

Article

Beyond Perfect Scores: Leakage-Aware Multi-Corpus Evaluation of Low-Resolution ToF and Depth Gesture Recognition

Burak Aggul 

Department of Database, Network Design and Management, Plato Vocational School, Istanbul Topkapi University, Istanbul 34020, Turkey; burakaggul@topkapi.edu.tr

Abstract

Touchless interfaces for embedded systems require models that are both compact and evaluated under protocols that reflect realistic generalization. Public low-resolution gesture datasets can obscure this requirement because repeated samples, temporal neighbors, or subject identity may leak across train and test partitions. This study presents a leakage-aware multi-corpus benchmark in which four public corpora are evaluated within their own dataset-specific partitions after conversion to a common 8×8 depth representation when needed: a public IEEE DataPort ToF CSV with 8400 frames from four gestures, DS.GestureRecognition_TOF.1.0 with 3109 frames extracted from 40 low-resolution ToF sequences and three gestures, the depth component of the public Kinect+Leap dataset with 1400 samples from 14 subjects and 10 gesture classes, and the public Senz3D depth archive with 1320 samples from 4 subjects and 11 gesture classes. No blurred RGB image dataset, deblurring target, or synthetic blur augmentation is produced in the experiments; the only spatial reduction applied to external depth corpora is deterministic block averaging for a shared low-resolution depth input. The term multi-corpus is used here to clarify scope: the experiments do not train on one dataset and test on another, but instead compare optimistic random splits against structure-aware alternatives matched to each corpus: chronological block-aware, sequence-aware, and subject-wise evaluation. On the original IEEE ToF benchmark, the best macro F1-score decreases from 1.0000 under random splitting to 0.8494 under block-aware evaluation. On DS.GestureRecognition_TOF.1.0, random splitting again reaches 1.0000, whereas the best sequence-aware result falls to 0.6697. On the downsampled Kinect+Leap depth benchmark, random frame classification reaches 0.7639 macro F1, while subject-wise evaluation drops to 0.0566. On Senz3D reduced to 8×8 , random frame classification reaches 0.9068 macro F1, while subject-wise evaluation drops to 0.0717. Device-side validation on an NVIDIA Jetson Nano Developer Kit preserved a macro F1-score of 0.8667 for the selected GRU on the primary IEEE ToF dataset with 14.48 ms/window steady-state CPU latency. The results show that low-resolution gesture recognition conclusions are strongly protocol- and corpus-dependent: compact temporal models remain viable for genuine ToF streams, but naive random splitting can substantially overstate generalization, and subject-wise recognition after aggressive 8×8 conversion remains difficult for the evaluated datasets and architectures.



Academic Editor: Juan Gabriel Avina-Cervantes

Received: 20 May 2026

Revised: 27 July 2026

Accepted: 30 July 2026

Published: 1 August 2026

Copyright: © 2026 by the author.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons](#)

[Attribution \(CC BY\)](#) license.

Keywords: gesture recognition; time-of-flight sensor; depth sensing; lightweight deep learning; temporal modeling; multi-corpus benchmark; data leakage

1. Introduction

Touchless human–machine interaction is increasingly relevant in embedded systems, industrial interfaces, healthcare devices, and autonomous platforms because it reduces mechanical wear, enables hygienic operation, and supports more natural user interaction [1,2]. Many gesture recognition systems rely on RGB cameras or higher resolution depth sensors, but these solutions can increase privacy concerns, storage requirements, and computational demand. Time-of-Flight sensing provides an attractive alternative because it directly measures depth, is less sensitive to color and texture variation, and can be integrated into small low-power devices [3–6].

The practical challenge is that low-cost ToF modules often deliver low-resolution spatial grids. In the present study, each frame contains only 64 depth values arranged on an 8×8 lattice. This resolution is attractive for edge deployment, but it also places more burden on temporal modeling because the spatial signature of a single frame is limited. Gesture recognition on low-resolution depth sequences therefore depends not only on model accuracy, but also on whether the evaluation protocol reflects realistic generalization behavior.

This issue is especially important for public datasets that were acquired in one or a few contiguous recording sessions. If train and test samples are randomly mixed, temporally adjacent frames from the same gesture execution can enter both subsets and produce misleadingly high scores. Such evaluation leakage is easy to overlook when the dataset is balanced and classification accuracy appears strong. The broader machine learning literature has similarly warned that data leakage, non-independent validation, and model-selection bias can lead to optimistic estimates when evaluation protocols do not match the intended prediction setting [7–11]. For a benchmark intended to inform deployable model design, this problem is a substantive methodological issue.

This paper addresses that gap through a reproducible and deployment-oriented benchmark spanning four public datasets: the original IEEE DataPort ToF CSV [12], DS.GestureRecognition_TOF.1.0 [13], the public Kinect+Leap depth dataset [14], and the public Senz3D depth dataset [15]. The phrase *multi-corpus evaluation* is used instead of cross-dataset generalization because the experiments are not designed as train-on-one-corpus/test-on-another-corpus domain-transfer trials. Each corpus is evaluated within its own metadata-aware partitioning scheme. The scientific question is therefore narrower but more reproducible: how much the apparent performance of low-resolution depth gesture recognition changes when the split respects the temporal, sequence, or subject structure available in each public corpus.

The study asks three focused questions: first, whether lightweight temporal models outperform frame-based baselines on genuine ultra-low-resolution ToF streams; second, how much naive random splitting overestimates performance relative to structure-aware protocols matched to each dataset; and third, whether aggressive 8×8 compression remains credible when transferred to subject-labeled public depth benchmarks. The analysis identifies how protocol design, spatial compression, and metadata availability change the interpretation of the same recognition task. The main contributions are as follows:

- We extend the benchmark from a single public CSV file to a multi-corpus evaluation under a common 8×8 input representation, including two real ToF sources and two subject-labeled depth benchmarks.
- We implement structure-aware evaluation matched to available metadata: chronological block-aware splitting for the IEEE ToF CSV, sequence-aware splitting for DS.GestureRecognition_TOF.1.0, and subject-wise splitting for Kinect+Leap and Senz3D depth.

- We show that naive random splitting inflates the best macro F1-score by 0.1506 on the IEEE ToF benchmark, by 0.3303 on DS.GestureRecognition_TOF.1.0, by 0.7073 on the 8×8 Kinect+Leap depth benchmark, and by 0.8351 on the 8×8 Senz3D depth benchmark.
- We report when lightweight temporal models remain useful on genuine low-resolution ToF streams and when aggressive 8×8 depth compression does not preserve subject-generalizable discrimination for the evaluated public corpora.
- We explicitly separate deterministic 8×8 depth normalization from artificial image blurring, and we report statistical confidence intervals, seed-level variance, paired tests, a limited GRU sensitivity check, and a device-side steady-state CPU inference validation on an NVIDIA Jetson Nano Developer Kit.

The paper is organized as follows. Section 2 reviews relevant literature. Section 3 describes the dataset audit, preprocessing pipeline, evaluation protocols, and models. Section 4 presents benchmark results. Sections 5 and 6 discuss the findings and their limits, and Section 7 concludes the paper.

2. Related Work

Vision-based hand gesture recognition has been widely studied for more than a decade, with applications ranging from consumer interaction to assistive and industrial systems [1,2]. Early systems often used RGB imagery, handcrafted descriptors, or instrumented gloves, while later work increasingly adopted depth sensing and deep neural networks. Depth-based methods are attractive because they reduce the influence of color, illumination, and background texture while preserving a compact representation of hand motion.

Recent surveys and benchmark-oriented studies continue to emphasize the role of evaluation design, sensor modality, hardware constraints, and dataset choice in hand gesture recognition. Fertl et al. review device-free edge gesture-recognition systems across sensing technologies, algorithms, and processing hardware [16]. Emporio et al. review continuous hand gesture recognition benchmarks and report that publicly available datasets and evaluation procedures remain limited and insufficiently standardized [17]. Yaseen et al. further show that benchmark dataset properties and lightweight model selection jointly affect dynamic gesture recognition performance [18]. These recent studies support the need for protocol-aware benchmarking rather than accuracy reporting under a single split.

Time-of-Flight cameras are a particularly practical depth modality for embedded interaction because they directly estimate distance rather than reconstruct it from stereo disparity [3–5]. Recent compact ToF modules have made short-range depth sensing feasible on development boards and microcontroller-oriented platforms [6]. However, many embedded ToF devices operate at much lower spatial resolution than laboratory depth cameras. This makes the problem structurally different from high-resolution video understanding: the model must extract temporal evidence from sparse spatial measurements.

Deep sequence models are appropriate for this setting. Three-dimensional convolutional networks, temporal convolutional networks, and recurrent architectures have shown strong performance in gesture recognition, action segmentation, and sequence analysis [19–21]. More broadly, recent work on time-series classification has shown that compact recurrent and convolutional models can outperform simpler baselines when temporal structure is informative [22–25]. At the same time, lightweight neural designs, compression methods, quantization-aware inference, and mobile/embedded inference systems illustrate the importance of balancing predictive power with deployment efficiency [26–32].

Recent attention-based video models and mobile vision transformers provide another relevant baseline family. TimeSformer demonstrates that space–time self-attention can model video patches directly [33], and MobileViT shows how convolutional inductive bias

and transformer-style global processing can be combined for mobile vision [34]. These architectures motivate future work on attention for low-resolution gesture streams. Large language models adapted for time-series reasoning are also an emerging research direction, but they are not a primary baseline for these small 8×8 public corpora without task-specific pretraining or substantially larger sequence collections. The present benchmark uses compact recurrent and convolutional baselines because the public ToF corpora are small, the input grid contains only 64 depth cells, and the target deployment discussion focuses on CPU-side embedded inference rather than large-scale pretraining or large video backbones.

