eng

IMPACT FACTOR 3.5

CITESCORE 4.1

Article

Enhancing Sensorless Speed Estimation Accuracy Through Global Parameter Identification and Neural Network-Based Residual Compensation

Mana Poyai, Dechrit Maneetham and Petrus Sutyasadi

MDPI

https://doi.org/10.3390/eng7080407

eng

MDPI

Article

Enhancing Sensorless Speed Estimation Accuracy Through Global Parameter Identification and Neural Network-Based Residual Compensation

Mana Poyai [1], Dechrit Maneetham [1,*] and Petrus Sutyasadi [2]

[1] Department of Mechatronics Engineering, Rajamangala University of Technology Thanyaburi, Khlong Luang District, Pathum Thani 12120 ,Thailand; mana_p@mail.rmutt.ac.th
[2] Department of Mechatronics Engineering, Sanata Dharma University, Yogyakarta 55281, Indonesia; peter@usd.ac.id
* Correspondence: dechrit_m@rmutt.ac.th

Abstract

Sensorless speed estimation replaces fragile shaft encoders in cost-sensitive Permanent Magnet Direct Current (PMDC) motor drives, but classical model-based observers degrade under brush friction, commutation ripple, and thermal drift, while purely data-driven estimators sacrifice physical interpretability. This paper presents a Hybrid Physics-Data-Driven Observer (HPDDO) that couples an identified lumped-parameter electrical model with a compact multilayer-perceptron residual compensator, which is executed in real time on a low-cost ESP8266 microcontroller. Global parameters are identified from a short labeled recording, after which the network corrects only the nonlinear residual that the physics model cannot explain. Under a strictly time-series-aware evaluation (chronological 80/20 split), the proposed estimator achieves an average root mean square error (RMSE) of 4.11 RPM across dynamic PWM sweeps, abrupt load transitions, and a long-duration thermal-drift test, outperforming an extended Kalman filter (9.49 RPM), a sliding mode observer (10.89 RPM), and a pure neural-network estimator (7.14 RPM) implemented on the identical dataset. An ablation study shows that accuracy is insensitive to network size, with a 0.9 kB variant matching the deployed model, and a residual-clamping safeguard bounds the estimation error under unseen operating conditions. The framework provides an accurate, interpretable, and computationally lightweight solution for industrial PMDC drives without dedicated speed sensors.

Keywords: sensorless speed estimation; PMDC motor; hybrid physics-data-driven observer (HPDDO); residual compensation; lumped parameter identification; industrial automation; real-time implementation

Academic Editor: Huanyu Cheng

Received: 12 June 2026
Revised: 28 July 2026
Accepted: 7 August 2026
Published: 12 August 2026

1. Introduction

1.1. Industrial Significance of PMDC Motors

Permanent Magnet Direct Current (PMDC) motors are among the most important electromechanical actuators used in industrial automation systems, offering a high torque-to-weight ratio and fast dynamics [1], excellent controllability, and relatively simple drive circuitry [2]. Consequently, the above-mentioned features make PMDC motors especially suitable for precision applications such as robotic manipulators and conveyor systems [3], automated assembly lines, and electric mobility platforms [4].

However, in recent years, industrial drive systems have rapidly evolved due to Industry 4.0 advancements. Robust, adaptable, and self-diagnostic systems that can work in dynamic and uncertain environments are subject to increasing demand as modern manufacturing environments evolve [5], and sensorless motor drives are becoming a key enabler of this transformation [6]. In this context, electric motors are not considered passive components anymore but rather active agents in a cyber–physical system that is monitored and controlled in real time to ensure operational efficiency and reliability.

The rotor speed is one of the key variables that describes the system. An accurate estimation of speed is required for many important tasks, e.g., closed-loop control, fault detection and localization, energy optimization, and system-level coordination, among others. As a result, measuring and estimating motor speed is essential.

1.2. Limitations of Sensor-Based Measurement Systems

Conventionally, motor speed is measured using actual physical sensors, such as optical encoders, tachometers, and resolvers. Although these devices are capable of high-accuracy measurements, when they are installed in harsh environments and restricted spaces, several challenges can arise. Mechanical sensors may be affected by environmental factors such as dust, moisture, thermal effects [6] and vibration during their use, resulting in incorrect measurements [7]. For example, the encoders used in the CNC, pump, and mixer may be affected. Their sensor-based systems might not be reliable over long periods of time due to the vibration factor.

Secondly, physical sensors are also expensive and difficult to use. For example, they need additional hardware, such as signal conditioning circuits, wiring, and mechanical mounting structures, for their high-precision encoders. These requirements not only increase the cost but also make the assembly and maintenance of the system more complex [8]. In embedded systems or compact robotic and drone platforms, the size of physical sensors can also act as a bottleneck.

Third, there are certain operating environments where it is impossible to use physical sensors, such as vacuum environments [8] and high-temperature industrial processes [6]. These limitations have driven the exploration of alternate concepts that do not require a physical speed sensor.

1.3. Sensorless Speed Estimation: Concept and Challenges

Sensorless methods to estimate motor speed are developed to determine motor rotational speed solely based on electrical measurements (mostly armature voltage (V) and current (I)) [6]. Using the inherent coupling between electrical and mechanical quantities, speed estimation can be performed without dedicated physical sensors [8].

The working principle of a DC motor is based on fundamental physical laws such as Kirchhoff's voltage and Newton's second laws of motion. These laws provide a mathematical relationship between electrical input and mechanical output. However, achieving accurate sensorless estimation remains a challenge in practice. Because parameter variation during operation presents a significant challenge, electrical parameters such as armature resistance and back-EMF can vary during prolonged operation, overload, or startup [9]. That makes it difficult for conventional techniques to cover the nonlinear behavior of practical motor such as brush-contact nonlinearity and commutation ripple. These nonlinear behaviors will cause the actual result to deviate from the theoretical estimates.

1.4. Conventional Model-Based Estimation Techniques

As discussed in the previous section, sensorless speed estimation is required for certain DC motor applications. Several techniques and methods have been proposed to address this need to estimate speed without having a physical encoder mounted on the motor.

These techniques commonly employ mathematical models derived from the dynamic behavior of the DC motor. Representative examples include the Luenberger observer [10], the extended Kalman filter (EKF) [11], the sliding mode observer (SMO) [12] and Model Predictive Control (MPC) [13], among others.

However, as noted in [14], these techniques still present several challenges, such as thermal drift during prolonged operation of the DC motor, among others, as listed below.

- Computational Complexity: The EKF technique is based on the calculation of Gaussian noise, which involves the expensive calculation of the inversion matrix and the Jacobian evaluation at each time step in real time [11]. This creates an edge device with high computational density, as mentioned in the work of Mohamed et al. [15].
- Disturbance Sensitivity: Sliding mode observers are widely favored for their robustness against external load disturbances [8]. However, they are susceptible to the so-called chattering phenomenon, where the switch nature of the observer gain causes high-frequency torque ripples [12] that can adversely affect mechanical drive components [12]. A further critical practical problem is DC offset in the current and voltage measurement chains, which propagates into severe estimation errors in conventional SMO-based observers; recent work addresses this with improved SMO structures combined with multi-stage cascaded filters and enhanced phase-locked-loop architectures [16]. Section 4.11 discusses how the residual-learning structure of the framework proposed in this paper offers a complementary pathway for such non-ideal measurement conditions.
- Parameter Dependency: Several classical observers, such as the Luenberger-based designs discussed by Suwankawin et al. [10], assume a constant electrical time constant. This assumption no longer holds in high-dynamic-demand applications where magnetic saturation [9] and flux leakage are significant [17].

In recent years, efforts have been made to address these issues using adaptive mechanisms [17] and smart parameter identification via deep learning [18]. For example, AI-based diagnostic layers were suggested by Awadallah et al. [19] to adaptively optimize observer parameters online. However, the underlying physics may not be straightforward to define for the nonlinear and non-ohmic contacts arising from the brush–commutator interface [20]. This motivates the use of neural-network-based approaches to compensate for these unresolved dynamics, achieving higher-fidelity solution capabilities that exceed the scope of classical analytical models [21].

1.5. Emergence of Data-Driven Approaches

Data-driven approaches based on machine learning have been attracting increasing attention as a powerful alternative to traditional model-based methods in recent years. They learn the mapping between high-dimensional electrical inputs and mechanical outputs from experimental datasets to directly bypass any explicit analytical modeling of highly nonlinear phenomena [5], enabling applications such as condition monitoring and fault detection [22]. The availability of a well-defined input–output system (the motor drive) enables data-driven methods to capture hidden dynamics that are often oversimplified in traditional models [21].

Artificial neural networks have shown the ability to handle nonlinear factors and to identify relationships among multiple input parameters. Multilayer perceptrons (MLPs), in particular, can capture the hidden, complex mapping [23]—as formalized by the universal approximation theorem [24]—of nonlinear phenomena such as friction, commutation ripple, and the nonlinear brush contact resistance [20]. Advanced optimizers such as Adam are, in addition, employed to accelerate convergence during training [25].

In addition, the current trend in neural network development goes beyond purely data-driven models: e.g., batch normalization [26] is used to reduce internal covariate shift and makes training more stable, resulting in a higher accuracy of estimation. Moreover, combining physical governing equations with a data-driven neural network into a Hybrid Physics-Data-Driven Observer (HPDDO) will increase the precision of the prediction by aligning the estimation with the physics law of DC motor operation [27].

However, to merge between physics and neural network prediction, it might have some challenges that need to be considered, as shown in the list below.

- Data Dependency and Generalization: The data are fed to the neural network as information for learning. Therefore, to ensure a high-quality learning model, the training data should cover a wide range of operating conditions, including load step and sweep speed values; as mentioned in [15], the model will achieve improved generalization performance.
- Computational and Memory Need: For the neural network learning concept, it will have more requirements than the classical mathematical calculation technique. For this technique, it will require the higher calculation and memory as stated by Goodfellow et al. [24] and Tom et al. [21]. However, nowadays, the advancement of microcontroller technology is more advanced compared to the price, such as the ESP8266 microcontroller, which can run the model deployed from the neural network for a small effort price.
- Non-Interpretability: By using the pure data of a neural network, sometimes the black box phenomenon occurs in which the output seems unreasonable and physically implausible [28]. Therefore, incorporating physical constraints into the neural network yields more physically consistent predictions.

In summary, the hybrid model [28] combines the strengths of both approaches: the physics-based calculation in the linear region and the neural-network prediction in the nonlinear region, achieving higher speed-estimation precision than either approach independently.

1.6. Research Gap and Motivation

As noted previously [1], the purely analytical approach based on the basic DC motor equations predicts rotor speed effectively in the linear region and uses far fewer controller resources than a purely data-driven neural network. However, as mentioned in [14], in nonlinear situations such as thermal drift, commutator ripple, mechanical friction, etc. [20], there will be degradation over the lifetime of the motor due to thermal effects [14] and parameter drift [17] that will result all in nonlinear phenomena corrupting any predictions made by this classical technique.

However, a purely data-driven neural network behaves as a black box [28]. This, as mentioned by [24], leads to a catastrophic forgetting of distribution training data and poor extrapolation [29]. Consequently, purely data-driven techniques cannot guarantee the accuracy of the prediction all the time during its operation [28].

Therefore, this gap is addressed by merging the benefits of both techniques [27] into the hybrid model of linear and nonlinear prediction of a high-performance unit controller such as DSP [7] or FPGA [6].

However, the high-performance unit controller is not the practical controller in the industrial environment [7]. The controller that merges the linear and nonlinear predictions should therefore be simpler and less expensive than these high-performance units [21] to remain practical for industrial DC motor drives.

Our work therefore focuses on merging the two benefits of sensorless estimation [6]—covering both the linear and nonlinear regions of DC motor operation by combining physics and AI—on a low-cost controller.

1.7. This Paper's Scope and Contributions

Motivated by the technical challenges presented above, this paper introduces a three-model identification method for DC motors. Specifically, it investigates a hybrid integration of physical modeling and residual learning with the expectation that this coupling effectively addresses the nonlinear behavior inherent in variable-load, PWM-sweep, and long-duration running conditions. This paper makes the following key contributions.

- Physics-Data-Driven Hybrid framework: The proposed framework comprises a hierarchical modeling pipeline that preserves the physical interpretability of the analytical baseline while exploiting the high-dimensional function approximation capabilities of neural networks. In contrast to previous hybrid schemes that rely on discrete switching between sub-models, the proposed architecture continuously compensates for and decouples spatially localised nonlinearities, such as brush–commutator impedance drift and magnetic saturation effects, through a continuous residual compensation layer.
- Identification of Global Parameters: The lumped-parameter electrical and mechanical constants of the motor are identified using onboard electrical sensing data synchronized with a one-time offline encoder reference thereby characterizing local friction and damping properties of the test bench assembly and minimizing systematic estimation bias in parameter identification [9] and friction characterization [20].
- Real-Time Implementation on Resource-Constrained Hardware: A primary contribution of this paper is the deployment of high-fidelity estimation models on low-cost edge microcontroller hardware. To this end, a low-overhead design pattern is proposed in which the master controller, driven by deterministic hardware interrupts (DHIs), offloads neural inference tasks to the ESP8266 co-processor in a non-blocking manner, thereby satisfying the real-time cycle-time constraints of industrial drive systems [7].
- Validation of Robustness and Generalization: An extensive experimental protocol that includes dynamic tests such as abrupt load switching transients, extended thermal drift cycles, and a PWM sweep condition is used to validate the proposed framework. We measure performance through Root Mean Square Error (RMSE), which reflects accurate tracking [30].

