# Modeling Methodology Based on Fast and Refined Neural Networks for Non-Isolated DC–DC Converters With Configurable Parameter Settings

Hanchen Ge, Zhihong Huang, *Student Member, IEEE*, and Zhicong Huang, *Member, IEEE*

***Abstract*— Compared with conventional physics-based methods, e.g., analytical modeling and numerical modeling, data-driven methods can extract input-to-output relationships from the data without much prior knowledge of the physical system, thus showing great potential in modeling power electronics (PE) converters with complex switching behaviors and configurable parameter settings. Previous data-driven PE circuit modeling approaches are mostly based on sequential neural networks, and their execution speed suffers from large sequential lengths due to a high sampling rate for high modeling accuracy. Moreover, modeling of refined singular ripples is missing and configurable parameter settings are not available in these data-driven modeling approaches. To address the above-mentioned issues, this paper proposes a hybrid physics-informed machine learning (ML) method to model the non-isolated DC-DC converters. The approach empirically decomposes the output signals into transient large signals and periodic small signals. For transient large signals, a fully-connected neural network (NN) is used to map circuit parameters with system characteristics, such that configurable circuit parameter settings are allowed. For periodic signals, a long short-time memory (LSTM) network together with convolutional neural network (CNN) is used to accelerate the simulation by predicting signal features in the compressed latent space. A buck converter with configurable parameter settings is modeled by the proposed hybrid physics-informed ML method. Periodic ripples are successfully generated, while execution speed is about 10 times faster than that of conventional numerical methods.**

***Index Terms*— DC–DC converter, physics-informed machine learning, modeling, signal decomposition.**

## NOMENCLATURE

| | |
|---|---|
| $\mathcal{F}(\boldsymbol{x}(t);\,\boldsymbol{p}), \boldsymbol{a}$ | Ideal model of DC-DC converters and initial conditions. |
| $\overline{\mathcal{F}}(\boldsymbol{x}(t);\,\boldsymbol{p})$ | Transient signal model of DC-DC converters. |
| $\widetilde{\mathcal{F}}(\boldsymbol{x}(t);\,\boldsymbol{p})$ | Periodic signal model of DC-DC converters. |
| $\boldsymbol{s}(t), c_s$ | Output signal and its channel number. |
| $\boldsymbol{x}(t), c_x$ | Excitation input signal and its channel number. |
| $\boldsymbol{p}, c_p$ | Parameter input and its channel number. |
| $\boldsymbol{u}_k$ | Concatenation of parameter and excitation inputs, in discrete form. |
| $\boldsymbol{h}_{\text{zs}}(t;\,\boldsymbol{p})$ | Zero-state unit impulse response of a system with parameter $\boldsymbol{p}$. |
| $\boldsymbol{h}_{\text{zi}}(t;\,\boldsymbol{p})$ | Zero-input unit response of a system with parameter $\boldsymbol{p}$. |
| $\boldsymbol{h}(t;\,\boldsymbol{p}), \boldsymbol{p}_{\text{imp}}$ | Concatenation of $\boldsymbol{h}_{zs}$ and $\boldsymbol{h}_{zi}$, and its representation in vector, also regarded as the representation of linear system characteristics. |
| $\overline{s}(t)$ | Transient output signal. |
| $\widetilde{s}(t), \widetilde{s}_{\text{raw}\ k}$ | Periodic output signal and its preprocessed form after discretization, subsection and resampling. |
| $\boldsymbol{o}_k, c_o$ | Latent representations of $\widetilde{s}_{\text{raw}\ k}$, and its channel number. |
| $\alpha$ | Amplitude of $\widetilde{s}(t)$. |
| $\gamma$ | Resample rate for $\widetilde{s}(t)$. |
| $f_{\text{ar}}, f_{\text{amp}}$ | LSTM and fully connected NN in the periodic signal model. |
| $f_{\text{DNN}}$ | Fully-connected NN in the transient signal model. |
| $G, Q, D$ | Generator, Encoder, and Discriminator network. |
| $w, k_p$ | Size of the perceptive field of CNNs and the number of signal peaks in each perceptive field. |
| $N, E, S$ | Shape of data in tensors, represents the batch size, channel number, and sequence length respectively. |
| $n, m$ | Order of circuits, the number of switching states in each switching period. |
| $n_v, n_b, n_s$ | Number of nodes, branches and PE switches in a circuit. |
| $r$ | Signal resolution. |

Manuscript received 30 November 2022; revised 24 January 2023; accepted 27 February 2023. Date of publication 2 March 2023; date of current version 13 June 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 52007067 and in part by the Natural Science Foundation of Guangdong Province under Grant 2022A1515011581 and Grant 2023A1515011623. This article was recommended by Guest Editor C.-T. Cheng. *(Corresponding author: Zhicong Huang.)*

Hanchen Ge and Zhicong Huang are with the Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, Guangzhou 510006, China (e-mail: zhiconghuang@scut.edu.cn).

Zhihong Huang is with the Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, Macau, China (e-mail: zhihonghuang2016@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JETCAS.2023.3251692.

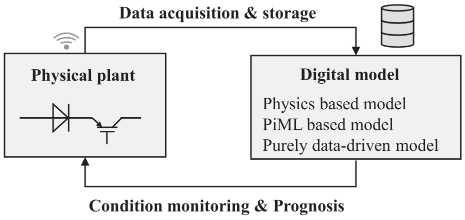
Digital Object Identifier 10.1109/JETCAS.2023.3251692

# I. Introduction

THE real-time performance of modeling and simulation of Power Electronics (PE) converters has been always in hot pursuit. At present, Industry 4.0 application scenarios such as the automatic design of power electronic circuits and digital twins have put forward higher requirements for the fast speed of circuit modeling and simulation [1]. However, commercially-used numerical simulation methods are not fast enough, especially when modeling PE converters with a high sampling rate, strong nonlinearity, complex and variable topology, and control logic [2], [3]. Thus, the real-time requirements are still challenging the numerical methods. In recent years, data-driven modeling methods are showing great advantages in speed and have been used in many fields such as electromagnetic field simulation [4]. Moreover, large physical data can be collected thanks to the rapid development of industrial IoT, which supports data-driven modeling methods, as is shown in Fig. 1(a). Therefore, this paper explores the possibility of using data-driven methods for PE modeling.

Three categories of modeling methods for PE converters are comprehensively summarized in Fig. 1(b). In general, there are physics-based, physics-informed data-driven, and purely data-based methods [5]. The physics-based method, like the analytical and numerical methods, assumes that PE and mathematical fundamentals are prior knowledge for building and solving system equations respectively. The purely data-based methods, like fully-connected NNs, rely on a large number of data and find the relationship between the input and output without knowing any physics [6]. The physics-informed data-driven methods, like neural operators, merge the aforementioned methods to lower the requirements of data and make the predictive result comply with the physics [7]. Moreover, based on computational structures, each method can be further classified into sequential and operator structures [8]. The operator structure maps the initial conditions directly into outputs, while the sequential structure solves the problem step by step.

As shown in the first column of Fig. 1(b), the physics-based approach includes numerical methods and analytical methods. Numerical methods are mostly based on sequential structures, to be specific, step-by-step ordinary differential equations (ODEs) solvers [9], [10]. Such sequential structures can be considered as a cause-and-effect chain that can emulate the real world, which is a significant characteristic of numerical methods. With the sequential structures, the numerical methods are flexible enough to model any PE converters regardless of whether they are linear or not. However, the sequential structures suffer from low speed because the solving process is step by step and cannot be made parallel, especially when a high sampling rate is required for high modeling accuracy [11]. On the contrary, analytical methods, including the time domain method and the complex frequency domain method, are to symbolically solve the ODEs of the circuits and get explicit solutions. Such symbolic methods are fast and stable for linear circuits, as those explicit solutions can map the initial conditions directly into the result. However, it is difficult to symbolically solve nonlinear ODEs, and the nonlinear circuits have to get linearized, resulting in accuracy degradation. Moreover, the complicated switching behaviors within the PE circuit always change the circuit topology, which leads to variations of ODE forms. The speed of the analytical method will suffer from the frequent changes of ODEs.



(a) Modeling of real plants for digital twins.

| Physics | Physics-based | Physics-informed ML | Purely Data-driven | Data |
|---|---|---|---|---|
| **Sequential Structure** | Numerical ODE Solvers [8] | Message Passing Neural PDE Solvers [7] | RNN; LSTM [15];<br>Transformer Decoder [12] | |
| **Operator Structure** | Analytical Methods | Fourier Neural Operators [3];<br>Physics-Informed NN [6] | DNN;<br>CNN [17] | |

(b) An overview of modeling methods.

Fig. 1. Modeling of physical plants for digital twins and an overview of modeling methods.

