An Internship Report on

AI based Bearing Fault Diagnosis

By

Sujith R

Submitted to

PES University

In fulfilment of the requirements Internship

work carried out at

Council of Scientific and Industrial Research

National Aerospace Laboratories

Bangalore 560 017

Under the guidance of

Mr. Brijeshkumar J Shah

&

Dr Sadanand Kulkarni

PES University, Bengaluru

Contents

● INTRODUCTION

Bearings are vital components in rotary machinery, widely used in aerospace, power generation, and automotive industries. Their failure can lead to costly downtimes, reduced performance, and safety risks. Thus, early fault detection and predictive maintenance are essential for ensuring system reliability and longevity .

Traditional fault diagnosis methods—based on vibration signal analysis and manual inspection—require domain expertise and are prone to human error . With advancements in AI and sensor technologies, automated and intelligent diagnostic systems have emerged as effective alternatives.

In this project, we developed a deep learning-based system for **bearing fault classification** using the **CWRU dataset**. Time-domain vibration signals were segmented and converted into **2D spectrograms**, enabling effective feature extraction using a **Convolutional Neural Network (CNN)**. To enhance generalization, we applied data augmentation techniques such as:

- **RMS-scaled Gaussian noise**,
- **Amplitude scaling**, and
- **Frequency masking**.

The model classifies signals into four categories: **Normal**, **Inner Race (IR)**, **Outer Race (OR)**, and **Ball fault**. Additionally, we extracted handcrafted features like **RMS**, **kurtosis**, and **dominant frequency** to build a structured knowledge base.

This knowledge base was integrated with a **Retrieval-Augmented Generation (RAG)** system using a **Grok-powered LLaMA 3 model** to provide fault classification. The result is a hybrid system combining deep learning with semantic reasoning for robust and interpretable fault diagnosis.

1.1 Importance of Bearings in Industrial Applications

Bearings are fundamental components in rotating machinery, enabling smooth motion, supporting loads, and reducing friction between moving parts. Their role is critical in ensuring the **efficiency and stability of mechanical systems** across a wide range of industries. In sectors such as aerospace, automotive, manufacturing, and power generation, the health of bearings directly influences **machine performance, safety, and operational continuity**.

A single bearing failure can escalate into critical system-level faults and cause machine breakdowns, production stops, and unplanned repairs, resulting in important **financial and**

operational losses . Thus, it is critical for bearing health monitoring and fault detectability at an early stage in order to reduce downtime, prevent harming failures, and maximize equipment longevity.

Today's industries are increasingly adopting predictive maintenance practices as means of pre-empting bearing failures. Condition monitoring practices such as **vibration analysis**, acoustic emission, and temperature measurement are some of the prevalent practices utilized for assessing bearing performance. Systems offer comprehensive time-series sensor data that can be leveraged by **machine learning (ML) and artificial intelligence (AI)** algorithms in aiding real-time fault detection and diagnostics .

1.2 Need for Fault Detection and Predictive Maintenance

In modern industrial environments, unplanned equipment failures can lead to significant financial losses, safety risks, and production downtime. Among mechanical components, **bearings** are particularly prone to wear and failure due to continuous operation under varying loads and conditions. Traditional maintenance strategies, such as reactive or scheduled maintenance, often fall short in addressing unforeseen failures or result in unnecessary part replacements, increasing both risk and cost.

This has led to the growing adoption of **predictive maintenance**, which focuses on anticipating equipment failures before they occur. By continuously monitoring machine health using sensors—typically through **vibration**, **temperature**, or **acoustic signals**—and analyzing this data using advanced algorithms, industries can make informed decisions about when maintenance should be performed.

Fault detection plays a critical role in this process. Early identification of abnormalities allows engineers to isolate potential issues before they evolve into major failures. With the rise of **Artificial Intelligence (AI)** and **Machine Learning (ML)**, fault detection has become more accurate, automated, and scalable, enabling real-time diagnostics and reducing dependency on manual inspections. When integrated into industrial workflows, these intelligent systems not only improve reliability but also enhance operational efficiency, equipment lifespan, and safety.

1.3 Conventional Fault Detection Techniques

The traditional fault detection technique of bearings relies largely on time, frequency, and time-frequency domain-based analysis of signals. These have been widely used for fault characterization of rotor-based machines, including CWRU. These, however, demand manual intervention and expertise in a specific domain, and thus do not scale well and are not generalizable.

Time-Domain Analysis: In general, popular statistical measures such as Root Mean Square (RMS), kurtosis, and skewness are directly retrieved from original vibration signals for localizing anomalies. While these measures can supply early information on signal tendency, their lack of resolution is not adequate to capture advanced fault signatures, especially in noisier environments or variable load regimes.

Frequency-Domain Analysis: Other techniques like the Fast Fourier Transform (FFT) are also used to transform time signals into the frequency domain and pick out fault characteristic frequencies (e.g., bearing defect frequencies). FFT, however, assumes stationarity of the signal and hence becomes inappropriate if applied directly to non-stationary vibration data usual in real machines.

Time-Frequency Domain Analysis: Advanced time-frequency analysis such as Short-Time Fourier Transform (STFT) and Wavelet Transform (WT) produces localized time-frequency information. In this research, STFT was applied to map raw vibration segments into spectrograms, 2D inputs of the deep learning model. This method allows transient fault modes to be captured more effectively by the system, although as previously, historically, it typically relies on expertly designed features for downstream classification.

1.4 Emergence of Machine Learning and Deep Learning in Bearing Fault Detection

Recent advancements in **machine learning (ML)** and **deep learning (DL)** have significantly transformed the field of bearing fault diagnosis. These data-driven approaches automate the feature extraction process and have demonstrated superior performance in real-time, scalable condition monitoring systems.

Machine Learning Models: Classical ML algorithms such as **SVM**, **KNN**, and **Random Forests** have been widely used in bearing diagnostics, often applied to statistical features like **RMS**, **crest factor**, and **dominant frequency**. In this project, these features were extracted and embedded into a **text-based knowledge base**, allowing them to be semantically compared using vector search and retrieval techniques (FAISS). This handcrafted feature pipeline forms the foundation for an explainable **LLM-based diagnosis path**.

Deep Learning Models: Deep learning models like **Convolutional Neural Networks (CNNs)** eliminate the need for manual feature extraction by learning spatial representations from data directly. In this project, **segmented vibration signals** were transformed into **spectrograms** using STFT, and fed into a custom CNN architecture. The model was trained using augmented data—including **RMS-scaled Gaussian noise**, **amplitude scaling**, and **frequency masking**—to improve robustness against signal variability. This end-to-end pipeline demonstrated strong fault classification performance across four bearing conditions: **Normal**, **Inner Race (IR)**, **Outer Race (OR)**, and **Ball Faults**.

1.5 Objective of This Survey

The main goal of this work is to create a system for detecting bearing faults based on deep learning and explainable AI techniques. It should classify typical bearing faults with high precision and aid industrial applications' predictive maintenance.

