IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 16 June 2024, accepted 13 August 2024, date of publication 16 August 2024, date of current version 29 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3444899

RESEARCH ARTICLE

Advancing Robotic Control: Data-Driven Model Predictive Control for a 7-DOF Robotic Manipulator

HAITHAM EL-HUSSIENY[1], IBRAHIM A. HAMEED[2], (Senior Member, IEEE), TAMER F. MEGAHED[3,4], AND AHMED FARES[5,6]

[1]Department of Mechatronics and Robotics Engineering, Egypt-Japan University of Science and Technology (E-JUST), Alexandria 21934, Egypt
[2]Department of ICT and Natural Sciences, Norwegian University of Science and Technology, 6009 Ålesund, Norway
[3]Department of Electrical Power Engineering, Egypt-Japan University of Science and Technology (E-JUST), Alexandria 21934, Egypt
[4]Electrical Engineering Department, Faculty of Engineering, Mansoura University, Mansoura 35516, Egypt
[5]Department of Computer Science and Engineering, Egypt-Japan University of Science and Technology (E-JUST), Alexandria 21934, Egypt
[6]Electrical Engineering Department, Faculty of Engineering at Shoubra, Benha University, Cairo 11629, Egypt

Corresponding authors: Haitham El-Hussieny (haitham.elhussieny@ejust.edu.eg) and Ibrahim A. Hameed (ibib@ntnu.no)

ABSTRACT In this study, we applied deep learning to improve the control of a KUKA LBR4 7 Degrees of Freedom (DOF) robotic arm. We developed a dynamic model using a comprehensive dataset of joint angles and actuator torques obtained from pick-and-place operations. This model was incorporated into a Model Predictive Control (MPC) framework, enabling precise trajectory tracking without the need for traditional analytical dynamic models. By integrating specific constraints within the MPC, we ensured adherence to operational and safety standards. Experimental results demonstrate that deep learning models significantly enhance robotic control, achieving precise trajectory tracking. This approach not only surpasses traditional control methods in terms of accuracy and efficiency but also opens new avenues for research in robotics, showcasing the potential of deep learning models in predictive control techniques.

INDEX TERMS Deep learning, model predictive control (MPC), robotic arm, trajectory tracking.

I. INTRODUCTION

Robotic manipulation and movement describe how robotic systems engage with and transform their surroundings through meticulous control over their mechanical components [1]. Robotic manipulators are versatile mechanical devices used extensively in various industrial, medical, and research applications due to their precision, efficiency, and adaptability. In manufacturing, they are employed for tasks such as assembly, welding, and material handling, significantly enhancing productivity and safety by performing repetitive and hazardous tasks with high accuracy [2]. In the medical field, robotic manipulators are integral to minimally invasive surgeries, providing surgeons with enhanced precision and control, which leads to reduced patient recovery times and improved surgical outcomes [3]. Additionally, these manipulators play a crucial role in space exploration, where they perform tasks like satellite servicing and assembly of large structures in the harsh environment of space [4]. The versatility and reliability of robotic manipulators make them indispensable tools across various domains.

Feedback control at the joint level of robotic manipulators is essential for ensuring precision, stability, and responsiveness during various tasks [5]. By continuously monitoring and adjusting the joint positions, velocities, and torques based on sensor feedback, feedback control systems can correct errors in real-time and maintain the desired trajectory [6]. This is crucial in applications requiring high accuracy, such as in assembly lines or surgical procedures, where even minor deviations can lead to significant errors or safety hazards. One of the key benefits of joint-level feedback control is its ability to compensate for uncertainties and disturbances. These can include variations in payload,

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

external forces, or changes in the robot's dynamic parameters over time. By providing immediate corrections, feedback control enhances the robustness and reliability of robotic operations [7]. Furthermore, feedback control allows for improved performance in dynamic environments. In tasks where the robot interacts with varying objects or operates in unstructured settings, feedback control ensures that the manipulator can adapt to changes and maintain performance without requiring manual recalibration or intervention [8].

Model-based control of robotic manipulators involves developing mathematical models that represent the robot's dynamics and kinematics, which are then used to predict and optimize the robot's behavior in real-time [9], [10], [11]. By incorporating detailed models, controllers can anticipate the effects of control actions, compensate for nonlinearities, and account for interactions between different joints, leading to enhanced performance [12]. This level of control is critical in industries such as aerospace, where precision and reliability are paramount. Moreover, model-based approaches enable the implementation of advanced control strategies, such as Model Predictive Control (MPC), which optimizes control inputs over a future time horizon to achieve desired performance while respecting constraints [13], [14].

However, implementing model-based control presents several challenges. One of the primary difficulties is accurately identifying and modeling the robot's dynamics, which can be highly nonlinear and subject to various uncertainties, such as friction, backlash, and external disturbances [15]. Additionally, real-time computation requirements for complex models can be demanding, necessitating powerful computational resources and efficient algorithms [16]. Another challenge is ensuring robustness to model inaccuracies and parameter variations, which requires sophisticated adaptive or robust control techniques to maintain performance in the presence of uncertainties.

Data-driven methods, particularly neural networks, have shown potential in accurately modeling these nonlinear dynamical effects without relying on exhaustive mathematical modeling [17], [18]. Integrating such models as surrogates in MPC systems could facilitate meeting the real-time control requirements for robotic manipulators. Previous studies utilizing predictive controllers have typically employed one of two approaches: using linear predictive models by linearizing the system around a fixed point [19], or implementing gain scheduling to establish a multi-level controller where each level handles a specific operational mode [20]. While these methods can facilitate real-time operation, they tend to provide limited accuracy in predicting system responses. To improve prediction accuracy, some research has introduced nonlinear predictive models, such as in [21]. However, these models often fail to support real-time operation because solving the nonlinear equations involved requires extensive computation [22]. Meanwhile, alternative control strategies to MPC were employed. These strategies are characterized either by their non-predictive nature, as discussed in [23], or by their avoidance of online optimization, as seen in the works by [24] and [25].