Several nearby sensing tasks further contextualize the scope of the present work. Kinect+Leap-based gesture recognition has been evaluated with explicit sensor fusion and handcrafted features [35], driver micro-gesture recognition has been studied with richer RGB, depth, and infrared modalities and mobile 3D CNNs [36], low-resolution thermal gesture datasets have been evaluated using TCN and spiking neural approaches [37,38], and radar-based hand gesture work has explored off-the-shelf array radar pipelines [39]. These studies show that broader gesture recognition is not restricted to ToF depth grids. They are not directly comparable to the present 8×8 ToF/depth benchmark, but they define useful future directions for testing whether leakage-aware protocols transfer to radar, thermal, sign-language, and multi-modal settings.

Despite this progress, the literature still contains many studies in which evaluation protocols are not sufficiently separated from acquisition order, subject identity, or repeated sequences. For small public datasets, this can produce inflated performance estimates that are difficult to reproduce outside the original split. The present work therefore treats evaluation design as part of the contribution rather than as a fixed background assumption, following the broader recommendation that structured data require validation splits aligned with the intended generalization target [7,11].

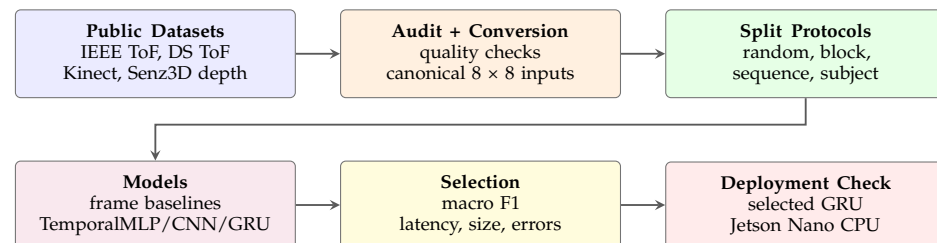
Theoretical Motivation for Leakage-Aware Low-Resolution Evaluation

The benchmark design follows a simple generalization argument. Let x_i denote a depth frame or temporal window, y_i its gesture label, and g_i the acquisition group that may encode contiguous time block, public sequence identity, or subject identity. A naive random split estimates performance on test samples drawn from nearly the same empirical groups as the training data. When frames from the same recording block, sequence, or subject are strongly correlated, the test risk estimated by random splitting can be lower than the risk for a new block, sequence, or subject. In that setting, the split itself changes the target of evaluation: random splitting asks whether the model can interpolate among repeated samples, whereas block-, sequence-, or subject-aware splitting asks whether the model can generalize across the available acquisition structure.

This distinction is especially important for ultra-low-resolution depth inputs. Reducing a depth map to an 8×8 grid can be viewed as applying a fixed many-to-one averaging operator A , so that the model receives $z_i = Ax_i$ rather than the original high-resolution observation. Such an operator suppresses local hand-shape and finger-contour information while preserving coarse depth mass and motion trends. Consequently, a single 8×8 frame may be insufficient for subject-independent recognition, whereas a temporal window $z_{t:t+w-1}$ can recover dynamic evidence when the source data are genuine low-resolution ToF streams. This theoretical view explains why the paper evaluates both frame-level and temporal models, why it separates native ToF streams from downsampled depth archives, and why the conclusions are tied to metadata-aware validation protocols rather than to random-split accuracy alone.

3. Materials and Methods

The overall workflow is summarized in Figure 1. The pipeline starts from public ToF or depth archives, audits the source format, converts external datasets to a canonical 8×8 representation when needed, constructs leakage-controlled frame and temporal inputs, compares models under optimistic and structure-aware protocols, and finally validates the selected GRU checkpoint under deployment-oriented CPU timing on the primary IEEE ToF benchmark.



All metrics are computed on held-out test partitions; hardware timing excludes sensor I/O and rendering.

Figure 1. Workflow of the leakage-aware multi-corpus gesture benchmark. The diagram summarizes the progression from public ToF and depth archives through auditing, canonical 8×8 conversion, split design, model comparison, metric reporting, and Jetson Nano CPU validation on the primary IEEE ToF benchmark.

3.1. Dataset Description and Audit

The study uses four public datasets. The primary benchmark remains the public IEEE DataPort CSV titled *Time-of-Flight (ToF) Gesture Recognition Dataset* [12]. External validation adds DS.GestureRecognition_TOF.1.0 from Zenodo [13], the public Kinect+Leap dataset released by the University of Padova [14], and the public Sens3D depth dataset released by the same group [15]. All datasets were converted to a canonical 8×8 depth representation with 64 input values per sample when necessary. The 8×8 target was selected because it is native to the primary IEEE ToF CSV and to current multizone embedded ToF devices such as 64-zone sensors, making it the common denominator for privacy-preserving and bandwidth-limited interaction. The intent is therefore not to claim that 8×8 is optimal for all depth data, but to test what remains recognizable under an intentionally low-resolution constraint. The IEEE CSV already matches this format. The DS archive stores $8 \times 8 \times 3$ tensors, which were collapsed to one 8×8 map by averaging the channel axis. The Kinect+Leap archive stores raw 640×480 depth maps, and the Sens3D archive stores raw 320×240 depth maps; both were reduced to 8×8 by non-overlapping block averaging. Block averaging was chosen because it is deterministic, reproducible, parameter-free, and does not introduce trainable reconstruction or super-resolution assumptions. Importantly, no artificial blurred image generation, image deblurring task, or synthetic blur augmentation is used for training, validation, or testing. The external Kinect+Leap and Sens3D samples remain real captured depth maps; they are only spatially averaged to match the low-resolution ToF premise. Intermediate representations such as 16×16 would likely preserve more hand-shape detail for Kinect+Leap and Sens3D, but they would no longer match the native resolution of the primary ToF benchmark and would change the deployment premise. Because each Kinect+Leap and Sens3D repetition is provided as a single depth acquisition rather than a temporal sequence, those external benchmarks are frame-based only.

Table 1 summarizes the dataset suite. The IEEE ToF CSV remains the only dataset with continuous time metadata and contiguous acquisition order, so it was audited in more detail for missing values, duplicates, depth ranges, and temporal structure. That audit found 0 missing values, 0 duplicated rows, and a single resolution value (8×8).

The depth values span 0 to 3295 mm, with a global mean of 1492.564 mm and a median of 1969 mm. The total recorded duration, measured from the provided time column, is 262.469 s. An important audit result is that the four classes appear as four contiguous blocks in CSV order: Approaching (indices 0–2099), Swipe (2100–4199), Hold (4200–6299), and Tap (6300–8399). This ordering suggests a single acquisition session or a small number of contiguous recording blocks.

Figure 2 shows class-wise mean frames for the primary IEEE ToF benchmark, while Figure 3 illustrates its frame-to-frame motion energy over time.

Table 1. Summary of the public dataset suite used in this study after conversion to a common 8×8 representation when needed.

Dataset	Samples	Cls.	Subj.	Native	8×8 /Split
IEEE DataPort ToF CSV	8400 fr.	4	n/r	8×8 ToF CSV	Native; block
DS.GestureRecognition_TOF.1.0	3109 fr./40 seq.	3	1	$8 \times 8 \times 3$ ToF	Mean ch.; seq.
Kinect+Leap depth	1400 samp.	10	14	640×480 depth	Block avg.; subj.
Senz3D depth	1320 samp.	11	4	320×240 depth	Block avg.; subj.

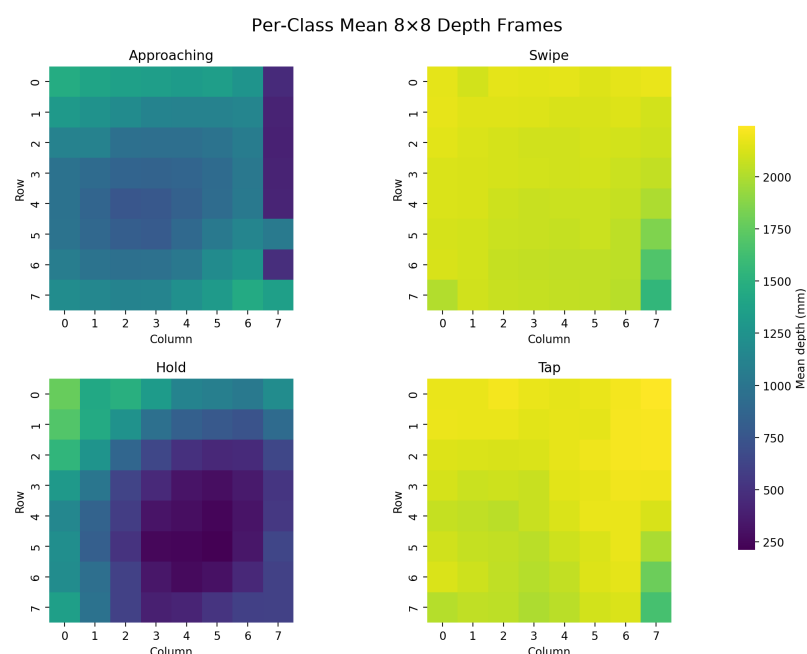


Figure 2. Per-class mean depth frames computed from the public ToF dataset. Distinct coarse spatial patterns remain visible despite the 8×8 resolution, which motivates temporal modeling.