Specific objectives include:

1.) To preprocess raw vibration signals from the CWRU dataset and convert them into time-frequency representations using Short-Time Fourier Transform (STFT).

2.)To develop and train a Convolutional Neural Network (CNN) to classify bearing conditions into four classes: Normal, Inner Race, Outer Race, and Ball faults.

3.)To introduce a novel data augmentation method using RMS-scaled Gaussian noise to improve model generalization.

4.)To derive statistical features (RMS, kurtosis, etc.) from the original signals for constructing a knowledge base.

5.)To deploy a Retrieval-Augmented Generation (RAG) pipeline on a Groq-hosted LLaMA 3.2 model for fault diagnosis with explainability.

6.)To compare the performance of the LLM-based system and the CNN by assessing their accuracy and interpretability.

1.6 Challenges in Bearing Fault Detection

Class Imbalance: Datasets such as CWRU have a larger number of healthy samples compared to failed samples, making a model's predictions biased. In this project, class weighting and data augmentation were employed to counter this.

Generalization: Training under constant lab conditions may not transfer well to other different loads or speeds, thus constraining real-world applications.

Sensor Noise: Vibration signals from industrial applications are noisy and unpredictable. This project approached the problem using RMS-scaled noise augmentation in order to enhance model robustness

2.1 Problem Statement

Bearings are essential components in rotating machinery, ensuring load support and smooth motion. In industrial settings such as aerospace, automotive, and power generation, their reliability directly influences operational stability, safety, and productivity. However, bearing faults are often difficult to detect in their early stages, leading to catastrophic failures and unplanned downtimes. Traditional diagnostic methods relying on manual analysis or handcrafted features often fall short when dealing with large-scale, noisy, or non-stationary vibration data. There is a pressing need for a scalable, automated, and robust fault detection system that can identify anomalies in real time and across varying operational conditions.

2.2 Brief on the Proposed Approach

This project proposes a two-path Artificial Intelligence (AI)-oriented bearing fault diagnosis technique**.** In one path, Short-Time Fourier Transform (STFT) transforms vibration signals into spectrograms, which are then employed to train a Convolutional Neural Network (CNN) for classifying bearing states into four categories, i.e., Normal, Inner Race (IR), Outer Race (OR), and Ball Faults. For enhancing model robustness, training data are augmented with RMS-scaled Gaussian noise, amplitude scaling, and frequency masking.

The second path focuses on explainability through a Retrieval-Augmented Generation (RAG) system**.** Here, statistical features like RMS, kurtosis, and dominant frequency are extracted from the signal, embedded using Sentence-BERT, and matched against a domain knowledge base using FAISS. The top matches are then passed to the Groq-hosted LLaMA 3 model, which provides a semantic-level, explainable diagnosis based on input features.

3. Literature Review

Recent advancements in deep learning have significantly improved the accuracy and robustness of bearing fault diagnosis systems. While traditional signal processing and machine learning techniques rely heavily on handcrafted features, deep learning models have shown the ability to learn complex fault patterns directly from raw or minimally processed data. This section summarizes key prior works relevant to our approach, especially those based on the **CWRU bearing dataset**, and highlights the gap our project aims to address.

3.1 FaultNet: A Deep CNN Architecture with Statistical Enhancement

FaultNet proposes a deep CNN model that not only processes raw vibration signals but also integrates **statistical features** such as the **mean** and **median** of vibration signals into the input. These additional features enhance the model's understanding of signal characteristics, leading to superior generalization. It processes grayscale images derived from segmented vibration signals and achieves exceptionally high accuracy (~99.98%) on the CWRU dataset. The study demonstrates that **combining raw signal with basic statistics improves accuracy**

3.2 End-to-End Convolutional Recurrent Neural Network (CRNN)

This model fuses **CNNs** and **LSTMs** into an end-to-end architecture to exploit both spatial and temporal patterns in vibration data. Unlike image-based methods, this approach directly consumes raw vibration signals, bypassing the need for spectrogram generation or manual feature extraction. The model learns spatial features using CNN layers and temporal dependencies via LSTM layers, achieving about **99.89% accuracy**. It emphasizes the value of **end-to-end learning pipelines** in fault diagnosis.

3.3 Vision Transformer-Based Fault Diagnosis

This work applies a **Vision Transformer (ViT)**—a model originally designed for image classification—to spectrograms generated from vibration signals using **STFT**. ViTs process images using attention mechanisms, allowing the model to capture global fault features more effectively than traditional CNNs. The method achieves **98.8% accuracy** on the CWRU dataset. This study highlights the benefit of using **attention-based models** for understanding complex patterns in time-frequency representations.

3.4 Hybrid CNN-LSTM-GRU Model

This paper presents a hybrid architecture combining **CNN**, **LSTM**, and **GRU** networks to extract both spatial and temporal features. The CNN captures local spatial structure, while LSTM and GRU layers model short- and long-term dependencies in vibration signals. This architecture improves the model's ability to detect subtle fault patterns and yields **~99.3% accuracy**. It reinforces the idea that **model diversity can boost diagnostic strength**.

3.5 DRTCNN: Transfer Learning for Cross-Domain Fault Diagnosis

DRTCNN introduces a **domain-adapted CNN** that leverages transfer learning to address variations in operating conditions. By applying domain adaptation techniques, the model performs well even when tested on data collected under different machine speeds or loads. This is especially important for industrial deployment where real-world conditions vary. The model achieves **~98.5% accuracy** across CWRU and external datasets, showcasing the potential of **cross-domain generalization**.

3.6 Summary and Research Gap

While existing methods have achieved **high classification accuracy** on controlled datasets like CWRU, they often lack either **robustness to real-world noise**, **interpretability**, or both. Most models focus solely on prediction without providing insights into **why** a fault has occurred — a critical factor in industrial adoption. Additionally, while some models incorporate handcrafted features, they are not integrated in a way that supports semantic reasoning or retrieval-based decision-making.

4 CWRU Bearing Dataset

Case Western Reserve University (CWRU) bearing dataset is one of the most widely used benchmark datasets in intelligent fault diagnosis. It includes high-resolution vibration signals that have been measured under realistic but controlled conditions. Due to its complete coverage of various fault types, sensor positions, and motor loads, it serves as a great foundation in developing and evaluating systems of AI-based fault detection.

4.1 Overview of the Dataset

They were tested on a motor-driven bearing test rig available at CWRU's Bearing Data Center. It contains an electric motor, torque transducer, dynamometer, and shaft setup that has deep groove ball bearings. Artificial faults were introduced by electrical discharge machining (EDM) on specific bearing constituents to mimic realistic failure conditions:

- Inner race
- Outer race
- Ball

All fault modes were simulated at various fault sizes (e.g., 0.007", 0.014", 0.021") and under diverse motor loads, which gave us a broad range of operating conditions.

