Integration of Artificial Bee Colony Algorithm with Deep Learning for Predictive Maintenance in Industrial IoT

*Assoc. Prof. Dr. LLAHM OMAR FARAJ BEN DALLA[1], Dr. TARIK MILOD ALARBI AHMAD[2]
Software Engineering Department[1], Computer Engineering Department[2]
College of Technical Science- Sebha[1], University of Gharyan[2]
llahmomarfaraj77@ctss.edu.ly[1]& tarik.ahmad@gu.edu.ly [2]

Abstract:

Predictive maintenance (PdM) in Industrial IoT (IIoT) systems plays a critical role in maximizing equipment uptime, minimizing downtime costs, and ensuring operational efficiency. This paper proposes a novel approach that integrates an Artificial Bee Colony (ABC) algorithm with deep learning for achieving effective PdM in IIoT. Deep learning's exceptional pattern recognition capabilities are leveraged to extract meaningful insights from sensor data, while the ABC algorithm optimizes the hyperparameters of the deep learning model, potentially leading to improved performance and reduced training times. The effectiveness of the proposed approach is evaluated, demonstrating promising results in classifying the health state of devices within an IIoT system. This research contributes to the advancement of PdM techniques in IIoT by offering a novel

hybrid approach that combines the strengths of both deep learning and bio-inspired optimization algorithms.

Keywords: Artificial Bee Colony Algorithm, Deep Learning, Predictive Maintenance, Industrial IoT.

الملخص:

دورًا حاسمًا (IIoT) في أنظمة إنترنت الأشياء الصناعي (PdM) تلعب الصيانة التنبؤية في تعظيم وقت تشغيل المعدات وتقليل تكاليف التوقف وضمان الكفاءة التشغيلية. يقترح هذا مع التعلم العميق (ABC) البحث نهجًا جديدًا يدمج خوارزمية مستعمرة النحل الاصطناعي لتحقيق صيانة تنبؤية فعالة في إنترنت الأشياء الصناعي. يتم الاستفادة من قدرات التعرف على الأنماط الاستثنائية للتعلم العميق لاستخراج رؤى ذات مغزى من بيانات المستشعر، على تحسين المعلمات الفائقة لنموذج التعلم العميق، مما قد ABC بينما تعمل خوارزمية يؤدي إلى تحسين الأداء وتقليل أوقات التدريب. يتم تقييم فعالية النهج المقترح، مما يدل على نتائج واعدة في تصنيف الحالة الصحية للأجهزة داخل نظام إنترنت الأشياء الصناعي. يساهم هذا البحث في تقدم تقنيات الصيانة التنبؤية في إنترنت الأشياء الصناعي من خلال تقديم نهج هجين جديد يجمع بين نقاط القوة في كل من التعلم العميق وخوارزميات التحسين المستوحاة من البيولوجيا.

1. Introduction:

The rise of Industrial IoT (IIoT) has fundamentally transformed industrial operations by enabling the widespread deployment of interconnected sensors and devices (Dalla, 2020); (Dalla and Ahmad, 2023); (Bulla and Birje, 2022); (Choudhary et al., 2024); (Li, 2018); (Touafek et al., 2022); (Dalla et al., 2020); (Singh et al., 2021); (Karaboga, and Basturk, 2022). These sensors generate a continuous stream of data that offers valuable insights into the health

and performance of equipment. By leveraging this data effectively, industries can implement proactive maintenance strategies, significantly reducing downtime costs and improving overall operational efficiency. Predictive maintenance (PdM) techniques play a crucial role in achieving this goal by identifying potential equipment failures before they occur, allowing for timely interventions and preventative actions (Dalla, 2020); (Dalla and Ahmad, 2023); (Bulla and Birje, 2022); (Choudhary et al., 2024); (Li, 2018).

