2019 Sixth Indian Control Conference (ICC)
December 18-20, 2019. IIT Hyderabad, India

Embedded Implementation of Deep Learning-based Linear Model Predictive Control

Saket Adhau[1], Sayli Patil[1], Deepak Ingole[2], and Dayaram Sonawane[1]

***Abstract*— Model predictive control (MPC) has emerged as an excellent control strategy owing to its ability to include constraints in the control optimization and robustness to linear as well as highly non-linear systems. There are many challenges in real-time implementation of MPC on embedded devices, including computational complexity, numerical instability, and memory constraints. Advances in machine learning-based approaches have widened the scope to replace the traditional and intractable optimization algorithms with advanced algorithms. In this paper, a novel deep learning-based model predictive control (DNN-MPC) is presented. The proposed MPC uses recurrent neural network (RNN) to accurately predict the future output states based on the previous training data. Using deep neural networks for the real-time embedded implementation of MPC, on-line optimization is completely eliminated leaving only the evaluation of some linear equations. Closed-loop performance evaluation of the DNN-MPC is verified through hardware-in-loop (HIL) co-simulation on ARM microcontroller and a 4x speed-up in computational time for a single iteration is achieved over the conventional MPC. Detailed analysis of DNN-MPC complexity (speed and memory requirement) is presented and compared with traditional MPC. Results show that the proposed DNN-MPC performs faster and with less memory footprints while retaining the controller performance.**

***Index Terms*— Machine learning, predictive control for linear systems, anesthesia control**

I. INTRODUCTION

Recent advances in machine learning-based approaches for optimization and control systems is leading to new paradigms to design and implement real-time embedded MPC for industrial use. Leveraging these machine learning-based methods for control systems, control laws could be extracted using improvised optimization algorithms and feature extraction methods [1].

Model predictive control has emerged as an excellent controller for complex systems with both, slow and fast dynamics. MPC is much popular as it the solves numerical optimization problem at every sampling instant to calculate a sequence of optimal control inputs over a prediction horizon subjected to system dynamics and constraints [2]. This optimization loop is repeated over the complete time period giving optimized outputs at every instant. The above discussed capabilities and robustness of MPC has made its way in industries and academics. However, solving the optimization problem is indeed a computationally expensive task restricting the usage of MPC with systems having slow dynamics. Furthermore, MPC requires an accurate description of the plant model and is susceptible to uncertainties in plant parameters [3].

For a successful real-time embedded implementation of MPC, microcontrollers or microprocessors should be able to solve the optimization problem at each sampling instant. This would require an efficient C/C++ code to be deployed on embedded devices [4]. Methods such as fast MPC [5] and alternating directions method of multipliers (ADMM) [6] have been used towards efficient implementation of MPC and optimization on embedded platforms. Apart from efficient embedded optimization solvers, another solution is to use explicit MPC (EMPC), which uses pre-computed control law to search for optimal solutions [7].

Reinforcement learning (RL), deep neural networks, and other statistical learning methods have been consistently used for learning-based MPC (LBMPC). In this paper, we present a machine learning-based approach to train MPC controllers as one of the alternative solution for on-line optimization. Authors in [8], used reinforcement learning in MPC for markov decision processes which can be used in areas like vehicle automation, traffic control, and logistics. Authors also validated the stability of proposed controller on models such as quadrotor (see, [9]), for HVAC models (see, [10]), etc. In addition to this, a robust learning model predictive control (RLMPC) for iterative tasks such as automatic racing cars is presented (see, [11]).

In past years, efforts have been made to design and implement anesthesia closed-loop control system using numerous control schemes which includes fixed-gain (proportional-integral-derivative controller (PID) [12]), knowledge-based (fuzzy logic [13]), and model-based (MPC [14]), etc. which performed well in case of inter-variability patient model, disturbance rejection, and constraints handling. Recently, there has been some works on reinforcement learning - based anesthesia control (see, [15]). Their results show that the RL-based control strategy is promising and robust to system uncertainties.