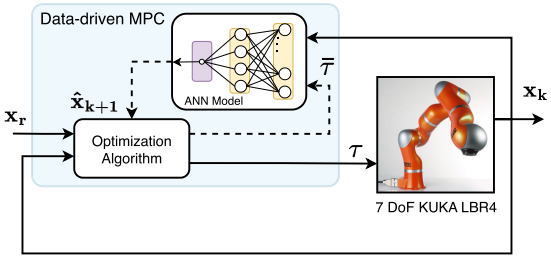

FIGURE 1. A block diagram depicting the proposed data-driven model predictive control (MPC) strategy used to control the KUKA LBR4 robotic manipulator.

In this study, we present a data-driven dynamic model for a seven Degrees-of-Freedom (DOF) KUKA LBR4 robotic manipulator utilizing deep learning techniques. This model is integrated into the Model Predictive Control (MPC) framework to enhance trajectory tracking, eliminating the need for traditional analytical dynamic models. We developed and validated the deep learning dynamic model using an extensive dataset of joint angles and actuator torques. Furthermore, we performed a theoretical analysis to ensure the stability and feasibility of the deep learning-based MPC. The model was successfully incorporated into the MPC framework with additional constraints to improve operational safety and efficiency. Experimental results demonstrated improved trajectory tracking capabilities, and we discussed the potential implications for future advancements in robotic control systems.

The structure of this paper is as follows: Section II describes the development of the data-driven Model Predictive Control (MPC) system. It covers the construction of the data-driven dynamic model for the 7 DOF robotic manipulator, sets forth the control objectives for the MPC, and includes an analysis of the system's stability. Section III provides experimental results that demonstrate the proposed data-driven MPC's ability to control the robotic manipulator to accurately follow predefined joint values and trajectories while adhering to specific constraints. Finally, Section IV concludes the paper and suggests potential directions for future research.

II. DATA-DRIVEN MODEL PREDICTIVE CONTROL

A. BACKGROUND

Model Predictive Control (MPC) is a multivariable control technique that uses a mathematical or data-driven model to predict the future state of the system being controlled. It then calculates a series of optimal control inputs within defined constraints. MPC fundamentally consists of three main components: the predictive model, the target trajectory, and the controller, which optimizes the outcomes in a rolling manner. The structure of a closed-loop MPC system is depicted in Figure 1. In this figure, $\boldsymbol{x}_r$ represents the desired

state trajectory of the robot, $\boldsymbol{\tau}$ indicates the manipulated torque variables, $\boldsymbol{x}_p$ represents the robot's state predicted from the data-driven model, $\boldsymbol{x}$ denotes the controlled joint state, and $\bar{\boldsymbol{\tau}}$ refers to the sequence of optimized torques.

The model of the 7-DOF robotic manipulator can be described in discrete terms as follows:

$$\boldsymbol{x}(k+1) = f\left(\boldsymbol{x}(k), \boldsymbol{\tau}(k)\right) \tag{1}$$

where $\boldsymbol{x} = [\boldsymbol{q}, \dot{\boldsymbol{q}}]^T \in \mathbb{R}^{14}$ represents the joint state, encompassing both joint positions and velocities, $\boldsymbol{\tau} \in \mathbb{R}^7$ denotes the joint torques, and $f(\cdot)$ is the system's unknown dynamic function. Given the nonlinear nature of the system, accurately identifying the precise function $f(\cdot)$ that mirrors the robotic behavior is challenging. Additionally, using the nonlinear system dynamics as the MPC prediction model to anticipate the robot's future states based on the sequence of actuation can be computationally exhaustive, making real-time control of the robot difficult. Therefore, the primary objective of this approach is to accurately predict the system's behavior under control and derive optimal control actions. In this research, we employ a data-driven dynamics model as the prediction model in the proposed MPC strategy.

B. DATA-DRIVEN DYNAMIC MODEL

The primary goal of developing a Deep Neural Network (DNN) model is to create a surrogate model that can effectively serve as a predictive model within the proposed MPC strategy for regulating the joint state of the robotic manipulator. Specifically, this effort aims to approximate the joint state $\boldsymbol{x}(k+1)$ at the next time step $k+1$, based on the current joint state $\boldsymbol{x}(k)$ and the applied input torque $\boldsymbol{\tau}(k)$ at the current time step k. Thus, the DNN is trained to directly approximate the solution in Eq. (1),

$$f\left(\boldsymbol{x}(k), \boldsymbol{\tau}(k); \boldsymbol{\theta}\right) \approx f\left(\boldsymbol{x}(k), \boldsymbol{\tau}(k)\right) \tag{2}$$

where $\boldsymbol{\theta}$ represents the free parameters of the DNN. This enhances the predictive capability and control precision within robotic manipulator systems by using the DNN model as the MPC prediction model, facilitating real-time decision-making.

A feed-forward shallow neural network, illustrated in Figure 2, functions as the dynamic predictive model for the robotic manipulator. This network includes an input layer that takes in the current joint state $\boldsymbol{x}(k)$, comprising joint positions, velocities, and input torques $\boldsymbol{\tau}$. It has two hidden layers: the first with 128 neurons and the second with 32 neurons, both utilizing the rectified linear unit (RELU) activation function. The architecture is finalized with an output layer that uses a linear activation function to predict the subsequent joint state $\boldsymbol{x}(k+1)$. In the experiment section, we evaluated different network architectures to assess their impact on prediction performance. Future work could explore using Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks to handle time-series data in building the deep learning model.

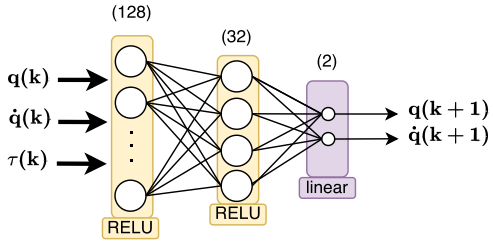

FIGURE 2. Structure of the DNN feed-forward neural network employed for predicting the joint state in the 7-DOF KUKA LBR4 robotic manipulator.

The dataset from [26] was utilized, consisting of ten trajectories generated during pick and place tasks executed by the KUKA LBR4 robot. The pick and place locations were randomly selected from two non-overlapping areas, each measuring 50 × 50 cm. The robots were considered to have successfully completed a task if they started and finished at the same location. The dataset includes approximately 18,000 samples, containing current joint positions, current joint velocities, applied torques, next joint positions, and next joint velocities. The data was split into 70% for training and 30% for validation.

