Deep Learning-based Control Perspective for Single-Phase Grid-connected Inverter Using Gated Recurrent Units

Slimane Sayah, Mouhoub Birane, Khalil Benmouiza

Materials, Energy Systems, Renewable Energies and Energy Management Laboratory (LMSEERGE), Aamar Telidji University of Laghouat, Laghouat-03000, Algeria

e-mails: s.sayah@lagh-univ.com; m.birane@lagh-univ.dz; k.benmouiza@lagh-univ.dz

Abstract: This paper introduces a novel approach to control single-phase grid-connected inverters (GCIs) using artificial intelligence (AI), specifically employing a deep learning-based method with Gated Recurrent Unit (GRU) networks. The proposed GRU-based controller is trained offline using TensorFlow and Keras libraries in Python, and is subsequently implemented for real-time applications. Comparative analysis between the GRU-based controller and the conventional PI controller reveals distinct advantages of the former, including improved transient response and reduced oscillations. Furthermore, the GRU-based controller demonstrates superior performance, reducing the total harmonic distortion (THD) and efficiently regulating current in the presence of varying grid conditions.

Keywords: Renewable sources; Grid-connected inverter; Artificial intelligence; Gated recurrent unit; Total harmonic distortion

1 Introduction

Nowadays, renewable energy sources, such as photovoltaic (PV) systems, are gaining increasing importance due to their availability and environmental friendliness [1]. These systems rely on sophisticated power electronic converter systems for seamless integration into the grid. Contemporary technological advancements have made it feasible to achieve a distributed setup of single-phase Grid-Connected Inverters (GCIs), offering superior efficiency and reliability [2-4]. In residential areas served by a single-phase distribution system, the integration of solar PV distributed generation (DG) plants with the smart grid commonly employs a single-phase GCI [5]. However, this integration process could potentially impact

the overall performance and integrity of the current injected into the grid. Therefore, researchers are proactively working to overcome these challenges, ensuring that technological progress not only enhances efficiency but also upholds the stability and reliability of the power infrastructure [6-7]. Among the challenges faced by researchers is the reduction of harmonics in the grid current generated by the interaction between the GCI and the grid or arising from non-linear loads in the system. Total Harmonic Distortion (THD) serves as the metric for measuring the extent of distortion in the injected current attributed to harmonics in the signal [8]. In grid-connected systems, the THD should be maintained at levels below 5% for optimal power quality [9]. Excessive harmonic distortion can lead to various issues such as increased losses, interference with communication systems, and degradation of power factor. Numerous control techniques exist to enhance overall performance, tolerate system disturbances, and mitigate THD within the desired range [5]. Conventional control methodologies, such as the proportional-integral (PI) controller have been widely used. This controller facilitates separate control of currents along the direct and transverse axis, enabling efficient tracking of DC signals. However, this approach exhibited certain limitations in power quality that could impact the system's performance and stability. Recently, AI controllers have garnered significant interest in the field of control system [10-16]. For instance, artificial neural network (ANN)-based control techniques have been widely studied for power electronics control. One study used recurrent neural networks (RNNs) to control grid-connected converters [17], while convolutional neural networks (CNNs) have been employed in grid-connected PV systems to enhance the quality of the current injected into the grid [18]. These studies underscore the effectiveness of AI-based control approaches. However, no research has yet explored the use of gated recurrent unit (GRU) networks for controlling single-phase grid-connected inverters (GCIs). The main contribution of this work is to introduce a GRU-based control approach for a single-phase GCI in electric power system applications. This proposed method gathers data from a well-tuned PI current controller via Simulink and employs it for training the GRU networks. The GRU is trained with an input window of prior data, aiming to minimize the disparity between the forecasted and the next measured value. Sequential methods forecast a single subsequent value based on the window of prior data. The complete training process is implemented in Python employing the TensorFlow and Keras libraries. The proposed GRU controller is then compared with the conventional PI controller and the CNN-based control approach presented in [18].

2 Methodology

In this section, we present our proposed methodology for implementing GRU neural networks to control single-phase GCI, as illustrated in Figure 1. The methodology begins by extracting multivariate time series data from the conventional PI

controller in Simulink. Next, in Python, the data are divided into a training dataset and a testing dataset. First, the GRU model is trained until it achieves optimal predictive capabilities. After training, the model's performance is evaluated using the testing dataset. Following this learning process in Python, the constructed GRU model is applied in Simulink for online implementation.

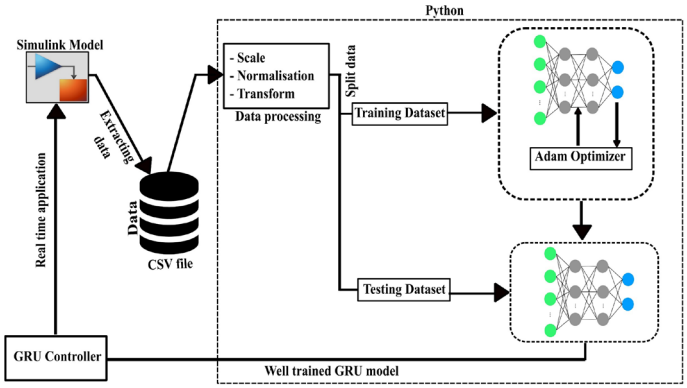


Figure 1
Methodological framework for implementing GRU neural networks in controlling single-phase GCI

3 Conventional Control Approach

3.1 Single-Phase GCI System Description and Mathematical Model

Figure 2 depicts the system structure of a single-phase GCI, which consists of four insulated gate bipolar transistors (IGBTs). The power circuit includes a DC voltage source, a single-phase GCI, *L*-type filters with internal resistance *R*, and the grid.