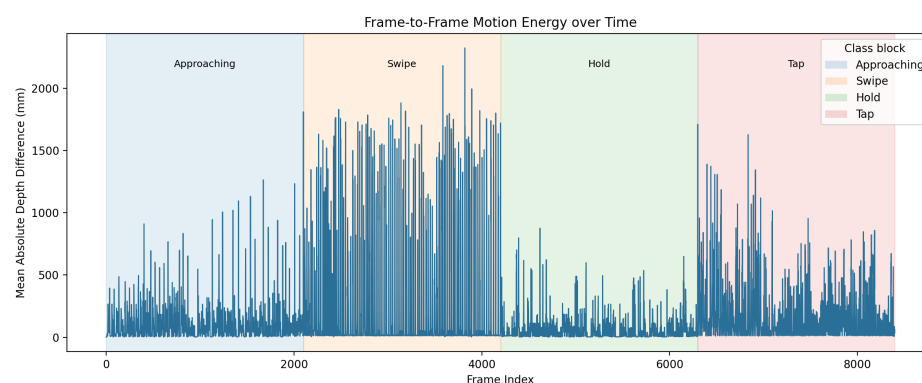


Figure 3. Frame-to-frame motion energy across the acquisition sequence. The shaded bands denote the four contiguous gesture blocks and motivate leakage-aware evaluation rather than naive random splitting.

3.2. Preprocessing and Temporal Window Construction

Only the 64 depth channels (d_0 to d_{63}) were used as model input. Timestamp and time columns were retained for auditing and ordering, but not used as predictive features. A channel-wise standardization step was applied when configured, and the mean and standard deviation were estimated exclusively from the training split of each protocol to avoid information leakage.

Two input modes were considered. The first mode used a single frame as input for classical frame-based baselines. The second mode used temporal windows for datasets that provide sequence structure. For the primary IEEE ToF benchmark, windows of length 8, 16, and 32 were evaluated with stride 2. For DS.GestureRecognition_TOF.1.0, shorter windows of 4, 8, and 16 were used because the public sequences are shorter. Kinect+Leap and Senz3D are distributed as single depth samples, so only frame-based baselines were evaluated there. No window was allowed to cross a class boundary or a sequence boundary.

The stride values were selected as conservative overlap settings rather than tuned hyperparameters. Stride 2 on the primary IEEE ToF benchmark gives enough temporal samples for training while reducing near-duplicate windows compared with stride 1; stride 1 on DS.GestureRecognition_TOF.1.0 was used because its public sequences are shorter. A limited GRU sensitivity check in Section 4 reports how stride 1 and stride 4 affect the primary block-aware result.

The resulting window counts were balanced by construction. For each class, 1047 valid windows were obtained with length 8, 1043 with length 16, and 1035 with length 32, yielding totals of 4188, 4172, and 4140 windows, respectively. These counts are summarized in Table 2.

Table 2. Number of valid temporal windows per class. Windows were generated with stride 2 and were not allowed to cross class boundaries.

Class	Window = 8	Window = 16	Window = 32
Approaching	1047	1043	1035
Swipe	1047	1043	1035
Hold	1047	1043	1035
Tap	1047	1043	1035
Total	4188	4172	4140

3.3. Evaluation Protocols

Each dataset was evaluated with an optimistic random protocol and one structure-aware protocol matched to the metadata actually available in that corpus.

Protocol A: optimistic random split. Frames or windows were randomly divided into training, validation, and test subsets using stratification by class. The primary IEEE ToF benchmark used a 70%/15%/15% split, the external DS and Kinect+Leap experiments used 60%/20%/20%, and the Senz3D experiment used 50%/25%/25% so that both validation and test partitions retained held-out subjects in the subject-wise protocol. This protocol preserves label balance but ignores the temporal organization or subject structure of the acquisition stream.

Protocol B: structure-aware split. For the IEEE ToF CSV, a chronological split was performed inside each contiguous class block. For DS.GestureRecognition_TOF.1.0, complete public sequences were grouped by `sequence_id` so that no sequence contributed windows to more than one partition. For Kinect+Leap and Senz3D, complete subjects were grouped by `subject_id` so that no person contributed samples to more than one partition. These protocols are not identical, but they serve the same methodological purpose: they

prevent train/test contamination through repeated temporal context or subject identity and therefore provide a more defensible estimate of generalization than naive random mixing.

3.4. Models and Training Setup

Three frame-based baselines and three temporal models were evaluated.

Frame-based models. A majority-class predictor was used as a lower-bound baseline. Logistic regression provided a linear discriminative model, and random forest provided a stronger non-linear frame-level baseline [40].

Temporal models. A temporal multilayer perceptron consumed flattened windows, a compact one-dimensional convolutional network processed the temporal sequence along the window dimension, and a gated recurrent unit modeled recurrent temporal dependencies [41,42]. These models were intentionally lightweight and CPU-oriented.

Table 3 lists the neural architectures used in the benchmark. The architectures were fixed before the main comparison so that the paper compares evaluation protocols and input conditions rather than performing a broad neural architecture search.

Table 3. Neural architecture and training settings used in the main benchmark. *C* denotes the number of classes in the corresponding dataset.

Model	Architecture	Dropout	Training Settings
TemporalMLP	Flattened window → dense 256 → dense 128 → dense <i>C</i>	0.2 after each hidden layer	Adam, learning rate 10^{-3} , weight decay 10^{-4} , batch size 64 on IEEE ToF and 32 on DS ToF
TemporalCNN	Conv1D 64 → 32, kernel 3 → Conv1D 32 → 64, kernel 3 → adaptive average pooling → dense <i>C</i>	0.2 before output layer	Same optimizer settings; evaluated with windows 8/16/32 on IEEE ToF and 4/8/16 on DS ToF
GRU	One GRU layer with 64 hidden units → final hidden state → dense <i>C</i>	0.2 before output layer	Same optimizer settings; maximum 35 epochs and patience 7 on IEEE ToF, maximum 25 epochs and patience 5 on DS ToF

All neural models were trained with Adam [43], learning rate 10^{-3} , weight decay 10^{-4} , dropout 0.2 [44], and early stopping based on validation macro F1-score. The primary IEEE ToF benchmark used batch size 64, a maximum of 35 epochs, and patience 7. The DS external validation used the same optimizer settings but a maximum of 25 epochs because the public sequence set is smaller. Each neural experiment was repeated with three random seeds (7, 17, and 27); the reported results use mean and standard deviation across seeds. The implementation used Python, scikit-learn, and PyTorch [45,46]. To assess whether the selected GRU configuration was unusually sensitive to a single setting, a limited one-factor sensitivity check was also run on the primary block-aware GRU with 32-frame windows, varying hidden size, number of layers, dropout, learning rate, batch size, stride, and optimizer family.

3.5. Metrics and Deployment-Oriented Measurements

For every model and protocol, we recorded accuracy, macro precision, macro recall, macro F1-score, and class-wise precision, recall, and F1-score. In addition, we measured average CPU inference time per sample or window, model parameter count for neural models, artifact size on disk, and training time. Macro F1 was treated as the main selection metric because the study is formulated as a balanced multi-class benchmark and because precision–recall-derived measures are useful when class-wise behavior matters beyond

overall accuracy [47]. For the three-seed neural comparisons on the primary benchmark, we additionally computed sample variance, t-distribution 95% confidence intervals, and paired t-tests over matched seeds. The null hypothesis for each paired test was that the mean seed-level macro-F1 difference between the selected GRU with 32-frame windows and the comparison model was zero; the alternative hypothesis was a non-zero mean difference, with $\alpha = 0.05$. Because only three seeds were available and the tests are exploratory, these inferential statistics are reported as a small-sample robustness check rather than as definitive proof of model superiority. No multiple-comparison correction was applied, so significant values are interpreted cautiously.

To ground the deployment discussion in actual embedded hardware, the selected block-aware GRU checkpoint was additionally reloaded on an NVIDIA Jetson Nano Developer Kit and evaluated over the full 624-window test split using prepared windows generated by the offline pipeline. For that device-side validation, we recorded checkpoint macro F1, bulk latency, streaming latency, throughput, the 62.5 ms decision interval implied by a stride of two at approximately 32 Hz, and the 1000 ms initial window-fill latency implied by a 32-frame context. The archived remote measurement used batch size 64 for bulk inference, 10 repeated timing runs, and 3 warm-up single-window inferences before streaming timing. Bulk inference used pre-standardized windows in mini-batches, whereas streaming inference processed raw prepared windows one at a time and included per-window standardization, NumPy-to-PyTorch tensor construction, transfer to the selected PyTorch CPU device, and the forward pass under `torch.no_grad()`. Sensor acquisition, frame buffering before a full window exists, rendering, and downstream application logic were excluded. The archived Jetson report captured the platform, Python version, PyTorch version, CUDA availability, and CPU execution path; CPU governor and Jetson power-mode metadata were not programmatically captured in that already-run remote measurement and are therefore reported as unavailable for that run.

4. Results

4.1. Audit Results and Temporal Structure

The audit confirmed that the dataset is balanced but temporally structured. Although the class distribution is uniform, the order of acquisition is not interleaved. Instead, each gesture occupies one contiguous segment. Consequently, neighboring frames within the same block are highly correlated and should not be randomly scattered across training and test subsets when the goal is to estimate realistic generalization to unseen temporal segments.

The class-wise mean frames in Figure 2 show that the gestures retain distinct coarse spatial depth patterns despite the limited spatial resolution. The motion energy profile in Figure 3 further indicates that the dynamic gestures are temporally distinguishable, with Swipe showing more sustained transitions than Hold. These observations support the use of temporal windows rather than single-frame classification alone.