Data-driven methods can be categorized into operator and sequential structures. Currently, networks with operator structures have seldom been applied in circuit modeling, while data-driven modeling for PE converters is most based on sequential structures. Sequential networks, such as RNN, LSTM and Transformer Decoder [12], [13], assume time invariance and their forms are very similar to the discrete state equations of circuits and the numerical iterative equations of ODEs [14]. Although some studies have carried out data-driven methods based on sequential structures to model the electrical signals [1], [15], [16], [17], [18], [19], it should be pointed out that, they are based on series and step-by-step calculations, which is the bottleneck of speeding up. In some other research fields, such as speech recognition and generation, operator structures like CNN [20] are more common with speech signals quantized into finite-size segments [21], [22]. Text-to-speech (TTS) studies like [23] use generative adversarial nets (GAN) [24] for speech generation, which means voices can be generated based on speech features such as phonogram, duration, and tunes. In some studies, sequential natural language processing (NLP) NNs such as BERT [25] can be added to the TTS network. Such a hybrid structure not only achieves speed advantages but also has wider receptive fields. Moreover, they are not prone to gradient vanishing problems, and the output results are more refined. Although the data-driven methods are generally faster, their generalization performance and accuracy are still challenging. For the ODEs of PE converters, the inputs and outputs are distributed in Banach spaces, and it is difficult to grasp the overview of the ODE only by finite-sized function samples [26]. For example, a PE converter can operate in a wide frequency range, and thus a large number of data with a wide range of frequencies are required if a time-domain sequential network is used for modeling [1]. The final prediction will lose generalization performance outside the frequency range of the dataset. Moreover, to support configurable circuit parameters, a whole family of ODEs is involved, and the amount of data will be further expanded, which significantly makes the network training difficult.

To be summarized, there are three major challenges when using data-driven methods for PE converter modeling as follows.

(a) A commonly-used network structure requires too much data, which makes it difficult to handle the configurable parameter settings.
(b) A sequential network structure is based on step-by-step calculations, which is the bottleneck of speeding up.
(c) The limited receptive field results in a loss of details such as periodic signal components.

To address these issues, some certain prior knowledge can be introduced to train the network, which is usually called physics-informed machine learning (PiML) [5], [27], [28]. PiML methods incorporate the physics-based and purely data-driven methods by introducing bias, such as inductive bias, observation bias, and learning bias, via the network structure, observation data, and loss function. Such that, the generalization performance can be enhanced with datasets reduced. In recent years, PiML has achieved remarkable results in fitting ODEs or partial differential equations and is successfully applied in simulations of electromagnetic field, fluid and thermal heat [7], [8], [29], [30]. Compared with the above-mentioned simulations, the PE converters systems are more complex in terms of their nonlinearity and variable ODEs.

To the best of our knowledge, modeling of PE converters with PiML methods has not yet been reported, which motivates us to investigate its feasibility. In this paper, a hybrid physics-informed machine learning (PiML) method is proposed to model the non-isolated DC-DC converters. The approach empirically decomposes the output signals of non-isolated DC-DC converters into transient large signals and periodic small signals. For transient large signals, a fully-connected NN is used to map circuit parameters with system characteristics, such that configurable circuit parameter settings are allowed. For periodic signals, LSTM together with CNN is used to accelerate the simulation by predicting signal features in the compressed latent space. A buck converter with configurable parameter settings is modeled by the proposed hybrid PiML method to validate its efficacy. Periodic ripples are successfully generated, while execution speed is about 10 times faster than that of conventional numerical methods.

The structure of this paper is as follows: Section II reveals the critical signal issues of modeling PE converters and proposes to decompose signals into transient and periodic components. Section III details the modeling approaches for the transient and periodic signals. Section IV validates the proposed method, and a conclusion is given in Section V.

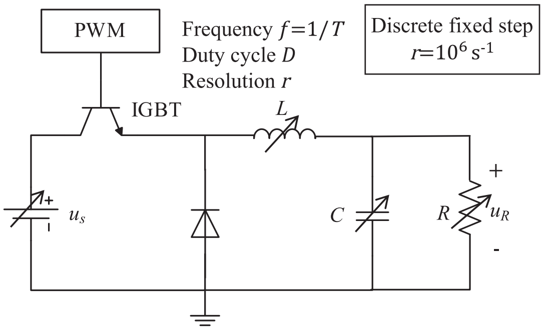


Fig. 2. The modeled Buck converter.

## II. Problem Statement and Signal Preprocessing

### A. Problem Statement

The basic notations of this paper are defined in the nomenclature, in which vectors are in the form of lowercase bold italic letters. In addition, notations for the special Buck converter example are defined in Fig. 2.

In general, an ideal mathematical model for a DC-DC converter can be illustrated by $\mathcal{F}(\boldsymbol{x}(t);\boldsymbol{p})$ and initial conditions $\boldsymbol{a}$:

$$\boldsymbol{s}(t)=\mathcal{F}(\boldsymbol{x}(t);\boldsymbol{p}),\ \text{and} \tag{1}$$

$$\boldsymbol{s}(0)=\boldsymbol{a}, \tag{2}$$

where $\mathcal{F}(\boldsymbol{x}(t);\boldsymbol{p})$ is the ideal mathematical model of DC-DC converter systems, which is a family of operators that map between the function spaces. $\boldsymbol{p}$ is the parameter input, which determines the system's inherent characteristics and is fixed through time. $\boldsymbol{x}(t)$ is the excitation input, which does not affect the inherent characteristics. $\boldsymbol{s}(t)$ are the output signals (states).

In addition, two assumptions are made for the mathematical model of the DC-DC converter system $\mathcal{F}(\boldsymbol{x}(t);\boldsymbol{p})$:

(a) All the resistors, capacitors and inductors in the system are linear components.
(b) The switching pattern of the system is periodical, and the number of switching states is finite.

The object of this paper is to propose a data-driven method that models DC-DC converters, which fits the mathematical model $\mathcal{F}(\boldsymbol{x}(t);\boldsymbol{p})$ using NNs and maps the inputs $\boldsymbol{x}(t)$ and $\boldsymbol{p}$ into the outputs $\boldsymbol{s}(t)$.

As shown in Fig. 2, a Buck converter is taken as an example to illustrate the proposed NN-based modeling methodology. The mathematical model of the Buck converter can also be expressed as (1) and (2), where the excitation input $\boldsymbol{x}(t)$, the parameter input $\boldsymbol{p}$ and the system outputs $\boldsymbol{s}(t)$ are given as

$$\boldsymbol{s}(t)=[u_C(t),i_L(t)]^{\mathrm{T}}, \tag{3}$$

$$\boldsymbol{x}(t)=[u_s(t),D(t)]^{\mathrm{T}},\ \text{and} \tag{4}$$

$$\boldsymbol{p}=[R,L,C,T]^{\mathrm{T}}. \tag{5}$$

This modeling methodology can also be extended to the rest of typical non-isolated DC-DC converters, such as Boost, Buck-Boost, Cuk, Sepic and Zeta converters.

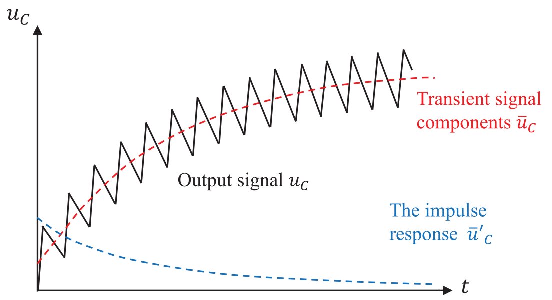


Fig. 3. An example of the output $u_C(t)$ and the decomposed transient components under step excitation inputs and zero states.

### B. Signal Decomposition

In the non-isolated DC-DC converters, the power switch periodically changes the circuit topology and chops the output signals into singular segments. Taking the Buck converter as an example, the output signal $u_C(t)$ is depicted in Fig. 3 with the circuit parameters fixed and the initial system states set as zero. Such a singular signal has ultra-wide frequency and amplitude ranges, which is challenging to model by an end-to-end NN.

Therefore, a signal decomposition scheme is proposed. In this scheme, the outputs are decomposed into a slow-changing transient signal component $\bar{\boldsymbol{s}}(t)$ and a singular periodic signal component $\tilde{\boldsymbol{s}}(t)$, given by

$$\boldsymbol{s}(t) = \bar{\boldsymbol{s}}(t) + \tilde{\boldsymbol{s}}(t), \tag{6}$$

where the periodic signal component $\tilde{\boldsymbol{s}}(t)$ results from the switching events, and the transient signal component $\bar{\boldsymbol{s}}(t)$ results from the states of the energy storage elements like inductors and capacitors. The characteristics of these two signal components are different, i.e., the former is periodic, singular, and fast-changing; the latter is aperiodic, differentiable, and slow-changing. Hence, different modeling methods are used to model these two components separately, given by

$$\bar{\boldsymbol{s}}(t) = \bar{\mathcal{F}}(\boldsymbol{x}(t); \boldsymbol{p}), \text{ and} \tag{7}$$

$$\tilde{\boldsymbol{s}}(t) = \tilde{\mathcal{F}}(\boldsymbol{x}(t); \boldsymbol{p}). \tag{8}$$

It should be noted that such a signal decomposition method is different from conventional methods such as Fast Fourier Transform (FFT) [31] and Empirical Mode Decomposition (EMD) [32]. The difference is that the differentiable and non-periodic transient components like exponential signal are fully reserved for post-processing.

A general decomposition procedure is given as follows, and a pseudo-code for the proposed signal decomposition is shown in Algorithm 1.