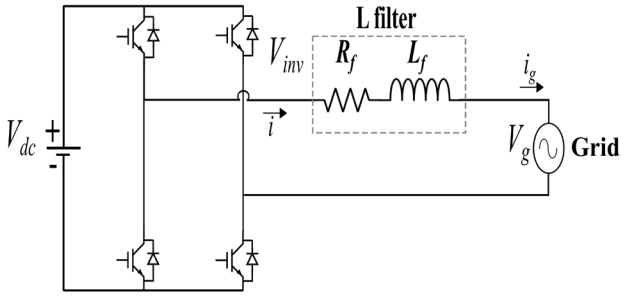


Figure 2
Single-phase GCI

3.2 Mathematical Model in the d-q Frame

To formulate the mathematical model of a single-phase system with an L filter in the stationary reference frame (α,β), it is necessary to generate an imaginary signal that is orthogonal to the original single signal. Since a single-phase system contains only one signal, a transport delay block is employed to create the imaginary signal β. This is achieved by delaying the primary signal α by one-fourth of the fundamental period. Therefore, the mathematical model of the system in the stationary reference frame can be expressed in (1).

$$\begin{cases} V_{inv\alpha} = L\frac{i_\alpha}{dt} + Ri_\alpha + V_{g\alpha} \\ V_{inv\beta} = L\frac{i_\beta}{dt} + Ri_\beta + V_{g\beta} \end{cases} \quad (1)$$

V_{inv}, i and V_g represent the inverter's output voltage, current and grid voltage, respectively. The dynamics of the GCI is described by the state-space equations in (2):

$$\begin{cases} \frac{dx}{dt} = Ax + Bu, \\ y = Cx \end{cases} \quad (2)$$

Where:

$$x = \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}, A = \begin{bmatrix} \frac{-R}{L} & 0 \\ 0 & \frac{-R}{L} \end{bmatrix}, B = \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & \frac{1}{L} \end{bmatrix}, u = \begin{bmatrix} V_{inv\alpha} - V_{g\alpha} \\ V_{inv\beta} - V_{g\beta} \end{bmatrix} \text{ and } C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Applying the Laplace transform to the dynamics, the laplace transfer function of the system is presented in (3):

$$\frac{i(s)}{V_{inv}(s) - V_g(s)} = \frac{1}{R_f + L_f s} \quad (3)$$

To convert (1) and (2) to a dq reference frame it is important to highlight a distinctive characteristic of the dq frame. In this frame, a space vector with an unchanging magnitude that rotates at the same speed as the frame, exhibits constant d and q components. Conversely, if this vector rotates at a different rate or exhibits a time-varying magnitude, it gives rise to pulsating elements. Consequently, within a dq reference frame undergoing rotation at the angular speed ω. (1) can be expressed as depicted in (4)[19]

$$\begin{cases} V_{invd} = L_f\frac{i_d}{dt} + R_f i_d + V_{gd} - \omega L i_q \\ V_{invq} = L_f\frac{i_q}{dt} + R_f i_q + V_{gq} + \omega L i_d \end{cases} \quad (4)$$

and state-space equations becomes as illustrated in (5):

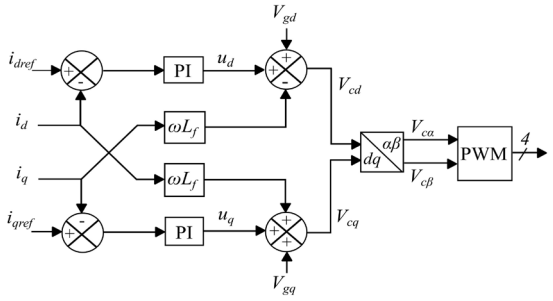
$$\begin{cases} \frac{dx}{dt} = A_c x + B_c u, \\ y = C_c x \end{cases} \tag{5}$$

Where:

$$x = \begin{bmatrix} i_d \\ i_q \end{bmatrix}, A_c = \begin{bmatrix} \frac{-R_f}{L_f} & \omega \\ -\omega & \frac{-R_f}{L_f} \end{bmatrix}, B_c = \begin{bmatrix} \frac{1}{L_f} & 0 \\ 0 & \frac{1}{L_f} \end{bmatrix}, u = \begin{bmatrix} V_{invd} - V_{gd} \\ V_{invq} - V_{gq} \end{bmatrix} \text{ and } C_c = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

3.3 Conventional PI Controller

The structure of the PI controller scheme for a single-phase GCI is illustrated in Figure 3. The output current from the inverter is decoupled into active and reactive currents i_d and i_q respectively, as expressed in (4). Each current is independently controlled through a PI controller, one controller regulates active current i_d, while the other focuses on controlling the reactive current i_q. Assuming a constant DC source voltage, the reference active and reactive currents are specified as user input commands. These reference signals are then compared to the actual d- and q-axis currents, and the controller's objective is to reduce the absolute root-mean-square (RMS) error. The output voltage control signals V_{cd} and V_{cq} are converted to the stationary frame $V_{c\alpha}$ and $V_{c\alpha}$ using (6). The $V_{c\alpha}$ signal is used for the generation of pulse width modulation (PWM) signal to control the GCI.



