Enabling gesture recognition on a resource-constrained device: a TinyML approach with envelope EMG data and real-time testing

Mohsin Ali Shah, Masood Ahmad Arbab, Shahkaar Ahmad Khan and Ansar Ullah

University of Engineering and Technology Peshawar, University Rd., Peshawar, 25120, Pakistan

Abstract. Gesture recognition from electromyography (EMG) is used to control prostheses and rehabilitation devices, but real-time classifiers must balance accuracy, latency, cost and computational load. This paper presents a tiny machine learning (TinyML) pipeline that classifies hand and arm gestures from envelope EMG on a low-cost microcontroller. Four-channel envelope EMG was recorded with a six-lead myoelectric module and an Arduino UNO from 10 healthy participants, who performed six gestures for 4 s each in two sessions. A joint accuracy-complexity search selected the window length, overlap and four time-domain features used to train an artificial neural network (ANN). On a 60/20/20 window-level training/validation/test split, the test accuracy was 98.35%. Five-fold cross-validation (97.37 ± 1.05%) indicated stable performance across data partitions, and leave-one-subject-out (LOSO) evaluation, the only subject-independent test, gave 97.22 ± 1.33%. Compared with a support vector machine (SVM), k-nearest neighbours (KNN), a decision tree and a random forest, the ANN had the lowest golden distance (GD = 0.0325). Deployed on a Raspberry Pi Pico in TensorFlow Lite format, the model uses 9.1% of flash memory and an estimated 10.6% of static random-access memory (SRAM), with a 23 ms inference time, a 254 ms total on-device cycle and a 12.9% active processing duty cycle. In a representative real-time test, with a PC relaying the EMG data to the Pico, per-gesture accuracy over 500 consecutive predictions was 95.8–99.6%. These results show that real-time envelope-EMG gesture recognition is feasible on a resource-constrained microcontroller; with validation on larger and more diverse populations, the approach may inform human-machine interfaces for prosthetics, rehabilitation and assistive devices.

Keywords: electromyography (EMG), TinyML, Raspberry Pi Pico, gesture classification, real-time testing

1. Introduction

Physiological signals carry a large amount of information about the human body. They are recorded either with invasive sensors, which are surgically implanted, or with non-invasive sensors placed on the skin, and are then sent to devices such as smartphones and computers for processing. The type and position of the sensor depend on the application [28]. Electromyography (EMG) measures the electrical activity of muscles through electrodes placed in or over the muscle. Surface EMG (sEMG), recorded with skin electrodes, is increasingly used because it is affordable, non-invasive and easy to use [32].

Gesture recognition supports natural human-robot interaction (HRI) [16], and EMG signals have been used to control prosthetic hands [17] and serious games [44]. In human-computer interaction, EMG-based gesture recognition allows users to control devices with their own muscle signals [50]. Gestures are also a natural way to interact in virtual reality (VR) [35], and EMG-based VR has been applied to motor rehabilitation [42].

Real-time EMG classification systems must balance accuracy, cost, latency and computational

0009-0002-7082-5780 (M. A. Shah); 0000-0002-3145-8124 (M. A. Arbab)
mohsinalishah13@gmail.com (M. A. Shah); arbabmasood@uetpeshawar.edu.pk (M. A. Arbab); shahkaa10@gmail.com (S. A. Khan); ansaruet@gmail.com (A. Ullah)

Received 2025-08-07 Accepted 2026-11-19 Published 2026-11-19

© Copyright for this article by its authors, published by the Academy of Cognitive and Natural Sciences. This is an Open Access article distributed under the terms of the Creative Commons License Attribution 4.0 International (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

load, and existing systems often satisfy some of these requirements at the expense of others. To address this trade-off, we developed a real-time hand and arm gesture classification system that uses envelope EMG signals (rectified and low-pass-filtered EMG) and is based on tiny machine learning (TinyML). The system is designed to provide sufficiently accurate real-time classification on low-cost hardware with a low active processing duty cycle. If validated with target users, low-cost EMG classification of this kind could make rehabilitation and assistive technologies, such as exoskeletons and artificial limbs, more accessible, and could support EMG-based research in fields such as sports science, neurology and biomechanics.

This work is an applied, deployment-oriented contribution: a complete pipeline from data acquisition to on-device inference on ultra-low-cost hardware. The study has five objectives: (1) to develop and deploy a TinyML system that recognises hand and arm gestures from envelope EMG signals on a resource-constrained microcontroller (Raspberry Pi Pico); (2) to select the window size, overlap ratio and time-domain feature subset through a joint accuracy-complexity search, so that the system is feasible for real-time embedded implementation; (3) to compare the proposed artificial neural network (ANN) with support vector machine (SVM), decision tree (DT), k-nearest neighbours (KNN) and random forest (RF) classifiers, using the golden distance (GD) [39] as a summary metric; (4) to assess the stability of the ANN across data partitions with 5-fold cross-validation and its cross-subject generalisation with leave-one-subject-out (LOSO) evaluation; and (5) to quantify the flash and static random-access memory (SRAM) use and the inference latency of the model on the Raspberry Pi Pico, and to test the system in real time.

Section 2 reviews EMG-based gesture recognition systems and the hardware on which they have been implemented. Section 3 describes the methods: data acquisition, pre-processing, segmentation, feature extraction, classification and deployment. Section 4 presents the offline evaluation, the on-device measurements and the real-time tests. Section 5 discusses the results, limitations and future work. Section 6 concludes the paper.

2. Literature review

Pipelines that extract features from short signal segments and classify them with conventional machine-learning classifiers have been applied to physiological signals other than EMG. Examples include classifying electroencephalography (EEG) recorded during 2D and 3D video viewing [37] and detecting transitions between 2D and 3D video segments [38] with SVM, KNN and linear discriminant analysis (LDA) classifiers, classifying sleep stages from EEG and electrooculography signals with spectral features and bagged-tree ensembles [40], and detecting cough and sneeze sounds from Mel-frequency cepstral coefficient features with SVM, DT and ensemble classifiers [41].

Many EMG-based gesture recognition studies have combined hand-crafted features with conventional machine-learning classifiers or shallow neural networks. Wan et al. [58] combined forearm EMG with pressure signals from an array of force-sensitive resistors on the back of the hand and classified 21 predefined hand gestures with a KNN classifier at 96.05% accuracy. Bagherian Azhiri, Esmaeili and Nourani [6] proposed feature functions computed on each level of a wavelet decomposition of the EMG signal; with a neural-network classifier, they achieved 95.5% accuracy for 10 gestures from only 800 ms of EMG. Afrin, Mahmud and Hasan [1] recorded EMG from healthy participants and participants with spinal cord injury performing gestures for controlling an electric wheelchair; with time-domain features from segmented signals, RF and DT classifiers achieved over 95% accuracy. Ghaffar Nia, Kaplanoglu and Nasab [24] classified four hand gestures with an ANN that reached 93% accuracy and outperformed long short-term memory (LSTM) networks, KNN, SVM and other methods. Fan et al. [21] extracted time-domain and time-frequency features from four-channel EMG, reduced them with principal component analysis (PCA) and classified six gestures with a DT at over 97% accuracy. Núñez-Montoya et al. [46] extracted 20 features from the EMG of four participants performing eight motions to control an anthropomorphic robotic hand; of eight machine-learning architectures, a multilayer perceptron gave the highest accuracy, 84.78%. Bhattacharjee and Singh

[11] evaluated eight time-domain features and six ensemble methods for eight hand gestures with training sets of different sizes; the best classifier-feature pair, XGBoost with integrated EMG (IEMG), reached 98.33% accuracy. A recent study [3] compared grid, random and Bayesian hyperparameter optimisation for KNN, DT, RF and SVM classifiers; KNN reached 98.4% accuracy with all three optimisers, and Bayesian optimisation gave the best overall trade-off between accuracy, adaptability to simulated electrode drift and noise tolerance. By contrast, Panda, Chakravarty and Moulik [47] used flex sensors and an Arduino UNO rather than EMG, recognising four gestures from finger-bending angles; their adversarial-learning approach reached 88.32% accuracy and outperformed conventional machine-learning classifiers.

The commercial Myo armband has been widely used to record sEMG. Benalcázar, Motoche et al. [9] and Benalcázar, Jaramillo et al. [8] classified Myo EMG in real time with KNN and dynamic time warping, achieving 89.5% and 86% accuracy, respectively, on the five gestures recognised by the Myo's proprietary system. Anil and Sreeletha [4] used discrete wavelet transform features and an SVM to recognise five gestures (wave in, wave out, fist, rest and finger spread), with 83% accuracy on 335 samples. Chen and Zhang [14] removed noise from raw Myo EMG, detected muscle-activity segments and classified five gestures with an SVM at 89% accuracy. Kaya and Kumbasar [33] recognised the digits 0–9 of Turkish Sign Language from seven time-domain features reduced by PCA and found that an SVM outperformed KNN and ANN classifiers. Karapinar Senturk and Bakay [32] compared six classifiers on seven hand gestures from a Myo armband dataset; naive Bayes (NB) performed best, with 96.43% accuracy and the lowest root-mean-square error (0.189). Young et al. [60] proposed a simplified pipeline (data collection, pre-processing, feature extraction, classification, post-processing and interfacing) that achieved 94.8% accuracy on the default Myo gestures for 10 participants. Chung and Benalcázar [15] combined an autoencoder for automatic feature extraction with a feedforward neural network and recognised the five Myo gestures in real time with 85.08% accuracy. Akhter et al. [2] evaluated logistic regression, LDA, KNN, DT and NB classifiers with 10-fold cross-validation on seven hand gestures and obtained 54–67% accuracy, with the highest precision for hand flexion and extension.

Other studies have used deep or evolved neural networks. Dweiri, Hajjar and Hatahet [19] used neuroevolution of augmenting topologies (NEAT) to evolve networks with gated recurrent units (GRUs) and achieved 88.76% accuracy for nine gestures from Ninapro DB2 with 150 ms windows. Nadjib, Bilal and Karima [45] combined a convolutional neural network (CNN), which extracts features from raw EMG, with an LSTM, which models temporal context, and classified six motions with 98.8% accuracy. Xiong et al. [59] fed instantaneous high-density EMG frames to four recurrent network variants built from LSTM or GRU cells; with a 100-frame majority vote, a bidirectional LSTM achieved the highest accuracy, 98.57%. More recently, Arsamy and Balasubramanian [5] proposed an optimised attention network for sEMG-based gesture recognition on edge artificial intelligence platforms to support HRI in industrial robotics, and Celik and Bayazit [13] combined CNNs, transfer learning and similarity learning to classify hand gestures from limited user sEMG data.