**Algorithm 1** The Proposed Signal Decomposition Method

```
Input: An 1xS array of discrete signal $\boldsymbol{s}(t)$, the switching
       period array 1xn $T$
Output: (n+1)xS arrays of discrete signal $\bar{\boldsymbol{s}}(t), \tilde{\boldsymbol{s}}(t)$
for i ← 1 to n do
    peak_indices ← scipy.signal.find_peaks($\boldsymbol{s}(t)$)
    For a peak detection failure, set peak_indices by T[i] in
    equal difference
    Initialize sample_indices, segment_ave
    for j in peak_indices do
        Subsect $\boldsymbol{s}(t)$ into segments by peak_indices
        sample_indices[j] ← the middle index of segment j
        segment_ave[j] ← the average of segments
    end for
end for
```

(a) The exact form of the output signal $\boldsymbol{s}(t)$ is first derived. An $n$th-order DC-DC converter with $m$ states in each switching period can be expressed by a group of linear ODEs:

$$\begin{cases} \mathcal{L}_{n1}(\boldsymbol{s}(t)) = \lambda_1 \boldsymbol{x}(t), & t \in (kT, kT + t_1) \\ \mathcal{L}_{n2}(\boldsymbol{s}(t)) = \lambda_2 \boldsymbol{x}(t), & t \in (kT + t_1, kT + t_2) \\ \cdots \\ \mathcal{L}_{nm}(\boldsymbol{s}(t)) = \lambda_m \boldsymbol{x}(t), & t \in (kT + t_{m-1}, (k+1)T) \end{cases} \tag{9}$$

where $\mathcal{L}_{nm}$ are linear differential operators, $\lambda$ are constants, $t_i$ is the moment when the $i$th switching event happens. The solutions of the $i$th ODE include a general solution $\boldsymbol{s}_i(\text{initial\_condition}, t)$ and a particular solution $\boldsymbol{s}_i^*(\text{inhomogeneous\_term}, t)$. The solutions in the period $k$ are given by

$$\boldsymbol{s}(t) = \begin{cases} \boldsymbol{s}_1(\boldsymbol{s}(kT), t') + \boldsymbol{s}_1^*(\lambda_1 \boldsymbol{x}, t'), & t' \in (0, t_1] \\ \boldsymbol{s}_2(\boldsymbol{s}(kT + t_1), t' - t_1) + \boldsymbol{s}_2^*(\lambda_2 \boldsymbol{x}, t' - t_1), \\ \qquad t' \in (t_1, t_2] \\ \cdots \\ \boldsymbol{s}_m(\boldsymbol{s}(kT + t_m), t' - t_m) + \boldsymbol{s}_m^*(\lambda_m \boldsymbol{x}, t' - t_m), \\ \qquad t' \in (t_m, T] \end{cases} \tag{10}$$

where $t' = t + kT$.

(b) The transient signal component $\bar{\boldsymbol{s}}(t)$ is then determined. The value of $\bar{\boldsymbol{s}}(t)$ at the midpoint of each period is determined as the average throughout the period, given by

$$\bar{\boldsymbol{s}}(kT + \frac{1}{2}T) = \frac{1}{T}\int_{kT}^{(k+1)T} \boldsymbol{s}(t)\mathrm{d}t \tag{11}$$

Since the value of $\bar{\boldsymbol{s}}$ at $t = kT + 0.5T$ are determined, the full $\bar{\boldsymbol{s}}(t)$ can be obtained by the cubic spline interpolation. Moreover, since $\mathcal{L}_n$ is linear, its solution $\boldsymbol{s}(t)$ is correlated linearly with the excitation inputs $\boldsymbol{x}$, and so does $\bar{\boldsymbol{s}}(t)$ as it correlated linearly with $\boldsymbol{s}(t)$:

$$\bar{\boldsymbol{s}}(t) \propto \boldsymbol{x}(t). \tag{12}$$

(c) The period signal component $\tilde{\boldsymbol{s}}(t)$ is finally calculated according to (6).

$$\tilde{\boldsymbol{s}}(t) = \boldsymbol{s}(t) - \bar{\boldsymbol{s}}(t) \tag{13}$$

To be specific, the Buck converter is taken as an example. The form of its output signal $\bar{u}_R(t) = \bar{u}_C(t)$ can be approximately derived. $C = 0$ is assumed for simplification. According to the ON and OFF states of the power switch, the Buck converter can be expressed as a piecewise ODE corresponding to (9), given by

$$\frac{L}{R}u'_R + u_R = \begin{cases} u_s, & t \in (kT, kT + t_{\text{on}}] \\ 0, & t \in (kT + t_{\text{on}}, (k+1)T] \end{cases} \tag{14}$$

where $t_{\text{on}} = DT$. The solutions of (14) are given by

$$u_R(t) = \begin{cases} u_R(kT)e^{-\frac{t-kT}{\tau}} + u_s(1 - e^{-\frac{t-kT}{\tau}}), \\ \qquad\qquad t \in (kT, kT + t_{\text{on}}] \\ u_R(kT + t_{\text{on}})e^{-\frac{t-kT-t_{\text{on}}}{\tau}}, \\ \qquad\qquad t \in (kT + t_{\text{on}}, (k+1)T] \end{cases} \tag{15}$$

where $\tau = \sqrt{L/R}$. It can be observed that the solutions are consistent with (10).

Based on Lagrange's mean value theorem, the average of each period can be approximated by the average of peaks and troughs. Assuming that $u_R(kT + t_{\text{on}}) = u_{\text{peak } k}$ and $u_R(kT) = u_{\text{trough } k}$, with (11), the average can be calculated as

$$\begin{aligned} \overline{u}_R(kT + \frac{1}{2}T) &= \frac{1}{T}\int_{kT}^{(k+1)T} u_R(t)\mathrm{d}t \\ &\approx \frac{u_{\text{peak } k} + u_{\text{trough } k}}{2} \\ &\approx u_s D(1 - e^{-\frac{T}{\tau}k})\frac{1}{2}(1 + e^{-\frac{T-t_{\text{on}}}{\tau}}) \end{aligned} \tag{16}$$

where

$$\begin{aligned} u_{\text{peak } k} &= u_{\text{peak } k-1}e^{-\frac{T}{\tau}} + u_s(1 - e^{-\frac{t_{\text{on}}}{\tau}}) \\ &= u_s(1 - e^{-\frac{t_{\text{on}}}{\tau}})\frac{1 - e^{-\frac{T}{\tau}k}}{1 - e^{-\frac{T}{\tau}}} \\ &\approx u_s D(1 - e^{-\frac{T}{\tau}k}) \end{aligned} \tag{17}$$

$$u_{\text{trough } k+1} \approx u_s D(1 - e^{-\frac{T}{\tau}k})e^{-\frac{T-t_{\text{on}}}{\tau}}. \tag{18}$$

Finally, the full signal of $\overline{u}_R(t)$ can be obtained by interpolation, while the period signal component can be calculated by

$$\widetilde{u}_R(t) = u_R(t) - \overline{u}_R(t). \tag{19}$$

## III. Proposed Modeling Methodology

### A. Modeling Approach For Transient Signals

As was depicted in Section II-B, the transient signals are aperiodic, slow-changing, and proportional to the excitation inputs according to (12). Introducing the biases about linear systems can make the model understand the intrinsic characteristics of the system and therefore improve the generalization performance of the NNs.

As is shown in Fig. 4, the transient signal model $\overline{\mathcal{F}}(\boldsymbol{x}(t); \boldsymbol{p})$ has introduced such inductive biases about linear systems. A group of simple fully-connected NNs is trained to map the parameter inputs into the linear system characteristics. The outputs are computed according to the linear system characteristics, excitation inputs and initial conditions.

In this section, the building and predicting processes of the transient signal model are illustrated respectively, and then a Buck converter example is presented.

*1) Three Steps to Building the Transient Signal Model:* First is to generate the output signal dataset that reflects the linear system characteristics. Such characteristics can be depicted by the zero-state unit impulse responses $\boldsymbol{h}_{\text{zs}}(t)$ and the zero-input unit state responses $\boldsymbol{h}_{\text{zi}}(t)$. For an $n$th-order DC-DC converters with $c_x$ channels of excitation inputs, the $\boldsymbol{h}_{\text{zs}}(t)$ and $\boldsymbol{h}_{\text{zi}}(t)$ can be generated by numerical methods or experimental measurements, as given by

$$\boldsymbol{h}_{\text{zs}}(t; \boldsymbol{p}) = \overline{\mathcal{F}}'(\varepsilon(t)\boldsymbol{I}_c; \boldsymbol{p}), \overline{s}(0) = \boldsymbol{0}_{n\times c} \tag{20}$$

$$\boldsymbol{h}_{\text{zi}}(t; \boldsymbol{p}) = \overline{\mathcal{F}}(\boldsymbol{0}_n; \boldsymbol{p}), \overline{s}(0) = \boldsymbol{I}_n \tag{21}$$

where $\boldsymbol{h}_{\text{zs}}(t)$ is a $n \times c_x$ function matrix and is generated by differentiating the unit step responses, $\boldsymbol{h}_{\text{zi}}(t)$ is a $n \times n$ function matrix, they can be concatenated into a $n \times (c_x + n)$ function matrix $\boldsymbol{h}(t)$. $\boldsymbol{I}_j$ is the $j$th-order unit matrix, and $\varepsilon(t)$ is the unit step signal.

Then, the system characteristics $\boldsymbol{h}(t)$ are compressed into finite dimensions vectors $\boldsymbol{p}_{\text{imp}}$ since they are infinite-length signals and cannot be directly treated as the output of NNs. Two methods of compression for $\boldsymbol{h}(t)$ are proposed.

