Comparing Three On-Device Classifier Deployments for Ring Microgesture Recognition

Yuxin Du
yuxin.du@utexas.edu
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, Texas, USA

Edison Thomaz
ethomaz@utexas.edu
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, Texas, USA

Abstract

Smart rings are becoming a common wearable platform, making low-power on-device gesture recognition increasingly important. Recognizers based on inertial measurement units (IMUs) can place classification inside the sensor or on the microcontroller, but their system-level trade-offs remain unclear. We compare three on-device deployments on one custom ring: an in-sensor decision tree executed by the IMU's machine-learning core (MLC), an MCU-side convolutional neural network (CNN), and an MCU-side hyperdimensional computing (HDC) classifier. The CNN achieves the highest within-user and mean leave-one-participant-out (LOPO) macro-F1, but the MLC consumes the least measured system power (6.64 mW) and produces roughly a quarter as many false activations as the CNN on a free-living recording. Raw-sample HDC trails; an offline ablation with window-level statistical features raises its within-user macro-F1 from 0.691 to 0.904, indicating that representation contributes substantially to its window-level gap. For this platform and task, the MLC provides the most resource-efficient operating point, whereas the CNN is the accuracy-first choice.

CCS Concepts

• **Human-centered computing** → **Ubiquitous and mobile computing**; • **Computer systems organization** → *Embedded systems*; • **Computing methodologies** → *Machine learning*.

Keywords

smart rings, gesture recognition, on-device machine learning, wearable sensing, in-sensor computing



This work is licensed under a Creative Commons Attribution 4.0 International License.
HumanSys '26, Austin, TX, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2959-1/2026/10
https://doi.org/10.1145/3842436.3843816

ACM Reference Format:
Yuxin Du and Edison Thomaz. 2026. Comparing Three On-Device Classifier Deployments for Ring Microgesture Recognition. In *The 4th International Workshop on Human-Centered Sensing, Modeling, and Intelligent Systems (HumanSys '26), October 26–30, 2026, Austin, TX, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3842436.3843816

1 Introduction

Smart rings are becoming an increasingly important platform for wearable interaction, moving from research prototypes into mainstream consumer devices [6, 9]. A ring sits within easy reach of the thumb, which makes thumb-to-finger microgestures a natural channel for subtle, always-available input [1]. Recognizing these gestures, however, requires sensing them on the ring itself, and the small form factor constrains the use of bulky or power-intensive modalities such as cameras and dense capacitive arrays [7]. Inertial measurement units (IMUs) are therefore a common sensing option because they capture both linear and rotational finger motion in a millimeter-scale, low-power package [5]. But a ring offers only tens of milliamp-hours of battery and a memory-limited MCU, and streaming raw data to a phone for classification costs radio energy and moves continuous motion data off the device [3]. As a result, recognition is increasingly run on the ring itself.

On modern hardware, on-device inertial recognition can execute inside the IMU through a machine learning core (MLC), on the microcontroller (MCU) as a convolutional neural network (CNN) quantized for deployment, or on the MCU through hyperdimensional computing (HDC) [2, 4, 8]. These approaches have been evaluated on different devices, datasets, and tasks. This leaves an open question for on-device recognition: which deployment approach to adopt, and under what conditions. To answer this, we conduct an end-to-end comparison of three on-device classifiers on the same ring, combining leave-session-out and leave-one-participant-out (LOPO) accuracy, continuous free-living behavior, and measured firmware cost. The comparison characterizes the performance–resource trade-off across the three deployments, identifies input representation as