Several studies have run EMG gesture recognition on microcontroller-class processors. Zanghieri et al. [61] developed TEMPONet, a temporal convolutional network for real-time EMG classification. On the Ninapro DB6 dataset, TEMPONet achieved 65.2% inter-session accuracy when transient segments were excluded. On a 20-session dataset collected with the authors' own eight-channel embedded system, it reached 93.7% accuracy, and an 8-bit quantised version was deployed on the GAP8 processor. Mongardi et al. [43] proposed a low-power embedded system for real-time gesture recognition based on average threshold crossing: events generated whenever the EMG signal crossed a threshold were averaged over time windows and used as features. A neural network running on an Apollo2 microcontroller (Arm Cortex-M4F) classified six gestures from 25 healthy participants, recorded with the device described in [27], with 96.34% accuracy. Gomez-Bautista et al. [26] deployed neural-network and bagged-forest models on an ESP32-S3 module and showed that systematic feature selection can reduce execution time on embedded hardware while preserving model performance. Fasulo et al. [22] introduced TinyMyo, a Transformer-based EMG foundation model with 3.6 million parameters that supports several downstream tasks, including hand gesture

recognition, and deployed it on the GAP9 microcontroller.

Other systems ran on single-board computers or field-programmable gate arrays (FPGAs). Raurale, McAllister and Rincon [51] developed a real-time wrist-hand motion recognition system on a Raspberry Pi 3B+, in which time-domain features of Myo armband EMG were classified by a radial basis function neural network with over 99% accuracy. Triwiyanto et al. [56] used single-lead EMG and embedded machine learning on a Raspberry Pi 3B+ to control an upper-limb exoskeleton. Fukano et al. [23] trained a CNN on raw EMG recorded with a Myo armband for 53 gestures, reporting over 80% accuracy for some gestures, and deployed the trained model on a Raspberry Pi 4 to translate EMG into robot-hand movements in real time. Pedrosa and Costa [48] built an EMG-based MIDI controller for musical performance: signals from three forearm EMG sensors were passed through an Arduino to a Raspberry Pi 4B, where RF, SVM and neural-network classifiers were compared for recognising four hand gestures in real time (78% accuracy with the SVM), and each detected gesture was converted into a MIDI note for a digital audio workstation. Cedeño Z. et al. [12] developed a wearable system on a Cyclone V FPGA that helps people with speech impairments communicate customised phrases; with a single root-mean-square (RMS) feature of Myo armband EMG and a KNN classifier, it achieved 95% accuracy for nine gestures using 2 s of data per gesture. Kerdjidj et al. [34] implemented feature extraction and KNN classification as hardware blocks on a Xilinx Zynq-7000 FPGA (ZedBoard) for EMG recorded with a Myo armband at 200 Hz. Majolo and Balbinot [36] proposed a hardware multi-class SVM for fast sEMG classification, designed for a classification time below 25 ms on a Xilinx Kintex-7 FPGA; with 12-channel RMS features from the Ninapro database, the FPGA implementation classified 17 hand movements with 59.2% accuracy (58.2% in MATLAB) in a few milliseconds.

Most of these systems prioritised accuracy or real-time operation, and many ran on single-board computers (Raspberry Pi 3B+ and Raspberry Pi 4) [23, 48, 51, 56] or FPGA boards [12, 34, 36], which are more expensive than microcontroller boards such as the Raspberry Pi Pico. The microcontroller-based systems used processors with a hardware floating-point unit (FPU), digital signal processing (DSP) extensions or dedicated machine-learning accelerators: the Arm Cortex-M4F-based Apollo2 [43], the ESP32-S3 [26], and the GAP8 and GAP9 RISC-V processors [22, 61]. The RP2040 used in this work is an entry-level dual-core Arm Cortex-M0+ microcontroller with no hardware FPU or DSP extensions and 264 KB of SRAM, less on-chip memory than the GAP8 and GAP9. End-to-end models such as CNNs, LSTMs and lightweight Transformers like TinyMyo [22] learn features directly from raw EMG and achieve good accuracy, but they usually exceed the memory and compute budget of such a device; TinyMyo, for example, has about 3.6 million parameters.

This work differs from the GAP8/GAP9-, Apollo2- and ESP32-S3-based systems [22, 26, 43, 61] in two ways beyond the choice of hardware. First, the RP2040 imposes a tighter accuracy-complexity trade-off, so instead of fixing the window and overlap in advance, the joint search in section 3.4 selects the window size, overlap and feature subset under this constraint. Second, this optimisation is combined with on-device real-time testing, subject-independent LOSO validation and a comparison of several classifiers summarised by a single measure (GD). To our knowledge, this combination has not previously been reported for this class of ultra-low-cost hardware.

3. Methods

The proposed pipeline has five stages, shown in figure 1: data collection and pre-processing, data segmentation, feature extraction, classification, and deployment.

3.1. Data collection and pre-processing

Envelope EMG data were recorded with a 6-lead myoelectric module and an Arduino UNO (figure 2). The module's instrumentation amplifiers have a gain of about 1000 V/V, a common-mode rejection ratio of at least 80 dB and input noise below 1 μV RMS in the 20–500 Hz band. The module full-wave rectifies the amplified signal and extracts a smoothed envelope with an on-board low-pass filter

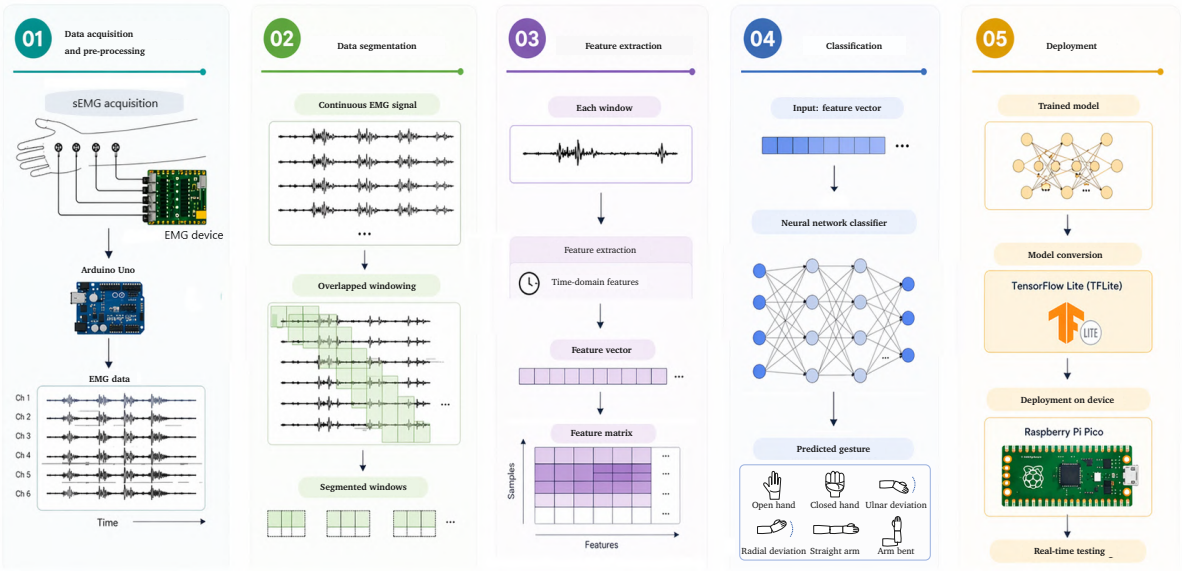


Figure 1: Workflow of the proposed study.

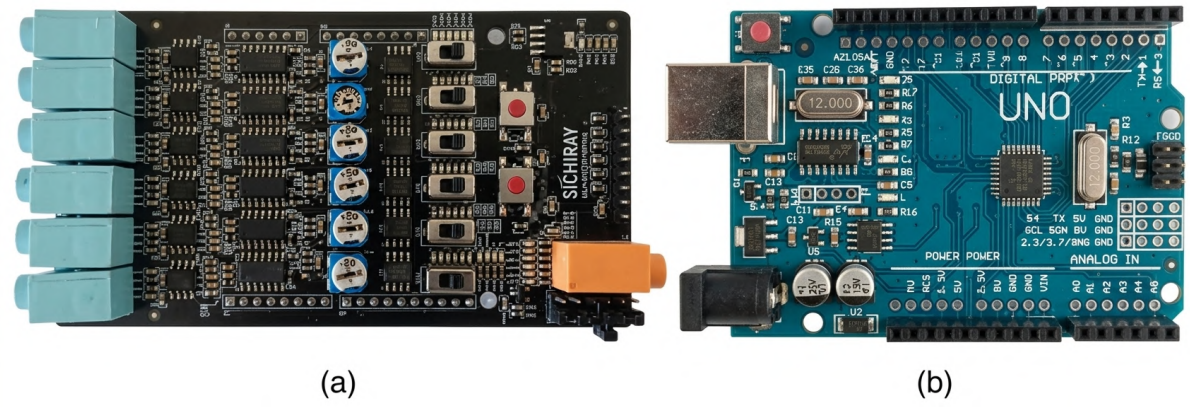


Figure 2: (a) 6-lead myoelectric module and (b) Arduino UNO.

(cut-off about 5 Hz); because this envelope is already a low-frequency signal, no further software filtering was applied before windowing.

To keep the number of acquisition channels small, four electrode pairs and one reference electrode were used (figure 3): channel 1 on the flexor carpi ulnaris (anterior forearm), channel 2 on the extensor carpi ulnaris (posterior forearm), channel 3 on the biceps brachii (anterior upper arm) and channel 4 on the triceps brachii (posterior upper arm), with the reference electrode on the elbow joint.

Data were collected from 10 healthy subjects in two sessions. In each session, the six gestures shown in figure 4 were each recorded for 4 s at a sampling frequency of $f_s = 1000$ Hz, giving 4,000 samples per recording. The six classes are open hand (class 0), closed hand (class 1), wrist extension (class 2), wrist flexion (class 3), straight arm (class 4) and bent arm (class 5). The 120 recordings (10 subjects × 2 sessions × 6 gestures) contain 480,000 samples, and the classes are balanced, with 80,000 samples (16.67%) each.

