COMPUTER VISION

Real-Time Static Hand-Gesture-to-Text Conversion Using MediaPipe Landmarks and Classical Machine Learning

Rudra Shelke, Samruddhi Kaule, Samarthi Mante, Pavan Memane, Viddulata Patil

Department of Computer Science and Engineering, MIT School of Computing, MIT Art, Design and Technology University, Pune, India

Published November 2026

Abstract

This paper presents a low-cost, browser-based system that converts static hand gestures into text in real time using only a standard webcam. Hand landmarks are extracted with MediaPipe Hands, converted into a 42-dimensional wrist-relative, scale-normalized feature vector, and classified into 36 classes (digits 0–9 and letters a–z). A hold-to-confirm state machine with a short voting filter turns the per-frame predictions into words and sentences, and the application is deployed as a Streamlit web service using WebRTC. On a custom landmark dataset of 3,263 samples, a Random Forest classifier reaches 95.9% accuracy and 0.952 macro F1 on a stratified 20% hold-out set (majority-class baseline: 4.6%). Repeated stratified cross-validation on the training partition gives 94.2 ± 1.0% accuracy, and Random Forest, *k*-nearest neighbours and RBF-SVM classifiers differ by about one percentage point after tuning. Classifier-only inference takes a median of 2.8 ms per sample on a laptop CPU. The evaluation is row-level and not signer-independent; this limitation, together with confusable classes such as 0/o and 6/w, is analysed and directions for signer-independent evaluation are given.

Keywords: gesture recognition; hand landmarks; MediaPipe; random forest; real-time systems; Streamlit

1 Introduction

Hand gestures are a natural channel for human–computer interaction and, in the form of fingerspelling, a way to communicate individual letters and digits without speech. Practical gesture-to-text tools are nevertheless often held back by their requirements: depth sensors, wearable gloves, or deep convolutional networks that need large labelled image collections and GPU-class hardware [1]. Landmark-based pipelines offer an attractive alternative. A pretrained hand tracker reduces each video frame to a small set of joint coordinates, so that a lightweight classifier can be trained on only a few thousand samples and executed on a CPU.

This work builds a complete, deployable system around that idea. The contributions are: (i) a 42-dimensional, wrist-relative and scale-normalized landmark representation shared between training, evaluation, and live inference; (ii) a hold-to-confirm decision layer that converts noisy per-frame predictions into text; (iii) a reproducible evaluation protocol that keeps a fixed hold-out set outside all model selection, compares Random Forest, *k*-NN and RBF-SVM classifiers, and reports classifier-only latency; and (iv) a candid analysis of what this protocol does and does not establish.

The remainder of the paper is organized as follows. Related work is reviewed first, followed by the system overview, the dataset and features, the classifiers and real-time logic, the experimental protocol, the results and the limitations; the final section concludes the paper.

2 Related work

Vision-based hand gesture recognition has been surveyed extensively, covering segmentation, feature representation and classification techniques [1]. Early fingerspelling recognizers relied on depth cameras: Pugeault and Bowden combined Kinect depth and colour cues with random-forest classification to recognize American Sign Language letters in real time [2]. Such systems require dedicated hardware, which limits deployment.

MediaPipe Hands provides on-device, real-time tracking that regresses 21 three-dimensional hand landmarks from a single RGB frame [3], built on the MediaPipe framework for perception pipelines [4]. Because the tracker handles localization and pose estimation, downstream classifiers can operate on compact feature vectors instead of raw pixels. Classical supervised learners are natural candidates for this setting: Random Forests [5] are robust to feature scaling and nonlinear class boundaries, support vector machines [6] are effective in low-dimensional dense feature spaces, and nearest-neighbour rules [7] offer a simple instance-based baseline. This paper evaluates all three on the same landmark features and the same fixed split using scikit-learn [8].

3 System overview

Figure 1 shows the processing pipeline. Video is captured by the browser and streamed to the Python backend through WebRTC (the streamlit-webrtc component). Each frame is mirrored for a selfie view, converted from BGR to RGB with OpenCV [9], and passed to MediaPipe Hands configured for a single hand. The 21 landmarks are converted into a 42-D feature vector, classified by the trained model, and the label and class probability are handed to a stability filter and a hold-to-confirm state machine that appends letters to a word buffer. The interface (Figure 2) shows the landmark overlay, the current prediction with its confidence, a progress bar for the hold timer, and controls to add a letter, delete a letter, end a word, or clear the text. The application is deployed on Streamlit Community Cloud; it uses public STUN servers and can use a TURN relay configured through application secrets when direct WebRTC negotiation is blocked.