(a) The first method is to simply discard the close-to-zeros part and downsample into a finite-length vector, as $\boldsymbol{h}(t)$ usually tends to diminish to zero because of the inevitable energy losses in a source-free system. This process is specified by $p(\boldsymbol{h}(t))$

$$\boldsymbol{p}_{\text{imp}} = p(\boldsymbol{h}(t)), \quad \text{if} \lim_{t\to+\infty} \boldsymbol{h}(t) = \boldsymbol{0} \tag{22}$$

(b) Another method is to fit the impulse response $\boldsymbol{h}(t)$ into analytic parameters, in which the order of the linear system has to be known. e.g. For a 2nd order linear ODE, in the over-damping case, the $\boldsymbol{h}(t)$ are in the form of

$$\begin{aligned} h(t; \boldsymbol{p}_{\text{imp}}) &= C_1e^{\lambda_1 t} + C_2e^{\lambda_2 t} + C_3, \\ \boldsymbol{p}_{\text{imp}} &= [C_1, C_2, C_3, \lambda_1, \lambda_2]^{\mathrm{T}}. \end{aligned} \tag{23}$$

The gradient descent method under PyTorch is used to find out those parameters $\boldsymbol{p}_{\text{imp}}$, choosing the Adam optimizer and MSE loss. The key that guarantees the iteration convergence is to set a proper initial value (0,0,0,0.1,1, respectively) and learning rate(0.01 for $\lambda s$ and 0.1 for $Cs$).

Finally, fully-connected NNs are trained to map parameter inputs into system characteristics. There are overall $n \times (c_x + n)$ fully-connected NNs to be trained, as given by

$$\boldsymbol{p}_{\text{imp}} = f_{\text{DNN}}(\boldsymbol{p}). \tag{24}$$

Commonly-used training methods, such as Adam optimizer and mean square error (MSE) loss, can be used for the proposed fully-connected NNs.

*2) The Predicting Process of the Transient Signal Model:* First, the system characteristics $\boldsymbol{h}(t)$ are predicted by the fully connected NNs, specified as

$$\begin{aligned} \boldsymbol{h}(t) &= p^{-1}(\boldsymbol{p}_{\text{imp}}) \\ &= p^{-1}(f_{\text{DNN}}(\boldsymbol{p})) \end{aligned} \tag{25}$$

where $p^{-1}$ is the decompress process.

Next, since the system characteristics $\boldsymbol{h}_{\mathrm{zs}}(t)$ and $\boldsymbol{h}_{\mathrm{zi}}(t)$ are predicted, the overall outputs of the system can be calculated as a superposition of a variety of responses, given by:

$$\begin{aligned}\overline{\boldsymbol{s}}(t) &= \boldsymbol{h}_{\mathrm{zs}}(t;\, \boldsymbol{p}) * \boldsymbol{x}(t) + \boldsymbol{h}_{\mathrm{zi}}(t;\, \boldsymbol{p})\overline{\boldsymbol{s}}(0),\\ \overline{\boldsymbol{s}}(0) &= \boldsymbol{a}\end{aligned} \tag{26}$$

where $*$ is the signal convolution operator that follows the rules of matrix multiplication. $\boldsymbol{x}$ is a $c_x \times 1$ function vector, and $\boldsymbol{a}$ is a $n \times 1$ vector.

In (26), the parameter inputs $\boldsymbol{p}$ are all fixed. However, variable parameter inputs such as a sudden step change of load have to be considered as well. The variable parameter inputs would impose a dynamic variation on the system characteristics while the system states remain static. In the proposed method, the parameter changes are treated as events that update system characteristics and restart the signal convolution. For example, if there is a sudden step of load specified as $\boldsymbol{p} \xrightarrow{t_e} \boldsymbol{p}'$, the outputs can be computed as

$$\overline{\boldsymbol{s}}(t) = \begin{cases} \boldsymbol{h}_{\mathrm{zs}}(t;\, \boldsymbol{p}) * \boldsymbol{x}(t) + \boldsymbol{h}_{\mathrm{zi}}(t;\, \boldsymbol{p})\overline{\boldsymbol{s}}(0), & t \in (0, t_e)\\ \boldsymbol{h}_{\mathrm{zs}}(t;\, \boldsymbol{p}') * \boldsymbol{x}(t) + \boldsymbol{h}_{\mathrm{zi}}(t;\, \boldsymbol{p}')\overline{\boldsymbol{s}}(t_e), & t \in (t_e, +\infty) \end{cases} \tag{27}$$

*3) Modeling the Buck Converter as an Example:* To be specific, the Buck converter is again taken as an example. The basic equations of the transient model for the Buck converter are given as follows, which corresponds to (20), (21) and (26).

$$\begin{cases} [h_{C\mathrm{zs}}(t), h_{L\mathrm{zs}}(t)]^{\mathrm{T}} = \overline{\mathcal{F}}'(\varepsilon(t);\, \boldsymbol{p}), \;\; \boldsymbol{a} = [0, 0]^{\mathrm{T}}\\ [h_{C1\mathrm{zi}}(t), h_{L1\mathrm{zi}}(t)]^{\mathrm{T}} = \overline{\mathcal{F}}(0;\, \boldsymbol{p}), \;\; \boldsymbol{a} = [1, 0]^{\mathrm{T}}\\ [h_{C2\mathrm{zi}}(t), h_{L2\mathrm{zi}}(t)]^{\mathrm{T}} = \overline{\mathcal{F}}(0;\, \boldsymbol{p}), \;\; \boldsymbol{a} = [0, 1]^{\mathrm{T}} \end{cases} \tag{28}$$

$$\begin{aligned}[\overline{u}_C(t), \overline{i}_L(t)]^{\mathrm{T}} = {} & [h_{C\mathrm{zs}}(t), h_{L\mathrm{zs}}(t)]^{\mathrm{T}} * u_s(t)\\ & + [h_{C1\mathrm{zi}}(t), h_{L1\mathrm{zi}}(t)]^{\mathrm{T}} u_C(0^-)\\ & + [h_{C2\mathrm{zi}}(t), h_{L2\mathrm{zi}}(t)]^{\mathrm{T}} i_L(0^-),\end{aligned} \tag{29}$$

where initial conditions $\boldsymbol{a} = [u_C(0), i_L(0)]^{\mathrm{T}}$, $\boldsymbol{h}_{\mathrm{zs}}(t)$ is a $2 \times 1$ function matrix and $\boldsymbol{h}_{\mathrm{zi}}(t)$ is a $2 \times 2$ function matrix for the Buck converter model.

Suppose that the load resistance $R$ changes to $R'$ at the moment $t_e$:

$$\boldsymbol{p} = [R, L, C, T] \xrightarrow{t_e} \boldsymbol{p}' = [R', L, C, T], \tag{30}$$

when $t \in (0, t_e)$, the outputs $[u_C(t), i_L(t)]$ can be computed by (29). And since the states $u_C(t_e), i_L(t_e)$ are known, when $t \in (t_e, +\infty)$, the outputs could be computed as

$$\begin{aligned}[\overline{u}_C(t), \overline{i}_L(t)]^{\mathrm{T}} = {} & [h'_{C\mathrm{zs}}(t), h'_{L\mathrm{zs}}(t)]^{\mathrm{T}} * u_s(t)\\ & + [h'_{C1\mathrm{zi}}(t), h'_{L1\mathrm{zi}}(t)]^{\mathrm{T}} u_C(t_e)\\ & + [h'_{C1\mathrm{zi}}(t), h'_{L1\mathrm{zi}}(t)]^{\mathrm{T}} i_L(t_e),\end{aligned} \tag{31}$$

### B. Modeling Approach for Periodic Signals

The periodic signals result from complicated switching events and range widely in frequencies and amplitudes with high sampling rates. It's difficult for common sequential NNs to generate long sequences without gradient vanishing problems, and to generalize across a wide domain of inputs. In addition, the speed of prediction is also a major challenge. Hence, learning bias about frequencies and amplitudes has to be introduced.

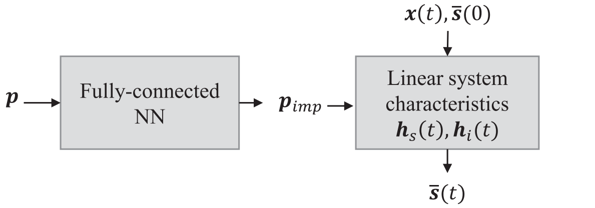

$\boldsymbol{x}(t), \overline{\boldsymbol{s}}(0)$
$\boldsymbol{p}$
Fully-connected NN
$\boldsymbol{p}_{imp}$
Linear system characteristics $\boldsymbol{h}_s(t), \boldsymbol{h}_i(t)$
$\overline{\boldsymbol{s}}(t)$

Fig. 4. Structure of the proposed transient signal model.

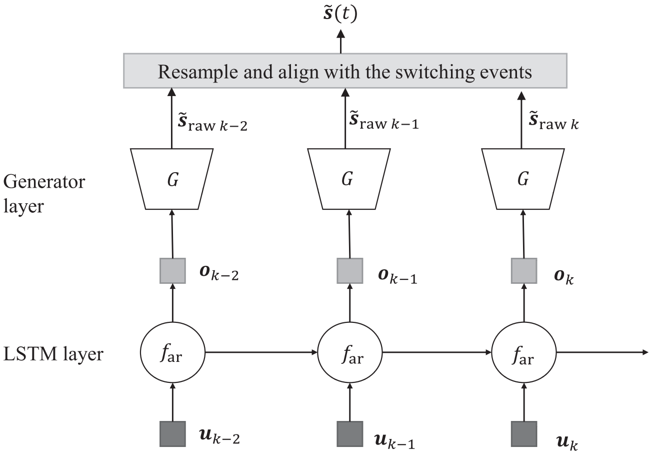


Fig. 5. Structure of the proposed periodic signal model.

