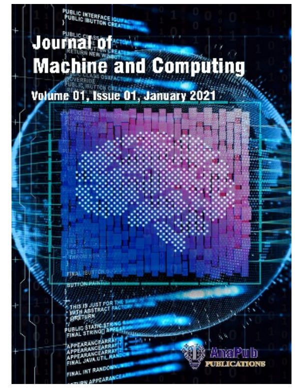
Journal Pre-proof



Automated Manufacturing Robot Fault Diagnosis in Real Time Using Convolutional Neural Networks

Hussein Ali Mahdi, K. Akilandeswari, Mayura Shelke, Sureshkumar Chandrasekaran, Vijaya Bhaskar Sadu, U. Sudha Rani

DOI: 10.53759/7669/jmc202404053

Reference: JMC202404053

Journal: Journal of Machine and Computing.

Received 23 August 2023; Revised form 20 March 2024; Accepted 22 May 2024.

Please cite this article as:

Hussein Ali Mahdi, K. Akilandeswari, Mayura Shelke, Sureshkumar Chandrasekaran, Vijaya Bhaskar Sadu, U. Sudha Rani, “Automated Manufacturing Robot Fault Diagnosis in Real Time Using Convolutional Neural Networks”, Journal of Machine and Computing. (2024). Doi: https:// doi.org/10.53759/7669/ jmc202404053

Automated Manufacturing Robot Fault Diagnosis in Real Time Using Convolutional Neural Networks

Hussein Ali Mahdi[1], K. Akilandeswari[2,*], Mayura Shelke[3], Sureshkumar Chandrasekaran[4], Vijaya Bhaskar Sadu[5], U. Sudha Rani[6]

[1]College of Nursing, University of Kerbala, Kerbala, 56001, Iraq.
Email: hussein.mahdi@uokerbala.edu.iq

[2]Department of Computer Science, Government Arts College (Autonomous), Salem, 636007, Tamil Nadu, India. **Corresponding Author Email: akila.gacslm7@gmail.com*

[3]Department of AI& DS, AISSMS IOIT, Pune, India. *Email: mayura.shelke@gmail.com*

[4]Department of Artificial Intelligence and Data Science, KCG College of Technology, Chennai, Tamil Nadu, 600097, India. *Email: csureshkumarmca@gmail.com*

[5]Department of Mechanical Engineering, Jawaharlal Nehru Technological University, Kakinada, Andhra Pradesh, India, 533003. *Email: sadhu.vijay@gmail.com*

[6]Department of Computer Science Engineering, Koneru Lakshmaiah Education Foundation, Vijayawada, Andhra Pradesh, 520002, India. *Email: rani.sudha47@gmail.com*

Abstract: This study introduced a novel real-time Fault Diagnosis Model (FDM) in manufacturing robots by integrating Depthwise Convolutional Neural Networks (CNNs) with Bidirectional Long Short-Term Memory (BiLSTM) networks. The objective is to design a model that can handle the complex high-dimensional sensor data that arrives out of complex, non-linear systems for effective FDM. The work introduced a Feature Extraction (FE) model based on Monte Carlo Filtering (MCF). The work integrates a Depthwise CNN with BiLSTM (DC-BiLSTM) for diagnosis. The integration helps to reduce the computational need and, at the same time, preserve the feature representation. The model was experimented against other models, such as CNN, Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), and *Feed-Forward Neural Networks* (FFNN), using a fault dataset sourced from a simulated environment. The results have shown that the proposed model fared well in terms of accuracy, precision, recall, and F1 score against all compared models. The results have judged the proposed model's applicability in the field of fault diagnosis, which could effectively predict mishaps in advance, thereby helping with efficient maintenance and ensuring continuous productivity.

Keywords: Real-Time Fault Diagnosis, Monte Carlo Filtering, Feature Extraction, CNN, BiLSTM, Accuracy

1. Introduction

Manufacturing robots have recently been playing a crucial role in various modern industrial operations because of their advantage in offering unparalleled precision, productivity, and efficiency. Their integration into the process of manufacturing production lines has shown a significant and enhanced output capability, and in this way, they have paved the method for further advancements in automation. The extent to which they show better performance to the same extent their level of complexity in their functioning and working environments together makes them prone to faults that ultimately result in disrupted operations. So, the proper fault diagnosis mechanisms are crucial for effective preemptive maintenance by ensuring continuous operation, thereby minimizing costly unplanned downtime. However, designing an effective fault diagnosis model has some critical challenges, mainly due to the nonlinear and non-Gaussian nature of the systems within which they operate. The current available traditional Fault Diagnostic Model (FDM) are all models that often fall short of efficiency when they are faced with such complexity. These methods often fail to cope with the high-dimensional data that are generated by an array of sensors embedded in modern robots.

In response to these limitations and to address the challenges recently, researchers and engineers have been involved in the process that explored several approaches to improve Fault Diagnosis (FD). The study is mainly performed to develop a more efficient use of the large amount of sensor data in order to find the anomalies that suggest the starting point of an issue at an earlier phase. Acknowledging that these attempts have been developed, there is still scope for development. In addition, there continually exists a demand for an approach that is adept at rapidly adapting to the unique functioning patterns of individual robots and the inherent subtle nuances that are FD. The challenges and limitations that have been highlighted are the motivation underpinning the recommended research. In recent years, Convolutional Neural Networks+ Recurrent Neural Network (CNN+RNN)-based models have been used in this domain. The objective of this investigation is to present an original framework to take advantage of the features of both methods. Additionally, the significance of handling high-dimensional data is of the highest priority, which is what prompts this research to look for methods that are feasible in order to handle such data sets.