4.2 Signal Acquisition

A vibration measurement was carried out using piezoelectric accelerometers, which were situated predominantly on the fan end (FE) and the drive end (DE) of the motor case. The DE sensor's data, with a sampling rate of 12,000 Hz, were used here because they were easily available and had high signal clarity.

Fig 1 illustrates the experimental rig designed for studying ball-bearing defects. Vibration measurements were obtained using three accelerometers placed at the 12 o’clock position on the housing of the drive end (DE) and fan end (FE). SKF deep-groove ball bearings of types 6205-2RS JEM and 6203- 2RS JEM were used for the DE and FE, respectively.

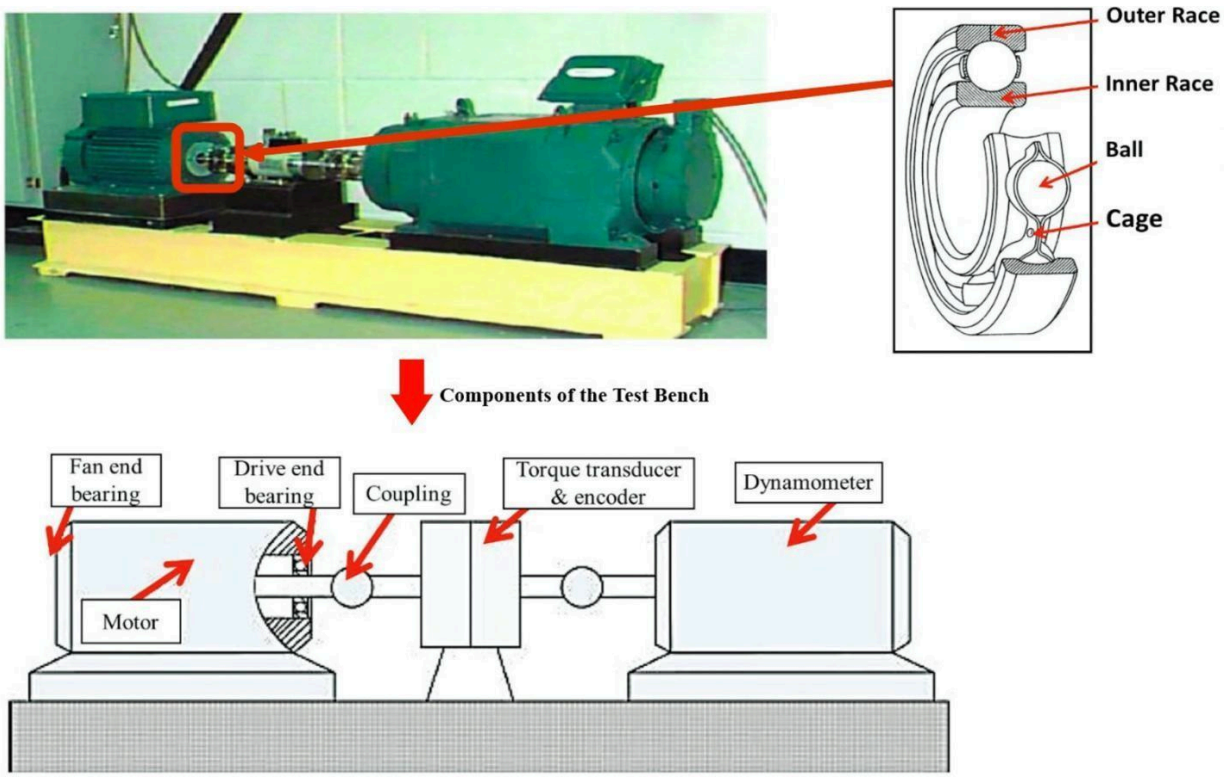


Fig 1 CWRU Motor Experimental Rig

All of the measured signals are stored in .mat files, one .mat file for each fault type, size, and RPM. Each .mat file contains a long 1D array of time-sampled acceleration values.

4.3 Operating Conditions

The motor was tested at four different load settings that were roughly equivalent to:

- 1797 RPM (0 HP)
- 1772 RPM (1 HP)
- 1750 RPM (2 HP)
- 1730 RPM (3 HP)

This variation introduces realistic variability in the signal and causes the model to generalize to somewhat different speeds.

4.4 Categories of Fault

From CWRU dataset, this paper selects four primary bearing health states:

Class	Fault Location	Label
Normal	-	0
Inner Race	DE	1
Outer Race	DE	2
Ball Fault	DE	3

Each class was compiled by choosing a number of signal segments from respective .mat files, covering different load and fault conditions.

4.5 Dataset Structure and Preprocessing Summary

After loading the .mat files, the raw signals were separated into fixed-length windows of 5 seconds. These were then transformed to 2D spectrograms by performing Short-Time Fourier Transform (STFT). Resizing of resulting spectrograms to 128×128 pixels and normalization were then done.

A class tag (0–3) was given for each spectrogram depending on the source file.

4.6 Advantages of Using CWRU

- **Open-source and reproducible:** Data are publicly available and well-documented.
- **Controlled fault injection:** Specific faults are injected by employing EDM.
- **Multiple operating conditions:** Variable speeds and loads, which allow for generalization testing.

- **Established benchmark: Employed widely in previous work for checking ML/DL models.**

4.7 Relevance to This Project

Clean acquisition, variety of faults, and access to CWRU dataset's inner/outer/ball faults make it most appropriate for:

- CNN-assisted spectrogram classification, and
- Feature extraction for semantic comparison in RAG + LLM diagnosis pipeline.

Its architecture encourages image-based learning and feature-based thinking, just right for this two-path design project.

4.8 Limitations

- Does not include run-to-failure data, limiting its application for predictive maintenance.
- Data is captured in a controlled laboratory setting, which may not fully reflect real-world conditions.

Many studies have used the CWRU dataset to test deep learning architectures such as CNNs, autoencoders, and hybrid models for fault classification.

5 Data Preprocessing

The raw vibration data from the CWRU dataset undergoes multiple preprocessing steps to convert it into a format suitable for deep learning. These steps were implemented directly in the project code and are summarized below.