The structure of PI-based d-q current control scheme

$$\begin{cases} V_{c\alpha} = V_{cd}\cos\omega t - V_{cq}\sin\omega t \\ V_{c\beta} = V_{cd}\sin\omega t + V_{cq}\cos\omega t \end{cases} \tag{6}$$

The control law is given by (7)

$$u_{d,q}(t) = K_p e_{d,q}(t) + K_i \int e_{d,q}(t)dt \tag{7}$$

The Laplace transfer function for the PI controller is provided. in (8):

$$G_{PI}(s) = k_p + \frac{k_i}{s} \tag{8}$$

Where k_p is the proportional gain and k_i is the integral gain. The open-loop Laplace transfer function of the system incorporating the PI controller is obtained by multiplication (3) by (8). as shown in (9):

$$G_O(s) = G_{GCI}G_{PI}(s) = (k_p + \frac{k_i}{s})(\frac{1}{R_f + L_f s}) \tag{9}$$

The closed-loop Laplace transfer function of the whole system is given by (10)

$$\begin{cases} G_{cl}(s) = \frac{G_{GCI}G_{PI}(s)}{1+G_{GCI}G_{PI}(s)} \\ G_{cl}(s) = \frac{sk_p+k_i}{s^2L_f+s(R_f+k_p)+k_i} \end{cases} \tag{10}$$

The proportional and integral gains k_p and k_i are determined using the modulus optimum approach as presented in (11) and (12) respectively[20].

$$k_p = \omega_c T_i R_f \sqrt{(1 + T_a^2 \omega_c^2)} \tag{11}$$

$$k_i = \frac{k_p}{T_i} \tag{12}$$

Where ω_c is the cut off frequency, and T_i is the time constant of the system as illustrated in (13), and T_a the first order delay of the inverter, given by (14):

$$T_i = \frac{L_f}{R_f} \tag{13}$$

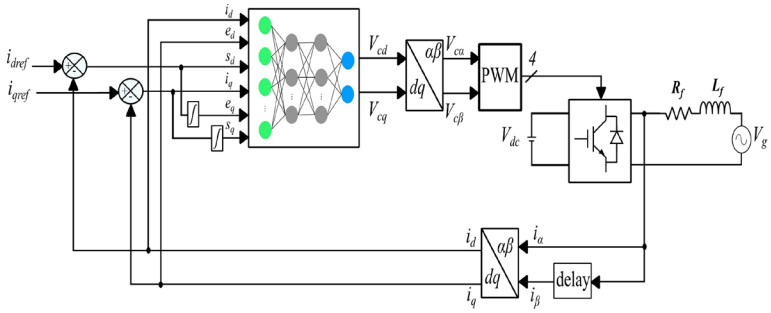
$$T_a = \frac{1}{2f_s} \tag{14}$$

4 Proposed GRU-based Control

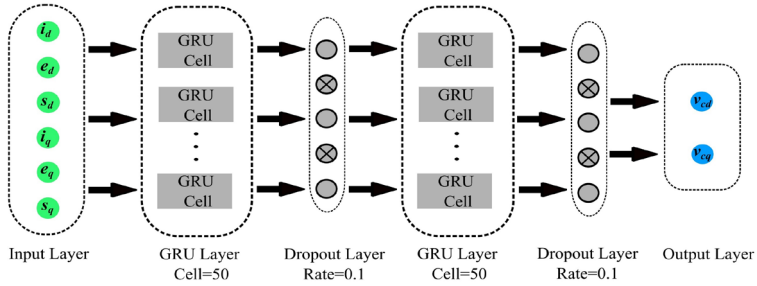
The proposed GRU based controller design for a single-phase GCI implements a fast inner current-loop control function [21] as shown in Figure 4, the controller is trained offline using data collected from the conventional PI-controller. The proposed GRU network architecture, as illustrated in Figure 5, consists an input layer, two GRU layers, each followed by a dropout layer, and a final output layer. The input layer contains six neurons that represent the measured currents i_{dq}, the error between the desired and actual currents e_{dq}, and the integrals of the error s_{dq} as expressed in (15).

$$s_{dq} = \int_0^t e_{dq}(t)\, dt. \tag{15}$$

The specific settings for the GRU layers and dropout layers are provided at the bottom of each corresponding layer in Figure 5. The output layer contains two neurons, representing the control signals V_{cd} and V_{cq}.



Proposed GRU based control architecture for single-phase GCI

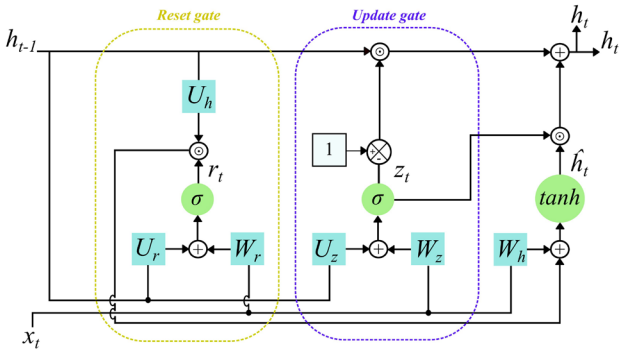


GRU network architecture

4.1 GRU Model

4.1.1 GRU Model Architecture

The GRU emerges as an effective solution to the vanishing gradient problem encountered in traditional RNNs. Through its inventive gating mechanisms [22], the GRU enables selective updates and resets within the hidden state. This distinctive feature allows the GRU to retain relevant data while discarding irrelevant information, thus optimizing the acquisition of long-term dependencies in the learning process. Figure 6 illustrates the basic architecture of the GRU cell which contains two specific gates: a reset gate, and an update gate. The reset gate allows the model to determine which portion of the previous information should be forgotten, while the update gate empowers the model to decide the degree to which past information should influence the update of the hidden state, thereby guiding its propagation into the future.



GRU cell structure

4.1.2 GRU Model Mathematical Formulation

The mathematical model that describes the behavior of the GRU involves various equations that describe the calculations happening at each time step [23]. First let's denote: h_t as the hidden state at time t, x_t as the input state at time t, r_t as the reset gate at time t, z_t as the hidden state at time t.