In addition to these technical contributions, this paper demonstrates that high-fidelity sensorless estimation is achievable at realistic performance levels using relatively low-cost computational hardware and a lightweight observer architecture. The proposed framework is inherently fault-aware, capable of detecting and isolating incipient failures, and easily scalable through a combination of fault-detection mechanisms and model compensation that draws on the complementary strengths of analytical physical models and neural networks. It therefore provides a comprehensive methodology for improving motor-drive performance in real-world scenarios where installing mechanical sensors is cost-prohibitive [6] or practically infeasible [31].

The remainder of this paper is structured as follows. Section 2 presents the theoretical background of the DC motor covering the governing electromechanical equations and the mathematical foundations of the MLP and Hybrid Physics-Data-Driven Observer (HPDDO) architecture. Section 3 describes the proposed three models method, experimental test bench and data processing pipeline. Section 4 presents the experimental results and comparative performance analysis. Section 5 concludes the paper and discusses its limitations.

2. Theory of DC Motor

A DC motor turns electrical energy into mechanical work by letting two magnetic fields interact. For sensorless estimation, this physical structure matters a great deal: the nonlinear effects built into the motor's construction are exactly what make estimation difficult.

Note on model notation: This section uses three labels—M1, M2, and M3—to tag the three estimation layers. M1 is the nameplate based physics model, M2 adds empirically identified parameters, and M3 is the MLP residual compensator. Full definitions and implementation details are in Section 3; the labels here are used only for consistency with the figures and equations.

2.1. Internal Physical Structure

Figure 1 shows the two main parts of a PMDC motor: the stator and the rotor (armature). Permanent magnets in the stator set up a constant flux density (**B**) across the air gap [1].

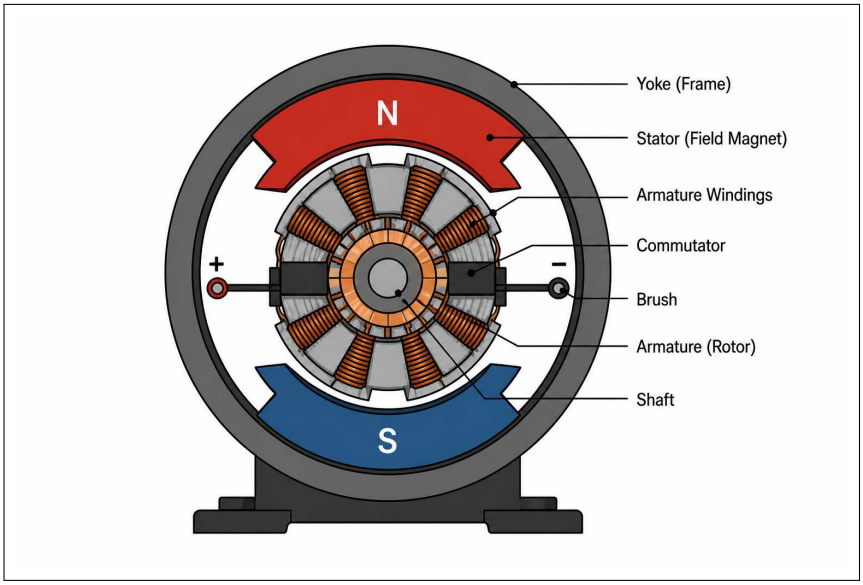

Figure 1. DC motor structure.

Inside the rotor, laminated iron carries copper windings. Current reaches those windings through carbon brushes pressing against the commutator, which is a device that reverses current in each winding every half turn, ensuring the resulting torque always points the same way [2].

2.2. Electromechanical Energy Conversion

Torque comes from the Lorentz force on each current-carrying conductor. For a conductor of length **L** carrying current I inside field **B** [1,3],

$$\mathbf{F} = I(\mathbf{L} \times \mathbf{B}) \tag{1}$$

As the coils spin, they also generate a back EMF. That voltage is the basis for sensorless speed estimation [6].

2.3. Structural Nonlinearities and Modeling Challenges

Linear models work as a starting point, but three nonlinear effects in real hardware are hard to capture.

- Brush Commutator Interface: Contact between the brush and commutator during operation introduces time-varying contact resistance, producing fluctuations in armature current that are difficult to model analytically [20].
- Magnetic Saturation: Under heavy load, high armature current causes the magnetic flux to saturate [1], resulting in a nonlinear relationship between current and torque that differs from nominal operating assumptions [2].
- Mechanical Friction: Static Coulomb friction and velocity dependent viscous drag [20] are not adequately captured by classical linear models. Dedicated data-driven identification is required to characterize these frictional nonlinearities beyond lumped parameter assumptions [32].

Working from the equivalent circuit in Figure 2, including R_a and L_a in the observer gives a more accurate state estimate.

The back EMF constant (K_e) is the link between electrical and mechanical domains. It scales the relationship between shaft speed (ω) and induced voltage (E), representing flux linkage per radian [1].

In a real machine, K_e drifts. Magnetic saturation and general parameter aging both shift it during a run [2,9], and brush-commutator arcing adds further disturbance to the back EMF waveform [2].

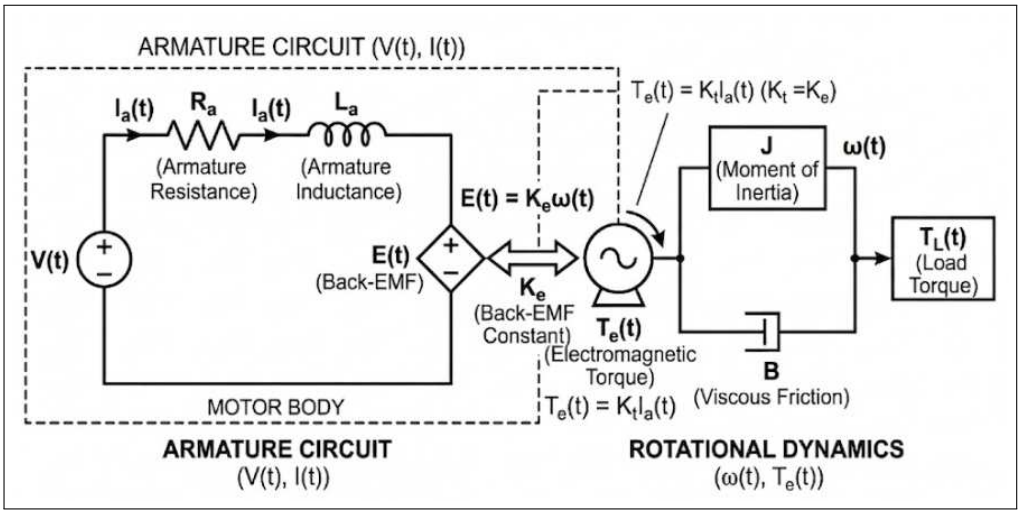

Figure 2. DC motor equivalent electrical circuit.

Together, the electrical and mechanical equations characterize a comprehensive model of motor operation. What allows a good observer to exist is having the correct model form.

2.4. Electromechanical Governing Equations

Applying KVL to the armature circuit with resistance R_a, inductance L_a [1], and back EMF $e(t)$ [2],

$$V(t) = R_a i_a(t) + L_a \frac{di_a(t)}{dt} + e(t) \tag{2}$$

$V(t)$ is terminal voltage and $i_a(t)$ is armature current. Back EMF scales linearly with speed through K_e.

$$e(t) = K_e \omega(t) \tag{3}$$

The electrical time constant $\tau_e = L_a/R_a$ sets how fast the current responds. Dropping the $L_a \frac{di_a}{dt}$ term is only safe in slow, steady-state applications; transient analysis and high-bandwidth observers must keep it [1,2].

2.5. Mechanical Rotational Dynamics

Newton's second law for rotation ties electromagnetic torque (T_e), load torque (T_L), and inertia together [1,4].

$$T_e(t) = J\frac{d\omega(t)}{dt} + B\omega(t) + T_L(t) \tag{4}$$

J is the rotor's moment of inertia and B is the viscous friction coefficient. Electromagnetic torque ties to current through K_t.

$$T_e(t) = K_t i_a(t) \tag{5}$$

In SI units, the torque constant equals the back EMF constant, so $K_t = K_e$ [1].

2.6. State-Space Representation

We collect the electrical and mechanical equations into state-space form with $\mathbf{x} = [i_a, \omega]^T$.

$$\begin{bmatrix} \dot{i}_a \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} -\frac{R_a}{L_a} & -\frac{K_e}{L_a} \\ \frac{K_t}{J} & -\frac{B}{J} \end{bmatrix} \begin{bmatrix} i_a \\ \omega \end{bmatrix} + \begin{bmatrix} \frac{1}{L_a} \\ 0 \end{bmatrix} V - \begin{bmatrix} 0 \\ \frac{1}{J} \end{bmatrix} T_L \tag{6}$$

The result is a complete linear model. Linearity is also its limitation: brush friction, contact resistance, and magnetic saturation all sit outside it, so some residual error is unavoidable [2].

Linear models like the Luenberger observer [10] and EKF [11] are still the starting point for speed estimation, but they assume ideal conditions that real hardware violates. Brush contact resistance [20], saturation [2], and friction [4] each introduce error. This paper extends the framework by targeting those nonlinear and thermal effects directly.

2.7. Mathematical Foundation and Universal Approximation

At the heart of the MLP is one basic result. According to the Universal approximation theorem, even a single hidden layer feedforward network can approximate any continuous function as closely as required [23].

Here, the MLP's job is to learn the residual—the gap between what the physics model predicts and what the motor actually does [27]. That is a much narrower target than estimating absolute speed, which is what makes the approach practical. The learned mapping is $\mathcal{F} : \mathbf{X} \to \epsilon$, where $\mathbf{X} = \{V, I, \hat{n}_{M2}, I_{t-1}, \hat{n}_{M2,t-1}\}$ packages the current electrical state together with one-step lags to give the network temporal context (see Section 3).

A linear observer is trapped inside its state-space assumptions. An MLP is not: it acts as a data-driven pattern-recognition machine for non-stationary signals that does not presuppose the governing equations. The network learns these effects via backpropagation [29], modeling brush contact nonlinearity [6], frictional hysteresis and electromagnetic interference to become an adaptive observer that continues tracking in the face of large disturbances [24].

2.8. Data Preprocessing and Feature Scaling

Before training begins, every input feature is normalized. The five variables in $\mathbf{X} = \{V, I, \hat{n}_{M2}, I_{t-1}, \hat{n}_{M2,t-1}\}$ span very different numerical ranges; volts, amperes, and RPM sit on completely different scales. Feeding them raw into the network produces an ill-conditioned loss surface on which gradient steps oscillate or diverge.

Applying StandardScaler maps each feature to approximately $\mathcal{N}(0,1)$ so none of them dominate the weight updates.

$$\tilde{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sigma_j} \tag{7}$$

where μ_j and σ_j denote the mean and standard deviation of the j-th feature, respectively:

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{i,j}, \quad \sigma_j = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_{i,j} - \mu_j)^2} \tag{8}$$

ReLU activations also benefit from feature normalization. Because ReLU is unbounded for positive inputs, unscaled features with large magnitudes can produce large pre-activations and correspondingly large, poorly conditioned gradients. Mapping each feature to approximately $\mathcal{N}(0,1)$ keeps the pre-activations within a moderate range, balances the contribution of every input, and promotes stable convergence of the Adam optimizer [24].

Scaler statistics (μ_j, σ_j) are computed only on the training fold, and the same transform is then applied to test and validation data. Fitting on the full dataset would leak target information and inflate the reported RMSE [24,28].

2.9. MLPRegressor Architectural Design and Forward Dynamics

Figure 3 shows the feedforward MLP used as the residual learner. Each of its L hidden layers applies a nonlinear map to the previous layer's output.

$$\mathbf{a}^{(l)} = \sigma(\mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}) \tag{9}$$

$\mathbf{W}^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$ is the weight matrix, $\mathbf{b}^{(l)} \in \mathbb{R}^{n_l}$ is the bias, and $\sigma(\cdot)$ is the activation. Stacking layers builds up the capacity to fit complex motor dynamics. L is chosen to stay within the ESP8266's memory capacity.

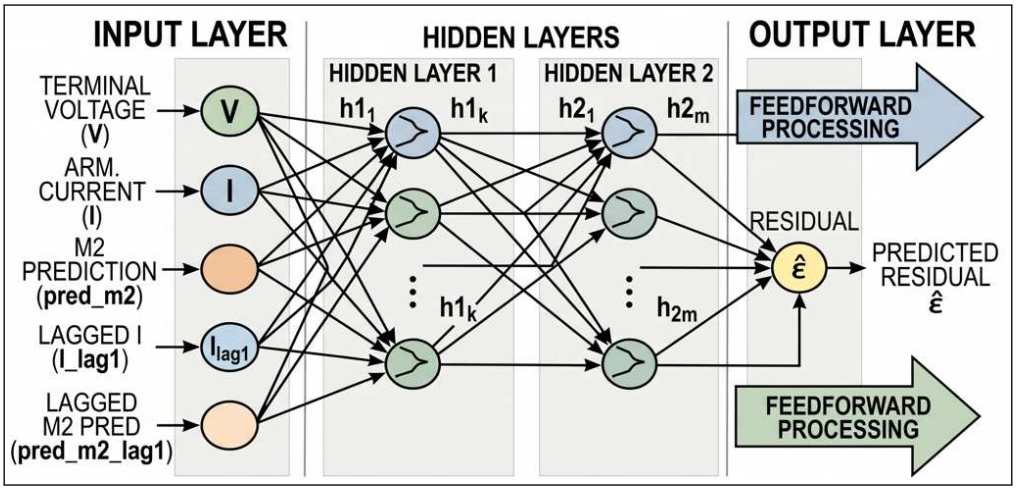

Figure 3. Conceptual architecture of the MLP residual compensator. The feedforward structure uses ReLU activation in the hidden layers, a linear output activation, and the Adam optimizer. Five engineered input features are used: terminal voltage (V), armature current (I), refined speed estimate ($\hat{n}_{M2}$), and their lagged values at $t-1$, namely armature current (I_{t-1}) and refined speed estimate ($\hat{n}_{M2,t-1}$). The network consists of two hidden layers with 128 and 64 neurons, respectively, and produces a single output, the predicted residual $\hat{\varepsilon}$.

