ranker. As a label checker for pseudo-labelling or a reinforcement-learning reward, it is less precise than the model’s own confidence (55.3% against 68.0% at equal coverage). It remains a transparent fallback and a diagnostic instrument. On the full vocabulary, keeping one bank row per clip instead of one averaged prototype per word lifted retrieval by 10.7 points, because 833 words are signed in more than one way.

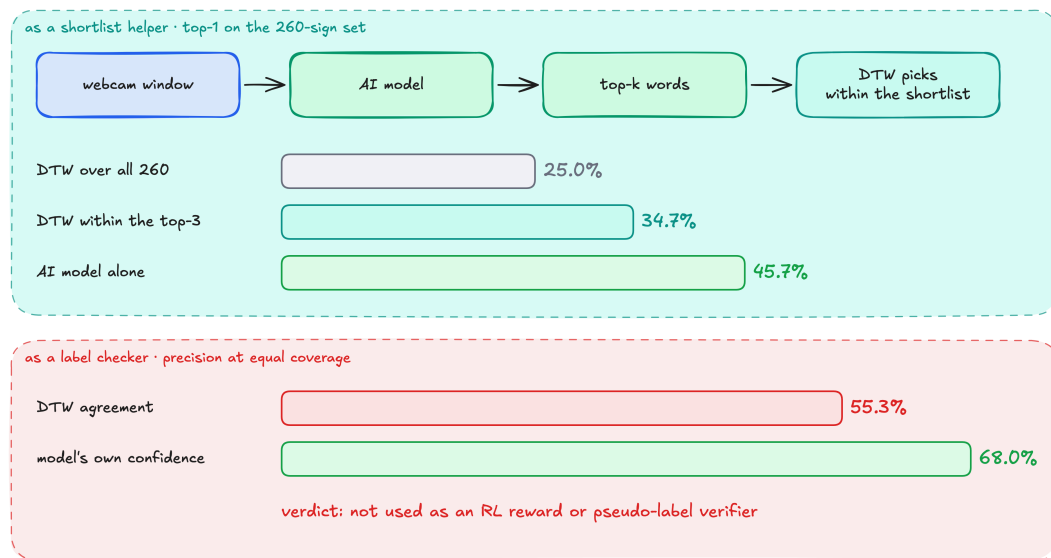


Figure 5.4. Shape match as a helper, measured: as a re-ranker inside the model’s shortlist (25.0% → 34.7%, against 45.7% for the model alone) and as a label checker (55.3% against 68.0% precision at equal coverage).

5.1.8 Data findings

The dataset tools measured problems that no model change could fix (Table 5.8).

Table 5.8. Data findings from the Dataset tab and the review loop.

Corpus / process	Finding
ISLRTC	64% of files exceed 8 s — lessons rather than words; median clip 14.7 s; 17% silently truncated at 36 s; 767 phantom words created from file names
INCLUDE	76% of frames are idle (hands at rest)
Vocabulary	833 words are signed more than one way; a variant per clip rebuilt the bank (+10.7 points)
Vision review	75 escalations; 35 settled by blind RGB review (21 by INCLUDE ground truth, 6 confirmed different, 8 rejected as the same sign)
Google ISLR (ASL)	94,477 clips ingested in 77 s on a CPU task; 94,290 kept after canonicalisation; 21 signers (19 for training), 250 signs

5.1.9 Negative results

The directions in Table 5.9 did not improve the model. They are reported because each one closed a direction or changed the plan, and because the promotion rule is only meaningful if failures are counted.

Table 5.9. Negative results, measured.

Direction	Measurement	Consequence
RTMW whole-body landmarks instead of MediaPipe [20]	5 seeds on the 599 clips: MediaPipe .257 vs RTMW .238; RTMW’s +13 points on INCLUDE’s own split do not transfer	Stay on MediaPipe, which the browser must run anyway
Photometric pre-processing for missing hands	Null or negative; hands go missing because they move fast (97% of misses) or leave the frame (68%)	No image pre-pass
Shape match as an RL reward	55.3% vs 68.0% precision for the model’s own confidence	Not used as a reward
Phase C objective (all-layer teacher)	A wash at full budget; Phase C students lose RKMVU	Phase C only as ensemble teachers
Phonological head through distillation	12-teacher student .5426 / .4864; 4-teacher student (two seeds) .5376 / .5000; both below jepa-x12	Head kept only inside ASL-pretrained teachers
Continued self-supervision with ASL clips	4 seeds: pool + ASL .4687 gate / .4273 RKMVU; pool only .4816 / .4250; Phase B .4733 / .4296	ASL helps as labels, not as unlabelled data
Browser ensemble of 2–3 models (pre-registered)	Best .5693 / .5182 vs jepa-asl-b3 alone .5576 / .5136 (mirror rule), at 2–3× the compute	Not shipped; a single student stays
Canonical 3-D body frame; margin losses;	Each null or negative over seeds	Closed

Direction	Measurement	Consequence
v5 pack; 16-teacher student		

5.2 Discussion

5.2.1 Interpretation of the results

The stranger number rose from 0.33 to 0.51. At Review 1 a supervised model scored 0.3318 on the unseen signer; the deployed model scores 0.5136 with the mirror rule the application uses, in a browser model of 5.2 MB. The accuracy test rose from 0.1469 to 0.5526.

The gains came from four sources, each measured. Self-supervised pretraining with the cut at layer four and the fine-tuning recipe took the stranger number from .33 to .48 (jepa-rr4). The reviewed dataset contract added 7.0 points on the accuracy test over four seeds. Distillation carried most of an ensemble into one model: student B scores about four points above the mean of its teachers' recipes. Transfer from ASL added 2–5 points on the accuracy test and up to 3.5 points on the unseen signer for single models, and gave the student that first passed both numbers.

CISLR is easier to move than RKMVU. Every lever moved CISLR more than the unseen signer — from jepa-rr4 to the 34-model ensemble CISLR gained 16.1 points and RKMVU 5.0. The reason is structural: the CISLR test clips share presenters with infusion clips. This is why the promotion rule is written on RKMVU as well, and why jepa-asl-b3, which loses 0.5 points of CISLR against jepa-x12, was still the right model to ship.

Teacher diversity sets the ceiling. The best single recipes were not the best ensembles, and the 34-model ensemble was three points above any four-model one on the stranger. The students that learned from the most diverse teachers (B) were the ones closest to the ensemble.

The data were the cheapest lever. The largest single gain on the accuracy test in this project came from correcting labels, not from a model change, and the largest gain in retrieval came from recognising that one word can be signed in several ways.

5.2.2 Comparison with the objectives and the Review 1 targets

Table 5.10 states how each objective of Section 1.4 was met, and Table 5.11 the targets set at Review 1.

Table 5.10. The objectives of Section 1.4 and how each was met.

Objective	How it was met
1. Analyse the state of ISL recognition	Chapter 2; the cross-corpus audit (0.82 → 0.33) and the dataset review (Table 5.8)
2. On-device recogniser	MediaPipe, the int8 classifier and retrieval run in the browser tab; nothing leaves the device (Figure 3.2)
3. Cover the vocabulary a learner meets	A calibrated 260-sign classifier and retrieval over 8,188 words; shape match as a fallback
4. Close the cross-signer gap	ISL-JEPA, infusion, synthetic signers, the data contract, ASL transfer and distillation: RKMVU .3318 → .5136
5. Evaluate honestly	One pre-stated 599-clip test through the deployed path; four seeds per claim; negative results reported (Table 5.9)
6. Ship a useful product	Learn, Practice, Studio and sign by sign live at studio.sanket.kadal.cc (Appendix C)

Table 5.11. Targets set at Review 1 and their status at the final review.

Metric	Target	Final status	Verdict
Accuracy test (599 clips)	≥ 0.50	.5526 (int8, deployed)	Met
Unseen signer (RKMVU-220)	≥ 0.50	.5091 int8; .5136 with mirror rule	Met
Mirrored signer	parity	mirror pass in every path	Met
Model size (260-sign set)	≤ 4.57 MB	5.23 MB int8	Exceeded by 0.66 MB; accepted

5.2.3 Limitations

The unseen-signer test is one signer, so the stranger number is a measurement of one person’s signing and its confidence interval is wide (about ± 6.6 points at 95%). The CISLR test role shares presenters with training clips. The full-vocabulary path has not been retrained on the JEPA encoder and is weak on live video. Continuous signing is supported only when the signer pauses between signs. The ASL data are licensed for non-commercial research; the deployed model credits the source under its licence.

5.3 Practical Implications

For users. A learner can practise ISL with a camera and get feedback on each attempt without creating an account, and with no video leaving the device. The application states what to expect: its model card reports 51% top-1 and 69% top-5 on a signer it has never seen, and the calibrated confidence is honest enough to drive the Practice credit.

For deployment. The whole recogniser is a 5.2 MB static file served from a content network; a browser downloads it once and runs it offline. There is no inference server to pay for or secure, so the marginal cost of a user is zero, which is what an accessibility tool for a large population with uneven connectivity needs.

For research practice. The project shows how much a within-corpus number can mislead (0.958 in-corpus against 0.33 for a stranger), and that a small, fixed, pre-stated test, computed through the deployed code and reported over seeds with its failures, is enough to steer a long sequence of decisions.

For other low-resource sign languages. The ingredients are not specific to ISL: self-supervision on unlabelled landmarks, transfer from a sign language with more signers, a reviewed data contract and distillation into a browser model all apply to any language with a small labelled corpus.

CHAPTER 6

Conclusion and Future Scope

6.1 Conclusion

This case study set out to build an Indian Sign Language recogniser that runs entirely in the user’s browser, answers over a large vocabulary, and is measured and improved on signers it has never seen. All three were achieved.

The system. Sanket runs MediaPipe landmark extraction, a 5.2 MB int8 classifier for a 260-sign set, retrieval over an 8,188-word vocabulary and a model-free shape match inside a web page, with no image, video or landmark stream leaving the device. Around the recogniser it offers Learn, Practice, a recognition Studio and a sign-by-sign mode, live at `studio.sanket.kadal.cc`.

The measurement. One accuracy test of 599 clips — an unseen signer and a dictionary test role never trained on — was fixed at Review 1 and computed through the same numeric path as the deployed application. Every recipe was compared over at least four seeds, and a model was promoted only if it beat the deployed one on both the accuracy test and the unseen signer.

