Neural Network based Model Predictive Controllers for Modular Multilevel Converters

Journal:	*IEEE Transactions on Energy Conversion*
Manuscript ID	TEC-00484-2020.R1
Manuscript Type:	ONLY BY INVITATION-Special Section: Model Predictive Control in Energy Conversion Systems
Date Submitted by the Author:	22-Jul-2020
Complete List of Authors:	Wang, Songda; Aalborg University, Department of Energy Technology Dragicevic, Tomislav; Danmarks Tekniske Universitet, Department of Energy Technology Gao, Yuan; University of Nottingham Teodorescu, Remus; Aalborg University, Department of Energy Technology
Technical Topic Area:	Solar/photovoltaic < Energy Development and Power Generation, Wind < Energy Development and Power Generation
Key Words:	Power electronics, model predictive control, Machine Learning, Modular Multilevel Converter

SCHOLARONE™
Manuscripts

Neural Network based Model Predictive Controllers for Modular Multilevel Converters

Songda Wang, *Student Member*, Tomislav Dragicevic, *Senior Member, IEEE,* Yuan Gao, *Student Member, IEEE,* and Remus Teodorescu, *Fellow, IEEE*

***Abstract*--Modular multilevel converter (MMC) has attracted much attention for years due to its good performance in harmonics reduction and efficiency improvement. Model predictive control (MPC) based controllers are widely adopted for MMC because the control design is straightforward and different control objectives can be simply implemented in a cost function. However, the computational burden of MPC imposes limitations in the control implementation of MMC because of many possible switching states. To solve this, we design machine learning (ML) based controllers for MMC based on the data collection from the MPC algorithm. The ML models are trained to emulate the MPC controllers which can effectively reduce the computation burden of real-time control since the trained models are built with simple math functions that are not correlated with the complexity of the MPC algorithm. The ML method applied in this study is a neural network (NN) and there are two types of establishing ML controllers: NN regression and NN pattern recognition. Both are trained using the sampled data and tested in a real-time MMC system. A comparison of experimental results shows that NN regression has a much better control performance and lower computation burden than the NN pattern recognition.**

***Index Terms*--Modular multilevel converter (MMC), Model predictive control (MPC), control design, neural network (NN), pattern recognition**

I. Introduction

MODULAR multilevel converter (MMC) is one of the most attractive topologies for high voltage direct current (HVDC) applications due to its merits: low harmonics, high efficiency, and good fault tolerance ability [1-4]. However, the complicated structure of the MMC makes it challenging to control effectively.

In order to control the MMC to achieve its merits, a lot of different control methods have been proposed, from the conventional proportional-integral controller (PI) [5] / proportional-resonant (PR) control [6] to non-linear sliding mode control [7], and to model predictive control (MPC) [8], [9]. Out of different control methods, the model predictive control (MPC) is widely accepted by power electronics engineers because of its merits: simplicity, fast dynamic response, easy inclusion of nonlinearities, and others [10]. Many earlier papers have studied the MPC based MMC controllers [11], [12]. MPC is a discrete model-based control method for multi-input-multi-output (MIMO) system based on the cost functions, established based on the predicted behavior of the system in accordance with its discrete dynamic model. For MMC in particular, by minimizing the cost function in every sampling step, the output current and circulating current can be controlled and the submodule voltages can also be balanced at the same time [9]. However, the main drawback of the MPC MMC is the heavy computational burden, which becomes particularly limiting when the number of submodules is high [8]. Namely, the MPC algorithm should manage to evaluate all the possible capacitor voltage combinations in one sampling period. However, as the number of modules rises, the amount of calculations rises rapidly in three-phase optimal switching state method in [13], and also in three-phase optimal switching vector method in [14]. In order to reduce this burden, researchers have proposed many modifications to the conventional MPC algorithms [15-17]. However, all these methods did not change the core feature of the MPC: the MPC algorithm exhaustively evaluates the switching signals online to find the one that minimizes some predefined cost function. This online optimization method leads to a high computational burden when the number of possible switching signals is high. And such computational burden reduction methods are very complex to implement and require very high-level math expertise [18].

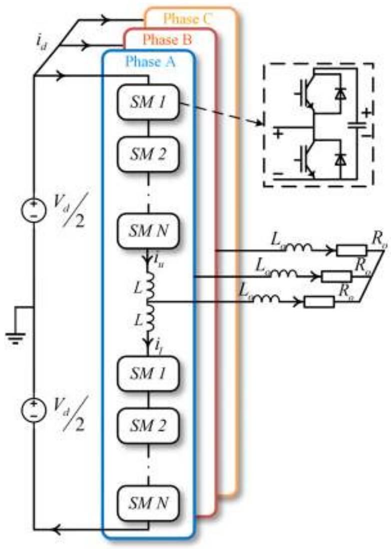


Fig. 1. MMC diagram.

Applying machine learning (ML) to emulate the MPC can help to reduce the computational burden. In [19], a supervised

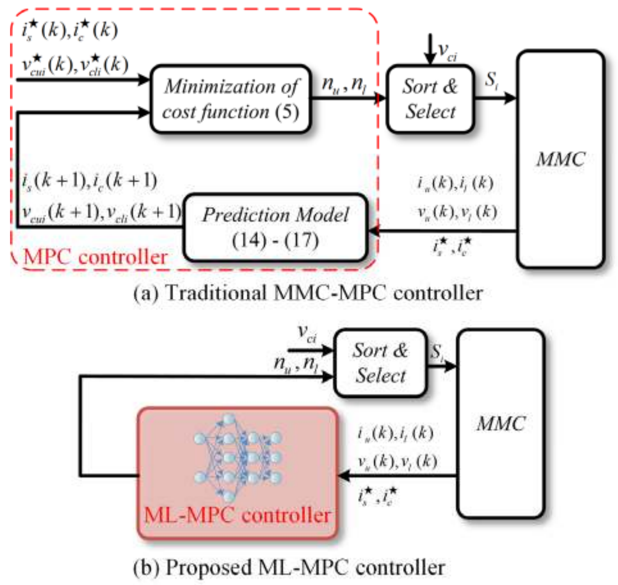

$i_s^\star(k), i_c^\star(k)$
$v_{cui}^\star(k), v_{cli}^\star(k)$
Minimization of cost function (5)
n_u, n_l
v_{ci}
Sort & Select
S_i
MMC
$i_s(k+1), i_c(k+1)$
$v_{cui}(k+1), v_{cli}(k+1)$
Prediction Model (14) - (17)
$i_u(k), i_l(k)$
$v_u(k), v_l(k)$
$i_s^\star, i_c^\star$
MPC controller
(a) Traditional MMC-MPC controller
ML-MPC controller
(b) Proposed ML-MPC controller

Fig. 2. The comparison of ML MPC and traditional MPC.

learning framework is proposed to emulate an MPC with reduced computational complexity. This article introduces the methodology of using machine learning models to simulate MPC. Paper [20] also introduces the same idea. Paper [21] extends this method to control the three-phase inverter with an output LC filter. By this method, a lower THD and a better steady and dynamic performance are achieved. In [22], a lower computational burden is achieved by machine learning based controller in a two-level converter, the performance is the same as the performance with model predictive control. However, such techniques have thus far mainly been proposed for robotic systems and very simple power converters. To best of our knowledge, an MPC-based machine learning imitator has not yet been developed nor experimentally tested for the MMC.