In developing our deep learning-based MPC, it was essential to seamlessly integrate and efficiently execute the predictive model across varied computational requirements. To achieve this, we used the Open Neural Network Exchange (ONNX) framework [27] for model conversion and interoperability. The initial predictive model, designed and trained using the TensorFlow API, showed promising performance in preliminary experiments. Converting this model from TensorFlow to ONNX involved using the `tf2onnx` tool to transform the TensorFlow computational graph into an ONNX model file. This streamlined process generally involves specifying the input and output nodes of the model to maintain the integrity of its predictive capabilities after conversion.

C. CONTROL OBJECTIVE

The main objective of the proposed data-driven nonlinear MPC is to stabilize the robotic manipulator by ensuring it follows a predefined reference joint trajectory $\boldsymbol{x}_r = [q_{1r}, q_{2r}, \ldots, q_{7r}]^T$ in joint space. This goal includes accounting for physical constraints, such as joint and actuation limits, when determining the optimal control actions, specifically the applied joint torques. Therefore, the cost function $\boldsymbol{J}$ is formulated to evaluate both tracking performance and the effectiveness of control actions over a prediction horizon N, defined as:

$$\begin{aligned}\boldsymbol{J}(k) = &\sum_{j=1}^{N} \boldsymbol{e}_{(k+j)}^T \boldsymbol{W}_1 \boldsymbol{e}_{(k+j)} \\ &+ \sum_{j=1}^{N} \Delta\boldsymbol{\tau}_{(k+j-1)}^T \boldsymbol{W}_2 \Delta\boldsymbol{\tau}_{(k+j-1)}\end{aligned} \tag{3}$$

Here, $\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}_r$ denotes the tracking error, while $\Delta\boldsymbol{\tau}$ signifies the predicted change in control input. The matrices $\boldsymbol{W}_1 = w_1 I_7 \geq 0$ and $\boldsymbol{W}_2 = w_2 I_7 \geq 0$ are positive weighting matrices, which are assumed to stay constant throughout the prediction horizon N.

The optimal control problem minimizing Eq. (3) is constrained by the physical and actuator limits. The robot's joint values and torques are restricted within the lower and upper bounds $[\boldsymbol{q}_{min}, \boldsymbol{q}_{max}]$ and $[\boldsymbol{\tau}_{min}, \boldsymbol{\tau}_{max}]$, respectively, as determined from the movement dataset. Additionally, nonlinear constraints function $\boldsymbol{g}(.)$ related to the robot's state, actuation, or system parameters can be incorporated when searching for the optimal control action:

$$\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\tau}) \leq 0 \tag{4}$$

From the perspective of supervised learning, the task of determining the optimal control law can be viewed as a nonlinear mapping executed by a single-layer neural network [28]. As a result, Gradient Descent (GD) proves to be a suitable algorithm for this task. Therefore, the sequence of control laws, $\boldsymbol{\tau}(\boldsymbol{k})$, is updated as follows:

$$\boldsymbol{\tau}(k+1) = \boldsymbol{\tau}(k) + \Delta\boldsymbol{\tau}(k) \tag{5}$$

$$\Delta\boldsymbol{\tau}(k) = \eta \left(-\frac{\partial \boldsymbol{J}(k)}{\partial \boldsymbol{\tau}} \right) \tag{6}$$

where $\eta > 0$ denotes the learning rate for the control sequence. According to [29], the control increment $\Delta\boldsymbol{\tau}(k)$ is defined as:

$$\Delta\boldsymbol{\tau}(k) = \frac{\eta w_1}{1 + \eta w_2} \left(\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\tau}} \right)^T \boldsymbol{e} \tag{7}$$

In summary, Algorithm 1 delineates the procedure for two primary tasks: First, it details the construction of the surrogate (DNN) dynamic model for the 7-DOF robotic manipulator, as outlined from lines 1 to 5. Second, it describes the implementation of the data-driven model predictive control (MPC) for precise tracking of the reference joint positions, covered from lines 6 to 12.

D. LYAPUNOV STABILITY ANALYSIS

To illustrate the stability of the system, a quadratic Lyapunov function expressed in terms of the tracking error is chosen as follows:

$$V(\boldsymbol{e}) = \frac{1}{2} \boldsymbol{e}^T \boldsymbol{e} \tag{8}$$

where $\boldsymbol{e} = \boldsymbol{x}_r - \boldsymbol{x}$. To ensure the global asymptotic stability of the system, the first derivative of $V(\boldsymbol{e})$ with respect to time must be negative definite, which implies that $\boldsymbol{e}$ converges exponentially to zero.

The first time derivative of $V(\boldsymbol{e})$ is calculated as follows:

$$\dot{V}(\boldsymbol{e}) = \boldsymbol{e}^T \dot{\boldsymbol{e}} \tag{9}$$

Algorithm 1 Data-Driven MPC

Require:
- $(\boldsymbol{q}_k, \boldsymbol{\tau}_k, \boldsymbol{q}_{k+1})$, $\forall k = [0, M]$: DNN training data
- $\boldsymbol{q}_r$: Desired joint values (reference trajectory)
- $\boldsymbol{q}_0$, $\boldsymbol{\tau}_0$: Initial joints and torque
- U_q, U_τ: Bounds of joints and torques
- $\boldsymbol{g}(q, \tau)$: Nonlinear constrain function
- $\boldsymbol{W}_1$, $\boldsymbol{W}_2$: Weighting matrices for the cost function
- N: Prediction Horizon

Ensure:
- $\boldsymbol{\tau}_k^*$: Optimal sequence of control inputs (torques) over the prediction horizon