The Rectified Linear Unit (ReLU) is employed as the activation of all hidden layers.

$$\sigma(z) = \mathrm{ReLU}(z) = \max(0, z) \tag{10}$$

ReLU is chosen over saturating activations for three reasons. First, it is computationally inexpensive, requiring only a threshold operation with no exponentials, which is critical for real-time inference on the resource-constrained ESP8266. Second, it does not saturate for positive inputs: its derivative is unity for $z > 0$ (Figure 4), so gradients propagate without attenuation, and the vanishing-gradient problem that affects saturating functions is mitigated. Third, negative pre-activations are mapped to exactly zero, yielding sparse and efficient representations.

$$\sigma'(z) = \begin{cases} 1, & z > 0 \\ 0, & z < 0 \end{cases} \tag{11}$$

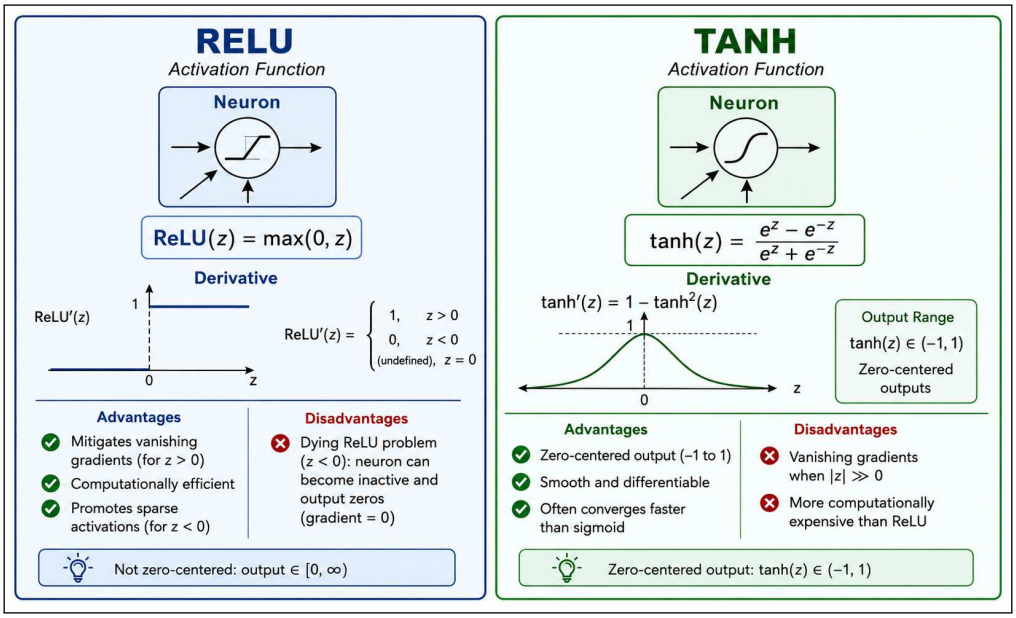

Figure 4. The ReLU activation function and its derivative compared with the saturating Tanh function.

Full connectivity lets every neuron see every input feature simultaneously, which is essential for capturing cross-couplings, for example, between current, speed, and thermally shifting resistance [33]. That coverage is what lets the MLP close the gap where the linear baseline falls short.

2.10. Backpropagation and Adam Optimization Dynamics

The training objective is MSE, $\mathcal{L}_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\epsilon_i - \hat{\epsilon}_i)^2$. RMSE is then used for reporting because it carries the same units as speed (RPM), making the number directly readable. Backpropagation carries the loss gradient back through the network. The error term at neuron j in layer l is

$$\delta_j^{(l)} = \left(\sum_k \delta_k^{(l+1)} w_{kj}^{(l+1)}\right)\sigma'(z_j^{(l)}) \tag{12}$$

The chain rule distributes responsibility for the error to every weight. Standard SGD struggles with the non-stationary signals that arise from real industrial measurements, so

Adam is used instead. Adam keeps exponential moving averages of the gradient and its squared value, giving each weight its own adaptive step size [25]:

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon} \tag{13}$$

$\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first and second moment estimates, while $\varepsilon \approx 10^{-8}$ keeps the denominator from hitting zero [25]. The per-parameter rates damp the noisy gradient signal that comes from raw electrical measurements [25,29] and keep optimization moving on the high-dimensional residual loss surface [25].

2.11. Residual Compensation and Hybrid Stability

Figure 5 shows the complete hybrid observer. The physics model gives a globally stable baseline. The MLP corrects localized nonlinear residuals on top of it. Combined, the final estimate is

$$\hat{\omega} = \omega_{phy} + \mathcal{F}(\mathbf{X}) \tag{14}$$

where ω_{phy} is the physical model's speed estimate and $\mathcal{F}(\mathbf{X})$ is the MLP's residual correction.

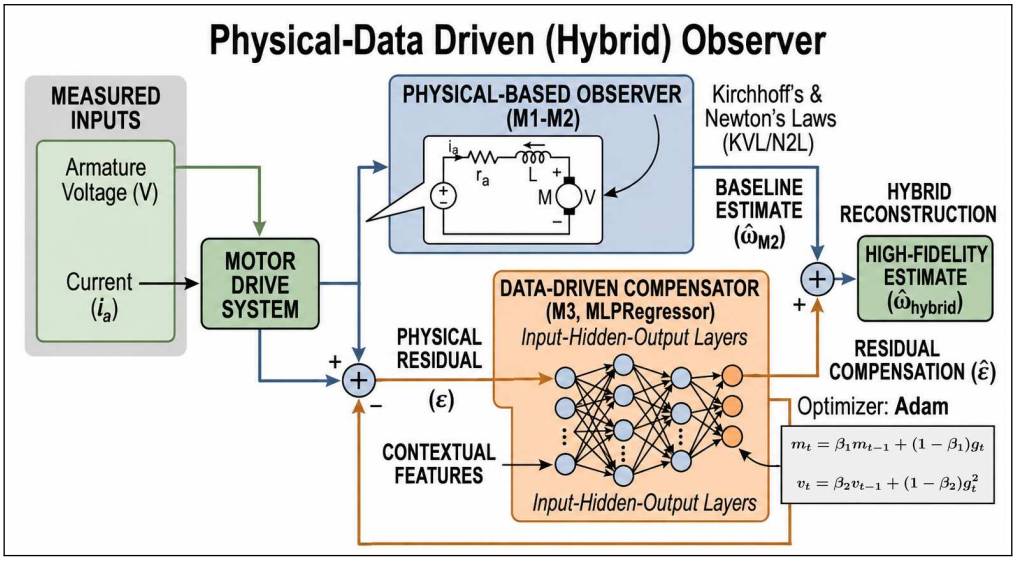

Figure 5. Physical data are required. Driven (Hybrid) observer architecture for sensorless speed estimation. The physical-based observer (M1–M2) processes armature voltage (V) and current (i_a) via Kirchhoff's and Newton's laws (KVL/N2L) to produce a baseline speed estimate ($\hat{\omega}_{M2}$). The physical residual (ϵ) is computed by comparing $\hat{\omega}_{M2}$ against the measured output, and it is passed alongside contextual features to the data-driven compensator (M3, MLPRegressor), which produces a residual compensation term ($\hat{\epsilon}$) using the Adam optimizer. The final high-fidelity estimate ($\hat{\omega}_{\text{hybrid}}$) is obtained by summing the baseline and residual outputs.

If the MLP ever encounters operating conditions far from its training data, the physical baseline still provides the absolute speed scale, although the residual correction itself is not explicitly bounded. The MLP then accounts for the drift parameters, thermally induced resistance changes [14] and friction nonlinearities [20], which cannot be tracked by a physics model.

The two components are trained separately: the physical model sets the baseline, and the MLP corrects what the baseline cannot capture. This split keeps the system both predictable and safe for industrial use. It also achieves higher accuracy than classical observers, which rely on manual parameter tuning [6,17]. Should the MLP become unavailable at run time, the physical baseline alone still provides a usable, lower-fidelity estimate.

2.12. Performance Evaluation Metrics the Role of RMSE

RMSE is the chosen evaluation metric:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{15}$$

where y_i is the encoder ground truth and $\hat{y}_i$ is the hybrid estimate. RMSE is preferred over MAE or MAPE for three reasons:

- Large Estimate Outlier Penalty: The RMSE metric gives more weight to larger deviations from the real value by squaring the residuals before averaging [30]. In industrial applications, RMSE is a standard performance metric for speed estimation in drive systems [30].
- Gradient Compatibility: The RMSE is derived from the MSE loss, which is an analytical quadratic function and therefore smooth and continuously differentiable everywhere. This property makes it directly compatible with gradient-based optimization algorithms, ensuring the stable convergence of the Adam optimizer during neural network training [25], which is consistent with deep learning optimization principles [24].
- Physical Interpretability: Because RMSE is expressed in the same units as the target variable (RPM), its value directly represents the average speed estimation error, facilitating a straightforward comparison between observer designs [30]. Under the assumption that residuals follow a Gaussian distribution, the RMSE also provides a statistically grounded basis for benchmarking estimation performance [30].

MSE and RMSE represent mathematically equivalent quantities, differing only by a square root operation. During training, squaring the residuals penalizes larger errors more heavily—a desirable property in our system given the occurrence of load switching—thereby discouraging convergence toward solutions that suffer from severe transient spikes [6,30].

Furthermore, squaring acts as a form of implicit regularization that penalizes systematic and repeatable errors (such as friction, thermal drift, and commutation ripple) more strictly than random measurement noise, establishing an appropriate error hierarchy for this application. Although MAE may be preferred when average performance is the primary concern [34], RMSE was selected for this paper due to its unit consistency (RPM), sensitivity to transient peaks, and direct connection to the training loss function.

3. Materials and Methods

The conceptual diagrams in this section and in Section 3.4 were drafted with the assistance of a generative AI tool and were subsequently edited and finalized by the authors, who verified all technical content and labels; tool details are given in the Acknowledgments.

3.1. System Architecture and Theoretical Framework

Three estimation layers make up the proposed framework. Their roles are summarized in Table 1.

As shown in Figure 6, the three methods are as follows.

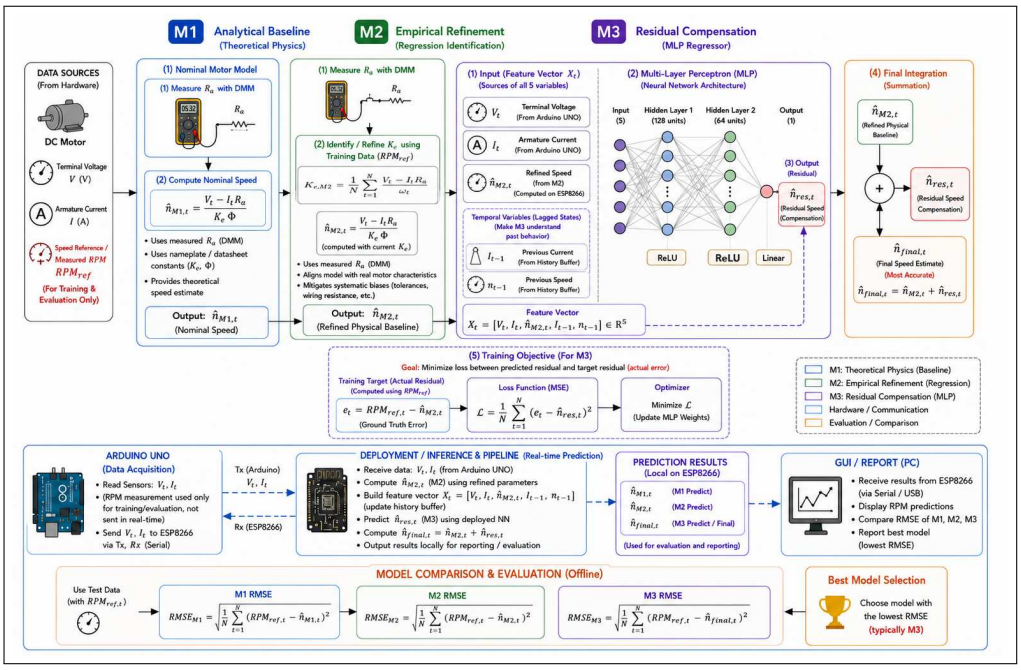

Figure 6. Overview of the M1, M2, and M3 speed estimation framework. In the feature vector, n_{t-1} denotes the lagged refined estimate $\hat{n}_{M2,t-1}$.

Table 1. Model M1, M2, and M3 summary definition.

Model	Methodology	Description
M1	Theoretical Physics	Establishes baseline from motor nameplate specifications.
M2	Empirical Identification	Refines K_e using steady-state regression.
M3	Residual Compensation via MLP	Captures nonlinear error (e) via MLP regressor [1].

[1] M3 method uses residual compensation to mitigate unmodeled dynamics.

