Master Thesis Research Paper

Spatio-Temporal Transformer for Load Estimation using EMG and IMU in Assistive Robotics

Bas Wingen[a], Arno Stienen[a] and Xucong Zhang[b]

[a] *Biomechanical Engineering, Delft University of Technology*
[b] *Intelligent Systems, Delft University of Technology*

Abstract

Intuitive control of assistive robotic devices, such as exoskeletons and arm supports, requires inferring the user's interaction with objects in the environment. Surface electromyography (EMG) and inertial measurement units (IMU) provide complementary information about muscle activation and limb kinematics, but interpreting these sensory modalities for real-time control remains challenging. Deep learning is effective for modeling human motion intention, but has seen limited use in estimating the handheld load during object manipulation. This paper proposes a sensor-fused spatio-temporal transformer (ST-Transformer) that regresses the handheld load from synchronized EMG and IMU signals, together with a real-time acquisition and processing pipeline for an arm support device. Data were used from 17 participants performing a weight-movement task spanning six weight classes ($0 - 6$ kg). EMG and IMU normalization, dataset-balancing augmentation, dropout, and weight decay were applied to improve cross-participant generalization. Trained and tested on the same participants, the sensor-fused model estimated load accurately (all metrics participant-class-balanced; $R^2 = 0.935$, MAE $= 0.316$ kg, RMSE $= 0.441$ kg) and significantly outperformed an EMG-only model ($R^2 = 0.913$, MAE $= 0.380$ kg, RMSE $= 0.520$ kg). Under Leave-One-Participant-Out (LOPO) cross-validation, however, the fused model ($R^2 = 0.853$, MAE $= 0.536$ kg, RMSE $= 0.680$ kg) retained only a slight, statistically non-significant edge over EMG alone ($R^2 = 0.839$, MAE $= 0.546$ kg, RMSE $= 0.703$ kg), while the IMU-only model degraded sharply. This indicates that the transferable load information is carried primarily by muscle activation, while the complementary IMU contribution is largely entangled with participant-specific characteristics. An attribution analysis localizes the load-relevant signal to the forearm muscles, indicating that a compact forearm-worn sensor set captures most of the usable signal, and the model (approximately $1.03 \cdot 10^6$ parameters) is feasible for real-time on-device inference on current microcontrollers.

Keywords: *Deep Learning, Transformer, EMG, IMU, Load Estimation, Regression, Sensor fusion, Assistive Robotics, Exoskeleton*

1. INTRODUCTION

Assistive robotics, such as exoskeletons and arm supports, has advanced rapidly in recent years, transforming physical rehabilitation and supporting human lifting in industrial settings [52]. The medical field has been particularly influenced by these developments, with robotic rehabilitation devices and mobility support systems becoming increasingly integrated into clinical practice [41, 47, 62, 63]. Devices such as the SaeboMAS [38] (Figure 1) assist in neurorehabilitation by supporting the arm under its own weight. In industry, actively and passively actuated exoskeletons (e.g., the EksoEVO [53]) reduce musculoskeletal loading during lifting and overhead work, with reported muscle-activity reductions of $10 - 40\%$ for passive devices and up to 80% for active ones [16], lowering injury risk and supporting workers' long-term employability.

Figure 1. Dynamic arm support system that counteracts gravity while preserving the range of motion of the arm. Manually tunable to the level of assistance, purposely built for stroke rehabilitation. [38]

Despite these benefits, designing assistive devices that operate in synergy with the user's movement remains a significant engineering challenge [49, 52]. Passive systems are simple, robust, and require little maintenance, but their lack of actuated control limits adaptability and often forces task-specific or individually customized designs [48]. Active systems restore adaptability at the cost of added complexity, power consumption, weight, and size [36, 39, 46, 49]. In both cases, assistance is only as effective as the device's ability to infer what the user is doing: seamless coupling of human and robotic motion depends on continuously estimating the user's intent and required assistance, and adapting the control accordingly [52]. This requirement has driven growing interest in sensory systems that enable the development of more intelligent and transparent control frameworks [32, 49, 57, 58].

Among available sensory modalities, surface electromyography (EMG) plays a central role because it provides direct access to neuromuscular activation [32, 33, 61]. Because muscle activation precedes movement by an electromechanical delay, EMG enables anticipatory control strategies that act before kinematic changes occur [4]. This is especially valuable for upper-extremity tasks requiring fine, rapid, and coordinated control. However, EMG is also characterized by substantial inter- and intra-subject variability, susceptibility to noise and electrode shifts, and dependence on user-specific physiology, which complicates generalization both across users and within users [47].

Machine learning (ML) and deep learning in particular have therefore become increasingly prominent in EMG interpretation [61] due to their ability to model nonlinear signal dynamics and automatically extract discriminative features from raw data [35, 49, 64]. Such approaches have demonstrated improvements in intent recognition and motion prediction, yet they also introduce significant computational demands, raising concerns regarding on-device feasibility for real-time control [54].

To reduce reliance on a single modality, sensor fusion combines EMG with other sensors, such as inertial measurement units (IMUs) [43, 46, 63]. EMG provides information about neuromuscular activation, whereas IMU provides measurements of limb kinematics [34]. Across a wide range of wearable robotics tasks, fusing such modalities improves accuracy and robustness compared with single-sensor approaches [52], and physics-informed approaches go further by embedding biomechanics directly into the model [58]. It remains unclear,

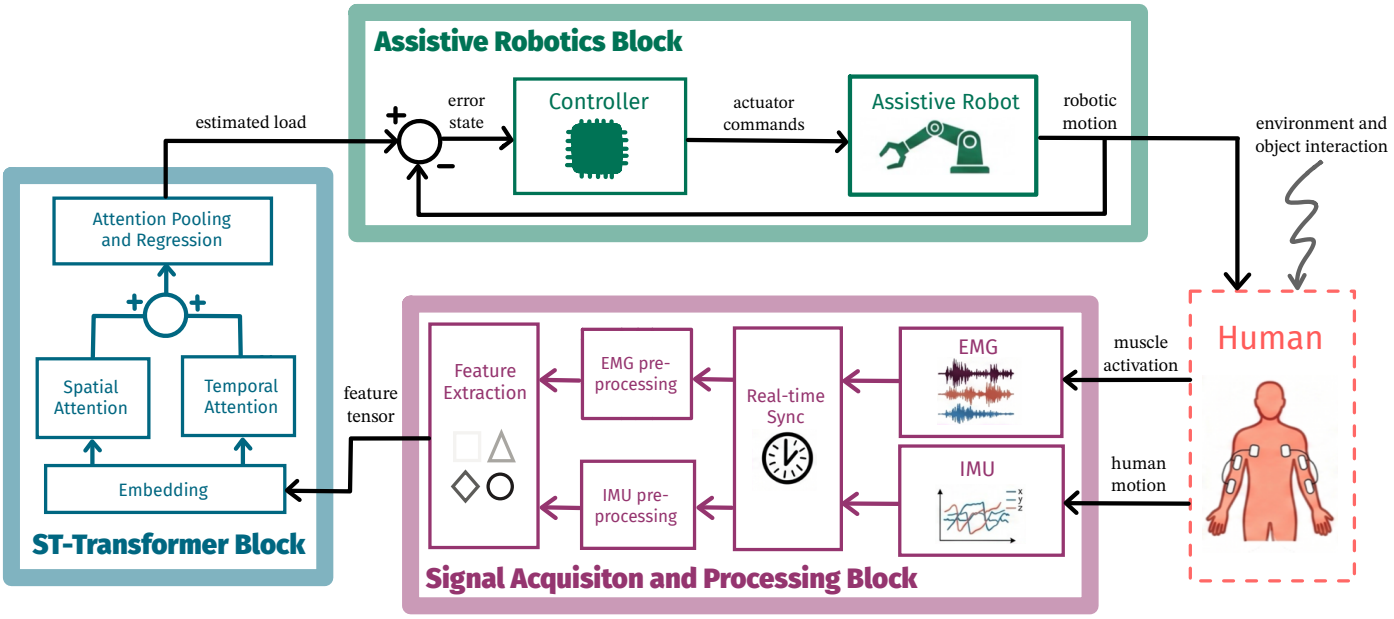


Figure 2. Proposed control framework utilizing ML-based load estimation as reference injection for human-in-the-loop control. The ST-Transformer and Signal Acquisition and Processing blocks are the contributions of this paper.

however, whether this benefit extends to continuous load estimation and, in particular, whether it transfers to users unseen during training.

Several studies integrate ML-based EMG interpretation into control frameworks of exoskeletons and assistive robotics, using EMG-only or sensor-fused approaches [36, 39, 42, 49, 50, 56]. Such estimators typically generate references for a low-level controller, introducing prediction accuracy, latency, on-device feasibility, transparency, and stability as critical design constraints. In this paper, the ML-based load estimation block (ST-Transformer Block in Figure 2) is introduced as a reference injector to the controller of an arbitrary assistive robot, as illustrated in Figure 2.

Among deep learning architectures, recurrent networks such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) have been widely applied to EMG and IMU interpretation [35, 44, 64], as their gated memory mechanisms capture temporal dependencies in sequential signals while mitigating the vanishing-gradient problem [5, 15]. However, recurrent models are less suited to the spatial structure of multi-channel sensor arrays and accumulate error in long-horizon autoregressive prediction [26].

Convolutional neural networks (CNN), originally developed for image processing [6], add spatial feature extraction and are often combined with recurrent layers in CNN-LSTM or CNN-GRU architectures [35, 44, 49, 64]. Their learned feature extraction enhances their ability to distinguish complex spatial patterns, thereby oftentimes outperforming standalone recurrent architectures. For convolutional models, subject transferability remains difficult too, and the added capacity demands larger, more diverse datasets and more parameters, increasing computational cost.

Attention mechanisms address some of these limitations and stand at the core of the Transformer architecture, which transformed natural language processing [20]. Attention has since been incorporated into EMG interpretation, often alongside recurrent architectures [64]. In the paper by Aksan et al. [26], a spatio-temporal transformer (ST-Transformer) architecture is proposed that inherits the spatial and temporal qualities of convolutional-recurrent architectures by leveraging the attention mechanism.

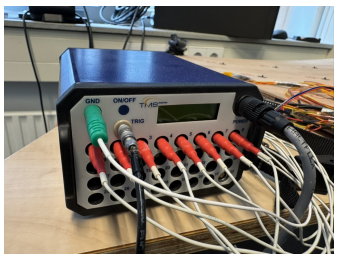
Despite strong performance in motion and intent recognition, ML-based EMG interpretation has largely focused on movement prediction rather than the external load a user handles. Where load has been addressed, it has been framed as discrete payload classification from IMU signals [37] or regressed from EMG alone using physics-informed priors [58]. Continuous load regression from fused EMG and IMU signals, and whether such a model transfers to users unseen during training, remains largely unexplored. Object manipulation introduces interaction dynamics among the user, the object, and the robot, requiring the controller to continuously estimate the external load rather than predict motion. The SaeboMAS [38] (Figure 1) provides a well-defined case study: by reducing actuation to a single upward-assistance degree of freedom, it isolates the load-estimation problem while remaining representative of real-world assistive use. This work therefore develops and evaluates a sensor-fused spatio-temporal transformer for continuous load estimation from EMG and IMU, analyzes how its accuracy and the fusion influence transfer across participants, and assesses its feasibility for real-time on-device deployment.

2. METHODOLOGY

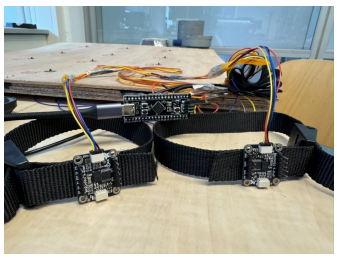
The objective of the deep learning model is to estimate the weight of a held object from synchronized electromyography (EMG) and inertial measurement unit (IMU) signals. The model was trained and evaluated on data collected during a custom experimental trial. For a prediction $\hat{y}_k$ at prediction step k, L denotes the input window length. At each prediction step k, the EMG and IMU inputs are the signal windows $\mathbf{X}^{(L)}_{\text{EMG}}(k)$ and $\mathbf{X}^{(L)}_{\text{IMU}}(k)$. C_{EMG} and C_{IMU} denote the number of signal channels for each sensory modality. The model is a nonlinear function $f_\theta(\cdot)$ with learned parameters θ:

$$\begin{aligned} \hat{y}(t) &= f_\theta\left(\mathbf{X}^{(L)}_{\text{EMG}}(k), \mathbf{X}^{(L)}_{\text{IMU}}(k)\right), \\ \text{with } \mathbf{X}^{(L)}_{\text{EMG}}(k) &\in \mathbb{R}^{L\times C_{\text{EMG}}}, \quad \mathbf{X}^{(L)}_{\text{IMU}}(k) \in \mathbb{R}^{L\times C_{\text{IMU}}} \end{aligned} \tag{1}$$

The signals are sampled at frequencies f_{EMG} and f_{IMU}, synchronized, and segmented into windows, after which weight estimates are produced at a prediction interval Δt. The input window length L introduces a trade-off between estimation accuracy and responsiveness. The data pipeline and model architecture were designed for real-time, on-device inference. The acquisition firmware, preprocessing pipeline, and model implementation are publicly available [65].



(a) TMSi Porti7 EMG system.



(b) Dual Adafruit BNO085 IMUs, acquired using a STM32F401 microcontroller.

Figure 3. Sensory acquisition systems synchronized using a PRBS combined with Kalman filtering.

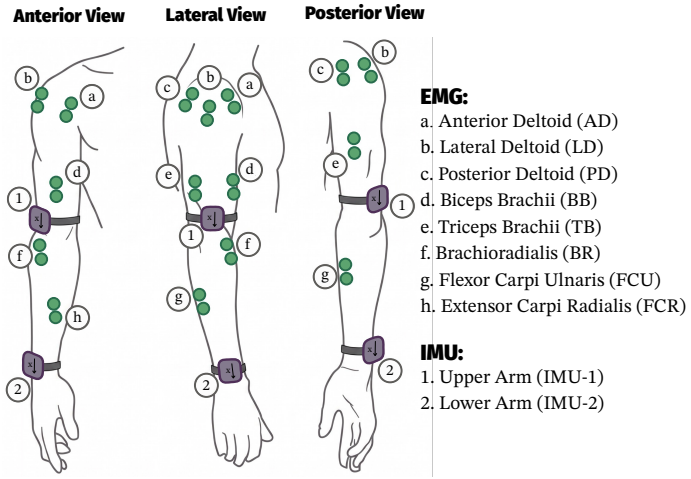


Figure 4. Schematic of EMG electrode and IMU sensor placement on the participants' arm following the SENIAM protocol for EMG electrode placement [7]. IMU sensors are aligned with the x-axis pointing towards the hand.

2.1. Sensory acquisition system

No single device capable of acquiring both EMG and IMU signals was available, so separate acquisition systems were used for EMG and IMU, as depicted in Figure 3. The two systems were synchronized using a pseudo-random binary sequence (PRBS) and real-time Kalman filtering, ensuring that muscle activation and limb motion were registered at the same instant. EMG signals provide a measure of arm muscle activation, directly related to load-bearing. The IMU signals provide kinematic information complementary to EMG, and the dynamics of the arm–object system change with object weight. Orientation- and acceleration-related features may help the model distinguish movement-associated activation from load-bearing activation. Appendix A provides extra details on the sensory acquisition system.

2.1.1. EMG system

EMG signals were acquired with a TMSi Porti7 [1] system at a sampling frequency of $f_{\text{EMG}} = 2000$ Hz and streamed to a PC over USB-A using the Python TMSi SDK. Eight bipolar channels were recorded ($C_{\text{EMG}} = 8$), with electrodes placed according to the SENIAM protocol [7]. Three electrode pairs were placed on forearm muscles associated with gripping (Extensor Capri Radialis, Flexor Carpi Ulnaris and Brachioradialis), since grip force is expected to increase with object weight; two pairs on the upper arm to capture elbow flexor–extensor co-contraction (Biceps, Triceps Brachii), which reflects increased arm stiffness during heavier lifts; and three pairs on the shoulder (Anterior, Lateral, Posterior Deltoid) to capture activation related to raising the arm and supporting the object against gravity. The selected muscles and their placement are shown in Figure 4. The resulting discrete-time EMG input over a window of length L is

$$\mathbf{X}_{\text{EMG}} = \begin{bmatrix} x_1 & x_2 & \cdots & x_{C_{\text{EMG}}} \end{bmatrix} \in \mathbb{R}^{L \times C_{\text{EMG}}}, \quad C_{\text{EMG}} = 8 \tag{2}$$

where x_c denotes the surface EMG signal of the c-th recorded muscle (Figure 4).

2.1.2. IMU system

Two Adafruit BNO085 [2] breakout boards were used, each combining a three-axis accelerometer, gyroscope, and magnetometer with an on-board Arm Cortex-M0 processor that fuses these measurements. The BNO085 was operated in UART-RVC mode, which outputs a 6-DOF fused output: three-axis orientation and three-axis linear acceleration. UART-RVC was the only mode supporting simultaneous readout of a dual IMU setup. An STM32F401[3] microcontroller received both IMU streams and forwarded the data to the PC over USB.

[1] TMSi Porti7 (discontinued) - URL: https://www.tmsi.artinis.com/product-overview
[2] Adafruit BNO085 - URL: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-9-dof-orientation-imu-fusion-breakout-bno085.pdf
[3] STM32F401 - Arm Cortex-M4, URL: https://www.st.com/en/microcontrollers-microprocessors/stm32f401.html

UART-RVC streams at a fixed rate of approximately 100 Hz. The STM32F401 sampled and timestamped both streams at $f_{\text{IMU}} = 500$ Hz, oversampling the 100 Hz sensor output so that each update was captured accurately for synchronization (Section 2.1.3). For IMU $i \in \{1, 2\}$, the resulting discrete-time signal is:

$$\mathbf{X}_{\text{IMU},i} = \begin{bmatrix} \phi_i & \theta_i & \psi_i & a_{x,i} & a_{y,i} & a_{z,i} \end{bmatrix} \in \mathbb{R}^{L \times 6} \tag{3}$$

where ϕ_i, θ_i, ψ_i are the Euler angles (yaw, pitch, roll) and $a_{x,i}, a_{y,i}, a_{z,i}$ the linear-acceleration components of IMU i. With two IMUs, this yields $C_{\text{IMU}} = 12$ channels in total.

2.1.3. Real-time signal synchronization

Because the EMG and IMU systems ran on independent clocks and connected to the PC via different ports, their streams exhibited time-varying offsets and clock drift. To align them, the STM32F401 generated a pseudo-random binary sequence (PRBS) that was transmitted directly to the PC and through the EMG device. Cross-correlating the two transmissions over a sliding window yields an estimate of the delay between both systems, together with a confidence measure [3]. A two-state Kalman filter then tracks the delay offset and clock drift, using the correlation confidence as the measurement uncertainty to smooth noisy estimates [2, 11]. The smoothed parameters are used to resample the 500 Hz STM32 stream onto the 2000 Hz EMG clock by linear interpolation, without truncating or distorting either signal. The full algorithm is detailed in Appendix B.

2.2. Participant Trial

For this study, the EMG and IMU sensor data are gathered from 18 participants. Their participant ID, gender, age, and anthropometric properties are listed in Table 1.

All participants participated in a weight-movement trial. The participants were instructed to move weights ranging from 1 to 6 kilograms across a 3-by-4 grid (with $d_{\text{button}} \approx 15$ cm between each button) mounted horizontally on a table. Instructions for the movement were displayed on a screen, while buttons integrated into the panel registered the real-world movements of the weights. Rubber bands under the panel ensured the buttons were pushed through the holes without breaking when a heavy load was applied. Movements were randomly generated and varied in direction and distance, with equal distribution across the different weights. A large number of possible start-end point combinations (12 grid positions for 11 possible destinations and 5 weights $12 \cdot 11 \cdot 5 = 660$, excluding free movements) makes it im-

Table 1. Participant characteristics and anthropometrics in centimeters [cm]. P13 was excluded due to age out of distribution. (circ. = circumference

ID	Gender	Age	Handedness	Total arm	Upper arm	Forearm	Hand	Upper arm circ.	Forearm circ.
P01	Male	25	Right	82	38	33	20	32	28
P02	Male	23	Right	78	35	26	22	26	25
P03	Male	25	Left	78	36	27	19	31	28
P04	Male	22	Right	77	32	26	22	28	28
P05	Female	24	Right	77	34	27	18	25	23
P06	Male	23	Right	82	37	29	22	30	27
P07	Male	24	Right	78	36	29	18	33	28
P08	Male	22	Right	75	34	27	20	27	27
P09	Female	22	Right	71	32	23	18	25	23
P10	Male	22	Right	78	32	29	22	33	30
P11	Male	26	Right	84	27	29	24	32	32
P12	Male	24	Right	89	41	29	20	28	26
P13	Female	59	Right	69	33	26	20	22	22
P14	Male	27	Right	81	38	27	20	30	28
P15	Female	25	Right	61	22	23	16	26	24
P16	Female	25	Right	75	31	25	19	22	22
P17	Male	26	Right	82	33	29	21	30	27
P18	Female	24	Left	72	30	25	19	27	26

probable that a model will learn specific motion patterns. Between each weight movement is a free movement, which is registered as lifting 0 kilograms. The microcontroller registers the state of the button matrix and transmits it, along with the IMU signals, through the same signal-acquisition and synchronization pipeline. Fig. 5 shows the instructions for a movement on the button matrix.