4.2. Main Benchmark Results

Table 4 summarizes the strongest frame-based and temporal results under both protocols. Neural temporal results are reported as mean $\pm$ standard deviation over seeds 7, 17, and 27; the frame-based baselines were trained once, so their cells are explicitly marked as single-run values. The protocol effect is substantial. Under the optimistic random split, the best frame-based model (random forest) reaches a macro F1-score of 0.8148, whereas the best temporal model (TemporalCNN with 32-frame windows) reaches 1.0000. Under the chronological block-aware protocol, the strongest frame model remains random forest, but its macro F1-score decreases to 0.6979. The best temporal model under the same

conservative protocol is the GRU with 32-frame windows, which reaches 0.8494 ± 0.0224 macro F1.

The difference between the best random-split and block-aware temporal results is 0.1506 macro F1 points. This is the central empirical finding of the study: the public dataset is informative, but random splitting materially overestimates achievable performance. At the same time, temporal modeling remains beneficial even after leakage-aware evaluation. Relative to the strongest frame baseline, the GRU with 32-frame windows improves block-aware macro F1 by 0.1515 points.

Table 4. Best frame-based and temporal results under the optimistic and leakage-aware protocols, with macro F1 as the primary comparison metric. Neural temporal values are mean $\pm$ standard deviation over seeds 7, 17, and 27. Frame baselines were single runs, so no cross-seed standard deviation is reported.

Protocol	Best Frame Model	Frame Macro F1	Best Temporal Model	Temporal Macro F1
Random split	Random forest	0.8148 (single run)	TemporalCNN ($w = 32$)	1.0000 ± 0.0000
Block-aware split	Random forest	0.6979 (single run)	GRU ($w = 32$)	0.8494 ± 0.0224

Table 5 adds small-sample statistical context for the primary block-aware neural ranking, including sample variance, confidence intervals, and paired hypothesis tests. The GRU with 32-frame windows has the highest mean macro F1, but the paired seed-level difference relative to the 32-frame TemporalMLP is not statistically significant with only three matched seeds ($p = 0.461$). The difference is clearer relative to the 32-frame TemporalCNN ($p = 0.036$) and to the 16-frame GRU ($p = 0.008$). Therefore, the manuscript interprets the 32-frame GRU as the best observed conservative model and a strong deployment candidate, not as a universally superior architecture.

Table 5. Small-sample statistical check for primary block-aware neural results. Confidence intervals are t-based 95% intervals over seeds 7, 17, and 27. Paired tests compare each row with the GRU, $w = 32$, over matched seeds.

Model	Mean [95% CI]	Variance	GRU32-Difference [95% CI]	Paired p
GRU, $w = 32$	0.8494 [0.7938, 0.9050]	0.00050	Reference	—
TemporalMLP, $w = 32$	0.8345 [0.8176, 0.8515]	0.00005	0.0148 [−0.0557, 0.0854]	0.461
TemporalCNN, $w = 32$	0.7846 [0.7708, 0.7983]	0.00003	0.0648 [0.0106, 0.1190]	0.036
GRU, $w = 16$	0.7912 [0.7568, 0.8256]	0.00019	0.0582 [0.0356, 0.0807]	0.008

The variance column provides a direct measure of seed-to-seed dispersion and complements the confidence intervals. The results support two bounded claims: the protocol effect is large and consistent across corpora, and the 32-frame GRU is the best observed conservative model on the primary benchmark. They do not support an unrestricted claim that the GRU architecture would dominate all compact sequence models under broader hyperparameter searches or larger subject-independent datasets.

Table 6 reports the limited one-factor GRU sensitivity check requested for model-selection transparency. The check used the primary IEEE ToF benchmark, the block-aware split, 32-frame windows, and seed 27. It was not used to reselect the final benchmark model, because the main paper compares fixed compact architectures over three seeds. Nevertheless, the results show that the qualitative conclusion is not dependent on a single dropout, batch-size, or optimizer choice. Larger hidden states and a second recurrent layer can improve a single run, but they also increase parameters and were not adopted as new main baselines without a full multi-seed sweep.

Table 6. Limited single-seed sensitivity check for the primary block-aware GRU with 32-frame windows. Values are macro F1 on the held-out test partition.

Factor	Settings and Observed Macro F1	Parameter Implication
Hidden size	32: 0.8503; 64 reference: 0.8265; 128: 0.8591	9.5k, 25.2k, and 75.0k parameters
Number of GRU layers	1 layer: 0.8265; 2 layers: 0.8543	25.2k versus 50.2k parameters
Dropout	0.0: 0.8276; 0.2 reference: 0.8265; 0.4: 0.8260	No parameter change
Learning rate	3×10^{-4} : 0.8144; 10^{-3} reference: 0.8265; 3×10^{-3} : 0.8154	No parameter change
Batch size	32: 0.8494; 64 reference: 0.8265; 128: 0.8282	No parameter change
Window stride	1: 0.8053; 2 reference: 0.8265; 4: 0.8116	Changes window count and overlap
Optimizer	Adam: 0.8265; AdamW: 0.8170	No parameter change

Table 7 interprets the sensitivity results at the level of functional blocks and interaction parameters. The goal is not to replace the fixed three-seed benchmark with a larger architecture search, but to make clear which parts of the pipeline affect the reported conclusions. Temporal context length interacts with the recurrent memory block because longer windows provide more motion evidence but also increase initial buffering. Stride interacts with leakage risk and training-set redundancy because overlapping windows can create near-duplicates. Hidden size, depth, dropout, learning rate, batch size, and optimizer family interact with model capacity and optimization stability. Finally, train-only normalization and metadata-aware partitioning are functional safeguards: without them, the numerical scores would answer a less realistic interpolation question.

Table 7. Functional-block interpretation of ablation and sensitivity findings. The table links interaction parameters to the numerical results and to the claims made in the manuscript.

Block/Parameter	Role in the Pipeline	Interpretation from the Experiments
Temporal context (w)	Controls how much motion evidence is available before a decision.	Random splits saturate quickly, whereas block-aware results improve most clearly at $w = 32$ for the GRU; the benefit is tied to genuine temporal ToF streams.
Window stride	Controls overlap, sample count, and redundancy between neighboring windows.	Stride 2 was used as the main compromise; the single-seed check shows that stride 1 and 4 did not improve the block-aware GRU result.
Recurrent capacity	Hidden size and recurrent depth determine how much temporal state the GRU can encode.	Larger or deeper GRUs can improve individual runs, but the gains were not adopted as main claims without a full multi-seed sweep.
Regularization and optimization	Dropout, learning rate, batch size, and optimizer affect stability and overfitting.	The qualitative result is not explained by one unusually favorable optimizer or dropout setting; the observed range remains near the main GRU result.
Validation safeguards	Train-only normalization and metadata-aware splits define the evaluated generalization target.	These safeguards are central functional blocks because they separate leakage-aware generalization from random-split interpolation.

4.3. Multi-Corpus External Validation

To test whether the protocol effect extends beyond the primary IEEE ToF CSV, the same within-dataset pipeline was applied to DS.GestureRecognition_TOF.1.0 and to 8×8 versions of the public Kinect+Leap and Senz3D depth corpora. Table 8 summarizes the

best optimistic and structure-aware results on each dataset. The pattern is consistent but not uniform across corpora. In both real ToF datasets, random splitting saturates at 1.0000 macro F1, whereas structure-aware evaluation lowers the best score to 0.8494 on the IEEE benchmark and 0.6697 on DS.GestureRecognition_TOF.1.0. The optimism gap is therefore larger on the second ToF dataset (0.3303 versus 0.1506), suggesting that naively mixed train/test windows can be even more misleading when the public archive contains repeated low-resolution sequences from the same user.

The external depth benchmarks further illustrate the limitation. After block-averaging Kinect raw depth maps to 8×8 , random frame-based evaluation still yields 0.7639 macro F1 with random forest, but subject-wise evaluation drops to 0.0566 macro F1 with logistic regression. Senz3D shows a similar pattern: random evaluation reaches 0.9068 macro F1 with random forest, yet subject-wise evaluation drops to 0.0717 macro F1, again with logistic regression. These external inputs are real captured depth samples rather than artificially blurred images; the observed degradation is therefore attributed to low-resolution spatial averaging, missing temporal context, and subject-wise generalization difficulty, not to a synthetic image-blur training or testing protocol. These results do not imply that the original depth data are unusable; rather, they show that aggressive privacy-preserving 8×8 compression and subject-independent generalization are jointly difficult on these corpora. Together, the external benchmarks shift the paper from a single-dataset result to a multi-corpus methodological claim: low-resolution depth recognition quality is strongly conditioned by both the evaluation protocol and the source corpus.

The near-chance subject-wise behavior has three likely causes. First, block averaging from 640×480 or 320×240 to 8×8 removes high-frequency hand-shape, finger-contour, and local pose cues that are useful for 10- and 11-class vocabularies. Second, Kinect+Leap and Senz3D are available here as single depth acquisitions rather than continuous low-resolution temporal streams, so the temporal evidence that helps the ToF benchmarks is absent. Third, subject-wise splitting exposes variation in hand size, camera distance, arm pose, and execution style that random frame splits hide. With only 14 Kinect+Leap subjects and 4 Senz3D subjects, the subject-wise estimates are also necessarily coarse. For these reasons, the results should be read as limited performance of the evaluated 8×8 conversion and model family on these public subject-wise splits, not as proof that all low-resolution depth or all subject-independent gesture recognition is impossible.

Table 8. Best random-split and structure-aware macro F1 across the four public datasets. Kinect+Leap and Senz3D remain frame-based after 8×8 conversion.