1. Theoretical Physics (M1): This method uses the fundamental back-EMF equation to predict the speed. The electrical constants are taken directly from the manufacturer's nameplate, providing a theoretical baseline estimate $\hat{n}_{M1}$.
2. Empirical Identification (M2): This method advances beyond the theoretical baseline (M1) by using steady-state regression to identify the effective back-EMF constant $K_{e,M2}$ of the motor. Model M2 aligns the physical model with actual hardware characteristics by processing experimental steady-state data points and reducing systematic biases, such as component tolerances and wiring resistance.
3. MLP Residual Compensation (M3): The final layer employs a multilayer perceptron (MLP) to compensate for the nonlinear residual ($e = RPM_{ref} - \hat{n}_{M2}$). The MLP takes an engineered feature vector of terminal voltage V, armature current I, the M2 prediction $\hat{n}_{M2}$, and lagged temporal variables at time $t-1$; then, it outputs the residual speed estimate $\hat{n}_{res}$.

The final speed estimate combines the M2 output and the M3 residual correction:

$$\hat{n}_{final} = \hat{n}_{M2} + \hat{n}_{res} \tag{16}$$

This HPDDO-based observer accounts for real-world nonlinearities such as brush friction, contact resistance variation, and thermal drift, which are not included in the physics-based equations.

3.2. HPDDO Loss Function

The MLP in M3 is trained to minimize the mean squared error (MSE) between the predicted residual $\hat{n}_{res}$ and the actual residual e, which is defined as the difference between the measured speed RPM_{ref} and the M2 prediction $\hat{n}_{M2}$:

$$e_t = RPM_{ref,t} - \hat{n}_{M2,t} \tag{17}$$

$$\mathcal{L} = \frac{1}{N}\sum_{t=1}^{N}(e_t - \hat{n}_{res,t})^2 \tag{18}$$

The MLP weights are updated using the Adam optimizer to minimize $\mathcal{L}$ over the training set. No physics-based penalty term is applied; physical consistency is instead enforced structurally through the M2 baseline, which ensures that the network learns only the residual deviation unexplained by the back-EMF model.

3.3. MLP Training Configuration

Table 2 lists the training hyperparameters for M3.

Table 2. MLP training hyperparameters.

Hyperparameter	Value
Hidden layers (neurons)	128, 64
Activation function	ReLU
Output activation	Linear
Optimizer	Adam
Learning rate (α)	1×10^{-3}
Batch size	200 (`auto`)
Max epochs	2000
Early stopping	Disabled
Evaluation protocol	Chronological 80/20 split (time-series-aware)
Input features ($\lvert\mathbf{x}\rvert$)	5
Input feature set:	$[V_t,\ I_t,\ \hat{n}_{M2,t},\ I_{t-1},\ \hat{n}_{M2,t-1}]$

3.4. Experimental Test Bench Overview

Validation used a dedicated test bench (Figure 7) with two 12 V DC gear motors mounted back-to-back. One motor runs under test. The other, driven as a generator through a resistive load, provides the load torque. A precision machined coupler locks the shafts together. Swapping the load resistor sweeps the torque from zero to high load. A complete bill of materials for the test bench is provided in Appendix A (Table A1).

An Arduino UNO generates the PWM drive signal for the MOSFET stage. A fast opto-coupler sits between the microcontroller and the power stage, blocking switching transients from corrupting the analog measurements.

Terminal voltage is scaled to the ADC range through a resistive divider. Armature current is read across a shunt resistor. Both channels feed the Arduino's 10-bit ADC at the same instant.

The raw sample is forwarded over serial to the ESP8266, which runs the two-stage hybrid estimator and streams the result back. This division keeps acquisition deterministic on the Arduino and moves the heavier floating-point work to the ESP8266.

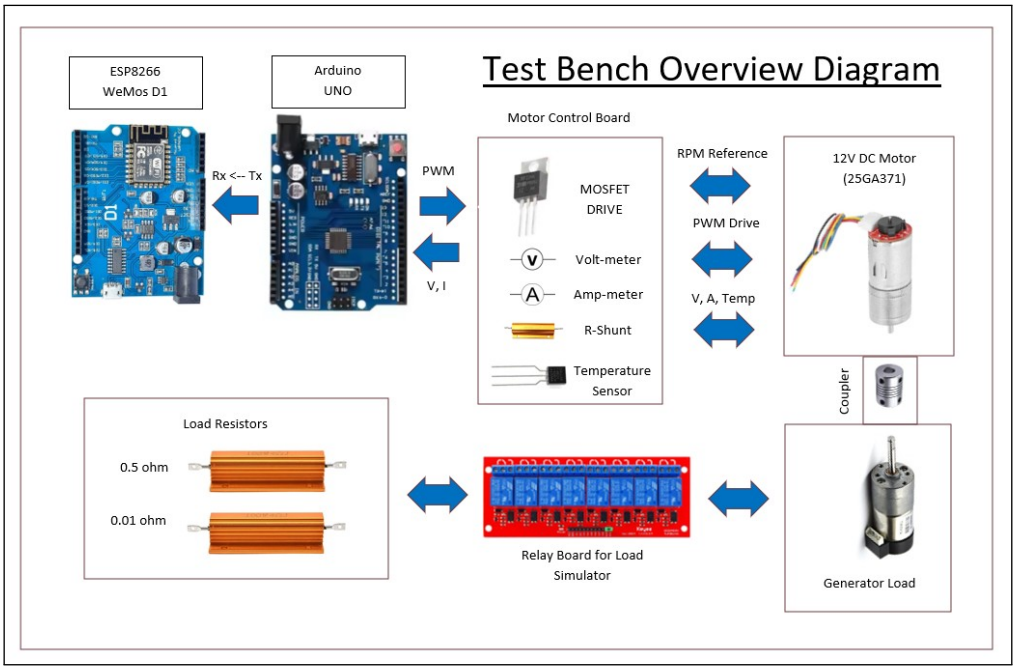

Figure 7. Overview of the experimental system architecture featuring the back-to-back motor generator setup.

3.5. Instrumentation and Controller Board Design

Figure 8 shows the motor driver and sensing circuit. The switching ground of the PWM stage is isolated from the measurement ground to prevent ground-loop noise.

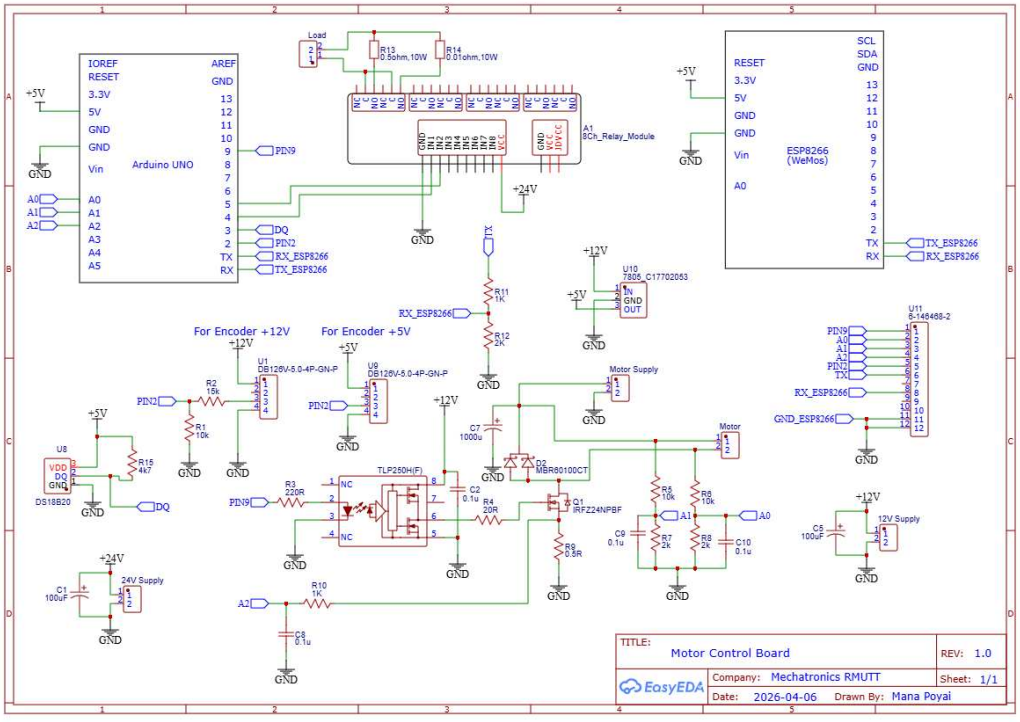

Figure 8. Detailed schematic of the DC motor drive and measurement circuit.

The design separates a low-voltage control side from a high-voltage switching side. Opto-couplers between them stop voltage spikes from reaching the microcontroller and keep the measurement ground undisturbed.

A resistor divider passively divides the motor terminal voltage into the ADC range. Due to its purely passive nature, the network is uninfluenced by bias and bandwidth-limiting effects, and it is generally not supply-noise coupled. The mapping from the sensor output voltage to the digital signal remains stable and linear whilst introducing no computational load.

Included on the board is a DS18B20 temperature sensor for monitoring and potential future diagnostic expansion; this is not utilized as an input feature in the current estimator with thermal effects instead inferred indirectly based on their influence on the measured V and I.

A shaft-mounted optical encoder supplies the ground-truth speed label during training, as visible in Figure 9. Once the model is trained and deployed on the ESP8266, the encoder wire is disconnected from the inference path, so the ESP8266 sees only V and I. The encoder stays on the shaft purely to record the true speed for comparison. The RMSE between the ESP8266 output and the encoder reading is the accuracy measure.

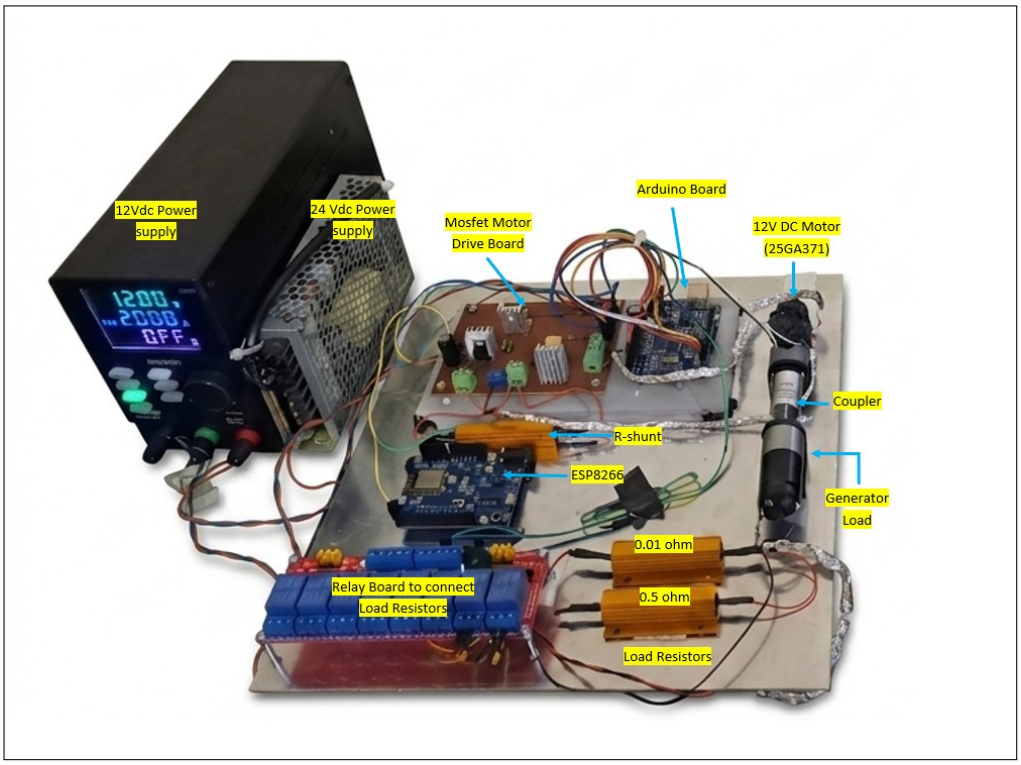

Figure 9. Actual experimental test bench configuration used for model validation.

3.6. Experimental Workflow and Data Processing

The framework combines physical motor dynamics (M1, M2) with residual learning (M3). The complete workflow is shown in Figure 10.

3.6.1. Data Acquisition Protocol

Three test protocols together span the motor's full operating range.

- PWM Sweep Test: To obtain dynamics across the entire range while mitigating hysteresis, the duty cycle was swept from 0% to 100% and back to 0% at every 20%. This bidirectional protocol design helps to ensure that the model easily learns a symmetric response behavior (into and against) when being accelerated as well as decelerated.
- Load Disturbance Test: Mechanical load transitions occurred due to relay-switched resistor loads (0.5 Ω, 0.01 Ω) at constant 80% PWM duty cycle states, simulating discrete mechanical stress inputs on the system.

- Long-Duration Stability Test: To evaluate how the estimations are consistent over thermal drift/mechanical fatigue, a long-duration endurance recording was performed at 80 % PWM with the 0.01 Ω load, during which the motor case temperature rose from 31.7 °C to a steady 71.3 °C.

A threshold filter removes sensor glitches from the raw data. One-step lags I_{lag1} and $\hat{n}_{M2,lag1}$ are appended to each sample so the model sees both the current state and the direction it came from.