Each time the state of the button matrix changes, the EMG and IMU signals are sliced and matched with the weight instruction to create a labeled segment. During weight pickup, the start of the segment is offset $t_{\text{offset}} = 0.2$ s back in time to capture the anticipatory muscle activation and electromechanical delay (EMD) [4]. No offset is applied at the end of a weight movement. This offset is determined empirically from the dataset and experimentally verified by training models across different configurations.

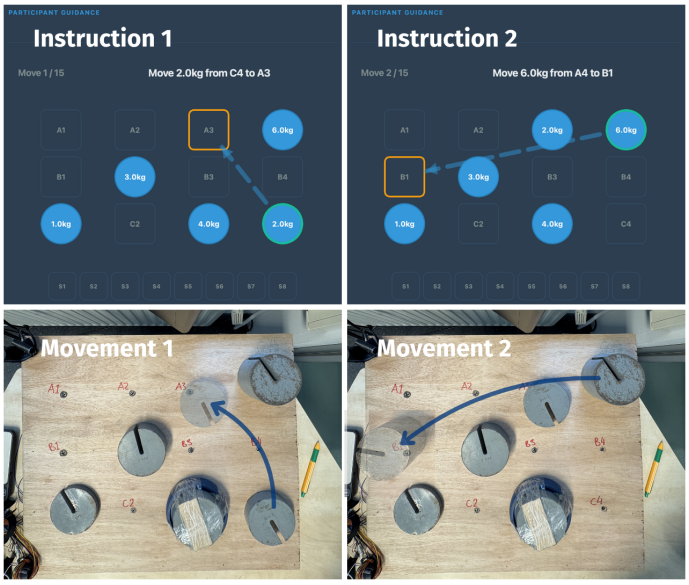


Figure 5. Trial instructions for two movements on a custom trial dashboard and button matrix, registering movements for signal segmentation.

2.3. Signal processing

Because deep learning can capture complex signal morphologies and discriminate informative features from noise through learned representations, the inputs were only lightly preprocessed. EMG, IMU orientation, and IMU linear acceleration signals are fundamentally different and were therefore processed with individually tailored pipelines.

2.3.1. EMG preprocessing

Traditional EMG preprocessing approaches increase the signal's interpretability at the expense of removing frequency content and subtle morphology [10]. Additionally, heavy filtering and smoothing introduce phase lag, thereby negatively affecting responsiveness in real-time implementation. To remove motion artifacts, high-frequency noise, and powerline interference, the following filtering was applied:

$$x_{\text{filt},c} = \mathcal{N}_{50}\left\{\mathcal{B}_{20-500}\left\{x_{\text{raw},c}\right\}\right\}, \quad c = 1, \ldots, C_{\text{EMG}} \tag{4}$$

where $x_{\text{raw},c}$ is the raw EMG signal of channel c, $x_{\text{filt},c}$ is the filtered EMG signal, $\mathcal{B}_{20-500}\{\cdot\}$ represents a fourth-order Butterworth band-pass filter with cut-off frequencies of 20 and 500 Hz, and $\mathcal{N}_{50}\{\cdot\}$ represents a 50 Hz notch filter.

Additionally, in EMG processing, it is common to normalize signals with respect to the maximum voluntary contraction (MVC) [8]. This mitigates inter-subject variability, yielding more generic signals for the model to handle. However, calibration by having a participant contract each muscle individually was deemed impractical, especially for forearm muscles. It is also an objective of the algorithm to work successfully on unseen users without requiring calibration.

However, during modeling and algorithm development, a similar approach to MVC was adopted, which does require calibration in a real-time application. First, after filtering, the EMG signal envelope for each participant was calculated by rectifying and smoothing the signal. Afterward, the 99% percentile of the envelope was calculated as a robust peak. Finally, the unrectified, unsmoothed signal is normalized by this peak. This way, the frequency content of the raw signal is preserved while scaling the signal down by a factor that helps overcome inter-subject variability. The normalization is a per-session EMG peak normalization, therefore requiring unlabeled signals from the unseen user as calibration (Section 4).

2.3.2. IMU preprocessing

The IMU signals are robustly upsampled using linear interpolation to match the EMG frequency of 2000 Hz. To smooth discontinuities as a result of linear interpolation and remove high-frequency motion jitter, a low-pass filter was applied [13]:

$$x_{\text{filt},c} = \mathcal{L}_{15}\left\{x_{\text{raw},c}\right\}, \qquad c = 1, \ldots, C_{\text{IMU}}. \tag{5}$$

where $x_{\text{raw},c}$ and $x_{\text{filt},c}$ denote the raw and filtered IMU signals of channel c, respectively, and $\mathcal{L}_{15}\{\cdot\}$ denotes a low-pass filter with a 15 Hz cut-off frequency. The orientation signals were unwrapped for filtering, rewrapped to ensure feature stability, and converted from degrees to radians. Finally, each IMU channel was normalized per recording session using robust statistics, centering on the session median

and scaling by the session interquartile range (IQR). Analogous to EMG peak normalization, this aims to reduce per-participant offsets and gains in the IMU signals arising from limb inertia and sensor placement.

2.4. Spatio-Temporal Transformer

After data collection, preprocessing, segmentation, and labeling, the EMG and IMU signal segments were used as training samples for the supervised regression model defined in Equation 1. Convolutional-recurrent architectures were evaluated first, following prior work that successfully estimated continuous elbow motion with a CNN-LSTM [35], particularly when combined with biomechanics in a physics-informed neural network [58]. A Spatio-Temporal Transformer (ST-Transformer), inspired by the architecture of Aksan et al. [26] for 3D human motion prediction, was compared against five established baselines (LSTM, GRU, a vanilla transformer, CNN-LSTM, and CNN-GRU) under both validation strategies. Under leave-one-participant-out (LOPO) cross-validation, the ST-Transformer is statistically indistinguishable from the recurrent models (LSTM, GRU) and the vanilla transformer, and clearly ahead of the convolutional models (CNN-LSTM, CNN-GRU). Within participants, the recurrent models achieve somewhat lower error, particularly on MAE. The full comparison is reported in Appendix H (Figure 12 and 13). The ST-Transformer is competitive with the established recurrent architectures. It was adopted as the primary architecture for this paper due to its explicit sensor-fusion mechanism, its intrinsic interpretability through spatial and temporal attention, and its natural extensibility toward a unified load–motion model (Section 4, Section 5).

The model was trained to estimate each segment's external load. The Mean Squared Error (MSE) was minimized as the training loss, penalizing the squared difference between the true and predicted load (Equation 6). The Root Mean Square Error (RMSE) is reported as a primary performance metric because it shares the same units as the load (kg) and is therefore more interpretable.

$$\mathcal{L}_{\text{MSE}}(\theta) = \frac{1}{N}\sum_{j=1}^{N}\left[y_j - f_\theta\left(\mathbf{X}^{(L)}_{\text{EMG},j}, \mathbf{X}^{(L)}_{\text{IMU},j}\right)\right]^2 \tag{6}$$

Here, N denotes the number of training samples in the batch or dataset, y_j is the true load label corresponding to segment j, and $f_\theta(\mathbf{X}^{(L)}_{\text{EMG},j}, \mathbf{X}^{(L)}_{\text{IMU},j})$ is the load predicted by the model f with parameters θ. The terms $\mathbf{X}^{(L)}_{\text{EMG},j}$ and $\mathbf{X}^{(L)}_{\text{IMU},j}$ represent the EMG and IMU input windows of length L for segment j, respectively. Because segments have variable lengths, each training batch was zero-padded to the length of its longest sample, and a padding mask prevented the attention mechanism from attending to the padded timesteps.

Additionally, an AdamW optimizer was used to stabilize training and provide regularization through weight decay [17]. Two learning-rate schedules were used, adapted to each validation strategy. For participant-specialized models, a fixed-epoch OneCycleLR schedule used super-convergence to fully fit each participant's data [24]. For the generalized (LOPO) models, training must be halted before the network begins to fit participant-specific patterns, requiring strong early stopping at an unpredictable epoch, which is incompatible with OneCycleLR's fixed epoch length. A ReduceLROnPlateau schedule with strong early stopping was therefore used instead, reducing the learning rate when the validation loss plateaued and eventually halting training.

2.4.1. Feature Extraction

After synchronization, both modalities are represented on a common grid ($f_{\text{grid}} = 2000$ Hz). Learning directly from these high-resolution samples is informative but computationally demanding. Instead, manually engineered features were extracted from the signals, introducing an inductive bias that proved beneficial for the ST-Transformer and improved generalization by suppressing noise and individual differences in EMG morphology. A CNN-based learned feature extractor was also evaluated, but it did not outperform the manually engineered features on this dataset, particularly in cross-participant generalization (Appendix H).

To preserve most of the signal's temporal information while effectively downsampling, a sliding-window feature-extraction algorithm was implemented. Because the two modalities have different dynamics, distinct window lengths were used: $w_{\text{EMG}} = 150$ ms and $w_{\text{IMU}} = 200$ ms, each advanced by a step $\Delta t_{\text{window}} = 100$ ms. The step sets the temporal resolution of the resulting feature sequence, while the longer, overlapping per-modality windows capture each signal's relevant dynamics. A distinct feature set was selected for each modality and validation strategy through a dedicated sweep. The selected features, together with their mathematical derivation and physical inductive bias, are listed in Appendix D (Table 4).

2.4.2. Architectural Design

The architectural design is based on the Spatio-Temporal Transformer of Aksan et al. [26], and the vanilla Transformer originally proposed by Vaswani et al. [20], but is adapted for sensor fusion and scalar regression. The spatio-temporal transformer architecture was found to be state-of-the-art for human motion prediction, a task related to this article's objective. The decoupling of spatial and temporal attention explicitly captures both structural and temporal dependencies. An overview of the model architecture is visualized in Figure 6.

After feature extraction, each input segment is represented as $\mathbf{X} \in \mathbb{R}^{T\times C\times F}$, where T is the number of timesteps, and F is the total number of extracted features. The feature vector at each timestep is divided into C channel-specific feature groups $\mathbf{x}_{t,c} \in \mathbb{R}^{F_c}$, with $F = \sum_{c=1}^{C} F_c$. Each channel-specific feature vector is projected into a shared D-dimensional embedding space:

$$\mathbf{e}_{t,c} = \mathbf{W}_c\mathbf{x}_{t,c} + \mathbf{b}_c, \qquad \mathbf{W}_c \in \mathbb{R}^{D\times F_c}, \quad \mathbf{e}_{t,c} \in \mathbb{R}^D \tag{7}$$

Each projected vector $\mathbf{e}_{t,c} \in \mathbb{R}^D$ is treated as a token: the elementary unit over which the transformer's attention operates, analogous to a word embedding in language models. A segment of T timesteps and C channels thus yields $T \times C$ tokens. A learned channel identity embedding $\mathbf{p}_c \in \mathbb{R}^D$ and a sinusoidal temporal positional encoding $\mathbf{p}_t \in \mathbb{R}^D$ are added to each token:

$$\tilde{\mathbf{e}}_{t,c} = \mathbf{e}_{t,c} + \mathbf{p}_c + \mathbf{p}_t \tag{8}$$

The embedded tokens are stacked into encoded embedding tensor $\tilde{\mathbf{E}} \in \mathbb{R}^{T\times C\times D}$.

The resulting tensor is processed by stacked spatio-temporal transformer blocks. Following Aksan et al. [26], spatial and temporal attention are decoupled. In this work, the spatial dimension corresponds to sensor channels rather than skeletal joints. This follows the same principle of separating within-timestep and across-timestep dependencies, but adapts it for sensor fusion.

Let the input to transformer layer $\ell \in \{1, \dots, N_L\}$ (where N_L is the total number of transformer layers) be denoted as $\mathbf{Z}^{(\ell)} \in \mathbb{R}^{T\times C\times D}$, with $\mathbf{Z}^{(0)} = \tilde{\mathbf{E}}$. For each timestep t, spatial attention is applied over the C channel tokens. The spatial sequence is defined as:

$$\mathbf{Z}^{(\ell)}_t = \left[\mathbf{z}^{(\ell)}_{t,1}, \dots, \mathbf{z}^{(\ell)}_{t,C}\right]^\top \in \mathbb{R}^{C\times D}. \tag{9}$$

For each attention head $h \in \{1, \dots, H_{\text{sp}}\}$, where H_{sp} is the number of spatial attention heads, learned projection matrices transform the spatial sequence into query, key, and value matrices:

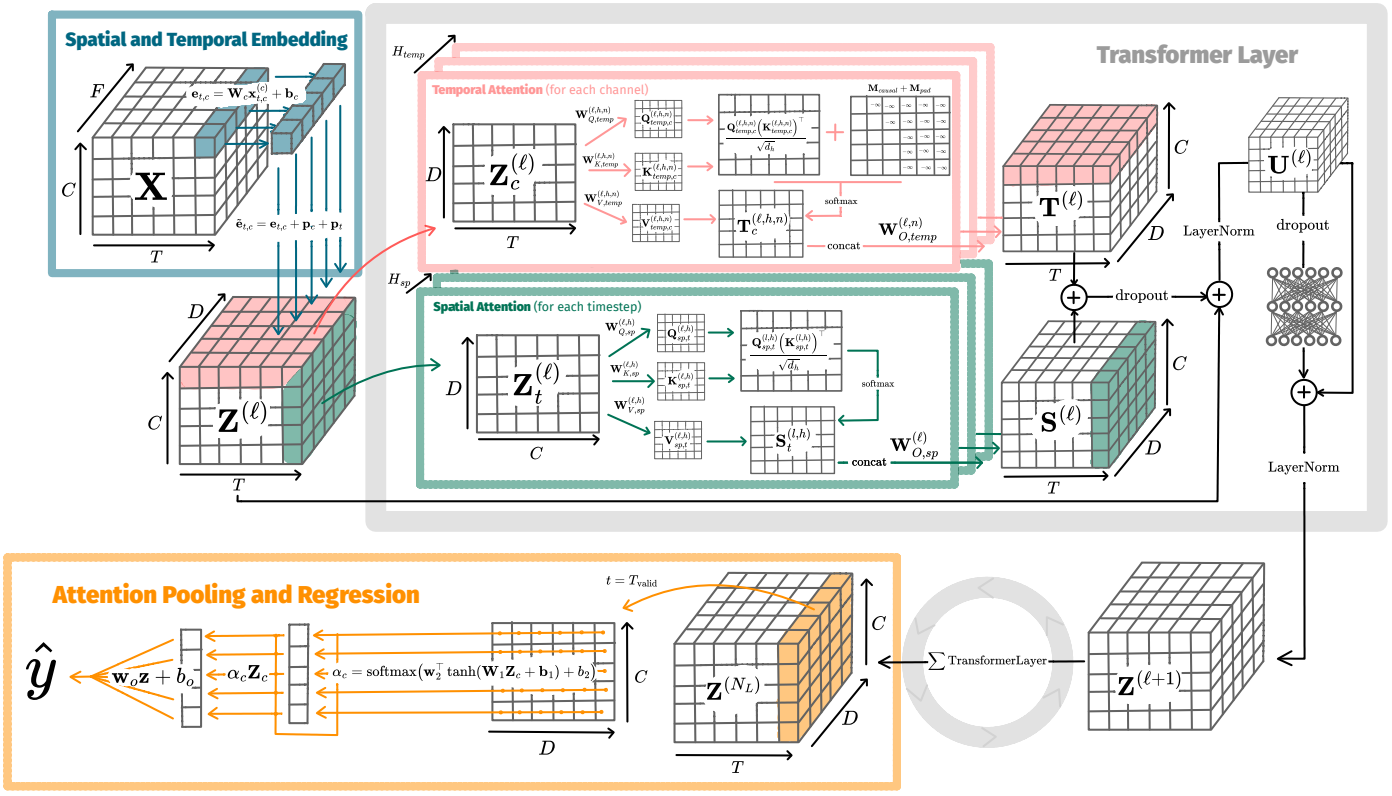


Figure 6. Overview of spatio-temporal transformer architecture, illustrating its structure and the flow of data through the network.
Disclaimer: tensor sizes, matrix sizes, vector sizes, and number of layers and attention heads are not to scale.

$$\begin{aligned}\mathbf{Q}_{\text{sp},t}^{(\ell,h)} &= \mathbf{Z}_t^{(\ell)}\,\mathbf{W}_{Q,\text{sp}}^{(\ell,h)},\\ \mathbf{K}_{\text{sp},t}^{(\ell,h)} &= \mathbf{Z}_t^{(\ell)}\,\mathbf{W}_{K,\text{sp}}^{(\ell,h)},\\ \mathbf{V}_{\text{sp},t}^{(\ell,h)} &= \mathbf{Z}_t^{(\ell)}\,\mathbf{W}_{V,\text{sp}}^{(\ell,h)}\end{aligned} \tag{10}$$

where the learnable projection matrices $\mathbf{W}_{Q,\text{sp}}^{(\ell,h)}, \mathbf{W}_{K,\text{sp}}^{(\ell,h)}, \mathbf{W}_{V,\text{sp}}^{(\ell,h)} \in \mathbb{R}^{D\times d_{h,\text{sp}}}$ compress the model embedding dimension D into smaller, head-specific subspaces of dimension $d_{h,\text{sp}} = D/H_{\text{sp}}$, so that $\mathbf{Q}_{\text{sp},t}^{(\ell,h)}, \mathbf{K}_{\text{sp},t}^{(\ell,h)}, \mathbf{V}_{\text{sp},t}^{(\ell,h)} \in \mathbb{R}^{C\times d_{h,\text{sp}}}$. This enables efficient, parallel attention computation across heads. The output of each spatial attention head is then computed from these query, key, and value matrices:

$$\mathbf{S}_t^{(\ell,h)} = \text{softmax}\left(\frac{\mathbf{Q}_{\text{sp},t}^{(\ell,h)}\left(\mathbf{K}_{\text{sp},t}^{(\ell,h)}\right)^\top}{\sqrt{d_{h,\text{sp}}}}\right)\mathbf{V}_{\text{sp},t}^{(\ell,h)}, \qquad \mathbf{S}_t^{(\ell,h)} \in \mathbb{R}^{C\times d_{h,\text{sp}}} \tag{11}$$

The outputs of all spatial heads in a layer are concatenated and projected back to the model dimension:

$$\mathbf{S}_t^{(\ell)} = \text{concat}\left(\mathbf{S}_t^{(\ell,1)}, \dots, \mathbf{S}_t^{(\ell,H_{\text{sp}})}\right)\mathbf{W}_{O,\text{sp}}^{(\ell)}, \qquad \mathbf{S}_t^{(\ell)} \in \mathbb{R}^{C\times D} \tag{12}$$

with $\mathbf{W}_{O,\text{sp}}^{(\ell)} \in \mathbb{R}^{D\times D}$. Stacking over all timesteps gives $\mathbf{S}^{(\ell)} \in \mathbb{R}^{T\times C\times D}$.

Temporal attention is applied along the time dimension. In contrast to using a single shared temporal attention module for all channels or a separate module for each channel, the implemented model uses modality-grouped temporal attention. Let $\mathcal{N}$ denote the set of modality groups, here consisting of EMG, IMU acceleration, and IMU orientation. Each modality group $n \in \mathcal{N}$ contains a subset of channel indices $\mathcal{C}_n \subseteq \{1, \dots, C\}$. Each modality group has its own temporal multi-head attention module, while channels within the same modality group share the temporal attention parameters.

Let the input to transformer layer ℓ for one sequence be $\mathbf{Z}^{(\ell)} \in \mathbb{R}^{T\times C\times D}$. For a channel $c \in \mathcal{C}_n$, the temporal sequence is defined as:

$$\mathbf{Z}_c^{(\ell)} = \left[\mathbf{z}_{1,c}^{(\ell)}, \dots, \mathbf{z}_{T,c}^{(\ell)}\right]^\top \in \mathbb{R}^{T\times D} \tag{13}$$

For modality group n, temporal head $h \in \{1, \dots, H_{\text{tp}}\}$, and transformer layer ℓ, modality-specific projection matrices are used:

$$\begin{aligned}\mathbf{Q}_{\text{tp},c}^{(\ell,h,n)} &= \mathbf{Z}_c^{(\ell)}\,\mathbf{W}_{Q,\text{tp}}^{(\ell,h,n)},\\ \mathbf{K}_{\text{tp},c}^{(\ell,h,n)} &= \mathbf{Z}_c^{(\ell)}\,\mathbf{W}_{K,\text{tp}}^{(\ell,h,n)},\\ \mathbf{V}_{\text{tp},c}^{(\ell,h,n)} &= \mathbf{Z}_c^{(\ell)}\,\mathbf{W}_{V,\text{tp}}^{(\ell,h,n)}\end{aligned} \tag{14}$$

where the projection matrices $\mathbf{W}_{Q,\text{tp}}^{(\ell,h,n)}, \mathbf{W}_{K,\text{tp}}^{(\ell,h,n)}, \mathbf{W}_{V,\text{tp}}^{(\ell,h,n)} \in \mathbb{R}^{D\times d_{h,\text{tp}}}$ map the model dimension D to the per-head dimension $d_{h,\text{tp}} = D/H_{\text{tp}}$, so that $\mathbf{Q}_{\text{tp},c}^{(\ell,h,n)}, \mathbf{K}_{\text{tp},c}^{(\ell,h,n)}, \mathbf{V}_{\text{tp},c}^{(\ell,h,n)} \in \mathbb{R}^{T\times d_{h,\text{tp}}}$. The superscript n indicates that these projections are specific to modality group n.

