IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 31 March 2024, accepted 10 May 2024, date of publication 20 May 2024, date of current version 31 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3403071

RESEARCH ARTICLE

Real-Time Validation of Enhanced Permanent Magnet Synchronous Motor Drive Using Dense-Neural-Network-Based Control

ARMITA FATEMIMOGHADAM, (Graduate Student Member, IEEE), LAKSHMI VARAHA IYER, (Senior Member, IEEE), AND NARAYAN C. KAR, (Senior Member, IEEE)
Electrical Engineering Department, University of Windsor, Windsor, ON N9A 3P4, Canada

Corresponding author: Armita Fatemimoghadam (fatemim@uwindsor.ca)

This work was supported in part by the Canada Research Chair (CRC) in Electrified Vehicles Program under Reference CRC-2019-00319, and in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

ABSTRACT High-performance current and speed control are required to obtain smooth output torque, current tracking, and speed tracking in permanent-magnet synchronous motor (PMSM) drives. The motor speed and stator current control rely on multiple nonlinear motor parameters, which play a crucial role in shaping the performance of PMSM. Moreover, tuning the speed and current controller parameters using the conventional control technique depends on these PMSM parameters, also variation of these parameters will have a decisive influence on the dynamic performance of PMSM. To enhance the robustness of vector control and tracking methodology against PMSM parameter uncertainties and load disturbances, a novel artificial intelligence (AI)-based advanced speed and current control technique for PMSM is proposed in this article. Subsequently, the methodology for designing and training the suggested Dense Neural Network (DNN) controllers are elicited. The proposed controllers can handle the inevitable fluctuation and non-linearity in motor parameters at different load points and drive conditions. The proposed DNN scheme is validated in terms of settling time, dynamic responsiveness, tolerance to parameter fluctuations, and overall robustness. A comparative analysis is conducted against adaptive proportional-integral (API) control applied to the same PMSM within the OPAL-RT real-time simulator (RTS). The viability of the proposed control scheme is substantiated through simulation, Software-In-the-Loop (SIL) and Hardware-In-the-Loop (HIL) testing with an RTS and an automotive-grade controller board across diverse conditions.

INDEX TERMS Dense neural network, hardware-in-the-loop, motor drive, permanent magnet synchronous motor, proportional-integral, software-in-the-loop, vector control.

ABBREVIATIONS

ADC	Analog to Digital Converter.
AI	Artificial Intelligence.
ANN	Artificial Neural Network.
API	Adaptive Proportional-Integral.
CAN	Controller Area Network.
DNN	Dense Neural Network.
DRL	Deep Reinforcement Learning.
DTC	Direct Torque Control.
FLC	Fuzzy Logic Controller.
FOC	Field Oriented Control.
FPGA	Field Programmable Gate Array.
HIL	Hardware-In-the-Loop.
IGBT	Insulated Gate Bipolar Transistor.
LM	Levenberg-Marquardt.
MFC	Model Free Control.
MPC	Model Predictive Control.
MSE	Mean Squared Error.
NFC	Neuro-Fuzzy Controller.
NN	Neural Network.
PI	Proportional-Integral.
PID	Proportional-Integral-Derivative.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhuang Xu.

PMSM	Permanent Magnet Synchronous Machine.
PR	Proportional-Resonant.
PWM	Pulse Width Modulation.
RTS	Real-Time Simulator.
SIL	Software-In-the-Loop.
SMO	Sliding Mode Observer.
VSI	Voltage Source Inverter.

I. INTRODUCTION

Permanent magnet synchronous machines (PMSMs) are widely used in the fields of automotive and robotics owing to their remarkable efficiency, high power density, and minimal weight [1], [2]. However, the variability of system parameters and the presence of unmodeled dynamics, particularly from inverters and state-dependent disturbances, present significant challenges in PMSM drive applications. To counter these uncertainties, various adaptive control strategies has been devised. Among these, techniques such as neural network-based control [1], proportional-integral (PI) with proportional-resonant (PR) control [2], model predictive control (MPC) [3], direct torque control (DTC) [4], fuzzy logic controllers (FLC) [5], [6], deep reinforcement learning (DRL)-based control [7], and mixed H2/H∞ control [8], [9], [10] have emerged. Despite their innovative approaches, each method encounters specific obstacles, highlighting the ongoing struggle to balance novelty, efficiency, and practicality in control system design.

In PI-PR controllers, owing to the need for precise parameter tuning the performance of the PI-PR controller is adversely affected whenever there is a change in the values of the machine parameters [2]. In the MPC controller [3], a current controller estimates the machine current and uses a control equation to determine the appropriate action. When the machine parameters deviate from the tuned values used in the current or speed controllers, despite their rapid current monitoring capabilities, instability may arise in the system [11], [12]. For DTC approach, [4], the speed of the motor can be adjusted only after the torque has been controlled. Significant torque ripple using this approach, as well as the need for variable switching frequency, high sampling rate, and digital implementation complexity, would degrade the overall performance of the control system. The FLC, designed to mitigate torque harmonics, demand an in-depth understanding of *if-then* rules and membership functions, complicating their design [5], [6].

To mitigate the impact of disturbances, [7] suggests a speed and current controller based on DRL. The complexity and computational requirements associated with DRL, however, are significant drawbacks. In the mixed H2/H∞ control a detailed model of the system is required [8], [9], [10], and the controller in question lacks the ability to deal with nonlinear constraints efficiently [9]. It is also highly challenging to experimentally implement them [10]. Furthermore, the integration of fuzzy logic with proportional-integral-derivative (PID) control in [13] overlooks the inherent nonlinear characteristics of PMSMs, potentially compromising real-time control accuracy [14].

Another class of advanced PMSM controllers are sliding mode observer (SMO)-based controllers [15], [16], [17]. The SMO structure is considered as one of the most popular methods specifically for sensorless applications due to its robustness and straightforward implementation. However, the SMO-based control methods for PMSM control applications include high computational cost, and limitations in low-speed and very-high-speed operations. The performance degrades in the low-speed region, especially in the presence of inverter nonlinearity, and also degrades in the very-high-speed region due to chattering and phase delay caused by the low-pass filter.

In [18], an adaptive fuzzy-based cooperative control strategy is proposed and it theoretically ensures finite-time stability in the presence of uncertainties and saturations simultaneously. While this method shows effectiveness under the presented scenarios, it lacks parameter tuning rules which could be a limiting factor for its real-time and practical applications. In [19], a robust resonant deadbeat predictive current control (RRDPCC) is proposed. This method effectively suppresses both non-periodic and periodic disturbances, providing stable current tracking and enhanced performance in PMSMs. On the downside, this approach may face difficulty in maintaining iteration accuracy during the dynamic state, and the update law for the iterative learning controller needs to be carefully selected to ensure convergence. Additionally, the work proposed in [20] discusses a deep learning-based fault diagnosis method for PMSM drive systems. In addition, a fuzzy PID control method for PMSM is proposed in [21]. Although the proposed methods enhance the reliability and accuracy of the overall drive system, they will significantly add to the computational burden of the system.

In [22], an improved model reference adaptive system (MRAS) observer for flux linkage observation in PMSMs is discussed. The method boasts enhanced dynamic response, accuracy, and convergence speed of the observer, however, the presented scheme has a relatively high reliance on accurate machine parameter identification, and possibly increased computational requirements for real-time adaptation and control.

The artificial neural network (ANN)-based control is independent of the mathematical model of the controlled object and can mitigate the effects of disturbances and changing parameters in the control system. In [23], a real-time adaptive controller based on ANN is proposed, but the computation of the training algorithm should be optimized for the required control action to mitigate the chance of hampered control performance due to insufficient generalization [24]. Despite the benefits of the intelligent control approaches discussed, the majority of these systems necessitate a significant computation. Also, given the scarcity of accessible research on intelligent-based control for PMSM drives, there is always a demand for a more dependable controller for these favorable machines.

TABLE 1. Summary of AI-Based PMSM control methods.

Method	Advantages	Disadvantages	Reference
DRL	Can learn optimal control policies through trial and error	Requires significant training data and time	[7], [25]
MFC	Does not require a precise model of the motor	Learning convergence can be very slow	[26], [27]
NFC	Combines the strengths of neural networks and fuzzy logic	Complexity in designing and tuning fuzzy rules	[5], [14]
NN	High adaptability and handling complex systems	Pre-training can be computationally intensive	[23]
DNN	Demonstrate improved performance and accurate control action	Generalization deficiencies should be prevented	[28]

In light of these challenges, this research proposes two novel dense neural networks (DNN)-based controllers for use in the drive system of a PMSM that are both based on vector control. In the proposed control strategy, DNNs are used in lieu of adaptive PI (API) controllers. Unlike conventional control approaches, the proposed method does not explicitly depend on the inverter and PMSM models. The DNN structure is adopted in the paper as it does not have the structural complexity of other advanced intelligent controllers, while demonstrating accurate control actions. Thus, using DNN would provide improved performance while minimizing the computational cost associated with the motor control.

In Table 1, an overview of the current intelligent control approach for PMSM drives is presented. This table compare the proposed controller with DRL, model free control (MFC), neuro-fuzzy controller (NFC), and neural network (NN). Based on the presented results and drawbacks of the other intelligent control approaches, the presented DNN-based control method is selected to be used for this study. The distinction of the DNN-based approach lies in its adaptability and efficiency, providing a practical solution to the intricate dynamics of PMSM drives.

The implementation of DNN-based methodologies for the control of PMSMs presents several challenges, including the requisite for data for model training, the necessity to achieve real-time operational speeds, and the establishment of rigorous training and testing protocols.

Despite these hurdles, the transition towards DNN-based control systems is justified as DNNs exhibit superior accuracy, possess the capacity to adapt to changes within the system dynamically, and demonstrate enhanced proficiency in managing complex operational scenarios compared to conventional control methods. Upon successful integration, these systems are capable of executing rapid decision-making processes and maintaining consistent performance.

To ensure the robustness and efficiency of the proposed DNN-based control method, multiple approaches using both empirical testing and theoretical considerations of formal verification techniques have been studied. Formal validation methods each have their own strengths and challenges. Model checking automates checking all states to find errors like deadlocks, but struggles with large, complex systems [29]. Abstract interpretation also offers a sound method for static analysis without execution, but may yield false positives due to over-approximation and involves complex design for precise abstract domains [30]. Moreover, deductive verification allows for the proving of algorithm correctness relative to formal specifications, ideal for critical systems, yet it is hindered by the labor-intensive nature of writing specifications [31]. The exploration of various intelligent control methods, including MPC, FLC, and DRL, identified advantages and limitations of each method, such as computational demands and complex model knowledge requirements. The selection of DNNs for current and speed control was based on their relative computational efficiency and robustness against parameter variations. This choice was validated through comparative analysis, demonstrating superior DNN performance in settling time, dynamic responsiveness, and parameter fluctuation tolerance. This validation underscores the departure from traditional control strategies, emphasizing the capability of DNN to manage nonlinear dynamics and adapt to changing conditions effectively. Empirical validations through Software-In-the-Loop (SIL) and Hardware-In-the-Loop (HIL) testing highlighted the computational efficiency and adaptability of DNN, reinforcing the decision to leverage DNNs over other intelligent control methods like MPC, FLC, and DRL. While direct application of formal methods like statistical model checking to DNNs presents challenges, the comprehensive approach adopted in this research signifies a forward-looking strategy to ensure the reliability and efficiency of advanced control systems.