3.2. Data segmentation

Segmentation is the first processing step: the continuous EMG signal is divided into fixed-length windows, which are analysed separately. Windows can be disjoint (non-overlapping) or overlapping,

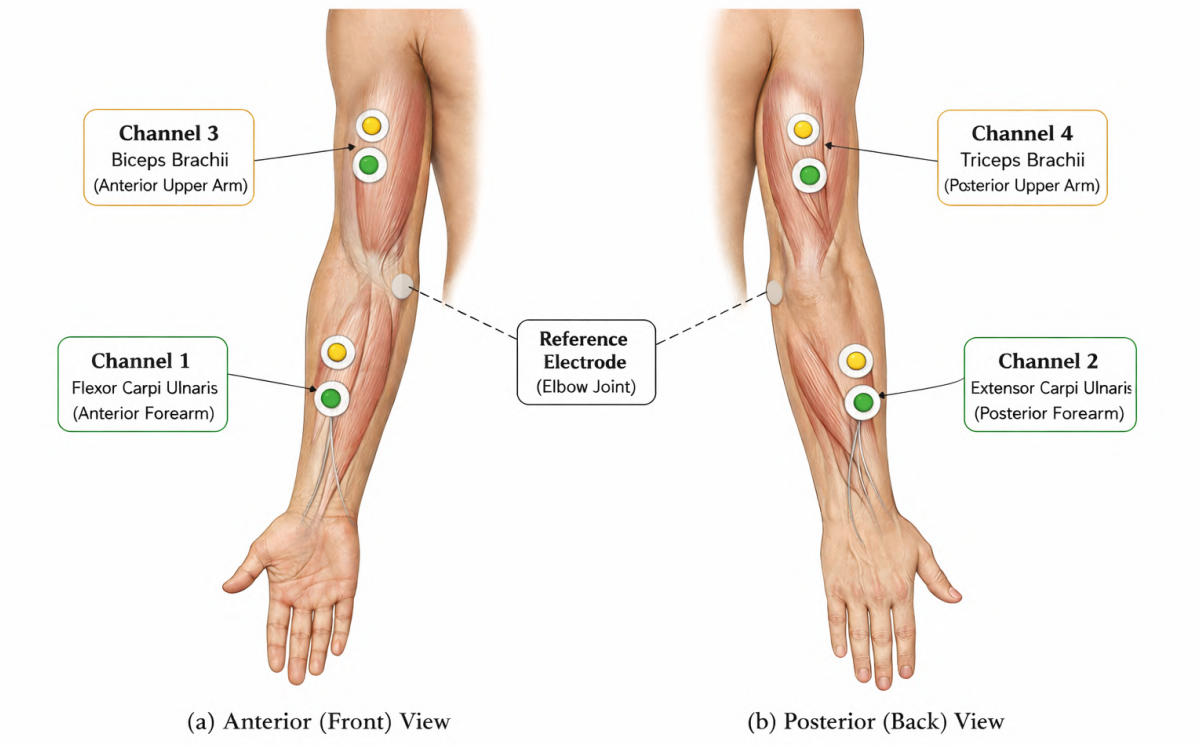


Figure 3: Electrode placement.

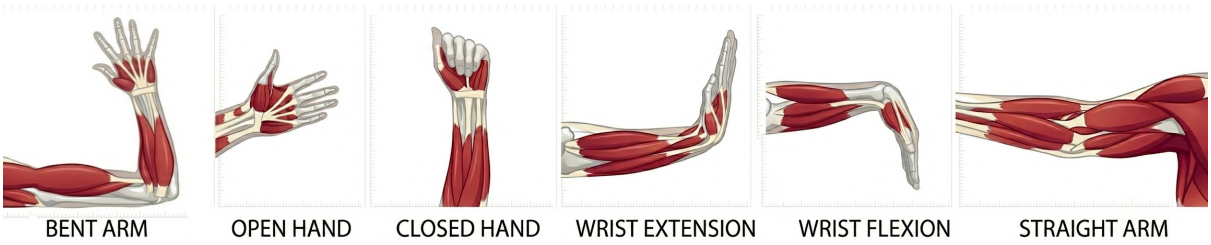


Figure 4: The six recorded gestures.

in which case consecutive windows share samples [18].

This study uses overlapping windows, with the overlap expressed as a percentage of the window length. Consecutive windows start one step apart (eq. (1)); for example, a 100 ms window with 50% overlap has a 50 ms step, so consecutive windows share 50 ms of data. Overlapping windows give more frequent classification updates and preserve temporal information between adjacent segments. The window length and overlap were selected by the joint accuracy-complexity search in section 3.4.

3.3. Feature extraction and selection

Features are computed for each channel of every window produced by segmentation. Seven time-domain features were considered: minimum (MIN), maximum (MAX), mean absolute value (MAV), root mean square (RMS), integrated EMG (IEMG), slope sign changes (SSC) and zero crossings (ZC). The subset used for classification was chosen by the joint accuracy-complexity search in section 3.4, which ranks configurations by the composite score of eq. (7), combining cross-validation (CV) accuracy with the total number of floating-point operations (FLOPs) over the recording. The search selected MIN, MAX, MAV and RMS. For a window of N samples $x_1, \ldots, x_N$ from one channel, these features are

$$\mathrm{MIN} = \min_{1 \le i \le N} x_i, \qquad \mathrm{MAX} = \max_{1 \le i \le N} x_i,$$

$$\text{MAV} = \frac{1}{N}\sum_{i=1}^{N}|x_i|, \qquad \text{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2}$$

MAV and RMS both measure the signal amplitude in the window; RMS is the square root of the mean signal power.

3.4. Windowing and feature optimisation via joint search

Window length, overlap and feature subset were selected jointly to maximise classification accuracy while minimising computational cost. The search covered window lengths of {100, 200, 300, 500, 700, 750, 1000} ms and overlaps of {0, 10, 20, 25, 30, 40, 45, 50, 55, 60, 70, 75}%, combined with subsets of up to four of the seven candidate features (the limit reduces computational cost). Each candidate configuration was evaluated by training the ANN for 80 epochs with early stopping on a held-out validation set.

3.4.1. Windowing parameters

Windows were extracted with the sliding-window scheme of Englehart and Hudgins [20]. Consecutive windows are separated by a step of

$$\text{Step (ms)} = \text{Win (ms)} \times \left(1 - \frac{\text{Overlap}}{100}\right) \tag{1}$$

where Overlap is expressed as a percentage of the window length. The number of windows n obtained from a recording of duration D is

$$n = \left\lfloor \frac{D - \text{Win}}{\text{Step}} \right\rfloor + 1 \tag{2}$$

With $f_s = 1000$ Hz, a window of Win ms contains $N = \text{Win}$ samples per channel. The duration $D = 480{,}000$ ms is the total length of the 120 recordings (480,000 samples at 1000 Hz) and is the same for every configuration.

3.4.2. FLOPs-based complexity metric

Computational cost was measured in FLOPs, a hardware-independent measure commonly used in the EMG processing literature [20, 29, 49, 62]. Each time-domain feature has $O(N)$ complexity, and each was assigned an arithmetic cost coefficient k equal to its number of elementary operations per sample (table 1).

Table 1
Per-sample arithmetic cost coefficient k for each time-domain EMG feature [49, 62].

Feature	Operation per sample	Operations	k
MIN	Track running minimum	1 comparison	1
MAX	Track running maximum	1 comparison	1
MAV	$\lvert x_i\rvert$, accumulate	1 abs + 1 add	2
RMS	x_i^2, accumulate	1 mul + 1 add	2
IEMG	$\lvert x_i\rvert$, accumulate	1 abs + 1 add	2
SSC	$(x_i - x_{i-1})(x_i - x_{i+1})$ sign check	2 sub + 1 mul + 1 cmp	4
ZC	$\text{sign}(x_i \cdot x_{i+1})$ + threshold	1 mul + 1 sub + 1 abs + 1 cmp	4

The cost of one window over all $C = 4$ channels is

$$\text{FLOPs}_{\text{perWin}} = C \times N \times \sum k \tag{3}$$

where the sum runs over the features in the subset, and the total cost over the recording is

$$\text{FLOPs}_{\text{total}} = n \times \text{FLOPs}_{\text{perWin}} \tag{4}$$

A larger overlap shortens the step and increases n, so the total FLOPs grow for a fixed duration D, whereas the FLOPs per window do not change [20, 49].

3.4.3. Normalisation and combined scoring

Both metrics were min-max normalised to $[0, 1]$ [49, 62]. With CV_i and FLOPs_i denoting the CV accuracy and total FLOPs of configuration i,

$$A_{\text{norm}} = \frac{\text{CV}_i - \text{CV}_{\text{min}}}{\text{CV}_{\text{max}} - \text{CV}_{\text{min}}} \tag{5}$$

$$C_{\text{norm}} = \frac{\text{FLOPs}_i - \text{FLOPs}_{\text{min}}}{\text{FLOPs}_{\text{max}} - \text{FLOPs}_{\text{min}}} \tag{6}$$

The composite score weights accuracy (the primary objective) and complexity (real-time feasibility):

$$\text{Score} = 0.6 \times A_{\text{norm}} + 0.4 \times (1 - C_{\text{norm}}) \tag{7}$$

A score of 1.0 would be reached only by a configuration with both the highest accuracy and the lowest cost in the evaluated set. The weights of 0.6 for accuracy and 0.4 for complexity reflect that correct classification is the main goal, while real-time deployability constrains the solution space [20, 29].

3.4.4. Joint search results

Table 2 lists the ten highest-scoring configurations. Configuration 1 (window 500 ms, overlap 55%, features MIN, MAX, MAV and RMS) ranks first, with a combined score of 0.9239, the highest CV accuracy (96.66 ± 1.40%) and the third-lowest computational cost (25.58M FLOPs). All subsequent training experiments used this configuration.

Table 2
Combined accuracy-complexity ranking of the ten highest-scoring configurations (f_s = 1000 Hz, C = 4 channels, D = 480,000 ms). FLOPs were computed with eqs. (3)–(4), C_{norm} with eq. (6) and the score with eq. (7), using A_{norm} from eq. (5); the minima and maxima used for normalisation are taken over the ten listed configurations. CV Acc is the cross-validation accuracy; the asterisk marks the selected configuration.

Rank	Win (ms)	Overlap (%)	Step (ms)	Features	FLOPs/ window	Total FLOPs	C_{norm}	CV Acc (%)	Score
1*	500	55	225	MIN+MAX+MAV+RMS	12,000	25.58M	19.0%	96.66 ± 1.40	0.9239
2	100	25	75	MAX+MAV+ZC+IEMG	3,600	23.04M	14.3%	93.37 ± 0.56	0.5542
3	300	75	75	MIN+SSC+IEMG+RMS	10,800	69.09M	100%	96.23 ± 1.85	0.5492
4	200	50	100	MAX+RMS+SSC+IEMG	7,200	34.55M	35.7%	93.88 ± 0.50	0.5287
5	200	25	150	MIN+MAX+RMS+IEMG	4,800	15.36M	0.0%	91.58 ± 2.18	0.4000
6	100	75	25	MIN+MAX+SSC+RMS	3,200	61.43M	85.7%	94.40 ± 1.22	0.3901
7	300	50	150	MIN+MAX+ZC+RMS	9,600	30.71M	28.6%	92.42 ± 1.45	0.3849
8	700	70	210	MAX+MAV+ZC+RMS	25,200	57.53M	78.5%	93.46 ± 1.68	0.3081
9	500	75	125	MIN+MAV+SSC+RMS	18,000	69.07M	100%	93.88 ± 1.59	0.2718
10	100	75	25	MIN+MAX+ZC+IEMG	3,200	61.43M	85.7%	92.43 ± 0.72	0.1574

Figure 5 shows the combined scores of the ten configurations. Configuration 1 (0.9239) scores 0.37 higher than the second-ranked configuration (0.5542) and therefore offers the best accuracy-cost trade-off under the 0.6/0.4 weighting. Lower-ranked configurations score lower because of higher total FLOPs, lower accuracy, or both.

