Joint Optimization of Temporal Windows and Hidden Neurons for Residual-Based Fault Detection*

Eduardo Jose Apaza Alvarez[1] and Hiroshi Kawakami[2]

***Abstract*— This paper proposes an enhanced residual-based fault detection framework for a 6-DOF manipulator modeled via URDF and CAD parameters. We address sensor fault anomalies such as noise, spike and bias that affect joint measurements. Residuals are processed by a neural network-based classifier where a training-stage optimization algorithm automatically determines the optimal temporal window and hidden neurons to ensure maximin reliability. Unlike heuristic approaches, the strategy accounts for residual propagation in downstream joints through a specialized decision mechanism. Validation across fault induced scenarios demonstrate a very low False Alarm Rate and a consistent detection performance, outperforming fixed-architecture baselines in robustness and diagnostic stability. The results highlight the effectiveness of the proposed joint-parameter optimization for dependable sensor fault detection and diagnosis in complex robotic chains.**

I. INTRODUCTION

Industrial robotic manipulators are widely deployed in modern manufacturing environments due to their precision, repeatability, and ability to perform multiple task such as assembly, welding, material handling, picking, and inspection [1][2]. As production lines become increasingly automated and flexible, these systems are strongly required to operate continuously under many different operating conditions [3]. However, frequent sensor faults caused by degradation, external disturbances, or harsh industrial environments [1] can significantly affect system performance and reliability. When a fault happens, the resulting downtime and maintenance procedures may lead to considerable production losses, as the recovery of the robot commonly requires fault diagnosis, system reconfiguration, and manual intervention. Consequently, reducing fault recovery time has become a key objective in industrial robotics.

In the last ten years, significant research efforts have focused on developing advanced fault detection methodologies to improve the reliability and availability of robotic manipulators [4][5]. Among these approaches, data-driven and learning-based techniques have gained increasing attention due to their ability to handle nonlinear dynamics and complex full patterns [5][6] in many structural and mechanical monitoring applications [7]. Specially, neural-network based fault detection methods combined with residual analysis have shown promising results in identifying irregular system behavior without requiring explicit fault models [8]. By learning multiple features from residual signals, these methods can detect and classify different fault types, enabling faster diagnosis and reducing maintenance time [9]. This motivates the development of enhanced residual-based fault detection frameworks that improve detection accuracy while maintaining practical applicability in industrial robotic systems.

In order to address the complexities of faults and anomalies in robotic manipulators, recent research has focused on classifying faults and anomalies into sensor, actuator and structural faults. Zhang et al. in [1] suggest that the increasing sophistication of industrial manipulators requires a transition from traditional manual inspection to automated data-driven diagnostic frameworks. These modern approaches, which includes digital twins and advanced machine learning models, are essential to identify subtle degradations in harsh environments where mathematical models often struggle to maintain accuracy due to unforeseen disturbances and aging components.

A first strategy to achieve real-time monitoring is the use of analytical redundancy for residual generation. Paviglianiti, et al. in [8] demonstrate that the integration of state observers with neural networks, such as Radial Basis Functions (RBF), allows for the effective compensation of non-linear dynamics, leading to more robust fault detection and isolation. However, the performance of these models is highly sensitive to the chosen model structure. In this context Andelic, et al. in [9] suggest that the optimization of neural architectures, including Convolutional Neural Networks (CNN) and Siamese Neural Networks (SNN), is crucial for maximizing classification accuracy while minimizing computational latency. This evolution in fault detection and isolation models highlights a significant research gap: the necessity of specialized and optimized architectures that are capable of processing complex residual signals in 6-DOF manipulators.