The result. Between Review 1 and the final review, accuracy on the unseen signer rose from 0.3318 to 0.5136 and on the accuracy test from 0.1469 to 0.5526, in a browser model of the same order of size. Both targets set at Review 1 (0.50) are met. The deployed model’s confidence is calibrated (expected calibration error 0.039), so the application can state what it knows.

The contributions. (i) ISL-JEPA, a latent-predictive self-supervised encoder over articulator tokens, with the finding that an encoder whose deep target collapses still transfers when cut at layer four; (ii) a fine-tuning recipe — gloss infusion, geometric synthetic signers, recover-and-resample — whose parts were each measured over seeds; (iii) a reviewed dataset contract that gained seven points on its own; (iv) supervised transfer from American Sign Language, which raised every ISL recipe and produced the first student to pass both numbers; (v) ensemble distillation into one browser-sized model with a cross-fitted calibration; and (vi) a geometric sign-by-sign decoder that doubles word accuracy on continuous signing.

What did not work is recorded as carefully as what did (Table 5.9): stronger whole-body landmarks that do not transfer across corpora, image pre-processing for missing

hands, shape match as a reward, unlabelled ASL as pretraining data, and browser ensembles that cost more than they gain.

The project's broader finding is that, for a low-resource sign language, accuracy on a stranger is set less by the classifier than by the data around it — the number of signers the model has seen, in any sign language, and the correctness of the labels it learns from.

6.2 Future Scope

Four next steps follow from the results (Figure 6.1). Each will be kept only if it beats the deployed model on the unseen signer.

1. **More strangers.** The unseen-signer test is one person. A signer study — the same 220 words from several Deaf signers, recorded with consent through the application itself — would turn the stranger number into a distribution and narrow its confidence interval.
2. **Per-user adaptation from Practice.** In Practice the target word is known, so every attempt a learner chooses to contribute is a free, correct label. Adapting a small part of the model on the device, from the user's own attempts, could close the gap for that user without sending anything to a server.
3. **A landmark bridge.** Whole-body trackers such as RTMW see hands that MediaPipe loses, but the browser must run MediaPipe. A learned mapping from one landmark set to the other would let training use the better tracker while the browser keeps its own.
4. **From sign by sign to continuous signing.** The sign-by-sign decoder depends on a pause between signs. A learned boundary model, trained on clips joined end to end, and a grammar-aware assembly of glosses into sentences are the steps towards translation.

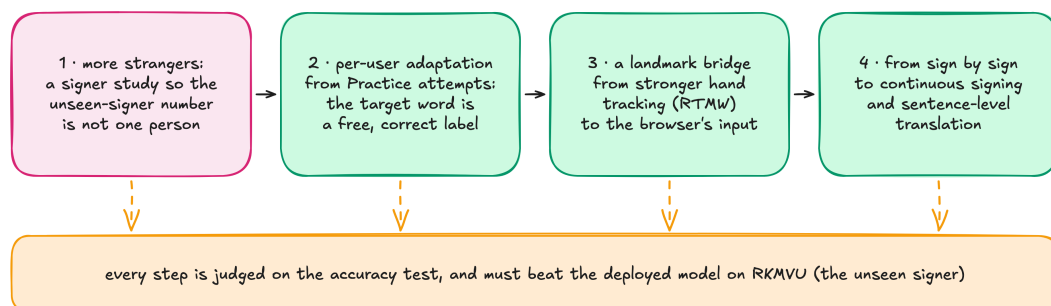


Figure 6.1. Future scope: four next steps, from a signer study to continuous signing; each is kept only if it beats the deployed model on the unseen signer.

Beyond the model, two research outputs are planned for the next semester: the manuscript prepared from this work (Appendix B), on on-device ISL recognition with an unseen-signer evaluation, and a cross-corpus audit of ISL datasets built on the data findings of Table 5.8.

References

- [1] Ministry of Statistics and Programme Implementation, “Disabled Persons in India: A Statistical Profile 2016,” Government of India, 2016. [Online]. Available: https://www.mospi.gov.in/sites/default/files/publication_reports/Disabled_persons_in_India_2016.pdf
- [2] Indian Sign Language Research and Training Centre, “Indian Sign Language Interpreters Directory.” [Online]. Available: <https://esic.gov.in/attachments/circularfile/4650b5eaafbec5cc169937302ce1205c.pdf>
- [3] C. Lugaresi *et al.*, “MediaPipe: A Framework for Building Perception Pipelines,” *arXiv:1906.08172*, 2019, [Online]. Available: <https://arxiv.org/abs/1906.08172>
- [4] F. Zhang *et al.*, “MediaPipe Hands: On-device Real-time Hand Tracking,” in *CVPR Workshop on Computer Vision for Augmented and Virtual Reality*, 2020. [Online]. Available: <https://arxiv.org/abs/2006.10214>
- [5] Microsoft, “ONNX Runtime Web — WebAssembly execution provider.” [Online]. Available: <https://onnxruntime.ai/docs/tutorials/web/>
- [6] D. Li, C. Rodriguez Opazo, X. Yu, and H. Li, “Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods Comparison,” in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 1448–1458. doi: [10.1109/WACV45572.2020.9093512](https://doi.org/10.1109/WACV45572.2020.9093512).
- [7] H. R. Vaezi Joze and O. Koller, “MS-ASL: A Large-Scale Data Set and Benchmark for Understanding American Sign Language,” in *British Machine Vision Conference (BMVC)*, 2019.
- [8] A. Sridhar, R. G. Ganesan, P. Kumar, and M. Khapra, “INCLUDE: A Large Scale Dataset for Indian Sign Language Recognition,” in *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*, 2020, pp. 1366–1375. doi: [10.1145/3394171.3413528](https://doi.org/10.1145/3394171.3413528).
- [9] A. Joshi *et al.*, “CISLR: Corpus for Indian Sign Language Recognition,” in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022, pp. 10357–10366. doi: [10.18653/v1/2022.emnlp-main.707](https://doi.org/10.18653/v1/2022.emnlp-main.707).
- [10] A. Joshi *et al.*, “iSign: A Benchmark for Indian Sign Language Processing,” in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024, pp. 10827–10844. doi: [10.18653/v1/2024.findings-acl.643](https://doi.org/10.18653/v1/2024.findings-acl.643).
- [11] United Nations, “Goal 10: Reduce inequality within and among countries.” [Online]. Available: <https://sdgs.un.org/goals/goal10>
- [12] United Nations, “Goal 4: Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all.” [Online]. Available: <https://sdgs.un.org/goals/goal4>
- [13] A. Chow *et al.*, “Google – Isolated Sign Language Recognition.” [Online]. Available: <https://www.kaggle.com/competitions/asl-signs>

- [14] T. Starner *et al.*, “PopSign ASL v1.0: An Isolated American Sign Language Dataset Collected via Smartphones,” in *NeurIPS (Datasets and Benchmarks Track)*, 2023, pp. 184–196. doi: [10.52202/075280-0011](https://doi.org/10.52202/075280-0011).
- [15] J. Carreira and A. Zisserman, “Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4724–4733. doi: [10.1109/CVPR.2017.502](https://doi.org/10.1109/CVPR.2017.502).
- [16] S. Yan, Y. Xiong, and D. Lin, “Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018, doi: [10.1609/aaai.v32i1.12328](https://doi.org/10.1609/aaai.v32i1.12328).
- [17] S. Jiang, B. Sun, L. Wang, Y. Bai, K. Li, and Y. Fu, “Skeleton Aware Multi-modal Sign Language Recognition,” in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021, pp. 3408–3418. doi: [10.1109/CVPRW53098.2021.00380](https://doi.org/10.1109/CVPRW53098.2021.00380).
- [18] M. Boháček and M. Hruží, “Sign Pose-based Transformer for Word-level Sign Language Recognition,” in *2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, 2022, pp. 182–191. doi: [10.1109/WACVW54805.2022.00024](https://doi.org/10.1109/WACVW54805.2022.00024).
- [19] P. Selvaraj, G. NC, P. Kumar, and M. Khapra, “OpenHands: Making Sign Language Recognition Accessible with Pose-based Pretrained Models across Languages,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022, pp. 2114–2133. doi: [10.18653/v1/2022.acl-long.150](https://doi.org/10.18653/v1/2022.acl-long.150).
- [20] T. Jiang, X. Xie, and Y. Li, “RTMW: Real-Time Multi-Person 2D and 3D Whole-body Pose Estimation,” *arXiv:2407.08634*, 2024, [Online]. Available: <https://arxiv.org/abs/2407.08634>
- [21] J. Snell, K. Swersky, and R. Zemel, “Prototypical Networks for Few-shot Learning,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.
- [22] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “ArcFace: Additive Angular Margin Loss for Deep Face Recognition,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4685–4694. doi: [10.1109/CVPR.2019.00482](https://doi.org/10.1109/CVPR.2019.00482).
- [23] Y. Huang *et al.*, “CurricularFace: Adaptive Curriculum Learning Loss for Deep Face Recognition,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 5900–5909. doi: [10.1109/CVPR42600.2020.00594](https://doi.org/10.1109/CVPR42600.2020.00594).
- [24] H. Sakoe and S. Chiba, “Dynamic Programming Algorithm Optimization for Spoken Word Recognition,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978, doi: [10.1109/TASSP.1978.1163055](https://doi.org/10.1109/TASSP.1978.1163055).
- [25] A. Wang, “An Industrial-Strength Audio Search Algorithm,” in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR 2003)*, 2003. doi: [10.5281/zenodo.1416340](https://doi.org/10.5281/zenodo.1416340).
- [26] M. Assran *et al.*, “Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture,” in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 15619–15629. doi: [10.1109/CVPR52729.2023.01499](https://doi.org/10.1109/CVPR52729.2023.01499).
- [27] A. Bardes *et al.*, “Revisiting Feature Prediction for Learning Visual Representations from Video,” *Transactions on Machine Learning Research (TMLR)*, 2024, [Online]. Available: <https://openreview.net/forum?id=QaCCuDfBk2>