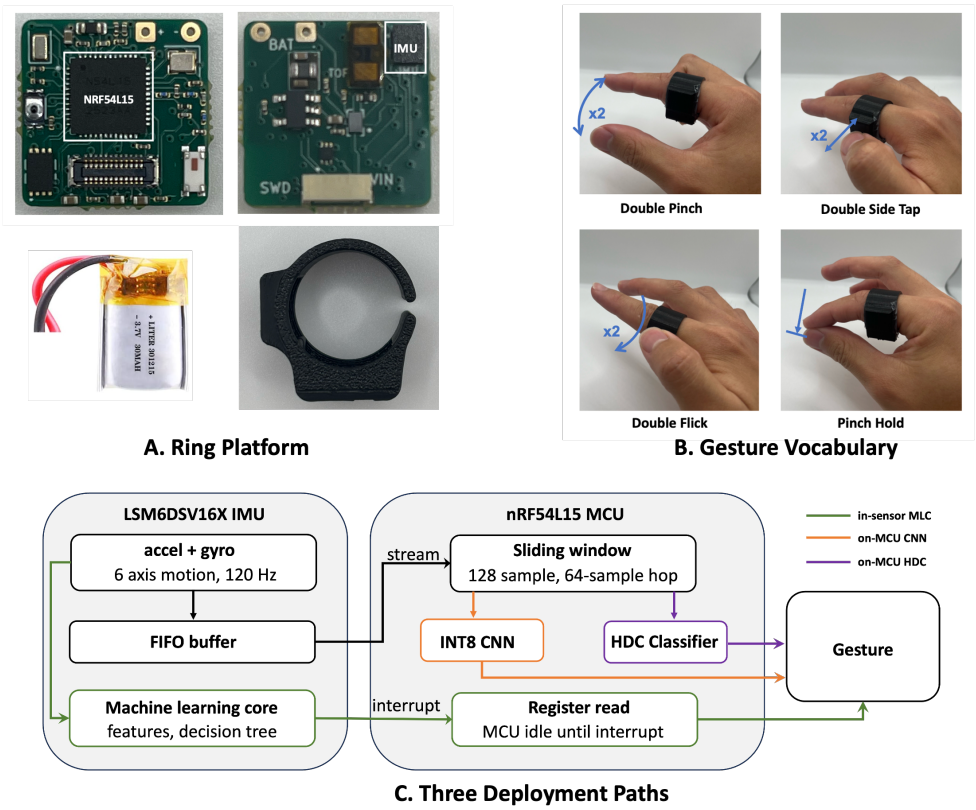


Figure 1: Ring platform, gesture vocabulary, and three on-device deployment paths. (A) Ring hardware. (B) Four target microgestures. (C) The MLC performs decision-tree inference inside the IMU and interrupts the MCU with the result; the CNN and HDC classifiers process windowed FIFO data on the MCU.

a major source of HDC's window-level accuracy gap, and shows that segmented accuracy alone is not enough to choose an always-on ring classifier.

2 Methods

2.1 Hardware

Figure 1 shows the ring platform, the gesture set, and the three deployment paths. The ring is built around a Nordic nRF54L15 MCU (a 128 MHz Arm Cortex-M33 with 256 KB of RAM) and an STMicroelectronics LSM6DSV16X six-axis IMU, both mounted on a 15 × 15 mm PCB powered by a 30 mAh LiPo battery. For MCU-side inference, the CNN and HDC read IMU samples through the sensor's first-in, first-out (FIFO) buffer. The MLC executes decision-tree inference inside the IMU, so the MCU can stay in its System ON idle mode with interrupt-based wake-up and wake only when a classification is ready. The ring is worn on the right index finger, and an asymmetric enclosure fixes its orientation. Bluetooth Low Energy (BLE) is used only for data collection and verification.

2.2 Study Design

2.2.1 Task Definition. We recognize four target ring gestures: double side tap, two quick thumb taps on the side of the ring; double pinch, two thumb–index pinches; pinch hold, a thumb–index pinch held steady; and double flick, two quick downward thumb flicks along the side of the ring. A null outcome represents all non-target activity. All three classifiers emit one of these five outcomes.

2.2.2 Data Collection. Six participants, denoted P1–P6, contributed 25 guided sessions. P1 completed 20 sessions, removing and re-donning the ring before each session, while P2–P6 completed one session each. Each session contained 15 repetitions of each target gesture, yielding 60 prompted gesture instances per session and 1,500 in total.

Participants also contributed null recordings containing non-target desk activity, object handling, and unconstrained wear. The IMU sampled three-axis acceleration and three-axis angular velocity at 120 Hz and streamed timestamped samples to an iOS application over BLE. The study was approved by our university's Institutional Review Board.

Table 1: Target-gesture macro-F1 on held-out guided windows. Scores are macro-averaged over the four target gestures, and a null prediction counts as an error. Within-user results use five leave-session-out folds on P1; LOPO columns identify the held-out participant, and the LOPO mean gives each participant equal weight.

Method	Within-user	LOPO held-out participant						LOPO mean
		P1	P2	P3	P4	P5	P6	
MLC	0.945 ± 0.036	0.907	0.696	0.636	0.718	0.808	0.637	0.734 ± 0.106
CNN	0.978 ± 0.015	0.857	0.884	0.600	0.772	0.750	0.810	0.779 ± 0.101
HDC	0.691 ± 0.077	0.277	0.458	0.513	0.340	0.290	0.190	0.345 ± 0.121

2.3 Three Classifier Deployments

All methods operate on six-channel windows of 128 samples (1.07 s). CNN and HDC advance by 64 samples (0.53 s), whereas the MLC advances by one non-overlapping 128-sample feature window. MLC and CNN learn null as a fifth class; HDC treats null as a rejection outcome. Continuous results are normalized by recorded time because the methods have different decision cadences.