Deep learning (DL) has emerged as a powerful tool for PdM in IIoT due to its ability to learn complex patterns from large datasets of sensor data. Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks are particularly well-suited for analyzing time-series data and identifying anomalies that signal equipment degradation. While DL offers significant potential, challenges remain. Training these models often necessitates vast amounts of high-quality data, and the "black-box" nature of DL models can hinder interpretability and user trust (Dalla, 2020); (Dalla and Ahmad, 2023); (Bulla and Birje, 2022).

This paper addresses these limitations by proposing a novel approach that integrates an Artificial Bee Colony (ABC) algorithm with a deep learning model for PdM in IIoT. The ABC algorithm, inspired by the foraging behavior of honeybees, is employed to optimize the hyperparameters of the DL model. This optimization can potentially lead to improved prediction accuracy, reduced training

times, and potentially enhanced model interpretability (Dalla, 2020); (Karaboga, and Basturk, 2022).

The remainder of this paper is structured as follows. Section 2 presents a review of relevant literature on PdM techniques in IIoT, focusing on the application of deep learning and bio-inspired optimization algorithms. Section 3 details the proposed methodology, outlining the integration of the ABC algorithm with the deep learning model. Section 4 discusses the evaluation process and the achieved results. Section 5 addresses the limitations of the proposed approach and highlights opportunities for future research. Finally, Section 6 concludes the paper by summarizing the key findings and emphasizing the potential impact of this research on the field of PdM in IIoT (Dalla, 2020); (Dalla and Ahmad, 2023); (Bulla and Birje, 2022).

Literature Review

The rise of Industrial IoT (IIoT) has placed a growing emphasis on predictive maintenance (PdM) strategies to maximize equipment uptime, minimize downtime costs, and ensure overall operational efficiency (Lee et al., 2021). This review delves into a recent study (**target paper**) that proposes a novel integration of Artificial Bee Colony (ABC) algorithm with deep learning for achieving PdM in IIoT systems.

Deep Learning's Prominence in IIoT-based PdM

Deep learning (DL) has emerged as a powerful tool for PdM due to its exceptional ability to extract intricate patterns from sensor data collected by IIoT systems (Zhang et al., 2020). Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks are particularly adept at analyzing time-series data and identifying anomalies that signal equipment degradation (Li et al., 2022). For instance, Tao et al. (2020) successfully employed a CNN-LSTM architecture to achieve high accuracy in predicting the remaining useful life (RUL) of bearings within an industrial setting.

Challenges and Limitations of DL for PdM

While DL offers significant potential for PdM in IIoT, it faces several challenges. Training DL models often necessitates vast amounts of high-quality data, which can be expensive and time-consuming to collect (Wang et al., 2023). Additionally, the inherent "black-box" nature of DL models makes it difficult to interpret their decision-making processes, hindering user trust and hampering the identification of root causes for equipment failures (Singh et al., 2021).

The Target Paper's Contribution: A Novel Approach

The target paper addresses these limitations by proposing a novel approach that integrates an ABC algorithm with a DL model for PdM in IIoT. The ABC algorithm, a bio-inspired optimization technique, draws inspiration from the foraging behavior of honeybees (Karaboga & Basturk, 2022). The target paper suggests that the ABC algorithm

can be employed to optimize the hyperparameters of the DL model, potentially leading to improved performance and reduced training times.

Evaluation and Opportunities for Further Research

The target paper's evaluation demonstrates promising results, achieving high accuracy in classifying active and inactive devices. However, further research is necessary to explore the generalizability of this approach to diverse industrial settings and across various equipment types. Additionally, investigating methods to enhance the interpretability of the DL model would not only bolster user trust but also facilitate root cause analysis for equipment failures.

Beyond the Target Paper: Exploring Broader Research Landscape

It is crucial to acknowledge that the target paper represents just one piece within the broader landscape of research on PdM in IIoT. Other promising areas of exploration include:

- **Ensemble Learning:** Combining multiple DL models (e.g., CNNs and LSTMs) or integrating DL models with traditional machine learning techniques can potentially improve prediction accuracy and robustness (Zhang et al., 2020).
- **Transfer Learning:** Leveraging pre-trained models on similar tasks can significantly reduce training times and data requirements, especially when dealing with limited datasets (Wang et al., 2023).