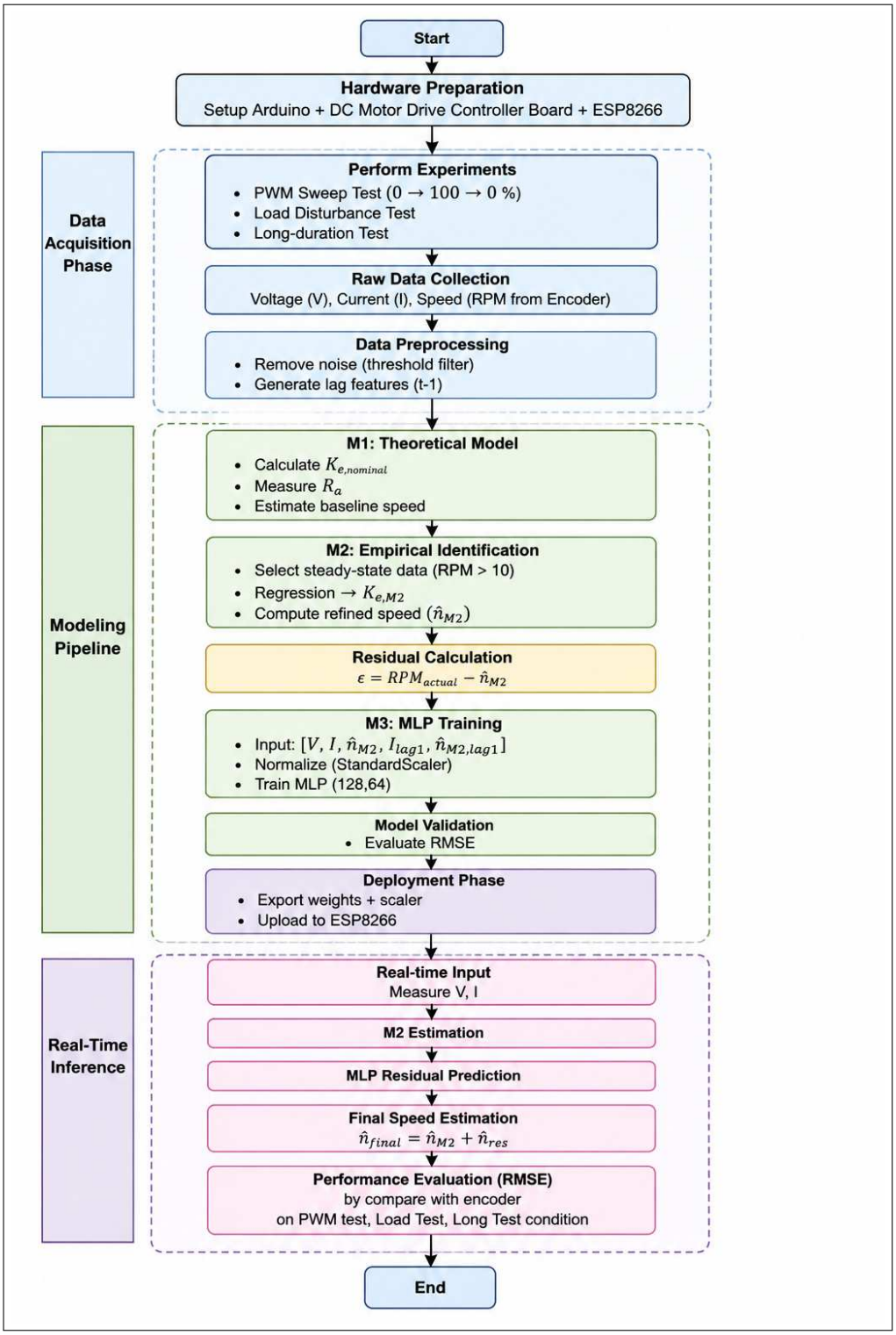

Figure 10. Methodological workflow for the hierarchical hybrid sensorless speed estimation framework.

3.6.2. Theoretical Physics (M1)

The simplest approach, Model M1, uses the motor's nameplate specifications to compute the nominal K_e as follows.

$$K_{e,nominal} = \frac{V - (I \cdot R_a)}{\omega} \tag{19}$$

3.6.3. Empirical Identification (M2)

M1 does not capture hardware-specific effects such as brush friction or contact resistance. To address this, Model M2 identifies the effective back-EMF constant $K_{e,M2}$ from steady-state data points (where $RPM > 10$) at which transient effects are minimal. The refined constant is

$$K_{e,M2} = \text{mean}\left(\frac{V - (I \cdot R_a)}{\omega_{ref} \cdot 0.10472}\right) \tag{20}$$

where 0.10472 is the conversion factor from RPM to rad/s ($2\pi/60$). This fit removes systematic bias and provides an accurate physical baseline for the M3 residual compensation.

The identified constant ($K_{e,M2} = 0.406$ V·s/rad) differs from the nameplate value ($K_e = 0.114$ V·s/rad) by a factor of approximately 3.6. This is expected and deserves explicit interpretation: because the steady-state regression maps terminal quantities directly to the shaft speed of the complete coupled assembly, $K_{e,M2}$ is a lumped effective constant that absorbs the gearbox ratio and efficiency, the brush-contact voltage drop, and speed-proportional drivetrain losses in addition to the electromagnetic back-EMF of the bare rotor. It should therefore be read as an assembly-level calibration constant rather than an estimate of the electromagnetic back-EMF coefficient of the motor alone. This interpretation also explains why M2 substantially outperforms the nameplate model M1 and why the constant must be re-identified for each motor-drivetrain assembly (see Section 4.12).

3.6.4. Residual Compensation via MLP (Model M3)

Model M3 estimates the remaining error $e = RPM_{ref} - \hat{n}_{M2}$ that the physical model cannot explain. An MLP with hidden layers of 128 and 64 neurons maps the input feature vector $\mathbf{X} = \{V, I, \hat{n}_{M2}, I_{t-1}, \hat{n}_{M2,t-1}\}$ to the residual e. All inputs are normalized using `StandardScaler` before training to ensure equal feature contribution.

3.6.5. Learning Strategy and Optimization

The Adam optimizer is used to minimize the MSE between the predicted residual $\hat{n}_{res}$ and the true residual $e = RPM_{ref} - \hat{n}_{M2}$. Training is controlled by the following settings:

- Training iterations: A maximum of 2000 iterations is used. Early stopping is disabled; the network is trained for the full iteration count.
- Data partitioning: A strictly time-series-aware chronological protocol is used: the first 80% of the time steps of every recording form the training pool, and the final 20% of each test recording is held out for evaluation. No shuffling is applied at any stage, and the one-step lag features never cross the split boundary, so no reported error is measured on data the model trained on. For comparison, results under shuffled five-fold cross-validation are additionally reported in Section 4.8; the chronological protocol yields better accuracy, confirming that the evaluation is not inflated by information leakage.
- Reproducibility: The random seed is fixed explicitly to ensure repeatability of the experimental results.

The final speed estimate is

$$\hat{n}_{final} = \hat{n}_{M2} + \mathrm{MLP}_{residual}(\mathbf{X}) \tag{21}$$

This hybrid design combines the reliability of the physical model with the MLP's ability to handle nonlinear effects.

3.6.6. Deployment and Sensorless Verification

After training, the MLP weights and `StandardScaler` parameters are converted and uploaded to the ESP8266. During the sensorless verification phase, the encoder feedback is disconnected from the ESP8266, allowing the microcontroller to operate exclusively on voltage and current measurements rather than physical encoder feedback. The encoder, which remains physically attached to the system, quietly logs the actual speed to compare against the estimated output after the test is completed. The RMSE computed from this comparison offers a clear and impartial metric for the real sensorless performance.

3.6.7. Real-Time Performance Characterization

The acquisition rate of the experimental pipeline is fixed (2 Hz, sample period 500 ms), as set in the logging scheduler of the acquisition firmware and verified against sample counts for each timed test protocol. Lastly, the real-time performance of the estimator was investigated in terms of throughput as it runs over 1000 consecutive inferences on the ESP8266 (Tensilica L106 at 80 MHz), as summarized in Table 3.

Table 3. Measured real-time characteristics of the deployed (128, 64) estimator on the ESP8266 at 80 MHz.

Metric	Value
Sampling frequency (acquisition pipeline)	2 Hz
Sampling period (cycle deadline)	500 ms
Mean single-inference latency	21.79 ms
Worst-case execution time (WCET)	22.40 ms
Execution jitter (std. dev.)	0.17 ms
Processor utilization at 2 Hz	4.4% (mean)/4.5% (worst case)
Flash usage (firmware incl. weights)	304 kB (29% of 1 MB)
RAM usage	28.1 kB (35% of 80 kB)
Network parameter footprint	35.5 kB

Because the estimator is feedforward and stateless apart from a one-sample lag feature, a computation delay τ manifests as a pure reporting latency rather than as a filtering or convergence error. With the measured worst-case latency of 22.40 ms against the 500 ms sample period, every inference completes within a small fraction of its sampling interval, leaving a worst-case timing margin of 477.6 ms (95.5% of the sampling period). During rapid transients, the additional speed error attributable to τ is bounded by $a_{\max}\,\tau$, where $a_{\max}$ is the instantaneous shaft acceleration; since the measured τ is smaller than one sample period, this contribution can never exceed the speed change between two consecutive samples. The deployed estimate is therefore at most one sample old, which is the intrinsic reporting granularity of any digital acquisition at 2 Hz rather than an additional estimation error. The compact (16, 8) variant identified in Section 4.9 reduces the multiply-accumulate count per inference by a factor of approximately 38 and provides a proportionally larger real-time margin for lower-cost deployments.

4. Results

For visual clarity, the time-series figures in this section show a representative segment of each recording; all reported RMSE values are computed over the held-out chronological test segments using the time-series-aware protocol of Section 3—not over the displayed window.

4.1. Manual Measurement of Armature Resistance (R_a)

The R_a value was measured with three multimeters, taking the average of three readings per meter. The results are listed in Table 4.

Table 4. Manual R_a measurement results.

Device No.	Manufacturer	Model	Trial 1 (Ω)	Trial 2 (Ω)	Trial 3 (Ω)	Average (Ω)
1	Pro's Kit (Prokit's Industries Co., Ltd., New Taipei City, Taiwan)	MT1210	4.52	4.50	4.51	4.51
2	UNI-T (UNI-TREND Technology (China) Co., Ltd., Dongguan, China)	UT204+	4.49	4.48	4.50	4.49
3	SUNWA (SUN-WA Technos Singapore Pte Ltd., Singapore)	YX-360TRD	4.51	4.52	4.50	4.51
Mean		-				4.50

The grand mean is 4.50 Ω, which is used as the armature resistance.

4.2. Use of Nameplate Data to Calculate Baseline Parameter

Figure 11 shows the motor nameplate: rated speed 1000 RPM, rated voltage 12 V. Under ideal steady-state assumptions, K_e for M1 follows directly as

$$V = IR_a + K_e\omega \tag{22}$$

Because the nameplate specifies neither whether the rated speed is a no-load or a loaded value nor a rated current, the listed speed is taken as the no-load speed. As described in [35], the armature voltage drop is negligible at no-load. Substituting the no-load speed ($\omega = 1000 \times \frac{2\pi}{60} \approx 104.72$ rad/s) and neglecting the armature drop IR_a, K_e is obtained as

$$K_e = \frac{V}{\omega} \approx 0.114\ \mathrm{V \cdot s/rad} \tag{23}$$

This value ($K_e = 0.114$ V·s/rad) defines the M1 model, which is the baseline benchmark that motivates the empirical identification and neural compensation steps that follow.

4.3. Dynamic Analysis and Deviation of the Nominal Model (M1) from In Situ Operating Conditions

When we executed M1 against the physical motor, an RMSE of 322.63 RPM was observed over the complete sweep recording (see Figure 12). Note that the nameplate parameters characterize the motor in an ideal laboratory environment rather than the coupled, friction-laden scenario of an actual test bench.

The gap occurs as the nameplate does not account for these losses. Rather than utilizing all input power to turn the rotor, a portion is absorbed by coupler and shaft misalignment, while generator friction and inertia increase drag even further. Combined, these mechanisms cause the effective constant $K_{e,\mathrm{eff}}$ to differ from the nominal value K_e given on the nameplate.

Figure 11. The manufacturer's nameplate of the tested DC motor.

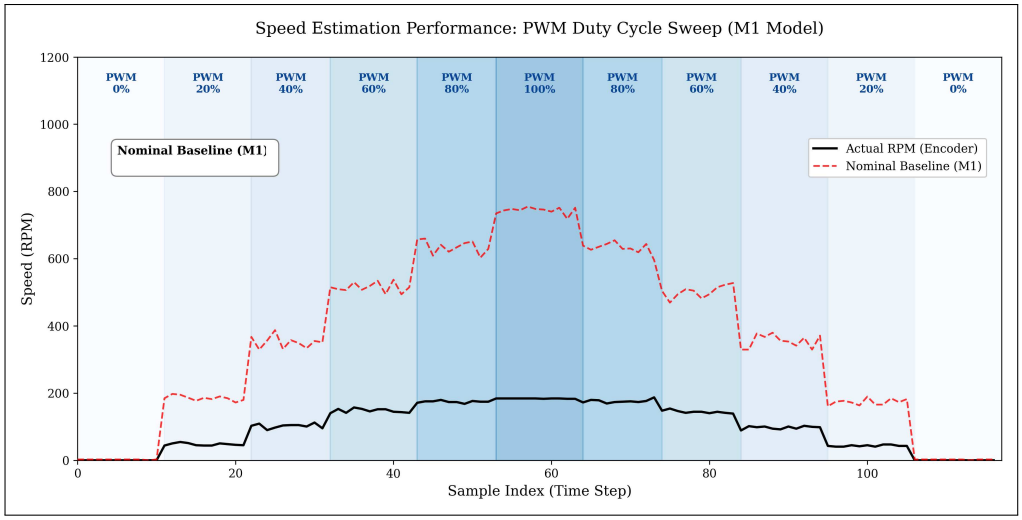

Figure 12. A comparison of speed estimation between the M1 baseline model and encoder readings during the sweep test.

Here, we define K_e as an effective parameter ($K_{e,\text{eff}}$) that is unique to each assembly and captures all associated energy losses. Estimating it requires only onboard voltage and current sensor measurements along with a single reference encoder [9,20].

