International Journal of Multidisciplinary Engineering in Current Research - IJMEC
Volume 8, Issue 12, December-2023, http://ijmec.com/, ISSN: 2456-4265

ANTICIPATING MACHINE FAILURES IN AUTOMATED INDUSTRIES USING ML ALGORITHM

Mohammed Abdul Muqtadir [*1], Dr.Syed Abdul Sattar[2]

[*1]Lecturer, Mazoon College, Affiliated to "Missouri University of Science and Technology"

mabdulmuqtadir@gmail.com

[*2]Principle, "Nawab Shah Alam Khan College of Engineering & Technology" Hyd., Telangana.

syedabdulsattar1965@gmail.com

Abstract: Developing predictive models for early detection of machine failures in automated industries using machine learning algorithms is a sophisticated approach aimed at anticipating and mitigating potential breakdowns before they occur. This involves leveraging advanced computational techniques to analyze historical data, identify patterns, and create models capable of forecasting potential failures. By employing machine learning algorithms, these models can continuously learn and adapt to changing conditions, providing a proactive and data-driven solution for maintenance planning and minimizing downtime in automated industrial settings. Monitoring the performance and predicting failures of industrial equipment is crucial for ensuring the quality of manufactured materials and optimizing time and cost in maintenance. This project aims to examine the ongoing research, development, and progress in employing AI/ML techniques for predicting equipment faults in various industries. The surveyed topics in this paper encompass ML algorithms, use cases, and concepts relevant to the application of this technology across diverse industries such as software and hardware.

KEYWORDS: LTSM, ANN, FDD, Deep Learning.

I. INTRODUCTION

Professionals refer to the current state of industries as "The Fourth Industrial Revolution," or "Industry 4.0." Industry 4.0 is centered on the integration of digital and physical technologies in manufacturing settings. Prognostics and health management, or PHM, is becoming a must in the fields of industrial big data and smart manufacturing with the advent of Industry 4.0. It also offers a dependable way to keep track of the condition of industrial equipment concurrently. Industrial systems can become autonomous thanks in large part to Industry 4.0 and its core technologies, which also make automated data collecting from industrial machinery and components possible..

Machine learning algorithms can be used to automatically detect and diagnose faults using the data that has been gathered. But for ML applications in industrial systems, choosing the right machine learning (ML) methods, data kinds, data volumes, and hardware is essential. Time loss and impractical maintenance scheduling might arise from the improper choice of predictive maintenance (PdM) methods, datasets, and data volumes.

In order to help researchers and practitioners select appropriate machine learning algorithms, data volumes, and data kinds for workable machine learning applications, this study intends to present an extensive literature review to discover previous studies and ML implementations. Industrial equipment that is intended to achieve almost zero risks of hidden hazards, malfunctions, pollutants, and accidents in the manufacturing process environment might have its performance degraded and can be identified by predictive maintenance.

DOI: https://doi-ds.org/doilink/12.2023-32776474

Valuable information that can improve overall efficiency in manufacturing processes, system dynamics, and decision support in areas like condition-based maintenance and health monitoring can be found in the large amount of data gathered for machine learning. Advances in technology such as computerized control, communication networks, and information techniques have made it easier to gather a large amount of data on process and operational conditions for automated fault diagnosis and diagnosis (FDD). The datasets that have been gathered can also be used to create more effective procedures for PdM, or intelligent preventative maintenance.

Reduced maintenance costs, fewer repair stops, fewer machine problems, longer spare-part lifespans, lower inventories, increased operator safety, better output, repair verification, higher overall profits, and more are just a few benefits of using machine learning technologies. These benefits are directly related to maintenance practices. Moreover, a crucial component of predictive maintenance is fault detection, which enables enterprises to recognize problems early on.

1.1 Problem statement:

Among the difficulties and obstacles encountered in this field are software-related equipment failures caused by the underutilization of AI/ML algorithms on available data to forecast industrial equipment failures and performance degradation.

Objective:

Manufacturers have recently begun investigating the use of more advanced machine learning algorithms for equipment monitoring and predictive maintenance. It is feasible to attempt developing these complex systems due to the accessibility and affordability of strong processing resources, vast industrial data sets from their own factories, and deep learning AI/ML algorithms.

EXISTING SYSTEM

Main directions followed in the previous studies on IOT based applications are:

1) Defining and specifying metrics to calculate complexity of software,

2)Verifying correctness and validating thoroughness□□

Some metrics have been presented to describe software quality in forms of static and dynamic platforms. In the static platform, features of code structure are measured as metrics. Static measurements are a number of supervisors and a number of bunches. Dynamic platforms measure testing perfectionism. Basic element measurements depend on auxiliary and information stream scope. The connection between product measurements and blame inclination, and also numerous quantifiable programming characteristics has been exactly demonstrated by many researchers.

PROPOSED SYSTEM

Early detection of a potential machine breakdown can notify the system. Using a Deep Learning LSTM model to forecast time series, the method predicts future values in advance and uses a classifier to determine whether to

DOI: https://doi-ds.org/doilink/12.2023-32776474

issue an alert or not.In this suggested system, a model is created using the LSTM method, which accepts an equipment data set as input. Machine learning models are used to train values, which are then used to monitor fault detection and generate alarms.

II. SOFTWARE & HARDWARE REQUIREMENTS:

Software Requirements

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation. The appropriation of requirements and implementation constraints gives the general overview of the project in regards to what the areas of strength and deficit are and how to tackle them.