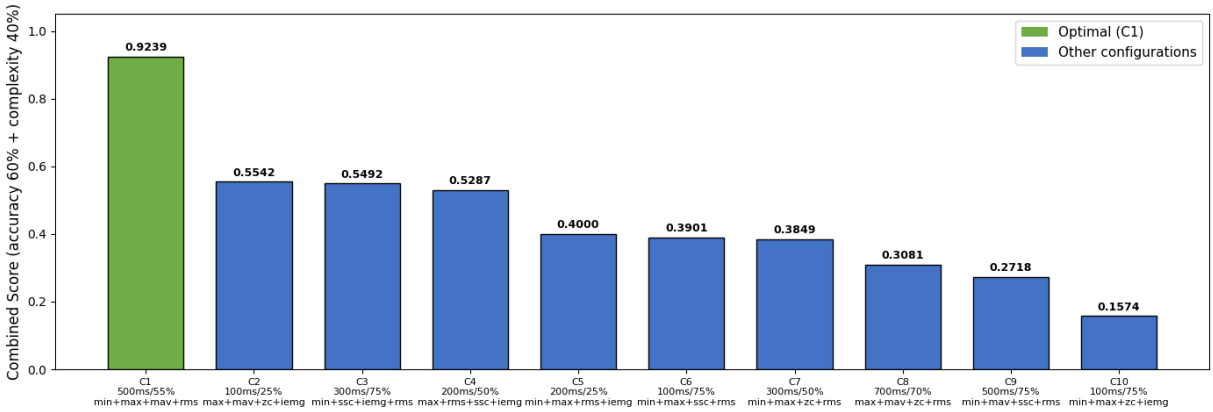


Figure 5: Combined scores of the ten configurations listed in table 2.

3.5. Classification

Segmentation with 500 ms windows, 55% overlap and a 225 ms step yielded $n = 2{,}125$ windows. Features were extracted from each window (16 inputs: 4 features × 4 channels), and the 60/20/20 split was applied at the window level, giving 1,275 training, 425 validation and 425 test windows. The ANN has three fully connected hidden layers of 256, 128 and 64 neurons with ReLU activations and a six-neuron softmax output layer, one neuron per gesture class. For regularisation, dropout rates of 0.30, 0.25 and 0.20 were applied after the first, second and third hidden layers, respectively, together with L2 weight decay ($\lambda = 0.001$); batch normalisation was not used. The network was trained with the Adam optimiser at a fixed learning rate of 0.0008 for up to 400 epochs, with an early-stopping callback that stopped training once the validation accuracy reached 99%. The test set was used only for the final evaluation.

3.6. Model deployment

Distributed intelligence in edge computing and the Internet of Things (IoT) requires machine learning (ML) on embedded devices with limited resources. Following [31, 52], we use the term TinyML for the deployment of a trained ML model on microcontroller hardware that is constrained in memory, computing resources and power. TinyML does not imply end-to-end feature learning: the model may use hand-crafted or automatically learned features. TinyML has made it easier to run ML on low-cost, power- and resource-constrained devices such as the Raspberry Pi Pico [7, 31, 52]; for example, the Arduino Nano 33 BLE has been used for image classification [10] and gas leak detection [25], and the Nordic Semiconductor nRF52840 for activity recognition in elderly care [55]. In this study, the model was deployed on a Raspberry Pi Pico (figure 6) using the Arduino IDE.

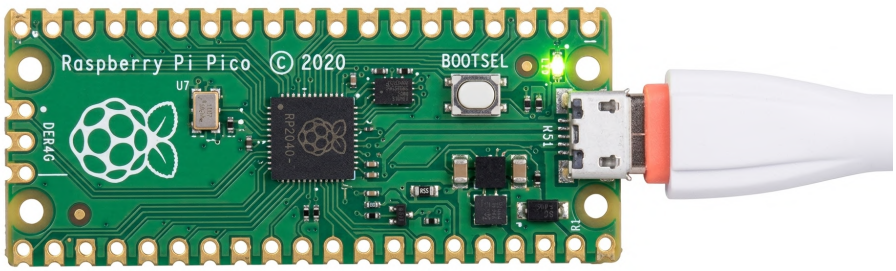


Figure 6: Raspberry Pi Pico.

3.6.1. Conversion to TensorFlow Lite

For deployment on the microcontroller, the trained ANN was converted to the TensorFlow Lite (TFLite) format and kept in 32-bit floating point (float32).

3.6.2. Configuring the Arduino IDE for the Raspberry Pi Pico

Support for the Raspberry Pi Pico board, including the libraries and tools needed to program it, was first added to the Arduino IDE through the Boards Manager. The TensorFlow Lite Micro library for Arduino, which provides the functions needed to load and run TFLite models on microcontrollers, was then installed manually following [54], because it is no longer listed in the Arduino Library Manager.

3.6.3. Deployment on the Raspberry Pi Pico

A custom Python function converted the TFLite model into a C byte array and saved it in a header (`.h`) file. This header was included in the Arduino sketch, which was compiled and uploaded to the Raspberry Pi Pico with the Arduino IDE. This step confirmed that the TFLite model runs on the microcontroller within the Arduino environment.

3.7. Real-time evaluation set-up and protocol

For real-time testing, a PC running a Jupyter notebook acted as a bridge between the EMG device and the Raspberry Pi Pico, both connected to the PC via USB (figure 7). The notebook received the envelope EMG data from the EMG device and forwarded them to the Pico, which performed windowing, feature extraction and inference and returned the predicted gesture to the notebook for display. The Pico therefore performs on-device inference, but the pipeline as a whole, from EMG acquisition to the displayed prediction, is not yet a standalone embedded system, because the PC is needed to relay data between the EMG hardware and the Pico. A self-contained, PC-free implementation is discussed as future work in section 5.1.

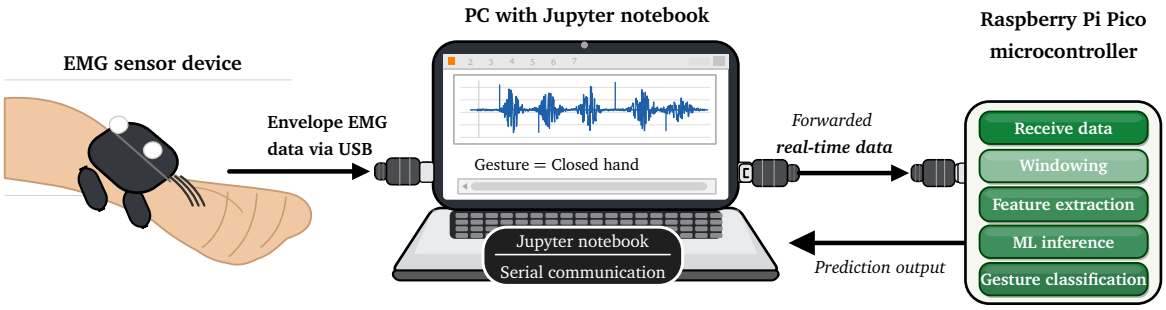


Figure 7: Bridge set-up for real-time testing: a PC running a Jupyter notebook relays envelope EMG data from the EMG device to the Raspberry Pi Pico and receives its predictions.

Real-time accuracy was assessed with a threshold-based onset detection and sustained-recognition protocol. Three consecutive identical predictions of the same gesture class were taken as an indication that the contraction had reached a steady state; the next 500 predictions were then recorded and compared with the known gesture label to compute the recognition accuracy. This design follows two common practices in surface EMG gesture recognition. First, classifiers are usually evaluated on steady-state EMG rather than on the transient signals at gesture onset and offset, because steady-state segments give more stable and repeatable outputs [20, 30]. Second, requiring a minimum number of consistent predictions before a decision is accepted, as in threshold-based activation schemes [63] and majority-voting frameworks [53, 57], suppresses errors caused by noise and by transitions between gestures.

Because the accuracy is computed over 500 consecutive predictions rather than a single window, any systematic misclassification, signal drift or model instability during the evaluation period lowers the reported value. No standard protocol for real-time evaluation of EMG gesture recognition has yet been adopted [30], which limits direct comparison between studies.

4. Experiment and results

The windows were divided into training (60%), validation (20%) and test (20%) sets as described in section 3.5; the validation set was used to monitor training and adjust the hyperparameters, and the test set was used only for the final evaluation.

Training stopped at epoch 283, when the validation accuracy reached the 99% early-stopping threshold (figure 8). At that epoch, the training and validation accuracies were 99.06% and 99.29%, and the training and validation losses were 0.0896 and 0.0804, respectively.

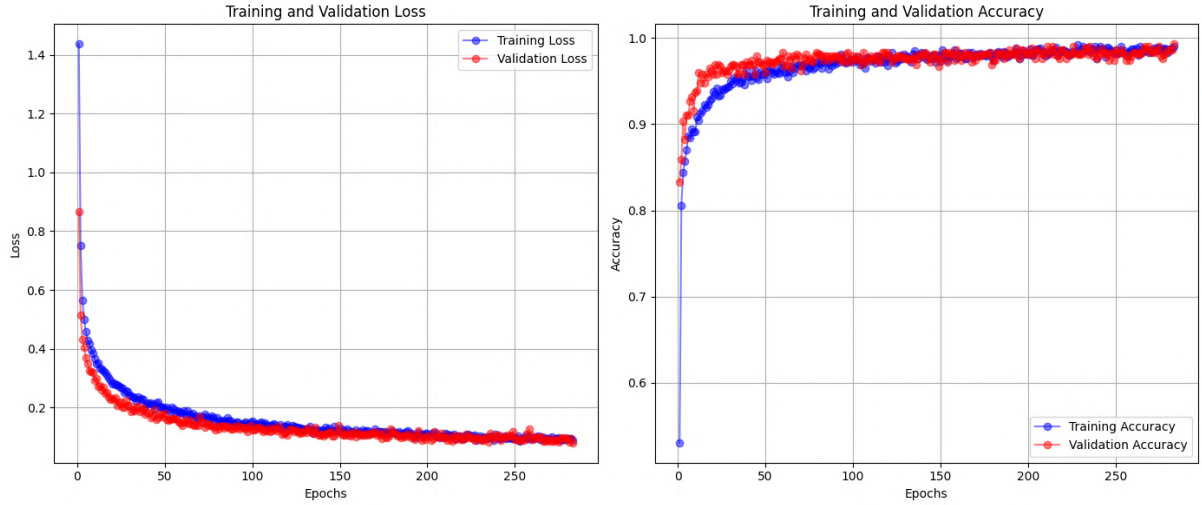


Figure 8: Training and validation loss and accuracy over epochs.

On the test set (425 windows), the ANN achieved an accuracy of 98.35%, a precision of 98.40%, a recall of 98.35% and an F1-score of 98.35% (figure 9; precision, recall and F1-score are weighted averages over the six classes). These are window-level estimates: because the 60/20/20 split was applied to overlapping windows drawn from the same continuous recordings (55% overlap; section 3.4), windows in different subsets can share many samples and are not statistically independent. The test result should therefore be read as an intra-subject, window-level estimate.