In-sensor MLC. For each fold, class-balanced training windows are exported to ST MEMS Studio [8] to train a depth-limited decision tree over sensor-supported statistics, including means, variance, energy, peak-to-peak range, and vector magnitudes. Each exported tree is instantiated in an independent offline evaluator. For the deployed fold, offline decisions were verified against live sensor outputs. At runtime the MCU waits in System ON idle, wakes on an MLC interrupt, and reads the predicted class from the sensor.

MCU CNN. The CNN contains three one-dimensional convolutional layers with 16, 32, and 64 filters, followed by global average pooling and a five-class output. It is trained per fold using training-fold standardization, rotation, temporal-shift, and noise augmentation, with early stopping on validation macro-F1. For deployment, the selected model is converted using full-integer 8-bit post-training quantization and executed with TensorFlow Lite for Microcontrollers [2]. The deployed fold retained its float validation and test macro-F1 after quantization.

MCU HDC. HDC uses 2,048-dimensional binary spatter codes. Each sample is quantized into 32 levels using integer percentile bounds fitted on training data. Channel and level hypervectors are bound into temporal trigrams and bundled into one window hypervector. Class prototypes are accumulated from training examples and refined with perceptron-style updates. At inference, the four gesture prototypes compete by Hamming distance; validation-calibrated maximum-distance and minimum-margin thresholds map low-confidence windows to null. Bit-exact tests verify agreement between offline and firmware inference.

2.4 Evaluation Protocol

We use two complementary evaluations. Within-user recognition uses P1 only, with five leave-session-out folds: each fold trains on 14 sessions, validates on two, and tests on four, so every session is tested once. LOPO uses six folds; each fold excludes one participant from training and validation. The remaining participants are downsampled to contribute equal amounts of gesture and null data, so every fold trains on the same amount of data.

Target-gesture performance is measured using macro-F1 over the four gestures, with a null prediction counted as an error. On continuous streams, a non-null window emits an activation (1 s refractory); guided-event recall is the fraction of prompted gestures with an activation within 1 s, and every activation on free-living data is a false activation.

3 Evaluation

3.1 Recognition Accuracy

Table 1 reports gesture macro-F1. In the within-user evaluation, the CNN achieves the highest mean and the lowest variation, followed by the MLC, while raw-sample HDC is both less accurate and more variable across the five folds. To understand why HDC trails, we conduct a representation ablation: we replace its raw-sample encoding with window-level statistical features (the same family the MLC uses) while leaving the HDC classifier otherwise unchanged. This raises HDC's within-user macro-F1 to 0.904±0.039. The variant was evaluated offline only, so the tables report the raw-sample configuration that we deployed and measured. The improvement points to input representation as a major contributor to HDC's window-level gap. Replaying the continuous streams with it, however, produced no comparable gain, so the window-level improvement did not carry over to continuous use.

Across participants, the CNN obtains the highest LOPO mean macro-F1 at 0.779, followed by the MLC at 0.734 and HDC at 0.345. Per gesture, double flick transfers most consistently (MLC 0.932; CNN 0.943), whereas pinch hold is the hardest target (MLC 0.550; CNN 0.523).

Table 2: Power, flash, and static RAM of the firmware configurations. Memory footprints are reported in KB (1 KB = 1024 B).

Config.	Current (mA)	Power (mW)	Flash (KB)	RAM (KB)
Baseline	1.86	7.81	61.9	26.6
MLC	1.58	6.64	62.0	20.7
HDC	3.34	14.03	74.4	40.2
CNN	4.90	20.58	166.6	73.8

3.2 On-Device System Cost

Table 2 reports power, flash, and static RAM for a sensing baseline and for the MLC, HDC, and int8 CNN firmware, all measured with BLE and logging disabled. The baseline drains the 120 Hz FIFO without classifying, so it separates the cost of streaming samples from the cost of recognition. Current is measured at 4.2 V, and flash and statically allocated RAM come from the linked firmware images.

The MLC has the lowest power and the smallest MCU footprint among the three classifiers. Its current is even lower than the baseline, because inference runs inside the IMU and the MCU no longer drains the raw-data FIFO. HDC sits at an intermediate cost, while the CNN is the most expensive on every axis, adding a larger model, an inference runtime, and a tensor arena on the MCU.