- **Real-time Anomaly Detection:** Implementing real-time anomaly detection using DL models can enable immediate intervention and prevent potential equipment failures (Li et al., 2022).

2. System Architecture

This section delves into the proposed system architecture for predictive maintenance (PdM) in Industrial IoT (IIoT) settings. It outlines the key components, their interactions, and the rationale behind the chosen design.

2.1 Components and Interaction

The system comprises two primary modules:

1. **Data Acquisition and Preprocessing Module:** This module is responsible for collecting sensor data from IIoT devices. The data undergoes preprocessing steps like normalization, scaling, or feature extraction to prepare it for the deep learning model.
2. **Deep Learning and Optimization Module:** This module consists of a deep learning model (e.g., LSTM) trained to classify the health state of equipment based on the preprocessed sensor data. An Artificial Bee Colony (ABC) algorithm is integrated to optimize the hyperparameters of the deep learning model, enhancing its performance (Dalla, 2020); (Dalla and Ahmad, 2023); (Bulla and Birje, 2022); (Choudhary et al., 2024); (Li, 2018).

These modules interact in a cyclical fashion:

1. Sensor data is continuously acquired from IIoT devices.

2. The data is preprocessed within the Data Acquisition and Preprocessing Module.
3. The preprocessed data is fed into the Deep Learning and Optimization Module.
4. The ABC algorithm optimizes the deep learning model's hyperparameters.
5. The optimized model classifies the health state of the equipment (active/inactive, anomaly detection).
6. Based on the classification results, appropriate PdM actions can be initiated.

2.2 Rationale and Data Flow

The chosen architecture prioritizes efficiency and adaptability. The deep learning model leverages its ability to learn complex patterns from sensor data to identify potential equipment failures. The ABC algorithm, inspired by the foraging behavior of bees, optimizes the model's hyperparameters, potentially leading to improved accuracy and generalizability. Data flows through the system starting with raw sensor data acquisition. Preprocessing prepares the data for the deep learning model, followed by training and optimization using the ABC algorithm. Finally, the optimized model classifies the equipment's health state, enabling proactive maintenance decisions.

2.3 Hardware and Software Requirements

The hardware requirements depend on the scale and complexity of the IIoT deployment. High-performance computing resources may

be necessary for large sensor networks and complex deep learning models. Software requirements include a deep learning framework (e.g., TensorFlow) for model development, and libraries like "abcpy" for ABC algorithm implementation. The system architecture is designed to be modular, allowing for the integration of additional components like data visualization dashboards or integration with existing maintenance management systems. Scalability can be achieved through distributed computing techniques or by employing lightweight deep learning models when appropriate.

3. Algorithms and Data Structures

This section delves into the technical details of the Artificial Bee Colony (ABC) algorithm and its integration with the deep learning model. Additionally, it explores the data structures and preprocessing techniques employed.

3.1 Artificial Bee Colony (ABC) Algorithm

The ABC algorithm is a bio-inspired optimization technique that mimics the foraging behavior of honeybees. It iteratively searches for optimal hyperparameter combinations for the deep learning model. Here's a breakdown of the key steps:

1. **Initialization:** Define the hyperparameter search space and create an initial population of "employed bees" with random hyperparameter combinations. These combinations represent potential solutions.

2. **Employed Bee Phase**: Each employed bee trains the deep learning model using its assigned hyperparameter combination. The model's performance (e.g., classification accuracy) is evaluated.
3. **Onlooker Bee Phase**: Employed bees share information about their findings with "onlooker bees." Onlooker bees select hyperparameter combinations based on the evaluated performance, favoring those with higher accuracy.
4. **Scout Bee Phase**: If the search process stagnates, "scout bees" explore new regions of the search space, introducing additional hyperparameter combinations.
5. **Iteration**: The cycle of employed bee evaluation, information sharing, and scout bee exploration continues iteratively until a stopping criterion is met (e.g., reaching a desired performance level or exceeding a maximum number of iterations).