Dataset	Best Random Model	Random F1	Best Structured Model	Structured F1	Gap
IEEE ToF CSV	TemporalCNN, $w = 32$	1.0000	GRU, $w = 32$ (block)	0.8494	0.1506
DS ToF 1.0	GRU, $w = 4$	1.0000	Logistic reg. (sequence)	0.6697	0.3303
Kinect+Leap 8×8	Random forest	0.7639	Logistic reg. (subject)	0.0566	0.7073
Senz3D 8×8	Random forest	0.9068	Logistic reg. (subject)	0.0717	0.8351

Figure 4 summarizes the optimism gap across all four datasets. The gap is smallest on the original IEEE ToF benchmark, larger on DS.GestureRecognition_TOF.1.0, and largest on the two subject-wise depth benchmarks. This ordering matters for the paper's claim: the evaluation protocol is not a minor implementation detail, but a dominant factor separating apparently solved random-split tasks from much harder structure-aware generalization settings.

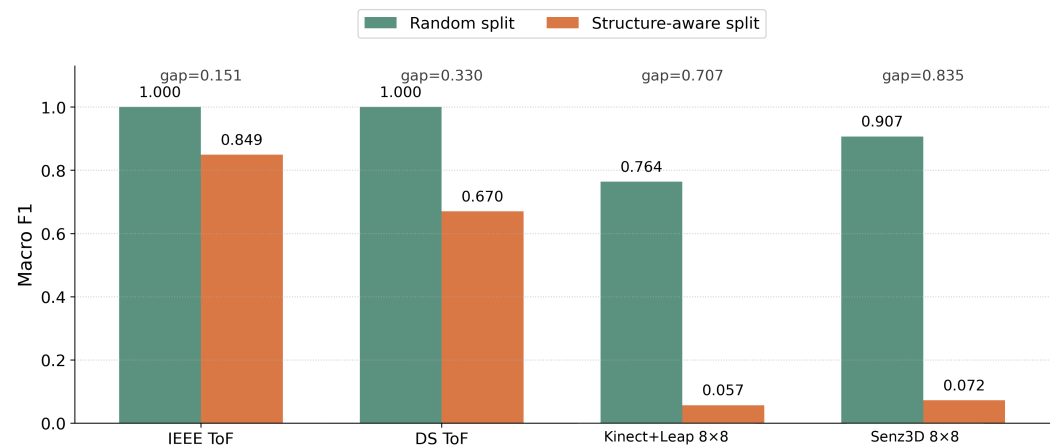


Figure 4. Best random-split and structure-aware macro F1-scores across the four benchmark datasets. The two subject-wise depth benchmarks exhibit the largest optimism gaps after conversion to 8×8 .

Figure 5 shows the best structure-aware confusion matrices for the external benchmarks. DS.GestureRecognition_TOF.1.0 retains a meaningful diagonal despite some confusion between *spin* and *stop*, whereas the two subject-wise depth benchmarks collapse toward a few dominant predicted classes after 8×8 conversion. The visual contrast shows that the multi-corpus analysis is not only a scalar F1 comparison; the error structure itself changes qualitatively between repeated ToF sequences and subject-wise depth evaluation.

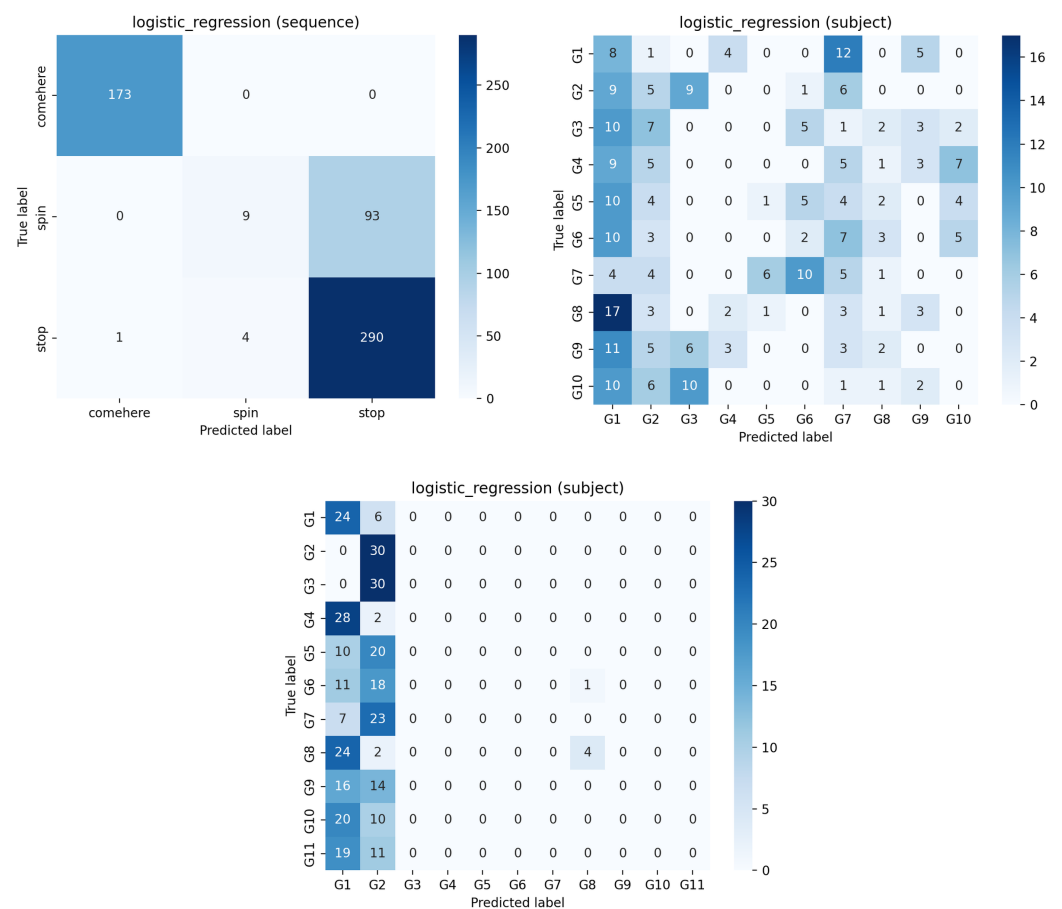


Figure 5. Best structure-aware confusion matrices for the external datasets. DS ToF remains partly separable, whereas Kinect+Leap and Senz3D degrade sharply after 8×8 conversion.

4.4. Comparison with Representative Prior Work

Table 9 compares the scope of the present benchmark with representative gesture-recognition studies. The table is intentionally qualitative rather than a direct accuracy leaderboard. The compared studies use different sensors, gesture vocabularies, subject counts, input resolutions, and validation protocols, so a direct numerical ranking would be structurally confounded. The comparison instead highlights that the present study contributes a leakage-aware, within-dataset, multi-corpus protocol analysis for an intentionally low-resolution 8×8 ToF/depth representation.

Table 9. Representative comparison with related gesture-recognition work. The comparison emphasizes sensing modality and evaluation scope rather than direct score ranking because the datasets and protocols differ.

Study	Input and Task	Evaluation Emphasis	Relation to This Work
Ohn-Bar and Trivedi [2]	Multimodal vision for real-time automotive hand gestures	Real-time interface evaluation	Higher-dimensional sensing; not focused on 8×8 ToF leakage analysis
Marin et al. [35]	Kinect depth and Leap Motion features for hand gestures	Sensor fusion and SVM-based recognition	Related Kinect+Leap source context; not an 8×8 compression benchmark
Molchanov et al. [20]	Dynamic hand gestures with recurrent 3D convolutional networks	Online detection and classification	Strong video/depth baseline family; much richer input than 64-cell ToF grids
Köpüklü et al. [36]	Driver micro gestures with RGB, depth, and infrared modalities	Lightweight 3D CNNs and real-time deployment	Mobile-oriented but multi-modal and not directly comparable to 8×8 ToF/depth inputs
Vandersteegen et al. [37]	Low-resolution thermal gesture recognition	Low-latency dynamic gesture recognition	Closest low-resolution sensing analogue, but thermal rather than ToF/depth
Safa et al. [38]	24×32 low-resolution thermal camera gestures	Resource-efficient spiking neural processing	Demonstrates broader low-resolution sensing directions beyond ToF
Sluyters et al. [39]	Off-the-shelf radar hand gestures	Radar-specific modeling and few-template recognition	Motivates future extension to non-depth low-resolution sensors
Fertl et al. [16]	Device-free hand gesture recognition on edge devices	Survey of sensor technologies, algorithms, and processing hardware	Provides current edge-device context, including hardware and model trade-offs
Emporio et al. [17]	Continuous hand gesture recognition benchmarks	Benchmark taxonomy, metrics, and method review	Supports the need for standardized evaluation procedures
Yaseen et al. [18]	Vision-based dynamic gesture benchmark datasets and lightweight models	Dataset–model interaction and transfer-learning evaluation	Recent benchmark context; higher-resolution vision setting rather than 8×8 ToF/depth
This study	8×8 ToF/depth representations across four public corpora	Random versus metadata-aware within-dataset protocols, statistics, and Jetson CPU timing	Quantifies leakage and compression effects under a shared low-resolution representation

Figure 6 expands the temporal comparison across all evaluated window sizes. The random-split curves saturate quickly, whereas the block-aware protocol shows a more gradual dependence on temporal context. Under the conservative protocol, the 32-frame GRU provides the strongest balance of recognition quality and compactness.