3.3 Continuous-Stream Behavior

Using the five within-user fold models from P1, we measure guided-event recall by replaying P1's guided recordings, and false activations by replaying a separate 2.66-hour free-living recording from P1 during which no gestures were prompted. Guided-event recall is 0.842 for MLC, 0.826 for CNN, and 0.530 for HDC. The corresponding false-activation rates, averaged across the five models, are 18.78, 84.15, and 728.13 per hour. The MLC decides once per 1.07 s window while the others decide every 0.53 s, so these rates correspond to 0.56%, 1.25%, and 10.8% of decisions. Most false activations come from the pinch hold and double side tap classes, which most resemble everyday motion.

4 Discussion

Our comparison supports three conclusions about this ring and task. First, the accuracy CNN adds over MLC is small, while its system cost is not. CNN draws 3.1 times the power and occupies 2.7 times the flash. MLC is therefore the stronger choice when energy and MCU footprint are constrained, and CNN when the added accuracy is worth its cost.

Second, HDC's low ranking is not a property of the classifier alone. With the classifier unchanged, window-level statistical features recover most of this gap. Comparisons between on-device paradigms therefore depend on the input representation chosen for each, not only on the classifier.

Third, segmented accuracy does not predict continuous reliability. MLC and CNN reach comparable guided-event recall, but MLC produces roughly a quarter as many free-living false activations, reversing their window-level ranking. Even MLC's rate would be problematic for an always-on interface; all three methods need contextual gating or more null data before deployment.

Our study has limitations. The cohort is small and unevenly sampled: one participant contributed 20 sessions and five others one each, so the within-user result relies on a single participant and most LOPO folds are tested on one session. All data were collected from the right index finger. Other fingers, placements, and gesture sets remain to be tested.

References

[1] Edwin Chan, Teddy Seyed, Wolfgang Stuerzlinger, Xing-Dong Yang, and Frank Maurer. 2016. User Elicitation on Single-hand Microgestures. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. Association for Computing Machinery, New York, NY, USA, 3403–3414. doi:10.1145/2858036.2858589

[2] Robert David, Jared Duke, Advait Jain, Vijay Janapa Reddi, Nat Jeffries, Jian Li, Nick Kreeger, Ian Nappier, Meghna Natraj, Shlomi Regev, Rocky Rhodes, Tiezhen Wang, and Pete Warden. 2021. TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems. arXiv:2010.08678 [cs.LG]. doi:10.48550/arXiv.2010.08678

[3] Yizheng Gu, Chun Yu, Zhipeng Li, Zhaoheng Li, Xiaoying Wei, and Yuanchun Shi. 2020. QwertyRing: Text Entry on Physical Surfaces Using a Ring. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4 (Dec. 2020), 128:1–128:29. doi:10.1145/3432204

[4] Chae Young Lee, Pu (Luke) Yi, Maxwell Fite, Tejus Rao, Sara Achour, and Zerina Kapetanovic. 2025. HyperCam: Low-Power Onboard Computer Vision for IoT Cameras. In *Proceedings of the 31st Annual International Conference on Mobile Computing and Networking (ACM MOBICOM '25)*. Association for Computing Machinery, New York, NY, USA, 529–542. doi:10.1145/3680207.3723490

[5] Chen Liang, Chun Yu, Yue Qin, Yuntao Wang, and Yuanchun Shi. 2021. DualRing: Enabling Subtle and Expressive Hand Interaction with Dual IMU Rings. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 3 (Sept. 2021), 115:1–115:27. doi:10.1145/3478114

[6] Ōura Health Oy. 2026. *Oura Ring. Smart Ring for Fitness, Stress, Sleep & Health.* https://ouraring.com/

[7] Roy Shilkrot, Jochen Huber, Jürgen Steimle, Suranga Nanayakkara, and Pattie Maes. 2015. Digital Digits: A Comprehensive Survey of Finger Augmentation Devices. *ACM Computing Surveys (CSUR)* 48, 2 (Nov. 2015), 30:1–30:29. doi:10.1145/2828993

[8] STMicroelectronics. 2023. *AN5804 – LSM6DSV16X: machine learning core.* https://www.st.com/resource/en/application_note/an5804-lsm6dsv16x-machine-learning-core-stmicroelectronics.pdf

[9] Zeyu Wang, Ruotong Yu, Xiangyang Wang, Jiexin Ding, Jiankai Tang, Jun Fang, Zhe He, Zhuojun Li, Tobias Röddiger, Weiye Xu, Xiyuxing Zhang, Huan-ang Gao, Nan Gao, Chun Yu, Yuanchun Shi, and Yuntao Wang. 2025. Computing with Smart Rings: A Systematic Literature Review. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 9, 3 (Sept. 2025), 137:1–137:54. doi:10.1145/3749480