Refined Estimation and Lumped Parameter Compensation (M2)

Unlike the nameplate constants, the M2 parameters are drawn directly from measured data. A parameter identification routine extracts records from every test, PWM sweep, load switching, and long-duration alike so that the fit covers the complete operating range.

The identification was made in Python 3.13 using pandas 3.0 and NumPy 2.4. The effective back EMF constant ($K_{e,eff}$) was found by rearranging the DC motor voltage equation as $\omega = (V - I_a R_a)/K_e$ and fitting it to the measured speed data. The armature resistance $R_a = 4.5\ \Omega$ from the earlier measurement was used to separate the resistive voltage drop from the back EMF. The objective function is

$$K_{e,eff} = \text{mean}\left(\frac{V - I_a R_a}{\omega_{actual} \cdot 0.10472}\right) \tag{24}$$

This regression was applied only to data points where $\omega > 10$ RPM to avoid low speed effects such as striction and quantization noise. Speed values were converted from RPM to rad/s using the factor 0.10472.

The result is $K_{e,eff} = 0.4060$ V·s/rad, which is a value that absorbs generator drag, coupler losses, and gearbox friction across the coupled assembly. Substituting it for the nameplate-derived constant lowers the steady-state tracking error and yields a more accurate physical baseline.

4.4. Empirical Identification of Steady-State Estimation Performance Improves Ke Regression (M2)

Table 5 shows the step from M1 to M2 in numbers on the held-out chronological test segments. M1's average RMSE of 346.00 RPM drops to 7.25 RPM with M2, a 97.90% reduction, by replacing the nameplate constants with values measured on the actual assembly.

Table 5. Quantitative performance comparison between M1 (nominal) and M2 (identified) models.

Condition	RMSE_{M1} (RPM)	RMSE_{M2} (RPM)	Improvement (%)
Sweep	114.46	4.57	96.01%
Load	473.80	9.57	97.98%
Long	449.74	7.62	98.31%
Average	346.00	7.25	97.90%

M1 does not factor in the actual losses. To overcome this, M2 replaces static factory specifications with parameters measured under realistic operating conditions.

Despite some optimization, there remains a small residual error across all three test conditions (Figures 13–15). This is where linear models stop: brush-commutator arcing, velocity-dependent friction [20], and thermal impedance drift [14] are nonlinear effects that simply cannot be captured with a linear model.

The model could be extended to incorporate those effects, but doing so through explicit formulations embedded in a DSP or FPGA would become prohibitively expensive [7]. The MLP residual is therefore a much leaner solution: M2 maintains a globally consistent estimate, and M3 corrects what M2 drifts away from without sensors or iterative solvers. Together, they push the RMSE lower.

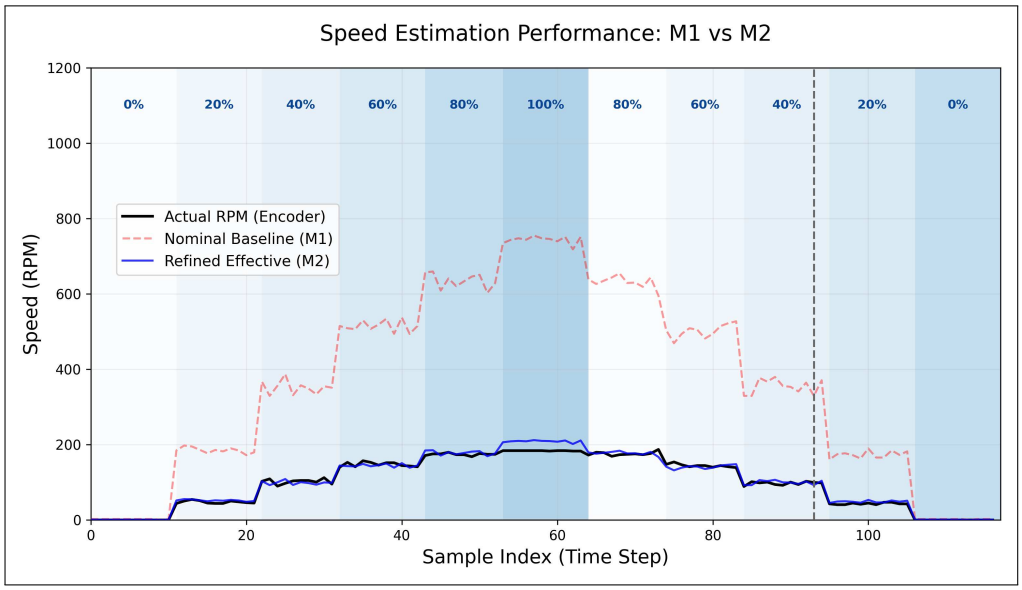

Figure 13. M2 model performance during PWM sweep (the dashed vertical line indicates the chronological 80/20 train/test split).

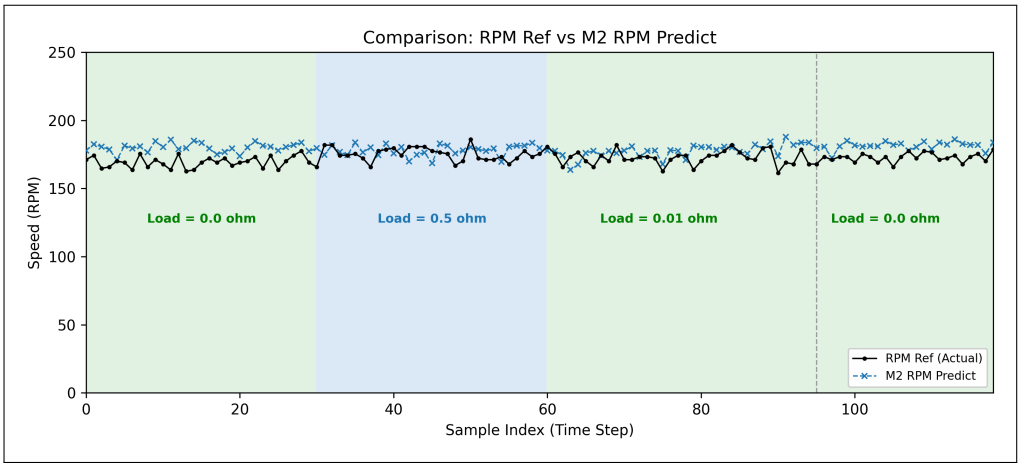

Figure 14. M2 model performance during variable load (the dashed vertical line indicates the chronological 80/20 train/test split).

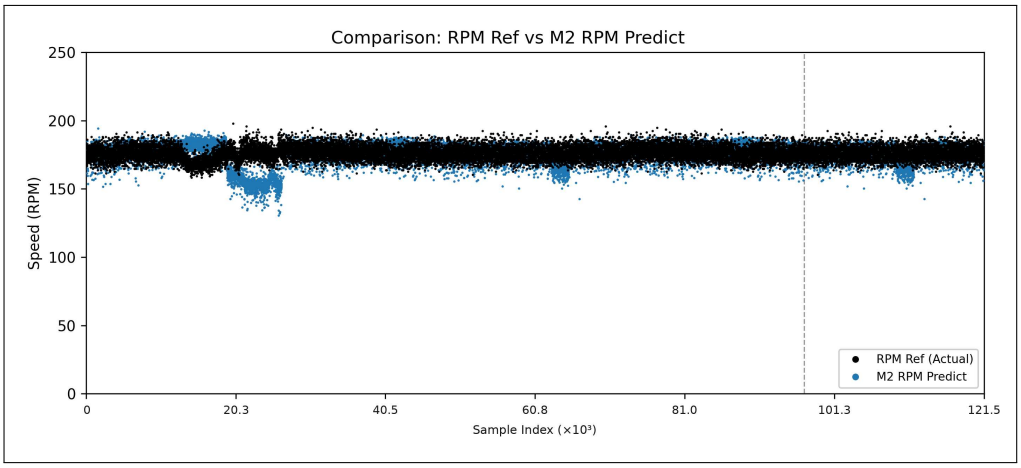

Figure 15. M2 model performance during long test (the dashed vertical line indicates the chronological 80/20 train/test split).

4.5. HPDDO Residual Learning, the M3 Hybrid Architecture

M3 sits on top of M2 and predicts only the gap $\epsilon = \omega_{actual} - \omega_{M2}$. Evaluating against this residual rather than the absolute speed simplifies training and speeds up convergence. The training of this model is based on the complete dataset available from all three tests (sweep, load switching and long test).

$$\epsilon_k = \text{MLP}(V_k,\ I_k,\ \hat{\omega}_{M2,k},\ I_{k-1},\ \hat{\omega}_{M2,k-1}) \tag{25}$$

Lagged values I_{k-1} and $\hat{\omega}_{M2,k-1}$ give the network a one-step memory of how the system was moving, which helps with friction hysteresis and current ripple at load transitions. All inputs are normalized with StandardScaler [36] before entering the (128, 64) MLP, which is trained with Adam [25].

For the ESP8266, the network weights and scaler parameters are stored as single-precision constants in flash memory. Each inference cycle runs two operations, computes $\hat{\omega}_{M2}$ from $K_{e,eff}$ and R_a, and then adds the MLP's residual ϵ_{MLP} to obtain $\omega_{M3} = \hat{\omega}_{M2} + \epsilon_{MLP}$. Both steps complete within the 500 ms sampling interval, as characterized in Section 3.6.7.

4.6. Comparative Analysis and Hybrid Model Superiority

M3 ranks first among all methods with an average RMSE of 4.11 RPM (Table 6). It cuts through the nonlinearities—thermal drift, friction, and commutation spikes—that keep M2 at 7.25 RPM.

PWM Sweep Test: This difference in performance is further demonstrated by the results of the PWM sweep shown in Figure 16. In particular, while M1 fails to follow speed accurately over that entire range, M2 follows much better but exhibits overshoot and considerable delay during large disturbances. In contrast, M3 aims to eliminate these errors through a residual correction step and generally maintains lock-on to the encoder reference at all operating points. This demonstrates that the MLP was able to capture nonlinear relationships that were not detected in M2.

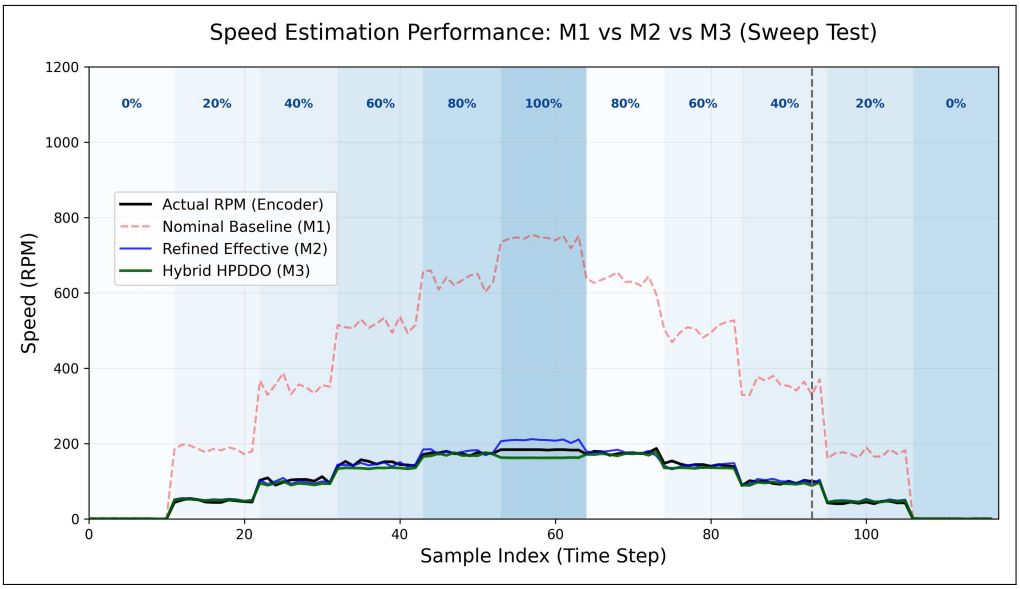

Figure 16. M1, M2 and M3 model during PWM sweep test (the dashed vertical line indicates the chronological 80/20 train/test split).

Table 6. Comparative quantitative performance of M1, M2, and M3.

Condition	$\mathbf{RMSE}_{M1}$ (RPM)	$\mathbf{RMSE}_{M2}$ (RPM)	$\mathbf{RMSE}_{M3}$ (RPM)
Sweep	114.46	4.57	4.19
Load	473.80	9.57	3.51
Long	449.74	7.62	4.63
Average	346.00	7.25	4.11

Variable Load Test: The performance is independent of the dynamic operations, as shown in the load testing results from Figure 17. Since the load varies between 0.5 Ω and 0.01 Ω, we see that these are extremely nonlinear operating points of the motor. M2 presents oscillating errors over these quick load changes, while an RMSE of 3.51 RPM is preserved for every fast generation update in the case of M3. This is evidence that the MLP has filtered out the asynchronous current ripple, which—to a classical observer—is perceived as speed noise.

Long Test: Results of the long-duration test are shown in Figure 18. Throughout the duration, M2's accuracy becomes more and more compromised by the build up of thermal drift. On the other hand, tracking by M3 remains robust over time and holds an RMSE of 4.63 RPM stably throughout the testing phase. Notably, the complete recording contains several transient disturbance events, during which the physics-only estimate M2 exhibits excursions of several tens of RPM while M3 remains substantially closer to the encoder

reference; these events provide direct visual evidence that the residual compensator absorbs disturbances that the linear physics model cannot. Even more impressively, the MLP seems to have learned how to correct for thermal drift and does not require temperature sensors at all.