The proposed work is built on the above motivation, which proposes a model for real-time FMD in manufacturing robots. The work proposed Depthwise convolutional BiLSTM (DC-

BiLSTM), which is a combination of proposed Depthwise Convolutional Neural Networks together with Bidirectional Long Short-Term Memory (BiLSTM) networks. To handle the high dimensionality issue from the sensor collected data, the work proposed Feature Extraction (FE) based on the Monte Carlo Filtering (MCF) technique, which was chosen for its effectiveness in feature extraction from complex, non-linear systems. The proposed DC-BiLSTM model employs standard convolution as the initial layers for spatial FE, followed by Depthwise separable blocks and BiLSTM layers. The proposed model was examined using a fault dataset and compared against other models, such as CNN, Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNN), and Feed-Forward Neural Networks (FFNN), for its efficiency in terms of accuracy, precision, recall, and F1 score. For all metrics, the proposed model had shown better performance.

The paper is structured as follows: Section 2 presents the literature review, Section 3 presents the methodology, Section 4 presents the analysis of experiments, and Section 5 concludes the work

2. Literature Review

[1] In their work, they have conducted a comprehensive review of the field of FD and the ML models used in this field. They surveyed several databases, selecting 44 primary studies highlighting the application of Artificial Neural Networks (ANN), Decision Trees (DT), hybrid models, and latent variable models in FD and prognosis. They revealed that despite the ML model's high performance and computational efficiency, the effectiveness of such models is often challenged by concept drift, clearly suggesting a gap in model adaptability to changing conditions.

The propagation level checking that is connected with multi-joint robots for industry was the main objective of the research they conducted [2]. Researchers succeeded in achieving this by introducing an original FDM that was based on behaviour data. The strategy that the researchers employed included using a hybrid Sparse Auto-Encoder (SAE) and Support Vector Machine (SVM) technique to evaluate the differences in performance that caused by the robot's end element link. Their finding that researchers managed to attain superior outcomes in accurately identifying faults proved the possibility of using specific data features for the intent of FD within sophisticated automated frameworks.

With the support of its decisions, the business was able to meet the essential requirement for real-time anomaly identification throughout the framework of manufacturing automation. The following is performed in order to avoid substantial economic losses that result from bottlenecks in firm operations. For the aim of maintenance based on condition, they developed an unsupervised detection of anomalies technique that had the power to manage heterogeneous time series data. Therefore, the invention of autonomous monitoring systems that have the ability to detect anomalies in both space and time has been significantly impacted by the outcome of the research conducted here, which currently has a readily evident result.

[4] performed studies to find out whether or not it would be feasible to integrate the Digital-Twin (DT) technique with Deep Learning (DL) in order to accomplish the objective of FD. As the outcome of their studies, the authors suggested an FDM that is suitable with DT as well as a strategy involving the use of an autonomous tool-holder that has the capacity to collect data from the Internet of Things (IoT). Throughout the use of experimental research, the researchers were able to prove the versatility of their approach and argue that it was useful in improving the automated operation of machinery tasks as well as the precision of FD.

The team of scientists who performed this research introduced a new FMD that had a chance to enhance the accuracy of FD. This FMD had been developed through the integration of Binarized Deep Neural Networks (BDNNs) with significantly enhanced Random Forests (RFs). Taking into account the implementation of BDNNs for the task of FE and the adaptation of the RF training method with ReliefF, researchers were capable of effectively try finding the faults by using those methods. By performing comprehensive tests as such, the research team showed their framework exhibited accuracy in diagnosis that was higher than that of Deep Neural Networks (DNNs) that have been shown to be state-of-the-art [6-10].

3. Methodology

3.1 MCF Algorithm for Feature Extraction

In settings during which there does not exist the potential for accurate tracking, the primary goal of MCF, which is additionally referred to as particle filtering, is to provide a precise forecast of the present state of a given system. The Bayesian theory acts as the basis against which the MCF is developed. The practical use of this approach can be achieved when it is used for non-linear frameworks, which tend to be challenging to manage using traditional approaches. In order to precisely forecast the subsequent distribution of the system's state, the process makes use of several types of data points, which are frequently referred to as particles, in addition to the measured weights that correspond with the particles. Additionally, when it comes to fresh data, the above approach allows an updating of the state by a continuous procedure [11-12].

The MCF is divided into its vital basics, which are drawn below:

- **State Representation:** The system's state at the time $'t'$ is denoted by $'x_t'$. Here, $'x_t'$ represent operational parameters that define the robot's current state.
- **Observation Model:** The observation at time 't' is represented by 'z_t', which corresponds to the data measured or recorded by sensors. The observation model relates the current state $'x_t'$ to the observation $'z_t'$, expressed as $'p(z_t \mid x_t)'$, where $'p'$ denotes the probability of observing $'z_t'$ given the state x_t.
- **State Transition Model:** This model describes how the state evolves from $'x_{t-1}'$ to 'x_t'. Which is expressed as $'p(x_t \mid x_{t-1})'$, referring to the probability of transitioning to a state 'x_t' from state $'x_{t-1}'$.
- **Particles and Weights:** The filter represents the probability distribution of the system's state using a set of particles $\left\{x_t^{(i)}\right\}_{i=1}^{N}$, where each particle $x_t^{(i)}$ is a state of the system at time $'t'$, and 'N' is the total number of particles. Each particle is associated with a weight $w_t^{(i)}$ that represents the importance or likelihood of that particle given the observed data up to time $'t'$.