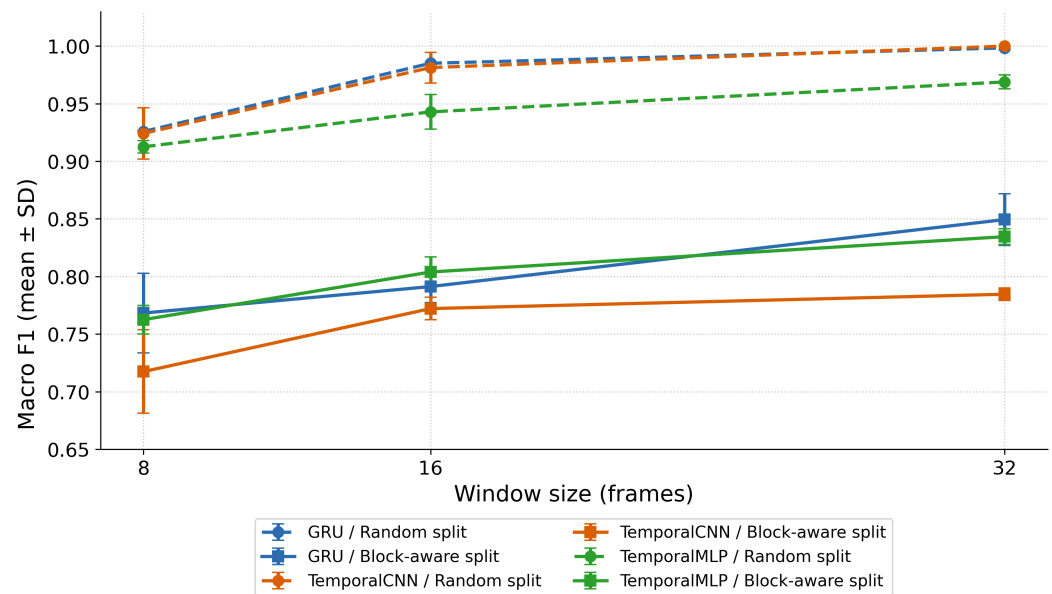


Figure 6. Macro F1 as a function of temporal window size for neural temporal models. Error bars indicate standard deviation across the three neural seeds.

4.5. Accuracy–Efficiency Trade-Offs

Block-aware performance and deployment-oriented costs for representative models, measured in the local x86 CPU benchmarking environment used for the full experiment sweep, are summarized in Table 10. The GRU with 32-frame windows yields the best conservative macro F1-score (0.8494) with 25,220 trainable parameters, an average artifact size of 103.7 kB, and average CPU inference time of 0.079 ms per sample. The TemporalCNN with 32-frame windows is smaller (12,836 parameters, 58.4 kB) and faster (0.046 ms per sample), but its macro F1-score is lower at 0.7846. The temporal multilayer perceptron with 32-frame windows achieves 0.8345 macro F1 and low measured latency, but it is much larger on disk (2.23 MB) and requires 557,956 parameters because it flattens the full window.

These results suggest that the GRU offers the strongest recognition performance without becoming computationally impractical for this scale of input. However, if binary size or parameter count is the leading deployment constraint, the convolutional model remains attractive. Figure 7 visualizes this trade-off across all evaluated models and separates the local x86 sweep from the local and Jetson deployment measurements for the selected checkpoint.

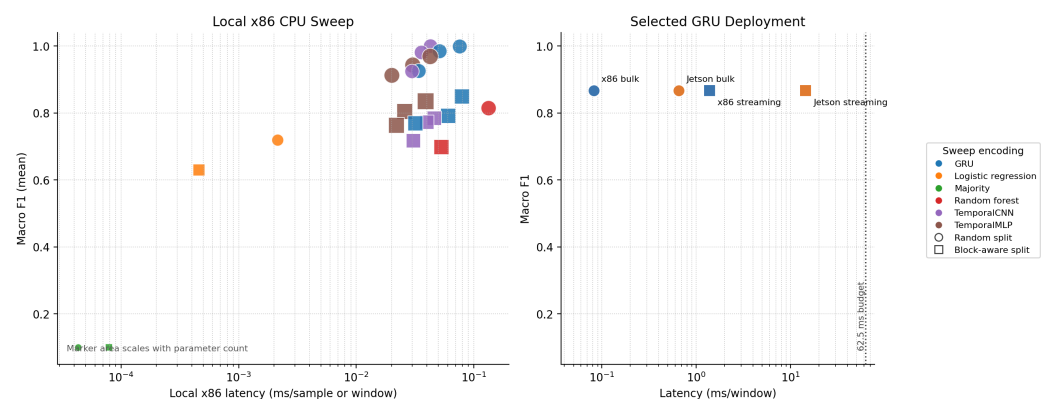


Figure 7. Latency versus macro F1. Left: local x86 CPU sweep across frame-based and temporal models, with marker area scaled by parameter count. Right: selected GRU checkpoint measured separately for bulk and streaming inference on local x86 CPU and Jetson Nano CPU, with the 62.5 ms decision budget shown for deployment context.

Table 10. Selected block-aware results with deployment-oriented statistics measured in the local x86 benchmarking environment. The random forest parameter count is a model-complexity proxy based on tree nodes and is not directly comparable to neural trainable parameters.

Model	Macro F1	Parameters	Artifact (kB)	Local CPU Latency (ms)
Random forest (frame)	0.6979	68,944	6700.1	0.0531
TemporalCNN ($w = 32$)	0.7846	12,836	58.4	0.0462
TemporalMLP ($w = 32$)	0.8345	557,956	2234.9	0.0388
GRU ($w = 32$)	0.8494	25,220	103.7	0.0792

4.6. Jetson Nano CPU Deployment Validation

The selected deployment candidate was further validated on an NVIDIA Jetson Nano Developer Kit running Linux 4.9.337-tegra, Python 3.8.0, and PyTorch 2.4.1 under the CPU execution path used for the device-side experiment. Reloading the best block-aware GRU checkpoint (window = 32, seed 27) preserved a macro F1-score of 0.8667 on the 624-window test split. Bulk inference averaged 0.6514 ms/window (1535.26 windows/s), while streaming inference averaged 14.4796 ms/window (69.06 windows/s). Given the stride-2 setup and approximately 32 Hz sampling rate, the decision interval budget is 62.5 ms, which yields 4.32× compute headroom for steady-state inference. The 32-frame context still imposes a 1000 ms initial window-fill latency before the first decision, so the deployment claim here should be interpreted as steady-state compute feasibility on prepared windows rather than instantaneous response from an empty buffer. The checkpoint-level macro F1 is slightly higher than the 0.8494 mean reported in Table 10 because the Jetson experiment reloads the single best seed-27 checkpoint rather than averaging across three seeds. Table 11 summarizes the Jetson Nano CPU-side validation protocol and measurements.

Table 11. Jetson Nano CPU-side deployment validation for the selected block-aware GRU checkpoint.

Metric	Value
Device	NVIDIA Jetson Nano Developer Kit; CPU execution path in the tested PyTorch environment
Reloaded checkpoint macro F1	0.8667 on 624 test windows
Timing repetitions	Bulk: 10 repeated mini-batch passes with batch size 64; streaming: 3 warm-up single-window inferences followed by 10 full-test streaming passes
Streaming path	Prepared windows processed one at a time, including per-window standardization, tensor construction, transfer to the PyTorch CPU device, and PyTorch forward pass; sensor I/O and rendering excluded
Power/governor metadata	CPU governor and Jetson power mode were not captured in the archived remote run; these fields are marked unavailable for that measurement
Bulk inference latency	0.6514 ms/window (1535.26 windows/s)
Streaming inference latency	14.4796 ms/window (69.06 windows/s)
Decision interval budget	62.5 ms for stride 2 at approximately 32 Hz
Window-fill latency	1000.0 ms for the 32-frame context window
Compute headroom	4.32×; compute budget satisfied

4.7. Class-Wise Behavior of the Best Block-Aware Model

Table 12 reports the class-wise performance of the best conservative model, namely the GRU with 32-frame windows. Swipe is the easiest class, with F1 = 0.9753, followed by Tap

at 0.9588. Approaching is moderate at 0.7937. Hold is the hardest class, with F1 = 0.6697, despite very high precision.

This pattern is consistent with the semantics of the gestures. Swipe and Tap contain more distinctive temporal transitions, whereas Hold is comparatively static and may overlap with neighboring depth profiles once the spatial representation is compressed to 8×8 measurements. The high precision but lower recall for Hold indicates that when the GRU predicts Hold it is usually correct, but it misses a substantial portion of true Hold instances.

Table 12. Class-wise performance of the best block-aware model (GRU, window = 32). Values are means over three random seeds.

Class	Precision	Recall	F1	Support
Approaching	0.6611	0.9936	0.7937	156
Hold	0.9892	0.5085	0.6697	156
Swipe	0.9780	0.9744	0.9753	156
Tap	0.9683	0.9509	0.9588	156

Figure 8 shows the saved confusion matrix for the representative seed-27 GRU checkpoint used in the device-side validation. In contrast to Table 12, which averages class-wise metrics across three seeds, this figure illustrates the concrete error pattern of the checkpoint reloaded for the Jetson experiment.