The equations used to calculate the update gate, reset gate, and hidden state of a GRU are as shown in (16), (17), (18), and (19):

1. Update Gate (z_t):

$$z_t = \sigma(W_z \cdot x_t + U_z \cdot h_{t-1}) \tag{16}$$

2. Reset Gate (r_t):

$$r_t = \sigma(W_r \cdot x_t + U_r \cdot h_{t-1}) \tag{17}$$

3. Candidate Hidden State ($\tilde{h}_t$):

$$\tilde{h}_t = \tanh(W_h \cdot x_t + (r_t \odot (U_h \cdot h_{t-1}))) \tag{18}$$

4. Hidden State Update (h_t):

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{19}$$

The symbol $\odot$ denotes Hadamard (element-wise) multiplication, σ and $\tanh$ represent the activation functions, sigmoid and hyperbolic tangent respectively. This equation combines h_{t-1} with $\tilde{h}_t$ based on z_t. If z_t is close to 1, it allows for more information from the candidate hidden state to be included; otherwise, it retains more of the previous hidden state. The matrices W_z, W_r, and W_h are the weight matrices associated with the update gate, reset gate, and candidate hidden state, respectively.

Algorithm for GRU-based Control of a Grid-Connected Inverter

1. Collect Data

Input matrix : $X \in \mathbb{R}^{M\times 6}, where\ X = \{[i_d, e_d, s_d, i_q, e_q, s_q]_t | t = 1, \dots M\}$

Target matrix : $Y \in \mathbb{R}^{M\times 2}, where\ Y = \{[u_d, u_q]_t \mid t = 1,2, \dots, M\}$.

Predicted control outputs: $\hat{Y} = \{\hat{u}_d, \hat{u}_q\}\ for\ given\ input\ data$

2. Data preprocessing

#Normalize the input and target matrices using Min-Max scaling:

For $x_j\ in\ X$ **do**

$$(x,y)_{j,norm} = \frac{(x,y)_j\ -\ (x,y)_j, min}{(x,y)_j, max\ -\ (x,y)_j, min}$$

End

3. Model Initialization

#Define GRU layers and dropout as follows

For $\boldsymbol{t\ = 1}$ **to** $\boldsymbol{M}$ **do**

#Compute the hidden state $h_t^{(1)}$

$$z_t = \sigma(W_z \cdot x_{t,norm} + U_z \cdot h_{t-1}^{(1)})$$
$$r_t = \sigma(W_r \cdot x_{t,norm} + U_r \cdot h_{t-1}^{(1)})$$
$$\tilde{h}_t^{(1)} = \tanh(W_h \cdot x_{t,norm} + (r_t \odot (U_h \cdot h_{t-1}^{(1)})))$$
$$h_t^{(1)} = (1 - z_t) \odot h_{t-1}^{(1)} + z_t \odot \tilde{h}_t^{(1)}$$

$\#with\ h_t^1 \in \mathbb{R}^{N_1}$, Where:$N_1 = 50\ units$

#Apply dropout layer to prevent overfitting

$$h_t^{(1,dropout)} = h_t^{(1)} \odot\ m$$

#$m \sim$ Bernoulli $(1 - p)$ is a binary mask tensor, where p is the dropout rate = 0.1

#Compute the hidden state $h_t^{(2)}$

$$z_t = \sigma(W_z \cdot h_t^{(1,dropout)} + U_z \cdot h_{t-1}^{(2)})$$
$$r_t = \sigma(W_r \cdot h_t^{(1,dropout)} + U_r \cdot h_{t-1}^{(2)})$$
$$\tilde{h}_t^{(2)} = \tanh(W_h \cdot h_t^{(1,dropout)} + (r_t \odot (U_h \cdot h_{t-1}^{(2)})))$$
$$h_t^{(2)} = (1 - z_t) \odot h_{t-1}^{(2)} + z_t \odot \tilde{h}_t^{(2)}$$

$\#with\ h_t^2 \in \mathbb{R}^{N_2}$, where: $N_2\ =\ 50\ units$

#Apply dropout layer to prevent overfitting

$$h_t^{(2,dropout)} = h_t^{(2)} \odot\ m$$

#Map the output using a dense layer

$$\hat{Y} = W.h_t^{(2,dropout)} + b$$

#Where $W \in \mathbb{R}^{2\times N_2}$ and $b \in \mathbb{R}^2$.

End

4.1.3 Training of GRU Controller

A total of $40{,}000$ data points were collected and divided into two parts: 75% for training data and 25% as the test set. The data collection process was executed in Simulink, where the total simulation duration was set to 2 seconds. The sampling time for the system was configured as $Ts = 0.05ms$. To compile the GRU model, adaptive momentum estimation (Adam) was selected as the optimizer and the loss function used was the mean squared error (MSE), with a learning rate of 0.001.

The Adam optimizer, an extension of stochastic gradient descent, has recently gained widespread acceptance for deep learning applications [24], [25]. The loss function computes the average squared difference between the actual value $y(t)$ and the target value $\tilde{y}(t)$, as presented in (20).

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} (y_i(t) - \tilde{y}_i(t))^2 \quad (20)$$

Figure 7 illustrates the flowchart of the GRU controller's training process, where the model is trained offline for 300 epochs. The learning curve, shown in Figure 8, demonstrates the successful training of the GRU model.

4.1.4 Computational Complexity

The computational complexity of the proposed GRU-based controller design is influenced by the architecture, training data, and optimization process. The network consists of two GRU layers, each with 50 units, and an input layer with 6 neurons, processing 30,000 training samples (75% of 40,000 data points). The time complexity for training the GRU model is dominated by the GRU layers, which is approximately $O(T.n.h^2)$, where T is the number of training samples (30,000), n is the number of time steps, and h is the number of GRU cells. This results in approximately 75 million operations per epoch. The Adam optimizer and MSE loss function add additional computational overhead, but their impact is relatively small compared to the GRU layers. Given the hardware setup of an Intel® Core™ i7-12650H Processor, 32 GB RAM, and an RTX 3060 GPU, the training process will be accelerated by the GPU, especially for operations like matrix multiplications and backpropagation. The 32 GB of RAM ensures smooth data handling without memory limitations, allowing efficient training and validation of the model. The overall training complexity is thus manageable with the provided hardware, ensuring fast model convergence.