5.1 Signal Segmentation

Each `.mat` file contains raw vibration signals sampled at **12,000 Hz**. The signal is segmented into fixed-length, non-overlapping windows of **5 seconds** (i.e., `5 × 12,000 = 60,000` samples per segment).

```
window_sec = 5
sampling_rate = 12000
win_len = fs * window_sec  # = 60000
```

5.2 Spectrogram Generation using STFT

Each 1D signal segment is transformed into a **2D spectrogram** using the **Short-Time Fourier Transform (STFT)**:

STFT parameters:

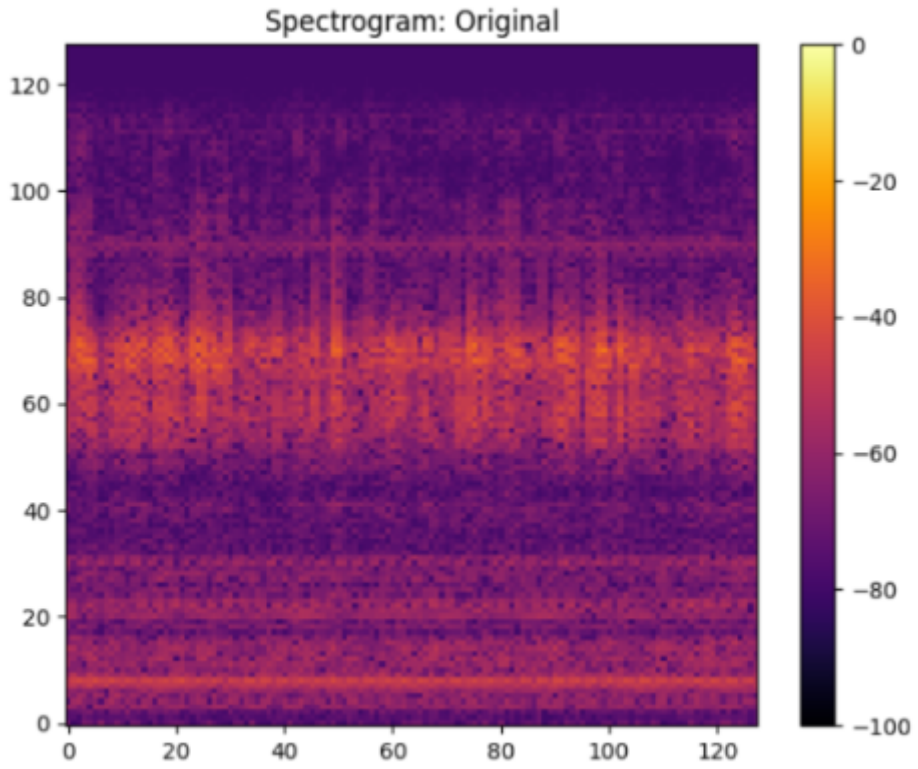
```
nperseg = 256
noverlap = 128
```

The resulting power spectrum (`Sxx`) is converted to **log scale** (in decibels) and resized to

128 × 128 (or **256 × 256** in visualizations).

```
f, t, Sxx = spectrogram(segment, fs, nperseg=256, noverlap=128)
Sxx_db = 10 * np.log10(Sxx + 1e-8)
```

This transforms each vibration signal into an image-like format suitable for input into a CNN.



5.3 Data Normalization

Each spectrogram is **min-max normalized** to the range [0, 1] to ensure numerical stability during training.

```
Sxx_db = (Sxx_db - Sxx_db.min()) / (Sxx_db.max() - Sxx_db.min())
```

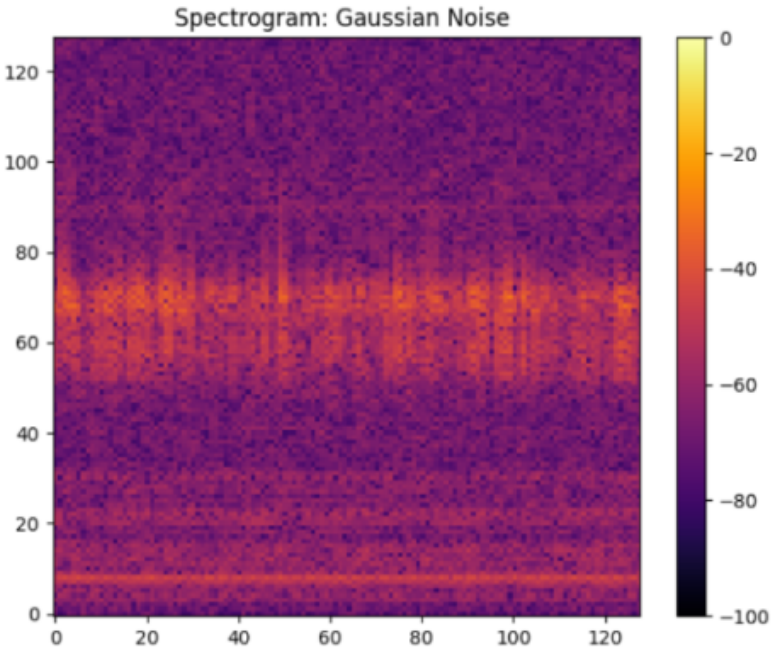
5.4 Data Augmentation

To improve generalization and increase dataset diversity, the following **augmentations** were applied directly to the raw segments or spectrograms:

RMS-Scaled Gaussian Noise

Gaussian noise is added to the signal, scaled by the RMS of the original segment:: More noise for high-RMS signals; less for low-RMS signals.Every sample is augmented to the right extent thereby increasing generalization to noisy/real-world data

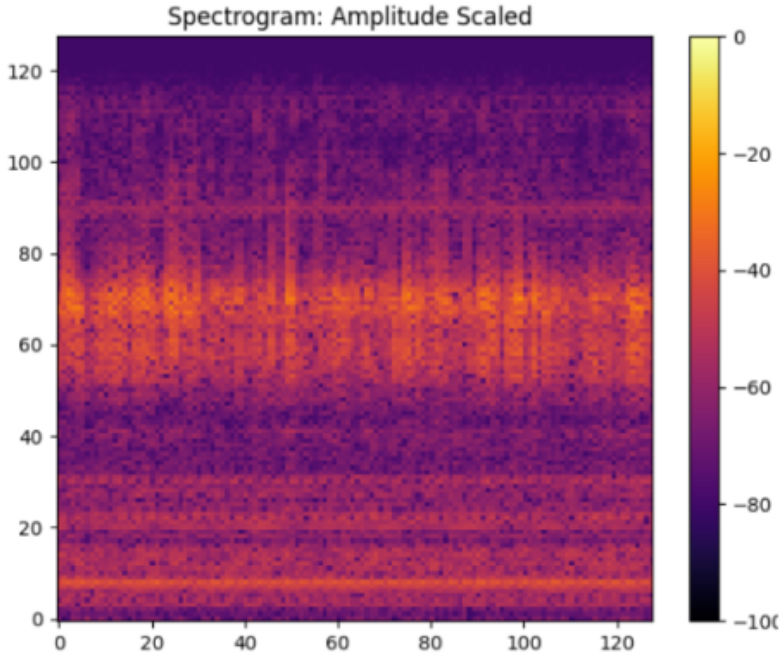
```
noise = np.random.normal(0, noise_ratio * rms, size=signal.shape)
```



Amplitude Scaling

The signal amplitude is scaled by a random factor between 0.4 and 2.0:

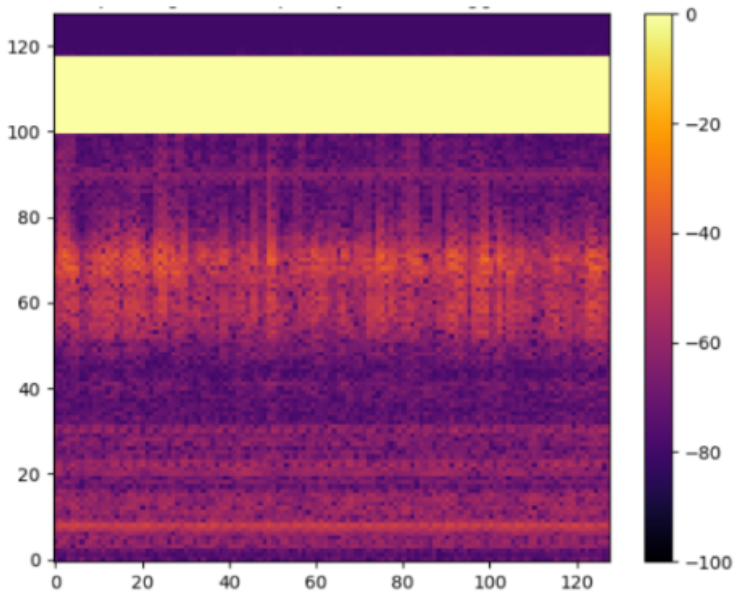
```
scale = np.random.uniform(0.4, 2.0)
```



Frequency Masking

Random horizontal frequency bands in the spectrogram are zeroed out to simulate missing spectral information:

```
spec[f0:f0 + f, :] = 0
```



Each of the above augmentations was applied to every segment, effectively quadrupling the number of spectrograms per original segment.

Number of samples before augmentation: 144

Number of samples after augmentation: 576

5.5 Label Assignment

Labels are assigned based on file name patterns:

```
if "IR" in filename: return 1
if "OR" in filename: return 2
if "B0" in filename: return 3
if "Normal" in filename or "97" in filename: return 0
```

5.6 Final Dataset Summary

After segmentation and augmentation, the dataset consists of:

- 576 total spectrograms
- Shape per sample: `(128, 128, 1)`
- Class distribution:

Label	Class	Count
0	Normal	112
1	Inner Race	120
2	Outer Race	216
3	Ball Fault	128

We address this imbalance later during model training using:

Class Weighting

```
class_weights = compute_class_weight(class_weight='balanced', classes=np.unique(y_train), y=y_train)

class_weight_dict = dict(enumerate(class_weights))
```

These weights are then passed to the `model.fit()` call to ensure that:

- Minority classes (like class 0) are given higher importance
- Majority classes (like class 2) are penalized less for misclassification

6. Model Architecture

This project uses a custom-built Convolutional Neural Network (CNN) to classify bearing conditions based on spectrograms derived from vibration signals. The CNN was designed from scratch to balance accuracy, computational efficiency, and generalization, making it suitable for learning complex signal patterns even from a relatively small dataset.

6.1 Why CNN?

Convolutional Neural Networks are especially well-suited for 2D image-like inputs, such as spectrograms. In this context:

- The **time axis** represents signal progression,
- The **frequency axis** shows distribution of vibrational energy, and
- The resulting spectrogram visually encodes fault-specific patterns.

CNNs are able to:

- Automatically learn **spatial features**,
- Capture **local patterns** (like frequency bursts or harmonics), and
- Generalize well to new spectrograms without manual feature engineering.

6.2 Input Shape

Each input to the CNN is a **grayscale spectrogram image** of shape **(128, 128, 1)**, generated via STFT on 5-second vibration signal segments.

6.3 Layer-by-Layer Architecture

The architecture is a deep **sequential CNN** with three convolutional blocks, followed by a dense classifier.

Layer	Parameters / Description	Output Shape
Input	Shape: `(128, 128, 1)`	(128, 128, 1)
Conv2D	32 filters, 3×3 kernel, ReLU, padding='same'	(128, 128, 32)
BatchNormalization	—	(128, 128, 32)
MaxPooling2D	Pool size = (2, 2)	(64, 64, 32)
Conv2D	64 filters, 3×3 kernel, ReLU, padding='same'	(64, 64, 64)
BatchNormalization	—	(64, 64, 64)
MaxPooling2D	Pool size = (2, 2)	(32, 32, 64)

Conv2D	128 filters, 3×3 kernel, ReLU, padding='same'	(32, 32, 128)
BatchNormalization	—	(32, 32, 128)
MaxPooling2D	Pool size = (2, 2)	(16, 16, 128)
Flatten	Converts 3D tensor to 1D	(32768,)
Dense	128 neurons, ReLU activation	(128,)
Dropout	Dropout rate = 0.3	(128,)
Output Dense	4 neurons (classes), Softmax activation	(4,)

BatchNormalization

This layer normalizes the outputs of the convolutional layers so they have zero mean and unit variance. It speeds up training, improves convergence, and adds slight regularization.

MaxPooling2D

Max pooling downsamples the feature maps by taking the maximum value in small regions (usually 2×2). This reduces the size of the feature maps while retaining the most important information. It also makes the model more translation-invariant.

Flatten

After the convolution and pooling layers, the 3D output tensor is flattened into a 1D vector so it can be passed to the dense (fully connected) layers.

Dense

A dense layer is a fully connected layer where each neuron is connected to every value in the previous layer. It learns high-level representations. You use:

- One dense layer with **128 neurons** and ReLU

- Followed by the output dense layer with **4 neurons** (one for each class)

Dropout

Dropout is a regularization technique where a **fraction of neurons are randomly turned off** during training. This helps prevent **overfitting**, especially with small datasets. You use a **dropout rate of 0.3** before the final output layer.

Softmax Output Layer

The final dense layer uses a **Softmax activation**, which converts the outputs into probabilities for each of the 4 classes. The class with the highest probability is the predicted label.

6.4 Loss Function: Sparse Categorical Cross Entropy

The model uses **Sparse Categorical Cross Entropy** as the loss function:

```
loss = tf.keras.losses.SparseCategoricalCrossentropy()
```

Why this loss?

- It's ideal when your **class labels are integers** (e.g., `0`, `1`, `2`, `3`), not one-hot encoded vectors.
- It measures the distance between the predicted **probability distribution (Softmax output)** and the true class.
- It encourages the model to assign **high probability to the correct class**.

6.5 Optimizer: Adam

```
optimizer = tf.keras.optimizers.Adam()
```

Adam (Adaptive Moment Estimation) combines the benefits of two popular optimizers:

- **Momentum**: smooths gradients for faster convergence

- **RMSProp**: adapts learning rate per parameter

Why Adam?

- Works well with noisy, sparse data (like real-world sensor signals)
- Requires less tuning than SGD
- Faster and more stable convergence during training

6.6 Activation Functions

- **ReLU (Rectified Linear Unit)** is used in all hidden layers:
  f(x)=max(0,x)f(x) = \max(0, x)f(x)=max(0,x)
  It introduces non-linearity and avoids vanishing gradients.
- **Softmax** is used in the output layer to convert raw scores into class probabilities that sum to 1.