Therefore, in this paper, we propose an enhanced residual-based fault detection framework for a simulated 6-degree-of-freedom (6-DOF) robotic manipulator under sensor and fault conditions. The proposed approach extends existing neural-based residual analysis methods under sensor signal anomalies conditions. This approach extends the existing neural-based residual analysis methods by incorporating the detection of bias fault label in addition to noise and spikes anomalies. Furthermore, a training-stage optimization strategy is introduced to select the most accurate temporal windows and neural network amount of neurons, addressing the sensitivity of learning-based methods to heuristic parameter

*This work was not supported by any organization

[1]Eduardo Jose Apaza Alvarez is with Graduate School of Engineering, Kyoto Univesity of Advanced Science, Kyoto, Japan 2024mm21@kuas.ac.jp

[2]Hiroshi Kawakami is with the Graduate School of Engineering, Kyoto University of Advanced Science, Kyoto, Japan kawakami.hiroshi@kuas.ac.jp

selection. In order to improve robustness against fault propagation effects, a fault decision mechanism is also developed to account for externally induced residuals that affect downstream joints. The effectiveness of the proposed framework is demonstrated through extensive simulation studies, showing improved fault detection accuracy and reliability compared to conventional fixed-window and fixed-neuron approaches.

II. Methodology

The proposed fault detection model is structured into two main operational stages, an offline training/optimization and an online detection stage. The integration of these stages ensures that the neural architecture is tailored to the specific dynamics of the 6-DOF manipulator.

A. Fault Detection Model Overview

Recent analysis in [10] shows that deep neural architectures have demonstrated superior adaptability and robustness in complex Fault Detection Models environments, compared to traditional rule-based systems. Therefore, Fig. 1 shows the overall process where the robot is controlled by torque τ_n signals provided by the PD-Control that has as input the desired position signals dq_n.

The first stage is focused on training and optimization; here, the neural network is trained using a grid-search algorithm that automatically selects the optimal number of neurons and the length of the sliding window for output processing. This ensures maximum sensitivity to faults while minimizing false positives or negatives.

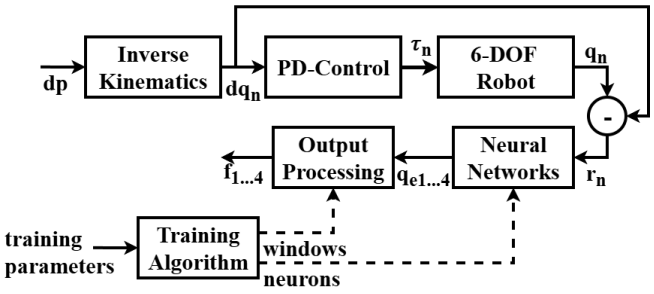

Fig. 1. Fault detection training and testing overview

On the other hand, the second stage comprises the robot controller and fault detection loop. The model begins with the desired end-effector position, which is processed by an inverse kinematics solver that use the desired position dp to generate six desired joint trajectories $dq_n, dq\dot{n}$. These serve as inputs for a PD-Computed Torque Controller, which calculates the required torques τ_n for the robot model. The resulting sensor positions q_n are then compared against dq_n to generate residual signals r_n. These are fed into the optimized neural network, producing four outputs $q_{e1,\ldots,4}$ that are further refined through a sliding window and a selection filter to determine the occurrence and specific type of fault $f_{1,2,3,4}$ corresponding to normal, noise, spike and bias labels, respectively. This last part of the stage is designed to enhance the main frequency characteristics beneficial for classification while suppressing low-amplitude noise components. Pang et al. states in [12] that incorporating adaptive windowing and multi-scale feature extraction into convolutional architectures significantly improves anti-noise performance in complex mechanical diagnostic tasks.

B. System Modeling and Residual Generation

As it was mentioned in the Introduction, the robotic manipulator is modeled as a 6-DOF system with all of the joints are rotational. As shown in Fig. 2, this manipulator is based on the PITSCO® modular system. To ensure high fidelity in simulation, a detailed digital model was developed. Rather than rely on theoretical values, the mass, center of gravity and inertia tensor for each link were extracted from a high-precision CAD model. These physical properties were integrated into a Unified Robot Description Format (URDF) file, which serves as the foundation for dynamic simulation.

Motion generation is achieved using inverse kinematics to produce trapezoidal trajectories, mapping the desired end-effector position and orientation into six reference joint angles. These position references, together with the corresponding velocity and acceleration profiles, are provided as inputs to the control system to ensure accurate motion tracking and stable operation.