The objective of this paper is to achieve comparable performance as of linear MPC on embedded platforms while replacing the optimization part using deep neural network (DNN) trained controller. The trained controller is supposed to mimic the control law of MPC without any significant performance loss Previously, deep neural networks have been employed for linear, nonlinear, and hybrid MPC (see, [16], [17]). As a case study, a single-input, single-output intravenous anesthesia drug delivery model [14] has been chosen.

[1] Department of Instrumentation and Control Engineering, College of Engineering Pune, Shivajinagar 411005, India. {adhauss17.instru,patilsd17.instru, dns.instru}@coep.ac.in
[2] University of Lyon, IFSTTAR, ENTPE, Lyon 69518, France. deepak.ingole@ifsttar.fr

We show how this controller, drastically reduces memory footprint and computational complexity and is easily deployable on embedded platforms. Through this application we want to show the applicability of the proposed methods on safety critical applications.To the best of author's knowledge, this is the first work that has investigated the performance of deep learning-based MPC for anesthesia control.

II. Model Predictive Control and Deep Neural Network

This section provides a brief introduction on two most important concepts viz: deep neural networks and model predictive control. We will show how these two successful techniques i.e., MPC and DNN can be integrated to design a robust and promising control system.

A. Linear Model Predictive Control

In this work, we assume that the state-update and output equations are both discrete-time and linear-time invariant (LTI) in the form as follows:

$$x(t+1) = Ax(t) + Bu(t), \tag{1a}$$
$$y(t) = Cx(t) + Du(t), \tag{1b}$$

where, $x(t) \in \mathbb{R}^{n_x \times 1}, y(t) \in \mathbb{R}^{n_y \times 1}$, and $u(t) \in \mathbb{R}^{n_u \times 1}$ are differential state vector, output vector, and control input vector of the plant, respectively. Also, $A \in \mathbb{R}^{n_x \times n_x}, B \in \mathbb{R}^{n_x \times n_u}$, $C \in \mathbb{R}^{n_y \times n_x}$, and $D \in \mathbb{R}^{n_y \times n_u}$ are the system's input and output matrices describing the plant dynamics. MPC solves constrained finite-time optimal control (CFTOC) problem for reference tracking at each iteration which is given as follows:

$$\min_{U_N} \sum_{k=0}^{N-1} (y_k - y_{\mathrm{r},k})^T Q (y_k - y_{\mathrm{r},k}) + \Delta u_k^T R \Delta u_k, \tag{2a}$$

s.t.

$$\begin{aligned}
&x_{k+1} = Ax_k + Bu_k, && k = 0, \ldots, N-1, && \text{(2b)}\\
&y_k = Cx_k + Du_k, && k = 0, \ldots, N-1, && \text{(2c)}\\
&\Delta u_k = u_k - u_{k-1}, && k = 0, \ldots, N-1, && \text{(2d)}\\
&u_{\min} \le u_k \le u_{\max}, && k = 0, \ldots, N-1, && \text{(2e)}\\
&\Delta u_{\min} \le \Delta u_k \le \Delta u_{\max}, && k = 0, \ldots, N-1, && \text{(2f)}\\
&x_{\min} \le x_k \le x_{\max}, && k = 0, \ldots, N-1, && \text{(2g)}\\
&y_{\min} \le y_k \le y_{\max}, && k = 0, \ldots, N-1, && \text{(2h)}\\
&u_{-1} = u(t-1), && && \text{(2i)}\\
&x_0 = x(t), && && \text{(2j)}
\end{aligned}$$

where, x_k, u_k, and y_k are the predictions of differential states, control input, and output at instant k. The term in (2a) is the objective function with prediction horizon N. The matrices $Q \in \mathbb{R}^{n_x \times n_x}$ and $R \in \mathbb{R}^{n_u \times n_u}$ are the weighting matrices, with conditions $Q \succeq 0$ to be positive semi-definite, and $R \succ 0$ to be positive definite. The term y_r, is the reference trajectory of the output. Finally, (2e)-(2h) are polyhedral constraint sets of state, input, and output, respectively. Solving the CFTOC problem in (2) for minimization of cost function for initial conditions x_0, we get sequence of optimal control input $(U_N^\star = u_0^\star, \ldots, u_{N-1}^\star)$ over the prediction horizon (N). Only the first term of the control sequence $(u_0^\star)$ is fed to the plant and current states $(x(t))$ are measured which are then again sent back to the controller. Using updated values of the state measurement x_0 entire procedure is repeated for next time instant $t+1$. This concept is known as receding horizon control (RHC) [18, Chapter 12].