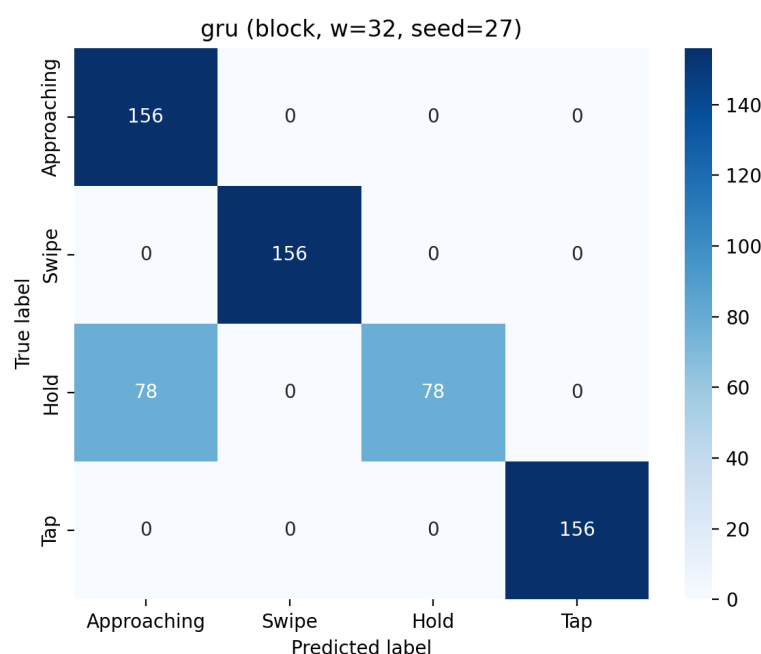


Figure 8. Confusion matrix for the representative block-aware GRU checkpoint with 32-frame windows.

5. Discussion

The expanded evidence supports three main conclusions. First, evaluation design is not a secondary benchmark detail; it materially changes the empirical interpretation across all four datasets. The optimism gap is 0.1506 on the original IEEE ToF benchmark, 0.3303 on DS.GestureRecognition_TOF.1.0, 0.7073 on the 8×8 Kinect+Leap depth benchmark, and 0.8351 on the 8×8 Senz3D depth benchmark. In other words, within the evaluated corpora, the apparent difficulty of low-resolution gesture recognition depends strongly on whether repeated temporal structure or subject identity is allowed to leak across partitions.

Second, genuine low-resolution ToF streams remain more promising than aggressively downsampled general depth data in this benchmark. On the two real ToF datasets, structure-aware macro F1 remains in the 0.6697–0.8494 range and temporal context is useful. On the converted Kinect+Leap and Senz3D benchmarks, however, subject-wise 8×8 recognition is near chance at 0.0566 and 0.0717 macro F1, indicating that these particular depth corpora do not retain enough subject-generalizable discrimination after heavy spatial compression with the evaluated models. This result is academically useful because it narrows the claim: the paper is not arguing that any depth dataset can be collapsed to 8×8 without consequence, but rather that benchmark conclusions should explicitly report where such compression still works and where it does not.

Third, lightweight temporal modeling remains relevant for deployment-oriented prototyping when the source data are genuine low-resolution ToF streams. On the primary IEEE benchmark, the GRU with 32-frame windows is still the strongest observed conservative model when recognition quality is prioritized, while the TemporalCNN remains attractive when parameter count and artifact size matter more. The Jetson Nano CPU validation keeps this argument concrete: the selected GRU remained within the steady-state compute budget with 4.32× headroom on embedded hardware, although its 32-frame temporal context still imposes a 1.0 s initial buffer fill before the first decision. This result supports steady-state inference feasibility on prepared windows; it should not be read as a complete sensor-to-decision deployment validation.

The study therefore has methodological implications beyond any one corpus. For compact depth streams, benchmark papers should report whether windows cross class boundaries, whether normalization uses only training data, and whether splitting preserves temporal, sequence, or subject structure. These details are critical to interpretability and reproducibility, especially when datasets are small, publicly shared as a single CSV file, or released as repeated samples from only a few users.

The theoretical and empirical results are aligned. The fixed 8×8 averaging operator reduces spatial detail, so the strongest claims are expected on native low-resolution temporal ToF streams where motion can compensate for sparse spatial evidence. Conversely, the subject-wise Kinect+Leap and Senz3D results show the expected failure mode: when high-resolution single depth samples are compressed to 8×8 and then evaluated across unseen subjects, the remaining signal is often insufficient for the evaluated compact models. The conclusions therefore distinguish between deployment feasibility for native ToF streams and the limitations of aggressive low-resolution conversion on broader depth archives.

Future work should proceed in four directions. First, intermediate spatial resolutions such as 16×16 should be evaluated on the high-resolution depth corpora to quantify the privacy–accuracy trade-off between native 8×8 ToF sensing and less severe downsampling. Second, additional native low-resolution recordings should be collected across more subjects, sessions, distances, and sensor placements so that block-aware and subject-aware validation can be combined in a single real ToF benchmark. Third, attention-based and mobile-optimized models, including compact transformer hybrids and MobileNet-style video backbones, should be tested under the same leakage-aware splits before claims about improved cross-subject generalization are made. Fourth, the device-side evaluation should be expanded to include quantized models, power draw, thermal throttling, CPU governor state, and additional hardware targets. The protocol should also be extended to other low-resolution sensing tasks, including radar, thermal, and sign-language or alphabet-style gesture recognition, where subject identity and repeated sequences may create similar optimism gaps.

6. Study Limitations

This study is intentionally conservative in its claims. Although the benchmark now spans four public datasets, the metadata remain uneven across corpora. The IEEE ToF CSV has no explicit subject or session identifiers, DS.GestureRecognition_TOF1.0 contains only one public subject, and only Kinect+Leap and Senz3D enable true subject-wise splits. Consequently, the experiments still cannot support a universal claim about subject-independent or session-independent low-resolution gesture recognition. They also do not establish robustness to occlusion, sensor displacement, lighting changes, or broader real-world variation.

The study also does not evaluate image deblurring, artificially blurred RGB images, or restoration from blurred observations. The term “downsampled” in the manuscript refers only to deterministic spatial block averaging of real depth maps into a common 8×8 representation. If that averaging is interpreted as a form of synthetic degradation, the resulting claim should be read narrowly: it evaluates low-resolution depth recognition under a fixed compression operator, not training or testing on artificially blurred images.

The reported latency trends remain environment-dependent. Although the study includes an actual Jetson Nano CPU validation for the selected GRU checkpoint, the manuscript does not claim end-to-end sensor-to-decision latency. The Jetson measurements cover only model inference on prepared windows. Power consumption, quantized inference, GPU execution, and cross-hardware speed comparisons were not measured in the archived run. A production-ready interface would still require joint measurement of sensor acquisition, buffering, rendering, application logic, and a stable hardware-specific runtime stack. In addition, the study focuses on four gestures in a controlled setting. A production-ready interface would require rejection of non-gesture states, calibration across multiple users and positions, and likely additional data collected across sessions.

These limitations define the scope of the benchmark. The present results should be interpreted as a careful study of evaluation design, low-resolution conversion, and lightweight temporal modeling across public ToF and depth datasets, not as a universal gesture recognition claim.

7. Conclusions

This paper presented a reproducible multi-corpus, within-dataset benchmark of frame-based and lightweight temporal models across four public low-resolution gesture corpora. The expanded results show that naive random splitting can substantially overstate performance on genuine ToF sequences and on downsampled depth data alike. On the primary IEEE ToF benchmark, the best macro F1-score decreases from 1.0000 under random splitting to 0.8494 under block-aware evaluation. On DS.GestureRecognition_TOF1.0, the best score decreases from 1.0000 to 0.6697 under sequence-aware evaluation. On the Kinect+Leap depth benchmark reduced to 8×8 , random performance reaches 0.7639 macro F1, but subject-wise evaluation drops to 0.0566. On the Senz3D depth benchmark reduced to 8×8 , random performance reaches 0.9068 macro F1, but subject-wise evaluation drops to 0.0717.

Taken together, the results support a practical recommendation: public low-resolution gesture benchmarks should report structure-aware splits before drawing strong conclusions, and the scientific claim should be tied to the dataset family on which it was tested. Lightweight temporal models remain effective for the evaluated genuine ToF streams, and the Jetson Nano CPU validation supports steady-state inference feasibility for the selected checkpoint at 14.48 ms/window. However, the external depth results also show that aggressive 8×8 compression does not automatically preserve cross-subject discrimination. The experiments use real public ToF/depth data and deterministic depth-grid conversion rather than synthetic blurred-image generation. The main contribution is therefore a more care-

fully controlled multi-corpus evaluation framework for privacy-preserving low-resolution gesture recognition rather than a universal gesture recognizer.

Funding: This research received no external research funding. The article processing charge (APC) was supported by Istanbul Topkapi University.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The analyzed datasets are publicly available from IEEE DataPort under the title *Time-of-Flight (ToF) Gesture Recognition Dataset* [12], <https://doi.org/10.21227/qf2z-yb93>; from Zenodo as *DS.GestureRecognition_TOF.1.0* [13], <https://doi.org/10.5281/zenodo.17386447>; and from the University of Padova as the public Kinect+Leap dataset [14] and Senz3D dataset [15], <https://ltnm.dei.unipd.it/downloads/gesture/> (accessed on 18 May 2026). The benchmark code, experiment configuration, generated figures, derived tables, raw metric files, converted canonical CSV files, and hardware benchmark provenance are available from the author upon reasonable request.

Acknowledgments: The author thanks the maintainers of the public datasets and the developers of the open-source scientific computing libraries used in this work.

Conflicts of Interest: The author declares no conflicts of interest.