1: $\theta_k = \theta_0$ ▷ initialize DNN model with random weights
2: **For** $k = 0$ to $M - 1$:
3: $\hat{q}_{k+1} = f_\theta(q_k, \tau_k)$ ▷ Predict the next state
4: $L = \mathbf{MSE}(\hat{q}_{k+1}, q_{k+1})$ ▷ Compute the loss
5: $\theta^{k+1} \leftarrow \theta^k - \alpha \frac{\partial L}{\partial \theta}$ ▷ Update DNN parameters
6: $q_k = q_0$ ▷ initial robot's joints
7: **For** $k = 0$ to $M - 1$:
8: $e_k = ||q_{rk} - q_k||^2_{W_1}$ ▷ Compute the error
9: $\boldsymbol{J}_k = \sum_{j=1}^{N} e + ||\Delta\tau||^2_{W_2}$ ▷ Compute cost
10: $\bar{\boldsymbol{\tau}} \leftarrow \text{IPOPT}(\boldsymbol{J}_k, f_\theta, g, e, U_q, U_\tau)$ ▷ Solve the optimization problem
11: $\boldsymbol{\tau}_k^* \leftarrow \bar{\boldsymbol{\tau}}[0]$ ▷ Apply the first control input
12: $q_{k+1} \leftarrow f_\theta(q_k, \boldsymbol{\tau}_k^*)$ ▷ Get the next state

Substituting the value of $\dot{\boldsymbol{e}}$ into Eq. (9) yields:

$$\begin{aligned}
\dot{V}(\boldsymbol{e}) &= \boldsymbol{e}^T (\dot{\boldsymbol{x}}_r - \dot{\boldsymbol{x}}) \\
&= \boldsymbol{e}^T \left(\frac{\partial \boldsymbol{x}_r}{\partial \boldsymbol{\tau}} - \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\tau}} \right) \frac{\partial \boldsymbol{\tau}}{\partial t} \\
&= -\boldsymbol{e}^T \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\tau}} \right) \frac{\partial \boldsymbol{\tau}}{\partial t} \\
&\approxeq -\boldsymbol{e}^T \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\tau}} \right) \Delta\boldsymbol{\tau}(k)
\end{aligned} \tag{10}$$

By substituting $\Delta\boldsymbol{\tau}(k)$ in Eq. (7) into Eq. (10):

$$\begin{aligned}
\dot{V}(\boldsymbol{e}) &= -\boldsymbol{e}^T \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\tau}} \right) \left\{ \frac{\eta w_1}{1 + \eta w_2} \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\tau}} \right)^T \boldsymbol{e} \right\} \\
&= -\frac{\eta w_1}{1 + \eta w_2} \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\tau}} \right)^T \left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\tau}} \right) \boldsymbol{e}^T \boldsymbol{e}
\end{aligned} \tag{11}$$

Given that $\dot{V}(\boldsymbol{e}) < 0$, following the principles of Lyapunov stability theory, we can conclude that the proposed control strategy is stable.

III. RESULTS AND DISCUSSION

A. PERFORMANCE OF THE DNN PREDICTION MODEL

In our initial experiments, we investigated how well our DNN-based model perform over predicting the movements of the 7 DOF robotic manipulator, as part of the MPC system.

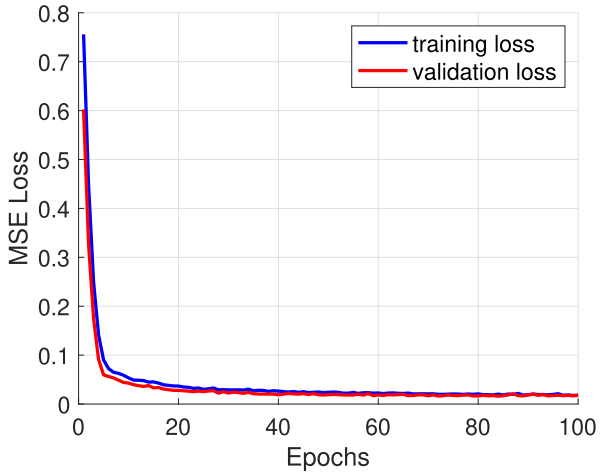

FIGURE 3. Training and validation loss of the DNN regression model over 100 epochs. The training loss decreases steadily, while the validation loss shows a similar downward trend, suggesting good generalization.

We conducted both training and prediction tasks with the TensorFlow 2.x API on a standard desktop computer. The choice of computer hardware, especially the CPU's clock speed of 2.8 GHz, significantly affected how long it took to train our model.

To evaluate the performance of the proposed DNN regression model, we monitored the training and validation loss over the course of 100 epochs. The training process aimed to minimize the Mean Squared Error (MSE) between the predicted and actual future joint states. Figure 3 shows the training and validation loss curves. The x-axis represents the number of epochs, and the y-axis represents the loss (MSE) value. The training loss (blue curve) and the validation loss (red curve) both exhibit a decreasing trend, indicating that the model is learning effectively and generalizing well to the validation data.

The training curve indicates that the model achieves a lower error rate as the number of epochs increases, with both the training and validation losses converging towards lower values. This convergence suggests that the model is not overfitting, as there is no significant divergence between the training and validation losses. Further, the model's performance on the validation set is critical for assessing its ability to generalize to unseen data. The consistent decrease in validation loss confirms that the DNN model is capable of effectively learning the underlying patterns in the robot's dynamics, thereby making accurate predictions on new data.

To assess the performance of our DNN model, we tested it on a dataset containing 1800 samples. Figure 4 illustrates the comparison between the model's predictions and the actual values. For brevity, we present results only for the first joint, displaying the predicted next joint positions and velocities. The close alignment between the predicted outputs and the actual data demonstrates the minimal discrepancy, highlighting the model's accuracy. However, the noise present in the velocity data, particularly during position transitions, can adversely impact the system behavior predictions and

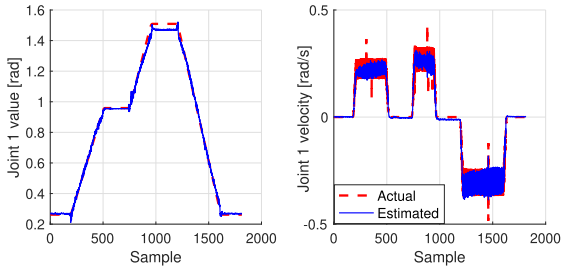

FIGURE 4. Comparison between actual next joint states $x(k+1)$ and estimated joint states $\hat{x}(k+1)$ using the proposed DNN-based prediction model, showing the first joint values on the left and the joint velocities on the right.

consequently the control performance. This indicates that pre-processing the dataset should be considered in the future before using it to build the DNN regression model.