The FE algorithm 1 using the above-discussed MCF fundamentals is presented in algorithm 1. The MCF technique processes the input sequence of observations $Z = \{z_1, z_2, \dots, z_T\}$,. The algorithm initiates a set of particles, with each particle symbolizing a potential state of the system, $\left\{x_0^{(i)}\right\}_{i=1}^{N}$. These initial states are distributed randomly across the expected range of states, particularly with each particle assigned with an equal initial weight, $w_0^{(i)} = \frac{1}{N}$. The algorithm then updates the state of each particle using the state transition function. $x_t = f(x_{t-1})$. Next to state prediction, the algorithm then adjusts the weights of each particle that are based on the observed data $'z_t'$ and the predicted state, utilizing the observation probability function $(z_t \mid x_t)$. The algorithm then normalizes the weights to ensure their sum equals 1. Following weight normalization [13-15], the algorithm performs a resampling step by selecting particles based on their normalized weights for the next iteration. In the FE phase, the algorithm calculates the mean μ_t and standard deviation $'\sigma_t'$ of the states from the resampled particles. These are then assembled into a feature vector $f_t = [\mu_t, \sigma_t]$ For each time step, provide a summary of the system's state.

Algorithm 1: FE Using MCF

Input:

- $Z = \{z_1, z_2, \dots, z_T\}$: sensor data up to time T.
- N : Number of particles.
- f : State transition function, $x_t = f(x_{t-1})$,
- g : Observation likelihood function, $p(z_t \mid x_t)$,

Output: $F = \{f_1, f_2, \dots, f_T\}$: Sequence of FE from the observations up to time T.

Algorithm:

1 **Initialization:**

- For $i = 1$ to N :
  - Initialize particles $x_0^{(i)}$ randomly based on a uniform distribution over the expected range of states.
  - Initialize particle weights $w_0^{(i)} = \frac{1}{N}$.

2 **For Each** time step $t = 1$ to T :

- **Prediction:**
  - **For** $i = 1$ to N :
    - Predict the next state of particle i using the state transition function EQU (1)

$$x_t^{(i)} = f\left(x_{t-1}^{(i)}\right). \quad (1)$$

- **Update:**
  - **For** $i = 1$ to N :
    - Update the weight of particle i based on the observation probability, EQU (2).

$$w_t^{(i)} = p\left(z_t \mid x_t^{(i)}\right). \quad (2)$$

  - **Normalize the weights, EQU (3)**

$$\hat{w}_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(i)}}. \quad (3)$$

  - **Resampling:** Resample N particles from the set $\left\{x_t^{(i)}\right\}$
- **FE:** Compute the mean μ_t and standard deviation σ_t of the resampled particles' states, EQU (4) and EQU (5).

$$\mu_t = \frac{1}{N} \sum_{i=1}^{N} x_t'^{(i)} \quad (4)$$

$$\sigma_t = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_t^{\prime(i)} - \mu_t\right)^2} \quad (5)$$

- **Define** the feature vector for time t as $f_t = [\mu_t, \sigma_t]$.

3 **Output** the sequence of FE 'F''.

3.2 Depth-Wise Separable Convolution

The depthwise separable convolutions reduce the computational cost and the number of parameters while retaining the capacity for detailed feature representation [16-20]. The depthwise separable convolution block, as shown in Figure 1, comprises two key steps:

1 **Depthwise Convolution:**

- **Input:** A high-dimensional input tensor from sensor data, with dimensions $D_f \times D_f \times M$, where D_f is the spatial dimension, and M is the number of channels.
- **Convolution:** A depthwise convolution involves M filters of size $k \times k \times 1$ that are convolved with each input channel separately, yielding an output of size $D_g \times D_g \times M$.

This step is employed to independently extract spatial features from each channel to reduce the number of computations.

2 **1 × 1 Convolution:**

- **Convolution:** Following the depthwise convolution, a pointwise convolution with N filters of size $1 \times 1 \times M$ combines the outputs of the depthwise convolution across the channels.
- **Final Output:** The resultant tensor has dimensions $D_g \times D_g \times N$, which captures the spatial features and combines them, thereby forming new features that would effectively represent channel-wise correlations.

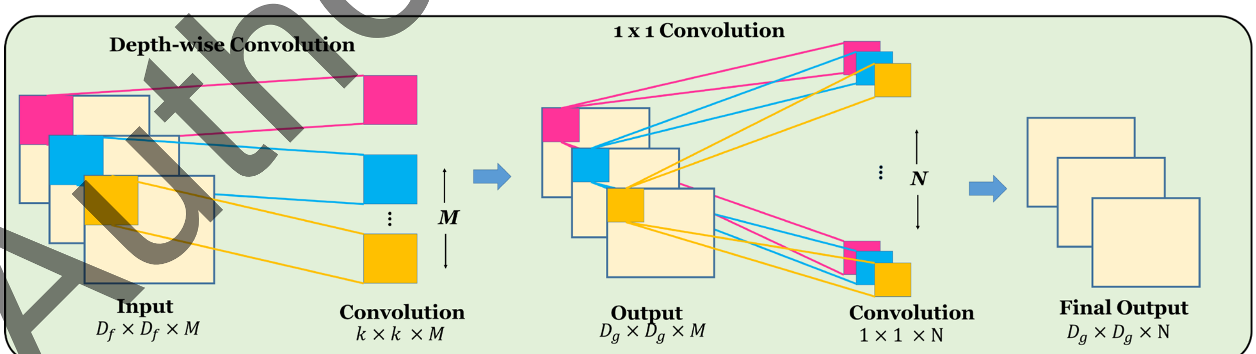


Figure 1: Depth-wise separable convolution.