References

- Wachs, J.P.; Kölsch, M.; Stern, H.; Edan, Y. Vision-based hand-gesture applications. *Commun. ACM* **2011**, *54*, 60–71. <https://doi.org/10.1145/1897816.1897838>.
- Ohn-Bar, E.; Trivedi, M.M. Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 2368–2377. <https://doi.org/10.1109/TITS.2014.2337331>.
- Hansard, M.; Lee, S.; Choi, O.; Horaud, R. *Time-of-Flight Cameras: Principles, Methods and Applications*; Springer: London, UK, 2013. <https://doi.org/10.1007/978-1-4471-4658-2>.
- Foix, S.; Alenya, G.; Torras, C. Lock-in Time-of-Flight (ToF) cameras: A survey. *IEEE Sens. J.* **2011**, *11*, 1917–1926. <https://doi.org/10.1109/JSEN.2010.2101060>.
- Kolb, A.; Barth, E.; Koch, R.; Larsen, R. Time-of-flight cameras in computer graphics. *Comput. Graph. Forum* **2010**, *29*, 141–159. <https://doi.org/10.1111/j.1467-8659.2009.01583.x>.
- STMicroelectronics. VL53L5CX Multizone Time-of-Flight Ranging Sensor. Available online: <https://www.st.com/en/imaging-and-photonics-solutions/vl53l5cx.html> (accessed on 14 May 2026).
- Kaufman, S.; Rosset, S.; Perlich, C.; Stitelman, O. Leakage in data mining: Formulation, detection, and avoidance. *ACM Trans. Knowl. Discov. Data* **2012**, *6*, 15:1–15:21. <https://doi.org/10.1145/2382577.2382579>.
- Arlot, S.; Celisse, A. A survey of cross-validation procedures for model selection. *Stat. Surv.* **2010**, *4*, 40–79. <https://doi.org/10.1214/09-SS054>.
- Varma, S.; Simon, R. Bias in error estimation when using cross-validation for model selection. *BMC Bioinform.* **2006**, *7*, 91. <https://doi.org/10.1186/1471-2105-7-91>.
- Bergmeir, C.; Benitez, J.M. On the use of cross-validation for time series predictor evaluation. *Inf. Sci.* **2012**, *191*, 192–213. <https://doi.org/10.1016/j.ins.2011.12.028>.
- Roberts, D.R.; Bahn, V.; Ciuti, S.; Boyce, M.S.; Elith, J.; Guillera-Aroita, G.; Hauenstein, S.; Lahoz-Monfort, J.J.; Schroder, B.; Thuiller, W.; et al. Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography* **2017**, *40*, 913–929. <https://doi.org/10.1111/ecog.02881>.
- Obana, I. Time-of-Flight (ToF) Gesture Recognition Dataset. IEEE DataPort. 2026. Available online: <https://ieee-dataport.org/documents/time-flight-tof-gesture-recognition-dataset> (accessed on 14 May 2026).
- Center for Research and Technology Hellas. DS.GestureRecognition_TOF.1.0. *Zenodo* **2025**. <https://doi.org/10.5281/zenodo.17386447>.
- Marin, G.; Dominio, F.; Zanuttigh, P. Microsoft Kinect and Leap Motion Gesture Dataset. University of Padova. 2014. Available online: https://ltnm.dei.unipd.it/downloads/gesture/#kinect_leap (accessed on 18 May 2026).
- Marin, G.; Dominio, F.; Zanuttigh, P. Senz3D Gesture Dataset. University of Padova. 2014. Available online: <https://ltnm.dei.unipd.it/downloads/gesture/#senz3d> (accessed on 18 May 2026).
- Fertl, E.; Castillo, E.; Stettinger, G.; Cuéllar, M.P.; Morales, D.P. Hand gesture recognition on edge devices: Sensor technologies, algorithms, and processing hardware. *Sensors* **2025**, *25*, 1687. <https://doi.org/10.3390/s25061687>.

17. Emporio, M.; Ghasemaghaei, A.; Laviola, J.J., Jr.; Giachetti, A. Continuous hand gesture recognition: Benchmarks and methods. *Comput. Vis. Image Underst.* **2025**, *259*, 104435. <https://doi.org/10.1016/j.cviu.2025.104435>.
18. Yaseen; Kwon, O.-J.; Kim, J.; Lee, J.; Ullah, F. Evaluation of benchmark datasets and deep learning models with pre-trained weights for vision-based dynamic hand gesture recognition. *Appl. Sci.* **2025**, *15*, 6045. <https://doi.org/10.3390/app15116045>.
19. Molchanov, P.; Gupta, S.; Kim, K.; Kautz, J. Hand gesture recognition with 3D convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Boston, MA, USA, 7–12 June 2015; pp. 1–7. <https://doi.org/10.1109/CVPRW.2015.7301342>.
20. Molchanov, P.; Yang, X.; Gupta, S.; Kim, K.; Tyree, S.; Kautz, J. Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4207–4215. <https://doi.org/10.1109/CVPR.2016.456>.
21. Lea, C.; Flynn, M.D.; Vidal, R.; Reiter, A.; Hager, G.D. Temporal convolutional networks for action segmentation and detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1003–1012. <https://doi.org/10.1109/CVPR.2017.113>.
22. Ismail Fawaz, H.; Forestier, G.; Weber, J.; Idoumghar, L.; Muller, P.-A. Deep learning for time series classification: A review. *Data Min. Knowl. Discov.* **2019**, *33*, 917–963. <https://doi.org/10.1007/s10618-019-00619-1>.
23. Bagnall, A.; Lines, J.; Bostrom, A.; Large, J.; Keogh, E. The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances. *Data Min. Knowl. Discov.* **2017**, *31*, 606–660. <https://doi.org/10.1007/s10618-016-0483-9>.
24. Wang, Z.; Yan, W.; Oates, T. Time series classification from scratch with deep neural networks: A strong baseline. In Proceedings of the International Joint Conference on Neural Networks, Anchorage, AK, USA, 14–19 May 2017; pp. 1578–1585. <https://doi.org/10.1109/IJCNN.2017.7966039>.
25. Bai, S.; Kolter, J.Z.; Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv* **2018**, arXiv:1803.01271. <https://doi.org/10.48550/arXiv.1803.01271>.
26. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861. <https://doi.org/10.48550/arXiv.1704.04861>.
27. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520. <https://doi.org/10.1109/CVPR.2018.00474>.
28. Howard, A.; Sandler, M.; Chu, G.; Chen, L.-C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for MobileNetV3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324. <https://doi.org/10.1109/ICCV.2019.00140>.
29. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. *arXiv* **2016**, arXiv:1602.07360. <https://doi.org/10.48550/arXiv.1602.07360>.
30. Han, S.; Mao, H.; Dally, W.J. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. *arXiv* **2015**, arXiv:1510.00149. <https://doi.org/10.48550/arXiv.1510.00149>.
31. Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 2704–2713. <https://doi.org/10.1109/CVPR.2018.00286>.
32. Lane, N.D.; Bhattacharya, S.; Georgiev, P.; Forlivesi, C.; Jiao, L.; Qendro, L.; Kawsar, F. DeepX: A software accelerator for low-power deep learning inference on mobile devices. In Proceedings of the 15th ACM/IEEE International Conference on Information Processing in Sensor Networks, Vienna, Austria, 11–14 April 2016; pp. 1–12. <https://doi.org/10.1109/IPSNS.2016.7460664>.
33. Bertasius, G.; Wang, H.; Torresani, L. Is space–time attention all you need for video understanding? In *Proceedings of the 38th International Conference on Machine Learning, Virtual, 18–24 July 2021*; PMLR: Cambridge, MA, USA, 2021; Volume 139, pp. 813–824. Available online: <https://proceedings.mlr.press/v139/bertasius21a.html> (accessed on 18 May 2026).
34. Mehta, S.; Rastegari, M. MobileViT: Light-weight, general-purpose, and mobile-friendly vision transformer. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022. <https://doi.org/10.48550/arXiv.2110.02178>.
35. Marin, G.; Dominio, F.; Zanuttigh, P. Hand gesture recognition with Leap Motion and Kinect devices. In Proceedings of the 2014 IEEE International Conference on Image Processing, Paris, France, 27–30 October 2014; pp. 1565–1569. <https://doi.org/10.1109/ICIP.2014.7025313>.
36. Köpüklü, O.; Ledwon, T.; Rong, Y.; Köse, N.; Rigoll, G. DriverMHG: A multi-modal dataset for dynamic recognition of driver micro hand gestures and a real-time recognition framework. In Proceedings of the 15th IEEE International Conference on Automatic Face and Gesture Recognition, Buenos Aires, Argentina, 16–20 November 2020; pp. 77–84. <https://doi.org/10.1109/FG47880.2020.00041>.

37. Vandersteegen, M.; Reusen, W.; Van Beeck, K.; Goedemé, T. Low-latency hand gesture recognition with a low-resolution thermal imager. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 440–449. <https://doi.org/10.1109/CVPRW50498.2020.00057>.
38. Safa, A.; Mommen, W.; Wambacq, P.; Keuninckx, L. Resource-efficient gesture recognition using low-resolution thermal camera via spiking neural networks and sparse segmentation. In Proceedings of the 2024 IEEE 18th International Conference on Automatic Face and Gesture Recognition, Istanbul, Turkey, 27–31 May 2024; pp. 1–5. <https://doi.org/10.1109/FG59268.2024.10582024>.
39. Sluÿters, A.; Lambot, S.; Vanderdonckt, J. Hand gesture recognition for an off-the-shelf radar by electromagnetic modeling and inversion. In Proceedings of the 27th International Conference on Intelligent User Interfaces, Helsinki, Finland, 22–25 March 2022; pp. 506–522. <https://doi.org/10.1145/3490099.3511107>.
40. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. <https://doi.org/10.1023/A:1010933404324>.
41. Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1724–1734. <https://doi.org/10.3115/v1/D14-1179>.
42. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>.
43. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980. <https://doi.org/10.48550/arXiv.1412.6980>.
44. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
45. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
46. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*; Curran Associates: Red Hook, NY, USA, 2019; pp. 8024–8035. Available online: <https://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library> (accessed on 18 May 2026).
47. Sokolova, M.; Lapalme, G. A systematic analysis of performance measures for classification tasks. *Inf. Process. Manag.* **2009**, *45*, 427–437. <https://doi.org/10.1016/j.ipm.2009.03.002>.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.