- [28] M. Assran *et al.*, “V-JEPA 2: Self-Supervised Video Models Enable Understanding, Prediction and Planning,” *arXiv:2506.09985*, 2025, [Online]. Available: <https://arxiv.org/abs/2506.09985>
- [29] L. Mur-Labadia *et al.*, “V-JEPA 2.1: Unlocking Dense Features in Video Self-Supervised Learning,” *arXiv:2603.14482*, 2026, [Online]. Available: <https://arxiv.org/abs/2603.14482>
- [30] M. Abdelfattah and A. Alahi, “S-JEPA: A Joint Embedding Predictive Architecture for Skeletal Action Recognition,” in *Computer Vision – ECCV 2024, Lecture Notes in Computer Science*, vol. 15090, Springer, 2024, pp. 367–384. doi: 10.1007/978-3-031-73411-3_21.
- [31] R. Xu, L. Huang, M. Wang, J. Hu, and W. Deng, “Skeleton2vec: A Self-supervised Learning Framework with Contextualized Target Representations for Skeleton Sequence,” *arXiv:2401.00921*, 2024, [Online]. Available: <https://arxiv.org/abs/2401.00921>
- [32] J. Zhang, L. Lin, S. Yang, and J. Liu, “Self-Supervised Skeleton-Based Action Representation Learning: A Benchmark and Beyond,” *International Journal of Computer Vision*, vol. 134, no. 1, 2026, doi: 10.1007/s11263-025-02644-8.
- [33] H. Hu, W. Zhao, W. Zhou, and H. Li, “SignBERT+: Hand-Model-Aware Self-Supervised Pre-Training for Sign Language Understanding,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 9, pp. 11221–11239, 2023, doi: 10.1109/TPAMI.2023.3269220.
- [34] S. Gueuwou, X. Du, G. Shakhnarovich, K. Livescu, and A. H. Liu, “SHuBERT: Self-Supervised Sign Language Representation Learning via Multi-Stream Cluster Prediction,” in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 28792–28810. doi: 10.18653/v1/2025.acl-long.1397.
- [35] K. Xie, Z. Cai, and K. Stefanov, “SignMAE: Segmentation-Driven Self-supervised Learning for Sign Language Recognition,” in *Pattern Recognition (ICPR 2026), Lecture Notes in Computer Science*, vol. 16821, Springer, 2026, pp. 48–62. doi: 10.1007/978-3-032-31438-3_4.
- [36] A. Basso Madjoukeng, J. Fink, P. Poitier, E. B. Kenmogne, and B. Frenay, “SSL-SLR: Self-Supervised Representation Learning for Sign Language Recognition,” *arXiv:2509.05188*, 2025, [Online]. Available: <https://arxiv.org/abs/2509.05188>
- [37] Z. Li, W. Zhou, W. Zhao, K. Wu, H. Hu, and H. Li, “Uni-Sign: Toward Unified Sign Language Understanding at Scale,” in *International Conference on Learning Representations (ICLR)*, 2025.
- [38] G. Tanzer and B. Zhang, “YouTube-SL-25: A Large-Scale, Open-Domain Multilingual Sign Language Parallel Corpus,” in *International Conference on Learning Representations (ICLR)*, 2025.
- [39] Y. C. Bilge, N. Ikizler-Cinbis, and R. G. Cinbis, “Cross-lingual few-shot sign language recognition,” *Pattern Recognition*, vol. 151, p. 110374, 2024, doi: 10.1016/j.patcog.2024.110374.
- [40] R. Holmes *et al.*, “From Scarcity to Understanding: Transfer Learning for the Extremely Low Resource Irish Sign Language,” in *ICCV Workshops*, 2023, pp. 2000–2009. doi: 10.1109/ICCVW60793.2023.00215.

- [41] A. A. Kindiroglu, O. Kara, O. Ozdemir, and L. Akarun, “Transfer Learning for Cross-dataset Isolated Sign Language Recognition in Under-Resourced Datasets,” in *IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, 2024, pp. 1–8. doi: [10.1109/FG59268.2024.10582029](https://doi.org/10.1109/FG59268.2024.10582029).
- [42] I. Ovodov, P. Surovtsev, K. Kvanchiani, A. Kapitanov, and A. Nagaev, “Logos as a Well-Tempered Pre-train for Sign Language Recognition,” in *EMNLP*, 2025, pp. 24351–24364. doi: [10.18653/v1/2025.emnlp-main.1238](https://doi.org/10.18653/v1/2025.emnlp-main.1238).
- [43] B. Cheng, A. Jin, and J. Zhang, “State Space Models are Effective Sign Language Learners: Exploiting Phonological Compositionality for Vocabulary-Scale Recognition,” *arXiv:2604.08761*, 2026, [Online]. Available: <https://arxiv.org/abs/2604.08761>
- [44] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” *arXiv:1503.02531 (NIPS 2014 Deep Learning Workshop)*, 2015, [Online]. Available: <https://arxiv.org/abs/1503.02531>
- [45] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826. doi: [10.1109/CVPR.2016.308](https://doi.org/10.1109/CVPR.2016.308).
- [46] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, “On Calibration of Modern Neural Networks,” in *Proceedings of the 34th International Conference on Machine Learning (ICML), PMLR vol. 70*, 2017, pp. 1321–1330.
- [47] B. Jacob *et al.*, “Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2704–2713. doi: [10.1109/CVPR.2018.00286](https://doi.org/10.1109/CVPR.2018.00286).
- [48] MSLR Workshop Organisers, “MSLR 2026: 2nd Multimodal Sign Language Recognition Workshop — signer-independent sub-task.” [Online]. Available: <https://m-slrt.github.io/MSLR2026/>
- [49] R. Zuo and B. Mak, “Improving Continuous Sign Language Recognition with Consistency Constraints and Signer Removal,” *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 20, no. 6, pp. 1–25, 2024, doi: [10.1145/3640815](https://doi.org/10.1145/3640815).

APPENDIX A

Source Code

The listings below are excerpts of the code that produces the numbers in this report, copied from the repositories without change except for elisions marked “...”. The complete source is listed in Appendix D.

A.1 Canonical window built in the browser

The function that turns MediaPipe landmarks into the model’s input tensor (apps/isl-spa/src/engine/classifier.ts). It reproduces the training packer exactly; unit tests drive it with synthetic frames.

```
for (let f = 0; f < frames; f++) {
  const poseOk = windowMask[f * groupCount + table.poseGroup] > 0;
  if (!poseOk) continue; // whole frame stays zero, like a pose-less training
  frame
  const base = f * featureDim;
  const scale = sw[f] > 1e-3 ? sw[f] : swMed; // shoulder width,
  median-clamped
  const ax = ((windowData[base + lsRing * coordCount]
    + windowData[base + rsRing * coordCount]) / 2) * aspect;
  const ay = (windowData[base + lsRing * coordCount + 1]
    + windowData[base + rsRing * coordCount + 1]) / 2;
  const o = f * stride;
  for (const [ring, coco] of pose) { // 23 COCO body slots
    out[o + coco * 2] = (windowData[base + ring * coordCount] * aspect - ax) /
    scale;
    out[o + coco * 2 + 1] = (windowData[base + ring * coordCount + 1] - ay) /
    scale;
  }
  for (const [start, group, bodySlot, localSlot] of hands) {
    if (windowMask[f * groupCount + group] <= 0) continue; // both blocks stay
    zero
    const wx = windowData[base + start * coordCount] * aspect;
    const wy = windowData[base + start * coordCount + 1];
    let span = 0;
    for (let i = 0; i < HAND_COUNT; i++) { // body-frame hand
      const px = windowData[base + (start + i) * coordCount] * aspect;
      const py = windowData[base + (start + i) * coordCount + 1];
      out[o + (bodySlot + i) * 2] = (px - ax) / scale;
      out[o + (bodySlot + i) * 2 + 1] = (py - ay) / scale;
      span = Math.max(span, Math.hypot(px - wx, py - wy));
    }
    if (span > 1e-6) { /* hand-local block: (p - wrist) / span, same loop */ }
  }
}
```

A.2 Ensemble distillation step

One step of the distillation loop (project/local/phaseb_distil.py), implementing Eq. (4.3). The teacher posterior pt is the mean softmax of the twelve fine-tunes; the student also keeps the auxiliary ASL loss.

```
pt = teacher_probs(xb) # mean softmax of the N
teachers, no grad
xf = ((xb - mu_d) / sd_d).float()
opt.zero_grad()
lg = logits(xf)
ce = F.cross_entropy(lg, ytr[b], label_smoothing=0.1)
tgt = pt.clamp_min(1e-8).pow(1.0 / TEMP)
kd = F.kl_div(F.log_softmax(lg / TEMP, 1), tgt / tgt.sum(1, keepdim=True),
              reduction="batchmean") * TEMP * TEMP
loss = (1 - ALPHA) * ce + ALPHA * kd + (phon_aux[0] if student_phon else 0.0)
if student_asl: # one ASL batch per ISL batch
    ab = a_tr[torch.randint(len(a_tr), (len(b),), generator=g_dev, device=dev)]
    ...
    loss = loss + asl_w * F.cross_entropy(head_a(oa[:, 0]), ya[ab],
label_smoothing=0.1)
loss.backward()
opt.step()
```

A.3 Temperature calibration with cross-fit

The calibration tool (apps/isl-spa/tools/fit-dictionary-calibration.py), implementing Eq. (4.5) and Eq. (4.6), and the transfer test of Algorithm 4.5.

```
def temper(p, t):
    logp = np.log(np.clip(p, 1e-12, 1.0)) / t
    logp -= logp.max(1, keepdims=True)
    q = np.exp(logp)
    return q / q.sum(1, keepdims=True)

def fit_t(p, y):
    grid = np.exp(np.linspace(np.log(0.2), np.log(3.0), 400))
    return float(grid[np.argmin([nll(temper(p, t), y) for t in grid])])

def ece(p, y, bins=10):
    conf, ok = p.max(1), p.argmax(1) == y
    edges = np.linspace(0, 1, bins + 1)
    total = 0.0
    for lo, hi in zip(edges[:-1], edges[1:]):
        m = (conf >= lo) & ((conf < hi) | (hi == 1))
        if m.any():
            total += m.mean() * abs(ok[m].mean() - conf[m].mean())
    return total

# cross-fit: each corpus's temperature must help the other corpus
cross = (ece(temper(p[cs], t_rk), y[cs]), ece(temper(p[rk], t_cs), y[rk]))
transfers = cross[0] < ece(p[cs], y[cs]) and cross[1] < ece(p[rk], y[rk])
```

A.4 The resting test of sign by sign

The geometric rule of Algorithm 4.7 (apps/isl-spa/src/engine/sign-by-sign.ts), in the model's canonical space (shoulder widths from mid-shoulder, y pointing down).

```
export const MIN_REST = 4; // consecutive resting frames that end a sign
export const MIN_SIGN = 10; // shortest span that counts as a sign
export const PAD = 2; // frames of run-up / run-down kept on each side

export function isRestingFrame(tensor: Float32Array, offset = 0): boolean {
  const y = (slot: number) => tensor[offset + slot * 2 + 1];
  const x = (slot: number) => tensor[offset + slot * 2];
  const hipsSeen =
    Math.abs(x(L_HIP)) + Math.abs(y(L_HIP)) + Math.abs(x(R_HIP)) +
    Math.abs(y(R_HIP));
  const hip = hipsSeen > 0 ? (y(L_HIP) + y(R_HIP)) / 2 : HIP_FALLBACK;
  for (const wrist of [L_WRIST, R_WRIST]) {
    const present = Math.abs(x(wrist)) + Math.abs(y(wrist)) > 1e-6;
    if (present && y(wrist) < hip - HIP_MARGIN) return false;
  }
  return true;
}
```

APPENDIX B

Research Paper

The manuscript below was prepared from this case study for submission to a peer-reviewed venue. It has not been submitted at the time of this report; it is included in the form in which it will be submitted, in the IEEE two-column format.