3.3 Proposed DC-BiLSTM Model

The proposed DC-BiLSTM model integrates depthwise separable convolutional blocks with a BiLSTM network. The model inputs the features into a standard convolutional block configured with 16 filters of size 3×3 to identify the spatial features from the input data. Next, batch normalization is applied to stabilize the learning process and improve the model's efficiency, followed by the Rectified Linear Unit (ReLU) activation function for normalization and max pooling for feature dimension reduction. Then, two depthwise convolutional blocks are placed one after the other, each with 32, 3×3 filters. After spatial FE, the model employs a BiLSTM layer with 32 units to analyze the temporal relationships inherent in the sequential data. The BiLSTM layer is followed by a dropout layer with a 0.5 dropout rate before feeding to a Fully Connected (FC) layer with 75 units. Finally, the model employs a SoftMax layer for the output. The architecture of the proposed model is presented in Figure 2, and the algorithm 2 presents the process flow.

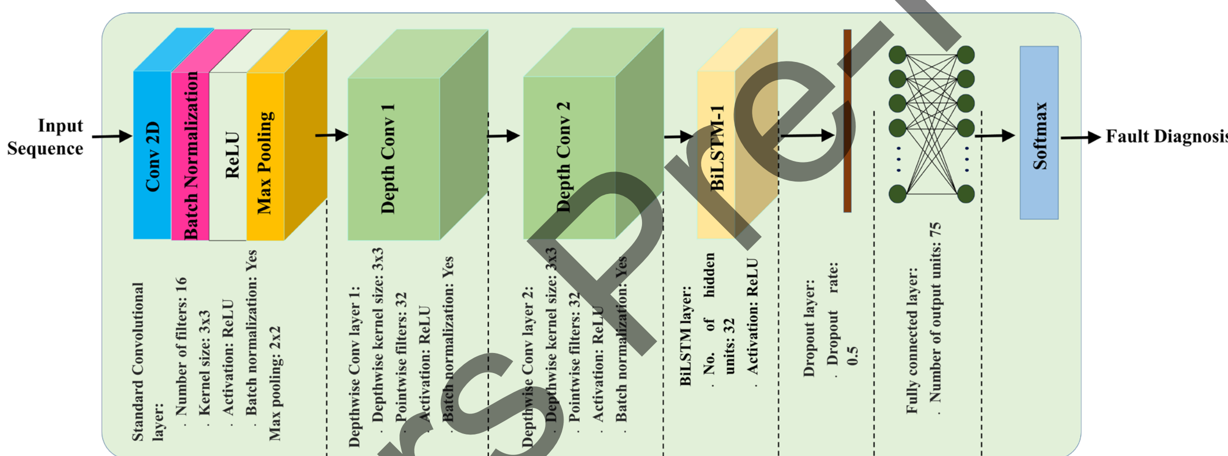


Figure 2: DC-BiLSTM architecture

Algorithm 2: DC-BiLSTM

Inputs:

- X : The input features extracted from the MCF process.
- Y : The true labels for the training data.

Outputs: $\hat{Y}$: The predicted labels for the input data.

Procedure:

1 **Initialization:**

- Initialize the weights and biases for all layers in the Depthwise CNN+BiLSTM network.
- Preprocess X as needed (*e.g.,* normalization, reshaping).

2 **Standard Convolutional Block:**

- Conduct 2D convolution on X with 16 filters of size 3×3.
- Apply batch normalization to the convolution output.
- Use the ReLU activation function.
- Employ max pooling with a 2×2 kernel to reduce spatial dimensions.

3 **DC Block 1:**

- Perform DC using 323×3 filters.
- Apply 1×1 convolution to the output of the DC.
- Execute batch normalization followed by ReLU activation.

4 **Depthwise Convolutional Block 2:**

- Apply DC 1×1 convolution as in Block 1.
- Follow with batch normalization and ReLU activation.

5 **BiLSTM Layer:**

- Input the output from the last depthwise block into the BiLSTM layer with 32 units.
- Allow the BiLSTM layer to process the temporal information.

6 **Dropout Layer:** Apply a dropout operation with a rate of 0.5 to prevent overfitting.

7 **FC Layer:**

- Pass the BiLSTM output into a fully connected layer with 75 units.
- Apply ReLU activation.

8 **SoftMax Output Layer:**

- Apply the SoftMax function to derive the probability distribution over fault classes.
- Compute the loss between Y and $\hat{Y}$ we are using cross-entropy loss.
- Update the weights by backpropagation using the Adam optimization algorithm.

9 **Training Loop:** Iterate steps 2 to 9 for a predetermined number of epochs or until convergence.

10 **Fault Diagnosis (Inference Phase):**

- Feed X into the trained model.
- Perform a forward pass through the model to obtain the predicted labels $\hat{Y}$.

End Procedure

3.5 Fault Diagnosis Using Proposed MCF FE and Depthwise CNN + BiLSTM (DC-BiLSTM)

The FDM begins with data preprocessing that includes filtering, normalization, and segmentation, which is followed by FE using MCF. These features are *tahn* labeled before the input into the Neural Network (NN). The DC-BiLSTM model then takes the labelled data for processing and effectively FD. The entire process is described in the following algorithm 3 steps:

Algorithm 3: FD Using DC-BiLSTM

Step 1: Data Collection and Preprocessing:

- **Collect Monitoring Data:** Gather sensor data from manufacturing that include a variety of fault conditions.
- **Preprocessing Steps:**
  - **Filtering:** Apply FCF techniques to remove noise and irrelevant fluctuations from the sensor data.
  - **Normalization:** Scale the sensor readings to a standard.
  - **Segmentation:** Divide the continuous stream of sensor data into fixed-size segments.

Step 2: FE with MCF:

- **Apply MCF:** Use MCF on the preprocessed sensor data to isolate features that are most indicative of operational states or potential faults.
- **Feature Preparation:** Organize the FE into a structured format for input into the Deep Learning (DL) model.

Step 3: Data Cleaning and Labeling:

- **Data Cleaning:** Examine the feature set for any inconsistencies or null values. Remove these instances to maintain the integrity of the training process.
- **Labelling:** Annotate each data segment with appropriate fault labels.