To prevent information leakage from future timesteps, a causal mask $\mathbf{M}_c \in \mathbb{R}^{T\times T}$ is added to the attention, setting all future timesteps to $-\infty$. For variable-length sequences, a padding mask $\mathbf{M}_p \in \mathbb{R}^{T\times T}$ prevents attention from being applied to the zero-padded timesteps introduced during batching: if the valid sequence length is T_{valid}, all key positions $j > T_{\text{valid}}$ are set to $-\infty$. Together, these masks make temporal attention causal and segment length-aware.

The temporal attention output for channel c, modality group n, layer ℓ, and head h is computed as:

$$\mathbf{T}_c^{(\ell,h,n)} = \text{softmax}\left(\frac{\mathbf{Q}_{\text{tp},c}^{(\ell,h,n)}\left(\mathbf{K}_{\text{tp},c}^{(\ell,h,n)}\right)^\top}{\sqrt{d_{h,\text{tp}}}} + \mathbf{M}_c + \mathbf{M}_p\right)\mathbf{V}_{\text{tp},c}^{(\ell,h,n)} \tag{15}$$

$$\mathbf{T}_c^{(\ell,h,n)} \in \mathbb{R}^{T\times d_{h,\text{tp}}}$$

For each channel $c \in \mathcal{C}_n$, the temporal heads within the same modality group are concatenated and projected back to the model dimension:

$$\mathbf{T}_c^{(\ell,n)} = \text{concat}\left(\mathbf{T}_c^{(\ell,1,n)}, \dots, \mathbf{T}_c^{(\ell,H_{\text{tp}},n)}\right)\mathbf{W}_{O,\text{tp}}^{(\ell,n)}, \qquad \mathbf{T}_c^{(\ell,n)} \in \mathbb{R}^{T\times D} \tag{16}$$

with $\mathbf{W}_{O,\text{tp}}^{(\ell,n)} \in \mathbb{R}^{D\times D}$. Repeating this for all channels in all modality groups and placing the outputs back at their original channel positions gives the temporal output of layer ℓ: $\mathbf{T}^{(\ell)} \in \mathbb{R}^{T\times C\times D}$.

The spatial and temporal outputs are fused by summation, followed by dropout, a residual connection, and layer normalization:

$$\mathbf{U}^{(\ell)} = \text{LayerNorm}\left(\mathbf{Z}^{(\ell)} + \text{dropout}(\mathbf{S}^{(\ell)} + \mathbf{T}^{(\ell)})\right) \tag{17}$$

A pointwise feed-forward network is then applied independently to each token:

$$\mathbf{Z}^{(\ell+1)} = \text{LayerNorm}\left(\mathbf{U}^{(\ell)} + \text{Dropout}\left(\text{FFN}(\mathbf{U}^{(\ell)})\right)\right) \tag{18}$$

After the final transformer block $\ell = N_L$, the representation at the last valid timestep $t = T_{\text{valid}}$ is selected:

$$\mathbf{Z} = \mathbf{Z}_{T_{\text{valid}}}^{(N_L)} \in \mathbb{R}^{C \times D} \tag{19}$$

To aggregate the spatial representations across all channels, a parameterized attention pooling mechanism is employed. First, a learned multi-layer perceptron (MLP) computes a scalar relevance score s_c for each channel representation $\mathbf{Z}_c \in \mathbb{R}^D$:

$$s_c = \mathbf{w}_2^\top \tanh(\mathbf{W}_1 \mathbf{Z}_c + \mathbf{b}_1) + b_2 \tag{20}$$

where $\mathbf{W}_1 \in \mathbb{R}^{(D/2)\times D}$, $\mathbf{b}_1 \in \mathbb{R}^{D/2}$, $\mathbf{w}_2 \in \mathbb{R}^{D/2}$, and $b_2 \in \mathbb{R}$ are the weights and biases of the two-layer pooling network (hidden dimension $D/2$, tanh activation).

These raw scores are normalized across the spatial channel dimension using the softmax function to yield the attention weights α_c. The pooled representation $\mathbf{z}$ is then computed as the attention-weighted sum of all channel embeddings:

$$\alpha_c = \text{softmax}(s_c), \qquad \mathbf{z} = \sum_{c=1}^{C} \alpha_c \mathbf{Z}_c \tag{21}$$

where $\mathbf{z} \in \mathbb{R}^D$. Because each weight α_c quantifies how strongly the model relies on channel c when forming its prediction, these pooling weights are later extracted and aggregated per channel and per sensor group to serve as the model's intrinsic attention-based importance measure (Section 2.4.4, and Section 2.4.5).

Finally, this unified spatio-temporal representation is mapped to a continuous scalar regression:

$$\hat{y} = \mathbf{w}_o \mathbf{z} + b_o \tag{22}$$

where $\mathbf{w}_o \in \mathbb{R}^D$ and $b_o \in \mathbb{R}$ are the regression projection weights and bias. This results in the estimated weight $\hat{y}$, which is the output of the spatio-temporal transformer architecture. This entire process encompasses the nonlinear model function $f(\cdot)$ in Equation 1.

2.4.3. Data augmentation and regularization

To help overcome inter- and intra-subject variability and prevent overfitting, data augmentation and several generalization techniques are implemented. As previously mentioned, AdamW introduces weight decay [17], and the proposed ST-Transformer architecture introduces dropout, both of which act as strong regularizers.

To address the imbalance in training samples across participants and weight classes, a joint participant-weight-class-balancing algorithm was implemented, leveling them out using data augmentation. When a class is underrepresented, data-augmented duplicates are added until the target is reached. When a class is overrepresented, samples are removed, and existing samples have a probability of being augmented. The joint class and participant targets are configured to minimize the number of samples lost in the dataset. Dataset cleansing and the lost samples due to dataset balancing are discussed in Appendix E.

The following data augmentations are introduced for the following reasons:

- **Gaussian noise** is added to the normalized features to make the model less susceptible to very noisy signals.
- **Temporal stretch** scales the sequence length of features using a random factor and accommodates for the changed sequence length through interpolation, simulating variations in velocity while preserving the raw signal physics within each individual sliding window. This makes the model less prone to overfitting on specific movement speeds and participants' specific movement speeds.
- **Channel dropout** will randomly mask a signal, forcing the model not to overfit on a specific channel and regularize its prediction for all channels.

A well-performing parameter configuration was established via a hyperparameter sweep, as described in Appendix D.

2.4.4. Channel Attribution Methods

From a practical and computational standpoint, reducing the number of input channels is desirable, and sensors placed close together can naturally be integrated into a single device. The commercial MYO armband (Thalmic Labs, now discontinued) illustrates this: a compact forearm-worn band combining 8-channel EMG and a 9-DOF IMU with wireless transmission, it has been widely used for EMG- and IMU-based control [29, 55, 59]. The acquisition setup in this study spans the whole arm, so identifying where the load-relevant information concentrates indicates where such a device should be placed.

To address this, several channel attribution methods were used to assess the importance of each channel. Multiple regularization techniques that promote the use of diverse channels for prediction, as well as channels that contain redundant, nonindependent information, made these results inherently ambiguous. Nevertheless, this article uses four complementary methods (ablation, DeepSHAP, attention-pooling weights, permutation importance, and sensor-group retraining ablation) in Section 3.3 and 3.4 to identify the channels and sensor groups most likely to contribute to the prediction.

First, a modality ablation was performed, retraining and evaluating the model on EMG channels only and IMU channels only. This isolates the contribution of each modality and reveals the effect of sensor fusion. An ablation over all individual channel combinations was not conducted due to computational cost.

Second, a DeepSHAP analysis was performed on the trained network [18, 19]. DeepSHAP propagates per-neuron attribution scores backward through the network using DeepLIFT's rescale rule [19], yielding an approximation of the Shapley value of each input feature relative to a baseline distribution. DeepSHAP decomposes a single prediction into feature contributions that sum to the difference between the prediction and the mean prediction over the baseline distribution. The baseline consisted of $N_{\text{bg}} = 200$ randomly sampled training segments, and attributions were computed for $N_{\text{exp}} = 100$ randomly sampled test segments.

The raw attributions $\boldsymbol{\phi} \in \mathbb{R}^{N_{\text{exp}} \times F \times T}$ are defined per input feature. A single importance score per channel is obtained by summing the absolute attributions over the channel's F_c features and over time, averaged across the explained segments:

$$\bar{\phi}_c = \frac{1}{N_{\text{exp}}} \sum_{n=1}^{N_{\text{exp}}} \sum_{f=1}^{F_c} \sum_{t=1}^{T} \left|\phi_{n,c,f,t}\right| \tag{23}$$

where $\phi_{n,c,f,t}$ is the attribution of the f-th feature of channel c for sample n at timestep t. These per-channel scores were grouped by modality, providing both a measure of sensor-fusion effectiveness (analogous to the ablation study) and a measure of channel-level reliance (analogous to the permutation analysis), but derived from the trained network's internal computations rather than from input perturbations.

Third, the model's intrinsic attention-pooling weights were used as an attribution measure. As described in the architecture (Equation 21), the final spatio-temporal representation is aggregated across the C channels by a learned attention-pooling layer, which assigns each channel a normalized weight α_c. Since α_c quantifies how strongly the readout relies on channel c when forming its prediction, these weights provide a channel-importance measure read directly from the trained network. The pooling weights were extracted during inference on the test segments and averaged across segments to obtain a single weight per channel. Like DeepSHAP, this measure is derived from the trained network's internal computations rather than from input perturbations.

Fourth, channel permutation importance was evaluated by randomly shuffling the channels between other segments in the test set and evaluating the model on each permuted variant. For each channel, performance degradation upon perturbation was measured, reflecting the trained model's reliance on that channel.

2.4.5. Sensor Group Ablation Methods

The attribution methods above are test-set or model-intrinsic measures of a channel (except for the modality ablation). Ablation on channel groupings (sensor groups) provides a more conclusive measure of each sensor's contribution, at the cost of training a separate model for each sensor group. Because reducing sensor count is desirable for deployment, and sensors in physical proximity can easily be combined, the channels were partitioned into the following five groups: the upper-arm IMU (IMU-1, 6 channels), the forearm IMU (IMU-2, 6 channels), the shoulder EMG sensors (EMG-S; anterior, lateral, and posterior deltoid; 3 channels), the two upper-arm EMG sensors (EMG-U; biceps and triceps brachii; 2 channels), and the three forearm EMG sensors (EMG-F; brachioradialis, ECR, FCU; 3 channels). A separate model was trained and evaluated for every non-empty subset of these five groups (31 combinations) and for both validation strategies.

For comparison with attribution measures as described in Section 2.4.4, each sensor group receives a single importance score. The contribution of each group was quantified by its Shapley value [1], applied as a sensor importance measure following coalitional-game-theory feature selection [12]. The procedure for finding the sensor-group Shapley values based on different model ablations is analogous to the method used by Hamavar et al. [45] for feature selection in EEG signals. The details for these calculations are described in Appendix C. To compare sensor groups, the per-channel attribution measures from Section 2.4.4 (permutation importance, DeepSHAP, and the attention-pooling weights) were aggregated to each sensor group by summing the scores of its corresponding channels and normalizing across groups.

3. RESULTS

This section presents the best-performing Spatio-Temporal Transformer models and analyzes their results. These models are trained and validated on the cleaned, participant-weight-balanced dataset ($N_{\text{segments}} \approx 10,000$), with details on dataset cleansing found in Appendix E, and all participants are listed in Table 1 (P13 excluded due to age). All reported performance metrics are participant-class-balanced: each metric is computed individually for each participant and weight class, then averaged, so that participants contributing more segments, or the over-represented free-movement (0 kg) class, do not dominate the result. Reported segment counts remain dataset totals.

Two validation strategies are compared. The participant-specialized model uses 5-fold cross-validation stratified across all participants, so every segment appears in the test set once, and performance is measured when the model is trained and tested on the user. The generalized model uses Leave-One-Participant-Out (LOPO) cross-validation, where each fold holds out one participant for testing and two for early-stopping validation (17 folds), measuring transfer to entirely unseen users. It must be noted that the model will perform better on new participants who match the demographic and physiological characteristics of those in Table 1.

The two validation strategies share the same loss function (Equation 6), but are tuned independently, so the hyperparameters, EMG and IMU feature sets, and learning-rate schedule differ (Appendix D). The participant-specialized model trains with a fixed-epoch OneCycleLR, and the generalized model trains with a ReduceLROnPlateau schedule and early stopping to prevent learning of participant-specific patterns. Results are presented in five parts: regression performance (per modality), an in-depth breakdown (participants, segment lengths, weight classes), sensor channel attribution analysis, sensor group ablation analysis, and computational feasibility. Statistical significance is assessed using a Friedman test (Appendix F). The dataset is available on reasonable request, and the code is publicly available on GitHub [65].

3.1. Regression Performance

Figure 7 reports regression performance for the fused, EMG-only, and IMU-only models under both validation strategies. Across all modalities, the participant-specialized models (left column) clearly outperform the generalized models, quantifying the cost of cross-participant transfer. Although participant-specialized training is impractical in deployment, this confirms the architecture's ability to capture the signal dynamics relevant to load estimation. Figure 7 presents these results as Tukey boxplots per weight class: the line marks the median prediction, the box spans the interquartile range (IQR, 50% of predictions), and the whiskers extend to 1.5·IQR, with points beyond plotted as outliers.

Between the participant-specialized models, sensor fusion performs best ($R^2 = 0.935$, MAE $= 0.316$ kg, RMSE $= 0.441$ kg), ahead of EMG-only ($R^2 = 0.913$, MAE $= 0.380$ kg, RMSE $= 0.520$ kg) and IMU-only ($R^2 = 0.719$, MAE $= 0.702$ kg, RMSE $= 0.974$ kg). Both modalities perform reasonably on their own, with EMG being clearly the superior modality, and fusing them improves accuracy (the improvement is statistically significant, Appendix F, Table 5).

Under generalization (LOPO), the benefit of sensor fusion is smaller and not statistically significant(Appendix F, Table 5). However, all performance metrics show slight benefits for sensor-fusion in the generalized model: sensor fusion performs best ($R^2 = 0.853$, MAE $= 0.536$ kg, RMSE $= 0.680$ kg), slightly ahead of EMG-only ($R^2 = 0.839$, MAE $= 0.546$ kg, RMSE $= 0.703$ kg) and far ahead of IMU-only ($R^2 = 0.491$, MAE $= 0.989$ kg, RMSE $= 1.298$ kg). The cost of cross-participant transfer differs by modality: the IMU-only model retains only 68.3% of its specialized R^2 ($0.719 \rightarrow 0.491$), whereas the EMG-only and fused models both retain $\approx 92\%$ ($0.913 \rightarrow 0.839$ and $0.935 \rightarrow 0.853$). Kinematic signatures are more subject-specific, while muscle activation generalizes better to unseen users.

3.2. In-Depth Performance

The first row of Figure 8 reports per-participant performance. Beyond the difference in performance between validation strategies, the spread across participants also differs: the generalized model varies roughly twice as much across participants as the specialized model ($\sigma_{\text{gen}} = 0.106$ kg against $\sigma_{\text{spec}} = 0.056$ kg for the fused model). This is expected, as the specialized model learns participant-specific patterns, whereas the generalized model's accuracy on a held-out participant depends on how closely that participant resembles the training set. The two single-modality generalized models fail in different ways. The IMU-only model spreads widely and symmetrically across participants ($\sigma = 0.19$ kg, Mean $= 0.99$ kg $\approx$ Median $= 0.94$ kg, range 0.67–1.37 kg): it is uniformly weak, indicating that kinematic load cues transfer evenly poorly to all unseen users. The EMG-only model spreads less overall ($\sigma = 0.17$ kg) but is more skewed (Mean $= 0.55$ kg above Median $= 0.50$ kg): it is accurate for most participants while

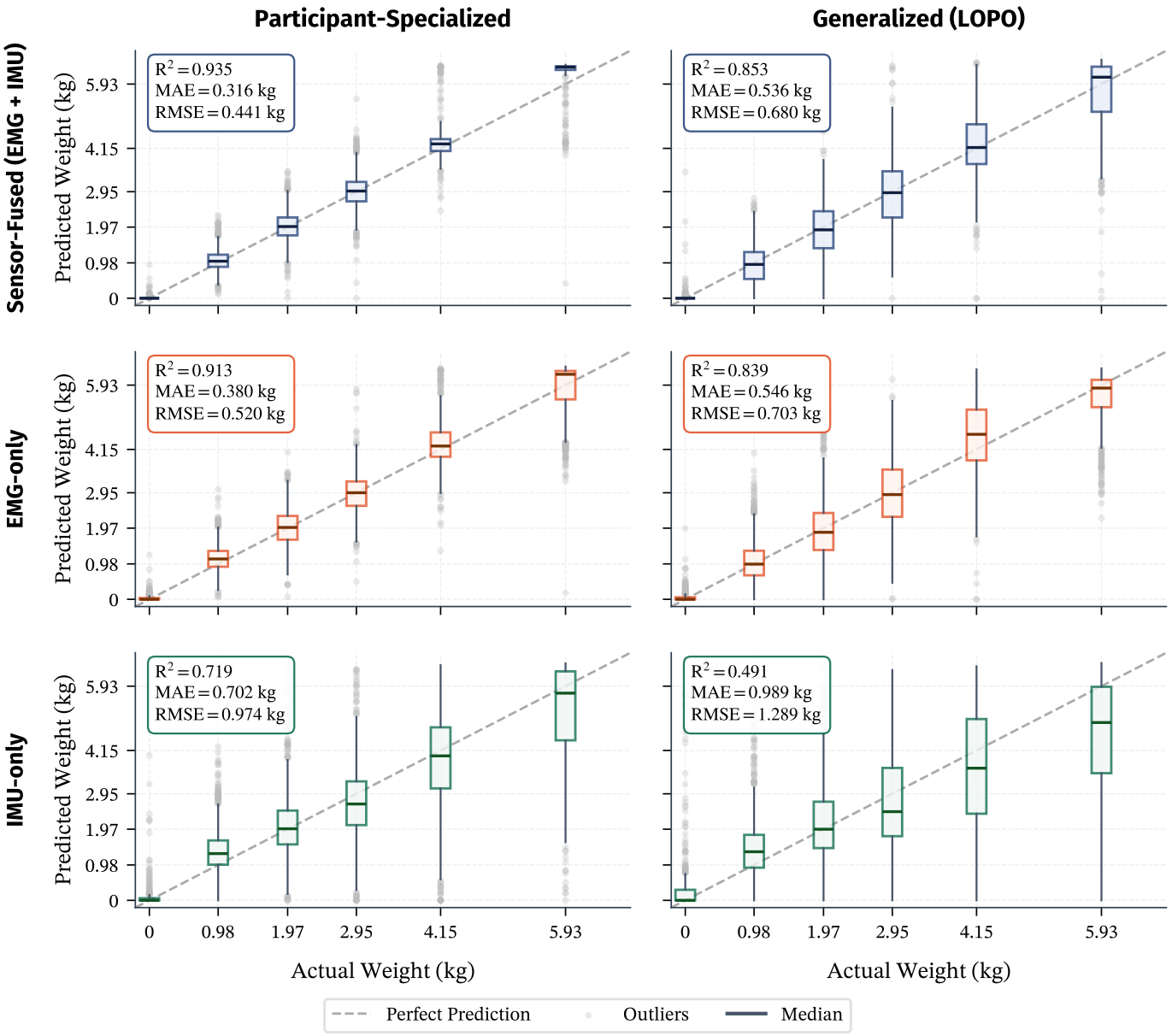


Figure 7. Regression performance across the three sensor modalities (rows: sensor-fused, EMG-only, IMU-only) and two validation strategies (columns: participant-specialized 5-fold cross-validation, generalized leave-one-participant-out). All metrics are computed per participant and averaged across participants with equal weight.

a few transfer poorly (the spikes in Figure 8). This spread between participants (and the limited number of participants) explains the lack of statistical significance between the generalized sensor-fused and EMG-only models. Across participants, fusion lowers MAE for only 10 of 17 and by an average of 0.010 kg, small against the 0.129 kg standard deviation of that per-participant difference.