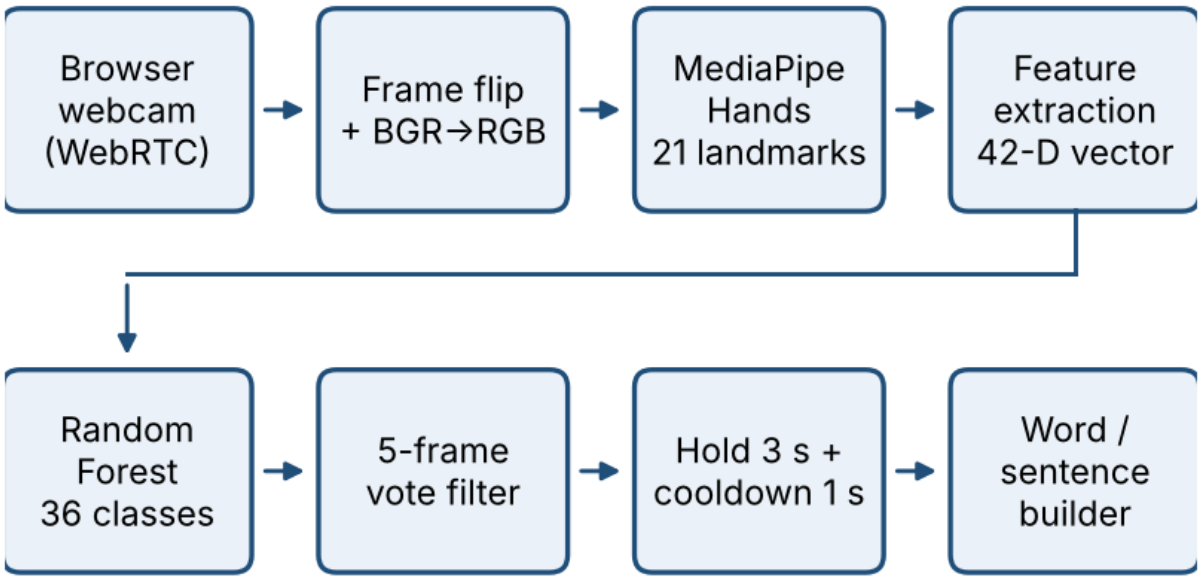


Figure 1. Processing pipeline of the proposed system, from browser webcam to word builder.