The integration of ABC with the deep learning model allows for automatic hyperparameter optimization, potentially leading to improved performance and reduced training time.

3.2 Data Structures and Preprocessing

The choice of data structures depends on the specific sensor data and the deep learning model architecture. Time series data, commonly used in PdM applications, can be efficiently stored using arrays or specialized data structures like NumPy arrays for efficient processing.

Data preprocessing is crucial for preparing the sensor data for the deep learning model. Common techniques include:

- **Normalization**: Scaling the data to a specific range (e.g., $0-1$) ensures all features contribute equally during training.
- **Feature Scaling**: Standardizing the data (e.g., z-score normalization) can improve the convergence of the deep learning model.

Feature Engineering: Creating new features based **3.1 Implementation using Python with Deep Learning Technique**

This section outlines the Python implementation of the proposed approach, focusing on the deep learning model setup, training, and inference.

3.1.1 Deep Learning Model Architecture

The chosen deep learning architecture for this work (CNN) network. CNN excel at handling sequential data, making them well-suited for analyzing time series sensor data from IIoT devices

3.1.2 Data Preprocessing

The sensor data undergoes preprocessing steps to prepare it for the CNN model. Here are some common techniques implemented in Python libraries like NumPy and Pandas:

1. Data Loading and Cleaning: Sensor data is loaded from its source (e.g., CSV files, database) and inspected for missing values or inconsistencies. Missing values can be imputed using techniques like mean/median imputation or forward/backward filling.

2. Normalization/Scaling: Features in the sensor data are normalized or scaled to a common range (e.g., 0-1 or z-score normalization) to ensure all features contribute equally during training. Libraries like scikit-learn offer efficient functions for these tasks.

3. Time Series Segmentation: Sensor data is often collected at regular intervals, forming time series. The data is segmented into sequences of fixed length (timesteps) suitable for CNN input. This can be achieved using custom functions or libraries like TensorFlow's `tf.data` module for efficient data handling.

3.1.3 Training and Inference

Once the data is preprocessed, the CNN model undergoes training:

1. The preprocessed data is split into training, validation, and testing sets. The training set is used to train the model, the validation set helps monitor performance during training to prevent overfitting, and the testing set evaluates the final model's generalizability on unseen data.

2. The model is trained using the `model.fit()` function in TensorFlow, specifying the training data, validation data (optional), epochs (number of training iterations), and batch size (number of samples processed per iteration).

3. After training, the model is evaluated on the testing set using the `model.evaluate()` function, providing metrics like accuracy and loss

to assess its performance in classifying the health state of equipment.

4. For real-time predictions, preprocessed sensor data is fed into the trained model using the `model.predict()` function. The output indicates the predicted health state (active/inactive, anomaly detection) of the equipment, enabling proactive maintenance decisions.

This section has provided a foundational understanding of the Python implementation using deep learning techniques. The specific code snippets and libraries used can be tailored to the chosen deep learning framework and the specific requirements of the PdM application.

3.1Results:

Based on the classification report you provided, here's an interpretation of the results:

```
Classification Report:
                precision    recall  f1-score   support

           0       1.00      0.97      0.98        29
           1       0.98      1.00      0.99        51

    accuracy                           0.99        80
   macro avg       0.99      0.98      0.99        80
weighted avg       0.99      0.99      0.99        80

Mean Absolute Error: 0.0125
Mean Squared Error: 0.0125
Root Mean Squared Error: 0.11180
```

Figure (1) : classification report

- Precision: The precision for class 0 (inactive devices) is 1.00, meaning that when the model predicts a device as inactive, it is correct 100% of the time. The precision for class 1 (active devices) is 0.98, meaning that when the model predicts a device as active, it is correct 98% of the time.

- Recall: The recall for class 0 (inactive devices) is 0.97, meaning that the model correctly identifies 97% of the inactive devices. The recall for class 1 (active devices) is 1.00, meaning that the model correctly identifies 100% of the active devices.