Step 4: CNN + BiLSTM Model Architecture and Training:

- **Build Model Architecture:** Construct the DC-BiLSTM model starting with:
  - A standard convolutional block for initial spatial feature extraction.
  - Two consecutive depthwise separable convolutional blocks.
  - A BiLSTM layer to capture and analyze temporal dependencies within the sequences of spatial features.
  - A fully connected layer and a SoftMax output layer for final fault classification.
- **Model Training:**

- Split the labelled feature set into training and validation subsets.
- Train the model on the training set, optimizing the network's weights and biases to minimize a predefined loss function through several epochs. Employ the Adam optimization algorithm for efficient learning.
- After each epoch, validate the model's performance on the validation subset to monitor accuracy and prevent overfitting.

Step 5: Evaluation and Deployment for Real-Time Diagnosis:

- **Evaluate Model Performance:** Test the trained model on a separate, hidden test set to assess its accuracy, precision, recall, and ability to FD.
- **Real-Time FD:** Deploy the trained model within the manufacturing robot's operational environment. Implement a real-time data collection, preprocessing, and FE system, feeding this data into the model for instantaneous FD and decision-making.

End of Algorithm

4. Experimental Setup

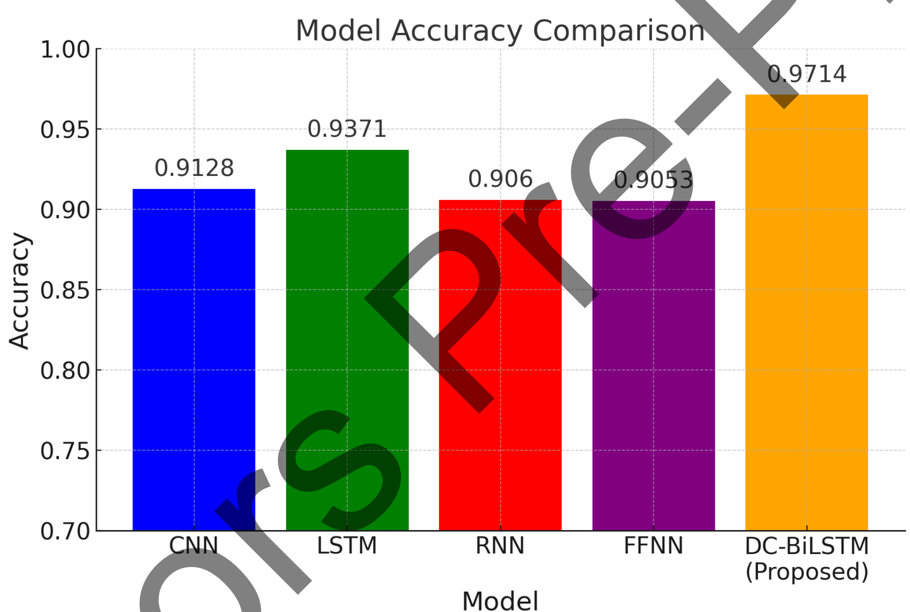
In the experimental setup, data was collected from a manufacturing robot using a FDM. A non-functioning actuator replaced a functioning one to simulate fault conditions. Data collection targeted the robot's 3rd and 5th axes. The 3rd axis provided 23,456 data points during normal operation and 21,234 in fault conditions. The 5th axis yielded 188,532 data points with faults and 46,789 data points during normal operation. The collection rate was 20 *Hz*. After collection, FE was conducted on this data to prepare for FD analysis. The dataset was split into training sets and testing with a (70:30) ratio. The proposed model was compared against CNN, RNN, LSTM, and FFNN models in metrics like i) Accuracy, ii) Precision, iii) Recall, and iv) F1. The proposed model was trained using the following parameters as shown in Table 1:

Table 1: Training parameters

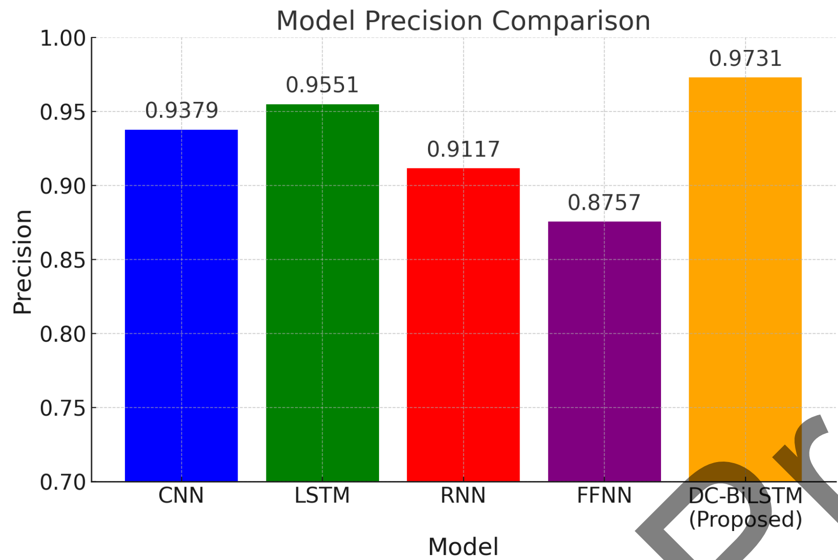
Parameter	Value
Learning Rate	0.001
Batch Size	64
Number of Epochs	100
Dropout Rate	0.5
Optimizer	Adam
Loss Function	Cross-Entropy
Normalization Technique	Batch Normalization