The proposed DNNs are trained offline and then deployed for real-time control. This can be advantageous in applications where low-latency control is critical. In addition, the proposed approach can provide continuous control output, making them well-suited for applications that require fine-grained control of PMSM systems. Also, DNNs can be computationally more efficient than DRL or NFC methods, especially if the training is offline. In this regard, the contributions of this paper are explicitly listed as follows:

1- This work is the first attempt at current and speed control using DNN-based controllers. In comparison with the existing methods, the proposed scheme provides an enhanced performance and a superior speed transient response.
2- The control law of the proposed DNN controller is obtained in an offline manner, which significantly improves the computational efficiency of the whole e-drive system in comparison to online approaches. It is also experimentally shown that the implementation of the proposed DNN controller becomes as simple as conventional PI controllers.
3- DNN controllers can be designed to be less sensitive to variations in PMSM parameters, leading to more

stable and reliable performance across varying operating conditions.

4- Two test setups have been used for validation. First, the control approach is executed on a real-time simulator in a SIL test setup to evaluate its real-time performance. Second, an automotive-grade controller board is used to validate the standalone performance of the DNN-based controllers, and check their controller area network (CAN) communication to the real-time system in a HIL test setup.

II. ANALYSIS OF API CONTROLLER FOR PMSM DRIVE

A. PMSM MODEL INTERFACE

The dynamic equations of the stator currents of PMSM for the purpose of the simulation are constructed in the rotor reference frame. Within the dynamic model the core loss is assumed to be negligible. Based on this, the PMSM model is constructed using the dynamic equations of the motor shown in (1) to (3) [32].

$$\tau_e = \tau_L + J\dot{\omega} + \omega B \tag{1}$$

$$\left.\begin{aligned} \dot{\psi}_d &= V_d - R_s i_d + \omega\psi_q \\ \dot{\psi}_q &= V_q - R_s i_q - \omega\psi_d \end{aligned}\right\} \tag{2}$$

$$\left.\begin{aligned} i_d &= \frac{\psi_d - \lambda_m}{L_d} \\ i_q &= \frac{\psi_q}{L_q} \end{aligned}\right\} \tag{3}$$

where τ_e and τ_L are electromagnetic torque and load torque, respectively. $\dot{\psi}_d$ and $\dot{\psi}_q$ represent the derivative of the d- and q-axis fluxes, ω is the rotor angular speed, and $\dot{\omega}$ is its derivative. J is the moment of inertia of the motor and load, R_s is the stator winding resistance, and B is the friction coefficient of the motor. The parameters L_d and L_q denote the d- and q-axis inductances, respectively. Figure 1 illustrates the functionality of the model representing the actual PMSM under study, incorporating both (2) and (3) in the machine model interface. In the Fig. 1, the Park-Clarke transformations have also been utilized. The following equation is used for the Clarke transformation (abc to $\alpha\beta 0$).

$$\begin{bmatrix} v_\alpha \\ v_\beta \\ 0 \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} \tag{4}$$

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} \tag{5}$$

The Park transformation ($\alpha\beta 0$ to $dq0$) converts signals from a two-phase stationary reference frame, as produced by the Clarke transformation, into a rotating reference frame. Likewise, the inverse transformation, used in the model, is represented by the following:

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} v_d \\ v_q \end{bmatrix} \tag{6}$$

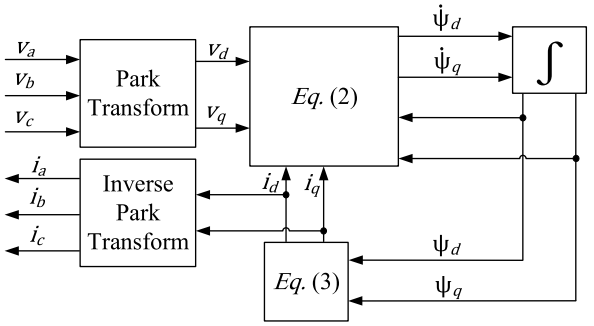


FIGURE 1. Machine model interface.

TABLE 2. Motor and inverter parameters used in the study.

Parameter	Value	Parameter	Value
Rated Power	4.25 kW	Pole Pairs, P	4
Nominal Speed	575 r/min	PM Flux, λ_m	0.577 Wb
Max. Current I_{max}	15.55 A	q-axis Inductance, L_q	65.87 mH
DC Bus, V_{dc}	450 V	d-axis Inductance, L_d	30.45 mH
Inertia, J	0.0375 kg.m^2	Stator Resistance, R_s	1 Ohm
Friction Coefficient, B	1	Frequency, f_{sw}	10 kHz

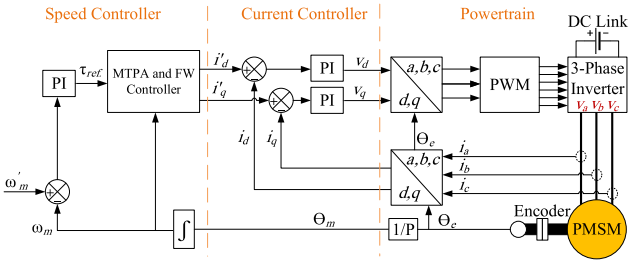


FIGURE 2. The schematic representation of API-based speed and current controllers, including the maximum torque per ampere and flux weakening control block.

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & 0 & \frac{1}{\sqrt{2}} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & \frac{1}{\sqrt{2}} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} v_\alpha \\ v_\beta \\ 0 \end{bmatrix} \tag{7}$$

where d and q are the direct and quadrature axis components in the rotating reference frame, α and β are the stationary axis components from the Clarke transformation, θ is the angle of rotation of the reference frame, and 0 represents the zero-sequence component, which remains unchanged.

All the inherent PMSM and drive parameters in this research are presented in Table 2, which then will be used to study the variation of parameter in the PMSM and the control system. The range of the variation in the motor parameters in real-life for the motor are found to be less than 20% in extreme conditions based on specific motor types, applications, and environmental factors [1].

B. API DESIGN AND STRUCTURE

Speed and current controls are pivotal elements in the control of stator current and torque in PMSM, exerting a significant

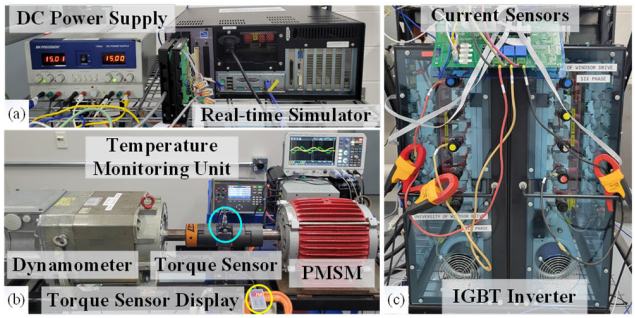


FIGURE 3. Experimental test bench for the PMSM under study with the API controller. (a) Integrated real-time simulator used for validation with HIL and SIL. (b) The PMSM connected to the dynamometer programed for the purpose of data acquisition and analysis. (c) IGBT inverter used to drive the PMSM.

impact on the transient and dynamic performance of the PMSM system. The two current and speed controllers are the main components of the PMSM drive system, as they are the brain for the driving inverter. The machine dynamics and transient performance are highly dependent on the performance of these two controllers [7]. In the API-based control approach depicted in Fig. 2, within both current and speed control loops, the input of each PI control block, parameters of the PMSM under study and the power inverter are used to tune the PI controllers as control block are the error signals, which are calculated by subtracting the command and feedback values. The two PI controller in the current control loop are responsible for delivering the reference torque (τ_{ref}) to the machine and also deliver the d- and q-axis voltages, which are denoted as V_d and V_q, respectively. The proportional (P) and integral (I) parameters of the API controller in (2) and (3) are derived from the discretization of the equivalent state-space reference model of PMSM [33] to be executed on the controller in experimental test setup depicted in Fig. 3. In these two equations, T_s denotes the sampling time of the system, f_{sw} is the sampling frequency. L_d and L_q are the d- and q-axis inductances. Also, parameters ω_{ref} and ω_{fb} refers to reference and feedback values of the rotor angular speed, respectively. Similarly, v_{d_ref} and v_{q_ref} represent the reference values for d- and q-axis reference voltages, and v_{d_ref} and v_{q_ref} represent the feedback values for d- and q-axis voltages. In order to comprehend the dynamic and transient performance of the PMSM, the API controllers are tested on the experimental testbed shown in Fig. 3. The current and speed controllers are evaluated by loading the motor using the depicted dynamometer. The API controllers are used to keep track of the d- and q-axis current, and also the rotor speed. For the purpose of control prototyping, a real-time simulator (RTS) along with an automotive grade controller board are used for controlling the insulated gate bipolar transistor (IGBT)-based inverter.

$$\tau_{ref} = \left(\left(\frac{\pi J f_{sw}}{100\frac{3}{2}P\lambda_m}\right) + \left(\frac{\pi B f_{sw}}{100\frac{3}{2}P\lambda_m}\right) T_s \frac{1}{z-1}\right)(\omega_{ref} - \omega_{fb}) \tag{8}$$

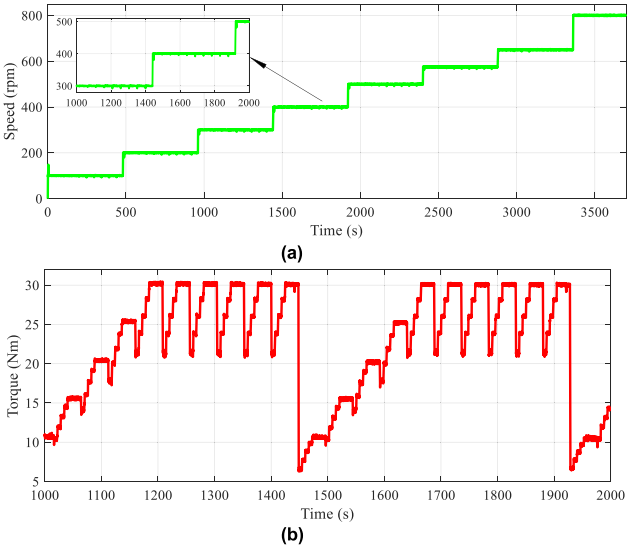


FIGURE 4. Experimental results of dynamic performance of the adaptive PI controllers, encompassing a) speed tracking and b) torque tracking.