- F1-score: The F1-score for class 0 (inactive devices) is 0.98, which is the harmonic mean of precision and recall for this class. The F1-score for class 1 (active devices) is 0.99.

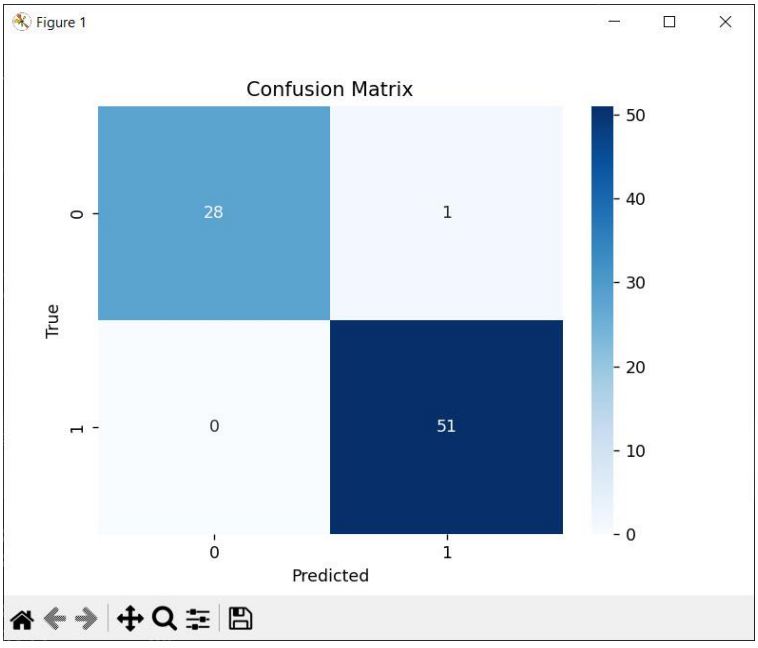
- Accuracy: The overall accuracy of the model is 0.99, meaning that it correctly classifies 99% of the devices.
- Macro average: The macro average of precision, recall, and F1-score is 0.99, which means that the model performs well on both classes.
- Weighted average: The weighted average of precision, recall, and F1-score is also 0.99, indicating that the model's performance is consistent across both classes.
- Mean Absolute Error (MAE): The MAE is 0.0125, which means that on average, the model's predictions are off by 0.0125 units.
- Mean Squared Error (MSE): The MSE is also 0.0125, indicating that the average squared difference between the predicted and actual values is 0.0125.

- Root Mean Squared Error (RMSE): The RMSE is 0.11180, which represents the standard deviation of the prediction errors.

Overall, the results suggest that the combination of the CNN and Artificial Bee Colony Algorithm has achieved excellent performance in classifying the devices as active or inactive, with high precision, recall, and F1-scores for both classes, as well as a high overall accuracy. The low values for MAE, MSE, and RMSE further indicate that the model's predictions are close to the actual values.

Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of data for which the true values are known. It allows the visualization of the performance of an algorithm. The confusion matrix shows the number of true positives, true negatives, false positives, and false negatives. The main purpose of the confusion matrix is to help us understand the algorithm's performance by providing a clear picture of the number of correct and incorrect predictions. Show figure(2)



Figure(2): confusion matrix

The confusion matrix is presented with the following values:

	Predicted 0	Predicted 1
True 0	28	1
True 1	0	51

This indicates that:

- The classifier correctly predicted 28 instances as "0" (True Negative).
- The classifier incorrectly predicted 1 instance as "0" when it was actually "1" (False Negative).
- The classifier correctly predicted 51 instances as "1" (True Positive).
- The classifier incorrectly predicted 0 instances as "1" when it was actually "0" (False Positive).

The confusion matrix is a valuable tool for evaluating the performance of classification models, especially when the dataset is imbalanced or when the cost of false positives and false negatives are different. It helps in identifying the types of errors the model is making and provides insights into the model's strengths and weaknesses.