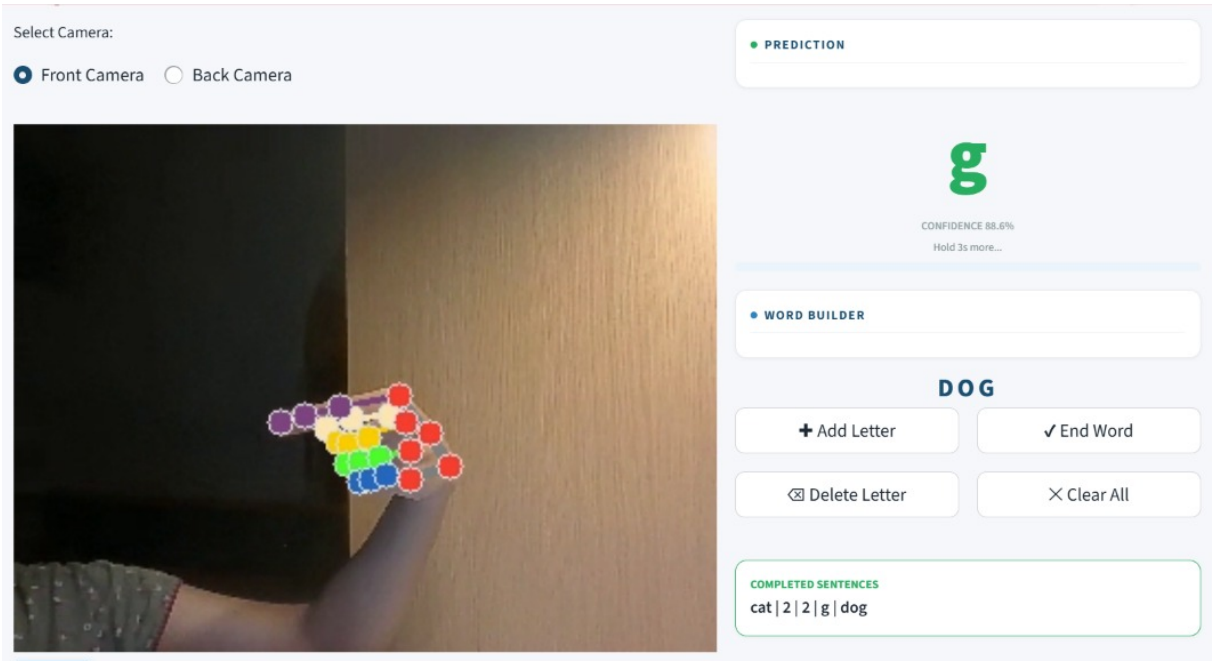


Figure 2. Deployed web interface: live landmark overlay, predicted class with confidence, hold-progress bar, word builder, and completed-sentence history.

4 Dataset and feature extraction

4.1 Data collection

A custom dataset was recorded with a webcam. For each class, the signer places the hand inside a fixed on-screen box and a 200 × 200 px region of interest is saved approximately every 150 ms for a 15 s capture window. The gesture set contains 36 static classes: digits 0–9 and letters a–z.

4.2 Landmark features

MediaPipe Hands is run in static-image mode (one hand, minimum detection confidence 0.3) on every saved crop; images without a detected hand are discarded. This yields 3,263 labelled samples, with 39 (class a) to 150 (class f) samples per class (mean ≈ 90.6). Let (x_i, y_i), $i = 0, \ldots, 20$, be the normalized landmark coordinates of a frame of width W and height H, with $i = 0$ the wrist. The aspect-corrected, wrist-relative coordinates are

$$q_i = \left(x_i \frac{W}{H} - x_0 \frac{W}{H},\ y_i - y_0 \right) \tag{1}$$

and the feature vector is the flattened array of

$$\hat{q}_i = \frac{q_i}{s},\ s = \max_{i,c} |q_{i,c}| \tag{2}$$

where c indexes the two coordinates, giving a 42-D vector invariant to hand position and to image scale. Samples whose landmark set is not exactly 21 finite points are rejected. The same function is used for dataset construction, evaluation, and live inference.

5 Classification and real-time decision logic

5.1 Classifiers

The deployed model is a Random Forest (RF) with 100 trees (scikit-learn defaults, fixed seed). Two comparators were evaluated on the same data: a k-NN classifier and an RBF-kernel SVM, each preceded by a StandardScaler fitted on training rows only. A majority-class dummy classifier serves as a sanity baseline.

5.2 From frame predictions to text

For each frame, the class with the highest predicted probability is taken as the candidate. A buffer of the last five candidates is kept, and a label is considered stable when it occurs at least twice in the buffer; the mean probability of the matching entries is displayed as confidence. A letter is appended to the current word only after the same stable label has been held for 3 s, after which a 1 s cool-down prevents repeated insertions. Frames with more than one detected hand are ignored, and a word is moved to the sentence history when no hand has been visible for 8 s. The buffer is cleared whenever no prediction is available.

6 Experimental protocol

The dataset was first split once into 2,610 training and 653 hold-out rows with a stratified 80/20 split (fixed seed). All model selection used the training partition only: repeated stratified 5-fold cross-validation (10 repeats, 50 folds, shared fold indices across models) with default hyperparameters, followed by a grid search with 5-fold cross-validation scored by macro F1. The grids were {100, 200, 300} trees and maximum depth {none, 20, 40} for RF; $k \in \{3, 5, 7, 9\}$ with uniform or distance weights for k-NN; and $C \in \{1, 10, 100\}$, $\gamma \in$ {scale, 0.01, 0.1} for SVM. Each tuned estimator was refitted on the full training partition and evaluated once on the hold-out rows. Metrics are accuracy, macro F1 and weighted F1.

Latency was measured as warm single-row predict() time: 50 warm-up calls, then five passes over the 653 hold-out rows (3,265 timed calls) with a monotonic high-resolution timer. Feature-row

slicing is excluded, scaler transforms are included for *k*-NN and SVM, and camera capture, MediaPipe inference, rendering and network time are excluded. Experiments ran on a 13th-generation Intel Core i5-1335U laptop with 15.7 GiB RAM under Windows 11 with Python 3.12.10, NumPy 1.26.4 and scikit-learn 1.5.2.