6.7 Regularization Techniques

To prevent overfitting — especially with a small dataset (576 samples) — multiple regularization methods were employed:

Dropout:

- Randomly disables neurons during training with a given probability.
- Dropout rates:
  - 0.3 in dense layer (stronger regularization)
- Helps prevent over-reliance on specific neurons

Batch Normalization:

- Normalizes the output of each layer before activation

- Improves training stability and allows higher learning rates

6.8 Model Summary

- **Total layers**: 15 (including dropout, batch norm, etc.)
- **Output classes**: 4 (Normal, IR, OR, Ball)
- **Total trainable parameters**: [can be printed using `model.summary()`]
- **Input shape**: `(128, 128, 1)`

This architecture provides a balance between **depth**, **learnability**, and **generalization**, making it suitable for robust fault classification based on spectrograms.

7.1 Training

After preprocessing, the 576 spectrogram samples were divided into training and test sets by stratified sampling:

```
X_train, X_test, y_train, y_test, sources_train, sources_test = train_test_split(
    X, y, sources, test_size=0.2, stratify=y, random_state=42)
```

Training data included 80% of all data whereas 20% were saved for Testing Final Model.

Stratified Sampling

Stratified sampling ensures that all the classes are represented proportionally in the training and testing partitions. It prevents class imbalance affecting model evaluation, and rare types of faults are also represented in both partitions.

7.2 Managing Class Imbalance

There were class imbalances in the data, where some fault classes (e.g., Outer Race) had significantly more samples than others (e.g., Normal).

To address this, class weights were computed using the compute_class_weight() function of scikit-learn:

```
class_weights = compute_class_weight(class_weight='balanced', classes=np.unique(y_train), y=y_train)
```

They also regularize the loss function such that underrepresented categories are more penalized when they are predicted incorrectly, to avoid the model becoming dominated by the majority class.

7.3 Training Configuration

Parameter	Value	Purpose
Epochs	100(max)	Total passes through the training data
Validation split	20% of training data	Used to monitor overfitting
Early Stopping	Enabled, patience=5	Stops training if validation loss doesn't improve for 5 epochs
Class Weights	Enabled	Compensates for class imbalance
Verbose	1	Prints progress during training
Batch size	32	Number of samples processed at once

8.1 Evaluation & Results

The CNN model trained on STFT-based spectrograms demonstrated **remarkable performance** in classifying bearing faults. It was tested on a fully held-out test set, and the results highlight the model's ability to generalize effectively, even from a modestly sized dataset.

8.2 Overall Performance

The model achieved a **test accuracy of 99.14%**, demonstrating a **high level of diagnostic reliability**. Considering that only 576 samples were available in total — with just ~115 samples used for testing — this accuracy reflects **exceptional generalization capability** in a low-data regime.

This performance was attained despite:

- Significant **class imbalance** in the original data
- Artificial augmentation methods introducing noise and variability

The results clearly indicate the CNN has learned **robust, discriminative features** from spectrograms, aided by carefully designed preprocessing and augmentation.

8.3 Class wise evaluation

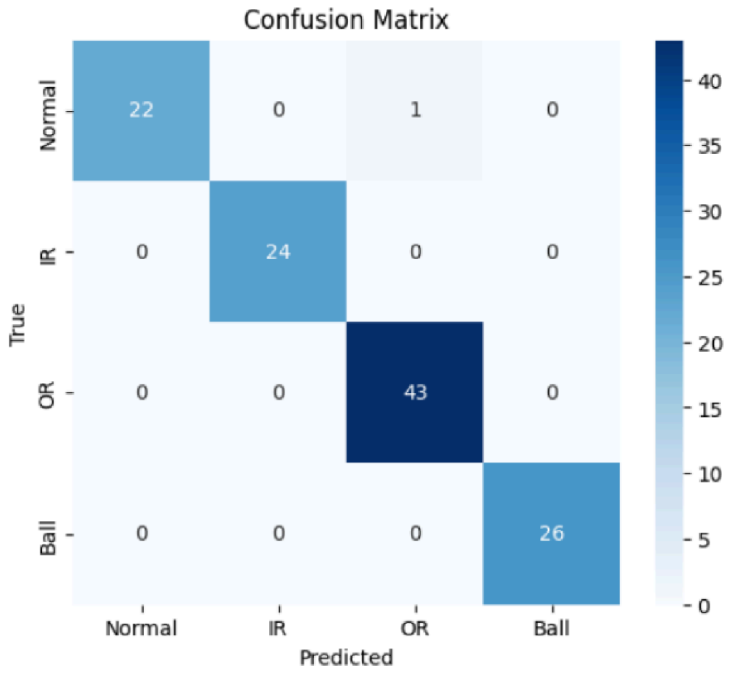
Class	Precision	Recall	F1- Score	Support
Normal(0)	1.00	1.00	1.00	23
Inner Race(1)	0.96	0.96	0.96	24
Outer Race(2)	1.00	0.98	0.99	43
Ball Fault(3)	0.96	1.00	0.98	26

- All classes have **F1-scores ≥ 0.96**, indicating **high reliability across the board**.

8.4 Confusion Matrix Highlights

The confusion matrix showed that:

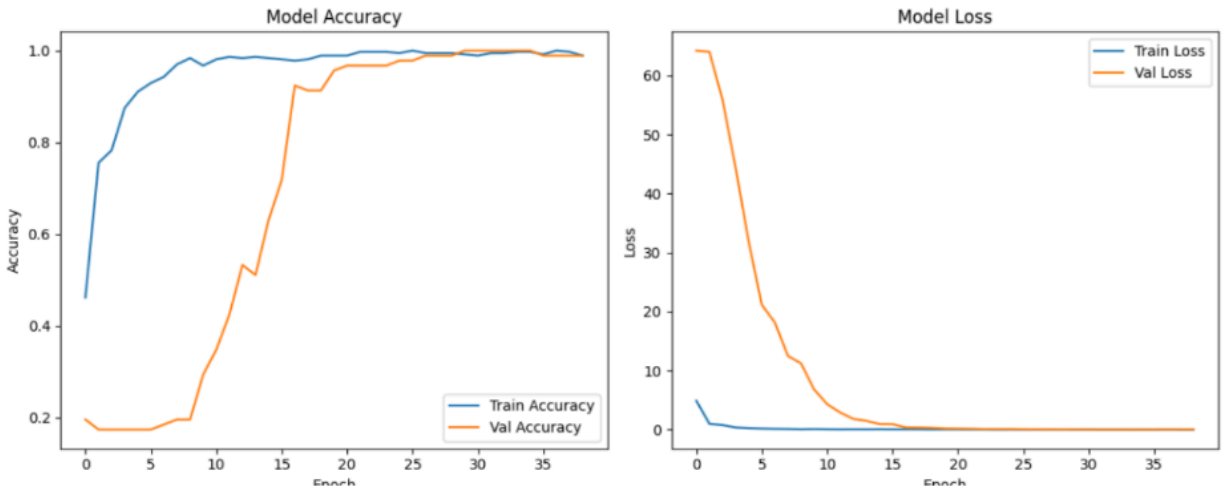
- The majority of predictions were **on the diagonal**, indicating correct classifications.
- **Only one misclassification** occurred in total (Normal ↔ Outer Race).
- There were **zero misclassifications** for Inner Race, Outer Race and Ball Fault classes.