In spite of this tremendous success of MPC, there are still many restrictions to the use of MPC for real-time implementation. As previously mentioned, MPC has a huge burden of solving complex optimization problem within every sampling instant. Further, linear MPC has huge matrices which take up the major chunks of available memory on the embedded devices which restricts the implementation of MPC for small systems. However, deep learning methods for MPC are able to accurately approximate the linear MPC control law reducing the computational complexity, numerical instability and memory footprints of the controller [19].

B. Deep Neural Network

Deep learning has gained immense recognition due to the wide and easy applicability to many applications in almost every domain. Deep learning architecture includes reinforcement learning, convolutional neural networks, and variational encoders (VAEs) which provide a robust mathematical framework for supervised as well as unsupervised learning. Using the power of multi-layer deep architecture, huge labeled data sets can be trained and analyzed.

Mathematical validation to DNN can be found in [20], [21], wherein they proved that with enough hidden layers and a linear output layer can be used to learn any arbitrary function using DNN. Training should be done meticulously as over-fitting or under-fitting might compromise the accuracy of the trained model. Also, there are no fixed rules to determine the required number of hidden layers to locate the global optimal.

In this paper, we explicitly investigate the use of deep neural networks for MPC. Deep neural networks are a combination of multiple non-linear processing layers which are made up of simple elements in parallel operation and are inspired by biological nervous systems. These networks consist of multiple hidden layers, an input layer and an output layer which are connected using nodes and neurons. Recurrent neural networks prove to be excellent for sequential data which throughput state of the art performance for the tasks including speech recognition, machine translation, language modeling, and time-series data [22]–[24].

C. Structure of RNN

The model which is to be trained is approximated using RNN. The recurrent neural networks basically consist of two components: encoder and decoder, see Fig. 2. Actual future states prediction is done by the decoder module which consists of N cells, one for every single time step. The encoder takes care of long term dynamics and predicts the latent states [25]. Latent states or latent variables are simply

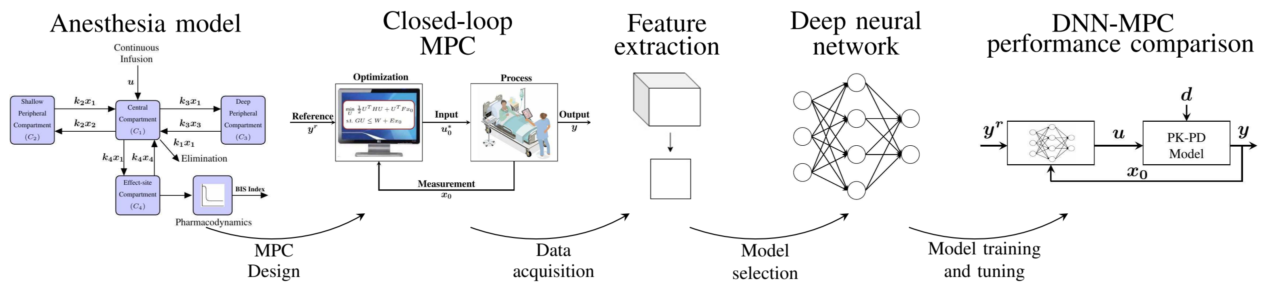

Fig. 1. Schematic of the DNN-MPC. From left to right: four compartment PKPD model of the patient is used as a model to design MPC. Through simulation in MATLAB, random data is generated to train DNN-MPC controller to replicate the original control law of the MPC. Eventually, both the controllers: MPC and DNN-MPC are compared.

the hidden variables or states in a system. Only a small portion of the decoder cell is contained in the encoder cell to predict the latent variables.