The LOSO evaluation in section 4.4, in which entire subjects are held out, gives a more conservative estimate of performance for a new user. The confusion matrix (figure 10) shows that 418 of the 425 test windows were classified correctly. Four of the seven errors were open-hand windows predicted as bent arm; the other three were confusions among open hand, closed hand and wrist flexion. All wrist-extension, straight-arm and bent-arm windows were classified correctly.

4.1. Comparison with traditional classifiers

To check the choice of the ANN against simpler options, four traditional classifiers were trained with the same features and windowing parameters: SVM, DT, KNN and RF. Table 3 summarises the results. The ANN achieved the highest test accuracy (98.35%), 2.11 percentage points above SVM and RF (both 96.24%); KNN and DT reached 96.00% and 93.41%, respectively. DT had the lowest accuracy, possibly because a single decision tree is less expressive than an ensemble such as RF.

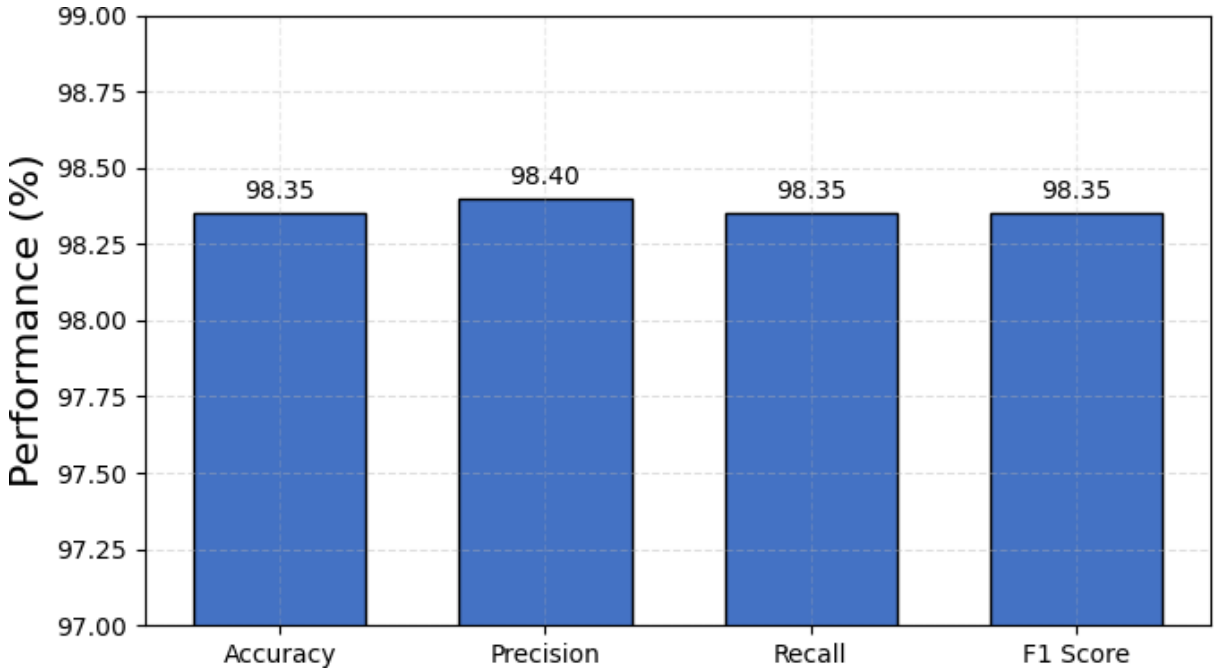


Figure 9: Accuracy, precision, recall and F1-score of the ANN on the test set (425 windows).

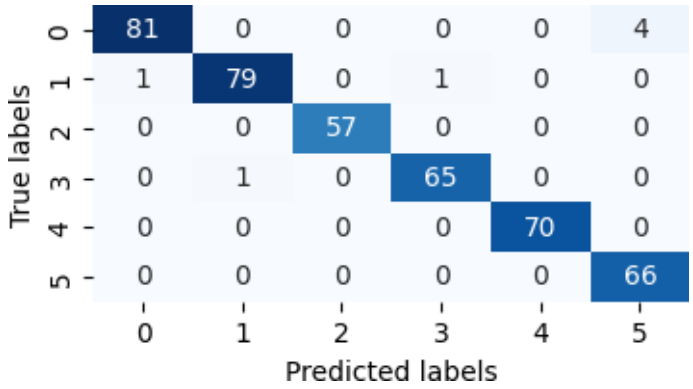


Figure 10: Confusion matrix of the ANN on the test set. Classes 0–5: open hand, closed hand, wrist extension, wrist flexion, straight arm, bent arm.

Table 3
Test accuracy of the ANN and four traditional classifiers with identical features (MIN+MAX+MAV+RMS, Win = 500 ms, overlap = 55%).

Classifier	Test accuracy (%)
SVM	96.24
Decision tree	93.41
KNN	96.00
Random forest	96.24
ANN (proposed)	98.35

4.2. Golden distance evaluation

The golden distance (GD) [39] summarises several performance metrics in a single value. Each classifier is represented by the point (Av-NAV, Av-DCO), where Av-NAV is the averaged normalised area value and Av-DCO is the averaged distance of the centroid from the origin. GD is the Euclidean distance between this point and the ideal point (1, 0): $\text{GD} = \sqrt{(1-\text{Av-NAV})^2 + \text{Av-DCO}^2}$. A lower

GD indicates performance closer to that of an ideal classifier. In this study, GD was computed from accuracy, precision, recall and F1-score with the seven-step procedure of Melek, M. and Melek [39]; table 4 lists the results.

Table 4
Golden distance (GD) of all classifiers (lower is better). Rating scale from [39]: < 0.2 Excellent, 0.2–0.4 High, 0.4–0.6 Moderate, > 0.6 Low.

Classifier	GD	Av-NAV	Av-DCO	Rating
SVM	0.0738	0.9262	0.0000	Excellent
Decision tree	0.1273	0.8727	0.0001	Excellent
KNN	0.0785	0.9215	0.0000	Excellent
Random forest	0.0739	0.9261	0.0001	Excellent
ANN (proposed)	0.0325	0.9675	0.0001	Excellent

With Av-NAV = 0.9675 and Av-DCO = 0.0001, the ANN has the lowest GD (0.0325), meaning that its multi-metric performance polygon is the largest and best centred of the five classifiers. All five classifiers fall in the excellent range (GD < 0.2). The GD of the ANN is about 56% lower than that of the best traditional classifier (SVM, 0.0738).

4.3. 5-fold cross-validation

The stability of the ANN (256→128→64→6) across data partitions was assessed with 5-fold cross-validation, with training for up to 250 epochs in each fold. Figure 11 shows the per-fold validation accuracy (96.00–98.12%). The mean was 97.37% ± 1.05%, with a 95% confidence interval (CI) of [96.07%, 98.67%], indicating consistent performance across folds. Generalisation to unseen subjects is assessed separately by the LOSO evaluation (section 4.4). The ANN was selected for LOSO evaluation and deployment on the basis of its test accuracy (table 3), its GD (table 4) and its compatibility with TFLite.

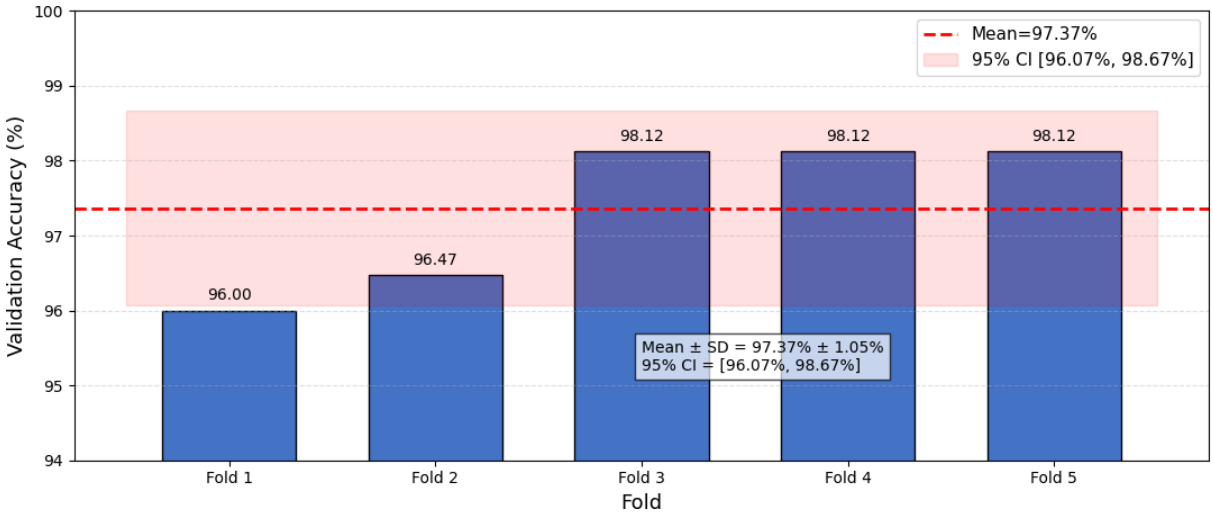


Figure 11: Validation accuracy of the ANN in each fold of the 5-fold cross-validation, with the mean and 95% confidence interval.

4.4. Leave-one-subject-out (LOSO) evaluation

To evaluate cross-subject generalisation, the model was trained on data from nine subjects and tested on the held-out subject; this was repeated for each of the ten subjects. This protocol gives a stricter estimate of performance for a new user than the window-level test split. The mean LOSO

accuracy was 97.22% ± 1.33% (95% CI: [96.27%, 98.17%]; figure 12), 1.13 percentage points below the window-level test accuracy (98.35%). Per-subject accuracy ranged from 95.75% to 99.53%, indicating reasonable cross-subject generalisation within this 10-subject dataset. Subjects 6 and 10 had the highest accuracy (99.53% and 99.06%) and subjects 2 and 5 the lowest (95.75% each), which may reflect inter-subject variability in EMG envelope patterns.

