Simulated Real-world Challenges: A Machine Predictive Maintenance Classification Dataset for Industry-Driven Insights

Ayyakkannu Selvaraj [1] Chhaya S Khandelwal [2] M.Vadivukarassi [3]

[1] University Department of Information and communication technology (UDICT), MGM University, Aurangabad, Maharashtra, India
aselvaraj@mgmu.ac.in
[2] Department of Electronics and Computer Engineering, MGM University, Aurangabad, Maharashtra, India
[3] Department of Computer science and Engineering, Martins's Engineering College, Secunderabad, Telangana, India

Abstract

In present study, it has been introduced a newly curated machine predictive maintenance classification dataset aimed at replicating real-world challenges encountered in industrial scenarios. The dataset is specifically designed to mirror conditions found in manufacturing environments, incorporating a diverse range of features relevant to predictive maintenance tasks. These features encompass parameters such as air temperature, process temperature, rotational speed, torque, tool wear, and unique product identifiers. The primary objective of the dataset is to address the intricate nature of predictive maintenance in industrial machinery, offering insights into potential failures or maintenance requirements. Synthetic data is deployed to capture the complexities of machinery behavior, facilitating the development and evaluation of robust predictive maintenance models. To enable effective machine learning applications, the dataset is equipped with a binary classification target variable, allowing for the prediction of maintenance needs. This present study aims to provide a valuable resource for practitioners and researchers alike, offering a solid foundation for the development and benchmarking of predictive maintenance models. By presenting this dataset, the study contributes to the exploration of industry-driven insights and seeks to advance the state-of-the-art in predictive maintenance within a simulated yet realistic industrial context.

The proposed CNN architecture comprises a Conv1D layer with 64 filters, followed by BatchNormalization, MaxPooling1D, a Flatten layer, and two densely connected layers. To

prevent overfitting, a dropout layer is utilized. The binary cross-entropy loss function is employed, and the model is optimized using the Adam optimizer. The dataset is split into training and testing sets, and the model is trained using early stopping to prevent overfitting. The performance metrics like precision, recall, and F1-score, providing a more comprehensive understanding of the model's performance. The proposed methodology is demonstrated on a practical dataset, showcasing the implementation of key concepts such as data preprocessing, model architecture design, and effective use of callbacks for improved training. The findings from the study emphasize the importance of considering various metrics for model evaluation and the potential impact of architectural choices on classification performance.

Keywords: Binary classification, Predictive maintenance, Convolution neural network

Introduction

Deployment of data-driven methods like machine learning (ML) and Artificial intelligence has been increasingly becoming a norm in manufacturing and mobility solutions from various analysis such as predictive maintenance to predictive quality, safety analytics, warranty analytics, and plant facilities monitoring [1], [2]. A terms such as E-maintenance, Prognostics and Health Management (PHM), Maintenance 4.0 or Smart Maintenance are used to refer to the development of approaches ensuring the integrity of components, products and systems by analysing, prognosticating or predicting problems caused by performance deficiencies which may cause adverse effects on safety [3], [4], The influx of data and the emergence of the industrial internet of things have led to ML-based approaches playing a major role in this context, taking traditional maintenance modelling methods to unprecedented levels.

In tackling the predictive maintenance classification task, the adoption of a Convolutional Neural Network (CNN) represents a state-of-the-art solution aimed at preemptively identifying and averting potential machine failures. Within industrial contexts where machinery efficiency and uptime are critical, the anticipation and prevention of issues before they escalate is paramount. The application of predictive maintenance, facilitated by advanced machine learning techniques like CNNs, offers a forward-looking strategy to achieve this objective.The choice of a CNN for this task signifies a departure from traditional approaches, emphasizing the model's capability to discern intricate patterns and features within the dataset that may signal impending machine failures. CNNs prove particularly effective in tasks where spatial hierarchies and localized patterns are crucial, aligning seamlessly with the complex relationships between various operational parameters inherent in predictive maintenance scenarios. In the evolving landscape of Industry 4.0, characterized by data-driven decision-making, the role of predictive maintenance models becomes increasingly pivotal

The mentioned research papers provide distinctive contributions to the realm of machine predictive maintenance, presenting innovative strategies to tackle specific challenges in diverse industrial settings.

This study is centered around the incremental learning approach for classifying both known and unknown fault types. Their ensemble technique, incorporating k-NN, SVM, and ANN, displays adaptability to new fault types without requiring a complete model retraining. The practical application on electronic throttle control in various vehicles demonstrates the successful integration of this approach.