Learning Indian Sign Language from Other Signers: Self-Supervision, Cross-Lingual Transfer and an Unseen-Signer Test for a Browser-Sized Recogniser

Mithunish Prabhakaran

Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur, Chennai, India
pmithunish@gmail.com

S. Saravanan

Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur, Chennai, India

Abstract—Isolated sign recognisers for Indian Sign Language (ISL) are trained on corpora recorded by a handful of signers and are usually evaluated on those signers, so their reported accuracy says little about the person in front of the camera. We build a word-level ISL recogniser that runs entirely in a web browser and improve it against one pre-stated test: 599 clips from a signer absent from every training corpus and from dictionary clips never trained on, scored through the deployed int8 graph. The recogniser is an articulator-token transformer over 107 MediaPipe landmarks, pretrained with a joint-embedding predictive objective (ISL-JEPA) on 123,340 unlabelled clips, taught 250 American Sign Language (ASL) signs from 94,290 clips of 21 signers, fine-tuned on ISL with gloss-matched dictionary clips and geometric synthetic signers, and distilled from twelve such models into one 5.2 MB student. Over four seeds per recipe, a supervised ASL stage raises single-model accuracy on the test by up to 4.8 points and on the unseen signer by up to 3.5 points; unlabelled ASL clips used as extra pretraining data do not help. The deployed student reaches 0.553 on the test and 0.514 on the unseen signer, against 0.147 and 0.332 for a supervised baseline, with an expected calibration error of 0.039 after a cross-fitted temperature. A geometric rest-pose segmenter doubles word accuracy on continuous signing (0.230 to 0.498). We report the negative results alongside.

Index Terms—Indian Sign Language, sign language recognition, self-supervised learning, cross-lingual transfer, knowledge distillation, on-device inference

I. Introduction

India's 2011 census counted 5.07 million people with a hearing disability [1], and the national directory lists 339 certified ISL interpreters [2]. A recogniser that lets a learner check their signing, or lets a Deaf signer be understood by a device, must work for signers it has never seen. Public ISL resources make that hard: INCLUDE has 4,287 videos of 263 words from a few signers [3], CISLR one or two dictionary clips per word [4], and the ISLRTC dictionary single takes by a few presenters. Within-corpus numbers hide the problem: in our pipeline a supervised model scored 0.82 on held-out signers of its own corpus and 0.33 on a signer from another corpus, 0.12 when that signer led with the left hand.

We study how far a browser-sized ISL recogniser can be moved on an unseen signer *without recording new signers*, using only public data. Our contributions are:

- 1) an evaluation protocol: one 599-clip test of an unseen signer and never-trained dictionary clips, computed through the deployed int8 graph, with four seeds per claim and a promotion rule on both the whole test and the unseen signer;
- 2) ISL-JEPA, a joint-embedding predictive objective over articulator tokens of landmark sequences, and the observation that its encoder transfers when cut at layer four although its deep target collapses;
- 3) a seed-level measurement of supervised transfer from ASL to ISL, with ensembles and distillation, against unlabelled ASL as pretraining data;
- 4) a deployed 5.2 MB recogniser with cross-fitted calibration and a geometric sign-by-sign decoder.

II. Related Work

Datasets. Word-level recognition matured on ASL with WLASL [5] and MS-ASL [6]; Google's Isolated Sign Language Recognition corpus (PopSign ASL v1.0) provides 94,477 landmark sequences of 250 signs from 21 Deaf signers [7], [8]. ISL has INCLUDE [3], CISLR [4] and iSign [9].

Pose-based recognition. Landmark models such as SPOTER [10], SAM-SLR [11] and OpenHands [12] trade appearance for speed and invariance; MediaPipe [13], [14] makes landmarks available in a browser.

Self-supervision. Joint-embedding predictive architectures predict latent targets of masked content [15], [16]; S-JEPA [17] and Skeleton2vec [18] apply them to skeletons, and SignBERT+ [19] and SHuBERT [20] to sign language.

Cross-lingual transfer. Transfer between sign languages has helped low-resource languages [21], [22], [23], and pre-training on large multi-language collections is the premise of [24], [25]. PhonSSM structures the classifier around sign phonology [26].

Distillation and calibration. Distillation transfers an ensemble into one model [27]; temperature scaling calibrates it [28].

III. Method

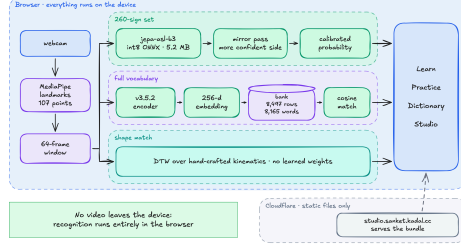


Fig. 1. The system at run time. Landmark extraction, the 260-sign classifier, and the shape matcher run in the browser; only static files are downloaded.

The recogniser runs entirely in the browser (Fig. 1): MediaPipe extracts landmarks in a worker, the video frame is discarded, and the classifier runs in ONNX Runtime Web [29].

A. Input representation

Each frame is reduced to 107 points: 23 COCO body points (the legs are zeroed), 21 points per hand in the body frame, and 21 per hand in a hand-local frame. The x coordinate is multiplied by the camera aspect; body and body-frame hand points are translated to the mid-shoulder and divided by the shoulder width (degenerate widths are replaced by the window median); hand-local points are taken relative to the wrist and divided by the hand's span. A window is 64 frames; velocities and bone vectors are derived inside the model. The same function runs in the browser, in extraction and in evaluation.

B. ISL-JEPA pretraining

Each frame contributes three articulator tokens (body, left hand, right hand), 192 tokens per window, encoded by an 8-layer transformer of width 256. A 4-layer predictor with mask queries predicts the layer-normalised, centred latents of an EMA target encoder at masked positions. Masks hide one articulator over a span (M1) or a temporal span of all articulators (M2); a third term (M3) maps a nuisance view — mirrored, tempo-warped, morphed, jittered, with a hand dropped — onto the clean latents. A hinge on the per-dimension standard deviation of the context latents guards against collapse:

$$\mathcal{L} = \text{SL}_1(\hat{z}_M, \bar{z}_M - c) + \lambda_v \mathcal{L}_{\text{view}} + \lambda_\sigma \text{mean}_d[\gamma - \sigma_d]_+ (1)$$

Phase B ran 100 epochs on a 123,340-clip pool of INCLUDE, CISLR, ISLRTC and unlabelled dictionary clips (11.1 h on an A100). Its last-layer target over-specialised late in training, but layers up to four kept transferring; all models below use the encoder cut at layer four.

C. ISL fine-tuning

An attentive-pooling head is trained on 260 classes with label smoothing 0.1, AdamW (encoder 10^{-4} , head 10^{-3} , weight decay 0.05) and a cosine schedule over 40 epochs. Three additions were each kept only after a multi-seed gain: *gloss infusion* (CISLR and ISLRTC clips whose gloss matches a class; the CISLR test role is excluded), *geometric synthetic signers* (landmarks lifted to 3-D with MediaPipe depth, rotated, retargeted to new limb proportions and projected back), and *recover-and-resample* (with probability 0.5, up to 25% of frames are dropped from each end and the rest is stretched to 64 frames). A reviewed data contract — sense-conflated labels, lesson-length files and variant signs found by a dataset-review tool — produced the v4 training pack.

D. Transfer from ASL

The Google ISLR landmarks were streamed from the 40 GB archive by HTTP byte ranges on a CPU task (77 s) and canonicalised with the same rules; the unknown capture aspect was estimated at 4:3 from five cues against pixel-true corpora, and 187 clips under four frames were dropped (94,290 kept). Two variants were tested: *aslp_{re}*, a supervised 250-way ASL stage of 6 or 16 epochs (batch 128, one-cycle learning rate 2×10^{-4}) before the ISL fine-tune, and *asl*, an ASL head kept during the fine-tune as an auxiliary loss on one ASL batch per ISL batch (weight 0.5). A *phonological head* [26] optionally replaces the linear head: four branches (handshape, location, movement, orientation) with 30, 15, 10 and 8 learned prototypes are fused and scored by cosine similarity against one prototype per sign, with orthogonality (0.1) and diversity (0.01) terms.

E. Distillation, export and calibration

The teacher is the mean softmax $\bar{p}$ of N fine-tunes; the student minimises $(1 - \alpha)\text{CE}_{0.1} + \alpha T^2 \text{KL}(\bar{p}_T \parallel \text{softmax}(z/T))$ with $\bar{p}_T \propto \bar{p}^{1/T}$, $\alpha = 0.7$, $T = 2, 40$ epochs. The student is exported to ONNX with normalisation and softmax inside the graph and quantised to int8. At run time each window is scored as captured and mirrored, and the more confident side answers. One temperature T^* is fitted by negative log-likelihood on that posterior; because the only cross-signer labels are the test clips, the temperature fitted on each corpus must lower the expected calibration error (ECE) on the other before the pooled T^* ships. A temperature cannot change top-1.