B. POINT STABILIZATION PREDICTIVE CONTROL

In this study, we aim to evaluate the performance of the proposed data-driven MPC controller in point stabilization tasks for the KUKA LBR4 robotic manipulator. This MPC controller was developed using the `do-mpc` framework [30]. The goal is to command the robot to reach fixed targets in joint space $\boldsymbol{q}_r \in \mathbb{R}^7$, with these targets being updated every 500 samples. The experiment was conducted with a sampling time of $T = 0.05$ seconds and a prediction horizon of $N = 10$. For the optimization cost function, we used diagonal state and input weighting matrices, with $\boldsymbol{W}_1 = 100\boldsymbol{I}_7$ to emphasize the importance of accurately reaching the target joint positions and $\boldsymbol{W}_2 = 0.01\boldsymbol{I}_7$ to slightly penalize the input torques needed to achieve these positions.

This setup provides a rigorous test of the controller's ability to adapt to changing targets and maintain stability in the joint positions. The MPC controller is using the data-driven model to predict future states and optimize control inputs accordingly. The model was trained on a dataset encompassing a wide range of joint configurations and corresponding torques to ensure robust performance across various scenarios. To assess the controller's effectiveness, we used the tracking error performance metrics. Tracking error was measured as the difference between the desired and actual joint positions that is visualized through plots showing joint positions over time for multiple cycles of 500 samples each. These plots, illustrated in Figure 5, highlight the transitions between different targets and the controller's ability to follow the desired trajectories closely accompanied with the applied joints torques.

The results indicate that the data-driven MPC controller effectively tracks the desired joint trajectories with minimal error. The use of a data-driven model in the MPC framework proved beneficial, offering accurate predictions and efficient control. Meanwhile, the data-driven controller provides nonlinear control approach while maintain the real-time capability of the controller. Extending the approach to more dynamic goals could also provide valuable insights.

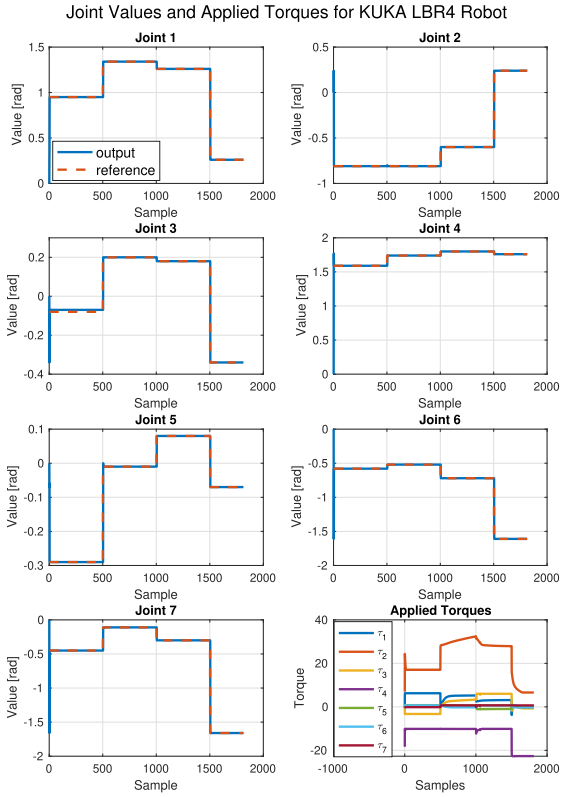

FIGURE 5. Results of point stabilization scenarios to evaluate the the proposed deep learning-based MPC for reaching predefined joints references.

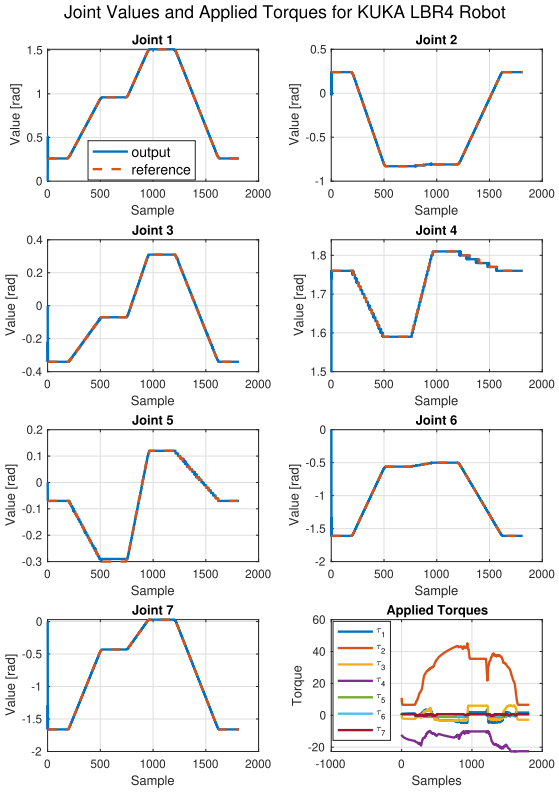

FIGURE 6. Results of trajectory tracking scenarios evaluating the proposed deep learning-based MPC in following reference joint trajectories.

C. TRAJECTORY TRACKING PREDICTIVE CONTROL

To assess the performance of the proposed data-driven MPC controller in trajectory tracking, we utilized a reference trajectory. This trajectory is particularly useful in scenarios where the robotic manipulator must follow a specific sequence of goals, such as in pick-and-place applications. Specifically, the reference trajectory, denoted as $\boldsymbol{q}_r \in \mathbb{R}^7$, consists of a series of desired joint values.

The robot starts with initial joint values $\boldsymbol{q}_0 = [0.26, 0.24, -0.34, 1.76, -0.07, -1.61, -1.666]^T$ (rad). The controller's time step is set to $t = 0.01$ seconds, with a prediction horizon of $N = 12$, and the total simulation time is 18 seconds. The weighting matrices are chosen as $\boldsymbol{W}_1 = 100$ and $\boldsymbol{W}_2 = 0.01$. The performance of the MPC is depicted in Figure 6, demonstrating the MPC's effectiveness in minimizing the difference between the measured and reference trajectories. The Mean Squared Error (MSE) between the reference trajectory and the actual robot trajectory is calculated to be 2×10^{-4} (in radians squared).