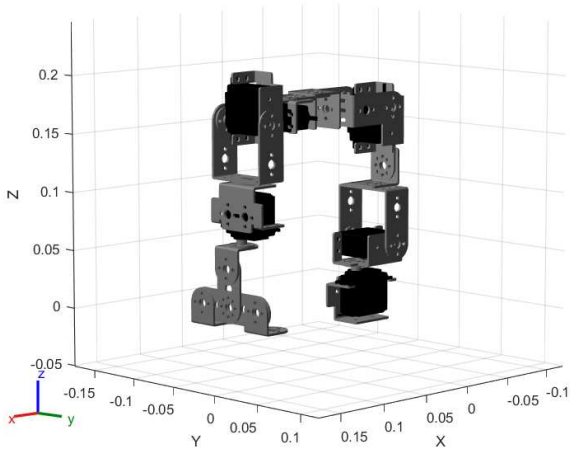

Fig. 2. Modeled 6-DOF manipulator robot applied in this research

To achieve accurate trajectory tracking, a proportional-derivative (PD) computed torque controller is implemented. Unlike decentralized schemes, this controller operates in a multi-variable (matrix) framework, simultaneously processing the position $\mathbf{q}$, velocity $\dot{\mathbf{q}}$, and torque τ vectors for all six joints. The controller compensates for the robot's dynamics effects, including inertia $M(\mathbf{q})$, Coriolis and centrifugal forces $C(\mathbf{q}, \dot{\mathbf{q}})\mathbf{q}$, and gravitational forces $G(\mathbf{q})$, as shown in (1) and (2).

The measured joint positions $\mathbf{q}$ are compared with the reference trajectories $\mathbf{q}_d$ to generate the residual vector $\mathbf{r}_q$, defined as the absolute difference between them, as show in (3). These residuals capture deviations from nominal operation; under a robust control law like the PD-computed

torque, any significant deviation will be categorized as fault rather than just steady-state tracking site error [11].

$$\tau = M(\mathbf{q})(\ddot{\mathbf{q}}_d + K_d\dot{\mathbf{e}} + K_p\mathbf{e}) + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + G(\mathbf{q}) \quad (1)$$

$$\mathbf{e} = \mathbf{q}_d - \mathbf{q} \quad (2)$$

$$r_n = |q_n - dq_n| \quad (3)$$

C. Sensor Fault Modeling

In this research, three types of faults are induced into the manipulator: noise, spikes and bias. These faults directly affect the position of each joint, compromising the accuracy and reliability of the measurements used by the previous explained controller.

Noise faults happen when random anomalies are added to the position signal delivered by the sensor or encoder. This fault is typically caused by electronic interference or hardware degradation. This leads to fluctuations around the true joint position, which may reduce the stability and precision of the controller [13]. Spike faults are characterized by unexpected and short duration deviations in the sensor output. These abrupt anomalies often arise from communication errors or transient failures in the sensing hardware, and they can severely disrupt the estimation process if not detected promptly [14]. Finally, bias faults are characterized by a persistent offset in the sensor measurements, causing the reported joint position to systematically deviate from the true value. These faults typically arise from sensor miscalibration, thermal drift, or long-term hardware degradation, and they introduce a constant or slowly varying error that affects the accuracy of the measured joint positions [15]. All these fault scenarios (Fig. 3) are injected at the sensor level, directly affecting joint position measurements and generating distinct residual patterns that are later exploited for fault detection and diagnosis.

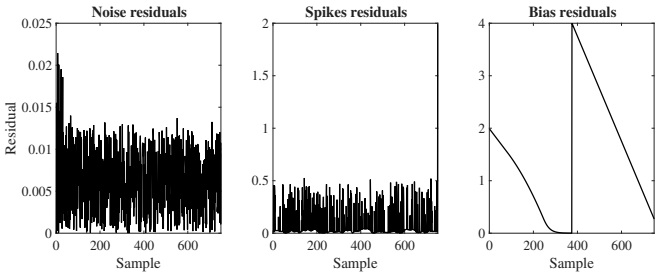

Fig. 3. Residual joint position signals under noise, spike, and bias faults.