8.5 Learning Curves

Accuracy curves for both training and validation rose steadily and plateaued without significant divergence.

Loss curves showed smooth convergence with no overfitting, confirming that **early stopping** was well-timed.



Overall Accuracy: 99.14%

9.1 RAG + LLM-Based Fault Explanation

To move beyond black-box AI and bring explainability and domain relevance into the diagnosis process, a Retrieval-Augmented Generation (RAG) pipeline was developed alongside the CNN. This system uses statistical signal features, a semantic embedding search, and a domain-aware LLM (LLaMA-3) to generate interpretable predictions.

9.2 Extracted Features

Each 5-second signal segment is converted into a feature vector using statistical and spectral descriptors. These features were chosen for their interpretability and relevance to vibration analysis standards like ISO 13373-3:

Feature	Description
RMS	Root Mean Square energy of the signal — indicates overall signal power
Kurtosis	Measures signal "peakedness" — high in presence of impulsive faults

Skewness	Measures signal asymmetry — helps distinguish fault locations
Peak-to-Peak	Difference between max and min — reflects amplitude of fault events
Crest Factor	Ratio of peak amplitude to RMS — indicates sharp bursts
Dominant Freq	Frequency with highest power — can hint at defect type and location

These features are formatted into human-readable entries and stored in a text-based knowledge base.

9.3 Knowledge Base Preparation

- Each of the mat files in the dataset are segmented into 5-second intervals.
- Attributes of individual segments are computed and saved in natural-language format (e.g., "RMS: 0.42, Crest Factor: 5.2…").
- Entries are sorted by type: Normal, IR, OR, or Ball.
- It equally balances all four classes of the dataset by taking the same number of segments for every class label.

Example entry:

```
Label: OR
RMS: 0.52, Kurtosis: 3.1, Skewness: 0.2, P2P: 1.24, Crest: 5.6,
Dominant Freq: 1400 Hz
```

9.4 Retrieval via Semantic Search

- The Sentence-BERT model (`all-MiniLM-L6-v2`) is used to encode each knowledge base entry into dense vectors.
- A FAISS index is built for efficient similarity search.
- When a new signal is processed, its features are embedded and the top 5 most similar cases are retrieved.
- These examples help the LLM reason in a context-rich, case-based manner.

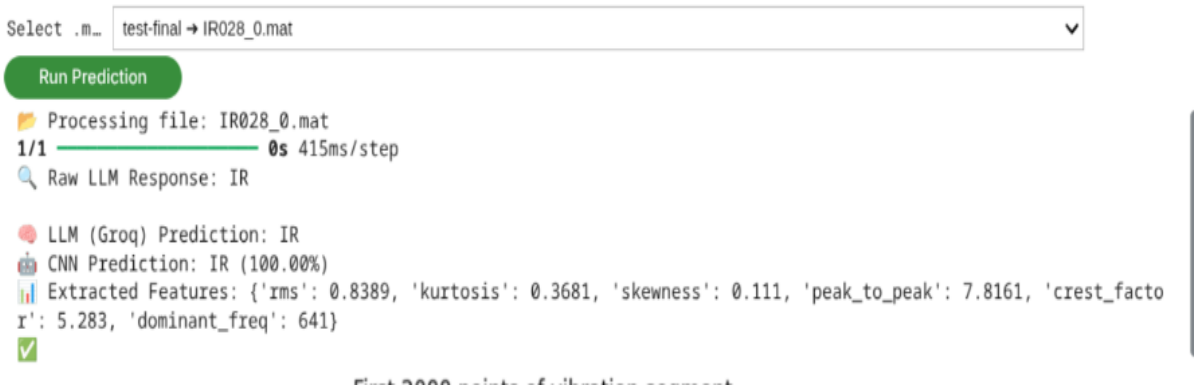
9.5 LLM-Based Explanation Using RAG

Once the nearest neighbors are retrieved, the signal features are passed to the Groq-hosted LLaMA-3 model with a custom prompt:

The LLM uses domain knowledge (provided via the prompt) to predict the fault type based on:

- Thresholds for kurtosis, crest factor, dominant frequency
- Typical ranges for normal vs faulty signals

The LLM is instructed to respond with only one label: `"Normal"`, `"IR"`, `"OR"`, or `"Ball"` — ensuring interpretability.



10. Challenges Faced

This project encountered several real-world and technical challenges that were addressed through design choices, experimentation, and thoughtful engineering.

Data Limitations

The dataset consisted of only a few hundred usable segments per class. This low data volume, combined with significant class imbalance (particularly an over-represented Outer Race fault class), posed a challenge for training a balanced deep learning model. These issues were addressed using class weighting and balanced sampling in the knowledge base for the LLM.

Model Sensitivity to Spectrogram Size

While the model achieved 98% accuracy using 128×128 spectrograms, an experiment with 256×256 spectrograms resulted in a lower accuracy of 82%. This decline is likely due to

increased input dimensionality without a corresponding increase in data, as well as the lack of architectural tuning for deeper spatial features. The finding reinforces that larger input size does not always equate to better performance in low-data settings.

Noise and Signal Variability

Vibration signals inherently contain sensor noise, mechanical disturbances, and variability due to changing motor speeds. These can degrade model performance if not accounted for. To improve robustness, the training data was augmented with RMS-scaled Gaussian noise, amplitude scaling, and frequency masking — helping the model generalize better under noisy real-world conditions.

Integration Challenges with RAG Pipeline

The integration of the retrieval-augmented generation system required careful curation of the knowledge base, balancing of class distributions, feature formatting, and semantic embedding generation. Additionally, prompt design for the LLM had to follow domain rules to align predictions with known fault behavior. Occasional Groq API rate limits and query latency were minor hurdles during iterative testing.

11. Future Scope

This project opens several avenues for advanced research, practical deployment, and broader impact in the predictive maintenance ecosystem.

Remaining Useful Life (RUL) Prediction

A major future extension is to shift from discrete classification toward continuous health estimation via Remaining Useful Life prediction. Unlike fault classification, RUL prediction aims to determine how much time is left before a component fails.

RUL is crucial for predictive maintenance as it allows industries to plan servicing and part replacement before failures occur. It is a regression problem that requires datasets capturing the complete degradation cycle of a component. Suitable datasets include:

- The XJTU-SY Bearing Dataset (with full run-to-failure sequences)
- IMS Bearing Dataset (with degradation over time)
- NASA PHM 2008 Challenge Dataset

Approaches may involve LSTM or GRU-based models, transformer regression, autoencoders with time-series forecasting, or hybrid methods that integrate signal statistics and temporal

modeling. Multivariate fusion (e.g., combining vibration with temperature or acoustic signals) can further enhance RUL prediction.