Furthermore, Figure 7 illustrates the performance of the data-driven MPC in tracking a reference joint and the corresponding applied torque, considering a nonlinear inequality constraint on the robot's joint $(q_1 - 1)^2 \leq 0$. This constraint acts as a limitation on the first joint within the joint space. While this constraint does not significantly impact the overall system, it has been included to test the algorithm's robustness. The results demonstrate that the proposed MPC achieves satisfactory tracking performance while adhering to this constraint and effectively managing constraints on the other joints.

D. COMPARISON WITH PID CONTROL

In this simulation experiment, we compare the performance of our data-driven MPC controller with a PID independent joint control approach. The PID control loops were applied to the robot model obtained using a deep learning model trained from data. The PID gains were determined through trial and error. To evaluate the controller, we used tracking error, measuring the difference between the desired and actual joint positions. Within this PID controller, the torque τ_i for each joint is determined individually in the following manner:

$$\tau_i = K_p\, e_i + K_d \frac{de_i}{dt} + K_i \int e_i\, dt \tag{12}$$

In contrast, the PID independent joint control approach yielded poor results. Despite the effort to tune the PID gains, the controller struggled to maintain stability. The tracking error was significantly higher than that of the MPC. More concerningly, the applied torques exhibited signs of instability, with significant amplifications when the reference joints changed.

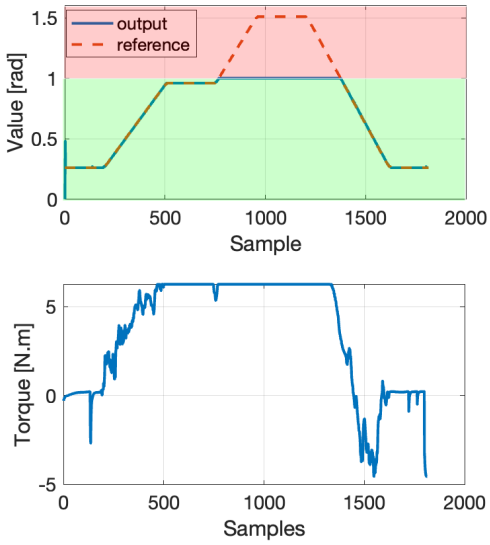

FIGURE 7. The results of the trajectory following experiment, highlighting the joint constraint on q_1 in the red area.

Figure 8 shows the tracking performance and applied torques for the PID controller. The plots reveal substantial deviations from the desired trajectories and unstable torque values, indicating that the PID control was not effective for this application.

The comparison highlights the advantages of the data-driven MPC approach over PID independent joint control for the KUKA LBR4 robotic manipulator. The MPC controller's ability to predict and optimize future states based on the data-driven model provided superior tracking accuracy and stability. In contrast, the PID controller, even with carefully tuned gains, could not achieve the same level of performance and exhibited instability in the applied torques.

The superior performance of the data-driven MPC can be attributed to its predictive capabilities and optimization framework, which allow it to handle the nonlinear dynamics of the robotic system more effectively. The PID controller's poor performance underscores the challenges of tuning PID gains for complex, nonlinear systems and highlights the limitations of relying solely on feedback control without predictive modeling.

E. DNN MODEL SELECTION

We conducted a thorough evaluation of the proposed DNN model for the robotic manipulator using the K-Fold Cross-Validation technique, as outlined by Anguita et al. [31]. This approach is crucial for rigorously assessing the model's predictive capability under various architectural designs, ensuring its effectiveness and reliability. The dataset was divided into five distinct subsets, enabling a cyclic process of training and evaluation. This comprehensive evaluation provides valuable insights into the model's performance across different conditions, helping to identify the most effective architectural design and demonstrating the model's ability to generalize to real-world scenarios.

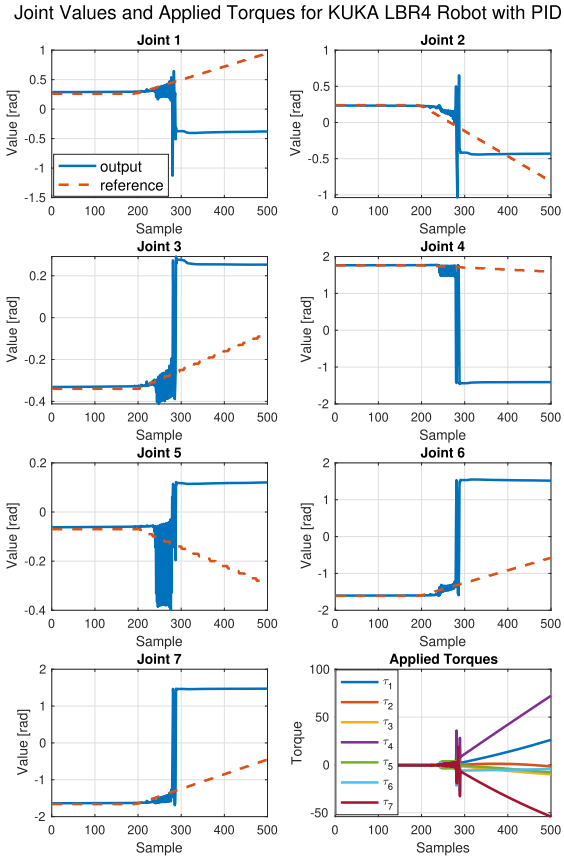

FIGURE 8. Performance of the PID tracking controller in following the reference joint trajectories.

In this study, we developed a set of five deep neural network (DNN) models, all utilizing feed-forward architectures. These models vary in complexity, with trainable parameters ranging from 594 to 47,694. Each model consists of three fully connected layers, differentiated by the number of hidden units, as detailed in Table 1. The layers are denoted with an 'F' and a subscript indicating the number of neurons in each. The primary activation function used is the hyperbolic tangent (tanh), except for the output layer, which employs a linear activation function.