D. Neural Network Architecture and Dataset Characterization

The fault detection model is based on a feedforward neural network that process r_n generated from the robot system. The network is made of a single hidden layer, where the number of neurons is treated as treated as a design parameter and determined during the training phase, as described in the following subsection. As is shown in Figure 4, r_q is provided as input to the network, and the output layer employs a softmax activation function to produce a four-dimensional output vector that correspond to the system states: normal operation q_{e1}, noise q_{e2}, spike q_{e3}, and bias faults q_{e4}. Each output represents the normalized likelihood of the system belonging to a specific class.

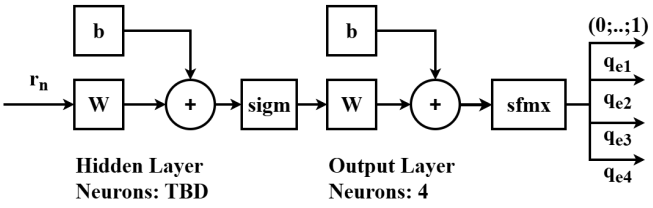

Fig. 4. Supervised learning fault detection architecture

To train this architecture, a balanced dataset of 9000 samples was generated, with 2250 samples per system state. The training phase utilized the Levenberg-Marquardt backpropagation algorithm with a maximum limit of 2000 epochs and a validation stop criterion of 10 iterations to ensure generalization. The performance is validated trough a dynamic test scenario consisting of a 6-second robotic trajectory following performance. During this simulation faults are injected sequentially every 1.5 seconds. Data is captured with a high-resolution of a period of $T_s = 0.002s$.

To enhance robustness against fluctuations and improve classification stability and accuracy, the softmax outputs are post-processed using a moving mean average filter as shown in Fig. 5. The window length of this averaging operation is also selected during the training stage to maximize classification performance. The averaged outputs are the passed to a decision module that assigns the final system state by selecting the label $f_{1,2,3,4}$ with the maximum probability.

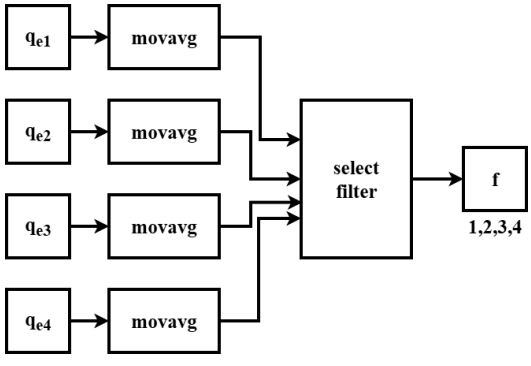

Fig. 5. Post processing estimated signal architecture

E. Training-Stage Optimization of Temporal Windows and Neurons

The proposed research includes a training-stage optimization procedure shown in Fig. 6 to jointly determine the number of neurons n in the hidden layer and the length

of the temporal averaging window applied to the neural network outputs. Using the dataset described previously, the optimization starts with an initial number of hidden neurons of 20 increasing 20 by 20 and iteratively trains the single-layer feedforward neural network explained before using fixed training parameters (2000 epochs and a threshold of 10). For each network configuration, the trained model is evaluated on a validation dataset to compute the classification accuracy.