7 Results

7.1 Cross-validation

Table 1 reports repeated cross-validation with default hyperparameters on the training partition. RF is best on every metric, followed by *k*-NN and the untuned RBF-SVM.

Table 1. Repeated stratified 5-fold cross-validation (10 repeats) on the training partition.

Model	Accuracy	Macro F1	Weighted F1
Random Forest	0.942 ± 0.010	0.933 ± 0.012	0.942 ± 0.010
k-NN ($k = 5$)	0.919 ± 0.011	0.907 ± 0.013	0.919 ± 0.011
RBF-SVM ($C = 1$)	0.862 ± 0.012	0.835 ± 0.018	0.854 ± 0.014

7.2 Tuned models on the hold-out set

Table 2 gives the tuned results. Selection by training-only cross-validation macro F1 favours the RBF-SVM, whereas the hold-out accuracy is highest for RF. The three classifiers lie within about one percentage point of each other, comparable to the fold-to-fold standard deviation in Table 1, so this study does not establish a statistically meaningful winner. The deployed system uses RF because it exposes class probabilities natively. The deployed RF (100 trees, untuned) attains 95.87% accuracy (626 of 653 correct), 0.952 macro F1 and 0.958 weighted F1 on the hold-out set, against 4.59% for the majority-class baseline.

Table 2. Tuned classifiers: training-CV macro F1 and hold-out results (653 rows).

Model	Best parameters	CV F1	Accuracy	Macro F1
RF	depth 20, 300 trees	0.932	0.962	0.955
k-NN	$k = 3$, distance	0.931	0.960	0.953
RBF-SVM	$C = 100$, γ = scale	0.943	0.953	0.942

CV F1, macro F1 in training-partition cross-validation.

7.3 Error analysis

Figure 3 shows per-class F1 for the deployed RF. Six classes fall below 0.90: 0 (0.75), 6 (0.76), a (0.86), o (0.87), q (0.89) and w (0.90). For the SVM selected by cross-validation, the most frequent confusions are 6→w (4 cases), 0→o (3), v→2 (3), 5→4 (2) and o→0 (2). These pairs are visually similar by construction (for example the digit 0 and the letter o), which suggests that a single static hand shape is intrinsically ambiguous for part of the label set. The two smallest classes (a, 39 samples, and 0, 40) are among the six weakest, but so are o (117 samples) and w (143), so class imbalance alone does not explain the errors.

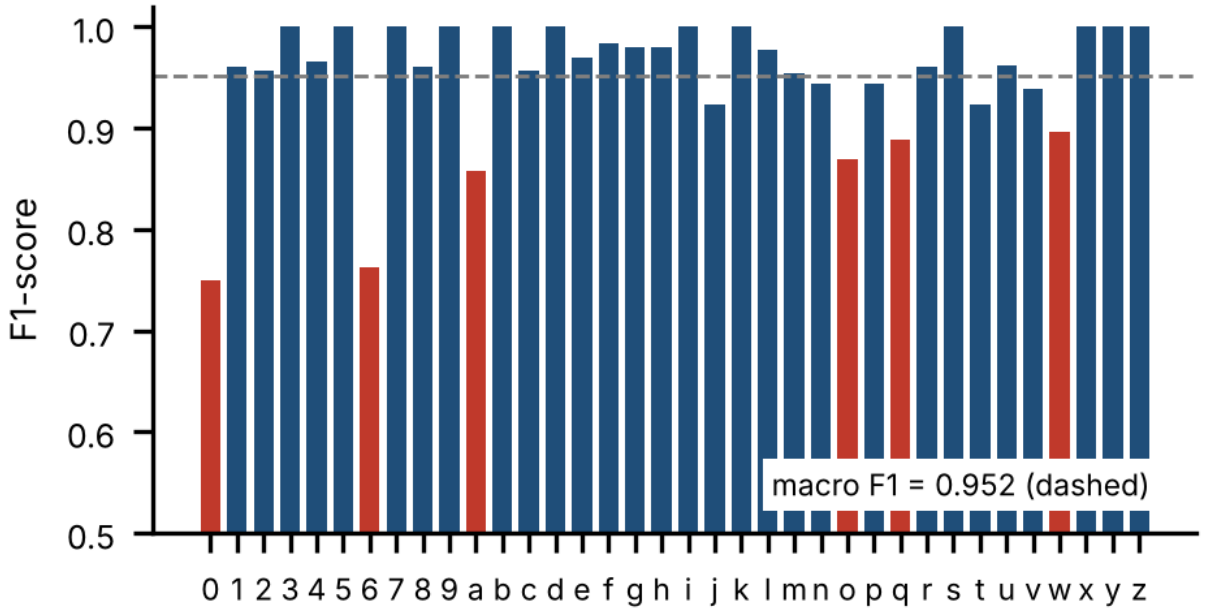


Figure 3. Per-class F1-score of the deployed random forest on the 653-row hold-out set. Classes with F1 below 0.90 are highlighted.