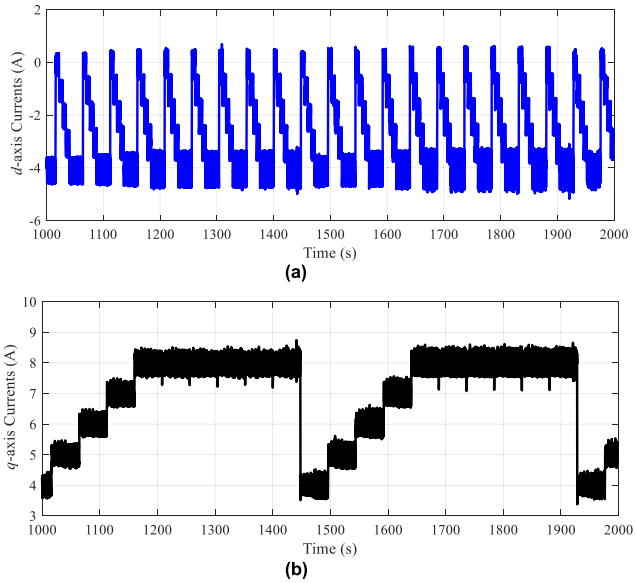


FIGURE 5. Current tracking profile achieved by experimental results of adaptive PI control, showcased in both the (a) d-axis and (b) q-axis reference frames.

$$\left.\begin{aligned} v_d &= \left(\left(\frac{\pi L_d f_{sw}}{5}\right) + \left(\frac{\pi R_s f_{sw}}{5}\right) T_s \frac{1}{z-1}\right)(v_{d\_ref} - v_{d\_fb}) \\ v_q &= \left(\left(\frac{\pi L_q f_{sw}}{5}\right) + \left(\frac{\pi R_s f_{sw}}{5}\right) T_s \frac{1}{z-1}\right)(v_{q\_ref} - v_{q\_fb}) \end{aligned}\right\} \tag{9}$$

Under the conducted API validation tests, the rotor speed, electromagnetic torque, and d- and q-axis currents are recorded and presented in Figs. 4 and 5. The torque and speed variation of the motor can be observed in Fig. 4, in which small transient overshoot and undershoot can be observed due to variation in torque. Figure 4(b) illustrates the torque profile of the machine in response to variations in the d- and q-axis

currents depicted in Fig. 5. The torque profile, detailed in Fig. 4(b), systematically adjusted in the range of 6 to 30 Nm. Although the performance remains stable, some instances of overshoot and undershoot are observed in the waveform. Furthermore, these fluctuations in torque have a noticeable impact on the accuracy of speed controller, as evidenced in Figure 4(a). In addition, the current tracking profiles presented in Fig. 5 shows the transient peaks in the d- and q-axis currents, which mainly resulted from the decoupling inaccuracy of the API controllers. That is the transient overshoots observed in d-axis currents are resulted from changes in the q-axis current. To minimize the decoupling inaccuracy of API controllers, and also improve the performance at the presence of parameter fluctuation, a DNN-based vector control approach is proposed. In order to develop and validate the presented approach, a range of 20% for the variation of motor parameters is an acceptable span [1]. In addition, the inertia $\boldsymbol{J}$ and friction coefficient $\boldsymbol{B}$ of the system will slightly change due to load torque variation in different drive cycle conditions, but they are not considered in this work.

III. METHODOLOGY OF DENSE NEURAL NETWORK-BASED VECTOR CONTROL IN CURRENT AND SPEED CONTROL LOOPS

The proposed DNN-based controller brings forth a model free control approach and have strict data requirement for training purposes. On the other hand, the schemes discussed in references [34] and [35], employs model-based and observer-based strategies for machine control and estimation, respectively. The work presented in [34] improves the robustness of control by using a sliding mode observer (SMO) to estimate and compensate for errors due to parameter disturbances. This work presents a relatively simple design for reducing the dependency on predictive control parameters by utilizing an improved robust finite control set predictive current control algorithm. This work aims to reduce the complexity and inaccuracy of selecting voltage vectors in conventional finite control set model predictive control (FCS-MPC). However, this method still requires prior knowledge of motor parameters and the complexity of implementation may be still higher in these approaches due to design and integration of SMO. The research presented in [35] discusses FCS-MPC methodologies for PMSM drive systems. Based on this review, the online parameter identification-based control methods benefit from the fact that these schemes can adapt to changes in the motor parameters in real-time, thus by continuously updating model parameters, these approaches can optimize controller for higher efficiency and better performance. However, these strategies, suffer from implementation complexity due to the potential computationally intensive algorithms. Additionally, continuous update of parameters adversely affects the cost and energy consumption of the control system. Moreover, incorrect parameter estimation may lead to system instability or degraded control performance.

In order to minimize the above mentioned challenges, the proposed DNN-based control approach, offers higher flexibility and adaptability to various operational conditions without extensive prior motor parameter knowledge, as it is a model-free scheme based on the neural network. This contrasts with the FCS-MPC approach, which, while effective, might be more reliant on model parameters and conditions. However, in order to minimize the chance on any generalization issues and their effects on the control performance, adequate data should be at hand for the purpose of offline training of the DNN structure. Based on the presented discussion, the proposed DNN-based scheme offers innovative solutions to PMSM control challenges particularly in terms of adaptability and reducing reliance on prior parameter knowledge. Meanwhile, computational demands and generalization challenges must be considered while designing these controllers. In comparison, the approaches presented in [34] and [35] provide valuable perspectives on the strengths and limitations of model-based control, highlighting areas where DNN-based approaches could complement or offer alternative solutions.

This paper presents a new DNN-based control for PMSM e-drives, which significantly enhances the control efficiency. The proposed controller demonstrates increased robustness and stability, as well as improved responsiveness and accuracy. These features make it a promising solution for modern electric drive systems aiming to meet the stringent demands of industries. The proposed control approach ensures reliable performance under various operating conditions, thereby addressing the challenges faced by traditional methods in maintaining efficiency and precision in dynamic environments.

The fundamental principle, supported by the universal approximation theorem [36], emphasizes the intrinsic capability of dense neural networks to accurately approximate complex and nonlinear functions, which is a relevant characteristic that is intrinsic to PMSMs. Architecturally, the stratification of neural networks into layers, usually denoted as L, each characterized by weight matrices $w^{(L)}$ and bias vectors $b^{(L)}$, enables a more intricate representation of motor dynamics. The backpropagation mechanism effectively reduces empirical risk across a meticulously selected dataset, thereby exemplifying the proficiency of DNN-based approach in data-driven control. The optimization of performance is contingent upon the careful calibration of hyper-parameters, which is accomplished using the grid search technique in this research, ensuring a proper and accurate selection of the required hyper-parameters of the networks.

In order to properly train the developed DNNs, a model-based tuning method and the system transfer function are used initially to optimize the API controllers for minimal error under conventional field oriented control (FOC) approach for PMSM. Then, the data acquired from the performance of these API controllers is then used to train the DNN-based speed and current controllers. The results of the proposed network are then compared to those of the API

controllers. Clearly, the larger the training dataset available for offline network training, the better the DNN-based controllers are tuned.

The first proposed DNN-based controller will regulate both the d- and q-axis currents fed to the PMSM, while the second proposed controller will regulate the speed of the PMSM. The proposed DNN-based controllers benefit from nonlinear dynamics handling. PMSMs inherently exhibit complex and nonlinear dynamics, while traditional control methods may struggle to accurately model and control such intricate behaviors. Also, DNNs excel at approximating nonlinear functions, allowing them to effectively capture and control the nonlinear dynamics of PMSMs while maintaining a minimalistic use of recourses in the system, making it a prominent robust choice to be used in real-time systems. This capacity of intelligent controllers is also demonstrated by the work presented in [28]. Another benefit of the DNN-based controller is adaptability to varying conditions. They are adaptive and can learn from data, enabling them to dynamically adapt to variations in the system. This adaptability is crucial in real-world applications where conditions may evolve, providing the flexibility needed for the controller to maintain optimal performance under changing circumstances. Hence, such intelligent, model-independent controllers will improve the performance and efficiency of the whole machine drive system, as demonstrated by real-time simulation and experimental results presented in the following sections.

To enhance the controllers' comprehension of the system they control, each control objective is analyzed alongside its associated error, the integral of the error, and the error from the previous time-step. These parameters serve as inputs for the controllers during both their training phase and experimental implementation. Specifically, in the case of the speed controller, the inputs include the speed, the speed error, which is the deviation of the machine from the reference values, the integral of this error over time, and the error achieved in the preceding time-step. Likewise, the current controller receives the d- and q-axis currents, their deviations relative to the reference value, the integral of these deviations, and the error from the previous time-step as inputs. The configuration of this control strategy is illustrated in Fig. 6. In this figure, two sets of dense neural networks are utilized in current and speed control loops of the PMSM drive system. The parameter θ_e is the measured electrical rotor position, and ω_m symbolizes the mechanical angular velocity of the rotor. Moreover, i and v represents measured currents and voltages in the system, which are differentiated by their indices. The presented configuration of inputs will provide the controller with a more in-depth understanding of the system and the motor being controlled; thus the controller is capable of anticipating the proper control action at each time-step for tracking error minimization.

To control currents, the traditional vector control employs two sets of PI controllers, whereas just one PI block is utilized for the speed control loop. On the other hand, in the presented scheme, only one dense neural network has taken the role of the two PI controllers to tune currents in the system. In a similar manner, the speed PI controller was also replaced with another dense neural network. In this work, the objective is to train a set of dense neural networks that comprehend the conventional vector control approach and provide a more effective control action than API controllers in terms of responsiveness and accuracy.

IV. TRAINING OF DNN-BASED CONTROLLER CONSIDERING HYPER-PARAMETERS OPTIMIZATION

Training of the two separate dense neural network controllers, as speed and current control, in PMSM plays a crucial role in finding the optimal e-drive control. To begin with, each DNN has numerous hyper-parameters to be defined, including the number of hidden layers and neurons, the activation function at each neuron, the number of inputs, and the error threshold values. In this paper, in order to determine these hyper-parameters, a grid search analysis has been conducted. To do so, a grid with various values of each hyper-parameter is constructed. Then the training algorithm tries all the possible combinations and calculates the final error value for them. Based on the determined error values, the best combination is then selected. In addition, to further expedite the process, automated scikit-learn hyper-parameter tuning libraries are used.

The Levenberg-Marquardt (LM) training algorithm employs a numerical strategy to resolve the minimization problem of a function [37]. Thus, it is an efficient algorithm for DNN training because of its rapid and consistent convergence by combining the steepest descent algorithm and Gauss-Newton training methods. Fortunately, it inherits the speed advantage of the Gauss-Newton algorithm as well as the stability of the steepest descent. That is, the algorithm employs a hybrid training procedure: around regions with complex curvature, it switches to the steepest descent algorithm until the local curvature is found suitable for a quadratic approximation, and then it switches to the Gauss-Newton algorithm to accelerate the convergence toward minimizing the error values and pass the set threshold.