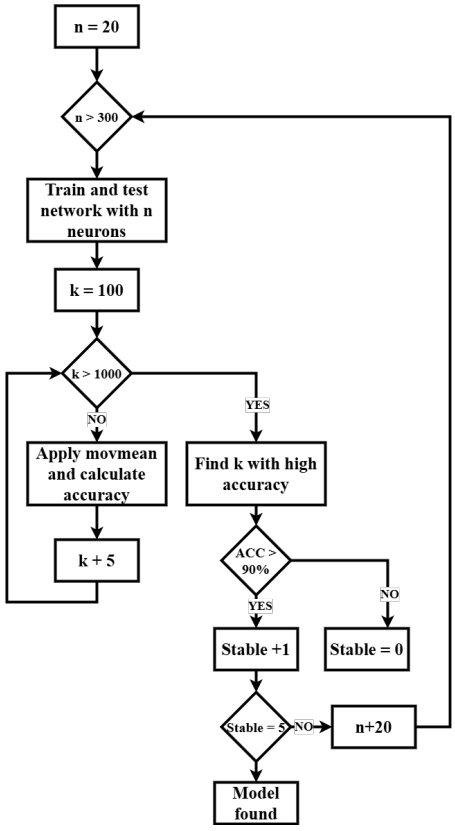

Fig. 6. Training-state optimization procedure

For a given neural architecture, a moving mean average filter is applied to the network softmax outputs using increasing windows lengths k from 100 to 1000 increasing five by five the amount of windows. The window size that yields the highest validation accuracy is selected, and the corresponding performance is recorded. If the achieved accuracy exceeds a predefined threshold Acc_{th}, the configuration is considered stable, this early-stop criteria minimize computational overhead and avoid redundant iterations with overly complex architectures.

The optimization proceeds by incrementally increasing the number of hidden neurons and repeating the process until a predefined number of five consecutive stable configurations is reached. The parameters settled for the optimization algorithm is shown in Table I.

Acc_{th} was set at 90% because it serves as a performance benchmark during the grid-search process; only configurations that meet or exceed this accuracy on the validation subset are considered candidates for the final architecture.

TABLE I
OPTIMIZATION ALGORITHM PARAMETERS

Parameter	Value/Range
n	$n \in [20, 300]$
k	$k \in [100, 1000]$
Acc_{th}	90%

This ensures that the final network architecture and temporal window length are selected based on the highest observed accuracy, ensuring a robust balance between model complexity and fault detection performance.

F. Fault Decision Strategy with Residual Propagation

When a fault is induced in a given joint, for example q_1, its effect is not limited to that joint alone, but propagates to subsequent joints due to the dynamic coupling inherent to serial robotic manipulators. Although the magnitude of the propagated affect is typically lower in downstream joints, certain fault types, particularly spike and bias faults, can generate residual deviations large enough to be incorrectly interpreted as additional independent faults. In order to prevent false positives and to reflect the physical reality that a single primary fault can influence multiple joints, a fault decision strategy based on residual propagation is adopted. Under this strategy, residual anomalies detect in downstream joints following the identification of a primary spike or bias fault are reclassified as fault-related residuals rather than separate faults. This approach allows the system to maintain a coherent fault interpretation at the system level and ensures that, when a recovery or fault-tolerant control mechanism is integrated, both the primary fault and its propagated effects can be addressed simultaneously.

III. Simulation Results

A. Simulation Setup

As previously described, the simulated system corresponds to a six-degree-of-freedom (6-DOF) serial robotic manipulator with exclusively joints. The generated trajectories shown in Fig. 7 emulate an industrial pick-and-place operation, in which the robot joints execute coordinated motions to grasp and object and transport it to a different location within the workspace. Each simulation run has a total duration of six seconds. During the first 1.5 seconds, the system operates under nominal conditions, followed by successive fault intervals of 1.5 seconds corresponding noise, spike, and bias sensor faults respectively. Faults are injected into the position measurement of a single joint per simulation, and no combination of multiple fault types is considered within the same run. Table I shows the optimal parameters of window length and neurons for each fault detection model. This configuration allows a clear and isolated evaluation of the fault detection performance for each fault category.