A sequence of past observations and current control input is fed as input to each RNN cell. These RNN cells are further divided into three parts which capture long term dynamics of the system, the effect of the control input and current dynamics. The input of every single input cell, i is divided into an observable time series $(z,h)_{i-2d-1,\ldots,i}$ and the corresponding sequence of control inputs, d being the delay is $h_{i-d,\ldots,i}$. This combined system is used to capture the system dynamics and perform the predictions.

Problems including MPC coupled with RNN can be solved using Levenberg Marquardt, which interpolates between gradient descent method and gauss-newton algorithm (GNA). Generalized back-propagation through time is used in the process. In the next section, we will see how system dynamics are captured using previous data, the training of the controller and its implementation.

III. Case study: Anesthesia Control

To demonstrate real-time implementation of proposed DNN-MPC on embedded platforms, a biomedical application- anesthesia control system is considered. Hospitals these days have well equipped operating rooms (ORs). Advance surgical techniques bring new biomedical technologies and more instrumentation in the ORs, which often results in complex configurations and big size machinery which occupies more space. Efforts have been made to improve the quality control and replace big size machines with embedded systems (see, [26]).

Anesthesia is an amount of anesthetic drug(s) given to the patient to control his state of arousal during surgery. In traditional practice, the amount of initial drug dose and it's variation during surgery is decided by an anesthesiologist depending on patient's physical factors such as age, weight, height, and gender. Manual delivery of drugs can cause over or underdosing of the anesthetic which can put patient's safety and health at risk. Due to these issues, closed-loop control of anesthesia drug delivery is a challenging research topic over the last few decades.

For automatic feedback control of anesthesia which fuses drugs based on the anesthetic level of a patient, we consider four compartments, single-input (drug rate) single-output (concentration) model of the patient with Hill's Sigmoid model to measure the depth of anesthesia (DoA) using a bi-spectral index (BIS). The output represents Propofol concentration which is controlled by Propofol infusion rate.

Anesthesia model is a combination of two elements, a pharmacokinetic and pharmacodynamic (PKPD) model. Pharmacokinetic is a study of absorption, distribution, metabolism, and excretion (ADME) of bio-active compounds in a higher organism. As shown in Fig. 3, the pharmacokinetic model consists of four-compartment i.e., central compartment (C_1), shallow peripheral compartment (C_2), deep peripheral compartment (C_3), and effect-side compartment (C_4). Pharmacodynamics is given by Hill's sigmoid model which is a nonlinear and static relationship between concentration and BIS. For brief explanation of model see [27], [28]. The state-space representation of the pharmacokinetic model is given by,

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{3a}$$

$$y(t) = Cx(t) + Du(t), \tag{3b}$$

where, $x(t)$ is a state vector representing Propofol concentration in different compartments, $u(t)$ is the controlled vector representing Propofol infusion rate, $y(t)$ is the output vector representing effect site concentration of Propofol, and A, B, C, and D are pharmacokinetic parameters which are given by,

$$A = \begin{bmatrix} -\dfrac{k_1+k_2+k_3+k_4}{V_1} & \dfrac{k_2}{V_1} & \dfrac{k_3}{V_1} & \dfrac{k_4}{V_1} \\ \dfrac{k_2}{V_2} & -\dfrac{k_2}{V_2} & 0 & 0 \\ \dfrac{k_3}{V_3} & 0 & -\dfrac{k_3}{V_3} & 0 \\ \dfrac{k_4}{V_4} & 0 & 0 & -\dfrac{k_4}{V_4} \end{bmatrix},$$

$$B = \begin{bmatrix} \dfrac{1}{V_1} & 0 & 0 & 0 \end{bmatrix}^T, C = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T, \text{and } D = \begin{bmatrix} 0 \end{bmatrix}.$$

Here, the subscripts 1, 2, 3, and 4 correspond to the central, shallow peripheral, deep peripheral and effect site compart-