ML is a cutting-edge technology that has been widely used for establishing non-parametric models of various complicated power electronic processes using solely the process data. Several papers have used neural network (NN) to solve power electronics problems, e.g. identifying power system's active power fluctuations in real-time [23]; automatically designing the power electronics system for reliability [24]; online weighting factor adjustment for predictive torque control of induction machines fed by 3L-NPC converters [25]. ML is a technology which can learn from the sample data, also known as "training data", which could be from the real world or a software. ML can change its memory based on different settings. All machine learning technologies build desired math models based on the training data in order to make predictions or decisions without being explicitly programmed. Therefore, the training and prediction performance of ML is highly related to the quality of the data. For now, the data that ML can process is getting larger because the computing power of modern computers is getting stronger with the development of computing technology [26].

There are different algorithms in machine learning, such as regression, decision tree, NN, and support vector machine. In this paper, we only use neural network but focus on two different types, regression [22] and pattern recognition [27]. This category is due to their output types: outputs of NN regression are usually continuous but NN pattern recognition only has two or more features as the output. Initially, NNs were proposed to simulate the structure of human brain. NN could have one or several hidden layers, where each layer has several neurons. Every neuron is a node that determines the input-output relationship of the signal. NN can be trained to a nonlinear model using a proper data set. Such a general nonlinear model can approximate any given input-output function with arbitrary precision [24], [28].

In this paper, two ML-based emulations of the MPC algorithm are designed to control the MMC with much lower computational burden compared to the original MPC and, one emulation (NN regression) achieves excellent control performance. The ML models are trained offline and such trained models can either be used offline or implemented in a digital microprocessor for online operation. In fact, we show in this paper that the computational burden of the NN regression model, that perfectly emulates the MPC controller, is much lower than the MPC itself.

II. System Model

In this section, the structure and working principle of MMC will be introduced. The topology of half bridge MMC will be explained briefly and the large signal model of MMC will be derived.

A. MMC Introduction

Fig. 1 shows the topology of a half-bridge submodule MMC. Normally, an MMC consists of three phases, where each phase has two arms: upper and lower arm. Each arm is comprised of N series-connected half bridge submodules, and an arm inductance L_{arm}[29]. The submodule capacitor voltage is kept close to the rated dc voltage by the MMC controller. In this way, the single submodule can be controlled as a voltage source by inserting or bypassing the submodule. The MMC output AC voltage can be controlled by changing the number of inserted submodules [30].

B. Dynamics of MMC

The direction of the upper arm current and lower arm current is shown in Fig. 1. By applying the Kirchhoff's voltage law to the MMC circuit, the MMC dynamic equations can be derived as follows:

$$\frac{V_d(t)}{2} - v_{cuk}(t) - L_{arm}\frac{di_{uk}(t)}{dt} = R_s i_{sk}(t) + L_s\frac{di_{sk}(t)}{dt} \tag{1}$$

$$\frac{V_d(t)}{2} - v_{clk}(t) - L_{arm}\frac{di_{lk}(t)}{dt} = -R_s i_{sk}(t) - L_s\frac{di_{sk}(t)}{dt} \tag{2}$$

where V_d is DC line voltage, v_{cuk} and v_{clk} are the upper and lower arm voltage in phase k respectively, i_{uk} and i_{lk} are the upper and lower arm current in phase k respectively. k means the phase number, *k=0, 1, 2* (0 for phase a, 1 and 2 for b and c respectively); R_s and L_s are the output resistance and inductance respectively; i_{sk} is output current.

We define the output current i_{sk} and circulating current i_{ck}:

$$i_{sk}(t)=i_{lk}(t)-i_{uk}(t) \tag{3}$$

$$i_{ck}(t)=\frac{1}{2}\left[i_{uk}(t)+i_{uk}(t)\right]-\frac{1}{3}i_d(t) \tag{4}$$

The dynamic equations of the output ac current and circulating current can be derived from (1)-(4):

$$\frac{di_{sk}(t)}{dt}=\frac{1}{L+2L_0}[v_{clk}(t)-v_{cuk}(t)-2R_0 i_{sk}(t)] \tag{5}$$

$$\frac{di_{ck}(t)}{dt}=\frac{1}{2L}[v_d(t)-v_{clk}(t)-v_{cuk}(t)] \tag{6}$$

When the submodule is in on state, the dynamic of submodule capacitor voltage can be expressed by the relation of capacitance and the arm current:

$$\frac{du_{cuki}(t)}{dt}=\frac{i_{uki}}{C_{SM}} \qquad \frac{du_{clki}(t)}{dt}=\frac{i_{lk}}{C_{SM}} \tag{7}$$

where u_{cuki} and u_{clki} are the *i*th upper and lower submodule capacitor voltages respectively (*i=1...N*); C_{SM} is submodule capacitance.

III. Model Predictive Control of MMCs

In this section, the working principle of MPC for MMCs is introduced by using the dynamic equations in Section II. This MPC model will be used to collect the input/output sample data for training the NN-based controllers in Section IV.

A. MMC Control Scheme

Taking one-phase MMC as an example, the conventional MPC for MMCs can be introduced step by step as follows, where the detailed information of this MPC method is described in [17]. The control scheme of MPC MMC is shown in Fig.2 (a), and is executed in the following sequence:

1) Measurement of MPC input variables;
2) Prediction of the output current and circulating current for the next sampling period for every possible inserted submodule number;
3) Creation of the cost function including the information of circulating current and output current;
4) Selection of the best upper/lower arm inserted submodule number that minimizes the cost function;
5) Application of the optimized upper/lower arm inserted submodule number.

B. MMC Model Predictive Control Model

Based on the Euler forward equation in (8), the dynamic equations of MMCs can be transferred to the discrete mathematical model [17] :

$$\frac{dx(t)}{dt}\approx\frac{x(k+1)-x(k)}{T_s} \tag{8}$$

where $x(k+1)$ and $x(k)$ are the variable of at time instant *k+1* and *k* respectively; T_s is the sampling interval.

The dynamic equations (5)-(8) can be transferred to a discrete model by (8)

$$\begin{cases} i_s(k+1)=\mathbf{A}[(n_l(k)\cdot v_{cl}(k+1)-n_u(k)\cdot v_{cu}(k+1))/N]+\mathbf{B}i_s(k) \\ \mathbf{A}=2T_s/(L_{arm}+2L_s),\mathbf{B}=1-2T_sR_s/(L_{arm}+2L_s) \end{cases} \tag{9}$$

$$\begin{cases} i_c(k+1)=\mathbf{C}[V_d-(n_l(k)\cdot v_{cl}(k+1)+n_u(k)\cdot v_{cu}(k+1))/N]+i_c(k) \\ \mathbf{C}=T_s/(2L_s) \end{cases} \tag{10}$$

$$v_{cu}(k+1)=\frac{n_u(k)\cdot T_s}{C_{SM}}i_u(k)+v_{cu}(k) \tag{11}$$

$$v_{cl}(k+1)=\frac{n_l(k)\cdot T_s}{C_{SM}}i_l(k)+v_{cl}(k) \tag{12}$$

In order to reduce the computation burden of MPC, the sorting and selecting of submodule capacitor voltages are achieved by independent sorting and balancing block, which is outside of the MPC algorithm. Fig. 2 shows this block. The sorting and balancing method of MMC capacitor voltages is introduced in [2] and [30].