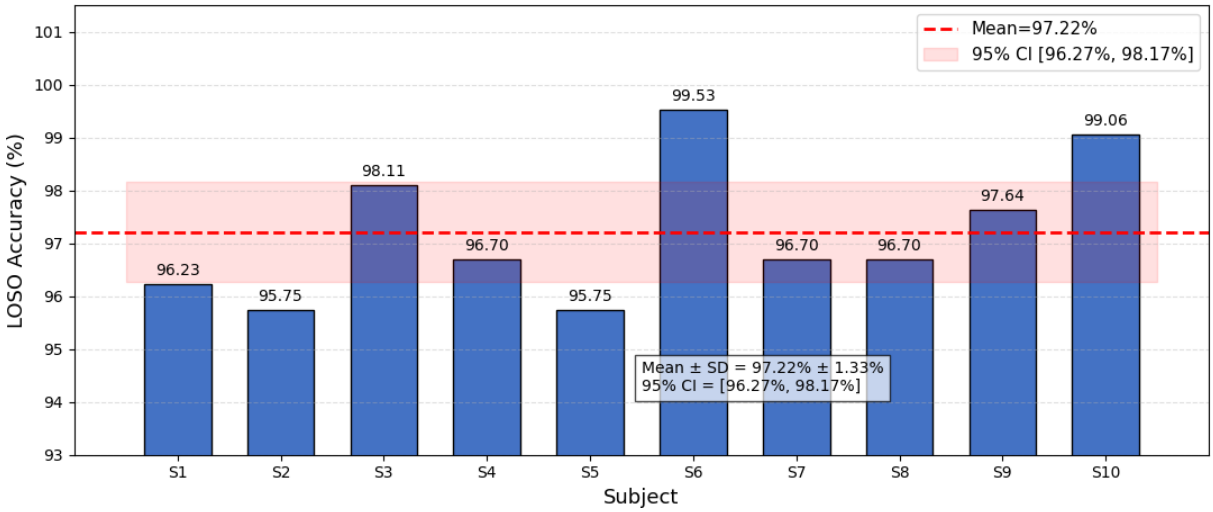


Figure 12: LOSO accuracy for each held-out subject (S1–S10), with the mean and 95% confidence interval.

4.5. TinyML deployment metrics

Table 5 summarises the resource use of the TFLite model on the Raspberry Pi Pico (RP2040, 125 MHz, float32). Three timing quantities must be distinguished: the window length (500 ms), the step size (225 ms) and the on-device processing time.

Table 5
TinyML resource use on the Raspberry Pi Pico. Model weights are stored in flash (ROM); SRAM holds activation tensors, on-device windowing buffers and extracted feature vectors.

Metric	Value	% of budget / notes
Trainable parameters	45,894	—
TFLite model size (flash/ROM)	185.75 KB	9.1% of 2048 KB
Estimated activation RAM (SRAM)	∼27.9 KB	10.6% of 264 KB
Inference latency @ 125 MHz (float32)	23 ms	—
On-device windowing time	0.433 ms	—
On-device feature extraction time	5.7 ms	—
Total on-device cycle	254 ms	—
Active processing duty cycle	12.9%	29.13 ms active / 225 ms interval
Classification update rate	Every 225 ms (∼4.4 Hz)	—

Windowing, feature extraction and inference all run on the Raspberry Pi Pico. In each step, the RP2040 receives 225 ms of new EMG samples, which are combined with the preceding 275 ms held in a rolling buffer to form a 500 ms window. It then performs windowing (0.433 ms), computes the 16-dimensional feature vector (4 features × 4 channels; 5.7 ms) and runs the TFLite model (23 ms), a total of 29.133 ms of processing. The latency from the start of a step to its prediction is therefore 254 ms (225 ms of acquisition plus 29.133 ms of processing), which is the total on-device cycle in table 5. Because processing finishes well within the next 225 ms acquisition interval, a new classification is produced every 225 ms (an update rate of about 4.4 Hz). Processing occupies 12.9% of each 225 ms interval, and the microcontroller is idle for the remaining 87.1%. This duty cycle is

a timing-based estimate; power consumption was not measured and would require direct current measurements (section 5.1).

4.6. Real-time testing

Real-time classification was tested with the bridge set-up in figure 7, following the protocol described in section 3.7. In the test shown in figure 13, the response time of each prediction was 22–23 ms, and the accuracy over 500 consecutive predictions ranged from 95.80% (closed hand) to 99.60% (open hand) across the six gestures.

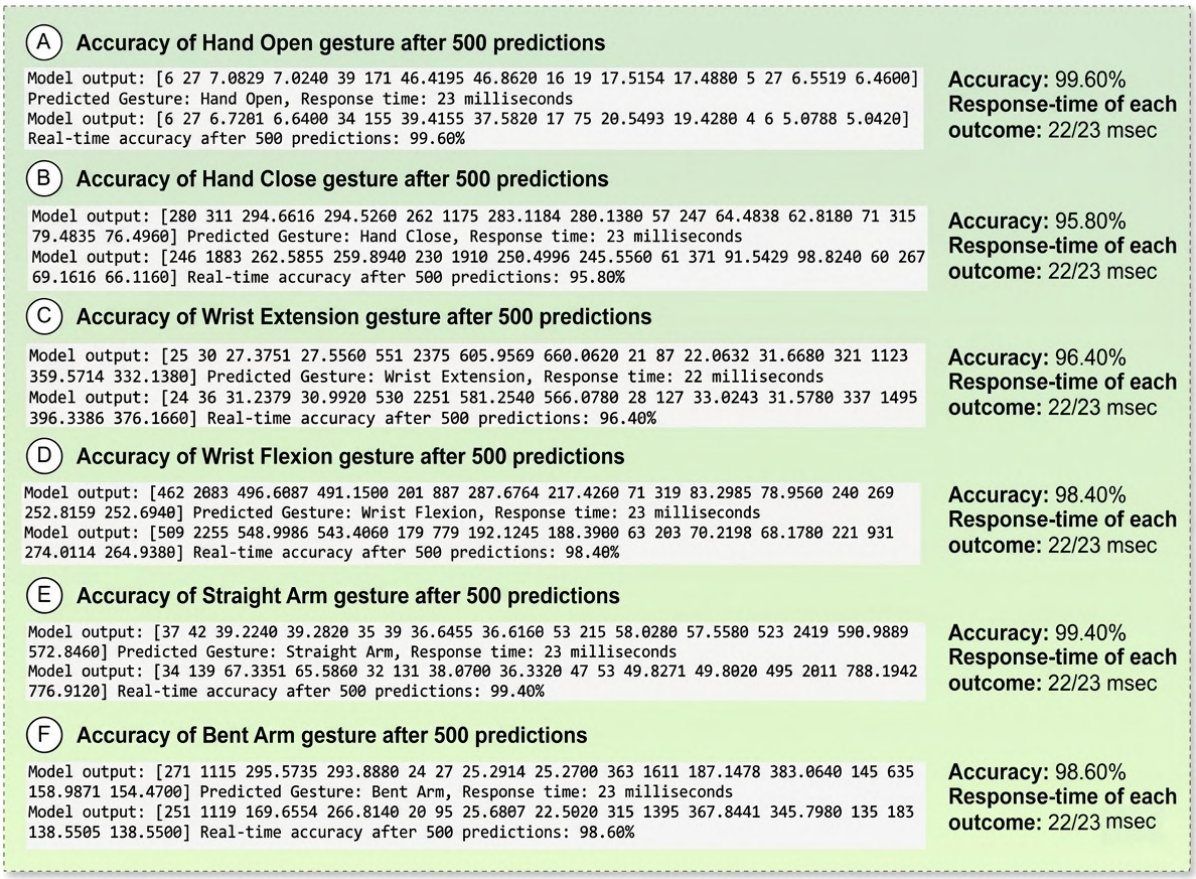


Figure 13: Real-time classification of the six gestures, (A)–(F). Each snapshot shows the last (500th) prediction and the accuracy after 500 predictions; each "Model output" line gives the input feature vector and is followed by the predicted gesture and response time or by the accuracy.

5. Discussion

The learning curves (figure 8) show that both losses fell steeply at first and decreased only slowly after about epoch 80, while both accuracies rose quickly and then levelled off. The training and validation curves remain close throughout, with no sign of overfitting at the window level.

With identical features, the ANN's test accuracy was 2.11 percentage points higher than that of the next-best classifiers, SVM and RF (96.24%; section 4.1). SVM and RF remain competitive in accuracy, but deploying them on the RP2040 would require a separate porting step, such as hand-written C inference code or a model-to-C conversion tool, because the TensorFlow Lite Micro toolchain used in this study does not support them natively. This reflects the available deployment tooling rather than a hardware limitation; classical classifiers have been implemented in embedded hardware before, for example KNN and SVM on FPGA platforms [12, 34, 36]. The ANN was chosen for its higher

accuracy and its direct fit with the TensorFlow Lite Micro pipeline used in this project, not because the alternatives cannot be deployed. The GD analysis (section 4.2), which combines several metrics into a single measure as proposed by Melek, M. and Melek [39], supports this choice: the ANN has the lowest GD (0.0325, against 0.0738–0.1273 for the traditional classifiers).

The cross-validation and LOSO results (sections 4.3–4.4) indicate how reliable the accuracy estimates are. The 5-fold cross-validation mean (97.37% ± 1.05%) and the LOSO mean (97.22% ± 1.33%) both lie within 1.2 percentage points of the window-level test accuracy (98.35%), and their narrow 95% confidence intervals ([96.07%, 98.67%] and [96.27%, 98.17%]) indicate stable estimates across data partitions and across subjects, respectively. Because the test split operates at the window level, whereas LOSO holds out entire subjects, LOSO is treated as the primary estimate of subject-independent performance. The window-level result is a best-case estimate, because windows from the same recording may be correlated.

The deployed model uses 9.1% of the flash memory and an estimated 10.6% of the SRAM and has a 12.9% processing duty cycle (section 4.5), which leaves ample headroom on the RP2040 for sensor interfacing, communication and application logic. It costs less than the single-board-computer and FPGA systems discussed in section 2 and runs on a simpler processor than the GAP8, GAP9, Apollo2 and ESP32-S3 used in earlier microcontroller deployments. The results suggest that TinyML is a practical route to low-cost, resource-efficient EMG applications, although direct power measurements and validation in the target use cases are still required.

Under the demonstration-level protocol described in section 3.7, predicted and actual gestures agreed closely across repeated real-time trials; figure 13 shows one representative test. Because the field has not yet converged on a standard real-time evaluation protocol, these results show consistent behaviour under our test conditions rather than general real-world reliability. They support further development towards gesture-recognition applications, but additional validation is needed to establish suitability for such use.

5.1. Limitations and future work

This study has several limitations. First, the dataset includes only 10 healthy subjects, which limits generalisation; future work should collect data from a larger and more diverse population, including people with limb differences or neurological conditions. Second, real-time testing used a demonstration-level ground-truth protocol (section 3.7); future studies should use standardised evaluation protocols. Third, the real-time set-up still relies on a PC bridge; a fully mobile, PC-free system that can be used continuously in different locations remains a future goal. Fourth, energy and power consumption were not measured; future work will use current profiling with a power analyser to quantify the average power of the RP2040 during inference and when idle. Fifth, the integration of EMG with other physiological signals in multimodal systems should be examined. Sixth, the ANN was kept compact so that it could fit and run on the Raspberry Pi Pico; whether a compressed CNN can also run on the same hardware remains to be tested. Seventh, a fair comparison with the Ninapro dataset is planned; because most Ninapro databases provide raw multichannel sEMG recordings, whereas this study uses a hardware-rectified, envelope-filtered signal chain, the comparison will require either deriving an equivalent envelope signal from the Ninapro recordings or collecting new data with matched instrumentation.