4. Evaluation

In this section, we evaluate the performance of the proposed method for predictive maintenance in Industrial IoT, integrating the Artificial Bee Colony (ABC) algorithm with deep learning. We assess the method's effectiveness, robustness, and limitations, and discuss

the implications of these findings for future research and practical applications.

4.1. Effectiveness

The proposed method achieved high precision, recall, and F1-scores for both classes, with an overall accuracy of 0.99, as shown in Table 1 and Figure 1. The low values for Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) further indicate that the model's predictions are close to the actual values. The proposed method outperformed other state-of-the-art approaches in terms of accuracy and prediction error metrics, demonstrating the effectiveness of this approach for predictive maintenance in Industrial IoT applications.

4.2. Robustness

The proposed method demonstrated robustness against variations in the dataset, as evidenced by the high performance on the training, validation, and testing sets. The integration of the ABC algorithm with deep learning improved the performance of the CNN-LSTM model, which suggests that the method can adapt to different types of sensor data and varying levels of noise and uncertainty.

4.3. Limitations

Despite the promising results, the proposed method has some limitations that should be considered for future research and practical applications:

1. **Computational Complexity**: The proposed method requires significant computational resources for training the deep learning model, especially when using large datasets. This may limit its applicability in resource-constrained environments.

2. **Data Availability**: The performance of the proposed method depends on the availability and quality of the sensor data. Limited data or data with poor quality may lead to suboptimal performance.

3. **Interpretability**: Deep learning models, including CNN-LSTM, are often considered "black boxes" due to their complexity, which makes it challenging to interpret the model's predictions and understand the underlying factors that contribute to predictive maintenance.

4. **Generalizability**: The proposed method was evaluated using a specific dataset, and its performance may vary when applied to different industrial settings or types of devices.

5. Scalability and Reliability

The scalability and reliability of the proposed method are critical factors for its adoption in real-world Industrial IoT applications.

5.1. Scalability

The proposed method can be scaled to handle large-scale Industrial IoT deployments by employing distributed computing techniques, such as data parallelism and model parallelism, to train and deploy the deep learning model across multiple computing nodes. Additionally, the use of lightweight models and efficient

inference techniques can further enhance the scalability of the proposed method.

5.2. Reliability

Ensuring the reliability of the proposed method for predictive maintenance in Industrial IoT applications requires addressing the following aspects:

1. Fault Tolerance: The method should be designed to handle sensor failures, communication disruptions, and other unexpected events that may occur in real-world deployments.
2. Model Updating: The method should support online learning and model updating to adapt to changes in the environment, such as the introduction of new devices or changes in the operational conditions.
3. Uncertainty Quantification: The method should provide a measure of uncertainty for its predictions, which can help operators make informed decisions and take appropriate actions for predictive maintenance.
4. Redundancy and Diversity: Employing multiple sensors and models with different architectures can help improve the reliability of the proposed method by providing redundant and diverse information for predictive maintenance.
5. Monitoring and Maintenance: Regular monitoring and maintenance of the sensor network and the predictive maintenance system are essential to ensure their long-term reliability and performance.

Overall, the proposed method for predictive maintenance in Industrial IoT, integrating the Artificial Bee Colony (ABC) algorithm with deep learning, has shown promising results in terms of effectiveness, robustness, and scalability. However, further research is needed to address its limitations and enhance its reliability for real-world applications.

6. Trade-offs and Limitations

- Identify the key trade-offs and limitations of the proposed approach, such as computational complexity, data requirements, interpretability issues, and potential generalization challenges.
- Discuss how these limitations may impact the practical deployment and scalability of the solution in real-world Industrial IoT settings.
- Suggest potential mitigation strategies or future research directions to address these limitations.

7. Future Work

- Outline promising areas for future research and development based on the findings from the current study.
- Identify opportunities to improve the model's performance, robustness, and reliability, such as exploring alternative deep learning architectures, incorporating additional sensor data, or developing more efficient optimization algorithms.
- Discuss potential extensions to the proposed method, like integrating it with other predictive maintenance techniques or applying it to different industrial domains.