- **Python idel 3.7 version (or)**
- **Anaconda 3.7 (or)**
- **Jupiter (or)**
- **Google colab**

Hardware Requirements

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

- **Operating system : windows, Linux**
- **Processor : minimum Intel i3**
- **Ram : minimum 4 gb**
- **Hard disk : minimum 250gb**

III. LITERATURE SURVEY

A. Rule-Based Systems

Some of the earliest applications of AI involved controlling industrial plants in Japan, employing primitive implementations of artificial intelligence. Programmable controllers ran software automatically generated from system design specifications. This process involved extracting necessary information and constructing a knowledge base. An intermediate program was initially generated based on specifications, and then it was converted into the target program for the specific controller. This practice dates back to as early as 1988 [1]. Early systems predominantly relied on rule-based knowledge for their operation.

DOI: https://doi-ds.org/doilink/12.2023-32776474

Ford and Nippon were among the pioneers in implementing AI in manufacturing during the early '90s. The implementations of these companies are discussed below [2].

Technical Information Engineering System (TIES) from Ford Auto:

Ford's TIES collected and stored relevant engineering data, experience, and knowledge using the design tool framework. For instance, TIES could analyze customer preferences for an instrument panel from marketing information and connect with the group responsible for designing it. This common framework allowed different groups to plan and design components consistently without losing design coherence.

Quality Design Expert System (QDES) from Nippon Steel:

Nippon Steel's QDES used a case-based reasoning approach to find earlier designs most similar to new customer requests. QDES also employed hypothetical reasoning to identify the best alternatives from a combination of design features. Neural net technology assisted in judging design feasibility, categorizing ambiguous knowledge using fuzzy logic, and providing training to learn new designs.

Vibration Analysis in Rotating Machines:

Expert systems in this context used a built-in analyzer to capture vibrations, identifying significant features associated with machine components. Complex checks were implemented to validate and protect the system from various types of failures. The system utilized expert and historical data to provide detailed fault reports and maintenance recommendations. Machine operators required no knowledge of vibrations, as the expert system could diagnose faults like bearing lubrication issues and motor misalignment early on.

B. Neural/Fuzzy Systems

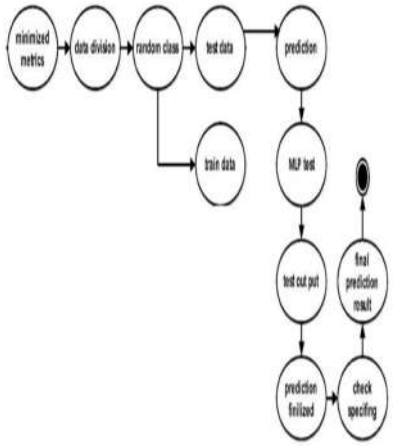
In 1995, research papers discussed the application of Neural/Fuzzy Systems in predicting faults in induction motors [3]. The manufacturing industry extensively uses electrical motors prone to incipient faults due to various conditions and environments. A hybrid neural/fuzzy fault detector was implemented to detect faults in the motor bearing and stator winding insulation. Once trained, this detector not only accurately identified faults but also provided the reasoning behind such detections. Understanding the reasoning facilitated the development of a more effective motor protection system, with artificial neural networks (ANNs) and fuzzy logic enabling easy inference of such reasoning.

IV. SYSTEM DESIGN

System Architecture

ISSN: 2456-4265

DOI: https://doi-ds.org/doilink/12.2023-32776474



What is Machine Learning : -

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of *building models of data*.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain.Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Categories Of Machine Leaning :-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into *classification* tasks and *regression* tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as *clustering* and *dimensionality reduction.* Clustering algorithms identify distinct groups of data, while

DOI: https://doi-ds.org/doilink/12.2023-32776474

dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application

.it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output : identified classes of application outputs must be exercised. Systems/Procedures: interfacing systems or procedures must be invoked.

ISSN: 2456-4265

DOI: https://doi-ds.org/doilink/12.2023-32776474

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, asablackbox.youcannot−seeintoit.Thetestprovidesinputsandrespondsto outputs without considering how the software works. Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should bellowed
- All links should take the user to the correct page.

DOI: https://doi-ds.org/doilink/12.2023-32776474

V. SOFTWARE INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defected countered.

VI. CONCLUSION

This research introduces a short-term prediction strategy that makes use of fault prediction technologies now in use in connection with the LSTM (Long Short-Term Memory) network. The suggested method improves the comprehensiveness and precision of the predicted data by feeding the extracted feature vector into the LSTM for processing. It is expected that this experimental approach will gradually improve when it is used in real-world settings, producing more accurate equipment failure forecasts.

REFERENCES

[1] Souri, A. Hussien, M. Hoseyninezhad, and M. Norouzi, "A systematic review of IoT communication strategies for an efficient smart environment," Transactions on Emerging Telecommunications Technologies, p. e3736, 2019

[2] R. Mahajan, S. K. Gupta, and R. K. Bedi, "Design of Software Fault Prediction Model Using BR Technique," Procedia Computer Science, vol. 46, pp. 849-858, 2015/01/012015.

[3] I. Umesh and G. N. Srinivasan, "Dynamic Software Aging Detection-Based Fault Tolerant Software Rejuvenation Model for Virtualized Environment," in Proceedings of the International Conference on Data Engineering and Communication Technology, 2017, pp. 779-787.

[4] Y. Abdi, S. Parsa, and Y. Seyfari, "A hybrid one-class rule learning approach based on swarm intelligence for software fault prediction," Innov. Syst. Softw. Eng., vol. 11, pp. 289-301, 2015.

DOI: https://doi-ds.org/doilink/12.2023-32776474