B. Simulation Performance Under Different Sensor Faults

Fig. 8 shows the residual signals obtained when a fault is induced at joint 3. As is observed, the impact on the residuals of joints 1 and 2 is minimal, remaining within a range that

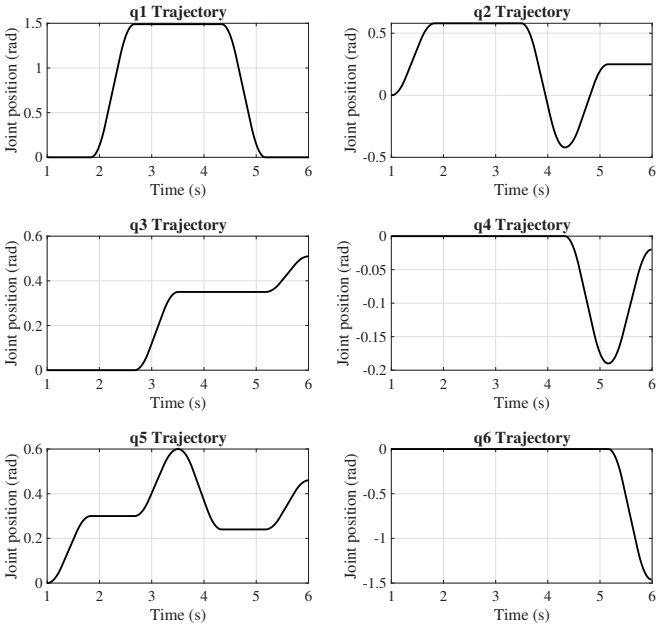

Fig. 7. Joint performance trajectories

TABLE II
OPTIMIZED WINDOWS AND NEURONS

Joint	Neurons	Windows
q_1	20	270
q_2	260	300
q_3	20	200
q_4	20	300
q_5	60	300
q_6	200	250

does not trigger external fault detection. This behavior is consistent with the serial structure of the manipulator, where upstream joints are weakly affected by faults occurring in downstream joints. On the other hand, a clear propagation effect is observed in the subsequent joints, specially in joints 4, 5, and 6, where residual amplitudes increase significantly and reach values close to 0.1 rad. These propagated residuals demonstrate how faults originating at an intermediate joint can influence the dynamic behavior of downstream joints, especially in the presence of spike and bias faults.

The corresponding fault detection and classification result when the fault affect Joint 3 are shown in Fig. 9. To better appreciate the classifier's dynamic response, the figure focuses on the affected joint and the subsequent residual propagation in downstream joints, showing the framework ability to distinguish between primary and propagated disturbances.

The proposed neural-based detection system successfully identifies and classifies all fault occurrences induced at joint 3. A short temporal offset between the desired fault label (blue) and the estimated output (ted) can be observed, which is attributed to the moving-average post-processing stage and the selected temporal window length. Nevertheless, the optimization algorithm selects the combination of temporal windows and neural network neurons that minimizes this delay while preserving classification accuracy. For down-

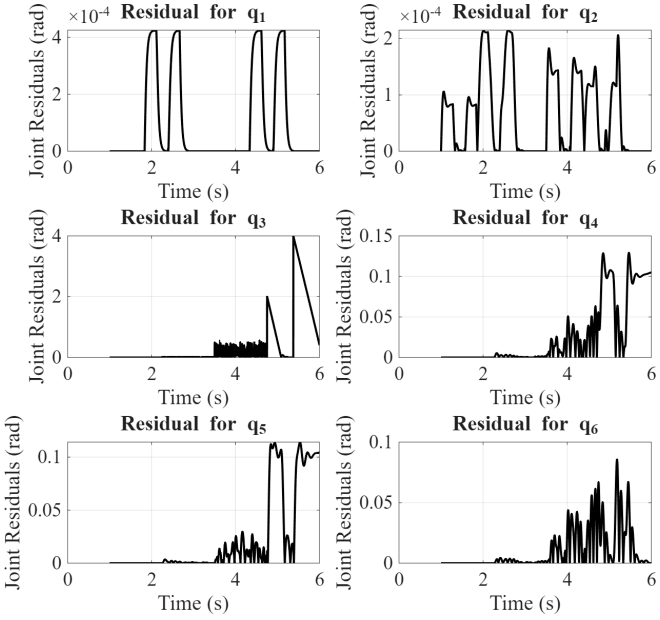

Fig. 8. Residual signals when fault is applied to joint 3

stream joints, propagated faults (Particularly spikes and bias) are consistently detected as external faults, confirming the effectiveness of the proposed fault decision strategy.