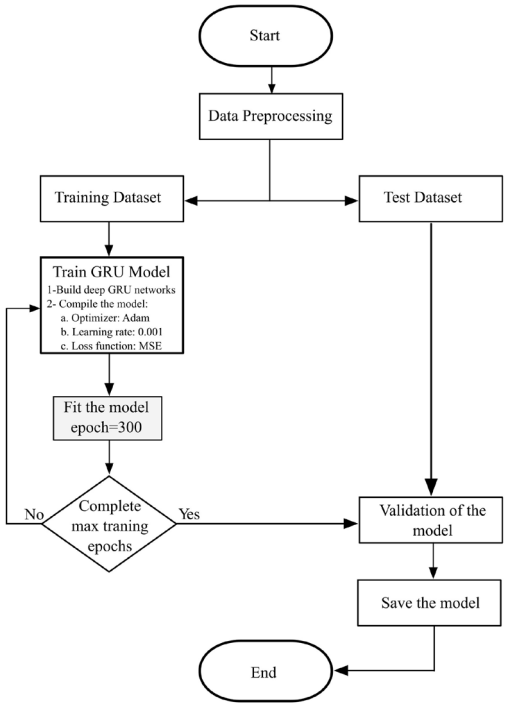


Figure 7
Flowchart of the training process for GRU-based controller

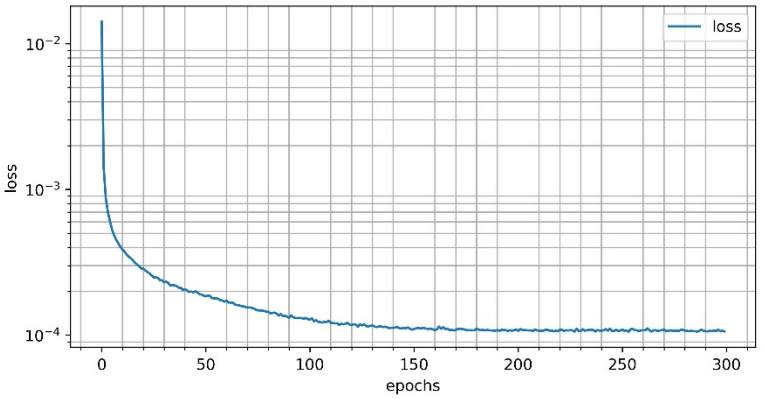


Figure 8
GRU-based controller learning curve

5 Simulation Results and Discussion

Table 1 provides details on the single-phase GCI system parameters, to assess the effectiveness of the proposed GRU controller and compare it with the conventional PI controller and CNN controller, a Simulink model has been created to conduct the necessary simulations.

Table 1
Single-phase GCI system parameters

Parameter	Symbol	Value	Unit
DC link voltage	V_{dc}	400	V
Grid voltage (rms)	V_g	230	V
Nominal grid frequency	f	50	Hz
Filter Resistance	R_f	0.19	Ω
Filter inductance	L_f	30	mH
Switching frequency	f_s	10	KHz

5.1 Assessment and Analysis of the Transient Response Under Standard Operating Conditions

To assess the transient response of the PI controller, CNN controller and the proposed GRU controller, two cases were tested. In the first case, a step change in the direct axis current (i_d) was initiated from 0 to 20 A, while the quadrature axis current (i_q) remained at zero A. This procedure aimed to observe the impact of altering the i_d step on i_q and to evaluate the transient performance of i_d when the reference changes. In the second case, the scenario was reversed: a step change in the quadrature axis current (i_q) was induced from 0 to 10 A, while the direct axis current (i_d) remained at zero A. Figure 10(a) and Figure 10(b) illustrate the first case, showcasing the step change in reference direct axis current under both the PI controller and the GRU controller, respectively.

Figure 11(a) and Figure 11(b) illustrate the second case, depicting the step change in reference quadrature axis current under both the PI controller and the GRU controller, respectively.

Table 2 provides an analysis for both scenarios of the obtained results, encompassing parameters such as settling time, overshoot, disturbance on the other current, and the time of the disturbance on the other current.

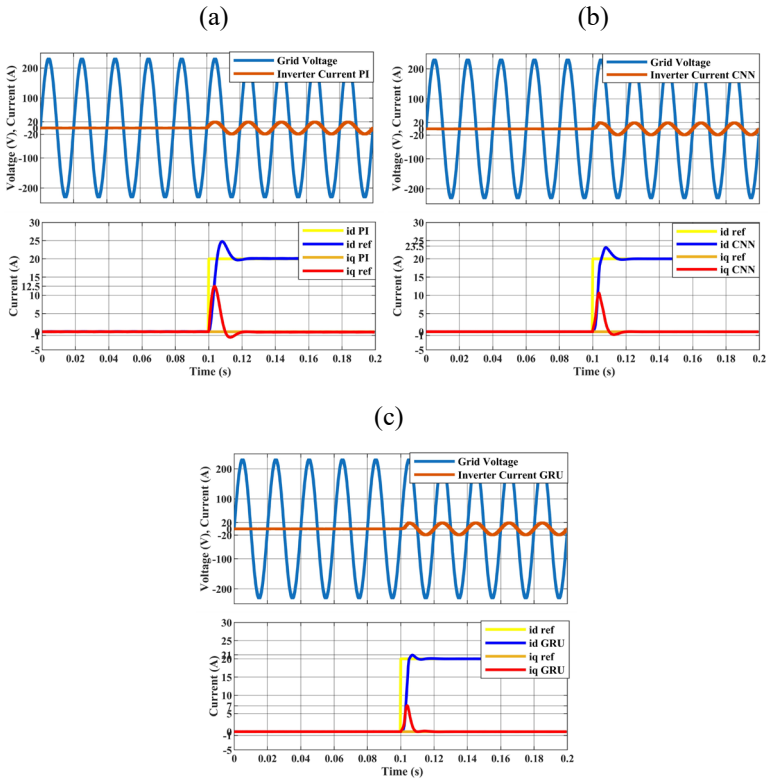
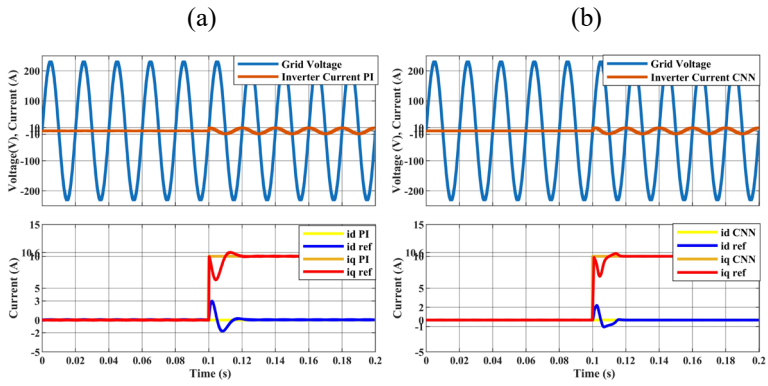