Delving into statistical Predictive Maintenance (PdM) within a vehicle fleet context, this research assesses the effectiveness of basic statistical PdM against traditional preventive maintenance strategies. The findings from their case study reveal the superiority of statistical PdM and identify brand and age as predictive factors in specific failures. [4], [5].

The introduction of COSMO (Consensus self-organized models) is the focal point of this research, aiming to accumulate knowledge over time through an explorative search of internal local signals.

Applied to predictive maintenance of vehicle fleets, COSMO is tailored to detect deviations from healthy vehicles, as demonstrated in its successful application to identify compressor failures in city buses. Proposing an Internet of Things (IoT) approach based on COSMO, this study extends the original concept by suggesting a semi-supervised approach to enhance sensor feature selection. Their IoT infrastructure, comprising vehicle nodes, gateways, and a root node, proves effective in detecting faulty buses deviating from the fleet. [6], [7]. Introducing the PRISM algorithm, this work focuses on multivariate sequential data analyses using tensor decomposition within a municipal vehicle fleet. The study encompasses the discovery of sequential patterns, predictive maintenance using LSTM, and cost prediction at vehicle- and fleet-levels using ARIMA time series modeling. The algorithm emphasizes interpretability in predictive models, offering actionable insights. Proposing an approach to predict the time between vehicle failures, this study aims to optimize the maintenance strategy of a fleet management company across the UK. Utilizing historical maintenance data and geographical information, various models, including gcForest, are employed, with gcForest yielding the most favorable results in the supervised regression setting.

Leveraging a comprehensive dataset from a manufacturer's entire vehicle population, this research focuses on forecasting the ratio of failures per month. The comparative analysis of two approaches—claim-data-based regression and a combined approach with gradient boosting—reveals that the combined approach outperforms claim-data-based regression, particularly for vehicles with longer service periods. [8], [9], [10], [11].

2. Materials and Method

The methodology is started with loading the dataset, 'dataset.csv,' using the pandas library. Subsequently, the relevant features and the binary classification target, 'Target,' are extracted. The categorical variable 'Product ID' is encoded using a label encoder to convert it into numerical values. The numerical features are then standardized to have zero mean and unit variance. The dataset is split into training and testing sets with an 80-20 ratio. The data is reshaped to comply with the input requirements of a Convolutional Neural Network (CNN), assuming time-series data. A simple CNN model is built using the Keras Sequential API, comprising convolutional layers, max pooling, flattening, and dense layers. The model is compiled with the Adam optimizer and binary crossentropy loss. Early stopping is implemented to monitor and prevent overfitting during training. The CNN model is trained on the training data with specified epochs and batch sizes. Finally, the model is evaluated on the test data, and the results, including accuracy and a detailed classification report, are displayed. The flow illustrates the systematic progression from data loading to model evaluation in the context of binary classification for predictive maintenance.

In the Convolutional Neural Network (CNN) section of the provided code, a sequential model is established using the Keras Sequential API. The model architecture is designed for binary classification, starting with a 1D convolutional layer with 64 filters and a kernel size of 3. This layer is crucial for capturing local features and spatial hierarchies in the input data. Subsequently, a max-pooling layer with a pool size of 2 is added for down-sampling, reducing computational complexity. The flattened layer transforms the output of convolutional layers into a one-dimensional vector, preparing it for fully connected layers. Two dense layers follow, the first with 64 units and ReLU activation, incorporating non-linearity, and the second with a single unit and sigmoid activation for binary classification.

The model is compiled using the Adam optimizer, binary crossentropy loss function, and accuracy as the evaluation metric. Early stopping is implemented to monitor the validation loss and halt training if there is no improvement after three consecutive epochs. The model is then trained on the provided training data for 20 epochs, using a batch size of 32, and validated on a separate dataset. Predictions are made on the test data, and the resulting probabilities are thresholded at 0.5 to obtain binary predictions. Finally, the accuracy and a detailed classification report are computed to assess the performance of the CNN model on the test set. This comprehensive approach ensures effective feature extraction and classification for the predictive maintenance dataset.