The cost function used in this case is:

$$g=w_1\cdot|i_s^{\star}(k)-i_s(k+1)|+w_2\cdot|i_c^{\star}(k)-i_c(k+1)| \tag{13}$$

where w_1, w_2 are weighing factors for i_s and i_c respectively. $i_s^{\star}$, $i_c^{\star}$ are the references of output current and circulating current, respectively. In this paper, $w_1=w_2=1$.

C. Delay Compensation

In order to compensate the delay in the experimental setup, the delay compensation method described in [31] is applied. The key idea is to estimate the controlled variables (circulating current and output current) in time instant *k+1* by considering the applied control signals and then predict the variables in time instant *k+2* for all the impossible control signals. In this way, the cost function is minimized by applying the best switching signals in time *k+2*. The new discrete mathematical equations are introduced in (14)-(17):

$$\begin{cases} i_s(k+2)=\mathbf{A}[(n_l(k+1)\cdot v_{cl}(k+2)-n_u(k+1)\cdot v_{cu}(k+2))/N] \\ \qquad +\mathbf{B}i_s(k+1) \\ \mathbf{A}=2T_s/(L_{arm}+2L_s),\mathbf{B}=1-2T_sR_s/(L_{arm}+2L_s) \end{cases} \tag{14}$$

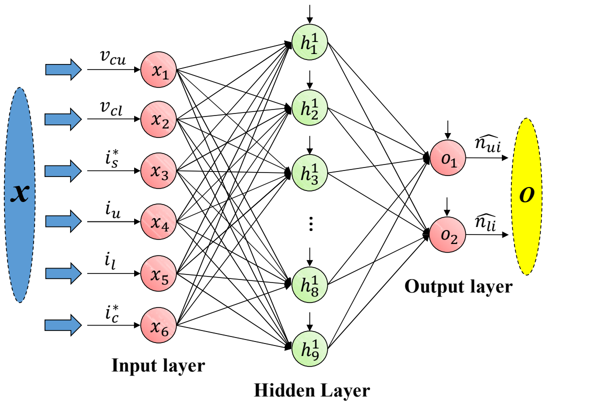
$$\begin{cases} i_c(k+2)=\mathbf{C}[V_d-(n_l(k+1)\cdot v_{cl}(k+2)+n_u(k+1)\cdot v_{cu}(k+2))/N] \\ \qquad +i_c(k+1) \\ \mathbf{C}=T_s/(2L_s) \end{cases} \tag{15}$$

$$v_{cu}(k+2)=\frac{n_u(k+1)\cdot T_s}{C_{SM}}i_u(k+1)+v_{cu}(k+1) \tag{16}$$

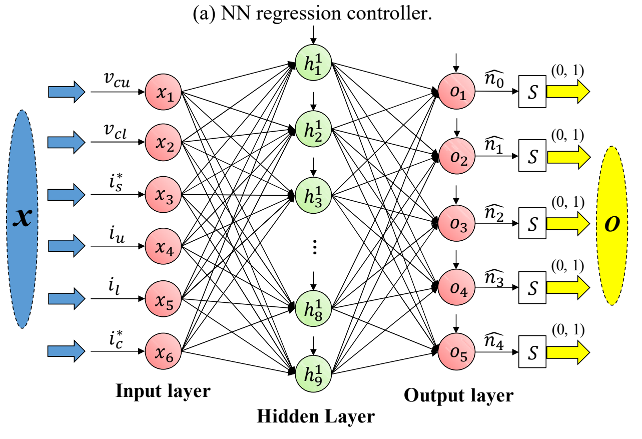
$$v_{cl}(k+2)=\frac{n_l(k+1)\cdot T_s}{C_{SM}}i_l(k+1)+v_{cl}(k+1) \tag{17}$$

D. Deterministic input-output relationship of MPC

Let us consider one digital sampling interval (T_s) of MPC as an example. During this interval, set of measured input variables are transferred to the MPC algorithm that accordingly



(a) NN regression controller.



(b) NN pattern recognition controller

Fig. 3. Deployment of the proposed two NN-based controllers. Both have 3 layers with 9 neurons in the hidden layer. Weights and biases are omitted for simplicity.

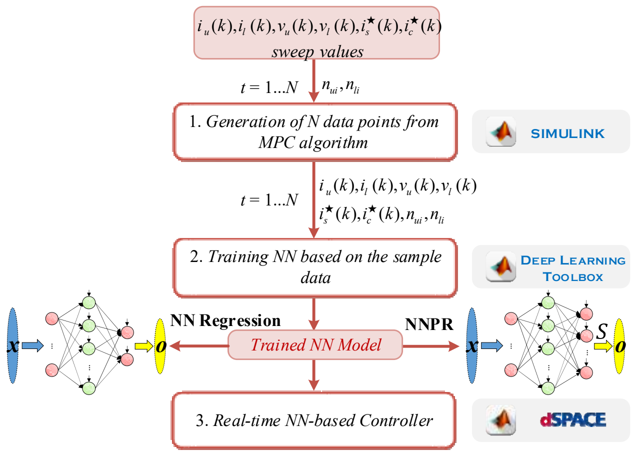
predicts the output currents and circulating currents for all possible switching signals and applies the one that minimizes the cost function. An important observation is that this process is completely deterministic, i.e. for the same set of input variables (i.e. measurements) and a given cost function, the outputs (inserted submodules) will always be the same. In this context, while the conventional MPC uses exhaustive search in every time instant to identify the optimal actuation, this is not necessary. On the contrary, it should be possible to represent the deterministic input-output relationship with more computationally efficient structure. Then, we could achieve the same control effect as the MPC, but with a lower online computational burden. This is indeed a key idea of this paper.

In the following parts, we will introduce two offline NN models to represent the deterministic relationship between inputs and outputs in the MPC algorithm.