In this section, the NN structure of the proposed periodic signal model is introduced first, and the processes of building and predicting are also illustrated respectively.

*1) The NN Architecture of the Periodic Signal Model:* As shown in Fig. 5, a multi-layer architecture that consists of an LSTM layer and a set of parallel upsampling CNNs (also called generators) is proposed to model the periodic signals. In the periodic model, the excitation inputs and parameter inputs are treated equally and concatenated into $\boldsymbol{u}_k$, which is different from the transient model. LSTM $f_{\mathrm{ar}}$ is used to generate latent periodic signal features $\boldsymbol{o}$, such as amplitude and shape. The generators $G$ then convert those latent representations into raw periodic signal chunks $\tilde{\boldsymbol{s}}_{\mathrm{raw}\;k}$ concurrently.

In the proposed NN structure, LSTM is chosen rather than other sequential NNs such as Transformer Decoder to accelerate the speed of predictions. Even though LSTM has the potential to model infinite dimension vectors (signals), its memory (receptive field) is not long enough to gain a full overview of signals, especially at a high resolution. To address this issue, CNN [20] is used to embed chunks of signals into latent representations. Transposed CNN, which is a special kind of upsampling CNN, is used to recover the chunk of signal from its latent representations. With the LSTM working in latent space, the CNNs are able to generate high-resolution signals chunk-by-chunk by predicting signal features in the compressed latent space, thus could be faster than single-layer sequential networks.

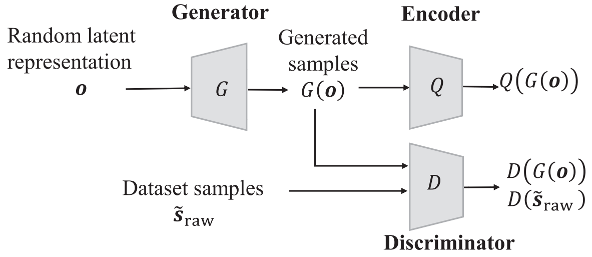


Fig. 6. The training method of InfoGAN.

*2) Three Steps to Building the Periodic Signal Model:* First is to generate the training set with random parameter and excitation inputs $\boldsymbol{u}_k$. The output signals generated by numerical methods or experiments are preprocessed before getting used as the training data.

(a) Periodic signals in the training set are resampled into a fixed frequency $f_{\text{train}}$ according to the receptive field of CNNs $w$, in which the number of periods in each receptive field $k_p$ is fixed, given by

$$f_{\text{train}} = \frac{rk_p}{w}. \tag{32}$$

(b) The signals are zoomed vertically into the same amplitude, with their original amplitudes logged as $\alpha$. This step could be skipped in some cases where the amplitude range is small.

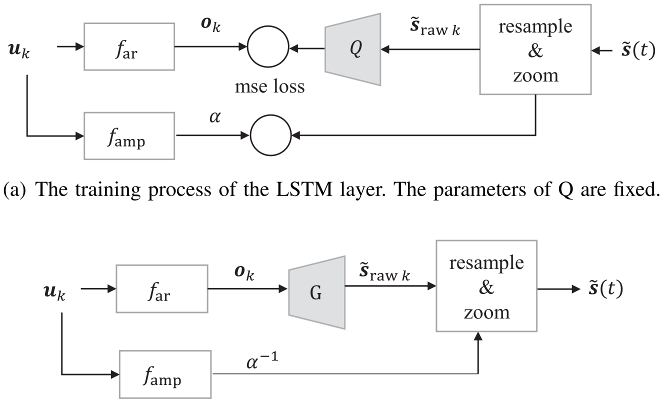
(c) The signals are chopped into equilong chunks $\tilde{\boldsymbol{s}}_{\text{raw }k}$, in which the chunk length is equal to the CNN receptive field $w$.

Then, The CNN generators are then trained based on an unsupervised technique called InfoGAN [33]. InfoGAN is an information-theoretic extension of the original GAN which encourages it to learn interpretable and meaningful representations, making the training process easier. As is shown in Fig. 6, InfoGAN consists of a Generator $G(\boldsymbol{o})$, a Discriminator $D(\tilde{\boldsymbol{s}}_{\text{raw}})$, and an Encoder $Q(\tilde{\boldsymbol{s}}_{\text{raw}})$. $G$ is a transpose CNN, while others are normal CNNs. The $G$ tries to generate samples that make $D$ not discriminable based on random noise $\boldsymbol{o}$. $D$ and $Q$ are CNNs that share the first few layers. $D$ discriminates whether the sample is generated by $G$ or from the dataset. $Q$ tries to restore $\boldsymbol{o}$ from the samples that $G$ generated, in order to make sure that $\boldsymbol{o}$ is not lost while generation, such that, samples that are not relevant to the latent representations won't be generated (over-fitting). The training process of the InfoGAN is as Algorithm 2.

Finally, The LSTM layer $f_{\text{ar}}$ is trained based on the trained encoders $Q$, as is illustrated in Fig. 7(a) and Algorithm 3. An additional fully-connected NNs $f_{\text{amp}}$ is also trained to predict $\alpha$ if the data has been zoomed vertically.

**Algorithm 2** Minibatch Stochastic Gradient Descent Training of InfoGAN. N Is Batch Size

```
for $i = 1$ to training iterations do
    Sample a batch of $N$ vectors $\boldsymbol{o}$ from random distribution as a latent representation.
    Get a batch of $N$ samples $\tilde{\boldsymbol{s}}_{\text{raw}}$ from the dataset.
    Update $\theta_D$ by ascending the entropy loss:
        $\nabla_{\theta_D} \text{ave}[\log D(\tilde{\boldsymbol{s}}_{\text{raw}}) + \log(1 - D(G(\boldsymbol{o})))]$
    Update $\theta_G$ by ascending the entropy loss:
        $\nabla_{\theta_G} \text{ave}[\log(1 - D(G(\boldsymbol{o})))]$
    Update $\theta_G$ & $\theta_Q$ by ascending the MSE loss:
        $\nabla_{\theta_{G,Q}} \text{ave}[\text{mse}(\boldsymbol{o}, Q(G(\boldsymbol{o})))]$
end for
```



(a) The training process of the LSTM layer. The parameters of Q are fixed.

(b) The predicting process of the LSTM layer.

Fig. 7. The training and predicting process of the LSTM layer.

TABLE I
RANGE OF PARAMS IN THE DATASETS

| **Param** | $u_s$(V) | $D$(%) | $f$(kHz) | $R$(Ω) | $L$(mH) | $C$(μF) |
|---|---|---|---|---|---|---|
| **Range** | 50-100 | 10-90 | 5-20 | 1-10 | 1-10 | 1-10 |

*3) The Predicting Process of the Periodic Signal Model:* The predicting process is an inversion of training, as illustrated in Fig. 7(b). First, the input sequence $\boldsymbol{u}_k$ is fed into LSTM $f_{\text{ar}}$ and turned into the latent representations $\boldsymbol{o}$. The amplitude $\alpha$ is also predicted. Then, $\tilde{\boldsymbol{s}}_{\text{raw}}$ is generated by $G$, and zoomed back into the original amplitudes $\alpha$. Finally, these signal segments are resampled into $\tilde{\boldsymbol{s}}(t)$ to be aligned with the periodical switching events. This process can be specified by

$$\boldsymbol{o}_k = f_{\text{ar}}(\boldsymbol{u}_k), \tag{33}$$

$$\alpha_k = f_{\text{amp}}(\boldsymbol{u}_k) \tag{34}$$

$$\tilde{\boldsymbol{s}}_{\text{raw }k} = G(\boldsymbol{o}_k)\alpha_k, \tag{35}$$

$$\tilde{\boldsymbol{s}}(t) = \text{resample}(\tilde{\boldsymbol{s}}_{\text{raw}}, f/f_{\text{train}}) \tag{36}$$

## IV. Experimental Verification

### A. Data Acquisition and Preprocessing

MATLAB/Simulink is used to simulate a Buck converter with parameter settings given in Table I, and the simulated results are sampled to generate datasets in mat 7.3 formats. The output signals $u_C$ and circuit parameters $\boldsymbol{p}$ are sampled in the size of $(N, E, S) = (8000, 1, 90000)$ and $(N, E) = (8000, 5)$ respectively, where $N$, $E$ and $S$ are the sample number, channel number, and sequence length respectively.

**Algorithm 3** Minibatch Stochastic Gradient Descent Training of LSTM

```
Fix $\theta_Q$ as Q was already trained.
for $i = 1$ to training iterations do
    for (a batch of data $\boldsymbol{s}$, label $\boldsymbol{u}$) in dateset do
        $\boldsymbol{o} \leftarrow Q(\boldsymbol{s})$
        $\widehat{\boldsymbol{o}} \leftarrow f_{\mathrm{ar}}(\boldsymbol{u})$
        Update $\theta_{\mathrm{ar}}$ by descending the MSE loss:
            $\nabla_{\theta_{\mathrm{ar}}} \mathrm{ave}[\mathrm{mse}(\boldsymbol{o}, \widehat{\boldsymbol{o}})]$
    end for
end for
```

Three different training settings are configured to generate and record data of $u_C$ and $\boldsymbol{p}$, and they are as follows.