Activation Function	ReLU

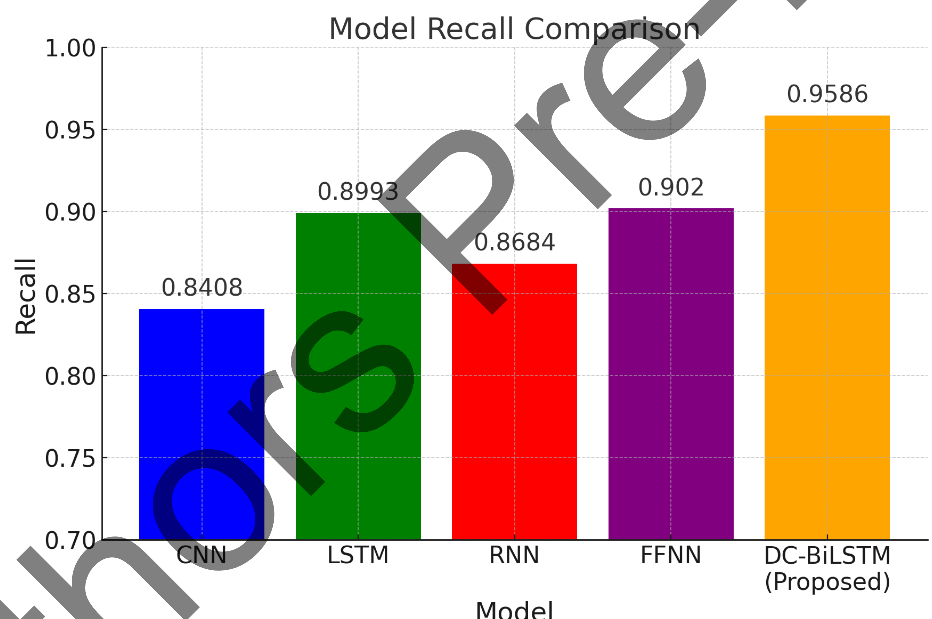
In evaluating the performance of various NN models for FD in manufacturing robots, the DC-BiLSTM model distinctly outperforms its counterparts across all key metrics: accuracy, precision, recall, and F1 score (Figure 3 (a)-(d)). The DC-BiLSTM achieves an accuracy of 0.9714, demonstrating its capability in correctly classifying normal and faulty conditions that are significantly higher than LSTM (0.9371), CNN (0.9128), RNN (0.9060), and FFNN (0.9053). For the Precision metric, the DC-BiLSTM model had again shown a leading score of 0.9731, which is higher when compared to the LSTM's precision of 0.9551, CNN's 0.9379, RNN's 0.9117, and FFNN's 0.8757, this had shown that the DC-BiLSTM model is accurate in fault identification at the same time with minimal error.



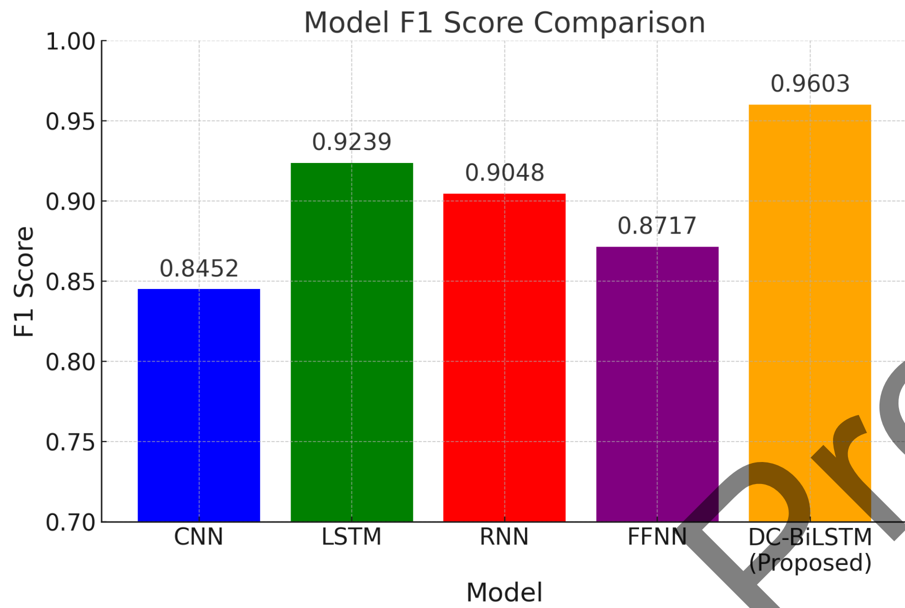
(a)


Model Precision Comparison
1.00
0.95
0.90
0.85
0.80
0.75
0.70
Precision
0.9379
0.9551
0.9117
0.8757
0.9731
CNN
LSTM
RNN
FFNN
DC-BiLSTM
(Proposed)
Model

(b)


Model Recall Comparison
1.00
0.95
0.90
0.85
0.80
0.75
0.70
Recall
0.8408
0.8993
0.8684
0.902
0.9586
CNN
LSTM
RNN
FFNN
DC-BiLSTM
(Proposed)
Model

(c)



(d)

Figure 3: a) Accuracy, b) Precision, c) Recall and F1 Score comparison

In terms of recall, the DC-BiLSTM model shows the highest at 0.9586, which again surpasses the LSTM (0.8993), CNN (0.8408), RNN (0.8684), and FFNN (0.9020). This result demonstrates that the DC-BiLSTM does not miss any instance in the prediction process compared to other models. For the F1 score that balances the precision and recall, the DC-BiLSTM model leads the compared models with a score of 0.9603 against LSTM (0.9239), CNN (0.8452), RNN (0.9048), and FFNN (0.8717), this shows that the DC-BiLSTM's model has a balanced and reliable FD capability.