IV. Neural Network Based Controllers for MMCs

A. Introduction of the Neural Network Method

From Fig. 2, we can see the difference between the traditional MPC MMC and the NN controlled MMC. That is, the NN-based controller replaces the MPC controller block. The NN inputs include six elements: upper arm voltage v_{cu}, lower arm voltage v_{cl}, output current reference $i_s^\star$, upper arm current


$i_u(k), i_l(k), v_u(k), v_l(k), i_s^\star(k), i_c^\star(k)$
sweep values
$t = 1...N$
n_{ui}, n_{li}
1. Generation of N data points from MPC algorithm
SIMULINK
$t = 1...N$
$i_u(k), i_l(k), v_u(k), v_l(k)$
$i_s^\star(k), i_c^\star(k), n_{ui}, n_{li}$
2. Training NN based on the sample data
Deep Learning Toolbox
NN Regression
NNPR
Trained NN Model
x
o
S
3. Real-time NN-based Controller
dSPACE
Fig. 4. Implement procedure of the proposed NN-based controller.

i_u, lower arm current i_l and circulating current reference $i_c^\star$. The upper arm insert number n_{ui} and lower arm insert number n_{li} are two outputs for the proposed NN controllers.

As mentioned, this paper represents two different neural networks: NN regression and Neural Network Pattern Recognition (NNPR). Though the training data collected from MPC algorithm (Section III) are the same for both networks, their data processing varies due to the different requires of NN outputs. NN regression has no limits for the output elements but NNPR requires the elements must be integer even can only be 0 or 1 in some applications. The following two subsections will introduce the two NN controllers in detail.

B. Proposed two Model Predictive Controllers for MMCs

For the NN regression controller, it is designed as a 3-layer NN whose inputs/outputs are directly using the same design with the training data (shown in Fig. 3(a)). Thus, this controller represents the following relation:

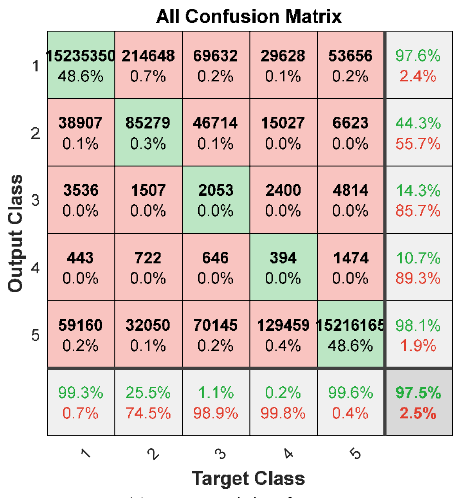
$$(n_{ui}, n_{li}) = F(v_{cu}, v_{cl}, i_s^\star, i_u, i_l, i_c^\star) \tag{18}$$

It is noted that the two outputs should be integer and their values could be 0, 1, 2, 3 or 4 in this study, thus the outputs of this controller should be rounded up and limited to these 5 features.

In contrast, NNPR controller is a novel design. As shown in Fig. 3(b), there are 5 elements in the NN output which corresponds to the 5 features instead of n_{ui}, n_{li} in Fig. 3(a). The purpose of this design is to comprehensively explore the value space ([0, 4]) of n_{ui}, n_{li} and to give an accurate prediction for MMC control. Based on that, according to the value of n_{ui}, n_{li}, each element in output can be set 0 or 1 for NNPR training. Therefore, the NNPR controller represents this relation:

$$(n_0, n_1, n_2, n_3, n_4) = F(v_{cu}, v_{cl}, i_s^*, i_u, i_l, i_c^*) \tag{19}$$

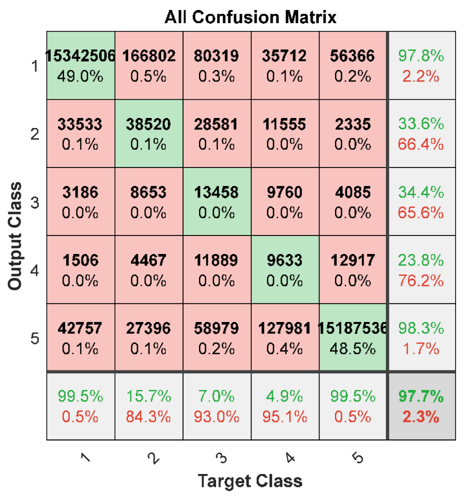
where n_i $(i = 0,1,2,3,4)$ denotes the NNPR value that could quantify the probability of "insert number equals i" because NNPR training can limit all predictions into (0, 1) by the Softmax function [22]. Softmax function, also known as normalized exponential function, takes as input a vector (the output vector of NNPR in this case) and normalizes it into a (0, 1) probability distribution of each component in this vector. It



All Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	
1	15235350 48.6%	214648 0.7%	69632 0.2%	29628 0.1%	53656 0.2%	97.6% 2.4%
2	38907 0.1%	85279 0.3%	46714 0.1%	15027 0.0%	6623 0.0%	44.3% 55.7%
3	3536 0.0%	1507 0.0%	2053 0.0%	2400 0.0%	4814 0.0%	14.3% 85.7%
4	443 0.0%	722 0.0%	646 0.0%	394 0.0%	1474 0.0%	10.7% 89.3%
5	59160 0.2%	32050 0.1%	70145 0.2%	129459 0.4%	15216165 48.6%	98.1% 1.9%
	99.3% 0.7%	25.5% 74.5%	1.1% 98.9%	0.2% 99.8%	99.6% 0.4%	97.5% 2.5%

(a) NNPR training for n_{ui}



All Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	
1	15342506 49.0%	166802 0.5%	80319 0.3%	35712 0.1%	56366 0.2%	97.8% 2.2%
2	33533 0.1%	38520 0.1%	28581 0.1%	11555 0.0%	2335 0.0%	33.6% 66.4%
3	3186 0.0%	8653 0.0%	13458 0.0%	9760 0.0%	4085 0.0%	34.4% 65.6%
4	1506 0.0%	4467 0.0%	11889 0.0%	9633 0.0%	12917 0.0%	23.8% 76.2%
5	42757 0.1%	27396 0.1%	58979 0.2%	127981 0.4%	15187536 48.5%	98.3% 1.7%
	99.5% 0.5%	15.7% 84.3%	7.0% 93.0%	4.9% 95.1%	99.5% 0.5%	97.7% 2.3%

(b) NNPR training for n_{li}

Fig. 5. Confusion matrix of NNPR training.

is used in the final layer of the NNPR controller as a classifier.

As the NNPR generates the probabilities for 5 features, this controller for the studied MMC should use 2 networks, one for n_{ui} and the other for n_{li}.Besides, after getting the output vector $(n_0, n_1, n_2, n_3, n_4)$ in each network, 5 feature values should be compared and the largest one n_{j} determines that the final resulting insert number is j.

Then the neuron network structure should be selected. Firstly, we select one-hidden-layer structure thus we only need

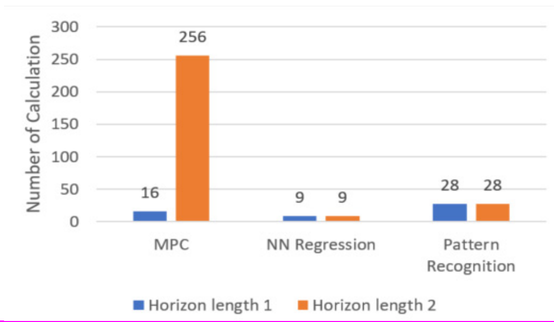


Fig. 6. Comparison of the number of calculations and the horizon length.

to select the neuron network number in the hidden layer. Rule of a thumb is used to determine the maximum and the minimum number of neurons [32], and then select a relatively high number of neurons to achieve better fitting performance within the recommended neuron number range:

Minimum neuron number:

$$0.5(N_{in} + N_{out}) = 4 \quad (N_{in} = 6, N_{out} = 2) \qquad (6)$$

Maximum neuron number:

$$2N_{in} = 12 \qquad (7)$$

where N_{in} is the input unit number $N_{in} = 6$, N_{out} is the output unit number $N_{out} = 2$.