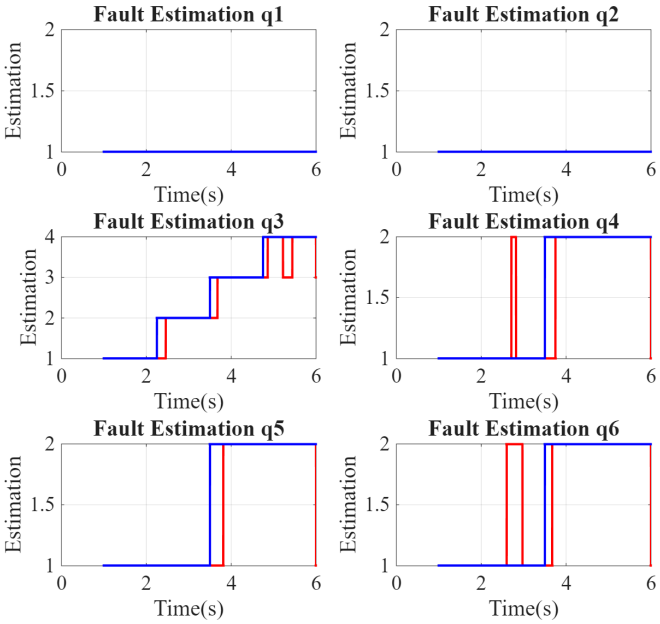

Fig. 9. Fault detection model applied in joint 3

The performance of the fault detection system in all six joints is summarized in Table III, where the system achieves a balanced average accuracy of 85.93% across all joints. Crucially, the False Alarm Rate (FAR) was 0.0% in all scenarios, demonstrating high reliability for industrial application where false stops must be avoided.

Regarding temporal performance, the average detection delay was 0.2024s. While some joints (e.g., q_3) show faster response times due to the optimized temporal window k, the

TABLE III

FAULT DETECTION ACCURACY VALUES

Joint Position	Accuracy	FAR	Delay
q_1	84.80%	0%	0.1806
q_2	85.83%	0%	0.2334
q_3	85.37%	0%	0.1473
q_4	86.43%	0%	0.2200
q_5	85.03%	0%	0.2345
q_6	86.10%	0%	0.1984
Total Avg	85.93%	0%	0.2024

overall delay remains within acceptable limits for real-time monitoring of slow-to-medium speed in industrial tasks.

Finally, it is important to note that while the optimization threshold was set at 90%, the final average accuracy observed in Table III shows a small deviation. This is attributed to the transition periods between the sequential faults injected every 1.5 seconds in the continuous 6-second simulation, which represent more challenging conditions that the static training samples. Moreover, this result demonstrates that the proposed model provides reliable fault detection performance using an entirely neural-based framework, even under fault propagation conditions.

C. Impact of Window and Neuron Optimization

Table IV presents a comparative analysis between the proposed optimization-based model and two fixed-architecture baselines: a single-layer network with 80 neurons and a fixed window (SL-FW), and a double-layer network with 60 and 30 neurons using the same fixed window size (DL-FW). Although the fixed-architecture models achieve higher accuracy values for certain joints, their performance exhibits strong variability across the manipulator, with significant accuracy drops in specific joints, most notably for q_6, where accuracy falls below 61%. In contrast, the proposed method delivers a more uniform performance across all joints, maintaining accuracy values consistently around $85-86\%$. This behavior highlights that the optimization of temporal window length and neuron configuration prioritizes robustness and stability rather than peak accuracy at isolated joints, leading to more reliable fault detection performance under varying fault propagation conditions.