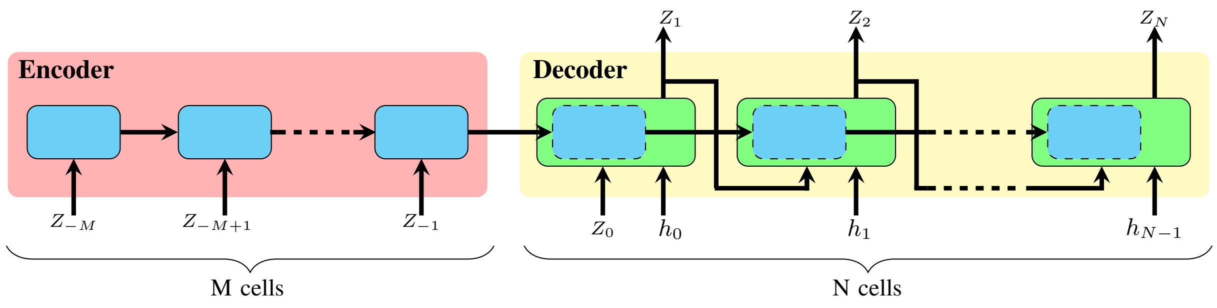

Fig. 2. DNN encoder-decoder network. Here, green cells depict decoder cell whereas encoder cells are shown in blue. Encoder cells are just a part of the decoder cells.

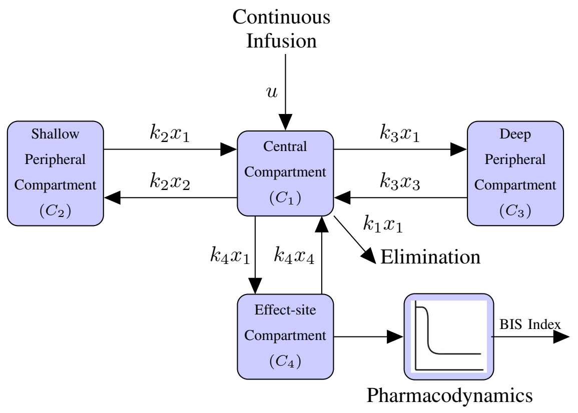
Continuous Infusion
u
Shallow Peripheral Compartment (C_2)
Central Compartment (C_1)
Deep Peripheral Compartment (C_3)
k_2x_1
k_2x_2
k_3x_1
k_3x_3
k_1x_1
Elimination
k_4x_1
k_4x_4
Effect-site Compartment (C_4)
BIS Index
Pharmacodynamics

Fig. 3. Three-compartment pharmacokinetic patient model with the effect-site compartment [27], [28].

ment, respectively. The parameter k and V represent clearance, and volume of different compartments which depends on the patients age and body weight. The relation between age, weight, clearance and volume is given in Table I.

The relationship between the effect-site concentration of Propofol and the BIS index value in the pharmacodynamic model is given by the following Hills sigmoid $E_{\max}$ model [27]:

$$BIS(t) = E_0 - E_{\max}\frac{y(t)^{\gamma}}{y(t)^{\gamma} + C_{50}^{\gamma}}, \tag{4a}$$

where $E_{\max} = E_0$, E_0 denotes BIS value before starting the Propofol infusion, $E_{\max}$ denote the change of the BIS index corresponding to the infinite Propofol concentration, γ denotes the Hill's coefficient, and C_{50} denote the effect site concentration corresponding to $\frac{E_{\max}}{2}$.

The objective is to control BIS index such that it will keep track of a prescribed reference by manipulating the input variable, Propofol infusion rate while strictly following the constraints.

IV. Embedded Implementation of DNN-MPC

The following sections describes the implementation of deep neural networks based MPC on a low-cost microcontroller and the method to implement it on embedded hardware. Numerical robustness, memory limitations, consideration of worst-case scenarios, tolerance against instability and safety are some of the challenges in the embedded realization of MPC [29].

TABLE I
Pharmacokinetic parameter values given as a function of patient's AGE (years) and BW (kg).