The second row of Figure 8 shows regression performance as a function of segment length (L in Equation 1), i.e., the number of sliding-window feature sets the model accumulates before its prediction. Bins are retained only over the range in which all six weight classes contain at least five segments, which truncates the results beyond 2.5 s. Error peaks for mid-length segments (1.6 s for the generalized and 1.5 s for the specialized model) and then falls steadily toward the longest windows, so over the full range the trend is only moderate and non-significant (generalized MAE $r = -0.436$, $p = 0.055$; specialized $r = -0.337$, $p = 0.146$. Beyond the peak, however, additional temporal context produces a statistically significant reduction in error: the generalized post-peak MAE correlates with duration at $r = -0.839$ ($p = 0.002$), declining at 0.22 kg/s from 0.574 kg (1.6 s) to 0.351 kg (2.5 s), a 38.9% reduction. The specialized model shows the same: ($r = -0.645$, $p = 0.032$; 0.336 → 0.220 kg, a 34.5% reduction). The longest-segment bins include the fewest segments ($N_{\text{segments},2.5} = 89$), so the gain in information content from a longer observation window is more valuable than the number of training segments, demonstrating the benefit of the architecture's variable-length handling.

The third row of Figure 8, together with Table 2, breaks performance down by weight class. Each load trend is defined by β, the slope of a least-squares linear fit of MAE to applied weight (error in kg per kg of load). Absolute error grows with load for both strategies: the fused fit gives $\beta = 0.091$ ($r = 0.972$, $p = 0.001$) for the specialized model and $\beta = 0.113$ ($r = 0.875$, $p = 0.022$) for the generalized model, with the IMU-only model steepest in both regimes ($\beta = 0.147$, $r = 0.916$, $p = 0.010$ specialized; $\beta = 0.216$, $r = 0.960$, $p = 0.002$ generalized). The free-movement (0 kg) class is by far the easiest, predicted near-perfectly by the fused and EMG-only models (MAE ≤ 0.07 kg), while the IMU-only models are worse (MAE = 0.13 kg specialized, MAE = 0.31 kg generalized). The rise tends to plateau or slightly reverse at the heaviest, most widely separated class (5.93 kg). This is most

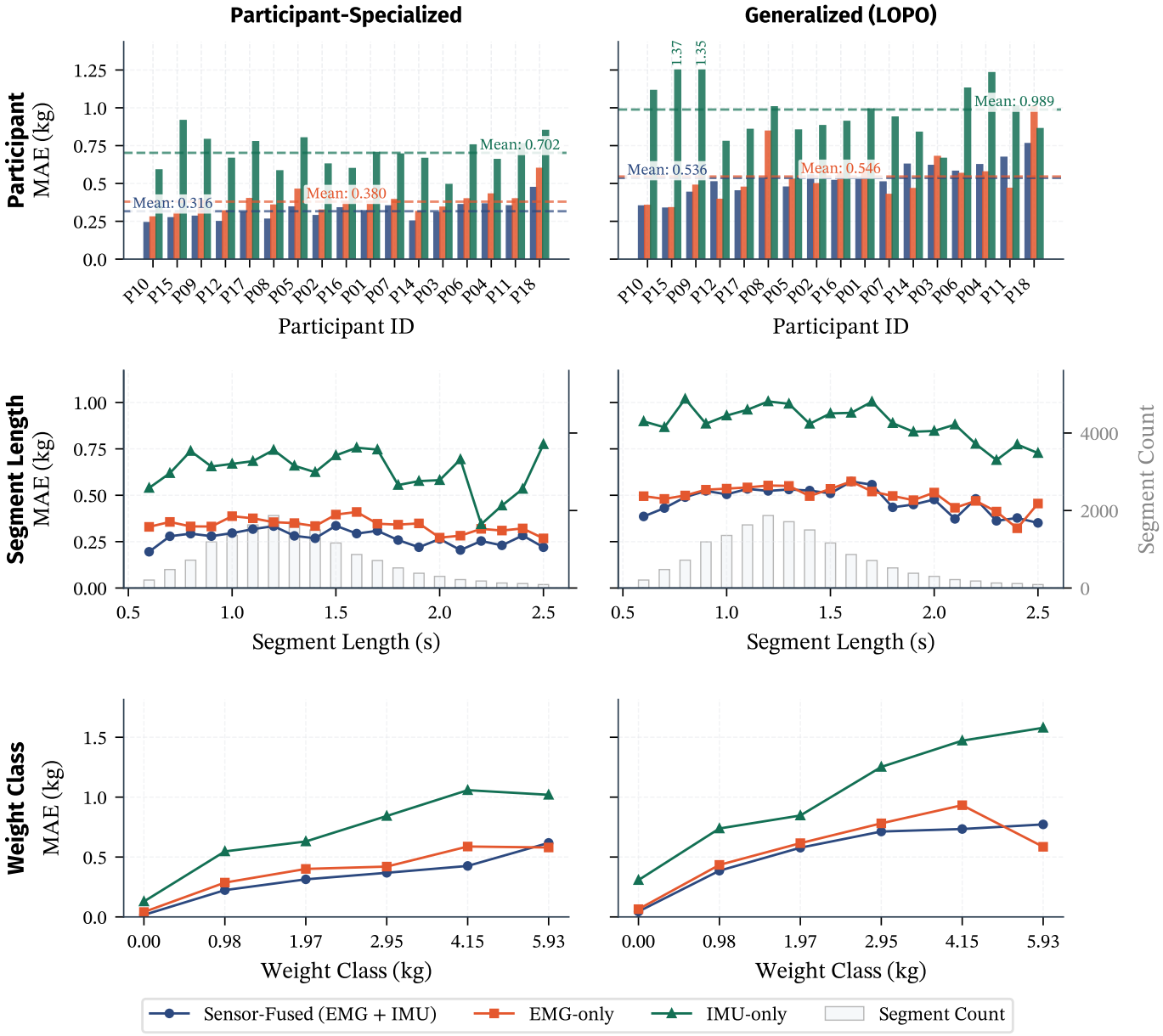


Figure 8. In-depth performance (based on MAE) across both participant-specialized and generalized models, for each modality. Includes performance across participants, segment lengths, and weight classes. RMSE variant found in the Appendix as Figure 14.

pronounced for the generalized EMG-only model, whose error peaks at 4.15 kg (0.93 kg) before falling to 0.59 kg at 5.93 kg. In relative terms, accuracy improves with load: the fused error falls from 23% of the applied weight at 0.98 kg to 10% at 5.93 kg for the specialized model, and from 39% to 13% for the generalized model, so absolute error grows sub-proportionally and heavy loads are estimated most precisely relative to their magnitude. Due to the large performance gap between the free-movement and loaded classes, the overall metrics can be skewed. Table 2 therefore reports the two groups separately, revealing substantially higher error on the loaded classes for every model.

3.3. Sensor Channel Attribution

Channel attribution to the prediction was assessed with three complementary, model-faithful measures (Figure 9): DeepSHAP, the model's intrinsic attention-pooling weights, and permutation importance. The modality ablation (Figures 7 and 8) already established EMG as the stronger modality for load estimation in both settings, and all three attributions reflect this, assigning EMG the majority of total importance in both models. DeepSHAP attributes 68.3% of importance to EMG in the specialized model and 68.4% in the generalized model (versus 31.7% and 31.6% for the IMU). The attention weights attribute 66.0% of importance to EMG in the specialized model and 79.6% in the generalized model (versus 34.0% and 20.4% for the IMU). The three measures also attribute importance similarly at the channel level, and the rankings are strongly correlated across importance methods (Spearman: $r = 0.88$, $p = 2.32 \cdot 10^{-7}$ for DeepSHAP; $r = 0.72$, $p = 3.41 \cdot 10^{-4}$ for permutation; $r = 0.75$, $p = 1.26 \cdot 10^{-4}$ for the attention weights), so the model relies on the similar channels whether personalized or generalized. The main difference is that the generalized model focuses more on the most reliable channels: its top three channels account for 84% of total permutation importance, compared with 75% for the specialized model, and the attention weights show the same shift even more sharply, concentrating 44% of their total mass on the Brachioradialis and Extensor Carpi Radialis (ECR) alone, versus 25% in the specialized model.

All three measures agree on which channels matter. DeepSHAP

Table 2. Performance comparison for the free-movement weight class (0.00 kg) and the loaded weight classes (> 0.00 kg) across sensor modalities for the participant-specialized (Spec.) and generalized LOPO (Gen.) models. All metrics are participant-class-balanced. R^2 cannot be defined for a single class.

Modality	Weight Class	Metric	Spec.	Gen.
Sensor-Fused	Free Movement	MAE (kg)	0.017	0.045
		RMSE (kg)	0.111	0.154
	Loaded	MAE (kg)	0.390	0.637
		RMSE (kg)	0.570	0.826
		R^2	0.885	0.758
EMG Only	Free Movement	MAE (kg)	0.041	0.065
		RMSE (kg)	0.157	0.213
	Loaded	MAE (kg)	0.454	0.670
		RMSE (kg)	0.644	0.881
		R^2	0.855	0.723
IMU Only	Free Movement	MAE (kg)	0.129	0.307
		RMSE (kg)	0.433	0.655
	Loaded	MAE (kg)	0.819	1.179
		RMSE (kg)	1.119	1.539
		R^2	0.565	0.161

and permutation both rank the Brachioradialis first, and the attention weights place it first in the generalized model (ranking ECR marginally ahead of it in the specialized model). The orderings are strongly correlated (DeepSHAP vs. permutation Spearman: $r = 0.812$, $p = 1.38 \cdot 10^{-5}$ specialized, $r = 0.715$, $p = 3.87 \cdot 10^{-4}$ generalized; attention vs. DeepSHAP: $r = 0.86$ and 0.92; attention vs. permutation: $r = 0.80$ and 0.75). They differ mainly in how concentrated that importance is. DeepSHAP attribution and the attention weights are both spread relatively evenly across channels, whereas permutation importance is concentrated: in the specialized model, the single most important channel accounts for only ∼ 18% of total DeepSHAP attribution and ∼ 15% of the attention mass, but ∼ 47% of total permutation degradation. This indicates that the model, encouraged by dropout, weight decay, and channel dropout, learns a distributed representation rather than relying on a few channels. Permutation exposes redundancy: 15 of the 20 channels each contribute less than 5% of the most important channel's degradation. This suggests the model could retain comparable accuracy with substantially fewer channels.

3.4. Sensor Group Ablation

The model was retrained on every combination of the five sensor groups defined in Section 2.4.5. The heatmap in Figure 10 shows the participant-class-macro MAE of all 31 combinations under both validation strategies. Under the participant-specialized strategy, the full 20-channel set performs best (0.323 kg), but several reduced sets come close. EMG groups dominate: the EMG-only set (8 channels) reaches 0.375 kg, far ahead of the IMU-only set (12 channels, 0.708 kg), and the three forearm EMG (EMG-F) channels alone (0.419 kg) outperform all other channels combined. Dropping the shoulder EMG (EMG-S) and the upper-arm IMU (IMU-1) loses almost nothing. The following subset is within 1.5% of the full model: forearm IMU (IMU-2), forearm EMG, and upper-arm EMG (EMG-U) (11 channels, 0.327 kg). A purely forearm-worn set (IMU-2 and EMG-F, 9 channels, 0.358 kg), as a MYO-like device, would retain most of the performance while benefiting from the two additional upper-arm EMG channels.

Ablation under the cross-participant (LOPO) strategy reproduces this pattern at a consistently higher MAE. The full set reaches 0.512 kg, the EMG-only set (0.606 kg) again far outperforms the IMU-only set (0.987 kg), and the three forearm EMG channels alone (0.629 kg) still beat all other channels combined. The forearm-worn MYO-like set holds up (0.592 kg), and a forearm-plus-shoulder EMG set with a single IMU (12 channels, 0.509 kg) matches the full model. The two strategies rank the 31 combinations almost identically (Spearman: $r = 0.93$, $p < 0.001$), so the practical conclusion (that load-relevant

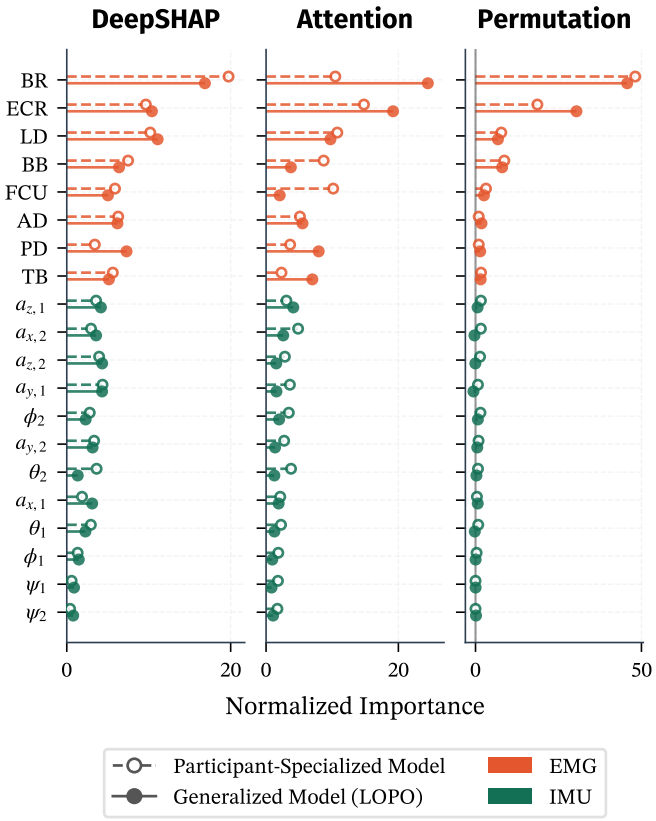


Figure 9. Channel importance for the participant-specialized and generalized sensor-fused models. Each plot shows the normalized per-channel importance for one attribution method: DeepSHAP (left), permutation (middle), and attention pooling (right). Participant-specialized versus generalized (LOPO) for both modalities.

information concentrates in the forearm EMG) transfers from within-subject calibration to unseen participants.

The lower two plots of Figure 10 compare the retraining-based ablation (Shapley) value of each group (Appendix C) with the normalized permutation, DeepSHAP, and attention-pooling importances, for the participant-specialized (upper) and generalized (lower) models. All four measures rank forearm EMG first by a wide margin (Shapley share 0.32 specialized, 0.33 generalized). Permutation tracks the Shapley ordering most closely and is the only attribution significant under both strategies (Spearman: $r = 0.90$, $p = 0.037$ specialized; $r = 1.00$, $p = 0.017$ generalized). Attention matches Shapley under generalization ($r = 0.90$, $p = 0.037$) but not specialization ($r = 0.30$, $p = 0.62$), and DeepSHAP correlates weakly in both regimes ($r = 0.30$, $p = 0.62$ specialized; $r = 0.60$, $p = 0.29$ generalized). The two model-internal attributions diverge from the retraining measures mainly by promoting shoulder EMG to second place, whereas Shapley and permutation rank it in the middle. All four nonetheless agree on the forearm-EMG dominance, confirming that the load-relevant information concentrates in the forearm muscles, whether the model is calibrated within-subject or generalized to unseen participants.

3.5. Computational Feasibility

Because the model is intended for real-time control, each estimate must be produced within the prediction interval $\Delta t = 100$ ms. The window length L trades accuracy (favoring longer windows) against responsiveness (favoring shorter ones): the segment-length analysis (Figure 8) shows error decreasing toward longer observation windows, while the 100 ms interval bounds how often a new estimate is issued. Within this budget, two costs must fit: feature extraction and a single model forward pass.

The parameter, memory, and operation counts below are exact properties of the trained model: the GPU timing is measured, the microcontroller latencies are analytical projections, as no on-hardware deployment was performed. The sensor-fused model has $N_{\text{params}} \approx 1.03 \cdot 10^6$ trainable parameters, occupying 4.12 MB in FP32 and 1.03 MB after post-training INT8 quantization, within the 2 MB on-chip Flash of an advanced microcontroller such as the STM32H7[4]. A single forward pass requires $\approx 1.15 \cdot 10^6$ multiply–accumulate operations and executes in 0.62 ms on the host NVIDIA A100 GPU.

Projected onto an ARM Cortex-M7 (480 MHz) under the throughput assumptions detailed in Appendix G, INT8 inference is bounded at $\approx$ 10 ms and feature extraction at $\approx$ 15 ms, a combined $\approx$ 25 ms per update and a $\gtrsim$ 75% idle margin within the 100 ms interval. These estimates indicate that the pipeline is feasible for real-time edge execution on current microcontrollers. On-hardware validation is left to future work (Section 5). A more detailed description of this feasibility estimate is given in Appendix G.

4. DISCUSSION

The central result in this paper is the effect of sensor fusion on load estimation. Fusing EMG and IMU improves the estimate, significantly so within a participant (specialized: SF < EMG < IMU, generalized: SF $\approx$ EMG < IMU, Table 5). Across participants, the fused model reaches the lowest error of the three modalities but is not significantly better than EMG alone (Figure 7, Table 2). The IMU thus carries load information that is complementary to the stronger EMG modality within a participant, but potentially redundant across participants. The IMU senses load only through the resulting motion (for a given movement, $a = F/m$), which has two consequences. First, the IMU is the weaker modality within a participant (specialized participant-class-macro MAE 0.702 kg versus 0.380 kg). The IMU measures the acceleration profile that differs due to the load's inertia, whereas EMG measures the direct muscle command that moves it. A direct readout of effort is more informative than its mechanical consequence. Second, the IMU transfers worse to unseen participants, retaining only 68.3% of its specialized R^2, compared with roughly 91% for the EMG-only and fused models. Recovering load from acceleration requires the participant's limb inertia, anthropometry, and sensor placement, so the IMU's load information is entangled with the participant's specific information. Previous work already established difficulties in cross-participant IMU-based recognition due to these factors [27]. EMG holds load information that is subject-variable too [51], but after MVC-style normalization, it maintains a more transferable relationship between effort and force, so transfer succeeds when a held-out participant resembles those of the training set.

These findings extend a well-documented observation. Inter-subject variability, e.g., from anatomy, sensor or electrode placement, and movement style, is the central barrier to cross-participant generalization, both in IMU-based activity recognition [27] and in EMG-based control [51]. Consistent with this, Zhao et al. [64] report a comparable cross-participant degradation (97.3% → 88.2%) for EMG state classification; Pesenti et al. [37] achieve strong within-participant IMU-based load estimation (88.16% median) without testing transfer; and Kumar et al. [58] jointly estimate motion parameters and external load from EMG using physics-informed modeling, with cross-participant transfer. This paper quantifies how inter-subject variability affects load and localizes it: the transferable load signal is carried by muscle activation, whereas the IMU's contribution appears participant-specific. Therefore, fusion holds only a slight, non-significant edge over EMG alone in the generalized model. The largest gain lies in extracting that participant-specific IMU information in a transferable form, potentially using per-subject calibration, gain-invariant features, or adversarial disentanglement of subject identity [60] (Section 5).

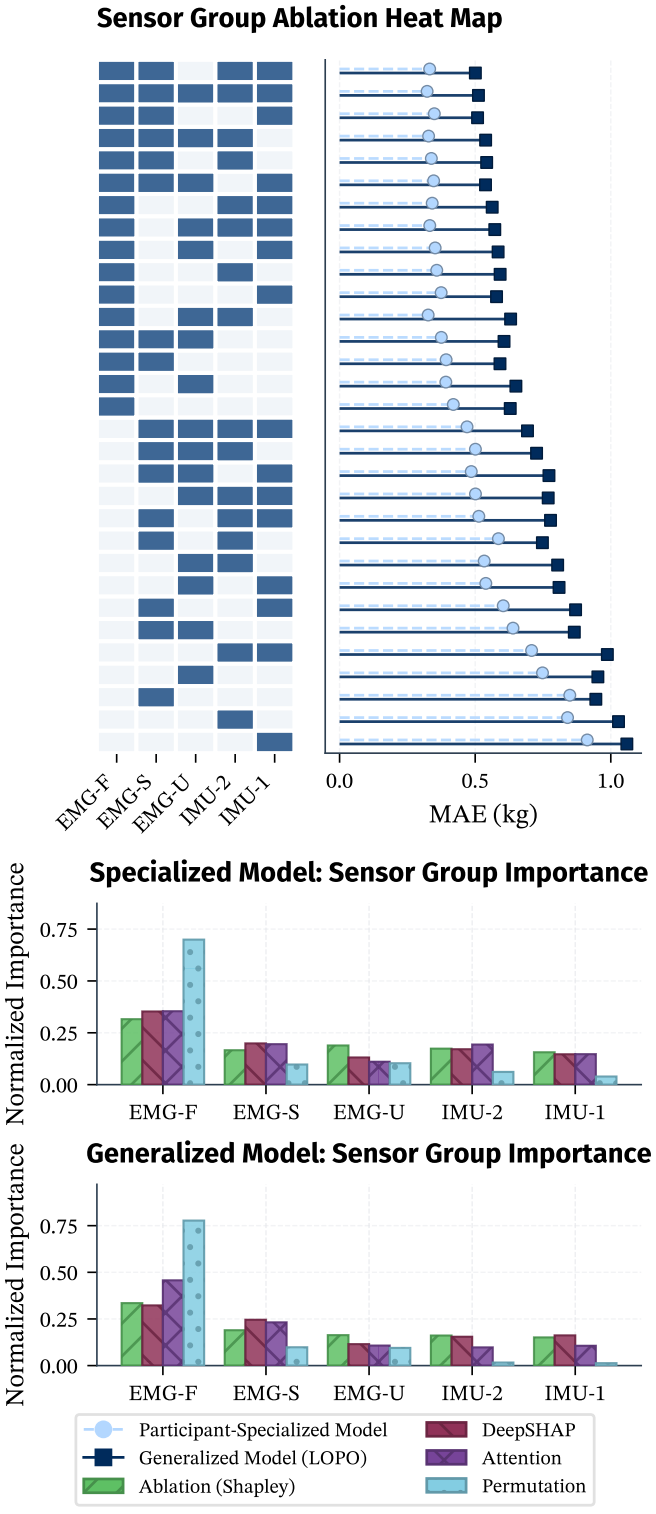


Figure 10. Sensor group ablation using both validation strategies. Top: MAE and heatmap for all 31 subsets. Bottom: per-group retraining Shapley value compared against the normalized DeepSHAP, attention-pooling importances, and permutation. RMSE variant found in the Appendix as Figure 15.