F. Sign by sign

For continuous signing, a frame *rests* when no present wrist is above the hip line; a span opens at the first non-resting frame, closes after four resting frames, is dropped if shorter than ten frames, padded by two, resampled to 64 frames and classified once.

IV. Evaluation Protocol

The test is clip-weighted top-1 over 599 clips: RKMVU, 220 clips from a signer in no training corpus, and the CISLR test role, 379 dictionary clips never trained on. One clip is

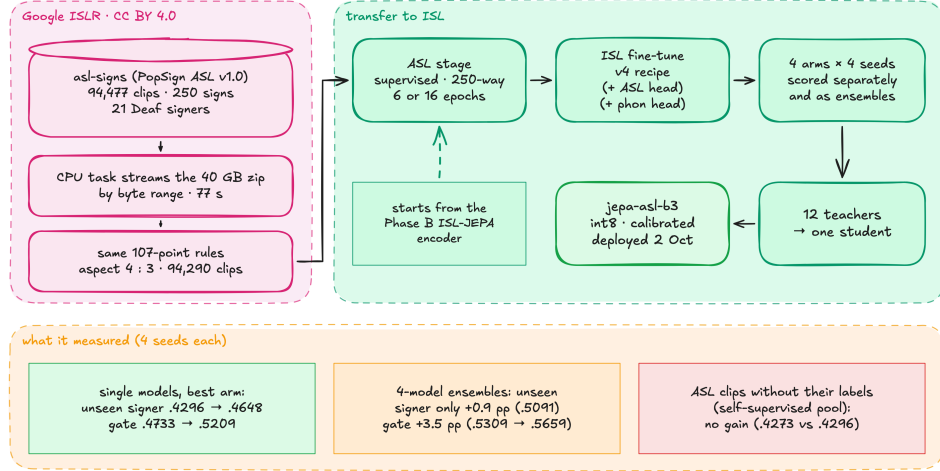


Fig. 2. Transfer from ASL: ingest and canonicalisation (left), the transfer recipes and their distillation (right), and what was measured (bottom).

TABLE I
Transfer from ASL: four-seed means of single models (fp32).

Recipe	RKMVU	CISLR	Test
Control (v4)	.4296	.4987	.4733
asl	.4375	.5257	.4934
aslppe (6 ep.)	.4523	.5449	.5109
asl+aslppe	.4648	.5284	.5050
aslppe (16 ep.)	.4239	.5495	.5034
16 ep. + phon. head	.4568	.5581	.5209
asl+aslppe+phon.	.4511	.5561	.5175

0.17 points of the test and 0.45 points of RKMVU. CISLR test clips share presenters with infusion clips, so RKMVU is the stranger number. Every recipe is run with four seeds; ensembles and students are fixed before they are scored; a model is promoted only if it beats the deployed model on both the test and RKMVU.

V. Results

Transfer from ASL. Every ASL recipe beat the control on the test, by 2.0 to 4.8 points (Table I); on RKMVU the best recipes gained 3.5 (asl+aslppe) and 2.7 points (16-epoch stage with the phonological head). A longer ASL stage alone moved CISLR (+5.1) but not the stranger (−0.6), consistent with CISLR rewarding presenter familiarity.

Unlabelled ASL does not help. Continuing ISL-JEPA on the pool plus the ASL clips and fine-tuning with the control recipe gave .4687 on the test and .4273 on RKMVU over four seeds, against .4816 and .4250 for the same continuation on the pool alone and .4733 and .4296 for Phase B — ASL helps as labels, not as more unlabelled motion.

TABLE II
Students distilled from ASL-pretrained teachers (fp32).

Model	Test	RKMVU	CISLR
Supervised baseline	.1469	.3318	.0396
Previous deployed (12 teachers, no ASL)	.5509	.5000	.5805
A: 8 teachers	.5426	.4864	.5752
B: 12 teachers, seed 1	.5543	.4955	.5884
B: seed 2	.5526	.4955	.5858
B: seed 3 (deployed)	.5526	.5136	.5752
C: 12 phonological teachers	.5376	.4727	.5752

Ensembles. Scored as four-model ensembles, the recipes rank differently: aslppe reaches .5659 on the test and asl+aslppe .5091 on RKMVU, while the phonological recipes gain least from averaging. All 34 fine-tunes together reach .5793 and .5318 (CISLR .6069), three points above any four-model ensemble on RKMVU, so teacher diversity sets the ceiling.

Students and the deployed model. Of five students (Table II) only B seed 3 beat the previous model on both numbers. As int8 it scores .5526 on the test, .5091 on RKMVU (.5136 with the mirror rule the application uses) and .5778 on CISLR, in 5.23 MB. Cross-fitting held (temperatures .731 and .843 on the two corpora); the pooled T^* = 0.804 lowers ECE from .160 to .039, and the area under the risk-coverage curve is .177 against .208 for the previous model.

Sign by sign. On four-sign sequences of joined RKMVU clips, sliding windows with a persistence rule reach a word accuracy of .230 (.376 when tuned on the same data), the rest-pose segmenter .498, and true boundaries .509; the segmenter finds exactly four signs in 98% of sequences. Played live into the application, a 16-clip RKMVU playlist scores 56–63% against 31–38% for the previous decoder.

Negative results. Whole-body RTMW landmarks lost to MediaPipe on the test (.238 vs .257, five seeds) although they gained 13 points on INCLUDE’s own split; photometric pre-processing did not recover missing hands, which go missing because they move or leave the frame; a model-free DTW shape matcher was a worse pseudo-label verifier than the model’s own confidence (55.3% vs 68.0% precision); the phonological head did not survive distillation from non-ASL teachers; and pre-registered browser ensembles of two or three models gained at most 1.2 points on the test and 0.5 on RKMVU at two to three times the compute.

VI. Discussion and Limitations

Across the project the stranger number rose from .332 to .514 and the test from .147 to .553 in a browser model of the same order of size. Four levers were measured: self-supervision with the layer-four cut, the reviewed data contract (+7.0 points on the test, four of four seeds), distillation, and supervised ASL transfer. Every lever moved CISLR more than RKMVU, which is why the promotion rule is written on both. The unseen-signer test is one person, so its 95% interval is about ± 6.6 points; the CISLR test shares presenters with training clips; continuous signing requires pauses between signs; and the ASL data are used under CC BY 4.0 with credit in the application.

VII. Conclusion

Without recording a single new signer, a browser-sized ISL recogniser was moved from .332 to .514 on a signer it has never seen, by learning from signers it could reach: unlabelled ISL motion, dictionary presenters, synthetic bodies and, most of all, 21 ASL signers whose labels transfer across languages. A multi-signer ISL test set is the most important next step.

References

- [1] Ministry of Statistics and Programme Implementation, “Disabled Persons in India: A Statistical Profile 2016,” Government of India, 2016. [Online]. Available: https://www.mospi.gov.in/sites/default/files/publication_reports/Disabled_persons_in_India_2016.pdf
- [2] Indian Sign Language Research and Training Centre, “Indian Sign Language Interpreters Directory.” [Online]. Available: <https://esic.gov.in/attachments/circularfile/4650b5eaa1bec5cc169937302ce1205c.pdf>
- [3] A. Sridhar, R. G. Ganesan, P. Kumar, and M. Khapra, “INCLUDE: A Large Scale Dataset for Indian Sign Language Recognition,” in *Proceedings of the 28th ACM International Conference on Multimedia (MM ’20)*, 2020, pp. 1366–1375. doi: 10.1145/3394171.3413528.
- [4] A. Joshi *et al.*, “CISLR: Corpus for Indian Sign Language Recognition,” in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022, pp. 10357–10366. doi: 10.18653/v1/2022.emnlp-main.707.
- [5] D. Li, C. Rodriguez Opazo, X. Yu, and H. Li, “Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods Comparison,” in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 1448–1458. doi: 10.1109/WACV45572.2020.9093512.
- [6] H. R. Vaezi Joze and O. Koller, “MS-ASL: A Large-Scale Data Set and Benchmark for Understanding American Sign Language,” in *British Machine Vision Conference (BMVC)*, 2019.
- [7] T. Starner *et al.*, “PopSign ASL v1.0: An Isolated American Sign Language Dataset Collected via Smartphones,” in *NeurIPS (Datasets and Benchmarks Track)*, 2023, pp. 184–196. doi: 10.52202/075280-0011.
- [8] A. Chow *et al.*, “Google – Isolated Sign Language Recognition.” [Online]. Available: <https://www.kaggle.com/competitions/asl-signs>
- [9] A. Joshi *et al.*, “iSign: A Benchmark for Indian Sign Language Processing,” in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024, pp. 10827–10844. doi: 10.18653/v1/2024.findings-acl.643.
- [10] M. Boháček and M. Hruš, “Sign Pose-based Transformer for Word-level Sign Language Recognition,” in *2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, 2022, pp. 182–191. doi: 10.1109/WACVW54805.2022.00024.
- [11] S. Jiang, B. Sun, L. Wang, Y. Bai, K. Li, and Y. Fu, “Skeleton Aware Multi-modal Sign Language Recognition,” in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021, pp. 3408–3418. doi: 10.1109/CVPRW53098.2021.00380.
- [12] P. Selvaraj, G. NC, P. Kumar, and M. Khapra, “OpenHands: Making Sign Language Recognition Accessible with Pose-based Pretrained Models across Languages,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022, pp. 2114–2133. doi: 10.18653/v1/2022.acl-long.150.
- [13] C. Lugaresi *et al.*, “MediaPipe: A Framework for Building Perception Pipelines,” *arXiv:1906.08172*, 2019, [Online]. Available: <https://arxiv.org/abs/1906.08172>
- [14] F. Zhang *et al.*, “MediaPipe Hands: On-device Real-time Hand Tracking,” in *CVPR Workshop on Computer Vision for Augmented and Virtual Reality*, 2020. [Online]. Available: <https://arxiv.org/abs/2006.10214>
- [15] M. Assran *et al.*, “Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture,” in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 15619–15629. doi: 10.1109/CVPR52729.2023.01499.
- [16] A. Bardes *et al.*, “Revisiting Feature Prediction for Learning Visual Representations from Video,” *Transactions on Machine Learning Research (TMLR)*, 2024, [Online]. Available: <https://openreview.net/forum?id=QaCCuDFbk2>
- [17] M. Abdelfattah and A. Alahi, “S-JEPA: A Joint Embedding Predictive Architecture for Skeletal Action Recognition,” in *Computer Vision – ECCV 2024, Lecture Notes in Computer Science*, vol. 15090, Springer, 2024, pp. 367–384. doi: 10.1007/978-3-031-73411-3_21.
- [18] R. Xu, L. Huang, M. Wang, J. Hu, and W. Deng, “Skeleton2vec: A Self-supervised Learning Framework with Contextualized Target Representations for Skeleton Sequence,” *arXiv:2401.00921*, 2024, [Online]. Available: <https://arxiv.org/abs/2401.00921>
- [19] H. Hu, W. Zhao, W. Zhou, and H. Li, “SignBERT+: Hand-Model-Aware Self-Supervised Pre-Training for Sign Language Understanding,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 9, pp. 11221–11239, 2023. doi: 10.1109/TPAMI.2023.3269220.
- [20] S. Gueuwou, X. Du, G. Shakhnarovich, K. Livescu, and A. H. Liu, “SHuBERT: Self-Supervised Sign Language Representation Learning via Multi-Stream Cluster Prediction,” in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 28792–28810. doi: 10.18653/v1/2025.acl-long.1397.
- [21] R. Holmes *et al.*, “From Scarcity to Understanding: Transfer Learning for the Extremely Low Resource Irish Sign Language,” in *ICCV Workshops*, 2023, pp. 2000–2009. doi: 10.1109/ICCVW60793.2023.00215.
- [22] Y. C. Bilge, N. Ikizler-Cinbis, and R. G. Cinbis, “Cross-lingual few-shot sign language recognition,” *Pattern Recognition*, vol. 151, p. 110374, 2024. doi: 10.1016/j.patcog.2024.110374.
- [23] A. A. Kindiroglu, O. Kara, O. Ozdemir, and L. Akarun, “Transfer Learning for Cross-dataset Isolated Sign Language Recognition in Under-Resourced Datasets,” in *IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, 2024, pp. 1–8. doi: 10.1109/FG59268.2024.10582029.
- [24] I. Ovodov, P. Surovtsev, K. Kvanchiani, A. Kapitanov, and A. Nagaev, “Logos as a Well-Tempered Pre-train for Sign Language Recognition,” in *EMNLP*, 2025, pp. 24351–24364. doi: 10.18653/v1/2025.emnlp-main.1238.
- [25] Z. Li, W. Zhou, W. Zhao, K. Wu, H. Hu, and H. Li, “Uni-Sign: Toward Unified Sign Language Understanding at Scale,” in *International Conference on Learning Representations (ICLR)*, 2025.