Figure 10

Direct and Quadrature Axis Current Response to a Step Change in Direct Current Reference: (a) PI controller, (b) CNN controller, (c) GRU controller



(c)

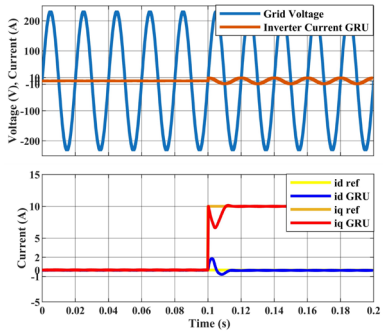


Figure 11

Direct and Quadrature Axis Current Response to a Step Change in Quadrature Current Reference: (a) PI controller, (b) CNN controller, (c) GRU controller

Table 2

Analysis of the transient response under standard operating conditions

Cases	Parameters	PI	CNN	GRU
Case 1: i_{d_ref} = 20 A i_{q_ref} = 0 A	Settling time [ms]	23	14	13
	Overshoot [%]	25	17.5	5
	Disturbance on i_q [A]	12.5	10	7.5
	Time of disturbance on i_q [ms]	20	9	9
Case 2: i_{d_ref} = 0 A i_{q_ref} = 10 A	Settling time [ms]	20	18	11
	Overshoot [%]	6	6	0.5
	Disturbance on i_d [A]	3	2.5	2
	Time of disturbance on i_d [ms]	24	16	14

In Scenario 1, where the i_d reference is 20 A and the i_q reference is 0 A, the PI controller has a settling time of 23 ms and 14 ms for CNN controller, while the GRU controller reduces this to 13 ms. This faster settling time indicates that the GRU controller reaches the desired value more quickly. The PI controller exhibits an overshoot of 25% and 17.5% for CNN controller, compared to the GRU controller's much lower overshoot of 5%, demonstrating significantly better transient response control with the GRU controller. When considering disturbances on i_q, the PI controller and CNN controller experience a pick of 12.5 A and 10 A respectively, whereas the GRU controller limits disturbances to 7.5 A, indicating better disturbance handling. Additionally, the GRU controller and CNN controller mitigate disturbances in 9 ms compared to the PI controller with 20 ms, leading to a more stable system. In Scenario 2, with an i_q reference of 10 A and an i_d reference of 0 A, the PI controller has a settling time of 20 ms and 18 ms for CNN controller, while the GRU controller improves this to 11 ms, again showing enhanced responsiveness. The overshoot is reduced with the GRU controller to 0.5%,

compared to the PI and CNN controllers with 6 % of overshoot, suggesting a significant improvement in transient response. For disturbances on i_d, the PI controller experiences 3 A and 2.5 A for CNN controller, while the GRU controller handles disturbances more effectively at 2 A. The time to resolve disturbances is also shorter with the GRU controller, at 14 ms versus the PI and CNN controllers with 16 ms and 24 ms respectively, contributing to system stability. Overall, the GRU controller consistently outperforms the PI controller and CNN controller across all metrics in both scenarios. The GRU controller achieves faster settling times, enhancing system responsiveness. It exhibits significantly lower overshoot, indicating much better transient performance. Additionally, it handles disturbances more effectively, both in terms of magnitude and duration. These improvements suggest that the GRU controller provides more robust and efficient control, leading to better overall performance.

5.2 Steady-State Performance Under Standard Operating Conditions

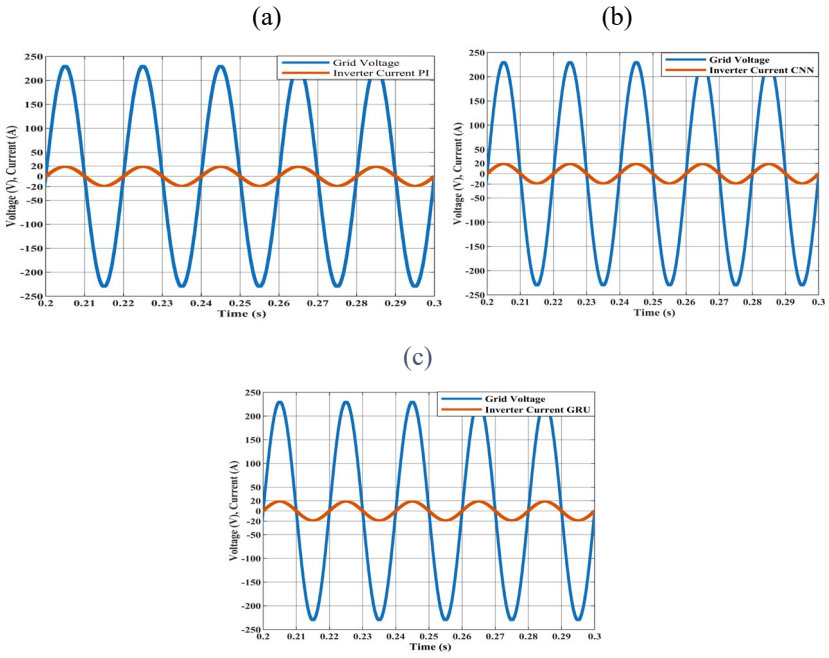


Figure 12

Grid voltage and inverter output current: Steady-state performance comparisons between conventional PI, CNN controller and proposed GRU controller. (a) PI controller, (b) CNN controller, (c) GRU controller