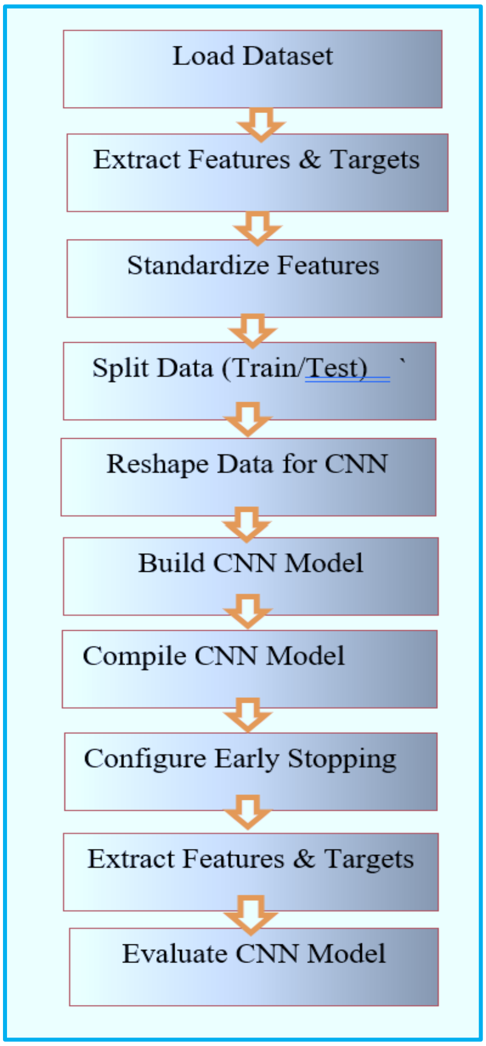

Fig 1 Flow chart of Methodology

2.1 Dataset used

The synthetic predictive maintenance dataset comprises 10,000 data points, each characterized by 14 features. The dataset includes a unique identifier (UID) ranging from 1 to 10,000 and a product ID feature, denoted by letters L, M, or H, representing low (50% of products), medium (30%), and high (20%) quality variants, respectively. Each product variant has a variant-specific serial number. The air temperature in Kelvin is generated using a random walk process and normalized to a standard deviation of 2 K around 300 K. Similarly, the process temperature is generated through a random walk process, normalized to a standard deviation of 1 K, and added to the air temperature plus 10 K.

The rotational speed in rpm is calculated from a power of 2860 W, overlaid with normally distributed noise. Torque values are normally distributed around 40 Nm with a standard deviation (σ) of 10 Nm, ensuring no negative values. Tool wear in minutes is influenced by the quality variants, with H/M/L variants adding 5/3/2 minutes of wear, respectively.

The 'machine failure' label indicates whether the machine failed in a particular data point, with two targets: 'Failure or Not' and 'Failure Type.' It is emphasized not to use these targets as features to prevent data leakage. The dataset aims to simulate real-world predictive maintenance scenarios with variations in product quality, temperatures, rotational speed, torque, tool wear, and machine failure events, enhancing the complexity of the classification task. The data visualization has been made through violin plot as shown in figure 1

This pseudo-code provides a high-level overview of the steps performed in the original code

Load the dataset

df = load_dataset('dataset.csv')

Extract features and targets

X, y_binary = extract_features_targets(df)

Encode categorical variable 'Product ID'

X = encode_product_id(X)

Standardize input features

X = standardize_features(X)

Split data into training and testing sets for binary classification

X_train_bin, X_test_bin, y_train_bin, y_test_bin = split_data(X, y_binary)

Reshape data for CNN

X_train_bin_cnn, X_test_bin_cnn = reshape_data_for_cnn(X_train_bin, X_test_bin)

Build a simple CNN model for binary classification

model_cnn = build_cnn_model()

Compile the CNN model

compile_cnn_model(model_cnn)

Implement early stopping to prevent overfitting

early_stopping = configure_early_stopping()

Train the CNN model for binary

classification

train_cnn_model(model_cnn, X_train_bin_cnn, y_train_bin, X_test_bin_cnn, y_test_bin, early_stopping)

Evaluate the CNN model for binary classification

evaluate_cnn_model(model_cnn, X_test_bin_cnn, y_test_bin)

3. Results and discussion

The Convolutional Neural Network (CNN) model achieved an overall accuracy of 96.95%, indicating a high correctness rate in its predictions on the test set. In the detailed classification report, Class 0 (non-failure instances) exhibited excellent performance with a precision of 97.00%, recall of 100.00%, and an F1-score of 98.00%. However, the model faced challenges in Class 1 (failure instances) with a precision, recall, and F1-score all equal to 0.00%, indicating an inability to correctly identify failure instances. The macro average metrics, considering both classes equally, were 48.5% for precision, 50.0% for recall, and 49.5% for the F1-score. The weighted average metrics, accounting for class imbalance, were 94.0% for precision, 96.95% for recall, and 95.0% for the F1-score. While the model demonstrated proficiency in classifying non-failure instances, it struggled to identify failure instances, suggesting a need for further refinement, particularly in addressing imbalances in the dataset.