7.4 Selective prediction and latency

Applying a 75% probability cut-off offline to the deployed RF retains 75.65% of hold-out predictions, and all retained predictions are correct on this hold-out set. This is a selective analysis on the same row-level partition, not a calibrated estimate of live performance.

Table 3 lists classifier-only latency. All three models respond in a few milliseconds per sample, so classification is unlikely to be the bottleneck of the live pipeline; end-to-end frame rate, which includes MediaPipe inference, encoding and network transport, was not measured. The saved RF model occupies 16.4 MB.

Table 3. Classifier-only single-row inference latency (3,265 timed calls).

Model	Median (ms)	Mean (ms)	P95 (ms)	Scaler
RF (100 trees)	2.76	3.06	4.85	No
k-NN (k = 5)	1.91	1.83	2.56	Yes
RBF-SVM (C = 1)	1.05	1.23	2.26	Yes

P95, 95th percentile; Scaler, whether a StandardScaler transform is included.

8 Limitations

Row-level evaluation. Samples are frames captured about 150 ms apart, so consecutive frames are nearly identical, and a random row-level split can place near-duplicates in both the training and hold-out partitions. The reported accuracies therefore describe within-collection recognition and are likely optimistic for new signers, cameras, and backgrounds. The dataset has no signer or session identifiers, so a group-wise split was not possible.

Static gestures only. Gestures that depend on motion cannot be represented by a single-frame feature vector.

Preprocessing consistency. Training features come from square crops and live features from full frames; both apply an aspect correction, but equivalence of the two paths on real video was not verified empirically.

Temporal logic. The effect of the voting filter and hold time on typing speed and error rate was not measured, and no user study was conducted.

Deployment. Cloud WebRTC performance depends on the viewer's network and device and was not benchmarked.

9 Conclusion and future work

A browser-based gesture-to-text system was built from MediaPipe landmarks, a compact normalized feature vector, and a Random Forest classifier, reaching 95.9% hold-out accuracy over 36 classes with millisecond-level classifier latency. Tuned Random Forest, *k*-NN and RBF-SVM models differ by less than the fold-to-fold variation, and the main remaining errors involve visually similar classes. Future work should (i) collect data from multiple signers and sessions and report group-wise accuracy, (ii) log labelled live video to evaluate the temporal logic and end-to-end latency, (iii) calibrate class probabilities and enforce a confidence threshold in the live decision layer, and (iv) add motion-aware models for gestures that a single frame cannot capture.

Acknowledgment

The authors thank Prof. Viddulata Patil for guidance throughout the project.

References

[1] Rautaray SS, Agrawal A. Vision based hand gesture recognition for human computer interaction: A survey. *Artif. Intell. Rev*. 2015;43(1):1–54.

[2] Pugeault N, Bowden R. Spelling it out: real-time ASL fingerspelling recognition. In: *Proc*. IEEE Int. Conf. Computer Vision Workshops (ICCV Workshops). 2011:1114–1119.

[3] Zhang F, Bazarevsky V, Vakunov A, Tkachenka A, Sung G, Chang CL, Grundmann M. MediaPipe Hands: on-device real-time hand tracking. *arXiv*:2006.10214; 2020.

[4] Lugaresi C, Tang J, Nash H, McClanahan C, Uboweja E, Hays M, et al. MediaPipe: a framework for building perception pipelines. *arXiv*:1906.08172; 2019.

[5] Breiman L. Random forests. *Mach. Learn*. 2001;45(1):5–32.

[6] Cortes C, Vapnik V. Support-vector networks. *Mach. Learn*. 1995;20(3):273–297.

[7] Cover T, Hart P. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory*. 1967;13(1):21–27.

[8] Pedregosa F, et al. Scikit-learn: machine learning in Python. *J. Mach. Learn. Res*. 2011;12:2825–2830.

[9] Bradski G. *The OpenCV library*. Dr. Dobb's Journal of Software Tools. 2000.