The LM algorithm uses an approximation of the Hessian Matrix, H, to ensure that it is reversible. Hence, the LM algorithm utilizes the following approximation:

$$H \approx G^T G + \mu I \tag{10}$$

where G is the Jacobian matrix of the error vector, I is the identity matrix, and μ is the combination coefficient which is always a positive scalar with the purpose of adjusting the LM algorithm dynamically. Based on the presented approximation, the updating rule for the weights of the networks in the algorithm can be stated as follows [37]:

$$w_{k+1} = w_k - \left(G_k^T G_k + \mu I\right)^{-1} G_k e_k \tag{11}$$

where w_k is the weight matrix, and e_k is the vector of network errors. If the latest calculated value of mean squared error (MSE) decreases in each update iteration, for example,

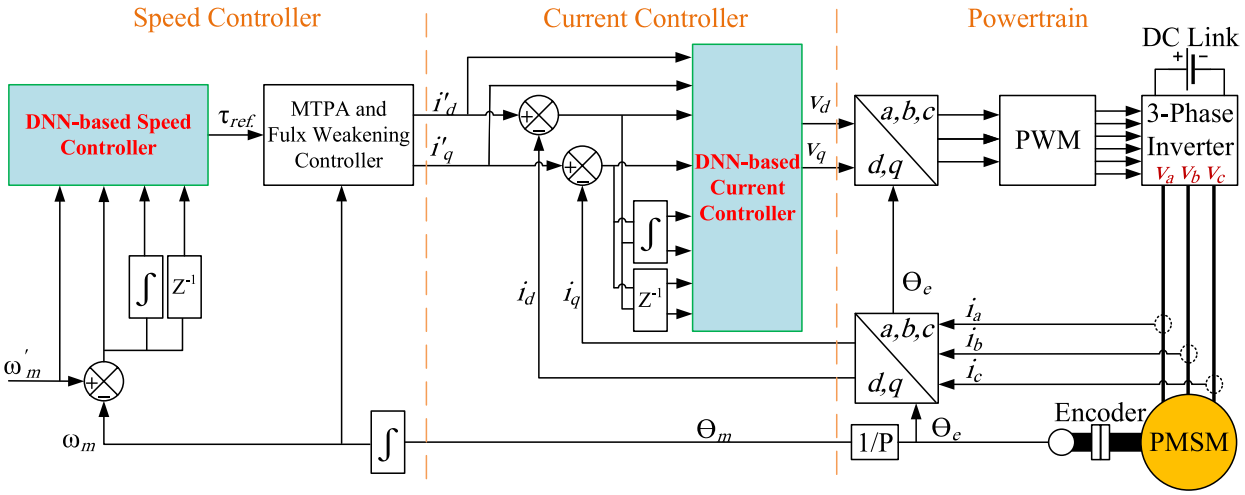


FIGURE 6. The schematic representation of the proposed DNN-based speed and current controllers, including the maximum torque per ampere and flux weakening control block.

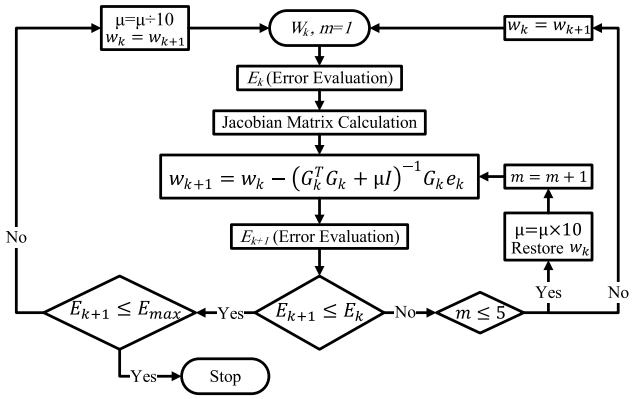


FIGURE 7. Flowchart of modified training procedure for the proposed dense neural networks.

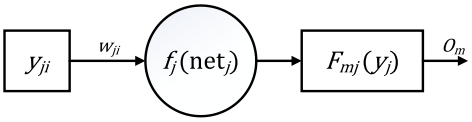


FIGURE 8. Connection between neuron j and the remainder of the network.

it is less than previously calculated error, the combination coefficient μ would be changed to smaller values in the update algorithm in order to reduce the influence of gradient descent. Finally, Fig. 7 depicts the modified flowchart of the training approach based on the novel DNNs. This figure depicts the process of the DNN training algorithm and outlines the stages through which the algorithm updates the network weights. In the training algorithm. If the most recent computation of the MSE indicate a decrease, signifying it is lower than the error calculated previously, the combination coefficient μ will be adjusted to smaller values within the update algorithm. This adjustment aims to diminish the impact of gradient descent. Conversely, an increase in MSE, suggesting it surpasses the error estimated earlier, necessitates a more pronounced adherence to the gradient's direction. Consequently, the combination coefficient μ will be increased to accommodate this requirement [37]. The steps the algorithm takes are summarized as follows:

1. Initiate with a randomly selected set of initial weights and evaluates the MSE or E_k after each iteration.
2. Employ (11) to update the weights.
3. Compute the MSE utilizing the newly adjusted weights.
4. In instances where the MSE increases, revert the weights to their prior settings and amplify them by a factor of ten. Subsequently, undertake another round of weight modification. Here, m serves as an iteration counter, with its values spanning from 1 to 6.
5. If there is a reduction in the newly calculated MSE, retain the current weights but decreasing them by a factor of ten.
6. Continue this iterative cycle until the MSE attains a value below the predetermined threshold.

Based on the illustration in Fig. 8, imagine a neuron j with n_i number of inputs. If neuron j is located in the first layer of the network, then all of its incoming data will be connected to the inputs of the network. If neuron j is located in a layer other than the first, then its inputs can be connected to the outputs of prior neurons or to the inputs of the network, if the connections between layers are permitted. In Fig. 8, which is the basis of the structure of proposed controllers, the i^{th} input of neuron j is indicated by y_{ji} while the output of a neuron is y_j. Based on this figure, y_j may be calculated as (12). The output for the current controller would be d- and q-axis reference voltages. For the speed controller, the output is the demanded toque.

$$y_i = f_j\left(\text{net}_j\right) \tag{12}$$

where f_j is the sigmoid function modified for our training algorithm, the activation function of neuron j and net value

net_j is the sum of neuron j's weighted inputs calculated from (13) as follows:

$$net_j = \left(\sum_{i=1}^{n} {}_i\left(w_{ji}\right) \times \left(y_{ji}\right)\right) + w_{j0} \tag{13}$$

where, w_{ji} are the network weights and w_{j0} is the bias weight of neuron j. Then, the inputs can be obtained as:

$$y_{ji} = \frac{\partial \text{net}_j}{\partial w_{ji}} \tag{14}$$

Also, the slope of the activation function, s_j, can be defined as:

$$s_j = \frac{\partial y_j}{\partial \text{net}_j} = \frac{\partial f_j(\text{net}_j)}{\partial net_j} \tag{15}$$

As seen in Fig. 8, a complicated nonlinear connection exists between the output node y_j of a hidden neuron j and network output o_m as the m^{th} output of the network, and it can be denoted as:

$$o_m = F_{mj}\left(y_j\right) \tag{16}$$

The elements of the Jacobian matrix required for the approximation of the Hessian Matrix are formulated based on the prerequisites of current and speed control loops, and will be implemented in Algorithm 1 for training. The elements are formulated as below in which d_m is the m^{th} desired output of the network:

$$\begin{aligned}\frac{\partial e_m}{\partial w_{ji}} &= \frac{\partial\left(d_m - o_m\right)}{\partial w_{ji}} = -\frac{\partial o_m}{\partial w_{ji}} \\ &= -\frac{\partial o_m}{\partial y_j}\frac{\partial y_j}{\partial \text{net}_j}\frac{\partial \text{net}_j}{\partial w_{ji}}\end{aligned} \tag{17}$$

Combining (14) to (16), (17) may be rewritten as follows:

$$\frac{\partial e_m}{\partial w_{ji}} = -F'_{mj} s_j y_{ji} \tag{18}$$

where F'_{mj} represents the nonlinear function's derivative between neuron j and output m.

Based on the provided equations, in first-order algorithms, traditional backpropagation computation can be used to structure the Jacobian matrix calculation process for the system, but there are some differences. For each pattern, only one backpropagation step is needed in first-order algorithms, but in the presented algorithm, the backpropagation process must be repeated for each output to obtain successive rows of the Jacobian matrix. Additionally, the parameter δ (consisting of output errors in first-order algorithms) must be adjusted through backpropagation. The algorithm calculates δ parameters independently for each neuron j and output m. In addition, the error is replaced with a unit value throughout the backpropagation procedure.

$$\delta_j = s_j \sum_{m=1}^{M} F'_{mj} e_m \tag{19}$$

$$\delta_{mj} = s_j F_{mj} F'_{mj} \tag{20}$$

TABLE 3. Structural hyper-parameters of trained DNNs in both Speed and current control loops.

Hyper-parameters	Current Loop DNN	Speed Loop DNN
Number of hidden layers	3	4
No. of neurons at each hidden layer	10	10
Number of inputs	8	4
Number of outputs	2	1
Learning rate	10^{-3}	10^{-3}
Activation function	tanh	tanh
Sample time [s], T_s	10^{-4}	10^{-4}
Training error threshold	10^{-5}	10^{-5}

Algorithm 1 LM Training Implementation

```
%Forward Computation:
for all layers do
    for all neurons in the layer do
        Calculate net;                %Equation (13)
        Calculate slope;              %Equation (15)
        Calculate output;             %Equation (16)
    end for
end for
%Backward Computation:
Initial delta as slope;
for all outputs do
    Calculate errors;
    for all layers do
        for all neurons in the previous layer do
            for all neurons in the current layer do
                Multiply delta through weights;
                Sum the backpropagated delta;
            end for
            Multiply delta by slope;
        end for
    end for
end for
```

Then elements of the Jacobian matrix can be determined by combining (18) and (20). The pseudo code for the forward and backward computations for the Jacobian matrix is presented in Algorithm 1.

The hyper-parameters of the designed speed and current DNNs, shown in Table 3, are used in the training algorithm to reach the threshold MSE value of 10^{-5}, based on the explained technique and normalized values. It should be noted that the proposed method is compared with the same controller from which the training data were acquired. Using MSE as the evaluation metric, the training procedure attained error values of less than 10^{-5} after 4,500 and 5,000 training iterations for the current and speed control blocks, respectively. It should be noted that the training procedure utilizes an RTX 2080Ti to ensure rapid and precise flow.

There are a few limitations and challenges associated with the proposed speed and current controllers. Firstly, there is a

need to prevent deficiencies in the capabilities and generalization of the proposed DNN controllers. Ensuring adaptability and robustness of the proposed approach is another important consideration during the design procedure of the DNN-based controllers. In order to inhibit these issues from affecting the control performance of the motor across all operational conditions, it is insured to use redundant data for the training procedure. That is, the data has been acquired in various speed and load conditions ensuring that all combinations and range of commanded speeds, torques and currents are utilized as training data. Additionally, a sectional training procedure has been used to minimize out-of-sample discrepancies. This has been done by randomizing and sectioning the whole raw training data in to ''Training'', ''Validation'', and ''Testing'' parts. Form the whole available training data, 60% of the data is allocated to network training. Another 30% is used for validation of the conducted training and implementing any required adjustment. Finally, 10% of the data is used for final testing of the trained network to validate the out-of-sample performance. Last but not the least, one limitation associated with the DNN-based controller is the computational efficiency. While it can potentially improve the performance and accuracy of the drive system, ensuring the computational efficiency, especially, in real-time applications, might be challenging for complex structures. In order to minimize the computational burden using the proposed DNN-based controller, two avenues have been considered. First, minimum number of neurons and hidden layers in the structure of the network have been used while maintaining the error values below the set threshold. Second, an offline training approach is used to design and adjust the DNNs. This significantly reduces the computational burden of the controller in comparison to those controllers with constant online training and adjustment like DRL.