(a) The parameter inputs $\boldsymbol{p}$ are set as random, while the $u_s$ input is set as step signal.
(b) The initial voltage of the capacitor is set as $u_C(0) = 50V$.
(c) The initial current of the inductor is set as $i_L(0) = 1A$.

Data processing includes 5 steps, and they are as follows. First, the outputs of $u_C(t)$ are decomposed into transient and periodic components by using the proposed peak-averaging digital filter Algorithm 1 detailed in Section II-B. Second, the transient signals of step responses are differentiated into impulse responses and compressed for training the transient model. Third, the periodic signals are chopped into chunks with $k_p = 20$. Fourth, the periodic segments are resampled into the same length as the receptive field $w$ of the discriminator and the resampling rate is recorded as $\gamma$ for training the periodic signal model. Fifth, the amplitudes of periodic segments are normalized and then fed into the discriminator network for the training of the periodic signal model.

### B. NN Settings

Multiple 1-dimensional CNN generators/discriminators with various receptive fields are designed for the periodic signal model. The InfoGAN v1 is a lightweight structure with a receptive field $w = 398$. The InfoGAN v2 is a deeper structure with a receptive field $w = 798$. The InfoGAN v3 and later are deep, residual [34] structures with larger receptive fields. The length of the receptive field is chosen based on the specific requirements of precision. In the Buck converter case, the InfoGAN v2 is selected, and its detailed structure is shown in Table II.

For the 1D convolutional GANs, a fully-connected hidden layer is necessary, which is different from the guidelines in 2D DCGAN [35]. We used the Adam optimizer with a learning rate of 0.001 and trained them for 200 epochs.

Table III shows the design of the LSTM layer $f_{\mathrm{ar}}$ in the periodic model and the fully-connected networks in the transient model $f_{\mathrm{DNN}}$.

### C. Prediction Results and Error Analyses

The predictions of periodic and transient signals are shown in Fig. 8 and Fig. 9 respectively. The circuit parameters can be arbitrarily set, while the duty cycle is randomly step changed every 0.03s. The resolution is set to $10^6$/s. The blue curve indicates the predicted result, while the red curve indicates the ground truth plotted with the datasets. By adding the predictions of transient and periodic signals, the prediction of the full signal $u_C(t)$ is depicted in Fig. 10. It can be observed that blue curves fit the red ones under different scenarios even though errors exist. The error is measured by mean absolute error (MAE), relative absolute error (RAE) and mean absolute percentage error (MAPE), which are listed in the figure captions. The MAPE cannot be calculated for periodic signals because the ground truth can be zero.

Apart from the examples shown in Fig. 8, Fig. 9 and Fig. 10, 32 random test samples are evaluated in total. The results are shown in Table IV. Qualitative analyses about the key factors for errors and possible solutions are given as follows.

(a) **The noise induced by signal decomposition.** The proposed decomposition method is based on averaging and interpolation, which may introduce noise into the dataset and contributes to the prediction error, especially when transient signals are changing rapidly. e.g. In the periodic signal dataset, the amplitudes are not static due to the noises resulting from decomposition, the fully-connected NN $f_{\mathrm{amp}}$ would thus suffer from low precision. Such an issue may be addressed by unifying the transient and periodic model to avoid signal decomposition.

TABLE II
THE DETAILED STRUCTURE OF INFOGAN V2

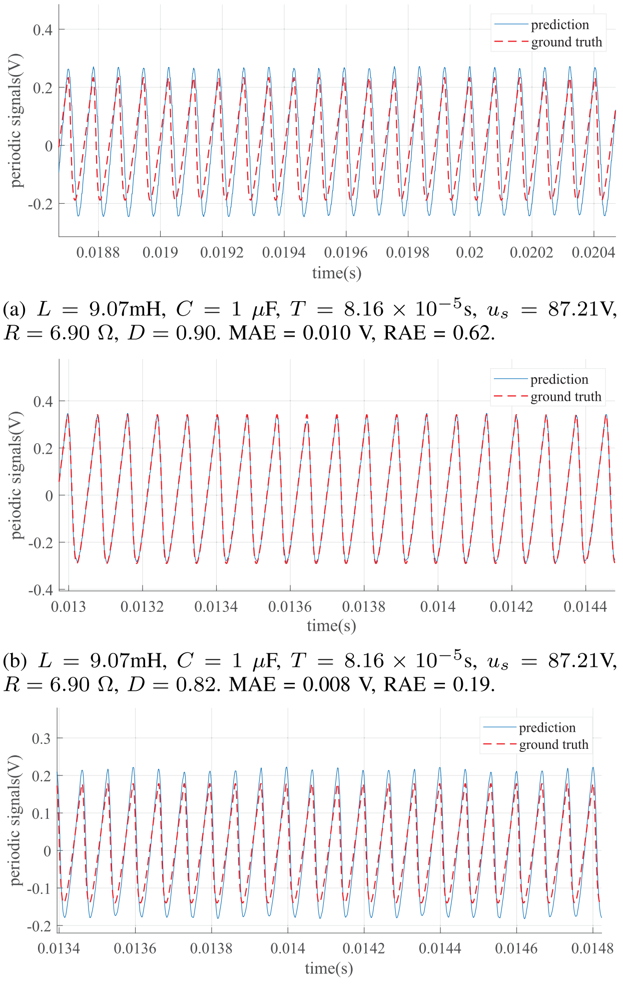
| Discriminator D/ Encoder Q | Generator G |
|---|---|
| Input 1 × 798 × 1 ES | Input $\in \mathbb{R}^{16}$ |
| 4 × 1 conv. 16 lRELU. stride 2 × 1 | FC. 6144 ReLU. batchnorm |
| 4 × 1 conv. 32 lRELU. stride 2 × 1 | reshape 128 × 48 × 1 ES. |
| 4 × 1 conv. 64 lRELU. stride 2 × 1 | 3 × 1 upconv. 96 RELU. stride 1 × 1. batchnorm |
| 4 × 1 conv. 128 lRELU. stride 2 × 1 | 4 × 1 upconv. 64 RELU. stride 2 × 1. batchnorm |
| Flatten. FC. 128 | 4 × 1 upconv. 32 RELU. stride 2 × 1. batchnorm |
| FC. output layer for D, FC. 128-FC.output for Q | 4 × 1 upconv. 16 RELU. stride 2 × 1. batchnorm |
| | 4 × 1 upconv. stride 2 × 1. batchnorm |

TABLE III
THE DETAILED STRUCTURES OF $f_{\mathrm{ar}}$ AND $f_{\mathrm{DNN}}$

| LSTM for the periodic model | Dense NN for the transient model |
|---|---|
| Input 5 × $s$ ES | Input $\in \mathbb{R}^4$ |
| Linear. 256 | FC. 256 lRELU |
| LSTM. 256 tanh | FC. 256 lRELU |
| FC. 8 RELU | FC. 100 lRELU |

TABLE IV
EVALUATION OF ERROR WITH 32 RANDOM TEST SAMPLES

| Model\Error | Average MAE | Average RAE | Average MAPE |
|---|---|---|---|
| Transient model | 1.074V | 0.078 | 2.94% |
| Periodic model | 0.102V | 0.67 | - |
| Overall | 1.178V | 0.086 | 3.48% |



(a) $L = 9.07$mH, $C = 1$ $\mu$F, $T = 8.16 \times 10^{-5}$s, $u_s = 87.21$V, $R = 6.90$ $\Omega$, $D = 0.90$. MAE = 0.010 V, RAE = 0.62.

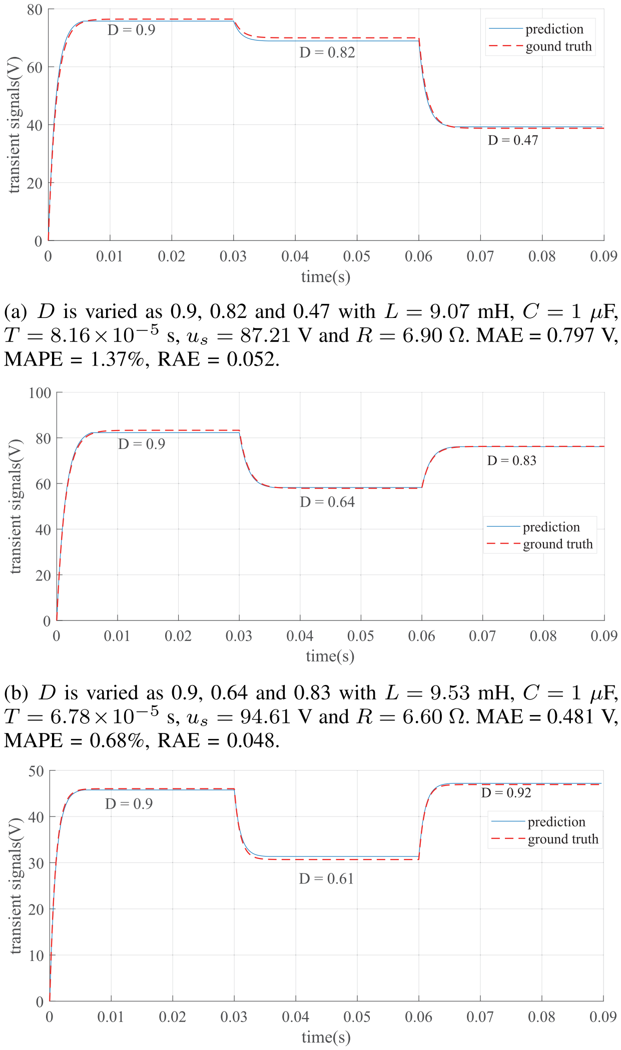
(b) $L = 9.07$mH, $C = 1$ $\mu$F, $T = 8.16 \times 10^{-5}$s, $u_s = 87.21$V, $R = 6.90$ $\Omega$, $D = 0.82$. MAE = 0.008 V, RAE = 0.19.