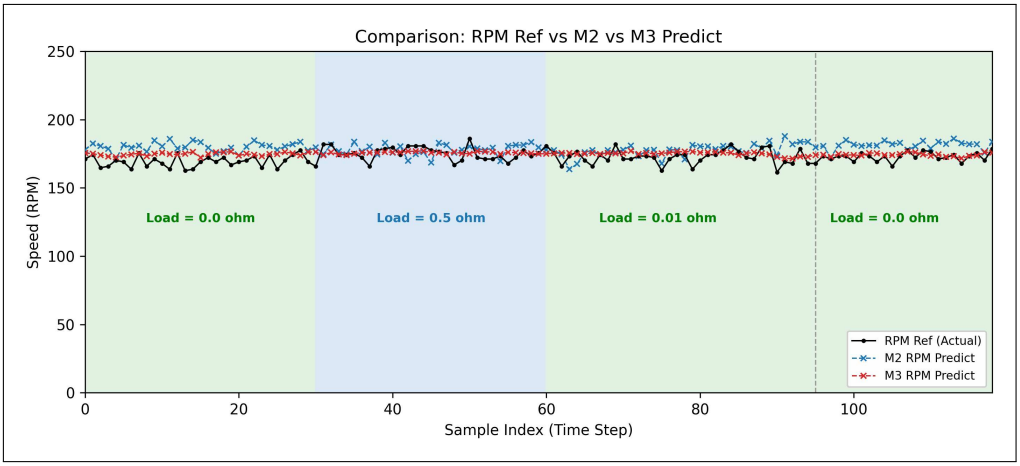

Figure 17. M1, M2 and M3 model during variable load test (the dashed vertical line indicates the chronological 80/20 train/test split).

Hybrid HPDDO: This method trained the MLP on residual error circumventing black-box limitations of purely data-driven approaches [28]. The M2 baseline guarantees that their speed estimations are physically valid. The MLP works only locally, correcting the localized nonlinear errors. Moreover, M3 removes the requirement of iterative solvers unlike complex adaptive observers [12]. Its measured single-inference latency on the ESP8266 is 21.79 ms (Section 3.6.7), which is within the 500 ms sampling period at 4.4% utilization, showing that it is practical for low-cost embedded systems.

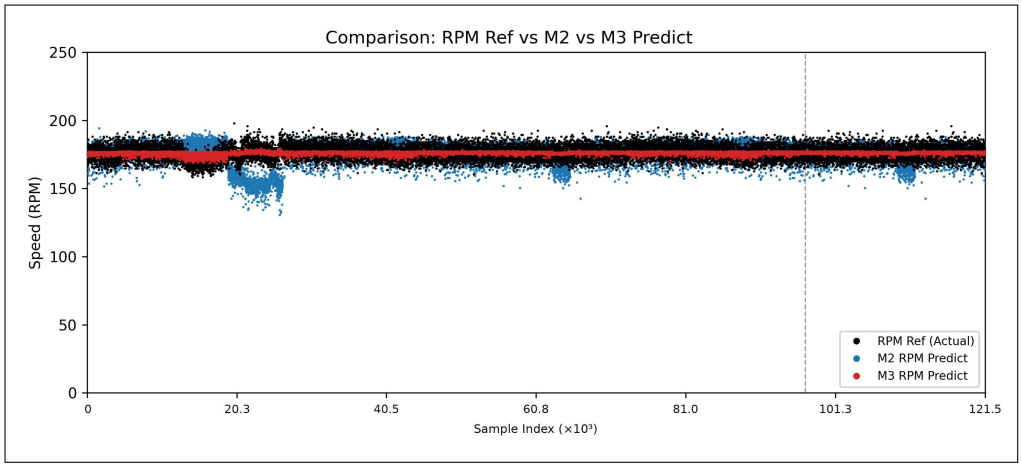

Figure 18. M1, M2 and M3 model during long test (the dashed vertical line indicates the chronological 80/20 train/test split).

4.7. Quantitative Benchmark Against Classical Observers and a Pure Data-Driven Model

To assess the proposed method against established sensorless techniques under identical experimental conditions, the extended Kalman filter (EKF) [11], a sliding mode observer (SMO) [12], and a pure data-driven neural network were implemented and evaluated on the identical dataset, the identical chronological train/test protocol of Section 3, and the identical error metric. All methods observe only the terminal voltage V and armature current I at inference time; the encoder signal is used solely for offline identification and tuning on the training portion, exactly as it is used for the proposed method.

EKF implementation. The EKF uses the standard mechanical-model formulation with state vector $[\omega, T_L]^T$, where the load torque T_L follows a random-walk model. The viscous friction coefficient ($B = 6.06 \times 10^{-3}$ N·m·s) and Coulomb friction torque ($T_c = 0.113$ N·m) were identified from the training portion by steady-state torque-balance regression, and the Coulomb term renders the process model nonlinear, requiring the extended (linearized) filter form. The measurement model is the quasi-static armature current $i = (V - K_e\omega)/R_a$. The rotor inertia and the process/measurement covariances were grid-tuned on the training portion.

SMO implementation. The SMO uses a quasi-steady-state armature-current formulation with a saturation boundary layer and a low-pass-filtered switching signal. This adaptation is necessary because the 2 Hz acquisition rate of the experimental pipeline cannot resolve the electrical time constant of the armature circuit ($\tau_e = L/R_a \ll 500$ ms), so a discrete-time current-dynamics observer would be numerically unstable at this rate. The switching gain, boundary-layer width, and filter constant were grid-tuned on the training portion.

Pure data-driven baseline. The pure neural network shares the same architecture and training configuration as the proposed residual compensator but maps (V, I, I_{k-1}, V_{k-1}) directly to speed without the physics baseline.

Two structural observations emerge from Table 7. First, the classical observers converge to approximately the accuracy of the identified physics model M2: they process the same V–I information through the same linear electromechanical relations, and therefore they cannot correct the unmodeled nonlinear residual caused by brush friction, commutation ripple, and thermal drift. Their residual error is a property of the information available to the physics—not of the observer structure. Second, the pure data-driven network is competitive in quasi-stationary conditions (Load, Long) but degrades markedly under the dynamic PWM sweep (12.88 RPM), because without the physics baseline, it must extrapolate the full input-speed mapping into transient regions sparsely covered by training data. The proposed hybrid achieves the lowest error in every condition precisely because it delegates the global mapping to physics and reserves learning capacity for the narrow residual.

Table 7. Measured RMSE (RPM) of all methods on the identical dataset and identical chronological held-out test segments.

Condition	M1	M2	EKF	SMO	Pure NN	M3 (HPDDO)
Sweep	114.46	4.57	13.30	13.55	12.88	4.19
Load	473.80	9.57	8.88	10.24	3.82	3.51
Long	449.74	7.62	6.29	8.88	4.73	4.63
Average	346.00	7.25	9.49	10.89	7.14	4.11

The EKF additionally required the tuning of four coupled covariance and inertia parameters, and the SMO exhibited the chattering-versus-lag trade-off documented in the literature [12]; the proposed method requires neither per-sample matrix operations nor switching-gain tuning at run time.

4.8. Evaluation Protocol Comparison

Two evaluation protocols are commonly applied to this class of estimator: shuffled five-fold cross-validation and a strictly chronological train/test split. Because random shuffling of a time series can, in principle, allow information from the test period to influence training, both protocols were evaluated on the identical dataset and model configuration for comparison (Table 8). The chronological protocol yields better accuracy (average 4.11 versus 5.60 RPM), indicating that the reported performance is not an artefact of information

leakage; the feed-forward feature structure, containing only one-sample lags, limits the leakage channel that random shuffling could otherwise open. All results reported in this paper follow the chronological protocol.

Table 8. M3 RMSE (RPM) under both evaluation protocols.

Condition	Shuffled 5-Fold CV	Chronological 80/20
Sweep	6.10	4.19
Load	5.84	3.51
Long	4.85	4.63
Average	5.60	4.11

4.9. Ablation Study

To assess the sensitivity of the estimation performance to network capacity and input features, an ablation study was performed along two axes under the identical chronological protocol.

Architecture axis. Eight configurations from a single eight-neuron layer to the deployed (128, 64) network were trained and evaluated (Table 9). The average RMSE varies only between 4.11 and 4.89 RPM across the entire capacity range, demonstrating that the method does not depend on delicate architecture selection and that overfitting is not observed at these capacities. Notably, a compact (16, 8) network with only 241 parameters (0.9 kB) matches the deployed network exactly at 4.11 RPM average — a 38× memory reduction at equal accuracy, which further strengthens the resource-constrained deployment argument.

Table 9. Architecture ablation: RMSE (RPM) on held-out chronological test segments with parameter count and memory footprint.

Hidden Layers	Params	Memory (kB)	Sweep	Load	Long	Average
(8)	57	0.2	5.82	4.23	4.62	4.89
(16)	113	0.4	4.36	4.01	4.62	4.33
(32)	225	0.9	4.69	3.94	4.62	4.42
(64)	449	1.8	4.19	3.83	4.62	4.21
(16, 8)	241	0.9	3.88	3.84	4.62	4.11
(32, 16)	737	2.9	4.90	3.92	4.62	4.48
(64, 32)	2497	9.8	4.24	3.87	4.62	4.24
(128, 64)	9089	35.5	4.19	3.51	4.63	4.11

Feature axis. With the deployed architecture fixed, Table 10 isolates the contribution of each input group. Removing the physics feature $\hat{n}_{M2}$ degrades the dynamic sweep condition from 4.19 to 7.53 RPM while quasi-stationary conditions are barely affected—this is consistent with the independent pure-NN benchmark result of Section 4.7, and it confirms that the physics baseline is the critical component for dynamic operating conditions. Removing the one-step lag features causes a milder overall degradation (4.11 to 4.40 RPM average), which is attributable to the loss of short-term memory at load transitions.

Table 10. Feature ablation with the (128, 64) architecture: RMSE (RPM).

Feature Set	Sweep	Load	Long	Average
Full set (5 features)	4.19	3.51	4.63	4.11
Without lag features (3)	4.30	4.29	4.61	4.40
Without physics feature $\hat{n}_{M2}$ (4)	7.53	3.94	4.64	5.37

4.10. Robustness Under Unseen Operating Conditions and Runtime Safeguards

The residual compensator is a learned component, and its output is not intrinsically constrained when inputs fall outside the training distribution. To characterize this failure mode and its mitigation, a deliberately harder out-of-distribution stress test was constructed: the compensator was trained only on the dedicated training recording and evaluated on the entire unseen test recordings.

The unconstrained residual severely degraded the dynamic sweep condition performance to 13.95 RPM—which was worse than the physics-only baseline of 9.81 RPM—while both quasi-stationary conditions remained accurate (Load 5.76 RPM, Long 5.02 RPM). This is exactly the out-of-distribution failure mode we would expect: the network extrapolates its correction into regions it has never seen.

Two runtime safeguards are introduced. First, a residual clamp: the compensation term is clipped at the 99th percentile of absolute training residuals ($\pm$20.9 RPM)—a bound based on training data alone with no test-set information. The clamp returns the stress-test sweep condition to 10.32 RPM and improves the average error from 8.24 to 7.08 RPM with no in-distribution performance trade-offs (0% of samples clamped in the load condition; 4.5% on average). Second, a physics fallback: when any input falls outside the training envelope, the estimate resets back to M2—the pure physics baseline—so that the estimator degrades gracefully to a physically valid value rather than failing unpredictably.

The clamp additionally yields an explicit error bound. Because the estimator is feedforward, the estimate at step k depends only on current and one-step-lagged measurements, never on previous estimates; no error propagation mechanism exists by which the correction could accumulate drift over time, and with the clamp active, the total estimation error is bounded by construction:

$$|\hat{n}_{M3} - n| \;\leq\; |\hat{n}_{M2} - n| + \epsilon_{clamp}, \qquad \epsilon_{clamp} = 20.9 \text{ RPM}. \tag{26}$$

The long duration recording confirms the absence of drift empirically (RMSE 4.63 RPM with no degradation trend). A formal closed-loop stability analysis for control applications is identified as future work. The present contribution targets open-loop monitoring and assessment.

4.11. Measurement Offset Considerations

A constant or slowly varying DC offset in the V or I measurement chain produces a systematic, operating point dependent error in any physics-based estimate, and conventional SMO structures must reject it through additional cascaded filter stages and enhanced PLL architectures [16]. The hybrid framework offers a complementary structural pathway: because the residual compensator is trained on measurements from the same acquisition chain used at deployment, any offset present during both training and deployment is absorbed into the learned residual mapping rather than propagating into the speed estimate. The limit of this property is an offset arising after training (for example, sensor drift over the device lifetime), which would require recalibration or the adaptive filtering techniques of the cited work—an interesting hybridization opportunity noted as future work.

For broader contextual reference, Gamazo Real et al. [37] reported a speed estimation mean absolute error (MAE) below 22 RPM for a brushless DC (BLDC) motor using an ANN deployed on a Xilinx Spartan 6 FPGA platform. The proposed M3 framework achieves an RMSE of 3.51–4.63 RPM on a PMDC motor, which is a stricter metric than MAE, since RMSE penalizes large transient deviations more heavily while running on an ESP8266 microcontroller whose hardware cost is far lower than that of an FPGA solution. Although a direct numerical comparison is constrained by differences in motor type (brushed PMDC vs. brushless BLDC), error metric (RMSE vs. MAE), and target platform, this shows that the

proposed hybrid framework achieves competitive sensorless estimation accuracy without requiring specialized DSP or FPGA hardware.