For the first DNN, acting as the current controller, the number of hidden layers is decided to be 3 as they would offer a proper estimation accuracy. After determining the number of hidden layers, it is necessary to determine the number of neurons in each of the hidden layers, which is established by grid search. Each hidden layer of the proposed DNN-based current controller contains 10 neurons along with an additional bias neuron. It should be highlighted that fewer neurons are more desirable as they require fewer resources and takes less time to be trained and integrated into the e-drive control. Moreover, the current control block has eight inputs, which are the d- and q-axis currents, the computed closed loop errors, the integral values of the d- and q-axis currents, and the calculated current error in the previous time-step.

The second DNN, acting as the speed controller, is also designed around the same methodology, the inputs of the neural network for speed control are the angular rotor speed, the error of angular rotor speed, the integral of the angular rotor speed error, and the calculated error from the preceding time-step. For this control block, the number of hidden layers is set at 4, with 10 neurons in each hidden layer with an additional bias neuron in each hidden layer. The speed controller has one hidden layer more than the current controller; that is the fact that the speed controller has fewer number of inputs in comparison, hence, it has less information to estimate the output based on those. An additional layer would increase the comprehension capability of the speed controller.

FIGURE 9. Experimental SIL test setup for real-time validation of the proposed DNN-based speed and current controller using OP-4510.

The architecture of the proposed DNN-based controller initiates with a minimal number of hidden layers, specifically one, and two hidden neurons within the said layer. The process of determining the appropriate number of hidden layers and neurons for each controller entails incrementally increasing these values until the threshold output error value is attained using the most fundamental structure. It is important to note that when determining the structure of the trained network, activation functions are also taken into account.

V. EXPERIMENTAL INVESTIGATION TEST SETUPS IMPLEMENTING SOFTWARE-IN-THE-LOOP (SIL) AND HARDWARE-IN-THE-LOOP (HIL)

The Software-In-the-Loop (SIL) simulation is conducted using an RTS [17]. In this real-time simulation, unlike the conventional simulation approach, the software components are executed on the RTS instead of the physical target hardware. This allows validation and fine-tuning of the controller without risking damage to physical hardware. In this regard, the control algorithms are programmed into the field-programmable gate array (FPGA)-based RTS to fulfill the purpose, while the remainder of the models are compiled on the host computer. The FPGA-based RTS is used to have an online emulation for the hardware environment of the system, whereas the host computer is used to accelerate the rest of computationally intensive duties. The combination of hardware in the RTS and software in the host computer provides a more accurate representation of the system's behavior in a real-world setting and can help with identifying and resolving any existing problems swiftly and effectively. Hence, to begin with the experimental validation, the SIL test setup presented in Fig. 9 has been utilized. As can be seen in this figure, the host computer interacts with RTS and provides a real-time simulation infrastructure for the proposed current and speed controller validations. In this setup, an OP-4510 RTS system has been utilized due to its portability and adaptability with the host PC, which have been used for his portion of conducted experimentation.

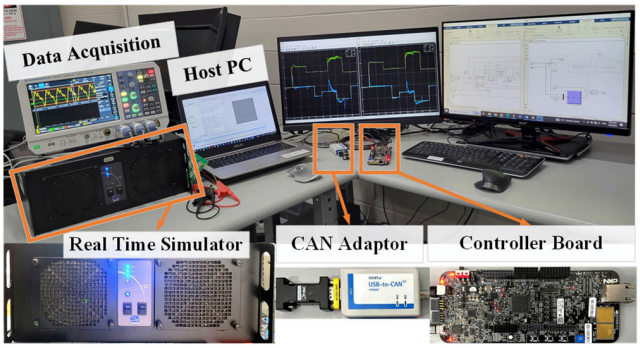


FIGURE 10. Experimental HIL test setup including the automotive-grade controller board acting as the platform for the controller in the real-time test system.

At the next stage of experimental validation, by adding an automotive-grade controller board to the system to facilitate the speed and current control duties, the HIL test system will be achieved. In the HIL test setup, the controllers are running on a dedicated processor or controller board. Within the HIL test setup depicted in Fig. 10, the automotive-grade controller board is connected to the RTS via CAN communication interface. In this test setup, the same RTS shown in Fig. 3 has been used due to easier access to the I/O pins. In addition, the motor will be emulated on RTS to avoid any problems caused by any errors that could result in damage to the driven object by failing to properly set various constraints of the software or hardware protections. Also, in order to properly implement the control scheme, the following steps should be taken. Firstly, to satisfy the determinacy requirement for code auto generation, the whole model must adopt a fixed step size solver. In this regard, the fixed step size is set to 50 μs based on the primary loop frequency. Moreover, the pulse width modulation (PWM) switching frequency should be set to 10 kHz, and it is used to activate the analog to digital converter (ADC). In addition, in the HIL test setup the real-time implementation of the presented control scheme relies on the high performance of a dual-core platform using Cortex-M7 cores, capable of operating up to 160 MHz, which will be deliver the 50 μs time-step.

For the implementation of the proposed control algorithm must be normalized and represented in 32-bit fixed-point integer representation, rather than in floating-point format in order for the controller board to function and communicate properly. Modular connections should also be implemented in this format to ensure seamless operation of the controller.

In order to assess the efficacy and applicability of the proposed DNN-based speed and current control technique, at the first step, SIL testing is performed utilizing an RTS. In the experimental testbed shown in Fig. 3, using a programmed dynamometer, the motor is spun at a constant torque or speed in order to evaluate the tracking capability of the DNN speed and current controllers. To correctly analyze and verify the performance of the proposed controller, the necessary C code for the testing is built and compiled. The real-time

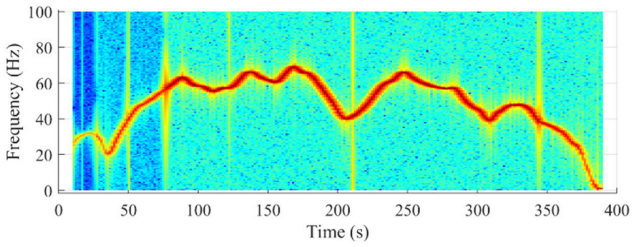


FIGURE 11. Frequency components in the 3-phase currents using the proposed DNN-based controller under WLTC class 1 drive-cycle.

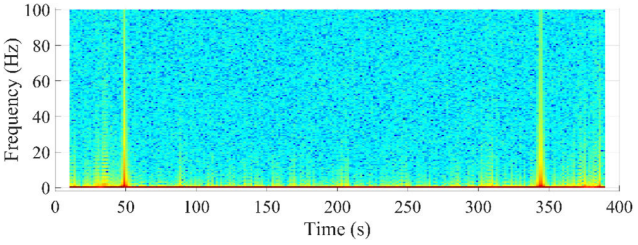


FIGURE 12. Frequency components in the d- and q-axis currents using the proposed DNN-based controller under WLTC class 1 drive-cycle.

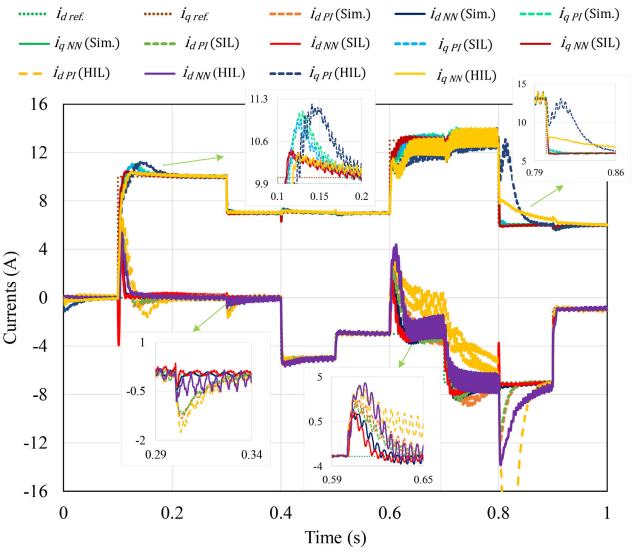


FIGURE 13. Current tracking of the DNN-based controller with 20% increased parameter R_s, L_d and L_q at the nominal speed.

performance, and robustness of DNN-based controllers will be fully evaluated via these experiments. Thus, the RTS will facilitate the effective assessment of the control algorithm, allowing for the performance improvement of the whole e-drive system.

VI. PERFORMANCE EVALUATION UNDER PARAMETER VARIATIONS FOR PMSM CONTROLLERS: DEEP NEURAL NETWORK VERSUS API

In the proposed DNN-based control method, due to the understanding of the control system and the reaction of the device under test to various inputs, the controller can overcome a

relatively sluggish response, and reduced parameter reliance compared to the API controllers. Since the DNN has numerous hyper-parameters, such as the number of optimal neurons in each layer and the number of hidden layers, the activation function at each neuron, and error threshold values, by finding optimal hyper-parameters in DNNs through grid search analysis and automated scikit-learn tuning algorithm library, the proposed controllers are meticulously designed to provide more efficient control commands.

During the initial phase of design and validation, a high-fidelity simulation of the system is created and serves as the foundation for the design and validation of the proposed DNN-based control approach. It also served as the basis for the SIL and HIL testing. As described in the preceding section, the SIL testbed utilized the RTS system to simulate the proposed control approach in real time. The RTS system compiles the simulation file into a C code which is then is used to generate the SIL executable file in the real-time target to be executed. After the SIL validation is completed, an automotive-grade controller board is installed at the RTS interface to provide a more realistic validation of the proposed algorithm under HIL test conditions. Additionally, as neural networks need parallel processing, it would be optimal to implement them on an FPGA board or an RTS that ensures their real-time functioning. However, owing to the simplicity of the structure of the proposed method, its implementation on a microcontroller board would not hinder the real-time performance. To verify the performance of the proposed controller, an RTS along with an automotive-grade controller board were used to realize the HIL testing setup.

This research has also studied the presence of the other lower and higher frequencies in the system, mainly caused by PWM controlled IGBT inverter, used to drive the PMSM under study. In order to do this study, a part of the WLTC class 1 drive cycle has been used, specifically, the last 40% of this drive cycle. The graphical representation depicted in Fig. 11 depicts the spectrogram of the existing frequencies in the system with the discussed drive cycle. This figure highlights the frequency components inherent within the three-phase currents of the machine. The color map in this figure represent the intensity of each frequency at each sampling time-step, as it moves toward the dark red color, the intensity is increased. The dark red color depicts the fundamental frequency of the system at each calculation step. As expected, at each step, the profile of the fundamental frequency in the system completely matches the changes in the speed of the PMSM validating the presence of non-fundamental frequencies in the system. The higher the speed of the machine, the higher fundamental frequency is recorded. In addition, it should be noted that the total length of this experimentation is 400 seconds, however the first and the last 10 seconds of the figure is left blank due to the delay and the amount of data required for frequency analysis calculations. In order to demonstrate how these the presented non-fundamental frequencies transferred to the rotor reference frame, Fig. 12 depicts the frequency components within the d- and q-axis currents encompassing the motor dynamics. Similarly, the higher the frequency, the spectrogram color is closer to dark red. In this figure also the dark red color shows the main frequency components which is of course DC. As can be seen, other than DC, there are other frequency components in the rotor reference frame rooted from the PWM excited voltage source inverter (VSI).