(c) $L = 9.53$mH, $C = 1$ $\mu$F, $T = 6.78 \times 10^{-5}$s, $u_s = 94.60$V, $R = 6.60$ $\Omega$, $D = 0.90$. MAE = 0.004 V, RAE = 0.54.

Fig. 8. The predictions of the periodic signal model $\widetilde{u}_C(t)$.

(b) **The over-fitting issue and unsmooth latent spaces.** In the proposed periodic signal model, even though InfoGAN does help to alleviate over-fitting issues, the problem still exists, which contributes to the prediction error. Unsupervised training techniques may make latent space unsmooth and thus difficult for the LSTM layer to converge. Such issues may be addressed by shrinking the latent space dimensions and increasing the number of parameters in the LSTM layer. Another solution to this issue is to introduce supervised learning approaches such as conditional GAN.

(c) **The compression of system characteristics.** In the proposed transient signal model, the compression of $\boldsymbol{h}(t)$ would result in a loss of information, and introduce another term of error. However, a more precise compression setting may help to mitigate this issue.



(a) $D$ is varied as 0.9, 0.82 and 0.47 with $L = 9.07$ mH, $C = 1$ $\mu$F, $T = 8.16 \times 10^{-5}$ s, $u_s = 87.21$ V and $R = 6.90$ $\Omega$. MAE = 0.797 V, MAPE = 1.37%, RAE = 0.052.

(b) $D$ is varied as 0.9, 0.64 and 0.83 with $L = 9.53$ mH, $C = 1$ $\mu$F, $T = 6.78 \times 10^{-5}$ s, $u_s = 94.61$ V and $R = 6.60$ $\Omega$. MAE = 0.481 V, MAPE = 0.68%, RAE = 0.048.

(c) $D$ is varied as 0.9, 0.61 and 0.92 with $L = 7.11$ mH, $C = 1$ $\mu$F, $T = 5.88 \times 10^{-5}$ s, $u_s = 52.35$ V and $R = 7.33$ $\Omega$. MAE = 0.392 V, MAPE = 1.10%, RAE = 0.055.

Fig. 9. The predictions of the transient signal model $\overline{u}_C(t)$ with random $D$ variations and circuit parameter settings.

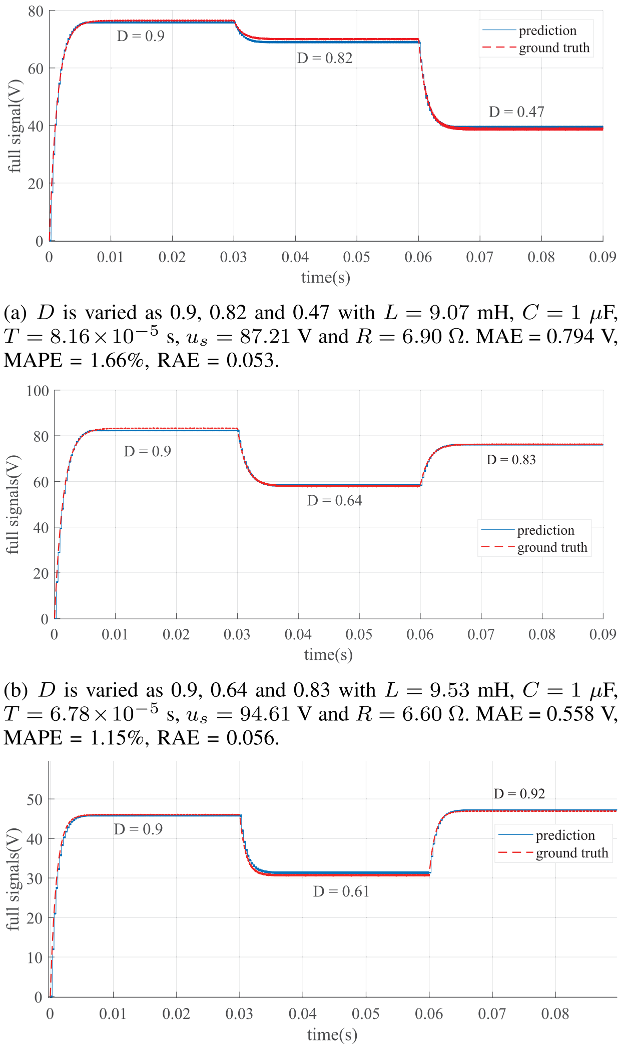
### D. Time Complexity and Speed Analyses

To evaluate the performance of the proposed modeling method, a comprehensive comparison is made with the numerical approach by taking the time complexity and computation time into consideration. A numerical approach based on the widely-used Forward Euler's method is used [10].

Supposed there are $S$ timesteps in the simulation and there are $n_s$ active and passive switches, $n_v$ nodes, $n_b$ branches, and $n$ components having state variables (order of the circuit) in the PE converters, $p_d$ is the probability of state change of the passive switches. For example, $n_v = 4$, $n_b = 6$, $n_s = 2$, $n = 2$, $p_d = f/r \in (0.005, 0.02)$ for a Buck converter. The computation cost of the numerical-based Forward Euler's method can be calculated based on Table V. There are six steps and their corresponding computation cost are highlighted in the table. Such that, the overall cost can be calculated by taking the sum and given by

$$O(S(n + n_s + (1 + p_d)(2n_b + n_v - 1)^3)) \tag{37}$$

On the other hand, the computation cost of the proposed method can be evaluated as follows. Supposed there are $c_p$ channels of parameter inputs, $c_x$ channels of excitation



(a) $D$ is varied as 0.9, 0.82 and 0.47 with $L = 9.07$ mH, $C = 1$ $\mu$F, $T = 8.16 \times 10^{-5}$ s, $u_s = 87.21$ V and $R = 6.90$ $\Omega$. MAE = 0.794 V, MAPE = 1.66%, RAE = 0.053.

(b) $D$ is varied as 0.9, 0.64 and 0.83 with $L = 9.53$ mH, $C = 1$ $\mu$F, $T = 6.78 \times 10^{-5}$ s, $u_s = 94.61$ V and $R = 6.60$ $\Omega$. MAE = 0.558 V, MAPE = 1.15%, RAE = 0.056.

(c) $D$ is varied as 0.9, 0.61 and 0.92 with $L = 7.11$ mH, $C = 1$ $\mu$F, $T = 5.88 \times 10^{-5}$ s, $u_s = 52.35$ V and $R = 7.33$ $\Omega$. MAE = 0.508 V, MAPE = 1.74%, RAE = 0.072.

Fig. 10. The predictions of the overall signal $u_C(t)$.

TABLE V

TIME COMPLEXITY ANALYSIS OF THE FORWARD EULER METHOD

| Step | Operations | Costs |
|---|---|---|
| 1 | Solving the output equations of the state components. | $O(n)$ |
| 2 | Check out the states of the switches. | $O(n_s)$ |
| 3 | Update the incidence matrix $A$ of the circuit. | $O(1)$ |
| 4 | Solve the KCL, KVL and branch equations, using the LU factorization method. | $O((2n_b + n_v - 1)^3)$ |
| 5 | Checkout if any passive switches were changed, if so, repeat the last step | $O(p_d(2n_b + n_v - 1)^3)$ |
| 6 | Update the state equations of components | $O(n)$ |

TABLE VI

TIME COMPLEXITY ANALYSIS OF THE PROPOSED MODEL

| Index | Operations | Costs |
|---|---|---|
| 1 | Fully-connected layer of the transient model. | $O(c_p)$ |
| 2 | Signal convolution of the transient model. | $O(nS)$ |
| 3 | LSTM layer of the periodic model. | $O((c_p + c_s)S/w)$ |
| 4 | Generator layer of the periodic model. | $O(c_s S/w)$ |

TABLE VII

SPEED TEST RESULTS OF THE BUCK CONVERTER MODELS

| Modeling approach | Platform | Cost of the first time(s) | Average cost of 32 times(s) |
|---|---|---|---|
| The Proposed Method | Pytorch-CPU | 1.37 | 0.08 |
| The Proposed Method | Pytorch-CUDA | 3.8 | 0.12 |
| Forward Euler with direct method | Pytorch-CPU | 8.35 | 8.33 |
| Forward Euler with iteration method | Pytorch-CPU | 8.98 | 8.85 |
| ode1 (Backward Euler) | Simulink | 1.98 | 0.78 |

inputs, $c_s$ channels of outputs, and $w$ is the receptive field of transposed CNNs. For the buck converter, the corresponding parameters are $c_p = 3, c_i = 2, c_s = 1, w = 798$. The computation cost of the proposed method can be calculated based on Table VI. Basically, there are four steps and their corresponding computation cost are highlighted in the table. Such that, the overall cost can be calculated by taking the sum and given by

$$O(c_p + nS + (c_p + c_x + c_s)S/w) \tag{38}$$

To sum up, the time complexity of the Forward Euler's method correlates linearly with the number of timesteps, and by order of 3 correlates with the number of nodes and branches. The number and frequencies of passive switches also have an impact. The time complexity of the proposed method only correlates linearly with the number of channels and timesteps, while the circuit topologies are not relevant. Under current settings, if the number of timesteps $S$ is considered, the time complexity of the proposed method is $O(2.007S)$, which is a lot smaller than the Forward Euler's method $O(3448.5S)$.