Figure 12 presents the simulation results for all controllers during the steady state performance from 0.2 s to 0.3 s. The inverter output current is shown in phase with the grid voltage. This setup uses a quadrature axis current reference of 0 A and a direct axis current reference of 20 A. These results demonstrate that both the direct and quadrature axis currents are well controlled to their reference values by the PI controller, CNN controller and the GRU controller, respectively. However, the GRU controller achieves a lower THD of 1.52 compared to the PI and CNN controllers with THD of 1.79 and 2.63 respectively, as illustrated in Figures 13(a), 13(b) and 13(c) respectively.

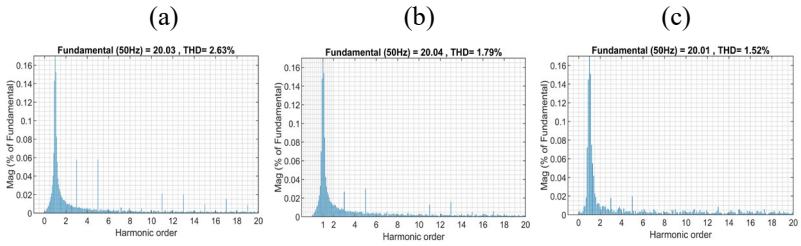


Figure 13

Total Harmonic Distortion (THD). (a) PI controller, (b) CNN controller, (c) GRU controller

5.3 Dynamic Performance Analysis Under Distorted Grid Voltage

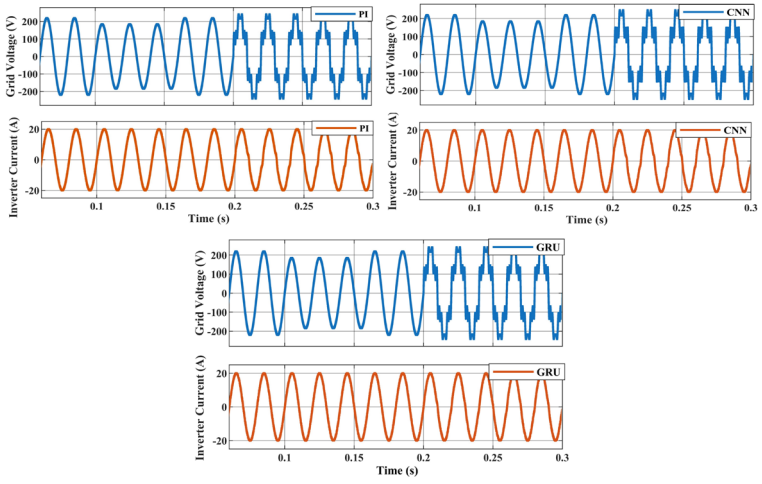


Figure 14

Simulation results for steady-state responses under a sudden change, grid-voltage sag, and grid-voltage distortion conditions

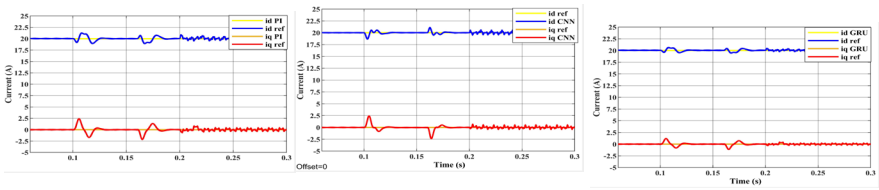


Figure 15

Simulation results for steady-state responses of direct and quadrature axis currents under a sudden change, grid-voltage sag, and grid-voltage distortion conditions

The robustness of the proposed GRU controller was evaluated under two challenging conditions: grid voltage sag and grid voltage distortion. A voltage sag was induced between 0.1 s and 0.16 s by reducing the voltage magnitude to 80% of its rated value. Following this, a grid voltage distortion scenario was applied between 0.2 s and 0.3 s, introducing 5th, 7th, 11th, and 13th harmonic orders with magnitudes equivalent to 15% of the fundamental component. Figure 14 and Figure 15 illustrates the performance of the GRU controller compared to the conventional PI and CNN controllers. The results show that the PI and CNN controllers are significantly affected by the voltage changes, particularly during the grid voltage distortion phase, leading to increased oscillations and degraded performance. In contrast, the GRU controller demonstrates superior harmonic compensation capability, effectively mitigating the adverse effects of voltage sags and distortions. This not only ensures stable operation but also enhances the overall grid performance during such conditions. The GRU controller's ability to handle grid abnormalities stems from its advanced learning capabilities, allowing it to effectively capture and adapt to non-linear dynamics. While the PI controller struggles due to its fixed-gain structure and the CNN controller's limitations in temporal dependency modelling, the GRU leverages its recurrent architecture to predict and respond to disturbances in real-time. The improved harmonic compensation highlights its potential to support grid stability, especially under varying operational conditions. This robustness makes the GRU controller a promising solution for modern power systems, where grid resilience is paramount in renewable energy integration and fluctuating grid conditions.

Conclusion

This paper introduced a GRU-based current controller designed for a single-phase grid-connected inverter. The methodology involved creating a GRU-based vector controller by collecting data from a conventional PI controller, followed by offline training in Python using the TensorFlow and Keras libraries. The fully trained GRU-based vector controller was then implemented for real-time applications. The simulation results demonstrated superior performance compared to the conventional PI controller and the CNN controller, including minimal oscillation and improve overall performance in response to change in system parameters.