4.12. Transferability and Deployment Considerations

The framework requires a one-time offline calibration per motor-drivetrain assembly: (i) measurement of the armature resistance R_a, and (ii) identification of the effective back-EMF constant and residual model from a short labeled recording. Both steps are performed once per assembly and require only a short labeled recording covering the intended operating range after which the encoder is removed from the inference path. The identification itself is a closed-form regression followed by a compact network fit on a small dataset, so the procedure runs offline on a standard desktop computer and adds no run-time cost; the calibration effort is therefore modest and comparable to the commissioning procedures already common for industrial drives.

The two-layer structure of the framework decouples what must be recalibrated from what may transfer. The physics layer (R_a, $K_{e,M2}$) is assembly-specific: it lumps the gearbox, brush interface, and drivetrain losses of the particular unit (Section 3) and must be re-identified per assembly. The residual layer learns correction patterns whose physical causes—brush friction, thermally induced resistance drift, and commutation ripple—are common to the PMDC motor class; partial transfer of the residual compensator across similar motors via short fine-tuning recordings is therefore plausible with assembly-specific components (gearbox-dependent friction) expected to dominate the required adaptation. We deliberately refrain from claiming measured cross-motor performance, which has not yet been evaluated; cross-motor and cross-load validation is identified as future work.

5. Conclusions and Discussion

This paper proposed and validated a hybrid speed estimation framework for PMDC motor drives integrating a physics-based lumped parameter model (M2) with an HPDDO-based neural residual compensator (M3). Experimental results under a strictly time-series-aware chronological evaluation demonstrate that M3 achieves an average RMSE of 4.11 RPM, marking a significant improvement over both the nominal baseline M1 (346.00 RPM) and the refined physical model M2 (7.25 RPM), and the lowest error among all benchmarked methods—EKF (9.49 RPM), SMO (10.89 RPM), and a pure neural-network estimator (7.14 RPM)—evaluated on the identical dataset. Furthermore, M3 delivers consistent performance across all three conditions—PWM sweep, variable load, and long test—confirming its robustness across a range of operating conditions.

Moreover, this performance now rests on a direct quantitative comparison: the EKF [11] and SMO [12] were implemented on the identical dataset and were outperformed in every operating condition, while M3 requires neither covariance tuning nor per-sample matrix operations. In addition, an ANN-based sensorless method for BLDC motors [37] reported a speed MAE below 22 RPM using an FPGA platform, whereas M3 achieves an RMSE between 3.51 and 4.63 RPM while running on a simple ESP8266 microcontroller. In the proposed system, an Arduino UNO handles high-speed signal acquisition while the ESP8266 performs real-time HPDDO inference. Because the ESP8266 is far more affordable than a DSP or FPGA, the resulting system is highly cost-effective. Beyond estimating speed, the residual-learning structure of the HPDDO also makes it a scalable baseline for future extensions: diagnostic modules for detecting mechanical failures or localized electrical faults could be added later and run on the fly. The framework is therefore well suited to low-cost sensorless DC motor drives in industrial applications.

Limitations of this paper should be mentioned. Initially, the models were validated on one PMDC motor in a controlled laboratory setting. Thus, the conclusion regarding its

generalization to a different motor rating or highly aggressive thermal environment is still speculative. Second, the model may need to be re-identified when deployed on motors with different winding resistance or inertia in each motor due to training data being collected from a single test bench setup. Third, the EKF and SMO benchmarks were implemented in software and evaluated on the identical dataset (Section 4.7).

Author Contributions: Conceptualization, M.P. and D.M.; methodology, M.P.; software, M.P.; validation, M.P.; formal analysis, M.P.; investigation, M.P.; writing—original draft preparation, M.P.; writing—review and editing, D.M. and P.S.; supervision, D.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: During the preparation of this manuscript, the authors used Google Gemini (Gemini 3.5 Flash) to draft the initial layouts of the conceptual diagrams in Figures 1–6 and 10, which the authors subsequently edited and finalized, and Anthropic Claude (Opus 4.8) for language editing (grammar and phrasing). The authors have reviewed and edited all outputs and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this article:

ADC	Analog-to-Digital Converter
ANN	Artificial Neural Network
Back-EMF	Back Electromotive Force
BLDC	Brushless Direct Current
DC	Direct Current
DMM	Digital Multimeter
DSP	Digital Signal Processor
EKF	Extended Kalman Filter
ESP	Microcontroller series
FPGA	Field-Programmable Gate Array
HPDDO	Hybrid Physics-Data-Driven Observer
KVL	Kirchhoff's Voltage Law
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MLP	Multilayer Perceptron
MPC	Model Predictive Control
MSE	Mean Squared Error
PMDC	Permanent Magnet Direct Current
PWM	Pulse-Width Modulation
RMSE	Root Mean Squared Error
SMO	Sliding Mode Observer

Appendix A. Bill of Materials (BOM) of the Experimental Test Bench

The following table provides the comprehensive Bill of Materials (BOM) for the experimental test bench, detailing all 26 hardware components utilized. Costs are estimated based on global average market prices for electronic components.

Table A1. Comprehensive 26-item Bill of Materials (BOM) for the test bench assembly.

ID	Component	Part Number/Specification	Qty	Est. Cost (USD)
1	Arduino	UNO R3	1	7.14
2	ESP8266	WeMos NodeMCU	1	3.43
3	MOSFET	IRFZ24NPBF	1	0.57
4	Opto coupler	TLP250H(F)	1	0.40
5	R shunt	0.5 ohm, 10 W	1	0.71
6	Motor	25GA371, Ratio 1:4.4	1	5.14
7	Generator	25GA370, Ratio 1:171	1	5.14
8	Coupler	4 mm to 4 mm	1	1.20
9	5V Supply	LM7805	1	0.29
10	Temp. Sensor	DS18B20	1	0.85
11	Relay Module	Isolated Opto Drive	1	1.43
12	Dummy Load	0.5 R, 10 W	1	0.71
13	Dummy Load	0.01 ohm, 10 W	1	0.71
14	Diode	MBR60100CT	1	0.57
15	Capacitor	100 uF	3	0.30
16	Capacitor	0.1 uF	3	0.15
17	Capacitor	1000 uF	1	0.30
18	Resistor	10 k	4	0.20
19	Resistor	15 k	1	0.05
20	Resistor	220 R	1	0.05
21	Resistor	20 R	1	0.05
22	Resistor	2 k	2	0.10
23	Resistor	1 k	2	0.10
24	Resistor	4 k7	1	0.05
25	PCB	Prototype Board	1	0.85
26	Connector	2 Pins, 10 A	5	0.75
Total				31.24 USD

References

1. Chapman, S. *Electric Machinery Fundamentals*, 5th ed.; McGraw-Hill Education: New York, NY, USA, 2012.
2. Boldea, I.; Tutelea, L. *Electric Machines: Steady State, Transients, and Design with MATLAB*, 2nd ed.; CRC Press: Boca Raton, FL, USA, 2017.
3. Fitzgerald, A.; Kingsley, C.; Umans, S. *Electric Machinery*, 6th ed.; McGraw-Hill: New York, NY, USA, 2003.
4. Sen, P. *Principles of Electric Machines and Power Electronics*, 3rd ed.; John Wiley & Sons: Hoboken, NJ, USA, 2013.
5. Zhao, S.; Blaabjerg, F.; Wang, H. An overview of artificial intelligence applications for power electronics. *IEEE Trans. Power Electron.* **2021**, *36*, 4633–4658. [CrossRef]
6. Mohan, H.; Pathak, M.; Dwivedi, S. Sensorless control of electric drives—A technological review. *IETE Tech. Rev.* **2020**, *37*, 504–528. [CrossRef]
7. Pacas, M. Sensorless drives in industrial applications. *IEEE Ind. Electron. Mag.* **2011**, *5*, 16–23. [CrossRef]
8. Xu, D.; Wang, B.; Zhang, G.; Wang, G.; Yu, Y. A review of sensorless control methods for AC motor drives. *CES Trans. Electr. Mach. Syst.* **2018**, *2*, 104–115. [CrossRef]
9. Wang, P.; Zhu, Z. Overview of online parameter identification of permanent magnet synchronous machines under sensorless control. *IEEE Access* **2026**, *14*, 11582–11606. [CrossRef]
10. Suwankawin, S.; Sangwongwanich, S. Design strategy of an adaptive full-order observer for speed-sensorless induction-motor Drives-tracking performance and stabilization. *IEEE Trans. Ind. Electron.* **2006**, *53*, 96–119. [CrossRef]
11. Zerdali, E. Adaptive extended Kalman filter for speed-sensorless control of induction motors. *IEEE Trans. Energy Convers.* **2019**, *34*, 789–800. [CrossRef]
12. Zuo, Y.; Lai, C.; Iyer, K. A review of sliding mode observer based sensorless control methods for PMSM drive. *IEEE Trans. Power Electron.* **2023**, *38*, 11352–11367. [CrossRef]
13. An, X.; Yao, Q.; Wang, S.; Chen, Q. Robustness sensorless control strategy for PMSM based on MPC with multi-parameter estimation. In Proceedings of the 2024 IEEE International Conference on Electrical Energy Conversion Systems and Control (IEECSC), Wuhan, China, 25–27 October 2024; pp. 86–91. [CrossRef]

14. Ali, S.M.N.; Hossain, M.J.; Wang, D.; Lu, K.; Rasmussen, P.O.; Sharma, V.; Kashif, M. Robust Sensorless Control Against Thermally Degraded Speed Performance in an IM Drive Based Electric Vehicle. *IEEE Trans. Energy Convers.* **2020**, *35*, 896–907. [CrossRef]
15. Mohamed, A.; Zaky, M.; El-Din, A.; Yasin, H. Comparative study of sensorless control methods of PMSM drives. *Innov. Syst. Des. Eng.* **2011**, *2*, 44–66.
16. Wu, X.; Wu, B.; Chen, S.; Ding, W.; Ma, Z.; Han, Y. An Anti-DC Offset Interference Sensorless Control Method for PMSM Based on Improved SMO and MDCF-FPS-PLL. *IEEE J. Emerg. Sel. Top. Power Electron.* **2026**, *14*, 555–568. [CrossRef]
17. Piippo, A.; Hinkkanen, M.; Luomi, J. Adaptation of motor parameters in sensorless PMSM drives. *IEEE Trans. Ind. Appl.* **2009**, *45*, 203–212. [CrossRef]
18. Bui, M.; Dutta, R.; Rahman, F. Application of deep learning in parameter estimation of permanent magnet synchronous machines. *IEEE Access* **2024**, *12*, 40710–40721. [CrossRef]
19. Awadallah, M.; Morcos, M. Application of AI tools in fault diagnosis of electrical machines and drives—An overview. *IEEE Trans. Energy Convers.* **2003**, *18*, 245–251. [CrossRef]
20. Buechner, S.; Schreiber, V.; Amthor, A.; Ament, C.; Eichhorn, M. Nonlinear modeling and identification of a dc-motor with friction and cogging. In Proceedings of the IECON 2013-39th Annual Conference of the IEEE Industrial Electronics Society, Vienna, Austria, 10–13 November 2013; pp. 3621–3627. [CrossRef]
21. Tom, A.; Daya, J. Design of machine learning-based controllers for speed control of PMSM drive. *Sci. Rep.* **2025**, *15*, 17826. [CrossRef] [PubMed]
22. Arifeen, S. AI-driven condition monitoring and fault detection in electrical power and industrial control systems. *ASRC Procedia Glob. Perspect. Sci. Scholarsh.* **2025**, *1*, 1778–1809. [CrossRef]
23. Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **1989**, *2*, 359–366. [CrossRef]
24. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
25. Kingma, D.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980. . [CrossRef]
26. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on Machine Learning (ICML 2015), Lille, France, 6–11 July 2015; pp. 448–456.
27. Li, Q.; Zhang, B.; He, H.; Wang, Y.; He, D.; Mo, S. A hybrid physics-data-driven approach for vehicle dynamics state estimation. *Mech. Syst. Signal Process.* **2025**, *225*, 112249. [CrossRef]
28. Nguyen, D.; Nguyen, T.; Tran, K.; Tran, K. Physics-informed machine learning for industrial reliability and safety engineering: A review and perspective. In *Artificial Intelligence for Safety and Reliability Engineering: Methods, Applications, and Challenges*; Tran, K., Ed.; Springer Nature: Cham, Switzerland, 2024; pp. 5–23. [CrossRef]
29. Haykin, S. *Neural Networks and Learning Machines*, 3rd ed.; Pearson: Upper Saddle River, NJ, USA, 2009.
30. Chai, T.; Draxler, R. Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature. *Geosci. Model Dev.* **2014**, *7*, 1247–1250. [CrossRef]
31. Mantala, C. Sensorless Control of Brushless Permanent Magnet Motors. Ph.D. Thesis, University of Bolton, Bolton, UK, 2013.
32. Liu, J.; Zhang, W.; Tao, G.; Huang, T. Data-driven friction compensation depending on modified disturbance observer for ball screw feed-drive system. In Proceedings of the 2023 International Conference on Robotics, Control and Vision Engineering (RCVE 2023), Chengdu, China, 25–27 August 2023; pp. 25–29. [CrossRef]
33. Bishop, C. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006.
34. Willmott, C.; Matsuura, K. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Clim. Res.* **2005**, *30*, 79–82. [CrossRef]
35. Smith, K. CHAPTER 3-Direct-Current Machines. In *Electrical Engineering Principles for Technicians*; The Commonwealth and International Library: Electrical Engineering Division: Pergamon, Turkey, 1970; pp. 50–113. [CrossRef]
36. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
37. Gamazo-Real, J.C.; Martínez-Martínez, V.; Gomez-Gil, J. ANN-based position and speed sensorless estimation for BLDC motors. *Measurement* **2022**, *188*, 110602. [CrossRef]