- [26] B. Cheng, A. Jin, and J. Zhang, "State Space Models are Effective Sign Language Learners: Exploiting Phonological Compositionality for Vocabulary-Scale Recognition," *arXiv:2604.08761*, 2026, [Online]. Available: <https://arxiv.org/abs/2604.08761>
- [27] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," *arXiv:1503.02531 (NIPS 2014 Deep Learning Workshop)*, 2015, [Online]. Available: <https://arxiv.org/abs/1503.02531>
- [28] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, PMLR vol. 70, 2017, pp. 1321–1330.
- [29] Microsoft, "ONNX Runtime Web — WebAssembly execution provider." [Online]. Available: <https://onnxruntime.ai/docs/tutorials/web/>

APPENDIX C

Screenshots of the Working System

The Studio, Sign by sign, Model Lab, Eval bench and Settings screenshots were taken on 6 October 2026 from the running studio at 1488×1000 pixels, light theme, with `jepa-asl-b3` deployed; Learn, Practice, Dictionary and Dataset date from 27 September and do not depend on the model. The developer build also shows the developer-only tabs (Model Lab, Eval bench, Dataset, Review). Where a camera is shown, the input is a public RKMVERI dictionary clip played into the pipeline in place of a webcam.

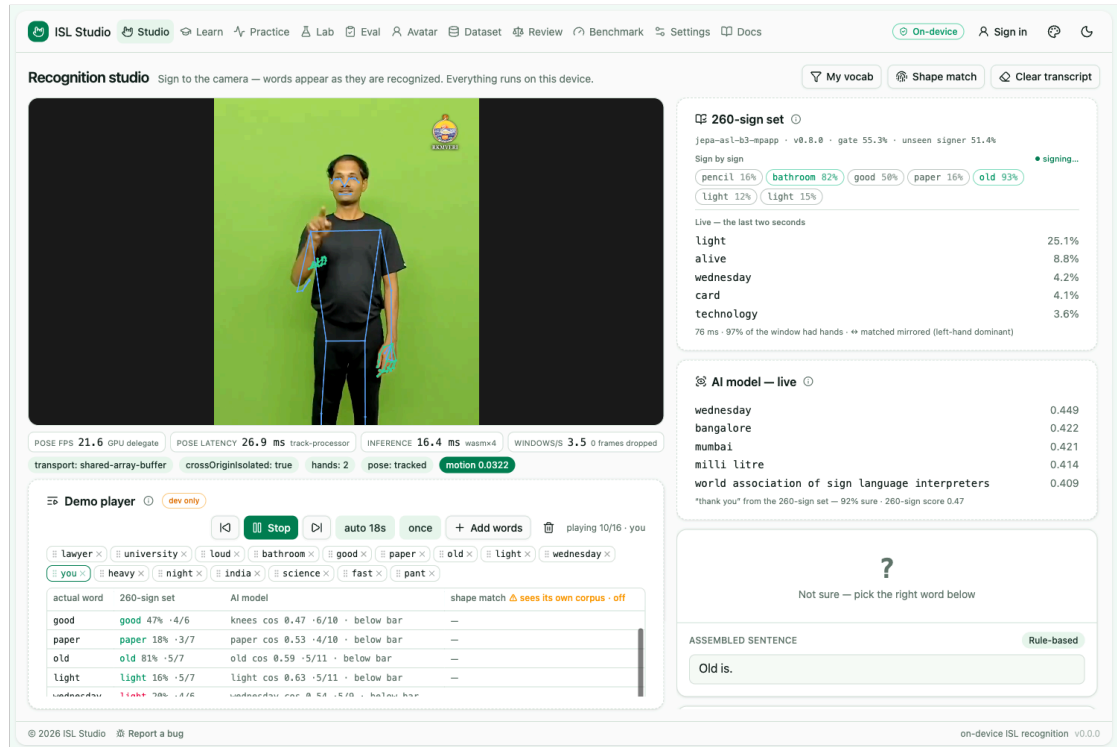


Figure C.1. Studio with the demo player: dictionary clips play through the real pipeline, and the table below the video records, per clip, the actual word against each recogniser.

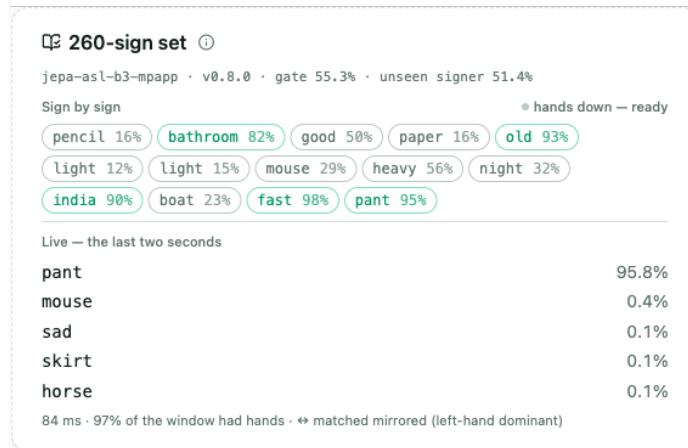


Figure C.2. Sign by sign: the stream is split where the hands come to rest, and each sign is named once, with its calibrated confidence.

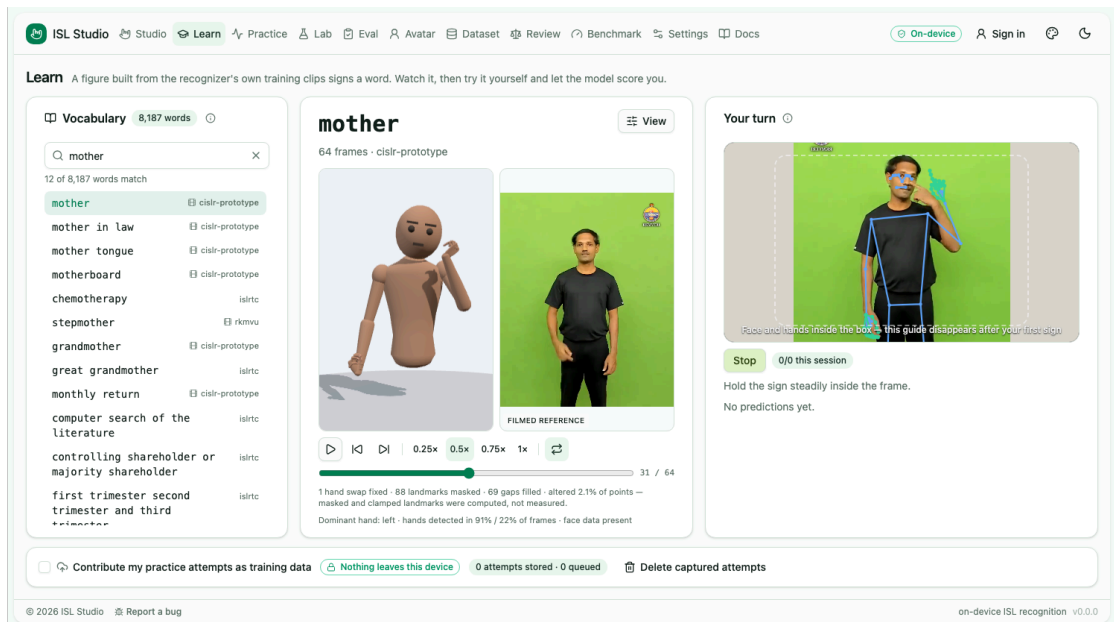


Figure C.3. Learn: the vocabulary list, a figure signing the word from the recogniser's own training clip beside the filmed reference, and *Your turn* with the camera running.