The proposed controllers are initially developed with the normal parameters of the motor in mind. Then, a set of parameter variations are predefined (increased or decreased) based on realistic scenarios and constraints relevant to the application domain of the research. Then, at the time of robustness validation, these variations were then incorporated into simulation and experimental setups to evaluate how changes in these parameters affect the performance and effectiveness of our proposed solution. That is a modified machine model with the implemented changes programed in to the RTS to validate the capabilities of the proposed controller. To state the obvious, the utilized controller was exactly the same in both increased and decreased scenarios. It should be noted that, the selection of fixed variations was guided by both literature review and preliminary analyses to ensure that they represent meaningful and challenging conditions under which the proposed scheme should operate.

A. ENHANCED PERFORMANCE OF DNN-BASED CURRENT CONTROLLER UNDER PARAMETER VARIATION

Under varying driving conditions, aging and loads, the PMSM parameters are susceptible to change. These parameter fluctuations, such as the stator resistance (R_s), and d- and q-axis inductances (L_d, and L_q), are often nonlinear. In order to investigate the impact of this system parameter variation on the proposed DNN-based current control technique, in both current control loops, the parameters R_s, L_d, and $L_q$are changed over 20% relative to their initial values and are introduced to the system. Variations in the inductances translate to changes in permanent magnet flux, which are mostly due to nonlinear characteristics of the PMSM in different operation condition and various load points. The outcomes of these case studies are presented in Figs. 13 to16. Figure 13 compares the d- and q-axis SIL, HIL experimental and simulation current tracking results of the proposed controller with a 20% increment in the parameters values. As illustrated in the figure, the d-axis current varies from 0 to -7 A, while the q-axis current varies from 0 to 13 A. The current tracking profile depicted in the figure clearly demonstrates that the proposed technique outperforms the API-based control scheme in terms of overshoot and undershoot magnitudes, as well as settling time. The improvements achieved by the proposed scheme are further validated by simulation, SIL, and HIL tests, as shown in the figure, and they all testify for improved performance of the drive system using the proposed control method. The resultant electromagnetic torque is also presented in Fig. 14. The torque profile clearly shows that the proposed technique achieves superior and faster response time. The proposed method exhibits a significantly more

TABLE 4. Machine parameter variations for validating current controllers robustness.

Machine Parameters	R_s	L_d	L_q
+20% Parameter Variation	1.2 ohm	36.54 mH	79.044 mH
-20% Parameter Variation	0.8 ohm	24.36 mH	52.696 mH

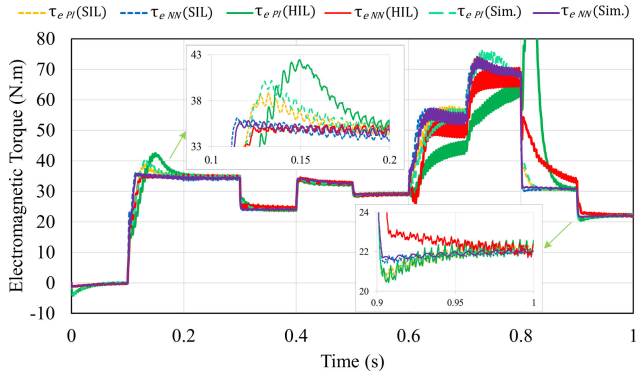


FIGURE 14. Experimental electromagnetic torque profile with HIL and SIL for the proposed controller and the adaptive PI controller with 20% increased R_s, L_d and L_q values at the nominal speed.

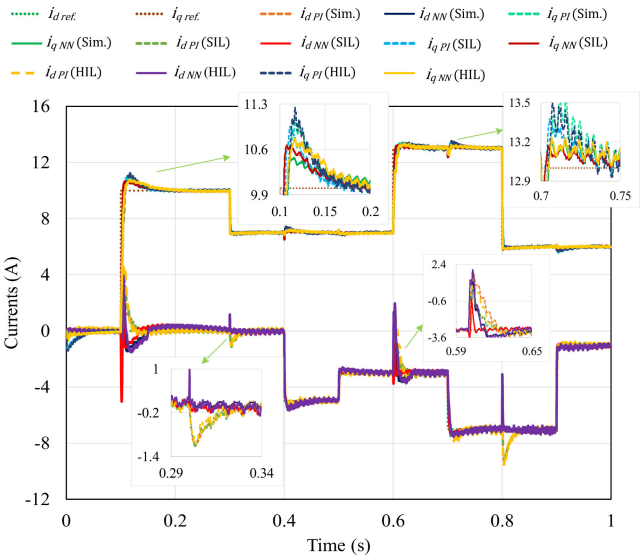


FIGURE 15. Current tracking of the DNN-based controller with 20% decreased parameter R_s, L_d and L_q at the nominal speed.

stable torque output compared to the API-based scheme. From these two figures, the proposed DNN-based current controller demonstrates superior stability and smoothness compared to API, especially amidst significant variations in demanded current along the d- and q-axis. Hence, implementing this controller in the e-drive system results in a more consistent and seamlessly operating electromagnetic torque. Table 4 represents the studied parameters of the machine changed with 20% in the validation process of the proposed DNN-based current controller. In the experimentation, these values have been used to produce the results depicted in Figs 13-16.

TABLE 5. Mean squared error comparison of DNN-based controllers and API at different time samples.

Time (s)		0.1	0.15	0.4	0.6	0.8
20% Increased Parameter	API ($\times 10^{-1}$)	7.1	6.0	7.8	7.2	4.8
	DNN ($\times 10^{-1}$)	3.5	3.6	2.4	3.7	3.1
20% Decreased Parameter	API ($\times 10^{-1}$)	4.2	3.1	4.7	4.3	3.9
	DNN ($\times 10^{-1}$)	2.2	1.8	1.6	1.9	1.7

On the other hand, Fig. 15 presents the comparative results for the d- and q-axis currents in SIL, HIL, and simulation tests, under the condition where the parameters R_s, L_d, and L_q are decreased by 20%. In the shown current tracking profile for the proposed controller, despite the reduction in system parameters, the DNN-based approach maintains a more desirable current tracking performance compared to the conventional methods. The results indicate that the proposed method is robust and can adapt to changes in system parameters, consistently achieving lower overshoot, reduced settling time, and better overall stability. This performance is evident across all testing scenarios, highlighting the effectiveness of the DNN-based approach in handling parameter variations.

Figure 16 depicts the resultant electromagnetic torque when the internal parameters of the machine are decreased by 20%. As can be seen in this figure, electromagnetic torque is smoother when the proposed DNN-based current controller is implemented in the e-drive system. It is evident from the numbers and overshoots magnitude that the proposed DNN-based controllers provide outcomes with less variation and spikes. As shown in these figures, the proposed DNN-based control method is more resilient to variations in motor parameter values (R_s, L_d, and L_q) and have a more dynamic performance. It reduced the amplitude of overshoots and shortened the settling time for the currents as well as for the profile of the electromagnetic torque. Hence, when the proposed DNN-based current controller is implemented in the system, the longevity of both the drive system and the machine would increase since less stress and strain would be introduced to the system in the long run. It is obvious in the figure that the torque profile does not demonstrate harsh overshoot as in the case with parameter increment, however, the profile is still smoother when the proposed DNN-based current controller is utilized in lieu of the API. In addition, Figs. 14 and 16, demonstrate that the experimental and simulation results for the performance validation of the DNN-based current controller, match each other to provide a solid proof for the effectiveness of the proposed approach.

By implementing the proposed DNN-based current controller in the presented test scenarios, based on the presented results, the proposed controller lessens the amplitude of overshoots by 50.4% with respect to the reference values in comparison with the API. Employing MSE, the discrepancy between steady-state and instantaneous current values at the moment of any step change is quantified and documented in Table 5. This table reveals that at each step of 0.1, 0.15, 0.4,

TABLE 6. Mean squared error comparison of DNN-based controllers and API in speed control loop under different speed.

Speed (r/min)		100	300	400	575	800
+30%	API ($\times 10^{-1}$)	2.7	2.8	2.9	4.0	4.1
Par. Var.	DNN ($\times 10^{-1}$)	1.3	1.6	1.7	2.5	2.6
-30%	API ($\times 10^{-1}$)	2.1	2.1	2	3.2	3.5
Par. Var.	DNN ($\times 10^{-1}$)	1.1	1.0	1.2	1.9	2.1

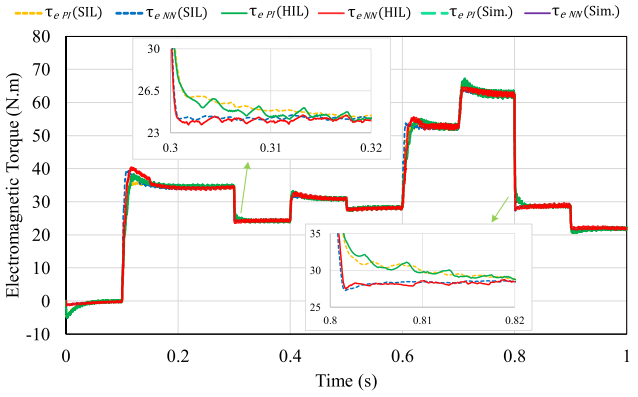


FIGURE 16. Experimental electromagnetic torque profile with HIL and SIL for the proposed controller and the adaptive PI controller with 20% decreased R_s, L_d and L_q values at the nominal speed.

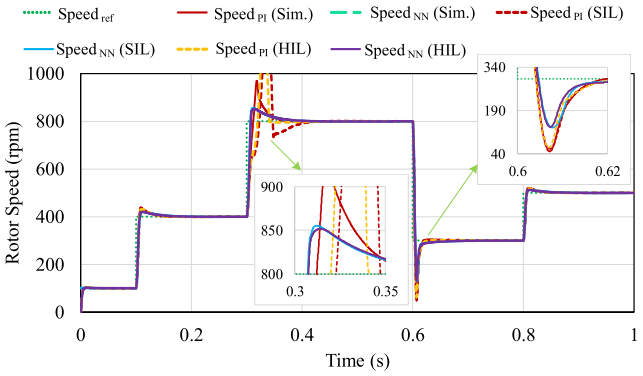


FIGURE 17. Speed tracking of the DNN-based speed controller with 30% increased R_s, L_d and L_q in no-load condition.

0.6, and 0.8 s, the MSE for both the API and the DNN-based controllers are recorded, taking into account a 20% increase and decrease in the parameters under study. Through this numerical analysis, it is evident that the DNN-based current controller exhibits reduced instances of overshoot and undershoot, thereby delivering a more consistent performance, and also it would be a more efficient control approach to be implemented in the PMSM drive systems. The MSE values for the proposed DNN-based speed controller are detailed in Table 6. This table lists the MSE results for a 30% parameter increase and decrease at speed steps of 100, 300, 400, 575, and 800 rpm. The data presented in Table 6 indicates that the DNN-based speed controller achieves superior performance in terms of speed regulation compared to the API method.