6. Conclusion

This study developed and deployed a TinyML gesture-recognition system for envelope EMG on a low-cost, resource-constrained microcontroller, the Raspberry Pi Pico. A joint accuracy-complexity search selected a 500 ms window with 55% overlap and four time-domain features (MIN, MAX, MAV and RMS). The ANN reached a window-level test accuracy of 98.35%, a 5-fold cross-validation accuracy of 97.37% ± 1.05% and a LOSO accuracy of 97.22% ± 1.33%; the LOSO result is the more realistic estimate for a new user. Compared with SVM, DT, KNN and RF, the ANN had the highest test

accuracy and the lowest golden distance (GD = 0.0325). The deployed TFLite model uses 9.1% of the flash memory and an estimated 10.6% of the SRAM, with an inference latency of 23 ms, a total on-device cycle of 254 ms and a 12.9% processing duty cycle, and it reached 95.8–99.6% accuracy per gesture in real-time tests. These results indicate that real-time envelope-EMG gesture recognition is feasible on an entry-level microcontroller and is worth pursuing for human-machine interaction. Before use in prosthetics, rehabilitation or assistive devices, the system requires validation on larger, clinically representative populations, a PC-free implementation and direct power measurements (section 5.1).

Conflicts of interest

The authors declare that they have no conflict of interest.

Ethics approval

The study was approved by the Ethics Committee of the University of Engineering and Technology Peshawar, and all procedures followed the institutional ethical guidelines. All participants provided informed consent.

Funding

This work has not been supported financially.

Data availability statement

The data are available from the corresponding author on reasonable request.

Author contributions

Conceptualisation: Mohsin Ali Shah and Masood Ahmad Arbab; Data collection: Mohsin Ali Shah and Shahkaar Ahmad Khan; Methodology: Mohsin Ali Shah and Shahkaar Ahmad Khan; Writing – original draft: Mohsin Ali Shah and Ansar Ullah; Writing – review & editing: Mohsin Ali Shah and Masood Ahmad Arbab.

Acknowledgments

The authors thank all those who supported and guided this research.

Declaration on Generative AI

Generative AI tools were used in preparing the text of this manuscript; the authors reviewed the output and take full responsibility for the content.

References

[1] Afrin, S., Mahmud, H. and Hasan, M.K., 2022. EMG-Based Hand Gesture Dataset to Control Electronic Wheelchair for SCI Patients. *Proceedings of the 2nd International Conference on Computing Advancements*, ICCA '22. New York, NY, USA: Association for Computing Machinery, pp.165–170. Available from: https://doi.org/10.1145/3542954.3542979.

[2] Akhter, J., Naseer, N., Rashid, A., Khan, M.J., Hong, K.S. and Ullah, G., 2022. EMG Signals Based Gesture Recognition Accuracy Improvement by Machine Learning Classifiers/Algorithms. *2022 13th Asian Control Conference (ASCC)*. pp.1–6. Available from: https://doi.org/10.23919/ASCC56756.2022.9828126.

[3] Ali Shah, M. and Uddin, N., 2025. Surface electromyography-based gesture recognition via optimizer selection and multi-criteria performance analysis. *Cureus Journal of Computer Science*, 2, es44389-025-09497-9. Available from: https://doi.org/10.7759/s44389-025-09497-9.

[4] Anil, N. and Sreeletha, S., 2018. EMG Based Gesture Recognition Using Machine Learning. *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. pp.1560–1564. Available from: https://doi.org/10.1109/ICCONS.2018.8662987.

[5] Arsamy, R. and Balasubramanian, G., 2026. Surface Electromyography-Based Hand Gesture Recognition Using Optimized Attention Network for Human-Robot Interaction. *Journal of Field Robotics*, 43(3), pp.1661–1678. Available from: https://doi.org/10.1002/rob.70109.

[6] Bagherian Azhiri, R., Esmaeili, M. and Nourani, M., 2022. EMG-Based Feature Extraction and Classification for Prosthetic Hand Control. In: H. Al-Mubaid, T. Aldwairi and O. Eulenstein, eds. *Proceedings of 14th International Conference on Bioinformatics and Computational Biology*, *EPiC Series in Computing*, vol. 83. EasyChair, pp.136–145. Available from: https://doi.org/10.29007/zflb.

[7] Banbury, C., Janapa Reddi, V., Elium, A., Hymel, S., Tischler, D., Situnayake, D., Ward, C., Moreau, L., Plunkett, J., Kelcey, M., Baaijens, M., Grande, A., Maslov, D., Beavis, A., Jongboom, J. and Quaye, J., 2023. Edge Impulse: An MLOps Platform for Tiny Machine Learning. In: D. Song, M. Carbin and T. Chen, eds. *Proceedings of Machine Learning and Systems*, vol. 5. pp.254–268. Available from: https://proceedings.mlsys.org/paper_files/paper/2023/file/49fe55f5e9574714dda575bfb2177662-Paper-mlsys2023.pdf.

[8] Benalcázar, M.E., Jaramillo, A.G., Zea, J.A., Páez, A. and Andaluz, V.H., 2017. Hand gesture recognition using machine learning and the Myo armband. *2017 25th European Signal Processing Conference (EUSIPCO)*. pp.1040–1044. Available from: https://doi.org/10.23919/EUSIPCO.2017.8081366.

[9] Benalcázar, M.E., Motoche, C., Zea, J.A., Jaramillo, A.G., Anchundia, C.E., Zambrano, P., Segura, M., Benalcázar Palacios, F. and Pérez, M., 2017. Real-time hand gesture recognition using the Myo armband and muscle activity detection. *2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM)*. pp.1–6. Available from: https://doi.org/10.1109/ETCM.2017.8247458.

[10] Bhamidipati, R.G., Jatoth, R.K., Naresh, M. and Surepally, S.P., 2023. Real-time classification of haze and non-haze images on Arduino Nano BLE using Edge Impulse. *2023 4th International Conference on Computing and Communication Systems (I3CS)*. pp.1–7. Available from: https://doi.org/10.1109/I3CS58314.2023.10127312.

[11] Bhattacharjee, D. and Singh, M., 2021. Time-domain feature and ensemble model based classification of EMG signals for hand gesture recognition. Preprint (Version 1). Available from: https://doi.org/10.21203/rs.3.rs-605286/v1.

[12] Cedeño Z., C., Cordova-Garcia, J., Asanza A., V., Ponguillo, R. and Muñoz M., L., 2019. k-NN-Based EMG Recognition for Gestures Communication with Limited Hardware Resources. *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. pp.812–817. Available from: https://doi.org/10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00170.

[13] Celik, S. and Bayazit, U., 2025. Gesture detection from sEMG signals based on similarity learning. *Neural Computing and Applications*, 37(22), pp.18663–18677. Available from: https://doi.org/10.1007/s00521-025-11393-1.

[14] Chen, W. and Zhang, Z., 2019. Hand Gesture Recognition using sEMG Signals Based on Support Vector Machine. *2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*. pp.230–234. Available from: https://doi.org/10.1109/ITAIC.2019.8785542.

[15] Chung, E.A. and Benalcázar, M.E., 2019. Real-Time Hand Gesture Recognition Model Using Deep Learning Techniques and EMG Signals. *2019 27th European Signal Processing Conference (EUSIPCO)*. pp.1–5. Available from: https://doi.org/10.23919/EUSIPCO.2019.8903136.

[16] Cicirelli, G., Attolico, C., Guaragnella, C. and D'Orazio, T., 2015. A Kinect-Based Gesture Recognition Approach for a Natural Human Robot Interface. *International Journal of Advanced Robotic Systems*, 12(3), 22. Available from: https://doi.org/10.5772/59974.

[17] Dalal, D. and Keshwala, U., 2021. Design and analysis of EMG controlled anthropomorphic Prosthetic hand. *2021 8th International Conference on Signal Processing and Integrated Networks (SPIN)*. pp.345–349. Available from: https://doi.org/10.1109/SPIN52536.2021.9565947.

[18] Dehghani, A., Sarbishei, O., Glatard, T. and Shihab, E., 2019. A Quantitative Comparison of Overlapping and Non-Overlapping Sliding Windows for Human Activity Recognition Using Inertial Sensors. *Sensors*, 19(22), 5026. Available from: https://doi.org/10.3390/s19225026.

[19] Dweiri, Y.M., Hajjar, Y. and Hatahet, O., 2023. A novel neuroevolution model for EMG-based hand gesture classification. *Neural Computing and Applications*, 35(14), pp.10621–10635. Available from: https://doi.org/10.1007/s00521-023-08253-1.

[20] Englehart, K. and Hudgins, B., 2003. A robust, real-time control scheme for multifunction myoelectric control. *IEEE Transactions on Biomedical Engineering*, 50(7), pp.848–854. Available from: https://doi.org/10.1109/TBME.2003.813539.

[21] Fan, C., Huang, F., Yan, Y., Liu, Y., Zhang, J. and Zhang, X., 2023. Hand gesture recognition model based on surface EMG. *International Conference on Computer Graphics, Artificial Intelligence, and Data Processing (ICCAID 2022)*, *Proceedings of SPIE*, vol. 12604. SPIE, p.126043R. Available from: https://doi.org/10.1117/12.2674601.

[22] Fasulo, M., Spacone, G., Ingolfsson, T.M., Li, Y., Benini, L. and Cossettini, A., 2026. TinyMyo: a Tiny Foundation Model for Flexible EMG Signal Processing at the Edge. *IEEE Transactions on Biomedical Engineering*, pp.1–12. Early access. Available from: https://doi.org/10.1109/TBME.2026.3736157.

[23] Fukano, K., Iiazawa, K., Soeda, T., Shirai, A. and Capi, G., 2021. Deep Learning for Gesture Recognition based on Surface EMG Data. *2021 International Conference on Advanced Mechatronic Systems (ICAMechS)*. pp.41–45. Available from: https://doi.org/10.1109/ICAMechS54019.2021.9661533.

[24] Ghaffar Nia, N., Kaplanoglu, E. and Nasab, A., 2023. EMG-Based Hand Gestures Classification Using Machine Learning Algorithms. *SoutheastCon 2023*. pp.787–792. Available from: https://doi.org/10.1109/SoutheastCon51012.2023.10115158.

[25] Gkogkidis, A., Tsoukas, V., Papafotikas, S., Boumpa, E. and Kakarountas, A., 2022. A TinyML-based system for gas leakage detection. *2022 11th International Conference on Modern Circuits and Systems Technologies (MOCAST)*. pp.1–5. Available from: https://doi.org/10.1109/MOCAST54814.2022.9837510.

[26] Gomez-Bautista, A., Mendez, D., Alvarado-Rojas, C., Mondragon, I.F. and Colorado, J.D., 2025. On the Deployment of Edge AI Models for Surface Electromyography-Based Hand Gesture Recognition. *AI*, 6(6), p.107. Available from: https://doi.org/10.3390/ai6060107.

[27] Guzman, D.A.F., Sapienza, S., Sereni, B. and Ros, P.M., 2017. Very low power event-based surface EMG acquisition system with off-the-shelf components. *2017 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. pp.1–4. Available from: https://doi.org/10.1109/BIOCAS.2017.8325152.