In this paper, we selected 9 neurons in the hidden layer.

C. *Collection and Training Steps*

To clarify the training procedure, all the steps are shown in Fig. 4. Each step is further elaborated as follows.

1) Generation of N data samples from the MPC algorithm: The data can be extracted solely from the MPC algorithm block (only the MPC controller in MMC is used to generate the training data). The sweep values of input data are v_{cu}, v_{cl} : [0:10:350], which means the range of upper and lower arm voltages is from 0 to 350V with the gap of 10V, therefore each arm voltage has 36 data points; $i_s^{\star}$: [-6:1:6], 13 data points; i_u, i_l : [-6:1:6], 13 data points each; $i_c^{\star}$:[0:0.2:2], 11 data points. Therefore, the training data includes around 31.32 million samples but their collection time is only 76 *secs* based on the MPC algorithm.
2) NN Model Training: In order to train NN and set stop conditions, the samples were randomly divided into three data sets, i.e. the training set (70 % of data), the validation set (15 % of data), and the testing set (15 % of data). The extracted data was then used to train the desired NN controllers, which represent the afore-mentioned relationships between input variables and output variables. A workstation PC with Dual Intel Xeon Silver 4110 CPU is used to train the NN controllers, i.e. a regression NN and two pattern NNs. All networks have the same structure in the hidden layer and are trained for 800 iterations. Their training tool is the MATLAB Deeping Learning Toolbox.
3) After getting the trained NNs, they can be used to calculate n_{ui}, n_{li} of MMC in a real-time simulation or experiment. As discussed in the last subsection, the outputs of the two networks should be typically processed to give the final accurate n_{ui}, n_{li}, with which the output current and the circulating current can be controlled to track their references in MMC.

D. *Training performance of two NN controllers*

To clearly show the training performance of the proposed NN controller, Fig. 5 gives two confusion matrixes obtained from the NNPR training in MATLAB (one for n_{ui} , the other for n_{li}). As discussed, there are 5 features/classes in each NNPR controller and for every feature, it is a 1-0 classification problem. Each row in Fig. 5 corresponds to an output class (i.e. n_0, n_1, n_2, n_3, n_4), and the columns are the target classes which

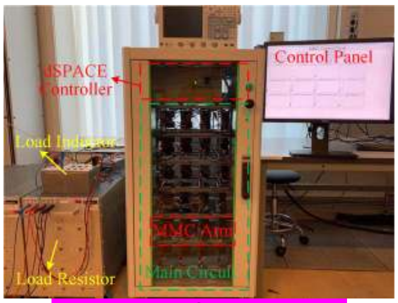


Fig. 7. The MMC setup Diagram

TABLE I MMC PARAMETERS IN EXPERIMENT

	Experiment
Number of SMs per arm (N)	4
Rated DC voltage (v_d)	200 V
Nominal SM capacitance (C_{SM})	2000 μF
Nominal SM capacitor voltage (v_c)	50 V
Rated frequency (f)	50 Hz
Arm inductance (L_{arm})	10 mH
Sample frequency	10 kHz
Load inductance (L_s)	1.8mH
Load resistance (R_s)	10.8Ω

are from the sample data. The green cells in the diagonal of the matrix show the number and percentage of correctly classified data points (at the final training iteration), while all other red cells show the incorrect classifications. Therefore, nearly 98% of the predictions are matched with sample data in the NNPR training. On the other hand, five light-grey blocks in the last column and the other five light-grey blocks in the last row show the specific prediction accuracies for every class/feature. As shown in Fig. 5, the 3rd and 4th classes (n_2, n_3) have very low accuracies for both NNPR nets. Then, we may adjust the training data set of n_3 to pursue a better training performance. However, in this study, we did not further change the data of n_2 , n_3 since all training data is obtained under certain conditions with sinusoidal references thus, it is very hard to manually determine which inputs can get n_2 , n_3 data. In addition, amounts of n_2, n_3 data are relatively small (0.56% for n_{ui}, 0.62% for n_{li})). More importantly, the NNPR nets do not demonstrate good control performance in the experiment (see Section V) even though they both have excellent training performance. In contrast, the NN regression can reflect the MPC characteristic of n_{ui} and n_{li} properly because it trained n_{ui} and n_{li} in the same NN rather than two independent nets. Therefore, the confusion matrix can provide significant information for NN controller training analysis.

Regarding the trained NN regression controller, its performance needs to be additionally calculated based on the sample data because there are no classification results during the training process. Two outputs (Fig. 3(a)) should be rounded up into [0, 4] and then compared with the target classes of the sample. Then, we can also obtain the confusion matrixes for n_{ui} and n_{li}, their general accuracies are both around 93.2%, smaller than NNPR. However, the prediction accuracy of n_2 using NN regression is 33.7% for n_{ui}, 34.9% for n_{li}, much higher than that of NNPR (1.1% for n_{ui}, 7.0% for n_{li}, see Fig. 5). The same phenomenon happens to n_1 and n_3: for example, the prediction accuracy of n_3 using regression is 27.3% for n_{ui}, 28.8% for n_{li} while, as shown in Fig. 5, we only got 0.2% for n_{ui}, 4.9% for n_{li} by using NNPR. Therefore, regarding the 2nd, 3rd and 4th classes, the prediction performance of NN regression is much better than NNPR. That is the main reason that NNPR performs very bad in the experimental regulation, see Section V.