These models were trained over 100 epochs using the Adam optimizer with mean squared error (MSE) as the primary loss metric. To ensure comprehensive evaluation and validation, a 5-fold cross-validation method was employed. The convergence trends of the learning algorithm, encompassing all network configurations and validation folds, are depicted in Figure 9-a. Additionally, Figure 9-b provides a detailed analysis of the average MSE losses and their standard deviations across different model architectures. Notably, networks with a higher parameter count are highlighted in dark red, indicating that larger networks tend to converge more effectively towards lower MSE values. To simplify the selection process, we chose a neural network that performed well during both training and testing phases. Exploring architectures that are both compact and capable of

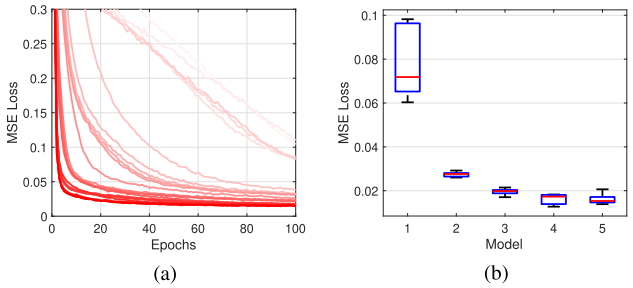

FIGURE 9. (a) Training losses expressed as mean squared error (MSE) for five deep neural network (DNN) models during 5-fold cross-validation, and (b) mean and standard deviations.

TABLE 1. Chosen architectures for the 5-fold cross-validation experiment.

Model	Hidden Layers	Units	Avg. MSE $\times 10^{-2}$
1	F_{16}, F_8, F_4	594	7.85 $\pm$1.54
2	F_{32}, F_{16}, F_8	1494	2.74 $\pm$0.10
3	F_{64}, F_{32}, F_{16}	4254	1.95 $\pm$0.14
4	F_{128}, F_{64}, F_{32}	13614	1.61 $\pm$0.22
5	F_{256}, F_{128}, F_{64}	47694	1.61 $\pm$0.23

rapid learning could potentially enhance the overall system's robustness.

IV. CONCLUSION

To address the challenges of constrained nonlinear joint control for a seven Degrees of Freedoms (DoF) robotic manipulator, this study introduces a new approach through the implementation of a data-driven-enhanced nonlinear Model Predictive Control (MPC) method. Initially, a data-driven predictive model was developed, capable of forecasting future joint positions based on current joint values and applied torques. This model was then integrated within an MPC framework, employing an online optimization problem with process constraints to determine the optimal control torques for accurate point stabilization and joint trajectory tracking of the KUKA LBR4 robotic manipulator.

In conclusion, the data-driven MPC controller exhibited strong performance in terms of tracking accuracy, handling of joint and actuators constraints, and effectively adapting to changing targets. The data-driven MPC outperformed the PID independent joint control in tracking reference joint trajectories for the KUKA LBR4 robotic manipulator. The MPC approach provided better tracking accuracy, consistent stabilization times, and stable control efforts, demonstrating its effectiveness and reliability for robotic control applications.

Furthermore, a K-fold cross-validation was conducted to select the optimal model architecture, ensuring robust performance and generalization across different data splits. A promising avenue for future research is the development of data-driven Moving Horizon Estimation (MHE) [32] along with the MPC, potentially mitigating the current assumption of full state observability that this work presupposes. Such advancements promise to further refine the precision and applicability of MPC in robotic manipulator control, contributing valuable insights into the integration of deep learning techniques within complex control systems.

REFERENCES

[1] B. Siciliano, O. Khatib, and T. Kröger, *Springer Handbook of Robotics*, vol. 200. Germany: Springer, 2008.
[2] S. Kalpakjian and S. R. Schmid, *Manufacturing Engineering and Technology*. London, U.K.: Prentice-Hall, 2009, pp. 568–571.
[3] R. H. Taylor and D. Stoianovici, "Medical robotics in computer-integrated surgery," *IEEE Trans. Robot. Autom.*, vol. 19, no. 5, pp. 765–781, Oct. 2003.
[4] D. Arney, R. Sutherland, J. Mulvaney, D. Steinkoenig, C. Stockdale, and M. Farley, "On-orbit servicing, assembly, and manufacturing (OSAM) state of play," Edition-NASA Tech. Rep. Server (NTRS), White Paper 20210022660, 2021. Accessed: May 1, 2023. [Online]. Available: https://ntrs.nasa.gov/api/citations/20210022660/downloads/osam_state_of_play%20(1).pdf
[5] C. C. Cheah, S. Kawamura, and S. Arimoto, "Feedback control for robotic manipulator with uncertain kinematics and dynamics," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 4, May 1998, pp. 3607–3612.
[6] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*. Germany: Springer, 2012.
[7] J. Son, H. Kang, and S. H. Kang, "A review on robust control of robot manipulators for future manufacturing," *Int. J. Precis. Eng. Manuf.*, vol. 24, no. 6, pp. 1083–1102, Jun. 2023.
[8] P. D. Nguyen, N. H. Nguyen, and H. T. Nguyen, "Adaptive control for manipulators with model uncertainty and input disturbance," *Int. J. Dyn. Control*, vol. 11, no. 5, pp. 2285–2294, Oct. 2023.
[9] L. Sciavicco and B. Siciliano, *Modeling and Control of Robot Manipulators*. Germany: Springer, 2012.
[10] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. Hoboken, NJ, USA: Wiley, 2006.
[11] W. Khalil and E. Dombre, *Modeling, Identification and Control of Robots*. London, U.K.: Butterworth, 2002.
[12] M. Ruderman, F. Hoffmann, and T. Bertram, "Modeling and identification of elastic robot joints with hysteresis and backlash," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 3840–3847, 2009.
[13] R. Fareh, S. Khadraoui, M. Y. Abdallah, M. Baziyad, and M. Bettayeb, "Active disturbance rejection control for robotic systems: A review," *Mechatronics*, vol. 80, 2021, Art. no. 102671.
[14] V. Bargsten, P. Zometa, and R. Findeisen, "Modeling, parameter identification and model-based control of a lightweight robotic manipulator," in *Proc. IEEE Int. Conf. Control Appl.*, 2013, pp. 134–139.
[15] T. W. Yang, W. L. Xu, and J. D. Han, "Dynamic compensation control of flexible macro–micro manipulator systems," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 1, pp. 143–151, 2009.
[16] F. Aghili, "Adaptive control of manipulators forming closed kinematic chain with inaccurate kinematic model," *IEEE/ASME Trans. Mechtron.*, vol. 18, no. 5, pp. 1544–1554, 2012.
[17] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-time neural MPC: Deep learning model predictive control for quadrotors and agile robotic platforms," *IEEE Robot. Autom. Lett.*, vol. 8, no. 4, pp. 2397–2404, Apr. 2023.
[18] H. El-Hussieny, I. A. Hameed, and A. A. Nada, "Deep CNN-based static modeling of soft robots utilizing absolute nodal coordinate formulation," *Biomimetics*, vol. 8, no. 8, p. 611, Dec. 2023.
[19] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *Proc. 6th IEEE-RAS Int. Conf. Humanoid Robots*, Dec. 2006, pp. 137–142.
[20] H. Li, R. J. Frei, and P. M. Wensing, "Model hierarchy predictive control of robotic systems," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3373–3380, Apr. 2021.
[21] G. García, R. Griffin, and J. Pratt, "MPC-based locomotion control of bipedal robots with line-feet contact using centroidal dynamics," in *Proc. IEEE-RAS 20th Int. Conf. Humanoid Robots (Humanoids)*, Jul. 2021, pp. 276–282.
[22] T. Ohtsuka and K. Ozaki, "Practical issues in nonlinear model predictive control: Real-time optimization and systematic tuning," in *Nonlinear Model Predictive Control: Towards New Challenging Applications*. Berlin, Germany: Springer, 2009, pp. 447–460.