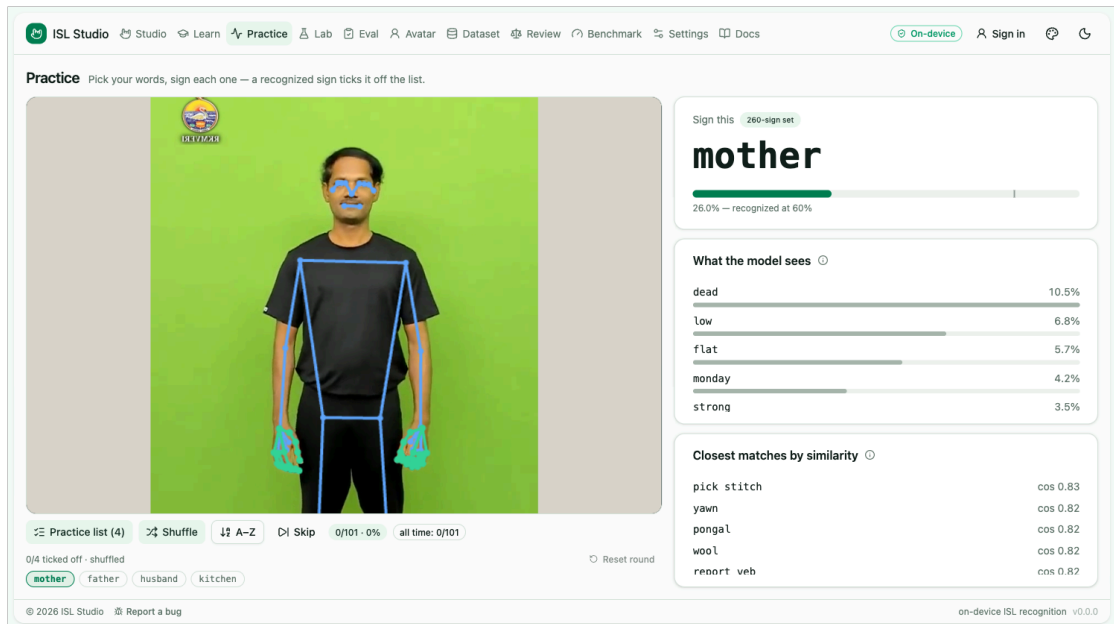


Figure C.4. Practice: the target word with live progress towards the credit bar, the 260-sign set's top five, and the closest full-vocabulary matches.

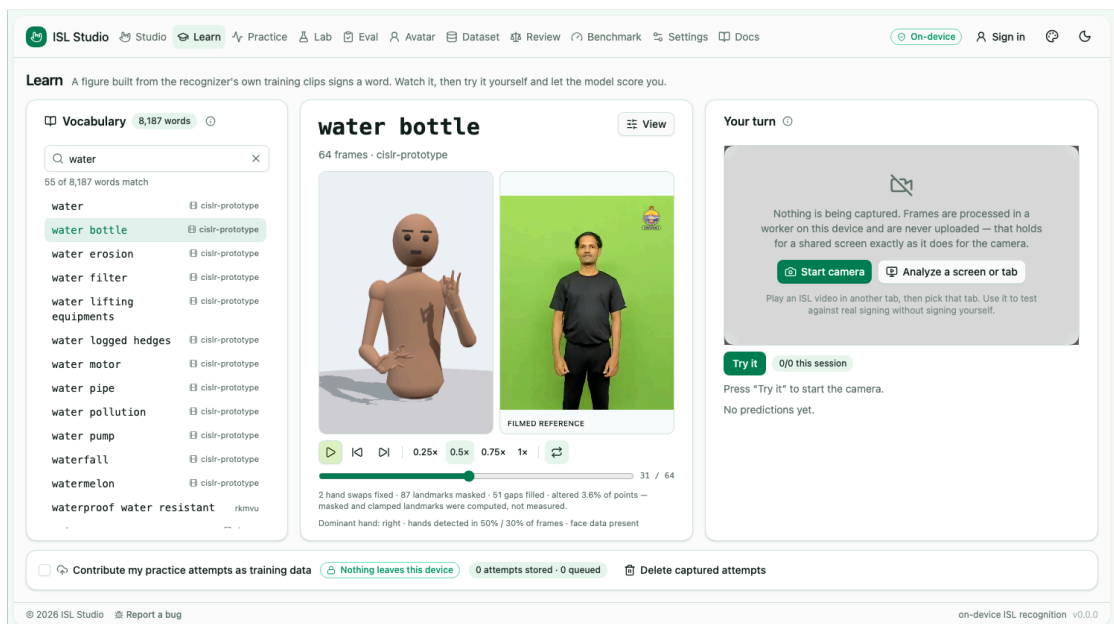


Figure C.5. Dictionary look-up in Learn: a search returns the matching words with their source corpus; the selected entry is demonstrated by the figure and the filmed reference.

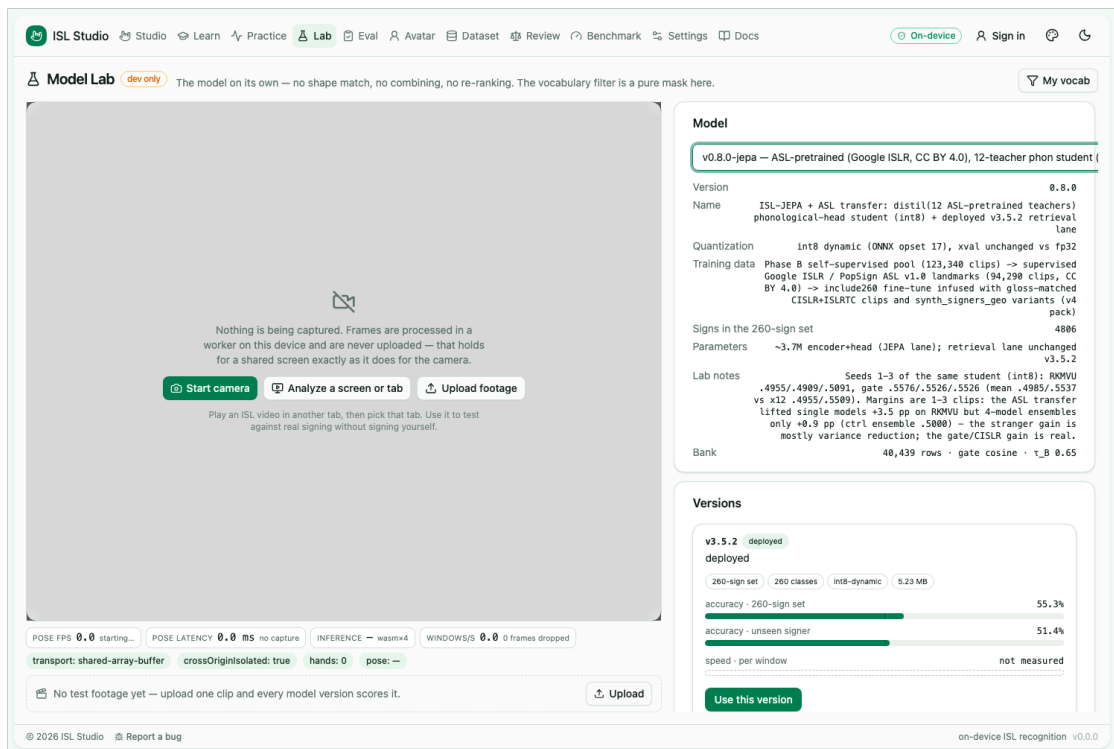


Figure C.6. Model Lab (developer only): the jepa-asl-b3 bundle’s model card, read from the bundle itself, with its numbers, calibration and the credit to the ASL source.

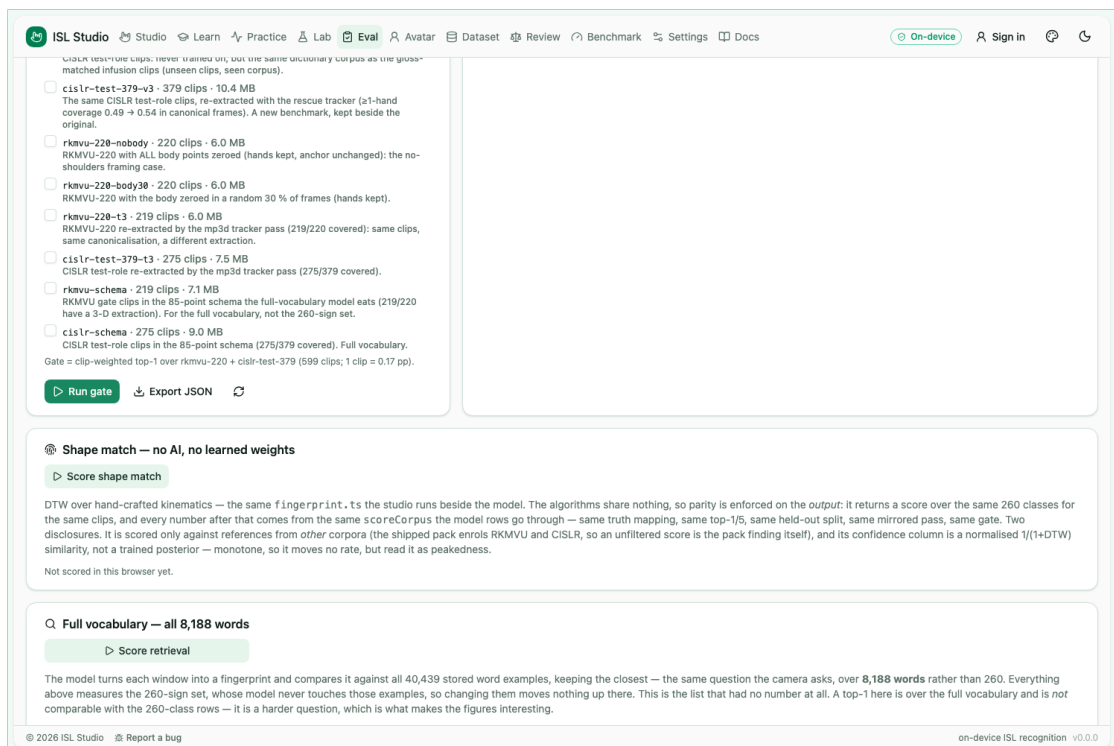


Figure C.7. Eval bench (developer only): a fresh in-browser run of the accuracy test on the int8 graphs of jepa-asl-b3, jepa-x12 and jepa-rr4 (gate .5559, .5492 and .4591); drift beyond one point from the bundle’s own claim would indicate a packaging bug.