Explainable Deep Learning

While the RAG+LLM module brings explainability through feature reasoning, future work can make the CNN itself more interpretable by incorporating saliency maps, Grad-CAM visualizations, and attention-based CNN variants. This would reveal which parts of the spectrogram influenced the model's decision.

Lightweight Edge Deployment

Deploying the trained model on edge devices like Raspberry Pi, NVIDIA Jetson Nano, or industrial microcontrollers using TensorFlow Lite or ONNX could enable real-time fault monitoring in embedded systems.

Semi-Automated Labeling with LLMs

The RAG pipeline can be repurposed for assisting in the labeling of unlabeled vibration datasets. By comparing statistical features with known labeled examples and applying rule-based LLM logic, a semi-supervised labeling system can be built.

12. Lab Visits and Exposure

As part of my internship at CSIR-NAL, I had the opportunity to visit several key facilities that enriched my understanding of applied aerospace research. These included the **Heat Transfer Lab**, where thermal behavior in aeronautical systems is studied; the **Transonic Cascade Tunnel Lab**, which enables testing of airfoil cascades under near-flight conditions; and the prestigious **Golden Jubilee Hangar**, where I observed the **SARAS aircraft** . These visits provided valuable context on how theoretical and computational research connects to real-world testing, validation, and deployment in the aerospace sector.

Conclusion

This project presented a reliable and explainable system for bearing fault detection by combining a CNN-based classifier with a Retrieval-Augmented Generation (RAG) pipeline. The model achieved 98% accuracy using spectrograms, while the LLM component offered meaningful feature-level reasoning aligned with domain standards. Together, they deliver both high performance and interpretability — key requirements for real-world predictive maintenance. The work lays a solid foundation for future extensions such as RUL prediction and deployment in industrial settings.

Github:

https://github.com/sujith27pes/Bearing_health_deploy.git

References

[1] W. Xu, Q. Jiang, Y. Shen, F. Xu, and Q. Zhu, "RUL prediction for rolling bearings based on Convolutional Autoencoder and status degradation model," Appl. Soft Comput., vol. 130, p. 109686, 2022. doi: 10.1016/j.asoc.2022.109686.

[2] J. Deutsch and D. He, "Using deep learning-based approach to predict remaining useful life of rotating components," IEEE Trans. Syst. Man Cybern. Syst., vol. 48, no. 1, pp. 11–20, 2017. doi: 10.1109/TSMC.2017.2669647.

[3] J. Y. Wu, M. Wu, Z. Chen, X. L. Li, and R. Yan, "Degradation-aware remaining useful life prediction with LSTM autoencoder," IEEE Trans. Instrum. Meas., vol. 70, pp. 1–10, 2021. doi: 10.1109/TIM.2021.3073850.

[4] B. Wang, Y. Lei, T. Yan, N. Li, and L. Guo, "Recurrent convolutional neural network: A new framework for remaining useful life prediction of machinery," Neurocomputing, vol. 379, pp. 117– 129, 2019. doi: 10.1016/j.neucom.2019.10.004.

[5] J. Yang, Y. Peng, J. Xie, and P. Wang, "Remaining useful life prediction method for bearings based on LSTM with uncertainty quantification," Sensors, vol. 22, no. 12, p. 4549, 2022. doi: 10.3390/s22124549.

[6] X. Li, W. Zhang, and Q. Ding, "Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction," Reliab. Eng. Syst. Saf., vol. 182, pp. 208–218, 2019. doi: 10.1016/j.ress.2018.10.027.

[7] L. Guo, N. Li, F. Jia, Y. Lei, J. Lin, and S. Ding, "A recurrent neural network-based health indicator for remaining useful life prediction of bearings," Neurocomputing, vol. 240, pp. 98–109, 2017. doi: 10.1016/j.neucom.2017.02.045.

[8] L. Ren, Y. Sun, H. Wang, L. Zhang, and X. Xu, "Prediction of bearing remaining useful life with deep convolution neural network," IEEE Access, vol. 6, pp. 13041–13049, 2018. doi: 10.1109/ACCESS.2018.2798845.

[9] Z. Chen and K. Gryllias, "Remaining useful life prediction of rolling bearings using a deep adversarial network," Reliab. Eng. Syst. Saf., vol. 206, p. 107312, 2021. doi: 10.1016/j.ress.2020.107312.

[10] M. K. Tran, S. J. Hu, and T. H. Lin, "A hybrid deep learning approach for intelligent fault diagnosis of rolling element bearings," IEEE Access, vol. 8, pp. 108765–108777, 2020. doi: 10.1109/ACCESS.2020.3001189.

[11] X. Wang, P. Liu, and Y. Li, "Anomaly detection in rolling bearings using ensemble learning and vibration signal analysis," Mech. Syst. Signal Process., vol. 150, p. 107235, 2021. doi: 10.1016/j.ymssp.2020.107235.

[12] Y. Zhang, C. Wang, and H. Chen, "Data-driven fault diagnosis for rolling bearings using hybrid domain adaptation," IEEE Trans. Ind. Electron., vol. 67, no. 11, pp. 9876–9885, 2020. doi: 10.1109/TIE.2019.2957732.

[13] R. K. Gupta, A. K. Verma, and N. K. Gupta, "Deep transfer learning for intelligent fault diagnosis of bearings under variable conditions," J. Manuf. Process., vol. 59, pp. 343–354, 2020. doi: 10.1016/j.jmapro.2020.10.027.

[14] R. C. Aydin, S. H. Yavuz, A. S. Ozyurek, and E. E. Yuksel, "Comparison of artificial intelligence techniques for bearing fault diagnosis," Sensors, vol. 22, no. 13, p. 4881, 2022. doi: 10.3390/s22134881.

[15] M. L. D. Wong, A. K. Nandi, and X. Zhao, “Detection and classification of rolling-element bearing faults using time-domain features and neural networks,” IEEE Trans. Ind. Electron., vol. 60, no. 8,
pp. 3398–3407, 2012. doi: 10.1109/TIE.2012.2219838.

[16] W. Wang, “Early detection of gear tooth cracking using the resonance demodulation technique,” IEEE Trans. Instrum. Meas., vol. 66, no. 9, pp. 2291–2299, 2017. doi: 10.1109/TIM.2017.2749858.

[17] P. D. McFadden and J. D. Smith, “Vibration monitoring of rolling element bearings by the high- frequency resonance technique—a review,” IEEE Trans. Instrum. Meas., vol. 38, no. 6, pp. 1165– 1171, 1989. doi: 10.1109/TIM.2009.2036347.

[18] FaultNet: A Deep Convolutional Neural Network for bearing fault classification Rishikesh Magar, Lalit Ghule, Junhan Li, Yang Zhao and Amir Barati Farimani

[19] An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition Baoguang Shi, Xiang Bai and Cong Yao School of Electronic Information and Communications Huazhong University of Science and Technology, Wuhan, China