Across weight classes, all models distinguish free movement from loaded movement well, and error grows with load while relative error falls (Figure 8). As effort increases, absolute EMG amplitude becomes less reliable [14], an effect magnified by the normalization used here (an approximation of maximum voluntary contraction (MVC) via the 99th percentile of each session [30]). Absolute error grows with weight ($\beta = 0.091$ sensor-fused specialized, $\beta = 0.113$ sensor-fused generalized), but relative error shrinks over weight classes (23% → 10%

[4]STM32H7 - Arm Cortex-M7 (480 MHz), URL: https://www.st.com/en/microcontrollers-microprocessors/stm32h7-series.html

sensor-fused specialized, 39% → 13% sensor-fused generalized). Yet the largest class is also the most widely separated, with no intermediate 5 kg level, so it is comparatively easy to distinguish. This is visible as the dip in EMG-only error at 5.93 kg relative to 4.15 kg, and the class carries the lowest relative error. A more granular set of weights would be needed for more conclusive results.

Estimation accuracy also depends on how much of the movement the model has observed. The low error at the shortest segments likely reflects movement length rather than observation time: short segments correspond to small-distance moves, in which the load is displaced only slightly and the EMG signal stays comparatively clean, whereas longer-distance moves accumulate more motion-related variability. However, this hypothesis is not validated. Within the range of longer segment lengths, error falls steadily once more information is accumulated: beyond the peak at 1.6 s, the generalized MAE drops by 38.9% (to 0.351 kg at 2.5 s), and the specialized model behaves the same way. Because the longest windows are also the least represented in the dataset, this gain is due to longer observations rather than to more training data at that length. For real-time control, it sets up a trade-off between accuracy and responsiveness: a longer window yields a better load estimate but delays it. A practical controller could send an early estimate from a short window and refine it as the lift progresses, which the architecture's variable-length, streaming operation directly supports.

Architecturally, the spatio-temporal transformer of Aksan et al. [26], originally designed for autoregressive 3D human-motion prediction over skeletal joints, was adapted in three main ways for this task. First, the spatial dimension was reinterpreted from skeletal joints to sensor channels, so that spatial attention models cross-sensor (EMG–IMU) relationships at each timestep rather than inter-joint dependencies. Second, the per-joint temporal attention of the original was replaced with modality-grouped temporal attention. EMG, IMU-acceleration, and IMU-orientation channels each receive their own temporal-attention parameters while sharing within a group, reflecting the distinct temporal dynamics of muscle activation and limb kinematics. Third, the autoregressive motion decoder was replaced with an attention-pooling regression over the channels at the final timestep, producing a single scalar load estimate, and causal and padding masks were added to enable the model to operate on variable-length segments.

With these adaptations, the architecture performs in the same range as the strongest baselines (Appendix H, Figure 12, Table 6). Within participants, the recurrent models (LSTM, GRU) reach significantly lower MAE than the ST-Transformer (0.18 and 0.20 versus 0.32 kg; $p_{\text{adj}} = 0.006$), but the gap is limited to MAE: on RMSE, the recurrent models and the ST-Transformer are statistically indistinguishable. This advantage of MAE stems from the fact that the classes are discrete. The recurrent models minimize error by snapping their predictions onto the six training loads, observable in Figure 13 (94% of LSTM and 91% of GRU predictions land within 10% of a training load, against 65% for the ST-Transformer), behaving more like classifiers than continuous estimators, a strategy that lowers MAE but not RMSE. Crucially, the advantage does not transfer across participants. Under LOPO, the recurrent models no longer snap to the discrete classes, and the difference is no longer significant: the LSTM, GRU, and ST-Transformer form a single statistically indistinguishable top group (0.49, 0.50, and 0.54 kg), with the vanilla transformer and the convolutional models behind (Table 6). The ST-Transformer, therefore, aligns with well-established model architectures in generalized modeling. With a prediction geometry essentially unchanged between participant-specialized and generalized.

The spatial and temporal attention map naturally onto this problem, which motivated the choice of architecture. Spatial attention introduces the cross-sensor (EMG–IMU) coupling on which fusion depends, and modality-grouped temporal attention captures muscle activation and limb kinematics independently. The model is competitive with the established sequence baselines (LSTM and GRU) while offering an explicit fusion mechanism. Another advantage is that the attention mechanism also makes the model intrinsically interpretable. The attention-pooling head assigns each sensor channel an explicit, normalized weight, which served as a model-intrinsic channel-attribution measure (Section 3.3). The spatial and temporal attention maps offer a further, complementary view of the model's internal reasoning, revealing which sensor channels attend to one another and which timesteps within a movement the model attends. These maps were not yet analyzed in this paper and are left to future work (Section 5). Because the architecture is inherited from a multi-joint motion predictor, it also carries capacity beyond what scalar load regression uses. Restoring a sequence-prediction head would potentially let the same spatio-temporal backbone unify load and multi-joint motion prediction, a model that assistive control ultimately requires (Section 5). Its appeal, therefore, lies in the fusion mechanism it offers, the intrinsic interpretability of its attention, and its extensibility toward a unified load estimation and motion-prediction model.

The results also point toward a compact, deployable sensor set. The ablation shows the load-relevant information concentrates in the forearm, a ranking that is near-identical across both validation strategies (Spearman $r = 0.93$): the three forearm EMG channels alone outperform all other channels. Within participants, a forearm-worn combination of EMG and IMU (9 channels) recovers most of the full model's accuracy, reaching within 1.5% of it once two upper-arm EMG channels are added (11 channels). Under cross-participant generalization, the same reductions cost more (roughly 15–25%), so such a forearm device benefits most when paired with calibration. A wrist- or forearm-mounted device along the lines of the MYO armband would therefore capture most of the usable signal, which is attractive for cost, integration, and practicality.

Several limitations bound these conclusions. Both modalities are normalized using per-session statistics (a 99th-percentile envelope peak for EMG and a robust median/IQR for IMU), which is itself a form of per-participant calibration. The generalized model is therefore not fully decoupled from each held-out participant's data, and a fully calibration-free deployment may transfer somewhat less well. The computational analysis is estimated rather than measured on hardware, and the load estimate is not yet evaluated in a closed loop within an assistive device. The interaction between human, system, and environment would alter dynamics and signal characteristics, likely requiring application-specific retraining. The dataset is not demographically diverse (17 participants aged 22–27) and uses six discrete, cylindrical weights moved across a button matrix, so grasp and posture are far more uniform than in real object manipulation. The binary button state also limits labeling to discrete classes and introduces a fixed pickup/in-air/drop-off segment structure absent in continuous deployment. The channel-level attribution estimates rely on measures evaluated on the trained model at test time (DeepSHAP, permutation importance, and the model's attention-pooling weights). Only the sensor-group importances were additionally checked against a retraining ablation (Section 3.4). There, permutation importance closely matched the retraining ranking, and all four measures identified the forearm EMG as the dominant group, though there was no agreement on the order of the lower-ranked groups. The architectural modifications themselves (the channel-wise spatial dimension, modality-grouped temporal attention, and the attention-pooling head) were motivated by the fusion and regression task but not isolated in ablation tests. The reported comparison establishes that the architecture as a whole is competitive with standard recurrent baselines and outperforms convolutional ones, not that each modification contributes independently.

5. FUTURE WORK

The primary direction for future work is to improve the fusion advantage and to establish it statistically for LOPO generalized modeling, thereby closing the gap between the participant-specialized and gen-

eralized models. The load-relevant IMU information is entangled with participant-specific factors such as limb inertia, muscle gain, and sensor placement, so the goal is to recover the load-relevant component in a participant-invariant form. Two further strategies could help to reach this goal. First, because the load reaches the IMU largely through this participant-specific information, representations that are invariant to it, such as physics-informed models or informative features relating EMG and IMU, could reduce the model's reliance on subject-specific features and improve transfer [58]. Second, at the architectural level, contrastive learning could extract discriminative cross-modal representations [31], while adversarial learning could explicitly disentangle participant-specific information from load-relevant IMU features [60]. Should full generalization remain out of reach, the significantly stronger participant-specialized performance makes per-user retraining a reasonable fallback. This approach is undesired, since labeling currently depends on the custom button matrix and is impractical outside a data-collection setting.

Beyond the modeling, the experimental setup and hardware leave substantial room for improvement. An embedded EMG and IMU system capable of on-device inference should be developed to ensure hardware-synchronized, higher-quality data and to streamline integration. Such a system would address the limitations of the current setup, including the jumper-wire connections, unaddressed timing delays, and the IMU sampling rate, which was interpolated to match the EMG rate. A natively higher IMU sampling rate may yield richer features and a larger overall IMU contribution. The button matrix should likewise be reconsidered. Replacing it with pressure pads would eliminate the small timing delays introduced at pickup and allow continuous weight labeling instead of discrete labeling. Continuous weight labels would enable training on sliding-window segments that capture continuous picking-and-placing dynamics rather than uniform segments, yielding a more realistic representation of real-world behavior and mitigating the segmentation bias that assumes each segment contains a pick-up, in-air, and drop-off phase.

The dataset should also be expanded in two ways. First, a more demographically diverse participant selection is required: the current group is not diverse (predominantly right-handed participants aged 22-27 from the Netherlands), which would enable stronger generalization. Second, the interactions themselves should be more varied and realistic. The current setup uses cylindrical weights of equal diameter, which results in identical grasping across loads and overly uniform movements. Adding more complex grasping and movements, along with more weight classes, would increase dataset diversity, better reflecting the wide variation in arm and hand postures in real-world object manipulating.

The attention mechanism also offers more interpretability beyond what was used in this paper. The attention-pooling weights were used here as a channel-importance measure (Section 3.3), but the model's spatial and temporal attention maps were not analyzed. Spatial attention across sensor channels directly reveals the cross-sensor coupling on which fusion depends, including which EMG and IMU features attend to one another, whereas causal temporal attention reveals which phases of a lift the model attends. Examining these maps could clarify how and when the IMU complements EMG, informing sensor placement and feature design. Because the reliability of attention weights is up for debate [23, 25], such explanations would need to be validated with other methods, like the ones used in this paper.

Finally, deploying this algorithm within an assistive robotic control loop introduces requirements beyond load estimation alone. Integrating the model and its sensory system into a physical device requires developing an application-specific variant of the model. Since movement and load-bearing are not independent actions, future work should investigate combining motion-intention prediction and load estimation, either as parallel estimators or as a unified load-motion prediction model. A unified load–motion estimator is feasible: Kumar et al. [58] jointly predict joint kinematics, torque, and external load from sEMG with a physics-informed BiGRU that generalizes across subjects without retraining, though from a single modality rather than fused EMG and IMU. This compatibility motivated the choice of the spatio-temporal transformer in the first place: its explicit spatial and temporal attention mirrors the structure of the spatio-temporal motion-prediction model by Aksan et al. [26], making the two naturally compatible. A shared spatio-temporal attention backbone could thus jointly perform motion prediction and load estimation, potentially providing the assistive controller with both.

6. CONCLUSION

Active exoskeletons and assistive robotic devices could provide more effective support by adapting their assistance to the weight of the objects a user manipulates. This requires inferring the held load in real time from the device's own sensors. With this objective, synchronized EMG and IMU signals were used as inputs to a spatio-temporal transformer (ST-Transformer) that regresses the held load. The architecture is competitive with the strongest established architectures (GRU, LSTM), and was adopted as the primary model for its explicit sensor-fusion mechanism, its intrinsic interpretability, and its natural extensibility toward joint load–motion estimation.

This paper showed that the model estimates the held load from arm muscle activation and limb kinematics with $R^2 = 0.935$, MAE 0.316 kg, and RMSE 0.441 kg when trained and tested on the same participants, dropping to $R^2 = 0.853$, MAE 0.536 kg, and RMSE 0.680 kg across unseen participants. All metrics are participant-class-balanced. Fusing EMG and IMU was significantly more accurate than either modality alone within a participant, but in the generalized (LOPO) setting, the fused model held only a slight, non-significant edge over the EMG-only model. The IMU-only model degraded sharply during generalization, indicating that load-relevant IMU information is entangled with participant-specific characteristics. The central finding is therefore that the transferable load signal is carried primarily by muscle activation, and that the cross-participant fusion problem is one of transfer rather than absence of information.

An attribution analysis combining modality ablation, DeepSHAP, permutation importance, attention-pooling weights, and a sensor-group retraining ablation confirmed this redundancy and localized the load-relevant signal to the forearm muscles: a compact forearm-worn EMG–IMU set recovered most of the full-array accuracy, indicating that a wrist- or forearm-mounted device could capture the bulk of the usable signal. The computational analysis estimated that the model is feasible for real-time control.

The dataset was self-collected using a custom, synchronized acquisition setup (a TMSi Porti7 EMG system and dual BNO085 IMUs, read by an STM32F401 microcontroller)as well as a custom-built button matrix that enabled automatic segmentation and labeling. The central problem is the gap between participant-specialized and generalized performance. Closing it is a priority for future work through improved calibration or domain adaptation, and through a more diverse, realistic dataset. Further development toward real-world application should pair this with an improved acquisition system, deployment of the model on a physical device, and an extension toward combined load–motion estimation for assistive control.

ACKNOWLEDGEMENTS

- I thank Arno Stienen and Xucong Zhang for their guidance, and the volunteers who participated in the data collection.
- I acknowledge the use of computational resources of the DelftBlue supercomputer, provided by the Delft High Performance Computing Centre [40].
- Google's Gemini and Anthropic's Claude language models were used to assist with code development and as a writing aid. All outputs were reviewed by the author, who takes full responsibility for the content.

REFERENCES

[1] L. S. Shapley, *17. A Value for n-Person Games*. Dec. 1953, pp. 307–318. DOI: 10.1515/9781400881970-018. [Online]. Available: https://doi.org/10.1515/9781400881970-018.

[2] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, Mar. 1960. DOI: 10.1115/1.3662552. [Online]. Available: https://doi.org/10.1115/1.3662552.

[3] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, 1976. DOI: 10.1109/TASSP.1976.1162830.

[4] R. W. Norman and P. V. Komi, "Electromechanical delay in skeletal muscle under normal movement conditions", *Acta Physiologica Scandinavica*, vol. 106, no. 3, pp. 241–248, Jul. 1979. DOI: 10.1111/j.1748-1716.1979.tb06394.x. [Online]. Available: https://doi.org/10.1111/j.1748-1716.1979.tb06394.x.

[5] S. Hochreiter and J. Schmidhuber, "Long Short-Term memory", *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. DOI: 10.1162/neco.1997.9.8.1735. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. DOI: 10.1109/5.726791.

[7] H. J. Hermens, B. Freriks, C. Disselhorst-Klug, and G. Rau, "Development of recommendations for semg sensors and sensor placement procedures", Roessingh Research and Development, Enschede, Netherlands, Tech. Rep., 1999, SENIAM Project.

[8] R. A. Ekstrom, G. L. Soderberg, and R. A. Donatelli, "Normalization procedures using maximum voluntary isometric contractions for the serratus anterior and trapezius muscles during surface EMG analysis", *Journal of Electromyography and Kinesiology*, vol. 15, no. 4, pp. 418–428, Dec. 2004. DOI: 10.1016/j.jelekin.2004.09.006. [Online]. Available: https://doi.org/10.1016/j.jelekin.2004.09.006.

[9] DemšarJanez, "Statistical Comparisons of Classifiers over Multiple Data Sets", *Journal of Machine Learning Research*, Dec. 2006. DOI: 10.5555/1248547.1248548. [Online]. Available: https://dl.acm.org/citation.cfm?id=1248548.

[10] M. B. I. Reaz, M. S. Hussain, and F. Mohd-Yasin, "Techniques of EMG signal analysis: detection, processing, classification and applications", *Biological Procedures Online*, vol. 8, no. 1, pp. 11–35, Apr. 2006. DOI: 10.1251/bpo115. [Online]. Available: https://doi.org/10.1251/bpo115.

[11] G. Welch and G. Bishop, "An introduction to the kalman filter", *Proc. Siggraph Course*, vol. 8, Jan. 2006.

[12] S. Cohen, G. Dror, and E. Ruppin, "Feature Selection via Coalitional Game Theory", *Neural Computation*, vol. 19, no. 7, pp. 1939–1961, May 2007. DOI: 10.1162/neco.2007.19.7.1939. [Online]. Available: https://doi.org/10.1162/neco.2007.19.7.1939.

[13] D. A. Winter, *Biomechanics and Motor Control of Human Movement*. John Wiley Sons, Oct. 2009.

[14] A. H. Oskouei, M. G. Paulin, and A. B. Carman, "Intra-session and inter-day reliability of forearm surface EMG during varying hand grip forces", *Journal of Electromyography and Kinesiology*, vol. 23, no. 1, pp. 216–222, Sep. 2012. DOI: 10.1016/j.jelekin.2012.08.011. [Online]. Available: https://doi.org/10.1016/j.jelekin.2012.08.011.

[15] K. Cho *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation", in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179.

[16] M. P. De Looze, T. Bosch, F. Krause, K. S. Stadler, and L. W. O'Sullivan, "Exoskeletons for industrial application and their potential effects on physical work load", *Ergonomics*, vol. 59, no. 5, pp. 671–681, Oct. 2015. DOI: 10.1080/00140139.2015.1081988. [Online]. Available: https://doi.org/10.1080/00140139.2015.1081988.

[17] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization", *arXiv (Cornell University)*, Nov. 2017. DOI: 10.48550/arxiv.1711.05101. [Online]. Available: http://arxiv.org/abs/1711.05101.

[18] S. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions", *arXiv (Cornell University)*, May 2017. DOI: 10.48550/arxiv.1705.07874. [Online]. Available: http://arxiv.org/abs/1705.07874.

[19] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences", *arXiv (Cornell University)*, Apr. 2017. DOI: 10.48550/arxiv.1704.02685. [Online]. Available: http://arxiv.org/abs/1704.02685.

[20] A. Vaswani *et al.*, "Attention is all you need", Tech. Rep., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[21] L. Lai, N. Suda, and V. Chandra, *Cmsis-nn: Efficient neural network kernels for arm cortex-m cpus*, 2018. arXiv: 1801.06601 [cs.NE]. [Online]. Available: https://arxiv.org/abs/1801.06601.

[22] M. C. Lenert and C. G. Walsh, "Balancing Performance and Interpretability: Selecting Features with Bootstrapped Ridge Regression.", *PubMed*, vol. 2018, pp. 1377–1386, Jan. 2018. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/30815182.

[23] S. Jain and B. C. Wallace, "Attention is not Explanation", *arXiv (Cornell University)*, Feb. 2019. DOI: 10.48550/arxiv.1902.10186. [Online]. Available: http://arxiv.org/abs/1902.10186.

[24] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates", in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, 2019. DOI: 10.1117/12.2520589.

[25] S. Wiegreffe and Y. Pinter, "Attention is not not Explanation", *Association for Computational Linguistics*, pp. 11–20, Jan. 2019. DOI: 10.18653/v1/d19-1002. [Online]. Available: https://doi.org/10.18653/v1/d19-1002.

[26] E. Aksan, M. Kaufmann, P. Cao, and O. Hilliges, "A Spatio-temporal Transformer for 3D Human Motion Prediction", *arXiv (Cornell University)*, Apr. 2020. DOI: 10.48550/arxiv.2004.08692. [Online]. Available: http://arxiv.org/abs/2004.08692.