References

[1] M. Birane, C. Larbes, A. Cheknane: Comparative study and performance evaluation of central and distributed topologies of photovoltaic system, Int J Hydrogen Energy, 2017, Vol. 42, No. 13, pp. 8703-8711

[2] Y. Xue, L. Chang, S. B. Kjær, J. Bordonau, T. Shimizu: Topologies of single-phase inverters for small distributed power generators: An overview, IEEE Trans Power Electron, 2004, Vol. 19, No. 5, pp. 1305-1314

[3] S. B. Kjaer, J. K. Pedersen, F. Blaabjerg: A review of single-phase grid-connected inverters for photovoltaic modules, IEEE Transactions on Industry Applications, 2005, Vol. 41, No. 5, pp. 1292-1306

[4] A. Anzalchi, A. Sarwat: Overview of technical specifications for grid-connected photovoltaic systems, Energy Conversion and Management, 2017, Vol. 152, pp. 312-327

[5] A. Chatterjee, K. B. Mohanty: Current control strategies for single phase grid integrated inverters for photovoltaic applications-a review, Renewable and Sustainable Energy Reviews, 2018, Vol. 92, pp. 554-569

[6] Y. Yang, K. Zhou, F. Blaabjerg: Current Harmonics from Single-Phase Grid-Connected Inverters-Examination and Suppression, IEEE J Emerg Sel Top Power Electron, 2016, Vol. 4, No. 1, pp. 221-233

[7] A. Menti, T. Zacharias, J. Milias-Argitis: Harmonic distortion assessment for a single-phase grid-connected photovoltaic system, Renewable Energy, 2011, Vol. 36, No. 1, pp. 360-368

[8] IEEE standard for interconnecting distributed resources with electric power systems: Institute of Electrical and Electronics Engineers, 2003

[9] J. H. Lee, Y. Ho Yoon, J. M. Kim: Analysis of IEC 61727 Photovoltaic (PV) systems Characteristics of the utility interface, International Journal of Internet Broadcasting and Communication, 2015, Vol. 7, No. 2, pp. 90-95

[10] A. Reda, R. Benotsmane, A. Bouzid, J. Vásárhelyi: A Hybrid Machine Learning-based Control Strategy for Autonomous Driving Optimization, Acta Polytechnica Hungarica, 2023, Vol. 20, No. 9, pp. 165-186

[11] I. A. Zamfirache, R.-E. Precup, R.-C. Roman, E. M. Petriu: Neural Network-based control using Actor-Critic Reinforcement Learning and Grey Wolf Optimizer with experimental servo system validation, Expert Systems with Applications, 2023, Vol. 225, p. 120112

[12] R.-C. Roman, R.-E. Precup, E. M. Petriu, A.-I. Borlea: Hybrid Data-Driven Active Disturbance Rejection Sliding Mode Control with Tower Crane

Systems Validation, Romanian Journal of Information Science and Technology, 2024, Vol. 27, No. 1, pp. 50-64

[13] S. Preitl, R.-E. Precup, J. Fodor, B. Bede: Iterative Feedback Tuning in Fuzzy Control Systems. Theory and Applications, Acta Polytechnica Hungarica, 2006, Vol. 3, No. 3, pp. 81-96

[14] T. Haidegger, L. Kovaacs, R. E. Precup, B. Benyo, Z. Benyo, S. Preitl: Simulation and control for telerobots in space medicine, Acta Astronautica, 2012, Vol. 81, No. 1, pp. 390-402

[15] R. E. Precup, S. Preitl, E. M. Petriu, J. K. Tar, M. L. Tomescu, C. Pozna: Generic two-degree-of-freedom linear and fuzzy controllers for integral processes, Journal of the Franklin Institute, 2009, Vol. 346, No. 10, pp. 980-1003

[16] B. Németh: Providing Guaranteed Performances for an Enhanced Cruise Control Using Robust LPV Method, Acta Polytechnica Hungarica, 2023, Vol. 20, No. 7, pp. 133-152

[17] X. Fu, S. Li: Control of Single-Phase Grid-Connected Converters with LCL Filters Using Recurrent Neural Network and Conventional Control Methods, IEEE Trans Power Electron, 2016, Vol. 31, No. 7, pp. 5354-5364

[18] S. Ramasamy, M. Perumal: CNN-based deep learning technique for improved H7 TLI with grid-connected photovoltaic systems, International Journal of Energy Research, 2021, Vol. 45, No. 14, pp. 19851-19868

[19] R. Teodorescu, M. Liserre, P. Rodríguez: Grid converters for photovoltaic and wind power systems, IEEE; Wiley, 2011

[20] A. D. Giles, L. Reguera, A. J. Roscoe: Optimal controller gains for inner current controllers in VSC inverters, IET Conference Publications, Institution of Engineering and Technology, 2015

[21] J. Dannehl, C. Wessels, F. W. Fuchs: Limitations of voltage-oriented PI current control of grid-connected PWM rectifiers with LCL filters, IEEE Transactions on Industrial Electronics, 2009, Vol. 56, No. 2, pp. 380-388

[22] H. Hua, M. Liu, Y. Li, S. Deng, Q. Wang: An ensemble framework for short-term load forecasting based on parallel CNN and GRU with improved ResNet, Electric Power Systems Research, 2023, Vol. 216, p. 109057

[23] J. Chung, C. Gulcehre, K. Cho, Y. Bengio: Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, arXiv, 2014

[24] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, B. Recht: The Marginal Value of Adaptive Gradient Methods in Machine Learning, arXiv, 2018

[25] D. P. Kingma, J. Ba: Adam: A Method for Stochastic Optimization, arXiv, 2014