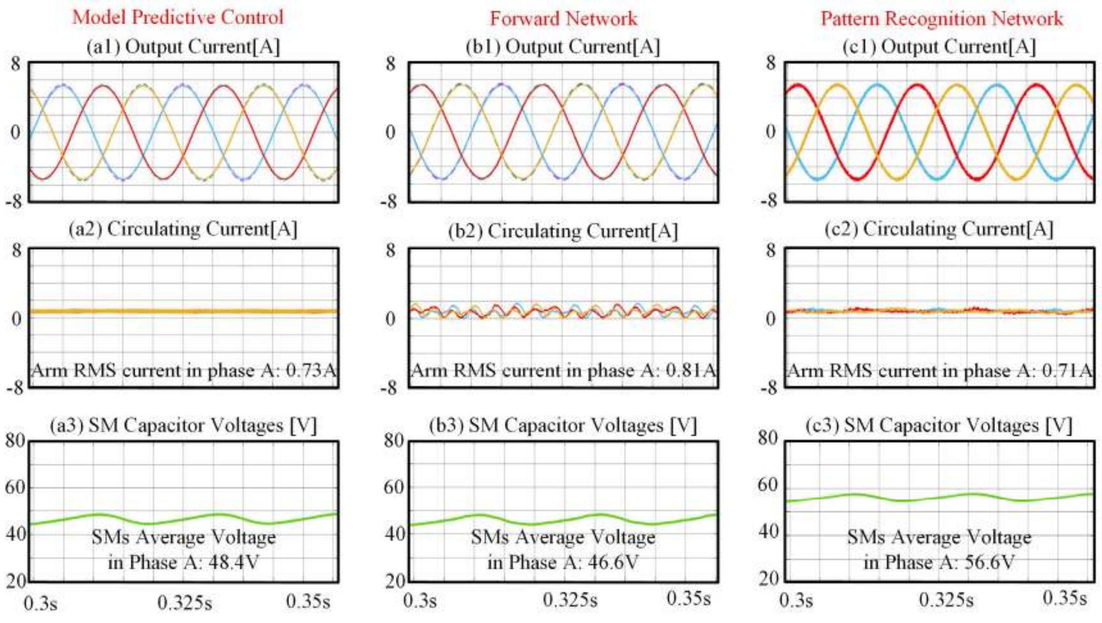


Fig. 8. Steady state performance of three controllers: (a) MPC controller, (b) NN regression controller, (c) NN pattern recognition controller.

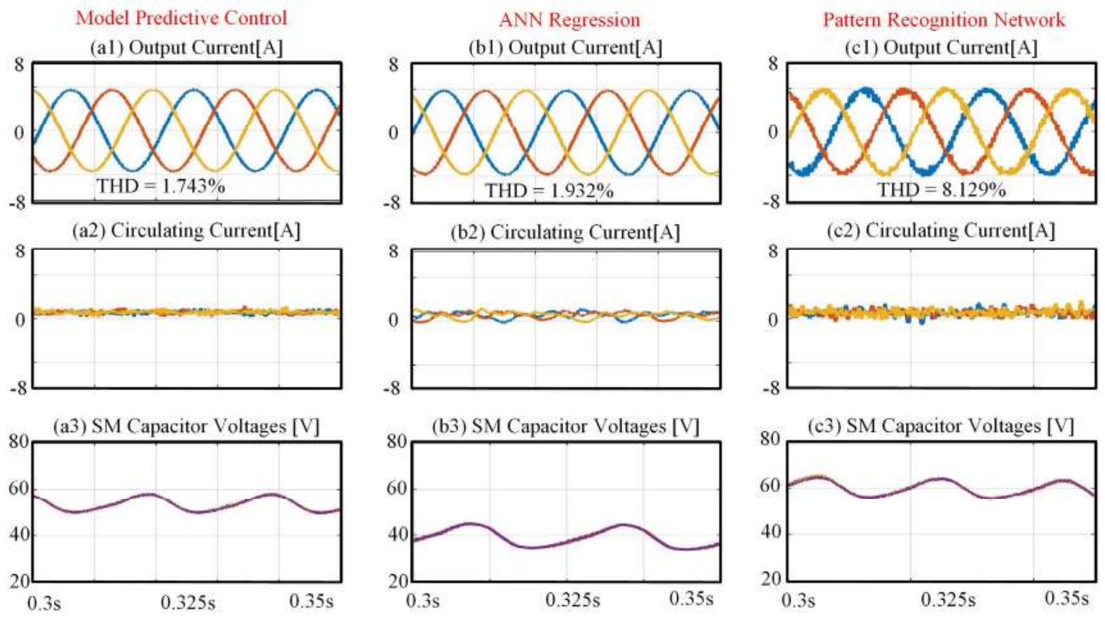


Fig. 9 Experimental Results of the 1). MPC, 2). NN Regression Network Method, and 3). Pattern Recognition Network

In Fig. 6, the number of calculations and the horizon length results are shown, the number of calculations in conventional MPC control will significantly increase with horizon length. When the length is 1, the number of calculations is 16, however, the number of calculations increases to 256 when the length is 2. The NN based controllers have a small number of calculations no matter what the length is. Due to the different structure of different NN controllers, the number of calculations is different. The NN regression is 9, and NNPR is 14*2.

V. Verification Results

Two proposed NN controllers are verified in a real-time simulation model. The simulations and experiments are carried out in a three-phase MMC, with 4 half-bridge SMs per arm. The proposed controller is implemented in DS1006 from dSPACE. The experimental setup is shown in Fig. 7. The parameters of the simulation model and experimental setup are shown in TABLE I. The MMC block diagram is shown in Fig. 1 The controller structure is shown in Fig. 2(b). The input variables of the ML controller are: $v_{cu}, v_{cl}, i_u, i_l, i_s^\star, i_c^\star$. The output variables are the upper/lower arm inserted numbers of MMC. These inserted numbers are sent to the sort & select block for balancing the capacitor voltages. The detailed information about sort & select block is introduced in [2]. Setup Steady State Performance

Fig. 8 shows the simulation steady performance of the MPC controller, NN regression controller, and NN pattern recognition controller. From Fig. 8 (a1) to (a3), the three-phase output currents are controller to track their references: AC currents with 5.5A amplitudes. Regarding circulating current, the MPC and NNPR controller both have good circulating current reduction effect, but the NN regression controller has the worst results which are shown in Fig. 8 (b2). The RMS of circulating currents of MPC and NNPR are 0.71A and 0.73A respectively, the RMS of NN regression is 0.81A. Finally, the capacitor voltages are well balanced because of the common sort & select block, but the average voltage in pattern recognition controller is higher.

Fig. 9 shows the experimental results of the proposed methods. This figure clearly shows that even though NNPR can achieve good control performance in simulation, NNPR nets do not work well in the experiment (THD becomes larger than 8%). The main reason is that, as discussed in the last Section, the training performance of the 2nd class (n_1), 3rd class (n_2) and 4th class (n_3) are very poor (their loss rates are up to 84.3%, 98.9% and 99.8% respectively). The simulation results using NNPR are still acceptable because the simulated circuit and the environment in Simulink are relatively ideal without noise. But in experimental operation, the low-accuracy predictions of n_1, n_2 and n_3, practical noise and uncertainty in-circuit all affect the practical current control performance.