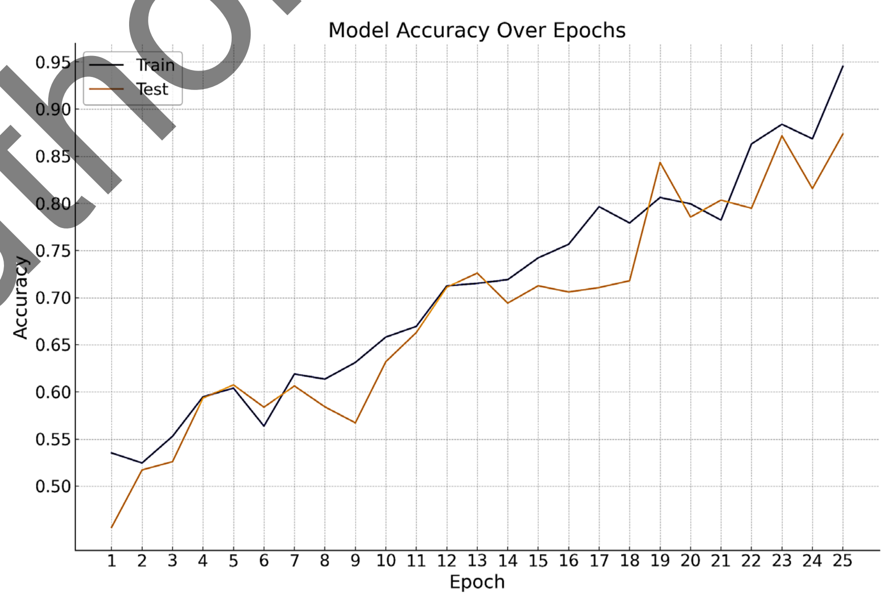


Figure 4: Accuracy vs Epochs

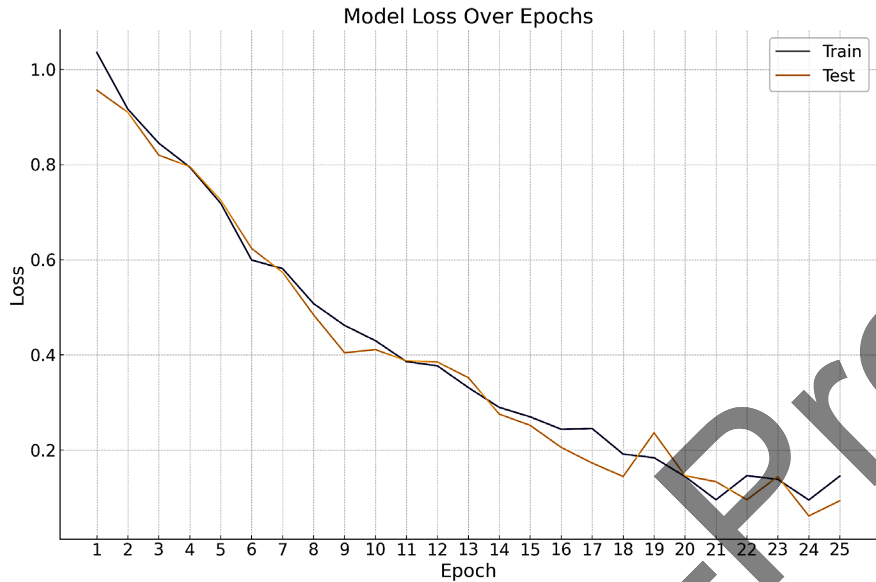


Figure 5: Loss vs Epochs

Figure 4 shows the graph that presents the proposed model's accuracy over 25 epochs, comparing the accuracy scores of the proposed model during the training and testing phases. Figure 5 shows the graph comparing the model's performance in terms of Loss for both training and testing datasets. In both the experiments comparing the models against the train set and test set, the proposed model has a steady progression for accuracy and loss analysis with occasional fluctuations. Compared to test data, the model has a similar trend in the observer, with the model performing at par with the training set for loss and a little less in accuracy.

5. Conclusion and Future Work

This study introduced a hybrid model for real-time Fault Diagnosis (FD) in the field of manufacturing robots. The work proposed a hybrid model combining the Depthwise Convolutional Neural Networks (CNN) with Bidirectional Long Short-Term Memory (BiLSTM) networks. It is focused on addressing the challenges inherently related to high-dimensional sensor data, which are obviously nonlinear and non-Gaussian—the proposed work employed Monte Carlo Filtering (MCF) for the purpose of initial Feature Extraction (FE). Using the FE, the work efficiently FD using the proposed DC-BiLSTM model. For the analysis, the work employed the model in comparison with CNN, LSTM, RNN, and FFNN for performance against accuracy, precision, recall, and F1 scores. For all the metrics, the proposed model showed a significant performance compared to other models.

Future research will explore the possibility of further optimizing the model and its application across different industrial settings.