[27] Y. Hao, B. Wang, and R. Zheng, "Invariant Feature Learning for Sensor-based Human Activity Recognition", *arXiv (Cornell University)*, Dec. 2020. DOI: 10.48550/arxiv.2012.07963. [Online]. Available: http://arxiv.org/abs/2012.07963.

[28] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A Survey of Quantization Methods for Efficient Neural Network Inference", *arXiv (Cornell University)*, Mar. 2021. DOI: 10.48550/arxiv.2103.13630. [Online]. Available: http://arxiv.org/abs/2103.13630.

[29] B. Schabron, J. Desai, and Y. Yihun, "Wheelchair-mounted upper limb robotic exoskeleton with adaptive controller for activities of daily living", *Sensors*, vol. 21, no. 17, 2021. DOI: 10.3390/s21175738. [Online]. Available: https://www.mdpi.com/1424-8220/21/17/5738.

[30] W. Cho, V. R. Barradas, N. Schweighofer, and Y. Koike, "Design of an Isometric End-Point Force Control Task for Electromyography Normalization and Muscle Synergy Extraction From the Upper Limb Without Maximum Voluntary Contraction", *Frontiers in Human Neuroscience*, vol. 16, p. 805 452, May 2022. DOI: 10.3389/fnhum.2022.805452. [Online]. Available: https://doi.org/10.3389/fnhum.2022.805452.

[31] S. Deldari, H. Xue, A. Saeed, D. V. Smith, and F. D. Salim, "COCOA", *Proceedings of the ACM on Interactive Mobile Wearable and Ubiquitous Technologies*, vol. 6, no. 3, pp. 1–28, Sep. 2022. DOI: 10.1145/3550316. [Online]. Available: https://doi.org/10.1145/3550316.

[32] D. Leserri, N. Grimmelsmann, M. Mechtenberg, H. G. Meyer, and A. Schneider, "Evaluation of semg signal features and segmentation parameters for limb movement prediction using a feedforward neural network", *Mathematics*, vol. 10, no. 6, 2022. DOI: 10.3390/math10060932. [Online]. Available: https://www.mdpi.com/2227-7390/10/6/932.

[33] C. R. Carvalho, J. M. Fernández, A. J. del-Ama, F. Oliveira-Barroso, and J. C. Moreno, "Review of electromyography onset detection methods for real-time control of robotic exoskeletons", *Journal of NeuroEngineering and Rehabilitation*, vol. 20, no. 1, 2023. DOI: 10.1186/s12984-023-01268-8. [Online]. Available: https://jneuroengrehab.biomedcentral.com/articles/10.1186/s12984-023-01268-8.

[34] R. Gao, S. Yang, M. Yuan, X. Song, P. N. Suganthan, and W. T. Ang, "Online ensemble deep random vector functional link for the assistive robots", in *2023 International Joint Conference on Neural Networks (IJCNN)*, 2023, pp. 1–8. DOI: 10.1109/IJCNN54540.2023.10191330. [Online]. Available: https://ieeexplore.ieee.org/document/10191330.

[35] H. Li, S. Shuxiang, D. Bu, H. Wang, and M. Kawanishi, "Subject-independent estimation of continuous movements using cnn-lstm for a home-based upper limb rehabilitation system", *IEEE Robotics and Automation Letters*, vol. 8, no. 10, pp. 6403–6410, 2023. DOI: 10.1109/LRA.2023.3303701. [Online]. Available: https://ieeexplore.ieee.org/document/10214163.

[36] S. Li, L. Zhang, Q. Meng, and H. Yu, "A real-time control method for upper limb exoskeleton based on active torque prediction model", *Bioengineering*, vol. 10, no. 12, 2023. DOI: 10.3390/bioengineering10121441. [Online]. Available: https://www.mdpi.com/2306-5354/10/12/1441.

[37] M. Pesenti, G. Invernizzi, J. Mazzella, M. Bocciolone, A. Pedrocchi, and M. Gandolla, "IMU-based human activity recognition and payload classification for low-back exoskeletons", *Scientific Reports*, vol. 13, no. 1, p. 1184, Jan. 2023. DOI: 10.1038/s41598-023-28195-x. [Online]. Available: https://doi.org/10.1038/s41598-023-28195-x.

[38] H. Rehab, *SaeboMAS - Hankamp Rehab*, Nov. 2023. [Online]. Available: https://hankamprehab.nl/saebomas/.

[39] T. S. Sangeetha, S. Soman, K. S. Sivanandan, A. P. Parameswaran, and T. Baiju, "Intelligent control of exoskeletons for human limbs using knowledge-based fuzzy inference system", in *2023 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, 2023, pp. 173–178. DOI: 10.1109/DISCOVER58830.2023.10316725. [Online]. Available: https://ieeexplore.ieee.org/document/10316725.

[40] D. H. P. C. C. (DHPC), *DelftBlue Supercomputer (Phase 2)*, https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase2, 2024.

[41] D. F. Brown and S. Xie, "Effectiveness of intelligent control strategies in robot-assisted rehabilitation: A systematic review", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 32, pp. 1828–1840, 2024. DOI: 10.1109/TNSRE.2024.3396065. [Online]. Available: https://ieeexplore.ieee.org/document/10517747.

[42] J. Charafeddine, T. Houda, S. Venkateswaran, and Y. Y. Dhaher, "Neuro-motor index for upper limb exoskeleton control: A machine learning approach", in *2024 International Conference on Computer and Applications (ICCA)*, 2024, pp. 1–6. DOI: 10.1109/ICCA62237.2024.10928075. [Online]. Available: https://ieeexplore.ieee.org/document/10928075.

[43] H. Chen *et al.*, "iP3T: an interpretable multimodal time-series model for enhanced gait phase prediction in wearable exoskeletons", *Frontiers in Neuroscience*, vol. 18, Sep. 2024. DOI: 10.3389/fnins.2024.1457623. [Online]. Available: https://doi.org/10.3389/fnins.2024.1457623.

[44] J. Dai, G. Lu, Z. Qin, X. Guo, X. Liu, and Q. Yin, "Research on gesture recognition method based on pso optimized lstm for surface muscle signals", in *2024 6th Asia Symposium on Image Processing (ASIP)*, 2024, pp. 82–87. DOI: 10.1109/ASIP63198.2024.00022. [Online]. Available: https://ieeexplore.ieee.org/document/10744764.

[45] R. Hamavar and B. M. Asl, "Feature selection based on game theory optimization to achieve desired performance metrics in seizure onset detection", *Biomedical Signal Processing and Control*, vol. 100, p. 107 008, Oct. 2024. DOI: 10.1016/j.bspc.2024.107008. [Online]. Available: https://doi.org/10.1016/j.bspc.2024.107008.

[46] N. Li *et al.*, "Multi-sensor fusion-based mirror adaptive assist-as-needed control strategy of a soft exoskeleton for upper limb rehabilitation", *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 1, pp. 475–487, 2024. DOI: 10.1109/TASE.2022.3225727. [Online]. Available: https://ieeexplore.ieee.org/document/9976470.

[47] S. Narula, R. S. Pol, and R. V. Patil, "An empirical review of supervised and reinforcement learning algorithms for personalized exoskeleton robot training systems in neurorehabilitation", in *2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS)*, 2024, pp. 1434–1441. DOI: 10.1109/ICACRS62842.2024.10841642. [Online]. Available: https://ieeexplore.ieee.org/document/10841642.

[48] D. M. G. Preethichandra *et al.*, "Passive and Active Exoskeleton Solutions: Sensors, Actuators, Applications, and Recent Trends", *Sensors*, vol. 24, no. 21, p. 7095, Nov. 2024. DOI: 10.3390/s24217095. [Online]. Available: https://doi.org/10.3390/s24217095.

[49] P. Sedighi, X. Li, and M. Tavakoli, "Emg-based intention detection using deep learning for shared control in upper-limb assistive exoskeletons", *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 41–48, 2024. DOI: 10.1109/LRA.2023.3330678. [Online]. Available: https://ieeexplore.ieee.org/document/10310091.

[50] C. Shen, Z. Pei, J. Zhang, Z. Li, Y. Zhang, and W. Chen, "An elbow bilateral rehabilitation system based on surface electromyogram: Design and validation", in *2024 19th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, 2024. DOI: 10.1109/ICIEA61579.2024.10665138. [Online]. Available: https://ieeexplore.ieee.org/document/10665138.

[51] X. Wang, D. Ao, and L. Li, "Robust myoelectric pattern recognition methods for reducing users' calibration burden: challenges and future", *Frontiers in Bioengineering and Biotechnology*, vol. 12, p. 1 329 209, Jan. 2024. DOI: 10.3389/fbioe.2024.1329209. [Online]. Available: https://doi.org/10.3389/fbioe.2024.1329209.

[52] X. Wu, J. Liang, Y. Yu, G. Li, G. G. Yen, and H. Yu, "Embodied perception, interaction, and cognition for wearable robotics: A survey", *IEEE Transactions on Cognitive and Developmental Systems*, pp. 1–18, 2024. DOI: 10.1109/TCDS.2024.3463194. [Online]. Available: https://ieeexplore.ieee.org/document/10682963.

[53] E. Bionics. "Eksoevo exoskeleton". Accessed: 2026-28-04. (2025), [Online]. Available: https://eksobionics.com/eksohealth/eksogt.

[54] H. O. Farag, M. M. Gaber, M. I. Awad, and N. E. Elhady, "Myoelectric Prosthetic Hands: A Review of Muscle Synergy, Machine Learning and Edge Computing", *ACM Computing Surveys*, vol. 57, no. 12, pp. 1–33, May 2025. DOI: 10.1145/3742471. [Online]. Available: https://doi.org/10.1145/3742471.

[55] J. G. V. Feria, J. E. H. Gonzalez, and R. R. Serrezuela, "Methodology for gesture recognition and hand exoskeleton control using surface electromyography (semg)", in *2025 IEEE VIII Congreso Internacional en Inteligencia Ambiental, Ingenieria de Software y Salud Electronica y Movil (AmITIC)*, 2025, pp. 1–7. DOI: 10.1109/AmITIC68284.2025.11214604. [Online]. Available: https://ieeexplore.ieee.org/document/11214604.

[56] L. Hu, D. H. Zhai, D. Yu, and Y. Xia, "A hybrid framework based on bio-signal and built-in force sensor for human–robot active co-carrying", *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 3553–3566, 2025. DOI: 10.1109/TASE.2024.3395921. [Online]. Available: https://ieeexplore.ieee.org/document/10521602.

[57] M. Karimi and M. Ahmadi, "Ilead: An emg-based adaptive shared control framework for exoskeleton assistance via deep reinforcement learning", *IEEE Transactions on Artificial Intelligence*, vol. 6, no. 10, pp. 2732–2743, 2025. DOI: 10.1109/TAI.2025.3556983. [Online]. Available: https://ieeexplore.ieee.org/document/10947342.

[58] R. Kumar, A. Gupta, S. P. Prakash Muthukrishnan, L. Kumar, and S. Roy, "Piman: A physics-informed motion prediction network using semg signal features for human movement parameters", *Neurocomputing*, vol. 651, 2025. DOI: 10.1016/j.neucom.2025.130884. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231225015565.

[59] C. Liu, K. Zhao, W. Si, J. Li, and C. Yang, "Neuroadaptive admittance control for human–robot interaction with human motion intention estimation and output error constraint", *IEEE Transactions on Cybernetics*, vol. 55, no. 6, pp. 3005–3016, 2025. DOI: 10.1109/TCYB.2025.3555104. [Online]. Available: https://ieeexplore.ieee.org/document/10955727.

[60] X. Niu and A. Furui, "Towards Cross-Subject EMG Pattern Recognition via Dual-Branch Adversarial Feature Disentanglement", *arXiv (Cornell University)*, Jun. 2025. DOI: 10.48550/arxiv.2506.08555. [Online]. Available: https://doi.org/10.48550/arxiv.2506.08555.

[61] J. H. Sul *et al.*, "Electromyography signal acquisition, filtering, and data analysis for exoskeleton development", *Sensors*, vol. 25, no. 13, 2025. DOI: 10.3390/s25134004. [Online]. Available: https://www.mdpi.com/1424-8220/25/13/4004.

[62] A. Toro-Ossaba, J. C. Tejada, and D. Sanin-Villa, "Myoelectric control in rehabilitative and assistive soft exoskeletons: A comprehensive review of trends, challenges, and integration with soft robotic devices", *Biomimetics*, vol. 10, no. 4, 2025. DOI: 10.3390/biomimetics10040214. [Online]. Available: https://www.mdpi.com/2313-7673/10/4/214.

[63] Y. Yang, H. H. Teo, and Y. J. King, "Toward intelligent human–robot interaction for upper limb rehabilitation: A review of emerging modalities and strategies", *IEEE Access*, vol. 13, pp. 185 513–185 532, 2025. DOI: 10.1109/ACCESS.2025.3625220. [Online]. Available: https://ieeexplore.ieee.org/document/11216408.

[64] D. Zhao *et al.*, "Upper limb human-exoskeleton system motion state classification based on semg: Application of cnn-bilstm-attention model", *Scientific Reports*, vol. 15, no. 1, 2025. DOI: 10.1038/s41598-025-02864-5. [Online]. Available: https://www.nature.com/articles/s41598-025-02864-5.

[65] B. Wingen, *EMG–IMU Sensor-Fusion Load Estimation: Code and Acquisition Firmware*, https://github.com/baswingen/Thesis, Accessed: 23-06-2000, 2026.

APPENDIX

A. Sensory acquisition system specifics

For reproducibility, this section provides all the details needed to rebuild the sensor hub used to produce the results in this paper. Figure 11 displays a schematic of the sensory acquisition system and its component connections.

The ARM Cortex-M4 STM32F401 microcontroller was selected for its high performance and its I/O flexibility [5]. It generates the pseudo-random binary sequence (PRBS) required for synchronization. It is configured at 921600 Baud on USB Serial and runs C++ firmware. To assure high-speed transmission, the data is packed using a 67-byte binary protocol. The custom firmware for this application is publicly available on GitHub [65].

The dual Adafruit BNO085 IMUs were selected for their high-accuracy measurements and reliable onboard sensor fusion algorithm. The chosen transmission mode for the IMU setup was UART-RVC, as I2C was not supported for a dual BNO085 setup, and SPI was deemed less practical due to its wiring. For transmission, 19-byte binary packets were used, containing timestamps, on-board fused three-dimensional Euler angles, and three-dimensional linear acceleration.

The 3x4 button matrix was manually constructed using tact buttons and diodes (1N4148). All buttons are $d_{\text{button}} \approx 15$ cm apart. The microcontroller reads all rows and columns and can detect multiple simultaneous presses using the diode matrix setup. To eliminate switch bounce, a debounce algorithm is applied to each of the 12 buttons individually. This ensures that the button matrix outputs consistent values. Rising edges and falling edges are recorded separately to ensure accurate matrix state switches. A key mask encodes the entire button matrix as a 16-bit integer.

The TMSi Porti7 EMG device was configured using the accompanying Python SDK from TMSi's GitHub repository. Eight channels were read out, corresponding to the muscle electrode channels, along with a TRIG channel that transmits the PRBS.

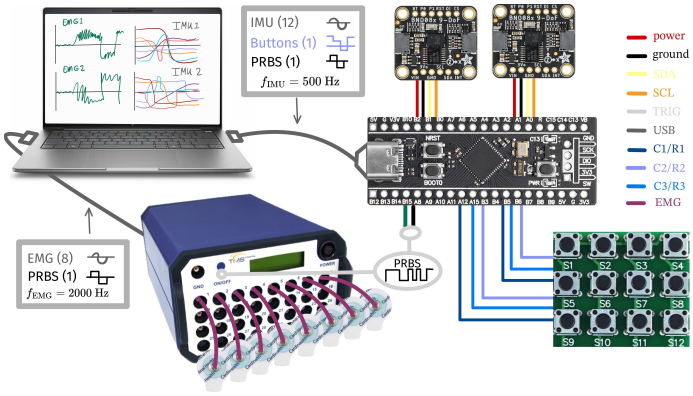


Figure 11. This figure displays the hardware sensory setup in an overview, together with the cable connections between all components.

B. PRBS-Kalman synchronization specifics

This algorithm synchronizes two independent acquisition systems running on different internal clocks and at different sampling frequencies: a TMSi Porti7 EMG system ($f_{\text{EMG}} = 2000$ Hz) and an STM32F401 microcontroller ($f_{\text{IMU}} = 500$ Hz). The STM32 generates a PRBS-15 sequence and wires it directly to the Porti7 TRIG input. This creates a shared reference signal that is observable in both systems, but at different times due to transmission delay. In this section, the implementation details of the real-time PRBS-Kalman algorithm will be discussed. The system code is publicly available on GitHub [65].

[5] STM32F401 - Arm Cortex-M4, URL: https://www.st.com/en/microcontrollers-microprocessors/stm32f401.html

Stage 1: Cross-correlation

The first stage of the algorithm uses cross-correlation to determine the offset between the two sensory systems at a given time point [3]. A PRBS-15 was selected at a chip rate of $f_c = 100$ Hz, outputting a new bit every 10 ms. This chip rate was experimentally determined and proved to be in the sweet spot. A higher chip rate would lead to signal degradation due to the EMG system transmission.

To find the estimated delay and offset, cross-correlation between the two acquired chip streams was used. The binary chips were made bipolar to shift their mean to zero:

$$\tilde{c} = 2c - 1 \in \{-1, +1\}$$

The normalized cross-correlation R at lag ℓ in window W can be calculated as follows:

$$\hat{\ell} = \arg\max_{\ell \in \mathcal{V}} |R(\ell)|, \quad R(\ell) = \frac{1}{O(\ell)} \sum_{n=0}^{W-1} \tilde{c}_n^{\text{EMG}} \cdot \tilde{c}_{n-\ell}^{\text{STM}}$$

Dividing by $O(\ell)$ normalizes all cross-correlations to the same range, even though their windows might have full or little overlap, essentially compensating for bias in the amplitude of the cross-correlation. The following formulation defines this normalization factor based on the lengths of the chip arrays W passed into the crosscorrelator:

$$O(\ell) = W - |\ell|, \quad \mathcal{V} = \{\ell : O(\ell) \geq \rho \cdot W\}, \quad \rho = 0.9$$

The valid search range for cross-correlation is defined as $\mathcal{V}$, which requires 90% overlap between signals ρ. This serves as a gate, in which cross-correlation is calculated only at statistically significant cross-correlation lags.

The window W determines the length of each chip subsequence passed to the correlation. Given a window duration of T_W, that was tuned to $T_W = 10$ s in this application:

$$W = T_W \cdot f_c$$

This can be tuned to set the maximum detectable lag, at the cost of computational overhead. Lowering the chip rate will reduce synchronization resolution but will allow the system to tolerate larger maximum detectable lag ranges without increasing computational cost.

In practice, the cross-correlation is computationally expensive and is run once per Kalman update, at $f_\Delta = 0.5$ Hz. Rather than a single lag, each run slides the correlation window across the buffer, yielding a series of N_W sub-window lag estimates indexed by j:

$$\left\{\left(\hat{\ell}_j,\, R_j(\hat{\ell}_j),\, t_j\right)\right\}_{j=1}^{N_W}$$

Before entering the next stage, the lag series is aggregated into a single scalar measurement. A quality gate first discards windows with insufficient correlation peak strength:

$$\mathcal{J} = \{j : |R_j(\hat{\ell}_j)| \geq 0.10\}$$

The surviving windows are aggregated via median, which is robust to outlier windows caused by noise bursts or signal dropouts:

$$z_k = \text{median}\{\hat{\ell}_j : j \in \mathcal{J}\} \cdot \frac{10^3}{f_c}\ \text{ms}, \quad T_c = \frac{1}{f_c} = 10\ \text{ms}$$

The corresponding confidence γ_k is calculated through the cross-correlation peak strength at the given lag $|R_k(\hat{\ell}_k)|$, as well as the consistency of that given lag across the window, which can be calculated using the standard deviation σ_ℓ. Thus, the corresponding confidence was calculated following this formula:

$$\gamma_k = |R_{j^\star}(\hat{\ell}_{j^\star})| \frac{1}{1 + \sigma_\ell}, \quad \sigma_\ell = \text{std}\{\hat{\ell}_j : j \in \mathcal{J}\}$$

Each update k thus yields a single offset measurement z_k and its confidence $\gamma_k \in [0, 1]$, aggregated over the sub-windows of that update. This pair is passed to the next stage, where the Kalman filter consumes one (z_k, γ_k) per step at the same rate f_Δ.