It is noted that features of training data and the decoupled training of n_{ui} and n_{li} determine the NNPR training performance (Fig. 5): on the one hand, the data amounts of n_2 and n_3 are both very small (smaller than 0.7%) thereby compromising their training accuracy to pursue higher holistic NN training accuracy; on the other hand, two NNs are trained separately thus only the error of one insert number (n_{ui} or n_{li}) is set as the training goal without considering the other, even though the NNPR training can achieve high prediction accuracy, the trained NNPR nets cannot reflect the MPC characteristic of n_{ui} and n_{li} properly. In order to further verify this conclusion, we trained the decoupled two nets using NN regression for n_{ui} and n_{li} and found that, even though this two-net-regression method can achieve good general training accuracy (95.61% for n_{ui}, 95.12% for n_{li}) and acceptable simulation control, the experimental results show a poor control of MMC (THD is 8.129%)

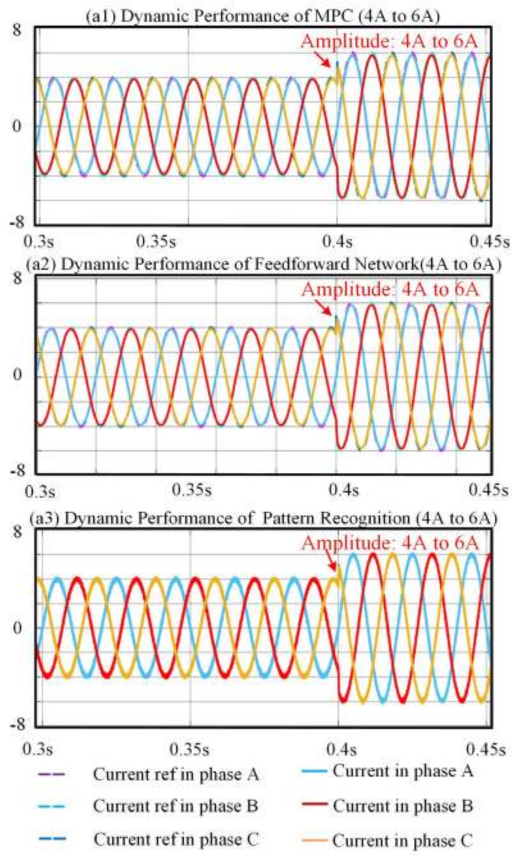


Fig .10. Dynamic performance within training range (a) NN regression controller, (b) NN pattern recognition controller.

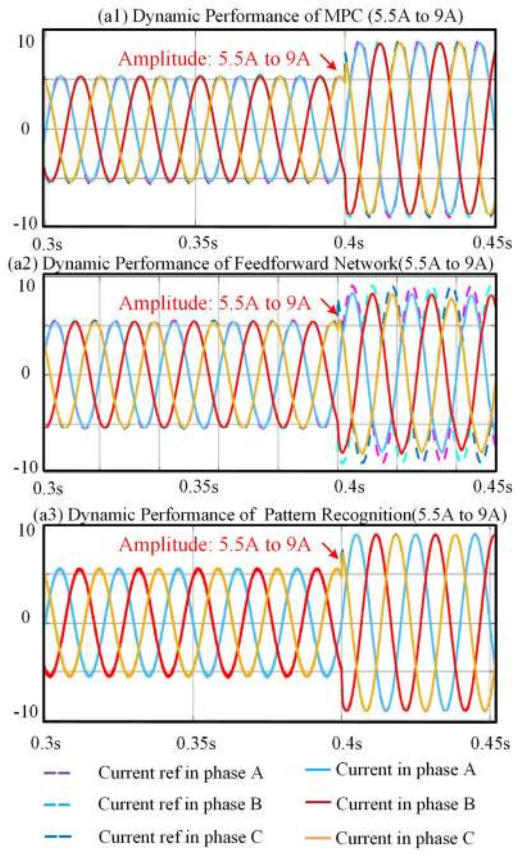


Fig .11. Dynamic performance out of training range (a) NN regression controller, (b) NN pattern recognition controller.

In contrast, the NN from one-net regression training (Eq. (18)) has a similar control performance with the conventional MPC in experiment which demonstrates the fact it has learned the control characteristics of MPC very well. Therefore, this regression NN is finally chosen as the best NN approach for MMC predictive control.

A. Dynamic Performance in Simulation

Fig. 10 shows the dynamic frequency of two NN controllers when the output current references are suddenly changed. The output current reference is suddenly stepped from 4A to 6A. From the results, both NN controllers can track the stepped references easily in this range with a very fast dynamic response in this range. Dynamic Performance out of the Training Range

In this paper, the data range of $i_s^\star$ is from -6A to 6A (Section IV. C). We tested the dynamic performance of two proposed controllers when the range of $i_s^\star$ is larger than the NN training range (9A). In Fig. 11, the results show that, the NN pattern recognition controller can track the reference even out of training range. However, NN regression controller, cannot properly track the reference, thus the robustness of the NN pattern recognition is better.

B. Computational Burden in Experiment

The networks are trained by the collected data offline, where the bias and weights are obtained through the training. The computational burden of the NNs is very low due to its simple operation with bias and weights, thus NNs are very suitable to be implemented in a DSP or dSPACE controller for the sake of time saving. The computational performance was verified using the dSPACE Profiler software. The turnaround time (i.e. code execution time) obtained using this software is given in Table II.

TABLE II
TURNAROUND TIME

	NN Regression	NNPR	MPC
Mean Turnaround Time (horizon length 1)	1.123 μs	6.073 μs	1.615 μs
Mean Turnaround Time (horizon length 2)	1.104 μs	6.088 μs	16.102 μs

From the experimental results, we can see that the NN regression has the lowest computational burden no matter what horizon length is. When the horizon length is high, the computational burden of MPC will increase significantly, but the NN based controllers keep the same computational burden which is a big gain.

C. Comparison of different methods

The advantages and disadvantages of the proposed methods are summarized in Table III. Regarding the computation burden, the NN regression method has the lowest computation no matter what horizon length is. What is more, the computation burden keeps the same when the horizon length increased. This is the key advantage of the NN based method

TABLE III
COMPARISON OF METHODS

	MPC	NN Regression	NNPR
Computational burden in horizon length 1	Low	Very Low	Medium
Computational burden in horizon length 2	High	Very Low	Medium
Output current THD performance	Low	Low	Medium
The ability to handle the input variables out of training range	Good	Medium	Good

compared to the MPC method. Regarding the control performance, the THD of output current is almost the same between MPC and NN regression. The THD of the NNPR is the worst. However, NNPR has a better ability to handle the input variables which beyond the training data range.

VI. Conclusion

In this paper, two machine learning (ML) based modular multilevel converter controllers are designed to emulate the model predictive controller (MPC): neural networking regression controller and neural network pattern recognition controller. The data extracted from MPC is used to train the ML controllers. Using the proposed ML controller, the computation burden of the controller will be reduced compared to MPC with regards to horizon length.

VII. References

[1] A. Lesnicar and R. Marquardt, "An innovative modular multilevel converter topology suitable for a wide power range," in *Proc. Int. Conf IEEE Bologna PowerTech*, 2003, vol. 3, pp. 272–277.

[2] P. Hu, R. Teodorescu, S. Wang, S. Li, and J. M. Guerrero, "A Currentless Sorting and Selection-Based Capacitor-Voltage-Balancing Method for Modular Multilevel Converters," *IEEE Trans. Power Electron.*, vol. 34, no. 2, pp. 1022–1025, 2019.

[3] H.-P. N. Antonios Antonopoulos, Lennart ¨ Angquist, "On Dynamics and Voltage Control of the Modular Multilevel Converter," pp. 4459–4466, 2009.

[4] M. A. Perez, J. Rodriguez, E. J. Fuentes, and F. Kammerer, "Predictive control of ac-ac modular multilevel converters," *IEEE Trans. Ind. Electron.*, vol. 59, no. 7, pp. 2832–2839, 2012.