References

1. Fernandes, M., Corchado, J. M., & Marreiros, G. (2022). Machine learning techniques applied to mechanical fault diagnosis and fault prognosis in the context of real industrial manufacturing use-cases: a systematic literature review. *Applied Intelligence*, *52*(12), 14246-14280.
2. Long, J., Mou, J., Zhang, L., Zhang, S., & Li, C. (2021). Attitude data-based deep hybrid learning architecture for intelligent fault diagnosis of multi-joint industrial robots. *Journal of manufacturing systems*, *61*, 736-745.
3. Chen, T., Liu, X., Xia, B., Wang, W., & Lai, Y. (2020). Unsupervised anomaly detection of industrial robots using sliding-window convolutional variational autoencoder. *IEEE Access*, *8*, 47072-47081.
4. Deebak, B. D., & Al-Turjman, F. (2022). Digital-twin assisted: Fault diagnosis using deep transfer learning for machining tool condition. *International Journal of Intelligent Systems*, *37*(12), 10289-10316.
5. Li, H., Hu, G., Li, J., & Zhou, M. (2021). Intelligent fault diagnosis for large-scale rotating machines using binarized deep neural networks and random forests. *IEEE Transactions on Automation Science and Engineering*, *19*(2), 1109-1119.
6. W. Du and Y. Wang, "Stacked Convolutional LSTM Models for Prognosis of Bearing Performance Degradation," 2019 Prognostics and System Health Management Conference (PHM-Qingdao), Qingdao, China, 2019, pp. 1-6, doi: 10.1109/PHM-Qingdao46334.2019.8943063.
7. Z. Ding, J. Zhou, B. Liu and W. Bai, "Research of Intelligent Fault Diagnosis Based on Machine Learning," 2021 International Conference on Computer Network, Electronic and Automation (ICCNEA), Xi'an, China, 2021, pp. 143-147, doi: 10.1109/ICCNEA53019.2021.00040.
8. A. Agarwal, A. Sinha and D. Das, "FauDigPro: A Machine Learning based Fault Diagnosis and Prognosis System for Electrocardiogram Sensors," 2022 International Conference on Maintenance and Intelligent Asset Management (ICMIAM), Anand, India, 2022, pp. 1-6, doi: 10.1109/ICMIAM56779.2022.10146898.

9. D. Tang, F. Ding, B. Deng, P. Zhang, Q. Wang and H. Lv, "An Intelligent Fault Diagnosis Method for Street Lamps," 2021 International Conference on Internet, Education and Information Technology (IEIT), Suzhou, China, 2021, pp. 300-303, doi: 10.1109/IEIT53597.2021.00073.
10. A. Almounajjed, A. K. Sahoo, M. K. Kumar and M. D. Alsebai, "Investigation Techniques for Rolling Bearing Fault Diagnosis Using Machine Learning Algorithms," 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2021, pp. 1290-1294, doi: 10.1109/ICICCS51141.2021.9432177.
11. V. Singh, A. Yadav and S. Gupta, "Open Switch Fault Diagnosis of Three-Phase Battery-Fed Capacitor Clamped Inverter using Machine-Learning Algorithm," 2023 1st International Conference on Circuits, Power and Intelligent Systems (CCPIS), Bhubaneswar, India, 2023, pp. 1-6, doi: 10.1109/CCPIS59145.2023.10291256.
12. H. Liang, Y. Liu, C. Liu, W. Liu and W. Li, "Fault Diagnosis with Spacecraft High-Dimensional Data Based on Machine Learning," 2021 CAA Symposium on Fault Detection, Supervision, and Safety for Technical Processes (SAFEPROCESS), Chengdu, China, 2021, pp. 1-6, doi: 10.1109/SAFEPROCESS52771.2021.9693733.
13. Q. He et al., "Application and Comparative Analysis of Traditional Machine Learning and Deep Learning in Transmission Line Fault Classification," 2022 IEEE 5th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Chongqing, China, 2022, pp. 1715-1719, doi: 10.1109/IMCEC55388.2022.10020121.
14. J. Gu, Z. Luo, J. Wang, and Y. Shen, "Research on Bearing Cross-domain Fault Diagnosis Based on Invariant Subspace Learning with Tensor Alignment," 2020 11th International Conference on Prognostics and System Health Management (PHM-2020 Jinan), Jinan, China, 2020, pp. 461-465, doi: 10.1109/PHM-Jinan48558.2020.00089.
15. J. Song, "Simulation of Hierarchical Fault Diagnosis Model for Aircraft Electromechanical Systems Based on Machine Learning Algorithms," 2023 International Conference on Power, Electrical Engineering, Electronics and Control (PEEEC), Athens, Greece, 2023, pp. 525-530, doi: 10.1109/PEEEC60561.2023.00108.
16. B. Yang, Y. Lei, F. Jia, and S. Xing, "A Transfer Learning Method for Intelligent Fault Diagnosis from Laboratory Machines to Real-Case Machines," 2018 International

Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC), Xi'an, China, 2018, pp. 35-40, doi: 10.1109/SDPC.2018.8664814.

17. Y. Chen, A. Zhu, and Q. Zhao, "Rolling Bearing Fault Diagnosis Based On Flock Optimization Support Vector Machine," 2023 IEEE 7th Information Technology and Mechatronics Engineering Conference (ITOEC), Chongqing, China, 2023, pp. 1700-1703, doi: 10.1109/ITOEC57671.2023.10292080.
18. Z. Lai, Y. Dong, H. Ren, and R. Lu, "A Multi-stage Optimized Fault Diagnosis Model for Imbalanced Fault Data in Manufacturing Process," 2022 IEEE 11th Data-Driven Control and Learning Systems Conference (DDCLS), Chengdu, China, 2022, pp. 210-215, doi: 10.1109/DDCLS55054.2022.9858502.
19. L. R. Chandran, K. Ilango, M. G. Nair, A. A. Kumar, A. A. Kumar and A. R. M, "Multilabel External Fault Classification of Induction Motor using Machine Learning Models," 2022 Third International Conference on Intelligent Computing Instrumentation and Control Technologies (ICICICT), Kannur, India, 2022, pp. 559-564, doi: 10.1109/ICICICT54557.2022.9917882.
20. Z. Liu et al., "Intelligent Fault Diagnosis of Nuclear grade Electric Equipment Based on Quantum Genetic Support Vector Machine," 2023 IEEE 12th Data-Driven Control and Learning Systems Conference (DDCLS), Xiangtan, China, 2023, pp. 1715-1719, doi: 10.1109/DDCLS58216.2023.10166736.