Stage 2: Kalman Filter

The measurement z_k is a noisy measurement accompanied by a correlation confidence γ_k. To address this uncertainty, track offset and drift in real time and align the system smoothly, the following two-state Kalman filter is implemented [2, 11]. The Kalman filter is coupled to the cross-correlation frequency f_Δ, and updates at $f_\Delta = 0.5$ Hz. The state vector of the Kalman filter is defined as follows:

$$\mathbf{x}_k = \begin{bmatrix} \tau_k \\ \dot{\tau}_k \end{bmatrix}$$

where τ_k is the clock offset in milliseconds and $\dot{\tau}_k$ is the drift rate in parts per million. At each update timestep k, the filter first predicts the new state forward in time, assuming constant drift:

$$\mathbf{x}_{k|k-1} = \mathbf{F}_k \, \mathbf{x}_{k-1|k-1}, \qquad \mathbf{F}_k = \begin{bmatrix} 1 & \Delta t \cdot 10^{-3} \\ 0 & 1 \end{bmatrix}$$

Here Δt is the update interval, derived from the Kalman update frequency. The Kalman filter utilizes a state uncertainty for its current state, in the form of a state covariance matrix $\mathbf{P}_k$, which is dynamically updated at every step and initialized at $\mathbf{P}_0 = \mathrm{diag}\{100, 1000\}$:

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \, \mathbf{P}_{k-1|k-1} \, \mathbf{F}_k^\top + \mathbf{Q}$$

Here $\mathbf{Q}$ is the process noise covariance matrix $\mathbf{Q} = \mathrm{diag}\{0.01, 10\}$, which is tuned to represent how much the true state is expected to change unpredictably between updates, independent from the constant drift rate introduced by $\mathbf{F}_k$.
The measurement z_k is then compared to the predicted offset via the innovation y_k. Since only the clock offset is directly observed, $\mathbf{H} = [1, \, 0]$:

$$y_k = z_k - \mathbf{H}\mathbf{x}_{k|k-1}$$

How much the filter trusts z_k over its own prediction is controlled by the measurement noise $\mathcal{R}_k$, which is scaled inversely by the correlation confidence γ_k:

$$\mathcal{R}_k = \frac{\mathcal{R}_0}{\max(0.01, \, \gamma_k)}$$

Here, the baseline measurement noise $\mathcal{R}_0$ is tuned to represent the expected variance of z_k, in this application $\mathcal{R}_0 = 1$ ms^2. A weak or inconsistent correlation inflates $\mathcal{R}_k$, causing the Kalman gain $\mathbf{K}_k$ to down-weight the measurement and rely more on the prediction, and vice versa. Ultimately, the Kalman gain $\mathbf{K}_k$ is calculated as a measure of the reliability of the measurement, and used to update the state based on the innovation:

$$\mathbf{K}_k = \frac{\mathbf{P}_{k|k-1}\mathbf{H}^\top}{\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^\top + R_k}, \qquad \mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}_k \, y_k$$

The state covariance matrix $\mathbf{P}_{k|k}$ at the current timestep is updated based on the Kalman gain, as the uncertainty should shrink with a reliable new measurement.

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}) \, \mathbf{P}_{k|k-1}$$

The smoothed offset $\hat{\tau}_{k|k} = \mathbf{x}_{k|k}[0]$ is extracted and applied to align the incoming EMG timestamps to the STM32 clock in real time:

$$t_{\mathrm{EMG}}^{\mathrm{aligned}} = t_{\mathrm{EMG}} - \hat{\tau}_{k|k}$$

Given the update frequency $f_\Delta = 0.5$ Hz, the signals are aligned every $\Delta t_K = 2$ s, and in the implemented configuration together with the used hardware, this resulted in an alignment with an accuracy of a couple of milliseconds.

C. Sensor Group Shapley Values

Each sensor group is treated as a player and a set of groups $S \subseteq \mathcal{G}$ as a coalition, where $\mathcal{G}$ is the set of the five groups. A characteristic function $v(S)$ assigns each coalition a value: the reduction in class-macro RMSE that the model trained on the groups in S achieves relative to an uninformed baseline,

$$v(S) = R_\emptyset - R(S),$$

where $R(S)$ is the participant-class-macro error of the model trained on the groups in S, and $R_\emptyset$ is the same metric evaluated for an uninformed baseline that always predicts the global mean load. The procedure is applied independently to both reported metrics, so R denotes whichever of the participant-class-macro MAE or RMSE is being attributed. By construction, the empty model reduces to this mean predictor, giving $v(\emptyset) = 0$, and a more useful coalition yields a larger $v(S)$.

The marginal importance of adding group g to a coalition S is the resulting gain in value:

$$\Delta_g(S) = v(S \cup \{g\}) - v(S).$$

The Shapley value ϕ_g of group g [1] is the average of this marginal importance over all orders in which the groups could be added, crediting g fairly regardless of when it joins:

$$\phi_g = \frac{1}{|\mathcal{G}|!} \sum_{\pi \in \Pi} \Delta_g\big(S_g(\pi)\big),$$

where Π is the set of all $|\mathcal{G}|!$ orderings of the groups and $S_g(\pi)$ the set of groups preceding g in ordering π. Because only the presence or absence of a group affects v, and not its position in the order, this reduces to an equivalent weighted sum over subsets [1]:

$$\phi_g = \sum_{S \subseteq \mathcal{G} \setminus \{g\}} \frac{|S|! \, \big(|\mathcal{G}| - |S| - 1\big)!}{|\mathcal{G}|!} \, \Delta_g(S),$$

where the combinatorial weight is the fraction of orderings in which exactly the groups in S precede g. By the efficiency property of the Shapley value, the contributions sum to the value of the full set, $\sum_{g \in \mathcal{G}} \phi_g = v(\mathcal{G}) = R_\emptyset - R(\mathcal{G})$, so the ϕ_g partition the total achievable RMSE reduction among the five groups. This formulation follows the use of Shapley values for performance-based feature selection [12], analogous to the EEG feature-selection procedure of Hamavar et al. [45].

With only five groups, all $2^5 - 1 = 31$ non-empty coalitions are trained, so the Shapley values are computed exactly rather than approximated by sampling. The resulting group importances are compared against the per-channel DeepSHAP, attention pooling, and permutation attributions (Section 3.3) to assess agreement between the retraining ablation and the cheaper test-set methods.

D. Hyperparameter Tuning

In this section, the hyperparameter sweeps that led to the configurations used for the two final models in the results are discussed: the cross-participant generalized model and the per-participant specialized model. Each model received its own feature and hyperparameter search. To eliminate the need for LOPO cross-validation at every sweep iteration, a fixed validation group of four participants (P01, P02, P06, and P17) was held out to serve as both the validation and test

sets, yielding slightly inflated results that are nonetheless comparable across iterations.

Generalized Model Sweep

The generalized configuration was selected through random search over a broad range of parameter configurations, split into four phases (150-iterations each) due to computational constraints and performed in the following order:

1. The **model architecture sweep** was performed first to determine a baseline model hyperparameter set that performs well, serving as a representative model that would yield reliable results across the upcoming sweeps.
2. The **feature selection sweep** was used to find the most contributing features and filter out features that are noisy or subject-volatile.
3. The **generalization sweep** was used to find optimal data augmentation types and parameters, and the optimal learning rate scheduler together with regularization parameters, dropout, and weight decay.
4. A second **model architecture sweep** identified the final best-performing model hyperparameters using the configurations carried over from the previous sweeps.

The best feature configuration was found by sweeping over 20 EMG and 13 IMU features. For each sweep iteration, each feature had a 50% chance of being toggled on or off. The model was trained on a fixed cross-participant split with a test set of 4 participants, and the search records the R^2 score achieved on the unseen participants for every iteration. After all iterations are completed, a Ridge Regression model mathematically isolates the impact of each feature [22]:

$$y_i = \beta_0 + \sum_{k=1}^{33} \beta_k X_{ik} + \epsilon_i$$

- X_{ik} (binary predictors): indicator matrix of shape $(150, 33)$, with $X_{ik} = 1$ if feature k was active in iteration i and 0 otherwise.
- y_i (target): the validation R^2 of iteration i.
- β_k (Ridge coefficient): the estimated average change in validation R^2 associated with activating feature k, i.e. its marginal contribution to generalization. A positive β_k includes the feature; a negative one excludes it.

Depending on whether the Ridge Coefficient for a given feature was negative or positive, the feature was excluded or included. This methodology reduced the number of selected features from 33 (20 EMG, 13 IMU) to 21 (11 EMG, 10 IMU).

Specialized Model Sweep

For the participant-specialized model, the feature set and hyperparameters were determined in a single consolidated search using the same held-out validation group. The same Ridge-coefficient selection produced a 17-feature set (11 EMG, 6 IMU). Table 4 lists the features selected for each model, and Table 3 reports the best hyperparameter configuration for both.

Table 3. Top-performing hyperparameters resulting from the sweeps, for the generalized and participant-specialized models.

Parameter	Gen.	Spec.	Description
Regularization and Augmentation Parameters			
Weight decay	0.005	0.001	Punishes reliance on high-value weights in the model.
Dropout	0.25	0.20	Randomly removes weights during training to reduce reliance on specific weights.
Learning rate	$3 \cdot 10^{-4}$	$2 \cdot 10^{-4}$	Controls the step size taken toward minimizing the training loss.
Augmentation probability	0.5	0.5	Controls how often augmentation is applied.
Noise standard deviation	0.05	0.05	Determines the standard deviation of the applied Gaussian noise.
Temporal stretch	{0.75, 1.25}	{0.90, 1.10}	Randomly stretches the signal over time, reducing reliance on movement speed.
Channel dropout prob.	0.25	0.10	Removes all features from a channel, reducing reliance on specific channels.
Model Architecture Parameters			
Embedding dimension (D)	128	96	Defines the transform from the feature set to the number of embedded features.
Transformer layers ($\mathcal{L}$)	4	4	Number of attention iterations before attention pooling and regression.
Spatial heads (H_{sp})	8	4	Defines the depth of the spatial attention mechanism.
Temporal heads (H_{tp})	4	2	Defines the depth of the temporal attention mechanism.
Feedforward size (d_{ff})	512	1024	Internal hidden layer size of the position-wise feedforward network.

E. Dataset Cleansing and Balancing

To guarantee the integrity of the dataset and prevent corrupted signals from degrading the performance, dataset cleansing was performed. To ensure that the model's performance is not artificially improved by streamlining the dataset to include only perfect samples, a cautious strategy was employed. Four segment checks were performed, with automatic blacklisting if any failed. Using this methodology, a total of 545 segments were excluded, 2.82% of the dataset. The following four checks were performed on the individual channel data:

1. **Dead channel detection:** When the variance of a channel is near zero, it can be marked as dead. The following mathematical formulation is used:

$$\text{Var}(x_c) = \frac{1}{N}\sum_{t=1}^{N}(x_{c,t} - \mu_c)^2 < 10^{-12}$$

This check blacklisted 136 segments, 0.704% of total dataset.

2. **Flatline detection:** The connections of the components can be unreliable, which can lead to signal flatlines. The algorithm calculates the flatline ratio (F) for each channel independently, which represents the proportion of adjacent time samples that are perfectly identical (a difference of zero):

$$\text{F} = \frac{1}{N-1}\sum_{t=1}^{N-1}\delta_{x_{t+1},x_t} > 0.05, \text{ with: } \delta_{x_{t+1},x_t} = \begin{cases} 1 & \text{if } x_{t+1} = x_t \\ 0 & \text{if } x_{t+1} \neq x_t \end{cases}$$

This check blacklisted 24 segments, 0.124% of total dataset.

3. **Extreme outlier detection:** Samples that diverge significantly from physical reality are treated as artifacts. The mean μ_c and standard deviation σ_c for a specific channel are calculated globally for the entire dataset. Any sample falling outside a $\pm 6\sigma_c$ threshold is classified as an outlier. The outlier ratio (O) is calculated as follows and segments are flagged according to the following threshold:

$$\text{O} = \frac{1}{N}\sum_{t=1}^{N}\delta_t > 0.02, \text{ with: } \delta_t = \begin{cases} 1 & \text{if } |x_t - \mu_c| > 6\sigma_c \\ 0 & \text{if } |x_t - \mu_c| \leq 6\sigma_c \end{cases}$$

This check blacklisted 14 segments, 0.072% of the dataset.

4. **Temporal duration bound:** This ensures that the movement segments are physically meaningful. Short movements can indicate accidental button matrix triggers, and long segments can indicate rest or pauses. Outliers in segment length can confuse

Table 4. Mathematical derivations of the sweep-selected features, grouped by modality, and the generalized and participant-specialized models each feature was selected for.

Feature	Derivation	Model
Electromyography (EMG) Features		
Log Detector	$x_{\text{LogDet}} = \exp\left(\frac{1}{N}\sum_{i=1}^{N} \ln\lvert x_i\rvert\right)$	Both
Waveform Length	$x_{\text{WL}} = \frac{1}{N}\sum_{i=1}^{N-1} \lvert x_{i+1} - x_i\rvert$	Both
Root Mean Square	$x_{\text{RMS}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$	Both
Bandwidth	$f_{\text{high}} - f_{\text{low}}$	Both
Total Spectral Power	$P_{\text{total}} = \sum_j P(f_j)$	Both
Mean Frequency	$f_{\text{MNF}} = \frac{\sum_j f_j P(f_j)}{\sum_j P(f_j)}$	Both
Hjorth Mobility	$\sqrt{\text{Var}(x')/\text{Var}(x)}$	Gen.
Zero Crossings	$x_{\text{ZC}} = \frac{1}{N}\sum_{i=1}^{N-1} \mathbb{I}(x_i x_{i+1} < 0)$	Gen.
Myopulse Pct. Rate	$x_{\text{Myo}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(\lvert x_i\rvert > \theta)$	Gen.
Slope Sign Changes	$x_{\text{SSC}} = \frac{1}{N}\sum_{i=1}^{N-2} \mathbb{I}(x_i' x_{i+1}' < 0)$	Gen.
Willison Amplitude	$x_{\text{WAMP}} = \frac{1}{N}\sum_{i=1}^{N-1} \mathbb{I}(\lvert x_{i+1} - x_i\rvert > \theta)$	Gen.
Hjorth Complexity	$\text{Mobility}(x')/\text{Mobility}(x)$	Spec.
Integrated EMG	$x_{\text{IEMG}} = \sum_{i=1}^{N} \lvert x_i\rvert$	Spec.
Mean Absolute Value	$x_{\text{MAV}} = \frac{1}{N}\sum_{i=1}^{N} \lvert x_i\rvert$	Spec.
Variance	$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$	Spec.
Skewness	$\frac{\frac{1}{N}\sum_{i=1}^{N}(x_i-\bar{x})^3}{\sigma^3}$	Spec.
Inertial Measurement Unit (IMU) Features		
Standard Deviation	$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2}$	Both
Variance	$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2$	Both
Mean	$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$	Both
Peak-to-Peak	$y_{\text{P2P}} = \max(y) - \min(y)$	Both
Signal Magnitude Area	$y_{\text{SMA}} = \frac{1}{N}\sum_{i=1}^{N} \lvert y_i\rvert$	Both
Total Spectral Power	$P_{\text{total}} = \sum_j P(f_j)$	Spec.
Mean Absolute Jerk	$y_{\text{Jerk}} = \frac{f_{\text{IMU}}}{N-1}\sum_{i=1}^{N-1} \lvert y_{i+1} - y_i\rvert$	Gen.
Dominant Frequency	$f_{\text{Dom}} = \arg\max_f P(f)$	Gen.
Kurtosis	$\frac{\frac{1}{N}\sum_{i=1}^{N}(y_i-\bar{y})^4}{\sigma^4} - 3$	Gen.
Skewness	$\frac{\frac{1}{N}\sum_{i=1}^{N}(y_i-\bar{y})^3}{\sigma^3}$	Gen.
Maximum	$y_{\text{Max}} = \max_i y_i$	Gen.

the model when it is accustomed to a certain range. The segments in the following range are eligible for model training:

$$0.5\text{ s} < L < 3.5\text{ s}$$

This check blacklisted 405 segments (117 too short and 288 too long), 2.10% of the dataset.

After cleansing, the dataset suffers from two compounding imbalances: participants contribute unequal numbers of segments due to differing session lengths and blacklisted segments, and the weight classes are inherently skewed because the free movement (0 kg) class is overrepresented.

To address both axes simultaneously, a joint class–participant balancing algorithm is applied to the training set (excluding the test set since testing is on unique segments only). Every unique participant, weight class combination is targeted at exactly $T = 100$ segments. Combinations that contain more than T originals are randomly downsampled. Combinations that contain fewer are filled to T by oversampling with replacement segments, forcing augmentation to each copy. With 14 training participants per LOPO fold (1 held-out for testing and 2 held-out for early stopping) and 6 weight levels, this yields a nominal training set of:
$N_{\text{train}} = N_{\text{participants}} \cdot N_{\text{weights}} \cdot T \approx 14{\times}6{\times}100 = 8,400$ segments per fold. The target of $T = 100$ was chosen conservatively: it is low enough that no combinations require large oversampling. In the participant-specialized cross-validation setting, all 17 participants contribute to each training fold, yielding $17 \times 6 \times 100 = 10{,}200$ segments. Finally, all reported performance metrics are computed as macro-averages across participants and weight classes, weighting each participant and weight class equally regardless of their segment count, ensuring a consistent evaluation criterion for both validation strategies.

F. Statistical Significance of Results Using the Friedman Test

To determine whether the performance differences between sensor configurations are statistically meaningful, a non-parametric Friedman test was conducted on the participant-level error metrics ($N = 17$ subjects) [9]. The Friedman test is a robust alternative to repeated-measures analysis of variance (RM-ANOVA) that does not assume normality of the underlying distributions or sphericity of the differences, making it highly suitable for cross-participant comparisons with individual variations. The test evaluates the null hypothesis that the ranks of the performances of all modalities across the participants are equal, implying that all modality configurations perform similarly:

$$H_0 : \text{Rank}_{\text{SF}} = \text{Rank}_{\text{EMG}} = \text{Rank}_{\text{IMU}}$$

When the null hypothesis is rejected ($p < 0.05$), post-hoc pairwise comparisons are performed using two-tailed Wilcoxon signed-rank tests.

Table 5 summarizes the mean errors, Friedman test statistics ($\chi^2(2)$), and Bonferroni-corrected post-hoc pairwise Wilcoxon comparisons for both the participant-specialized and LOPO-generalized settings. In the pairwise relations, the $<$ symbol denotes a statistically significant difference at the adjusted $p_{\text{adj}} < 0.05$ level, while the $\approx$ symbol denotes no statistically significant difference. In the participant-specialized setting, all three modalities are strictly ordered, with sensor fusion (SF) significantly outperforming EMG-only ($p_{\text{adj}} = 4.58 \times 10^{-5}$ for both MAE and RMSE), confirming that the IMU contributes complementary information in participant-specialized. Under generalization (LOPO), sensor fusion and EMG-only become statistically indistinguishable ($p_{\text{adj}} = 1.000$ for both MAE and RMSE), even though sensor fusion attains a slightly lower mean error in both metrics . This can be attributed to the larger spread between participants in cross-subject generalization and the limited sample size ($N = 17$). In all settings, IMU-only performance is significantly worse than both alternatives ($p_{\text{adj}} < 0.001$), and degrades severely under generalization.

G. Computational Methodology and Results

This appendix details the feasibility analysis summarised in Section 3.5. The reported parameter counts, memory footprints, and operation counts are exact properties of the trained models: the host-CPU and GPU timings are measured, and the Cortex-M7 latencies are analytical projections under explicitly stated assumptions. No deployment on physical microcontroller hardware was carried out, so the edge latencies should be read as conservative order-of-magnitude upper bounds rather than measured results.

Memory footprint: The sensor-fused model contains $N_{\text{params}} \approx 1.03 \times 10^6$ trainable parameters. In single-precision floating point (FP32, 4 bytes per parameter) this requires 4.12 MB of Flash; post-training 8-bit integer (INT8) quantization reduces the footprint to 1.03 MB [28], fitting within the 2 MB on-chip Flash of an advanced microcontroller such as the STM32H7.

Operation count A single forward pass requires $N_{\text{MAC}} \approx 1.15 \times 10^6$ multiply–accumulate operations. Since each MAC comprises one multiplication and one addition (1 MAC = 2 FLOPs), this is equivalent to $\approx 2.3 \times 10^6$ FLOPs. On the host NVIDIA A100 GPU, a forward pass executes in 0.62 ms.

Inference latency: Edge latency is estimated by dividing the operation count by the sustained hardware throughput:

$$T_{\text{model}} = \frac{2\,N_{\text{MAC}}}{\text{throughput}}.$$

Table 5. Friedman test and Bonferroni-corrected post-hoc Wilcoxon signed-rank comparisons of the modalities ($N = 17$ participants), computed on participant-class-macro MAE/RMSE. SF = sensor-fused, EMG = EMG-only, IMU = IMU-only.