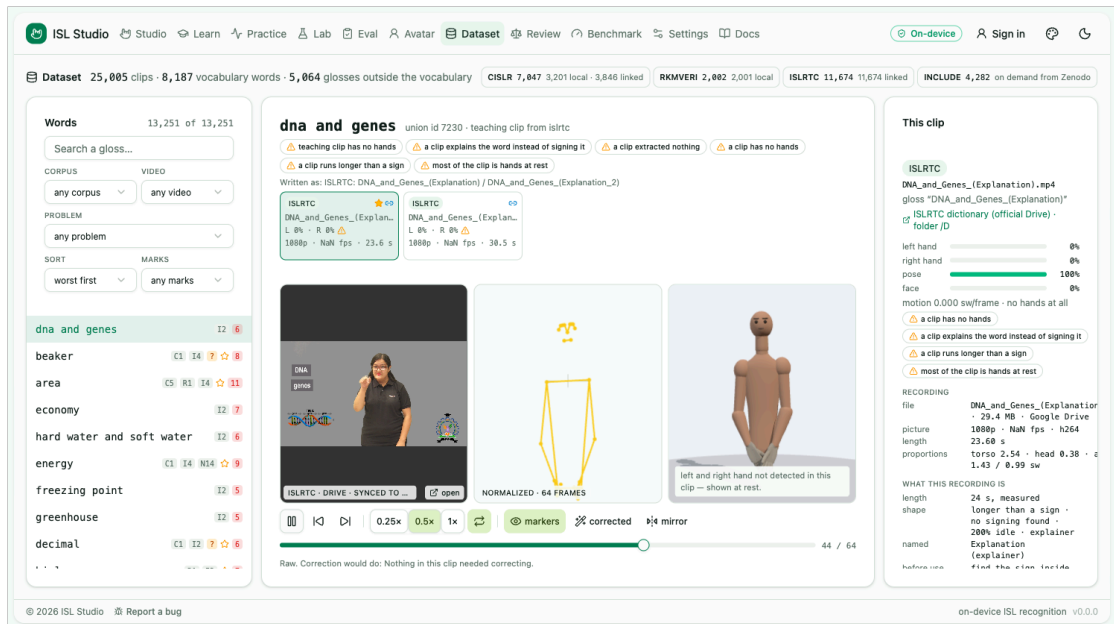


Figure C.8. Dataset tab (developer only): every clip of the four ISL corpora, sorted worst-first, with the video, its normalised landmarks and a figure rendering side by side, and the marks attached to it.

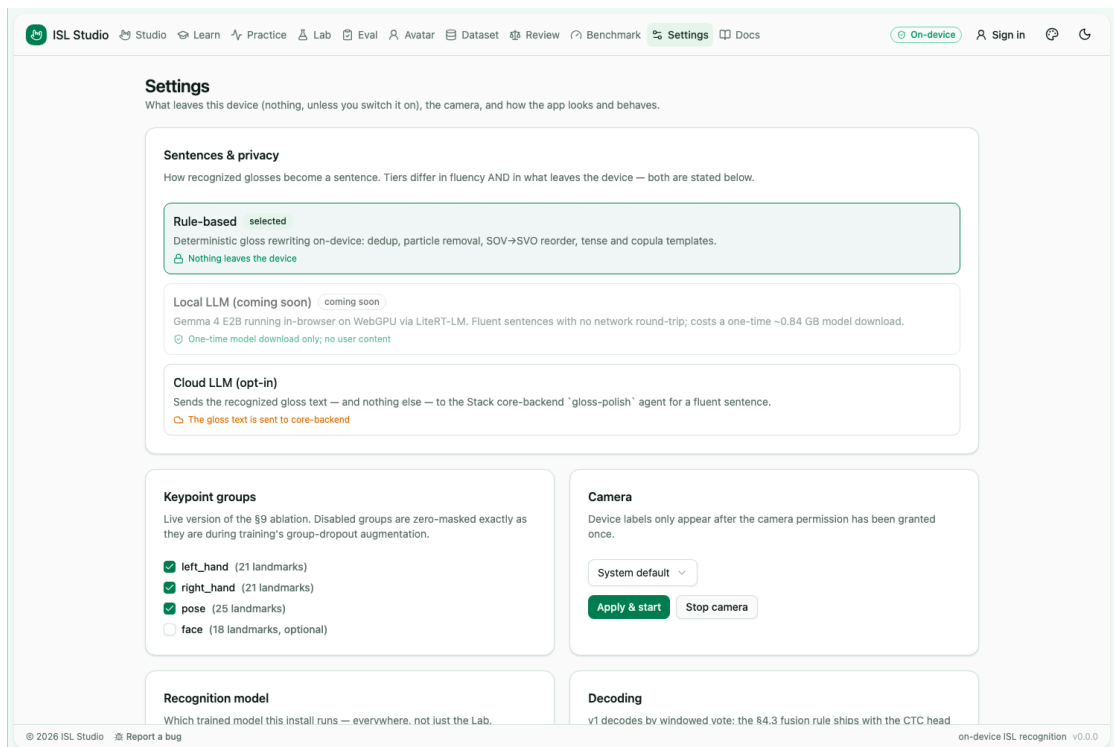


Figure C.9. Settings: what leaves the device (nothing, unless the user opts in), the deployed model's stated accuracy on an unseen signer, and the credit to the ASL source.

APPENDIX D

Reproducibility

D.1 Where the work lives

- **Application and documentation.** The studio is `apps/isl-spa` and the landing and documentation site `apps/isl-landing` in the candidate’s monorepo (branch `case-study/isl-spa`); this report is `apps/isl-landing/docs-src/final/report.typ`. Live: `sanket.kadal.cc` (documentation, with every review’s record under `/docs`) and `studio.sanket.kadal.cc` (application).
- **Training and extraction.** The case-study repository’s `project/` tree: `local/` holds the ISL-JEPA trainer (`train_isl_jepa.py`), the fine-tune and distillation scripts (`phaseb_infuse_ft.py`, `phaseb_distil.py`), the ASL ingest and canonicaliser (`islr_ingest.py`, `islr_canon.py`), the ensemble and ONNX scorers (`ens_eval.py`, `onnx_gate_score.py`, `ens_onnx.py`) and the exporter (`phaseb_ship_export.py`); `beam/` and `modal_jobs/` hold the same scripts’ cloud launchers.

D.2 Versions of record

Table D.1. The commits and artefacts behind the deployed model.

Item	Identifier
Deployed bundle	<code>public/models/current = lab/jepa-asl-b3</code> ; int8 ONNX, 5.23 MB; $T^* = 0.804$
Promotion commit (application)	<code>abfd67be</code> — promote <code>jepa-asl-b3</code> (previous bundle: <code>jepa-x12</code>)
Documentation updates	<code>a206bb31</code> , <code>ec917407</code> — results page
Training code (case-study repository)	<code>e9ef6d8</code> (Beam launcher), <code>226587f</code> (ASL ingest, canonicalisation, ASL arms, ASL-aware distillation), <code>f7d60ca</code> (ensemble and ONNX scorers), <code>4d44e1a</code> (continued self-supervision)
Data licences	INCLUDE, CISLR, ISLRTC and RKMVU under their published terms; Google ISLR / PopSign ASL v1.0 under CC BY 4.0 (competition data not redistributed)

D.3 Commands

Table D.2. Commands that reproduce the numbers in this report (abridged).

Step	Command
Ingest Google ISLR on a CPU task	<code>beam deploy islr_ingest_beam.py:ingest --name islr-ingest</code> , then one task request
Canonicalise ISLR	<code>ISLR_ASPECT=1.3333 python project/local/islr_canon.py</code>
Fine-tune arms with seeds	<code>INFUSE_ARMS='all+synth:full@2+rr+aslpres'</code> <code>INFUSE_SEEDS=4 python project/local/phaseb_infuse_ft.py</code>
Score ensembles	<code>python project/local/ens_eval.py 'w_aslpres/*.pt' 'w_aslboth/*.pt' ...</code>
Distil a student	<code>DISTIL_TEACHER_DIR=<12 teachers> DISTIL_SEED=3</code> <code>python project/local/phaseb_distil.py</code>
Score an int8 graph	<code>python project/local/onnx_gate_score.py <model.int8.onnx></code>
Calibrate a bundle	<code>uv run --with onnxruntime python tools/fit-dictionary-calibration.py --write</code>
Accuracy test in the browser	Eval bench in the running studio (bun run dev, port 5182)
Build this report	<code>typst compile --root .. final/report.typ final/report.pdf</code>

D.4 Evaluation discipline

Every accuracy-test number is computed on the same 599 tensors, built by the same canonicalisation the browser runs; the Eval bench replays them through the shipped int8 graph. The CISLR test role and the RKMVU clips are never admitted to training, and a leak check against the training rows runs before a number is recorded. Claims about recipes rest on at least four seeds; ensembles and students are pre-registered before they are scored; calibration follows the cross-fit rule of Algorithm 4.5.

D.5 Compute

The self-supervised pretraining (Phase B, 100 epochs on 123,340 clips) ran on an A100 in 11.1 hours. The Review 2 fine-tunes ran on Modal (A10G, about five minutes per seed) until that workspace was disabled on 27 September; later work ran on Beam (the ASL ingest on a CPU task, about US \$0.02) and on a Colab Pro A100 (the thirty-four ASL fine-tunes and the three students), with the local Apple GPU for scoring.

APPENDIX E

Record of Mentor Meetings

Per the course guidelines, the student met the mentor regularly and kept minutes of each meeting on the prescribed Minutes of Meeting form. The signed forms for August, September and October 2026 are attached after this page. Table E.1 lists the review milestones the meetings prepared for.

Table E.1. Review milestones of the case study and what was presented at each.

Review	Date	Presented
Review 0	9 Aug 2026	Title, one-page abstract, literature review, novelty, proposed system and time frame; in-corpus baseline (INCLUDE 0.958)
Review I	30 Aug 2026	Literature survey, novelty justification, architecture; the cross-corpus gap ($0.82 \rightarrow 0.33$) and the mirror fix
Review II	27 Sep 2026	Implementation, intermediate results (jepa-x12: gate .5509, RKMVU .5000), 50% report
Final review	11 Oct 2026	Overall design, justification, results and discussion (jepa-asl-b3: gate .5526, RKMVU .5136), this report