[23] T. Akbas, S. E. Eskimez, S. Ozel, O. K. Adak, K. C. Fidan, and K. Erbatur, "Zero moment point based pace reference generation for quadruped robots via preview control," in *Proc. 12th IEEE Int. Workshop Adv. Motion Control (AMC)*, Mar. 2012, pp. 1–7.
[24] S. Kolathaya, "Local stability of PD controlled bipedal walking robots," *Automatica*, vol. 114, Apr. 2020, Art. no. 108841.
[25] M. Sombolestan, Y. Chen, and Q. Nguyen, "Adaptive force-based control for legged robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 7440–7447.
[26] A. S. Polydoros and L. Nalpantidis, "A reservoir computing approach for learning forward dynamics of industrial manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 612–618.
[27] J. Bai, F. Lu, and K. Zhang. (2019). *ONNX: Open Neural Network Exchange*. [Online]. Available: https://github.com/onnx/onnx
[28] H.-G. Han, X.-L. Wu, and J.-F. Qiao, "Real-time model predictive control using a self-organizing neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 9, pp. 1425–1436, Sep. 2013.
[29] G. Wang, Q.-S. Jia, J. Qiao, J. Bi, and M. Zhou, "Deep learning-based model predictive control for continuous stirred-tank reactor system," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3643–3652, Aug. 2021.
[30] F. Fiedler, B. Karg, L. Lüken, D. Brandner, M. Heinlein, F. Brabender, and S. Lucia, "do-mpc: Towards FAIR nonlinear and robust model predictive control," *Control Eng. Pract.*, vol. 140, Nov. 2023, Art. no. 105676.
[31] D. Anguita, A. Ghio, S. Ridella, and D. Sterpi, "K-fold cross validation for error rate estimate in support vector machines," in *Proc. DMIN*, 2009, pp. 291–297.
[32] H. El-Hussieny, I. A. Hameed, and A. B. Zaky, "Plant-inspired soft growing robots: A control approach using nonlinear model predictive techniques," *Appl. Sci.*, vol. 13, no. 4, p. 2601, Feb. 2023.

HAITHAM EL-HUSSIENY received the M.Sc. and Ph.D. degrees in mechatronics and robotics engineering from Egypt-Japan University of Science and Technology (E-JUST), Alexandria, Egypt, 2013 and 2016, respectively. He is currently an Associate Professor of robotics and artificial intelligence with E-JUST. His research interests include robotics, soft robotics, haptics, teleoperation, data-driven control, and applied intelligence.

IBRAHIM A. HAMEED (Senior Member, IEEE) received the Ph.D. degree in industrial systems and information engineering from Korea University, Seoul, South Korea, and the Ph.D. degree in mechanical engineering from Aarhus University, Aarhus, Denmark. He is currently a Professor with the Department of ICT and Natural Sciences, Faculty of Information Technology and Electrical Engineering, Norwegian University of Science and Technology (NTNU), Norway. He is the Deputy Head of research and innovation within the Department of ICT and Natural Sciences, Faculty of Information Technology and Electrical Engineering, NTNU. His current research interests include artificial intelligence, machine learning, optimization, and robotics. He is elected as the Chair of the IEEE Computational Intelligence Society (CIS) Norway Section.

TAMER F. MEGAHED received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from Mansoura University, Mansoura, Egypt, in 2006, 2010, and 2015, respectively. Currently, he is an Associate Professor and the Chairperson of the Electrical Engineering Department, Egypt-Japan University of Science and Technology (E-JUST). Also, he has four patents in magnetic refrigeration, designed the IoT systems for monitoring electricity consumption, thrust-vector-control model rocket design, and magnetic gear. His research interests include power control, renewable energy sources, smart grid, electric vehicles, wireless charging, energy storage control and management, vector control, power system protection, model predictive control, and green hydrogen.

AHMED FARES received the Ph.D. degree in computer science and engineering from Egypt-Japan University of Science and Technology (E-JUST), Alexandria, Egypt, in May 2015. He is currently an Associate Professor with the Department of Computer Science and Engineering, E-JUST. He is also an Associate Professor with Shoubra Faculty of Engineering, Benha University (on-leave). His research interests include multimedia content analysis, cognitive science, psychological and brain science, and machine learning.