Parameter	**Value**
k_1	$0.0595^{0.75}$ L/min (age ≤ 60)
	$(0.0595\mathrm{BW}^{0.75} - 0.45\mathrm{AGE} + 2.7)$ L/min (age > 60)
k_2	$0.0969\mathrm{BW}^{0.62}$ L/min
k_3	$0.0889\mathrm{BW}^{0.55}$ L/min
k_4	0.12 L/min
V_1	$1.72\mathrm{BW}^{0.71}\mathrm{AGE}^{-0.39}$ L
V_2	$3.32\mathrm{BW}^{0.61}$ L
V_3	266 L
V_4	$0.01V_1$ L

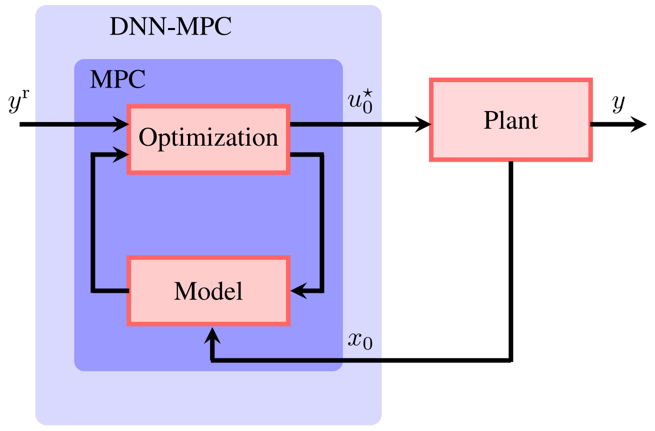

Fig. 4. Replacing MPC with deep neural network trained model.

A. *Training of the Controller*

Training neural networks is one of the most crucial tasks. All the dynamics of the plant should be taken into account while generating the data and training the controller. Fig. 4 shows the framework in which the MPC is replaced by DNN-MPC controller. We generated data for training the controller using simulations. Linear MPC was designed and simulated in MATLAB. The designed MPC had a sampling time of 1 [s] and prediction horizon, $N = 15$. For a robust and rigorous training, we used 50000 data-points in which 70% was allocated for training while 15% each for testing and validation, respectively.

The data was trained using MATLAB-based Neural Network Time Series function, invoked using `trainNetwork` [30]. The training was performed using Levenberg_Marquardt [31] method, as this algorithm usually takes less time but more memory. Training will automatically stop once there is no improvement in generalization as indicated by an increase in the mean square error (MSE) of the validation samples (upto 1000 epochs). The model was trained on a Windows 10 PC with 3.5 GHz Intel® Quad Core i7 with 16 GB of RAM. Training was done offline and no online training method has been used. Also the designed model is fixed for a particular patient which can be considered as a limitation to this method.

Results of the training are shown in Table II with training parameters as 12 hidden neurons and a single delay. From the table, we can see that mean squared error(MSE) and Regression (R) is close to zero which shows that the controller is properly trained with minimal errors.

TABLE II
TRAINING STATISTICS.

Scenario	Target Values	MSE	R
Training	35000	6.28241×10^{-9}	9.99999×10^{-1}
Validation	7500	1.14839×10^{-6}	9.94938×10^{-1}
Testing	7500	1.40538×1^{-8}	9.99999×10^{-1}

TABLE III
MEMORY FOOTPRINTS AND RUN-TIME RESULTS OF PROPOSED DNN-MPC AND LMPC.

Controller	Memory [%]		Run-time
	% Data Usage	% Program Usage	[ms]
Linear MPC	3.50	4.90	11.354
DNN MPC	3.01	3.92	2.999

B. *Implementation and Results*

We have demonstrated the design of deep neural network based MPC which approximates MPC control law. Complete work-flow of the mechanism used while designing and implementing DNN-MPC is shown in Fig. 1. Particularly in this model, the safety of the controller is of prime importance. The controller should be able to maintain the desired levels without any major fluctuations in the input. The BIS level is typically maintained between $40 - 60$ (score) to guarantee the effect of the drug on the patient. The HIL simulations in Fig. 5 show that deep neural networks have approximated the original LMPC control law precisely and follow the reference smoothly. As seen from the Fig. 5, it is important to note that, in spite of some persisting steady-state errors of very low magnitude, the controller does not show any kind of overshoot or undershoot, which is better for the health of the patient. These persisting steady-state error can be reduced by fine-tuning and training the model with more data. Also, the controller input stays between the prescribed infusion rate which should be between $0 - 20$ [mg/kg/h].