[5] J. Moon, S. Member, J. Park, and D. Kang, "A Control Method of HVDC-Modular Multilevel Converter Based on Arm Current Under the Unbalanced Voltage Condition," *IEEE Trans. Power Deliv.*, vol. 30, no. 2, pp. 1–8, 2014.

[6] S. Li, X. Wang, Z. Yao, T. Li, and Z. Peng, "Circulating current suppressing strategy for MMC-HVDC based on nonideal proportional resonant controllers under unbalanced grid conditions," *IEEE Trans. Power Electron.*, vol. 30, no. 1, pp. 387–397, 2015.

[7] Q. Yang and M. Saeedifard, "Sliding mode control of the modular multilevel converter," *Conf. Proc. - IEEE Appl. Power Electron. Conf. Expo. - APEC*, vol. 2018–March, no. 1, pp. 1036–1043, 2018.

[8] A. Dekka, B. Wu, V. Yaramasu, R. L. Fuentes, and N. R. Zargari, "Model Predictive Control of High-Power Modular Multilevel Converters 2013; An Overview," *IEEE J. Emerg. Sel. Top. Power Electron.*, vol. PP, no. c, p. 1, 2018.

[9] Z. Zheng, Gong ; Xiaojie, Wu; Peng, Dai; Rongwu, "Modulated Model Predictive Control for MMC-Based Active Front-End Rectifiers Under Unbalanced Grid Conditions," *IEEE Trans. Power Electron.*, vol. 66, no. 3, pp. 2398–2409, 2019.

[10] J. W. Moon, J. S. Gwon, J. W. Park, D. W. Kang, and J. M. Kim, "Model Predictive Control with a Reduced Number of Considered States in a Modular Multilevel Converter for HVDC System," *IEEE Trans. Power Deliv.*, vol. 30, no. 2, pp. 608–617, 2015.

[11] J. Qin and M. Saeedifard, "Predictive control of a three-phase DC-AC Modular Multilevel Converter," *2012 IEEE Energy Convers. Congr. Expo. ECCE 2012*, pp. 3500–3505, 2012.

[12] J. Qin and M. Saeedifard, "Predictive control of a modular multilevel converter for a back-to-back HVDC system," *IEEE Trans. Power Deliv.*, vol. 27, no. 3, pp. 1538–1547, 2012.

[13] A. Dekka, B. Wu, and N. R. Zargari, "Minimization of DC-Bus Current Ripple in Modular Multilevel Converter Under Unbalanced Conditions," *IEEE Trans. Power Electron.*, vol. 32, no. 6, pp. 4125–4131, 2017.

[14] A. Dekka, B. Wu, V. Yaramasu, and N. R. Zargari, "Integrated model predictive control with reduced switching frequency for modular multilevel converters," *IEE Electr. Power Appl.*, vol. 11, pp. 857–863, 2017.

[15] B. Yang, F. Guo, Z. Wang, and X. Tong, "Priority Sorting Approach for Modular Multilevel Converter Based on Simplified Model," *IEEE Trans. Ind. Electron.*, vol. 65, no. 6, pp. 4819–4830, 2018.

[16] Y. Wang, W. Cong, M. Li, N. Li, M. Cao, and W. Lei, "Model Predictive Control of Modular Multilevel Converter with Reduced Computational Load," *2014 IEEE Appl. Power Electron. Conf. Expo. - APEC 2014*, pp. 1776–1779, 2014.

[17] Z. Gong, P. Dai, X. Yuan, X. Wu, and G. Guo, "Design and Experimental Evaluation of Fast Model Predictive Control for Modular Multilevel Converters," *IEEE Trans. Ind. Electron.*, vol. 63, no. 6, pp. 3845–3856, 2016.

[18] T. Geyer and D. E. Quevedo, "Performance of Multistep Finite Control Set Model Predictive Control for Power Electronics," in *IEEE Trans. Power Electron.*, vol. 30, no. 3, pp. 1633-1644, 2015.

[19] M. Hertneck, J. Köhler, S. Trimpe and F. Allgöwer, "Learning an Approximate Model Predictive Controller With Guarantees," in *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 543-548, 2018.

[20] D. Georges, "A Simple Machine Learning Technique for Model Predictive Control," *2019 27th Mediterranean Conference on Control and Automation (MED)*, Akko, Israel, 2019, pp. 69-74.

[21] I. S. Mohamed, S. Rovetta, T. D. Do, T. Dragicević and A. A. Z. Diab, "A Neural-Network-Based Model Predictive Control of Three-Phase Inverter With an Output LC Filter," in *IEEE Access*, vol. 7, pp. 124737-124749, 2019.

[22] M. Novak and T. Dragičević, "Supervised Imitation Learning of Finite Set Model Predictive Control Systems for Power Electronics," *IEEE Trans. Ind. Electron.*, vol. Early Acce, Jan. 2020.

[23] T. Dragicevic and M. Novak, "Weighting Factor Design in Model Predictive Control of Power Electronic Converters: An Artificial Neural Network Approach," *IEEE Trans. Ind. Electron.*, vol. PP, no. c, p. 1, 2018.

[24] T. Dragicevic, P. Wheeler, and F. Blaabjerg, "Artificial Intelligence Aided Automated Design for Reliability of Power Electronic Systems," *IEEE Trans. Power Electron.*, vol. 34, no. 8, pp. 7161–7171, 2019.

[25] P. Michael, and P. Thomas, *MATLAB Machine Learning Recipes: A Problem-solution Approac*h. Apress, 2019.

[26] B. Christopher M. *Pattern Recognition and Machine Learning.* Springer, 2006.

[27] K. Hornik, M. Stinchcombe, and H. White, "' Multilayer feedforward networks are universal approximators ' Theoretical properties of multilayer feedforward networks," vol. 2, pp. 359–366, 1989.

[28] L. Bessegato, S. Member, K. Ilves, L. Harnefors, and S. Norrga, "Effects of Control on the AC-Side Admittance of a Modular Multilevel Converter," *IEEE Trans. Power Electron.*, vol. PP, no. c, p. 1, 2018.

[29] K. Ilves, S. Norrga, L. Harnefors, and H. P. Nee, "On energy storage requirements in modular multilevel converters," *IEEE Trans. Power Electron.*, vol. 29, no. 1, pp. 77–88, 2014.

[30] K. Sharifabadi, L. Harnefors, H.-P. Nee, S. Norrga, and R. Teodorescu, *Design, Control and Application of Modular Multilevel Converters for HVDC Transmission Systems*. 2016.

[31] P. Cortes, J. Rodriguez, C. Silva, and A. Flores, "of a Three-Phase Inverter," *IEEE Trans. Ind. Electron.*, vol. 59, no. 2, pp. 1323–1325, 2012.

[32] *Hinton, G. E.; Osindero, S.*; Teh, Y. W. (2006). "A Fast Learning Algorithm for Deep Belief Nets" Neural Computation. 18 (7): 1527–1554.