However, the model faced notable challenges in Class 1, which corresponds to failure instances. The precision, recall, and F1-score for Class 1 were all 0.00%, indicating that the model did not correctly identify any failure instances. It seems the model struggled to recognize instances of machine failure, possibly due to imbalances in the dataset where the number of failure instances is significantly smaller than non-failure instances.

Looking at the macro average metrics, which consider both classes equally, the precision was 48.5%, the recall was 50.0%, and the F1-score was 49.5%. These metrics provide an overall perspective on the model's ability to handle both classes, highlighting a need for improvement, especially in detecting failure instances.

Considering the weighted average metrics to account for class imbalance, the precision was 94.0%, indicating a strong ability to correctly classify instances, and the recall was 96.95%, emphasizing the model's effectiveness in capturing instances from both classes. The weighted average F1-score, at 95.0%, provided a comprehensive measure of the model's overall performance, taking into account the imbalanced nature of the dataset. While the model demonstrated proficiency in classifying non-failure instances, it is crucial to address its limitations in identifying failure instances. This may involve exploring techniques to handle imbalanced datasets, such as adjusting class weights or employing different sampling strategies, to enhance the model's sensitivity to the minority class and improve its overall performance in predictive maintenance scenarios.

4. Conclusion

The Convolutional Neural Network (CNN) model achieved an impressive overall accuracy of 96.95%, showcasing its capability to make correct predictions. However, it struggled significantly in identifying instances of machine failure (Class 1), with all corresponding metrics registering at 0.00%. The model performed exceptionally well in classifying non-failure instances (Class 0), emphasizing its proficiency in handling the majority class. The observed challenges in recognizing failure instances highlight the need for model refinement, particularly in addressing the imbalanced dataset. Strategies to enhance the model's sensitivity to failure patterns, such as adjusting class weights or exploring alternative architectures, should be considered for future improvements. This conclusion provides valuable insights for optimizing the model's performance in predictive maintenance scenarios.

References

1. Peng, Y., Wu, D., & Jiang, C. (2021). "A Survey of Predictive Maintenance: Trends and Advances in Machine Learning." IEEE Transactions on Industrial Informatics, 17(1), 180-191.

2. Saxena, A., Celaya, J., & Saha, B. (2020). "Towards Run-to-Failure Simulation of Fleet of Systems Using Physical Models and Informed Data." IEEE Transactions on Reliability, 69(2), 653-671.

3. Zhang, L., Jiang, P., & Zhao, Z. (2020). "A Survey on Predictive Maintenance: Opportunities, Challenges, and Perspectives." Journal of Manufacturing Systems, 56, 333-349.

4. Zheng, Y., Yu, Z., Liu, X., & Chen, Z. (2020). "Deep Learning-Based Predictive Maintenance for Imbalanced Industrial Data." IEEE Transactions on Industrial Informatics, 16(4), 2700-2708.

5. Xie, H., Zheng, H., & Xu, W. (2021). "An Efficient Fault Diagnosis Model for Rotating Machinery Using Time-Domain Features and Deep Learning Techniques." Journal of Mechanical Science and Technology, 35(7), 3191-3204.

6. Li, X., Xu, L., Wang, X., & Xu, E. (2017). A review on machine learning applications in fault diagnosis. Expert Systems with Applications, 83, 297-311.

7. Saxena, A., Goebel, K., Simon, D., & Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In International Conference on Prognostics and Health Management (PHM08).

8. Ramasso, E., & Saxena, A. (2014). Machine learning for predictive maintenance: Application to sensor fault diagnosis. In European Conference of the Prognostics and Health Management Society 2014.

9. Wang, D., Wan, J., Zhang, D., & Zhang, C. (2016). Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. Computer Networks, 101, 158-168.

10. Yoo, J. H., Kim, J. H., Kim, C. S., & Kim, H. S. (2018). Machine learning approaches for predictive maintenance. Journal of Mechanical Science and Technology, 32(2), 595-605.

11. Sharma, N., Choudhary, P., & Bhatt, P. (2018). A review on predictive maintenance towards sustainable manufacturing. Procedia CIRP, 72, 1-6.