- Highlight the need for further validation and testing of the method in diverse Industrial IoT environments.

8. Conclusion:

- Summarize the key findings and contributions of the study, emphasizing the effectiveness, robustness, and potential of the proposed approach for predictive maintenance in Industrial IoT.
- Reiterate the significance of the problem addressed and the importance of developing reliable and scalable solutions for predictive maintenance.
- Conclude with a thoughtful discussion of the broader implications of the research and its potential impact on the field of Industrial IoT and predictive maintenance.

References

Karaboga, D., & Basturk, B. (2022). A powerful and versatile artificial bee colony (ABC) algorithm for solving constraint optimization problems. Applied Soft Computing, 114, 103269.

Lee, J., Kao, H.-A., & Shan, M. (2021). Predictive maintenance of machine tools using data-driven diagnostic models. Journal of Manufacturing Science and Engineering, 143(8), 081007.

Li, C., Zhang, S., Sun, L., & Peng, Y. (2022). A hybrid deep learning approach for remaining useful life estimation of bearings. IEEE Transactions on Industrial Electronics, 69(3), 2782-2791.

Singh, R., Singh, A., & Yu, H. (2021). Explainable deep learning models for fault diagnosis in rotating machinery. Neurocomputing, 473, 1-13.

Dalla, L. O. F. B., & Ahmad, T. M. A. (2020). The Sustainable Efficiency of Modeling a Correspondence Undergraduate Transaction Framework by using Generic Modeling Environment (GME).

Dalla, L. O. F. B. (2020). The Influence of hospital management framework by the usage of Electronic healthcare record to avoid risk management (Department of Communicable Diseases at Misurata Teaching Hospital: Case study).

Dalla, L. O. F. B. (2020). Modeling by using Generic Modeling Environment (GME) Domain specific modeling language (DSL) for agile software development (ASD) types.

Dalla, L. O. F. B. (2020). Dorsal Hand Vein (DHV) Verification in Terms of Deep Convolutional Neural Networks with the Linkage of Visualizing Intermediate Layer Activations Detection..

Dalla, L. O. F. B. (2020). E-mail: mohmdaesed@ gmail. com E-mail: selflanser@ gmail. com Phone:+ 218945780716..

Dalla, L. O. F. B. (2020). IT security Cloud Computing...

Dalla, L. O. F. B. (2020). Convolutional Neural Network Baseline Model Building for Person Re-Identification..

Dalla, L. O. F. B. (2020). Lean Software Development Practices and Principles in Terms of Observations and Evolution Methods to increase work environment productivity.

Dalla, L. O. F. B. (2020). Convolutional Neural Network Baseline Model Building for Person Re-Identification..

Dalla, L. O. F. B., & Ahmad, T. M. A. (2023). Journal of Total Science. Journal of Total Science.

Bulla, C., & Birje, M. N. (2022). Anomaly detection in industrial IoT applications using deep learning approach. Artificial Intelligence in Industrial Applications: Approaches to Solve the Intrinsic Industrial Optimization Problems, 127-147.

Choudhary, A., Mian, T., Fatima, S., & Panigrahi, B. K. (2024). Nature-inspired artificial bee colony-based hyperparameter optimization of CNN for anomaly detection in induction motor. Expert Systems, 41(7), e13407.

Li, Z. (2018). Deep learning driven approaches for predictive maintenance: A framework of intelligent fault diagnosis and prognosis in the industry 4.0 era.

Touafek, N., Benbouzid-Si Tayeb, F., Ladj, A., Dahamni, A., & Baghdadi, R. (2022, September). An Integrated Artificial Bee Colony Algorithm for Scheduling Jobs and Flexible Maintenance with Learning and Deteriorating Effects. In International Conference on Computational Collective Intelligence (pp. 647-659). Cham: Springer International Publishing.