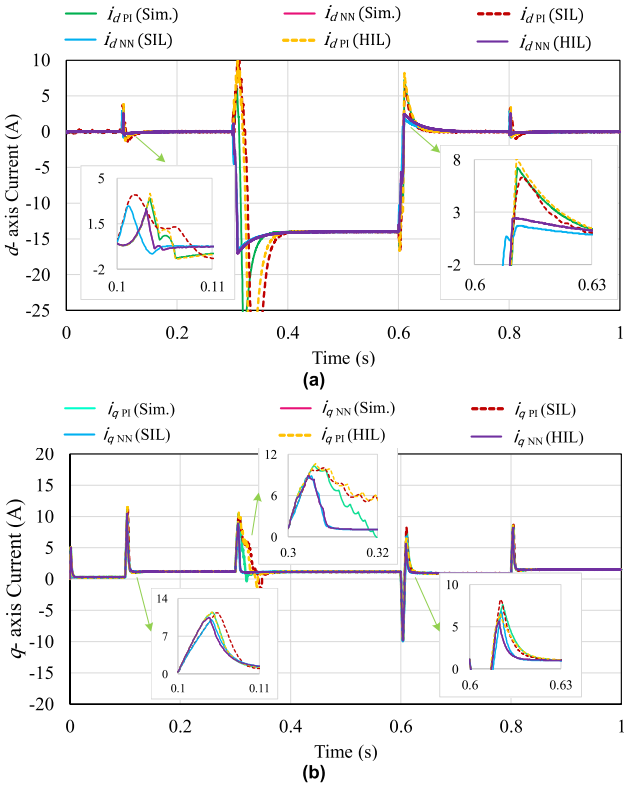


FIGURE 18. The d- and q-axis current regulation performance with the proposed speed controller and 30% increased R_s, L_d, and L_q under no-load. (a) *d*-axis. (b) *q*-axis.

B. ENHANCED PERFORMANCE OF DNN-BASED SPEED CONTROLLER BY CONSIDERING PARAMETER VARIATIONS

The robustness of the proposed DNN-based speed controller is examined by altering the values of R_s, L_d, and L_q, and then monitoring how the motor drive system as a whole performs under these conditions. The machine parameters are first increased by 30% in comparison to their base levels. The reference speed changed from 100 to 400 rpm at 0.1 s, then to 800 rpm at 0.3 s, to 300 rpm at 0.6 s, and to 500 rpm at 0.8 s. The findings of this investigation are shown in Figs. 17 to 20. In these figures, which represent the speed and current tracking of the proposed control method, the magnitude of overshoots are greatly reduced. In the case where the internal machine parameters are increased by 30%, Fig. 17 illustrates the comparative analysis of settling times between the proposed DNN-based control method with respect to API in HIL, SIL, and simulation test setups. As can be seen in this figure, the proposed method for the speed controller boasts a smooth tracking profile with respect to the demand values. Fig. 18, compared the current tracking profile, while the proposed speed controller is utilized. Based on this representation, the system experiences relatively smaller overshoots and undershoots. The *d*- and *q*-axis current profiles also demonstrate smoother profile for both currents, especially the *d*-axis current which experience relatively big overshoots while the API speed controller is implemented. This is mainly due to the fact

TABLE 7. Machine parameter variations for validating speed controllers robustness.

Machine Parameters	R_s	L_d	L_q
+30% Parameter Variation	1.3 ohm	39.585 mH	85.631 mH
-30% Parameter Variation	0.7 ohm	21.315 mH	46.109 mH

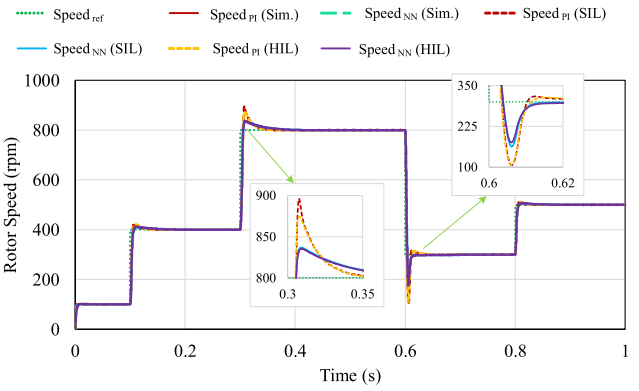


FIGURE 19. Speed tracking of the DNN-based speed controller with 30% decreased R_s, L_d and L_q in no-load condition.

that the conventional approach suffers from decoupling inaccuracy and its sensitivity to parameter variation. However, conspicuously the proposed DNN-based speed controller still outperforms the API.

Based on the presented experimental result, in this figure, the proposed speed controller provides an average of 47.9% smaller overshoots in comparison to the conventional approach with respect to the reference values.

Table 7 outlines the parameters of the machine that were adjusted by 30% during the validation process of the proposed DNN-based speed controller. These adjusted values were subsequently applied in the experimental test bed, yielding the results illustrated in Figs 17-20.

In the case of reduction in the studied parameters, motor parameter values (R_s, L_d, and L_q) are reduced by 30% as illustrated in Table 7. Figure. 19 depicts the speed tracking profile under such condition. Similar to the case study with the parameter increase, the result shows relatively smaller overshoots based on SIL and HIL experimental results for all speeds below and above the rated value. The d- and q-axis current profiles experience smaller overshoots and undershoots for the API compared to Fig. 18. However, using the proposed speed controller, the DNN-based controllers still present a superior performance with smoother tracking profile for both d- and q-axis currents as demonstrated in Fig. 20. This is mainly because the trained DNNs have adequate understanding of the system behavior in transient conditions. Hence, it can adjust the command signals properly under these transient conditions even in the presence of the experimented parameter variation. Consequently, the proposed speed controller would be more robust to parameter changes than the API control approach. Furthermore, based on the presented experimental results, and

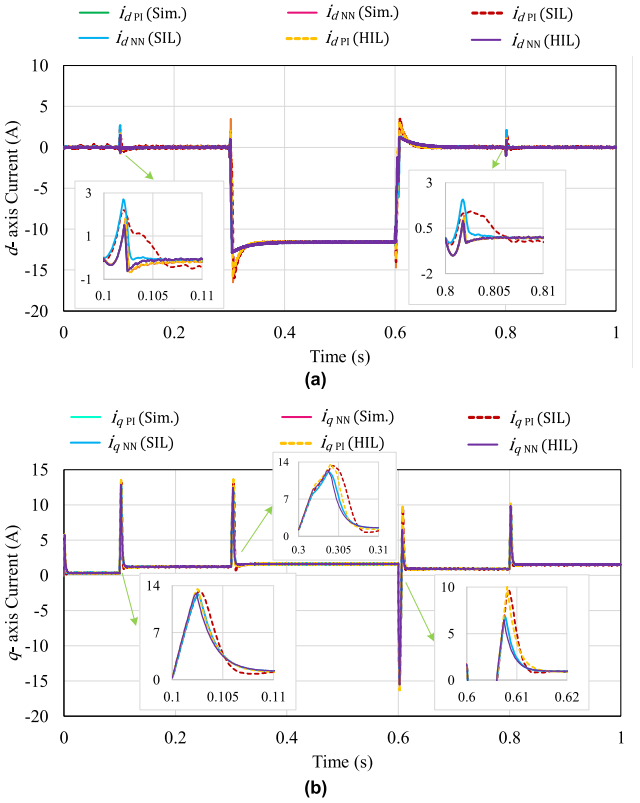


FIGURE 20. The d- and q-axis current regulation performance with the proposed speed controller and 30% decreased R_s, L_d and L_q under no-load. (a) d-axis. (b) q-axis.

TABLE 8. Average performance comparison of DNN and API controllers by mean squared error.

Control Method	Overshoot	Undershoot	Torque Ripple	Settling Time
API	3.7 (A)	3.4 (A)	9%	0.18 (s)
DNN	1.68 (A)	1.56 (A)	3%	0.11 (s)

the numerical MSE comparison between DNN-based and API controllers presented in Table 5, when the proposed DNN-based speed controller is implemented, the system experiences an average of 41.2% smaller overshoots and undershoots in the d- and q-axis current tracking profiles. As a result, the whole drive system would function more efficiently and the longevity of the drive and PMSM systems compared to the API speed controller would be positively affected.

An average numerical performance comparison is presented in Table 8. In the table, the amplitude of overshoots and undershoots in the current control loop, along with torque ripple and settling time at the time of each step change. The results are presented in terms of MSE. As demonstrated in the table, the proposed DNN-based controllers demonstrate a superior performance compared to that of API controller under the test scenarios described in section VI.

TABLE 9. Simulated computation time for the proposed DNN-Based controllers compared to API in 100 s execution period.

Controller		Computation time (s)
Current Control Loop	API	13.21
	DNN	27.65
Speed Control Loop	API	9.54
	DNN	16.87

C. IMPLEMENTATION COMPARISON BETWEEN THE PROPOSED DNN-BASED CONTROLLERS AND API

A detailed comparison of the computation times between the proposed DNN-based controllers and API controllers over a 100-second execution period is conducted. As indicated in Table 9, the computation time for the DNN-based current control loop is 27.65 s, compared to 13.21 s for the API. For the speed control loop, the DNN requires 16.87 s, while the API takes 9.54 s. While the DNN-based controllers require higher computation times, this increase is justified by the significant improvements in control accuracy, adaptability to system changes, and enhanced disturbance rejection capabilities, as previously detailed in the study. These advancements represent a substantial contribution to the field, demonstrating the potential of DNN-based controllers to optimize PMSM drive systems despite the increased computational requirements.

The research recognizes the importance of achieving an optimal balance between performance enhancement and cost-efficiency, particularly in the context of enhancing the general applicability and adoption rates of the proposed methodology. Although the integration of FPGAs or RTS into the framework may elevate the associated hardware expenses, it is believed that the robust performance offered by the proposed approach justifies the investment in specific applications where reliability and precision are paramount. Presented in the paper, the control method is also successfully implemented as HIL on an automotive grade controller board, which does not have any FPGA. The ultimate objective is to refine the proposed framework to ensure its broad accessibility and scalability. The result is discussed in detail and looks promising. Hence, the proposed method could be programmed onto a general-purpose controller board, such as TI or NXP boards.

VII. CONCLUSION

This research has developed and validated two dense neural network (DNN)-based controllers for permanent magnet synchronous motor (PMSM) drives, demonstrating exceptional adaptability and computational efficiency. These novel, robust, and efficient DNN-based speed and current controllers exhibit minimal dependency on parameters, addressing the challenges inherent in PMSM drive control with excellent performance. Through comprehensive testing, including Software-In-the-Loop and Hardware-In-the-Loop with a real-time simulator and automotive-grade controller board, the DNN-based control method has been proven superiority over traditional approaches. Based on the experimental results presented in the paper for both current and speed controllers the introduced method has shown over 41.2% reduction in average overshoots and undershoots in current and speed control tracking profiles compared to adaptive proportional-integral controllers, signifying an improvement in efficiency, accuracy, and adaptability in disturbance rejection. In addition, the proposed controllers are trained offline, hence they would not put a significant computational burden on the drive system. This research not only marks an advancement in PMSM drive technology by simplifying the advanced controller structure while capturing complex system dynamics but also paves the way for applying DNN-based control systems in various demanding applications, offering a scalable solution to nonlinear system challenges without the complexity of system tuning procedures.

REFERENCES

[1] S. Li, H. Won, X. Fu, M. Fairbank, D. C. Wunsch, and E. Alonso, "Neural-network vector controller for permanent-magnet synchronous motor drives: Simulated and hardware-validated results," *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 3218–3230, Jul. 2020.