However, the main focus of this paper is to reduce the computational complexity and the numerical instability associated while solving the optimization problem on-line. The proposed DNN-MPC addresses this problem. The proposed controller is implemented on ARM Cortex M3 microcontroller having 512 kB flash memory and 96 kB of SRAM running at 84 MHz. From Table III, it can be seen that the per iteration computational time is significantly reduced from 11.354 ms to 2.99 ms.

The total memory consumption of DNN-MPC as compared to LMPC, when deployed on ARM microcontroller, is shown in the results table. It can also be inferred from Table III, that even though the reduction in memory footprint is not that pronounced for this particular model, we still believe that for larger systems having more number of states and constraints, the proposed DNN-MPC will be much more memory efficient.

V. CONCLUSIONS

This work proposes the use of deep neural networks to approximate linear MPC control law, efficiently and effectively with minimal computational efforts for real-time embedded implementation. The trained DNN model was verified using HIL co-simulation with ARM Cortex M3 microcontroller. This work showcases the detailed performance comparison of linear MPC and DNN trained MPC in terms of memory utilization and time to solve one iteration. Closed-loop simulation results for anesthesia drug delivery system using DNN-MPC are presented as well. Presented results show that the proposed approach guarantees the safety of the patient and performs faster which make it as a promising approach for safety-critical applications running on embedded systems. Looking at the advantages of this particular approach, some of the limitations can be overseen like the ways to generate the data, getting the actual data instead of using the simulated data and the need to train the network everytime for a new patient/model. Future work will include to make complex non-linear MPC solutions using deep neural networks to use them on embedded hardware.

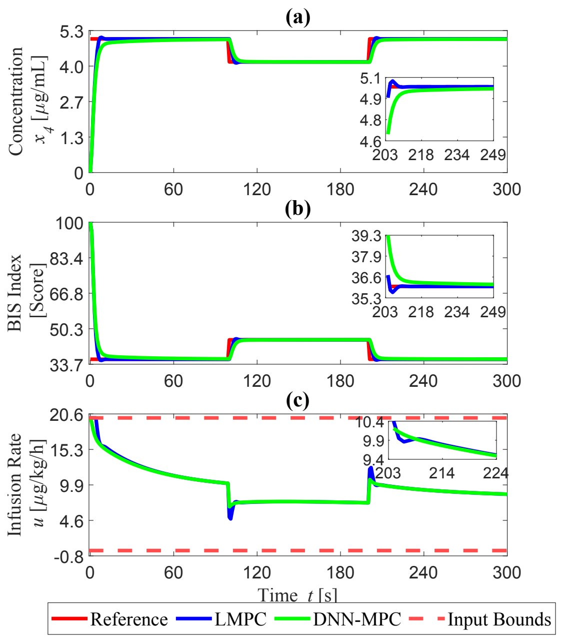

Fig. 5. Performance comparison of proposed DNN-MPC with LMPC for anesthesia control problem.

Acknowledgment

We gratefully acknowledge the support from R & D center of the COEP. Deepak Ingole would like to thank for a financial contribution from the ERC under the European Unions Horizon 2020 research and innovation program (grant agreement no. 646592 MAGnUM project).

References

[1] T. Baumeister, S. L. Brunton, and J. N. Kutz, "Deep learning and model predictive control for self-tuning mode-locked lasers," *JOSA B*, pp. 617–626, 2018.

[2] X. Yang, D. W. Griffith, and L. T. Biegler, "Nonlinear programming properties for stable and robust NMPC," *IFAC-PapersOnLine*, pp. 388–397, 2015.

[3] Y. Xie, R. Ghaemi, J. Sun, and J. S. Freudenberg, "Model predictive control for a full bridge DC/DC converter," *IEEE Transactions on Control Systems Technology*, pp. 164–172, 2011.

[4] S. Adhau, S. Patil, D. Ingole, and D. Sonawane, "Implementation and Analysis of Nonlinear Model Predictive Controller on Embedded Systems for Real-Time Applications," in *2019 European Control Conference (ECC)*. IEEE, 2019.