[28] Haris, M., Chakraborty, P. and Rao, B.V., 2015. EMG signal based finger movement recognition for prosthetic hand control. *2015 Communication, Control and Intelligent Systems (CCIS)*. pp.194–198. Available from: https://doi.org/10.1109/CCIntelS.2015.7437907.

[29] Hudgins, B., Parker, P. and Scott, R., 1993. A new strategy for multifunction myoelectric control. *IEEE Transactions on Biomedical Engineering*, 40(1), pp.82–94. Available from: https://doi.org/10.1109/10.204774.

[30] Jaramillo-Yánez, A., Benalcázar, M.E. and Mena-Maldonado, E., 2020. Real-Time Hand Gesture Recognition Using Surface Electromyography and Machine Learning: A Systematic Literature

Review. *Sensors*, 20(9), 2467. Available from: https://doi.org/10.3390/s20092467.

[31] Kallimani, R., Pai, K., Raghuwanshi, P., Iyer, S. and López, O.L.A., 2024. TinyML: Tools, applications, challenges, and future research directions. *Multimedia tools and applications*, 83(10), pp.29015–29045. Available from: https://doi.org/10.1007/s11042-023-16740-9.

[32] Karapinar Senturk, Z. and Bakay, M., 2021. Machine Learning Based Hand Gesture Recognition via EMG Data. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 10(2), pp.123–136. Available from: https://doi.org/10.14201/ADCAIJ2021102123136.

[33] Kaya, E. and Kumbasar, T., 2018. Hand Gesture Recognition Systems with the Wearable Myo Armband. *2018 6th International Conference on Control Engineering & Information Technology (CEIT)*. pp.1–6. Available from: https://doi.org/10.1109/CEIT.2018.8751927.

[34] Kerdjidj, O., Amara, K., Harizi, F. and Boumridja, H., 2023. Implementing Hand Gesture Recognition Using EMG on the Zynq Circuit. *IEEE Sensors Journal*, 23(9), pp.10054–10061. Available from: https://doi.org/10.1109/JSEN.2023.3259150.

[35] Li, Y., Huang, J., Tian, F., Wang, H.A. and Dai, G.Z., 2019. Gesture interaction in virtual reality. *Virtual Reality & Intelligent Hardware*, 1(1), pp.84–112. Available from: https://doi.org/10.3724/SP.J.2096-5796.2018.0006.

[36] Majolo, M. and Balbinot, A., 2019. Proposal of a Hardware SVM Implementation for Fast sEMG Classification. In: R. Costa-Felix, J.C. Machado and A.V. Alvarenga, eds. *XXVI Brazilian Congress on Biomedical Engineering*, *IFMBE Proceedings*, vol. 70/2. Singapore: Springer Singapore, pp.381–386. Available from: https://doi.org/10.1007/978-981-13-2517-5_58.

[37] Manshouri, N. and Kayıkçıoğlu, T., 2016. Classification of 2D and 3D videos based on EEG waves. *2016 24th Signal Processing and Communication Application Conference (SIU)*. pp.949–952. Available from: https://doi.org/10.1109/SIU.2016.7495898.

[38] Manshouri, N., Melek, M. and Kayıkçıoğlu, T., 2022. Detection of 2D and 3D Video Transitions Based on EEG Power. *The Computer Journal*, 65(2), pp.396–409. Available from: https://doi.org/10.1093/comjnl/bxaa116.

[39] Melek, M. and Melek, N., 2025. Golden Distance: A New and Comprehensive Metric Definition Study Facilitating Classification Performance Evaluations. *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, 49(3), pp.1111–1123. Available from: https://doi.org/10.1007/s40998-025-00870-x.

[40] Melek, N., 2023. Comparison of EEG and EOG signals in classification of sleep stages. *Pamukkale University Journal of Engineering Sciences*, 29(6), pp.607–616. Available from: https://doi.org/10.5505/pajes.2022.88122.

[41] Melek, N., 2024. Responding to challenge call for machine learning model development in diagnosing respiratory disease sounds. *Journal of Edge Computing*, 3(1), pp.43–64. Available from: https://doi.org/10.55056/jec.679.

[42] Melo, M. Santos Pessoa de, Silva Neto, J. Gomes da, Natario Teixeira, J.M.X., Elza Fontes Da Gama, A. and Teichrieb, V., 2019. An EMG-Based Virtual Reality Application for Motor Rehabilitation. *2019 21st Symposium on Virtual and Augmented Reality (SVR)*. pp.170–177. Available from: https://doi.org/10.1109/SVR.2019.00041.

[43] Mongardi, A., Ros, P.M., Rossi, F., Roch, M.R., Martina, M. and Demarchi, D., 2019. A Low-Power Embedded System for Real-Time sEMG based Event-Driven Gesture Recognition. *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. pp.65–68. Available from: https://doi.org/10.1109/ICECS46596.2019.8964944.

[44] Müller, P.N., Achenbach, P., Kleebe, A.M., Schmitt, J.U., Lehmann, U., Tregel, T. and Göbel, S., 2020. Flex Your Muscles: EMG-Based Serious Game Controls. In: M. Ma, B. Fletcher, S. Göbel, J. Baalsrud Hauge and T. Marsh, eds. *Serious Games*. Cham: Springer International Publishing, pp.230–242. Available from: https://doi.org/10.1007/978-3-030-61814-8_18.

[45] Nadjib, B.L., Bilal, C. and Karima, R., 2021. EMG-Based Hand gesture recognition for myoelectric prosthetic hand control. *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)*. pp.1–6. Available from: https://doi.org/10.1109/AI-CSP52968.2021.9671095.

[46] Núñez-Montoya, B., Valarezo Añazco, M., Saravia-Avila, A., Loayza, F.R., Valarezo Añazco, E. and Teran, E., 2022. Supervised Machine Learning Applied to Non-Invasive EMG Signal Classification for an Anthropomorphic Robotic Hand. *2022 IEEE ANDESCON*. pp.1–6. Available from: https://doi.org/10.1109/ANDESCON56260.2022.9989874.

[47] Panda, A.K., Chakravarty, R. and Moulik, S., 2021. Hand Gesture Recognition using Flex Sensor and Machine Learning Algorithms. *2020 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES)*. pp.449–453. Available from: https://doi.org/10.1109/IECBES48179.2021.9398789.

[48] Pedrosa, S. and Costa, A., 2023. EMG Based MIDI Controller. *2023 7th International Young Engineers Forum (YEF-ECE)*. pp.106–111. Available from: https://doi.org/10.1109/YEF-ECE58420.2023.10209314.

[49] Phinyomark, A., Phukpattaranont, P. and Limsakul, C., 2012. Feature reduction and selection for EMG signal classification. *Expert Systems with Applications*, 39(8), pp.7420–7431. Available from: https://doi.org/10.1016/j.eswa.2012.01.102.

[50] Qi, J., Jiang, G., Li, G., Sun, Y. and Tao, B., 2019. Intelligent Human-Computer Interaction Based on Surface EMG Gesture Recognition. *IEEE Access*, 7, pp.61378–61387. Available from: https://doi.org/10.1109/ACCESS.2019.2914728.

[51] Raurale, S., McAllister, J. and Rincon, J.M. del, 2019. EMG Wrist-hand Motion Recognition System for Real-time Embedded Platform. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. pp.1523–1527. Available from: https://doi.org/10.1109/ICASSP.2019.8683104.

[52] Ray, P.P., 2022. A review on TinyML: State-of-the-art and prospects. *Journal of King Saud University - Computer and Information Sciences*, 34(4), pp.1595–1623. Available from: https://doi.org/10.1016/j.jksuci.2021.11.019.

[53] Scheme, E. and Englehart, K., 2011. Electromyogram pattern recognition for control of powered upper-limb prostheses: State of the art and challenges for clinical use. *Journal of Rehabilitation Research and Development*, 48(6), pp.643–659. Available from: https://doi.org/10.1682/JRRD.2010.09.0177.

[54] TensorFlow, 2023. TensorFlow Lite Micro library for Arduino. Available from: https://github.com/tensorflow/tflite-micro-arduino-examples.

[55] T'Jonck, K., Kancharla, C.R., Vankeirsbilck, J., Hallez, H., Boydens, J. and Pang, B., 2021. Real-Time Activity Tracking using TinyML to Support Elderly Care. *2021 XXX International Scientific Conference Electronics (ET)*. pp.1–6. Available from: https://doi.org/10.1109/ET52713.2021.9579991.

[56] Triwiyanto, T., Caesarendra, W., Abdullayev, V., Ahmed, A.A. and Herianto, H., 2023. Single Lead EMG signal to Control an Upper Limb Exoskeleton Using Embedded Machine Learning on Raspberry Pi. *Journal of Robotics and Control (JRC)*, 4(1), pp.35–45. Available from: https://doi.org/10.18196/jrc.v4i1.17364.

[57] Wahid, M.F., Tafreshi, R. and Langari, R., 2020. A Multi-Window Majority Voting Strategy to Improve Hand Gesture Recognition Accuracies Using Electromyography Signal. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28(2), pp.427–436. Available from: https://doi.org/10.1109/TNSRE.2019.2961706.

[58] Wan, B., Wu, R., Zhang, K. and Liu, L., 2017. A new subtle hand gestures recognition algorithm based on EMG and FSR. *2017 IEEE 21st International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. pp.127–132. Available from: https://doi.org/10.1109/CSCWD.2017.8066682.

[59] Xiong, D., Zhang, D., Zhao, X. and Zhao, Y., 2020. Hand Gesture Recognition Using Instant High-density EMG Graph via Deep Learning Method. *2020 Chinese Automation Congress (CAC)*. pp.5143–5148. Available from: https://doi.org/10.1109/CAC51589.2020.9326536.

[60] Young, S., Stephens-Fripp, B., Gillett, A., Zhou, H. and Alici, G., 2019. Pattern Recognition for Prosthetic Hand User's Intentions using EMG Data and Machine Learning Techniques. *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. pp.544–550.

Available from: https://doi.org/10.1109/AIM.2019.8868766.

[61] Zanghieri, M., Benatti, S., Burrello, A., Kartsch, V., Conti, F. and Benini, L., 2020. Robust Real-Time Embedded EMG Recognition Framework Using Temporal Convolutional Networks on a Multicore IoT Processor. *IEEE Transactions on Biomedical Circuits and Systems*, 14(2), pp.244–256. Available from: https://doi.org/10.1109/TBCAS.2019.2959160.

[62] Zardoshti-Kermani, M., Wheeler, B., Badie, K. and Hashemi, R., 1995. EMG feature evaluation for movement control of upper extremity prostheses. *IEEE Transactions on Rehabilitation Engineering*, 3(4), pp.324–333. Available from: https://doi.org/10.1109/86.481972.

[63] Zhang, Z., Yang, K., Qian, J. and Zhang, L., 2019. Real-Time Surface EMG Pattern Recognition for Hand Gestures Based on an Artificial Neural Network. *Sensors*, 19(14), p.3170. Available from: https://doi.org/10.3390/s19143170.