[2] C. Xia, B. Ji, and Y. Yan, "Smooth speed control for low-speed high-torque permanent-magnet synchronous motor using proportional–integral–resonant controller," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2123–2134, Apr. 2015.

[3] Y. Han, C. Gong, L. Yan, H. Wen, Y. Wang, and K. Shen, "Multiobjective finite control set model predictive control using novel delay compensation technique for PMSM," *IEEE Trans. Power Electron.*, vol. 35, no. 10, pp. 11193–11204, Oct. 2020.

[4] Y. Zhou and G. Chen, "Predictive DTC strategy with fault-tolerant function for six-phase and three-phase PMSM series-connected drive system," *IEEE Trans. Ind. Electron.*, vol. 65, no. 11, pp. 9101–9112, Nov. 2018.

[5] G. Feng, C. Lai, and N. C. Kar, "A closed-loop fuzzy-logic-based current controller for PMSM torque ripple minimization using the magnitude of speed harmonic as the feedback control signal," *IEEE Trans. Ind. Electron.*, vol. 64, no. 4, pp. 2642–2653, Apr. 2017.

[6] C. Wang and Z. Q. Zhu, "Fuzzy logic speed control of permanent magnet synchronous machine and feedback voltage ripple reduction in flux-weakening operation region," *IEEE Trans. Ind. Appl.*, vol. 56, no. 2, pp. 1505–1517, Mar. 2020.

[7] S. Bhattacharjee, S. Halder, Y. Yan, A. Balamurali, L. V. Iyer, and N. C. Kar, "Real-time SIL validation of a novel PMSM control based on deep deterministic policy gradient scheme for electrified vehicles," *IEEE Trans. Power Electron.*, vol. 37, no. 8, pp. 9000–9011, Aug. 2022.

[8] K. A. Abuhasel, F. F. M. El-Sousy, M. F. El-Naggar, and A. Abu-Siada, "Adaptive RCMAC neural network dynamic surface control for permanent-magnet synchronous motors driven two-axis X-Y table," *IEEE Access*, vol. 7, pp. 38068–38084, 2019.

[9] L. Li, G. Pei, J. Liu, P. Du, L. Pei, and C. Zhong, "2-DOF robust H_∞ control for permanent magnet synchronous motor with disturbance observer," *IEEE Trans. Power Electron.*, vol. 36, no. 3, pp. 3462–3472, Mar. 2021.

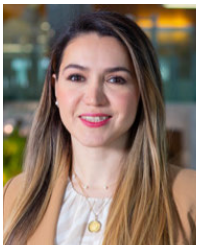
[10] Z. Li, C. Zang, P. Zeng, H. Yu, H. Li, and J. Bian, "Mixed H_2/H_∞ optimal control for three-phase grid connected converter," *Int. J. Electron*, vol. 104, no. 6, pp. 775–791, 2017.

[11] C.-K. Lin, T.-H. Liu, J.-T. Yu, L.-C. Fu, and C.-F. Hsiao, "Model-free predictive current control for interior permanent-magnet synchronous motor drives based on current difference detection technique," *IEEE Trans. Ind. Electron.*, vol. 61, no. 2, pp. 667–681, Feb. 2014.

[12] T. Türker, U. Buyukkeles, and A. F. Bakan, "A robust predictive current controller for PMSM drives," *IEEE Trans. Ind. Electron.*, vol. 63, no. 6, pp. 3906–3914, Jun. 2016.

[13] S. Masoudi, M. R. Soltanpour, and H. Abdollahi, "Adaptive fuzzy control method for a linear switched reluctance motor," *IET Electric Power Appl.*, vol. 12, no. 9, pp. 1328–1336, Nov. 2018.

[14] J. C. Ting and D. F. Chen, "Nonlinear backstepping control of synrm drive systems using reformed recurrent Hermite polynomial neural networks with adaptive law and error estimated law," *J. Power Electron.*, vol. 18, no. 5, pp. 1380–1397, 2018.
[15] Q. Wang, H. Yu, M. Wang, and X. Qi, "An improved sliding mode control using disturbance torque observer for permanent magnet synchronous motor," *IEEE Access*, vol. 7, pp. 36691–36701, 2019.
[16] A. K. Junejo, W. Xu, C. Mu, M. M. Ismail, and Y. Liu, "Adaptive speed control of PMSM drive system based a new sliding-mode reaching law," *IEEE Trans. Power Electron.*, vol. 35, no. 11, pp. 12110–12121, Nov. 2020.
[17] Y. Zuo, C. Lai, and K. L. V. Iyer, "A review of sliding mode observer based sensorless control methods for PMSM drive," *IEEE Trans. Power Electron.*, vol. 38, no. 9, pp. 11352–11367, Sep. 2023.
[18] Y. Sun, H. Qiang, L. Wang, W. Ji, and A. Mardani, "A fuzzy-logic-system-based cooperative control for the multielectromagnets suspension system of Maglev trains with experimental verification," *IEEE Trans. Fuzzy Syst.*, vol. 31, no. 10, pp. 3411–3422, Oct. 2023.
[19] F. Yang, X. Zhao, H. Jin, X. Wang, and X. Liu, "Improved deadbeat predictive current control with embedded resonant polynomial and disturbance observer for PMSM current distortion rejection," *IEEE J. Emerg. Sel. Topics Power Electron.*, vol. 12, no. 2, pp. 1934–1945, Apr. 2024.
[20] Y. Dai, L. Zhang, D. Xu, Q. Chen, and J. Li, "Development of deep learning-based cooperative fault diagnosis method for multi-PMSM drive system in 4WID-EVs," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1–13, 2024.
[21] Z. Li, X. Yang, and S. Zhao, "Flux linkage observation using an improved MRAS observer in case of uniform demagnetization of IPMSM," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1–10, 2024.
[22] B. Zhang, P. Niu, X. Guo, and J. He, "Fuzzy PID control of permanent magnet synchronous motor electric steering engine by improved beetle antennae search algorithm," *Sci. Rep.*, vol. 14, no. 1, p. 2898, Feb. 2024.
[23] Y. Yang, D. M. Vilathgamuwa, and M. A. Rahman, "Implementation of an artificial-neural-network-based real-time adaptive controller for an interior permanent-magnet motor drive," *IEEE Trans. Ind. Appl.*, vol. 39, no. 1, pp. 96–104, Jan./Feb. 2003.
[24] M. A. Rahman and M. A. Hoque, "On-line adaptive artificial neural network based vector control of permanent magnet synchronous motors," *IEEE Trans. Energy Convers.*, vol. 13, no. 4, pp. 311–318, Dec. 1998.
[25] J. Zhao, C. Yang, W. Dai, and W. Gao, "Reinforcement learning-based composite optimal operational control of industrial systems with multiple unit devices," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1091–1101, Feb. 2022.
[26] S. Gao, Y. Wei, D. Zhang, H. Qi, Y. Wei, and Z. Yang, "Model-free hybrid parallel predictive speed control based on ultralocal model of PMSM for electric vehicles," *IEEE Trans. Ind. Electron.*, vol. 69, no. 10, pp. 9739–9748, Oct. 2022.
[27] Y. Zhou, H. Li, and H. Zhang, "Model-free deadbeat predictive current control of a surface-mounted permanent magnet synchronous motor drive system," *J. Power Electron.*, vol. 18, no. 1, pp. 103–115, Jan. 2018.
[28] A. Fatemimoghadam, Y. Yan, L. V. Iyer, and N. C. Kar, "Permanent magnet synchronous motor drive using deep-neural-network-based vector control for electric vehicle applications," in *Proc. Int. Conf. Electr. Mach. (ICEM)*, Sep. 2022, pp. 2358–2364.
[29] S. Staroletov, "Architectural software-hardware co-modeling a real-world cyber-physical system: Arduino-based ArduPilot case," in *Proc. 30th Conf. Open Innov. Assoc. FRUCT*, Oulu, Finland, Oct. 2021, pp. 267–278.
[30] R. Bruni, R. Giacobazzi, R. Gori, and F. Ranzato, "A correctness and incorrectness program logic," *J. ACM*, vol. 70, no. 2, pp. 1–45, Apr. 2023.
[31] S. Foster, J. Baxter, A. Cavalcanti, J. Woodcock, and F. Zeyda, "Unifying semantic foundations for automated verification tools in isabelle/UTP," *Sci. Comput. Program.*, vol. 197, Oct. 2020, Art. no. 102510.
[32] A. B. Dehkordi, A. M. Gole, and T. L. Maguire, "Permanent magnet synchronous machine model for real-time simulation," in *Proc. Int. Conf. Power Syst. Transients*, Montreal, QC, Canada, Jun. 2005.
[33] K. Liu, C. Hou, and W. Hua, "A novel inertia identification method and its application in PI controllers of PMSM drives," *IEEE Access*, vol. 7, pp. 13445–13454, 2019.
[34] T. Li, X. Sun, M. Yao, D. Guo, and Y. Sun, "Improved finite control set model predictive current control for permanent magnet synchronous motor with sliding mode observer," *IEEE Trans. Transport. Electrific.*, vol. 10, no. 1, pp. 699–710, Mar. 2024.
[35] T. Li, X. Sun, G. Lei, Y. Guo, Z. Yang, and J. Zhu, "Finite-control-set model predictive control of permanent magnet synchronous motor drive systems—An overview," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 12, pp. 2087–2105, Dec. 2022.
[36] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Netw.*, vol. 4, no. 2, pp. 251–257, 1991.
[37] B. M. Wilamowski and J. D. Irwin, *Intelligent Systems*. Boca Raton, FL, USA: CRC Press, 2018.



ARMITA FATEMIMOGHADAM (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree with the University of Windsor. She is a Research Assistant with the Centre for Hybrid Automotive Research and Green Energy (CHARGE) Laboratories, University of Windsor, ON, Canada. She has collaborated as a Researcher with Magna International Inc., where she is focusing on software development applications to improve safety, reliability, and overall efficiency for autonomous electric vehicles. Her research interests include advancing AI-based methodologies for the precise control of electric motor drives and designing and developing the control strategies applied to electrical machines for electrified vehicle applications.

LAKSHMI VARAHA IYER (Senior Member, IEEE) received the B.Tech. degree in electronics and communication engineering from SASTRA University, Thanjavur, India, in 2009, and the M.A.Sc. and Ph.D. degrees in electrical and computer engineering from the University of Windsor, Windsor, ON, Canada, in 2011 and 2016, respectively.

He is currently a Senior Manager of Advanced Powertrain and Chassis—Americas with Magna International Inc. He is also an Adjunct Professor with the Department of Electrical and Computer Engineering, University of Windsor. He has been innovating in the area of electric machines and power electronic systems for electrified automotive and renewable energy applications, since 2008.

NARAYAN C. KAR (Senior Member, IEEE) received the B.S. degree in electrical engineering from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 1992, and the M.S. and Ph.D. degrees in electrical engineering from the Kitami Institute of Technology, Hokkaido, Japan, in 1997 and 2000, respectively.

He is currently a Professor with the Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON, Canada, where he is also the Tier 1 Canada Research Chair (CRC) of electrified vehicles. He is also the Founder and the Director of the CHARGE Laboratory. His current research interests include the analysis, design, and control of electric machines and drives for electric vehicle applications.