[5] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on control systems technology*, pp. 267–278, 2009.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, pp. 1–122, 2011.

[7] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit solution of model predictive control via multiparametric quadratic programming," in *Proceedings of the 2000 American Control Conference. ACC*, 2000, pp. 872–876.

[8] R. R. Negenborn, B. D. Schutter, and M. A. Wiering, "Experience-based model predictive control using reinforcement learning," *Proceedings of the 8th TRAIL Congress 2004 A World of Transport, Infrastructure and Logistics*, 2004.

[9] P. Bouffard, A. Aswani, and C. Tomlin, "Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 279–284.

[10] A. Aswani, N. Master, J. Taneja, D. Culler, and C. Tomlin, "Reducing transient and steady state electricity consumption in HVAC using learning-based model-predictive control," *Proceedings of the IEEE*, pp. 240–253, 2011.

[11] U. Rosolia and F. Borrelli, "Learning model predictive control for iterative tasks. a data-driven control framework," *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 1883–1896, 2017.

[12] F. Padula, C. Ionescu, N. Latronico, M. Paltenghi, A. Visioli, and G. Vivacqua, "Optimized PID control of depth of hypnosis in anesthesia," *Computer methods and programs in biomedicine*, vol. 144, pp. 21–35, 2017.

[13] J. A. Mendez, A. Leon, A. Marrero, J. M. Gonzalez-Cava, J. A. Reboso, J. I. Estevez, and J. F. Gomez-Gonzalez, "Improving the anesthetic process by a fuzzy rule based medical decision system," *Artificial intelligence in medicine*, vol. 84, pp. 159–170, 2018.

[14] D. Ingole and M. Kvasnica, "FPGA implementation of explicit model predictive control for closed loop control of depth of anesthesia," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 483–488, 2015.

[15] R. Padmanabhan, N. Meskin, and W. M. Haddad, "Closed-loop control of anesthesia and mean arterial pressure using reinforcement learning," *Biomedical Signal Processing and Control*, vol. 22, pp. 54–64, 2015.

[16] S. S. P. Kumar, A. Tulsyan, B. Gopaluni, and P. Loewen, "A Deep Learning Architecture for Predictive Control," *IFAC-PapersOnLine*, pp. 512–517, 2018.

[17] S. Lucia, D. Navarro, B. Karg, H. Sarnago, and O. Lucia, "Deep Learning-Based Model Predictive Control for Resonant Power Converters," *arXiv preprint arXiv:1810.04872*, 2018.

[18] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[19] B. Karg and S. Lucia, "Deep learning-based embedded mixed-integer model predictive control," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 2075–2080.

[20] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, pp. 359–366, 1989.

[21] ——, "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," *Neural networks*, pp. 551–560, 1990.

[22] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[23] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *2012 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2012, pp. 234–239.

[24] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[25] K. Bieker, S. Peitz, S. L. Brunton, J. N. Kutz, and M. Dellnitz, "Deep Model Predictive Control with Online Learning for Complex Physical Systems," *arXiv preprint arXiv:1905.10094*, 2019.

[26] D. Ingole, "Embedded Implementation of Explicit Model Predictive Control," Ph.D. dissertation, IAM FCHPT STU in Bratislava, 2017.

[27] J. Schüttler and H. Ihmsen, "Population pharmacokinetics of propofola multicenter study," *Anesthesiology: The Journal of the American Society of Anesthesiologists*, pp. 727–738, 2000.

[28] Y. Sawaguchi, E. Furutani, G. Shirakami, M. Araki, and K. Fukuda, "A model-predictive hypnosis control system under total intravenous anesthesia," *IEEE transactions on biomedical engineering*, pp. 874–887, 2008.

[29] T. A. Johansen, "Toward Dependable Embedded Model Predictive Control," *IEEE Systems Journal*, pp. 1208–1219, 2017.

[30] "Train neural network for deep learning Time-Series Prediction and Modeling," https://in.mathworks.com/help/deeplearning/ref/trainnetwork.html.

[31] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," in *Numerical Analysis*, G. A. Watson, Ed., Berlin, Heidelberg, 1978, pp. 105–116.