TABLE IV

ACCURACY COMPARISON VALUES

Joint Position	Proposed	SL-FW	DL-FW
q_1	84.80%	94.23%	89.90%
q_2	85.83%	89.61%	91.63%
q_3	85.37%	78.07%	84.64%
q_4	86.43%	79.86%	83.80%
q_5	85.03%	88.41%	94.84%
q_6	86.10%	60.84%	59.56%

IV. CONCLUSIONS

This paper proposed an enhanced residual-based fault detection framework for a 6-DOF robotic manipulator, specifically designed to identify sensor faults such as noise, spikes, and bias. By implementing a training-stage optimization algorithm to select temporal windows and hidden layer neurons, the proposed model achieved a consistent detection accuracy approximately 85% across all joints. Furthermore, the integration of a fault decision strategy effectively addressed the challenge of residual propagation in downstream joints, ensuring that externally induced anomalies were correctly reclassified as related to the primary fault rather than independent occurrences.

Unlike traditional fixed-architecture configurations (SL-FW and DL-FW), which exhibited significant performance drops in specific joints—most notably in q_6 where accuracy fell below 61%, the optimized framework provided more uniform and robust performance throughout the entire system. These results highlight that prioritizing the stability of temporal windows and neural configurations over peak isolated accuracy leads to more reliable diagnostic behavior. Consequently, this approach offers a practical and effective solution for improving the reliability and availability of industrial robotic manipulators in complex, modern manufacturing environments.

REFERENCES

[1] Y. Zhang, et al., "Fault Types and Diagnostic Methods of Manipulator Robots: A Review," in *Sensors*, vol. 25, no. 6, pp. 176, 2025.

[2] M. Anand, T. Selvaraj, S. Kumanan. "Fault detection and fault tolerance methods for industrial robot manipulator based on hybrid intelligent approach," in *Advances in Production Engineering & Management*, vol. 7, no. 4, pp. 225–236, 2012.

[3] A. Milecki, P. Nowak. "Review of fault-tolerant control systems used in robotic manipulators," in *Applied Sciences*, vol. 13, no. 4, pp. 2675, 2023

[4] M. Quamar, A. Nasir. "Review on fault diagnosis and fault-tolerant control scheme for robotic manipulators: Recent advances in AI, machine learning, and digital twin," in *arXiv preprint arXiv:2402.02980,* 2024.

[5] Khan, F., et al. "Enhancing robotic manipulator fault detection with advanced machine learning techniques," in *Engineering Research Express*, vol. 6, no. 2, pp. 025204, 2024.

[6] Eski, I., et al. "Fault detection on robot manipulators using artificial neural networks," in *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 1, pp. 115–123, 2011.

[7] H. Dabaja, H. Noura, M. Ouladsine, "Data-driven Crack Detection in the Realm of Structural Health Monitoring: An Overview," in *11th International Conference on Control, Decision and Information Technologies (CoDIT 2025),* 2025.

[8] Paviglianiti, G., et al., " Robust fault detection and isolation for proprioceptive sensors of robot manipulators," in *Mechatronics*, vol. 20, no. 1, pp. 162-170, 2010.

[9] Andelic, N., et al. "Neural Network-Based Model for Classification of Faults During Operation of a Robotic Manipulator," in *Tehnički vjesnik*, vol. 28, no. 4, pp. 1380–1387, 2021.

[10] Paolini, D., et al. "Advanced Fault Detection and Diagnosis Exploiting Machine Learning and Artificial Intelligence for Engineering Applications," in *Electronics*, vol. 15, no. 2, pp. 476, 2026.

[11] Hu, R., et al. "Current-residual-based stator interturn fault detection in permanent magnet machines," in IEEE Transactions on Industrial Electronics, vol. 68, no. 1, pp. 59–69, 2020.

[12] Pang, P., et al. "Research on Fault Diagnosis and Feature Extraction Mechanism of Rotating Machinery Based on WGS–CNN," in *IEEE Sensors Journal*, 2025.

[13] R. Algburi, H. Gao, Z. Al-Huda. "Improvement of an industrial robotic flaw detection system," in *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 3953–3967, 2022.

[14] Dev Anand, M., et al. "Fault diagnosis system for a robot manipulator through neuro fuzzy approach," in *International Journal of Modelling, Identification and Control*, vol. 3, no. 2, pp. 181–192, 2008.

[15] Q. Yang, *Model-based and data driven fault diagnosis methods with applications to process monitoring.* Case Western Reserve University, 2004.