	Modality Mean MAE/RMSE (kg)			Friedman Test		Post-hoc p_{adj}			Pairwise
Metric & Strategy	SF	EMG	IMU	$\chi^2(2)$	p	SF–EMG	SF–IMU	EMG–IMU	relations
MAE (kg)									
Participant-Specific	0.316	0.380	0.702	34.00	< 0.001	4.58×10^{-5}	4.58×10^{-5}	4.58×10^{-5}	SF < EMG < IMU
Generalized (LOPO)	0.536	0.546	0.989	20.59	< 0.001	1.000	4.58×10^{-5}	2.29×10^{-4}	SF ≈ EMG < IMU
RMSE (kg)									
Participant-Specific	0.441	0.520	0.974	34.00	< 0.001	4.58×10^{-5}	4.58×10^{-5}	4.58×10^{-5}	SF < EMG < IMU
Generalized (LOPO)	0.680	0.703	1.289	23.06	< 0.001	1.000	4.58×10^{-5}	9.16×10^{-5}	SF ≈ EMG < IMU

Table 6. Friedman test across the six architectures ($N = 17$ participants), on participant-class-macro MAE/RMSE. ST = ST-Transformer, TF = vanilla Transformer, C-L = CNN-LSTM, C-G = CNN-GRU. In the relations, {·} groups architectures that are mutually non-significant and < denotes $p_{adj} < 0.05$ (Bonferroni-corrected Wilcoxon) between all flanking members; lower error is better.

	Architecture mean						Friedman		Pairwise
Metric	ST	LSTM	GRU	TF	C-L	C-G	$\chi^2(5)$	p	relations
MAE (kg)									
Specialized	0.316	0.181	0.196	0.328	0.830	0.567	74.58	< 0.001	{LSTM,GRU} < {ST,TF} < C-G < C-L
Generalized	0.536	0.488	0.489	0.622	0.812	0.756	32.09	< 0.001	{LSTM,GRU} < {TF,C-G,C-L}; ST < C-L only
RMSE (kg)									
Specialized	0.441	0.417	0.407	0.490	0.943	0.708	68.80	< 0.001	{LSTM,GRU,ST,TF} < C-G < C-L
Generalized	0.680	0.669	0.636	0.803	0.932	0.866	24.56	< 0.001	{GRU,LSTM} < {TF,C-G,C-L}; ST n.s. vs all

On an ARM Cortex-M7 at 480 MHz, sustained FP32 throughput is conservatively taken as ≈ 90 MFLOPS, giving $T_{model} \approx 25$ ms. Under INT8 quantization, the CMSIS-NN library exploits the core's SIMD extensions (up to four 8-bit MACs per cycle), raising the effective throughput to ≈ 300 MFLOPS-equivalent and reducing inference latency to a conservative ≤ 10 ms [21].

Feature-extraction latency: Single-window feature extraction is benchmarked at $T_{feature} = 1.20$ ms on the host CPU. Unlike inference, this stage is not expressed as a FLOP count; its Cortex-M7 projection is obtained by scaling the host timing by an architectural and clock-frequency translation factor $\gamma \approx 12$, giving a conservative $T_{feature} \approx$ 15 ms (CMSIS-DSP, C++-compiled).

Real-time margin: For closed-loop control, the pipeline must complete within the prediction interval $\Delta t = 100$ ms. The available idle margin is

$$M_{CPU} = 100\% \times \left(1 - \frac{T_{feature} + T_{model}}{\Delta t}\right).$$

Using the most conservative INT8 estimates ($T_{feature} \approx 15$ ms, $T_{model} \approx 10$ ms), the worst-case margin is ≈ 75%, indicating substantial headroom for auxiliary control tasks. Because the inference latency is derived analytically from the operation count, while the feature-extraction latency is scaled from a host benchmark, the two stages rest on different projection bases; both are deliberately conservative.

H. Architecture Comparison

The ST-Transformer was compared against five established baseline models: LSTM, GRU, a vanilla Transformer, and two CNN-based models that learn end-to-end representations from raw segments (CNN-LSTM, CNN-GRU). They were trained under both validation strategies (participant-specific and generalized), with identical balancing, and using the participant-class macro metrics defined in Section 3. For each model, a non-parametric Friedman test across architectures, with Bonferroni-corrected Wilcoxon signed-rank tests over all 15 pairwise comparisons as post-hoc tests, was computed, as described in more detail in Appendix F.

For the ST-Transformer specifically, there are no significant pairwise differences compared to the other feature-based models (LSTM, GRU, TF) in three of the four metric/strategy combinations. The sole exception is participant-specialized MAE, where LSTM and GRU modestly outperform the ST-Transformer (ST–LSTM $p_{adj} = 0.0057$, ST–GRU $p_{adj} = 0.0057$). The CNN models are significantly worse in the specialized regime (all $p_{adj} < 0.001$); under LOPO, their separation is largely non-significant after correction, reflecting the high between-participant variance of cross-subject generalization. Overall, the ST-Transformer is statistically competitive with the established sequence models (Figure 12, Table 6).

A closer look at the prediction distributions (Figure 13) explains the specialized-MAE result. Because the protocol uses only six discrete loads, a model can minimize MAE by treating the task as a six-way classification and emitting the corresponding load centroid, rather than by learning a continuous load mapping. To quantify this, each prediction was labeled on-grid if it fell within 10% of the inter-load spacing of one of the six training loads, and interpolated if it fell in the central 20% band between two adjacent loads. In the participant-specialized regime, the recurrent baselines are almost entirely on-grid, LSTM 94% on-grid (1% interpolating) and GRU 91% (1%), whereas the ST-Transformer places only 65% of its predictions on-grid and leaves roughly four times as much mass between the loads (4%). This collapse onto the training grid is what produces their lower specialized MAE. The effect is regime-specific: under LOPO, where per-participant load centroids cannot be memorized, the recurrent on-grid fraction drops sharply (LSTM 94% → 70%, GRU 91% → 58%) and their MAE advantage over the ST-Transformer narrows accordingly, while the ST-Transformer's prediction geometry is essentially unchanged (65% → 58%). The ST-Transformer thus behaves as the more genuinely continuous estimator, trading a small amount of in-distribution accuracy for a smoother load mapping that transfers to unseen participants.

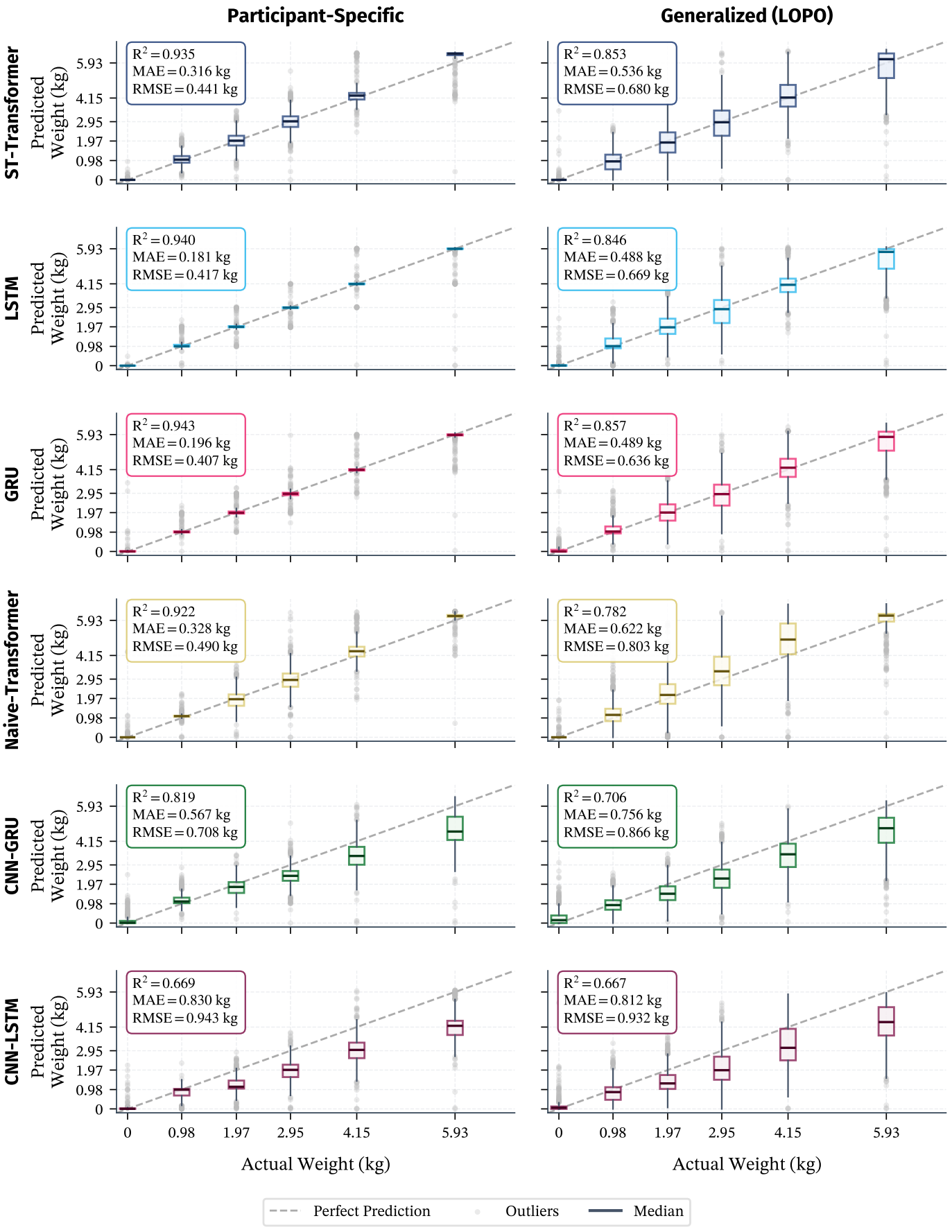


Figure 12. Predicted vs. actual load for all six architectures under participant-specialized (left) and generalized LOPO (right) evaluation. Each box shows the predicted-weight distribution at the six load classes with its corresponding participant-class-macro performance metrics (R^2, MAE and RMSE).

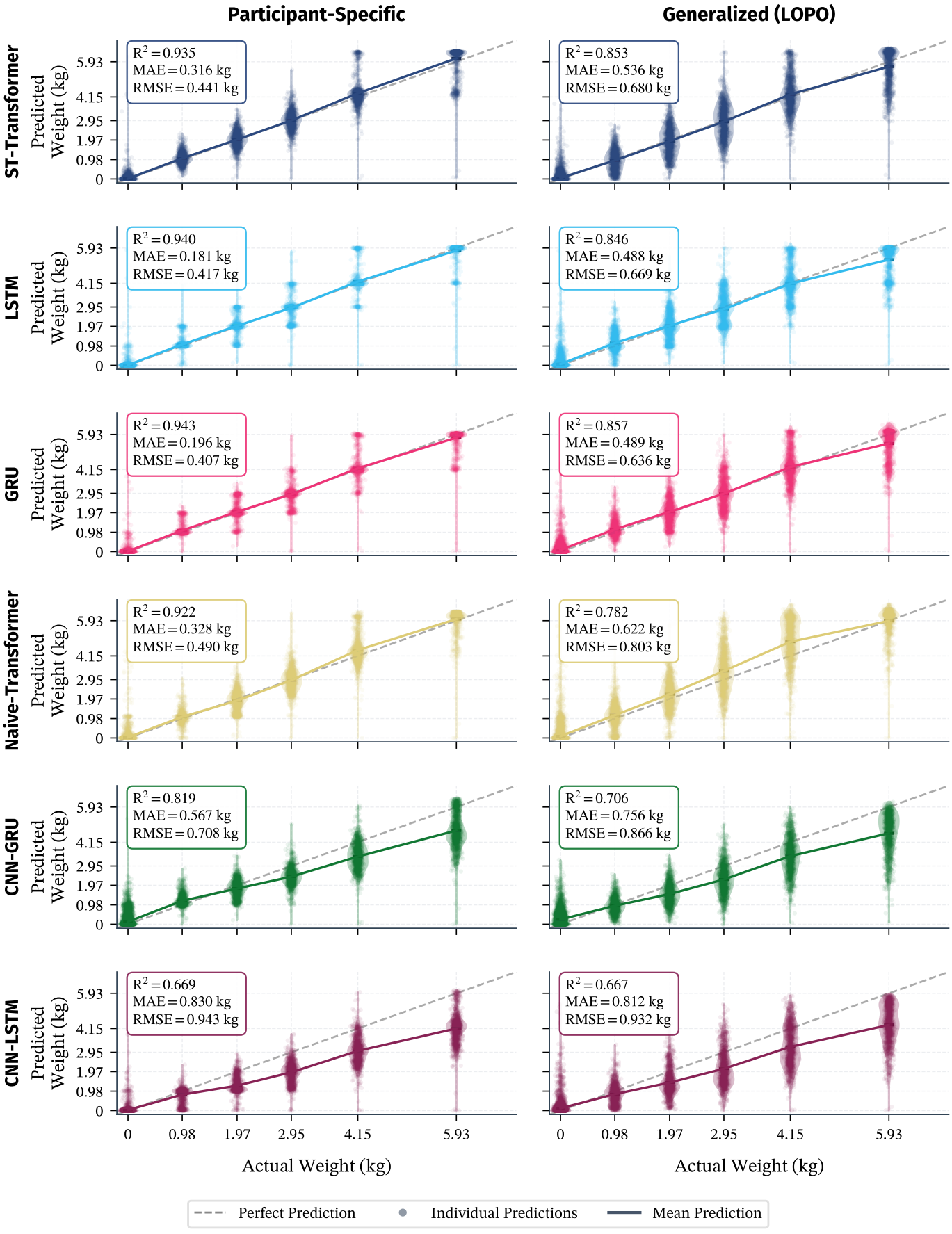


Figure 13. Predicted vs. actual load for all six architectures under participant-specialized (left) and generalized LOPO (right) evaluation. Each violin shows the predicted-weight distribution at the six load classes with its corresponding participant-class-macro performance metrics (R^2, MAE and RMSE).

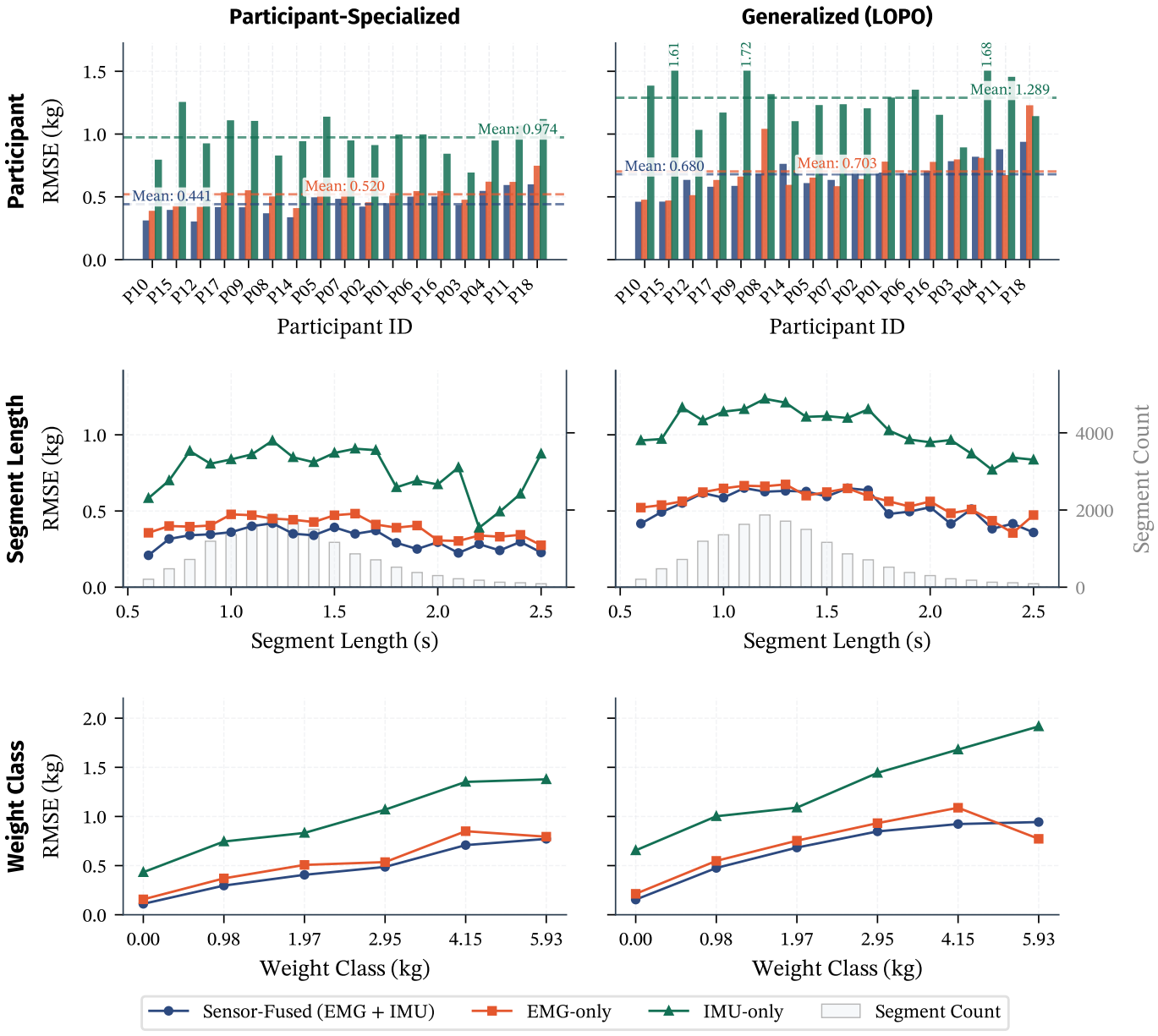


Figure 14. In-depth performance (based on MAE) across both participant-specialized and generalized models, for each modality. Includes performance across participants, segment lengths, and weight classes.

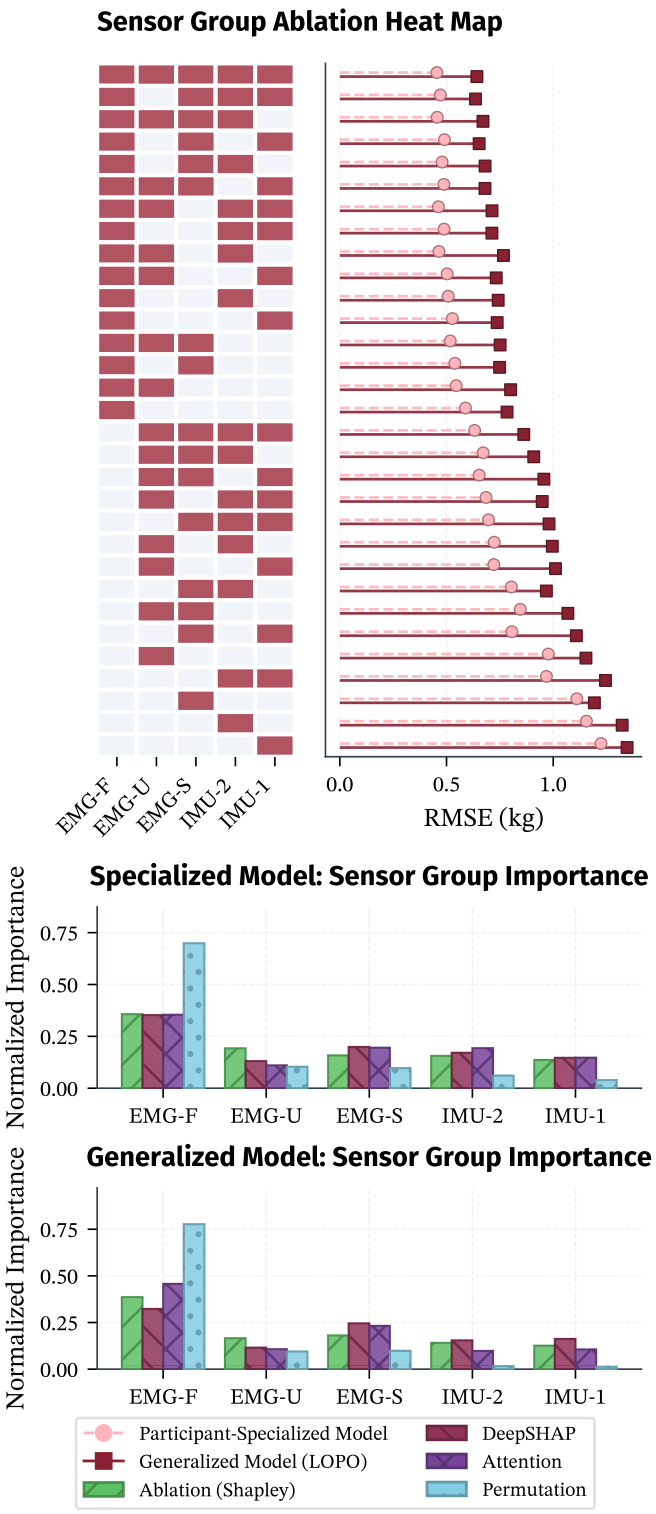


Figure 15. Sensor group ablation study using the participant-specialized validation strategy. Top: MAE and heatmap for all 31 subsets. Bottom: per-group retraining Shapley value compared against the normalized DeepSHAP, attention-pooling importances, and permutation.