A comparison of practical execution time is also made between the proposed method and the Forward Euler's methods under different platforms or settings. To make the comparison fair enough, all the methods are run on an identical platform, i.e., PyTorch platform. In addition, the commercially optimized Forward Euler's method in Matlab/Simulink is also considered. Each method is tested 32 times and the average of the results is taken. The simulation time and sampling rate are set as 0.09s and $10^6$/s respectively. The hardware configurations are Intel Core i9-12900K CPU and Nvidia GeForce RTX 3090Ti. The results are summarized in Table VII. The costs of the first time are listed since they include the time for PyTorch and Simulink loading the model and are longer than others. It can be observed that the proposed method is about 10 times faster than the Forward Euler's method in PyTorch platform. Even compared with the commercially optimized Backward Euler's method on the Matlab/Simulink platform, the proposed method is faster.

## V. Conclusion

This work proposes a novel hybrid PiML approach for the non-isolate DC-DC converters, which supports configurable parameter settings, accelerates the speed and can generate refined ripples. The approach decomposes the output signals into transient large signals and periodic small signals. For transient large signals, a fully-connected NN is used to map circuit parameters with system characteristics, such that configurable circuit parameter settings are allowed. For periodic signals, LSTM together with CNN is used to accelerate the simulation by predicting signal features in the compressed latent space, such that refined ripples are generated. A buck converter with configurable parameter settings is modeled by the proposed hybrid PiML method to validate its efficacy. Experiments show that the approach is about 10 times faster than the numerical solvers based on the Forward Euler's method.

## References

[1] A. Wunderlich and E. Santi, "Digital twin models of power electronic converters using dynamic neural networks," in *Proc. IEEE Appl. Power Electron. Conf. Expo. (APEC)*, Phoenix, AZ, USA, Jun. 2021, pp. 2369–2376.

[2] Z. Zhao, D. Tan, B. Shi, Y. Zhu, and H. Jin, "A breakthrough in design verification of megawatt power electronic systems," *IEEE Power Electron. Mag.*, vol. 7, no. 3, pp. 36–43, Sep. 2020.

[3] P. Yang, X. Chen, R. Chen, Y. Peng, S. Wu, and J. Xu, "Stability improvement of pulse power supply with dual-inductance active storage unit using hysteresis current control," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 11, no. 1, pp. 111–120, Mar. 2021.

[4] Z. Li et al., "Fourier neural operator for parametric partial differential equations," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Vienna, Austria, May 2021, pp. 1–16.

[5] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Rev. Phys.*, vol. 3, no. 6, pp. 422–440, May 2021.

[6] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proc. Nat. Acad. Sci. USA*, vol. 113, no. 15, pp. 3932–3937, Apr. 2016.

[7] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, Feb. 2019.

[8] J. Brandstetter, D. Worrall, and M. Welling, "Message passing neural PDE solvers," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Jan. 2022, pp. 1–27.

[9] Y. Zhu, Z. Zhao, B. Shi, and Z. Yu, "Discrete state event-driven framework with a flexible adaptive algorithm for simulation of power electronic systems," *IEEE Trans. Power Electron.*, vol. 34, no. 12, pp. 11692–11705, Dec. 2019.

[10] A. Abid and L. Sbita, "Discrete time model of DC–DC buck converter based on the Euler forward difference," in *Proc. 12th Int. Renew. Energy Congr. (IREC)*, Hammamet, Tunisia, Oct. 2021, pp. 1–4.

[11] S. Yan, Z. Zhou, and V. Dinavahi, "Large-scale nonlinear device-level power electronic circuit simulation on massively parallel graphics processing architectures," *IEEE Trans. Power Electron.*, vol. 33, no. 6, pp. 4660–4678, Jun. 2018.

[12] A. Graves, *Supervised Sequence Labelling With Recurrent Neural Networks*. Berlin, Germany: Springer, Feb. 2012, pp. 37–45.

[13] A. Vaswani et al., "Attention is all you need," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, USA, vol. 30, Dec. 2017, pp. 1–11.

[14] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, Montreal, QC, Canada, vol. 31, Sep. 2018, pp. 1–13.

[15] N. Geneva and N. Zabaras, "Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks," *J. Comput. Phys.*, vol. 403, Feb. 2020, Art. no. 109056.

[16] M. Moradi, S. A. Sadrossadat, and V. Derhami, "Long short-term memory neural networks for modeling nonlinear electronic components," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 11, no. 5, pp. 840–847, May 2021.

[17] H. Sharma and Q.-J. Zhang, "Automated time domain modeling of linear and nonlinear microwave circuits using recurrent neural networks," *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 18, no. 3, pp. 195–208, May 2008.

[18] H. S. Krishnamoorthy and T. Narayanan Aayer, "Machine learning based modeling of power electronic converters," in *Proc. IEEE Energy Convers. Congr. Expo. (ECCE)*, Baltimore, MD, USA, Sep. 2019, pp. 666–672.

[19] P. Qashqai, K. Al-Haddad, and R. Zgheib, "Modeling power electronic converters using a method based on long-short term memory (LSTM) networks," in *Proc. 46th Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Singapore, Oct. 2020, pp. 4697–4702.

[20] H. H. Aghdam and E. J. Heravi, *Guide to Convolutional Neural Networks*. New York, NY, USA: Springer, May 2017.

[21] A. V. D. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, Montreal, QC, Canada, Sep. 2018, pp. 1–13.

[22] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, vol. 33, Dec. 2020, pp. 12449–12460.

[23] M. Bikowski et al., "High fidelity speech synthesis with adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Apr. 2020, pp. 1–15.

[24] I. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Nov. 2020.

[25] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol. (NAACL-HLT)*, New Orleans, LA, USA, Jun. 2018, pp. 1–6.

[26] A. Anandkumar et al., "Neural operator: Graph kernel network for partial differential equations," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Apr. 2020, pp. 1–21.

[27] Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, and P. Perdikaris, "Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data," *J. Comput. Phys.*, vol. 394, pp. 56–81, Oct. 2019.

[28] J.-L. Wu, K. Kashinath, A. Albert, D. Chirila, and H. Xiao, "Enforcing statistical constraints in generative adversarial networks for modeling chaotic dynamical systems," *J. Comput. Phys.*, vol. 406, Apr. 2020, Art. no. 109209.

[29] Z. Li et al., "Multipole graph neural operator for parametric partial differential equations," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, vol. 33, Apr. 2020, pp. 6755–6766.

[30] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, "Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators," *Nature Mach. Intell.*, vol. 3, no. 3, pp. 218–229, Mar. 2021.

[31] E. O. Brigham, *The Fast Fourier Transform and Its Applications*. Boston, MA, USA: Pearson, Apr. 1988.

[32] G. Rilling, P. Flandrin, and P. Goncalves, "On empirical mode decomposition and its algorithms," in *Proc. IEEE Workshop Nonlinear Signal Image Process. (EURASIP)*, vol. 3, Jun. 2003, pp. 8–11.

[33] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, Barcelona, Spain, vol. 29, Dec. 2016, pp. 1–9.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Los Vegas, NV, USA, Jun. 2016, pp. 770–778.

[35] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Caribe Hilton, Puerto Rico, May 2016, pp. 1–16.

**Hanchen Ge** was born in Huangshan, China, in 1999. He received the B.Eng. degree in electrical engineering and automation from North China Electric Power University, Beijing, China, in 2021. He is currently pursuing the M.Eng. degree in intelligent engineering with the Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, Guangzhou, China. His current research interests include artificial intelligence (AI) techniques applied in power electronics.

**Zhihong Huang** (Student Member, IEEE) received the B.Eng. degree in computer science and technology from Sichuan University, Sichuan, China, in 2020. He is currently pursuing the M.Sc. degree in computer science with the Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, Macao, China. His current research interests include machine learning and artificial intelligence.

**Zhicong Huang** (Member, IEEE) received the B.Eng. degree in electrical engineering and automation and the M.Eng. degree in mechanical and electronic engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2010 and 2013, respectively, and the Ph.D. degree in power electronics from The Hong Kong Polytechnic University, Hong Kong, in 2018.

From January 2019 to February 2020, he was a Post-Doctoral Fellow with the UM Macao Talent Program, State Key Laboratory of Analog and Mixed-Signal VLSI, University of Macau, Macao, China. He is currently an Associate Professor with the Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, Guangzhou, China. His research interests include power electronics techniques in electric vehicles and power systems.

Dr. Huang received the Outstanding Reviewer Award from IEEE TRANSACTION ON POWER ELECTRONICS in 2021.