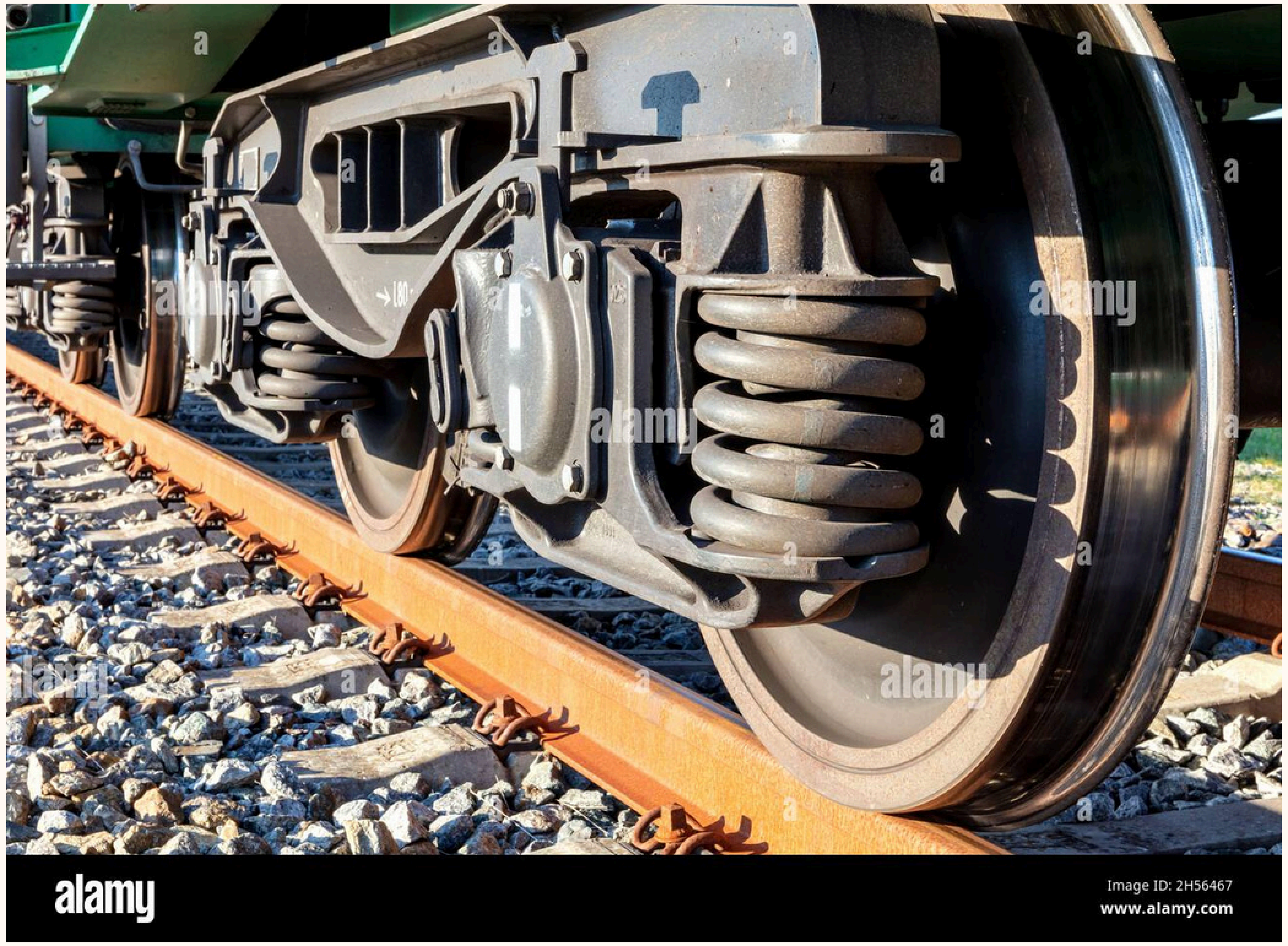
DEEP LEARNING BASED STRUCTURAL HEALTH CONDITION EVALUATION

Presentation by :
Vamshi Krishna Emmadi (21ME10033)

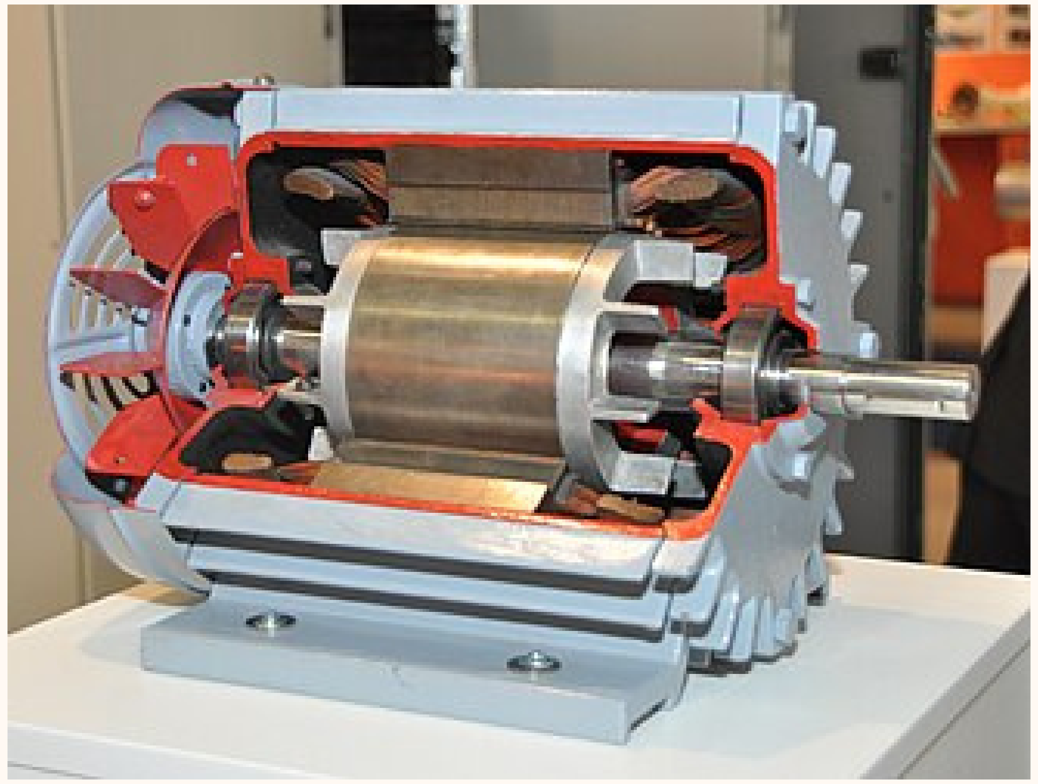
Under the Guidance of:
Prof. Mahesh Mohan MR

PROBLEM STATEMENT & MOTIVATION

- Key Points:
  - Railways are essential for urban and intercity transportation, supporting economic growth and societal mobility in India.
  - Undetected anomalies in rail tracks can lead to catastrophic accidents and costly service disruptions; therefore, a periodic and scheduled health monitoring of railway tracks is necessary.
  - Manual inspections are labor-intensive, time-consuming, and lack real-time responsiveness, so there is a need for smarter solutions. There's a growing demand for automated, accurate, and real-time health monitoring systems.
  - Our proposed solution integrates **vibration analysis, and deep learning to build an intelligent, scalable, and nondestructive rail monitoring system.**



Health condition of Railway track



Fault diagnosis of Induction motor

Damage detection in Mechanical gears

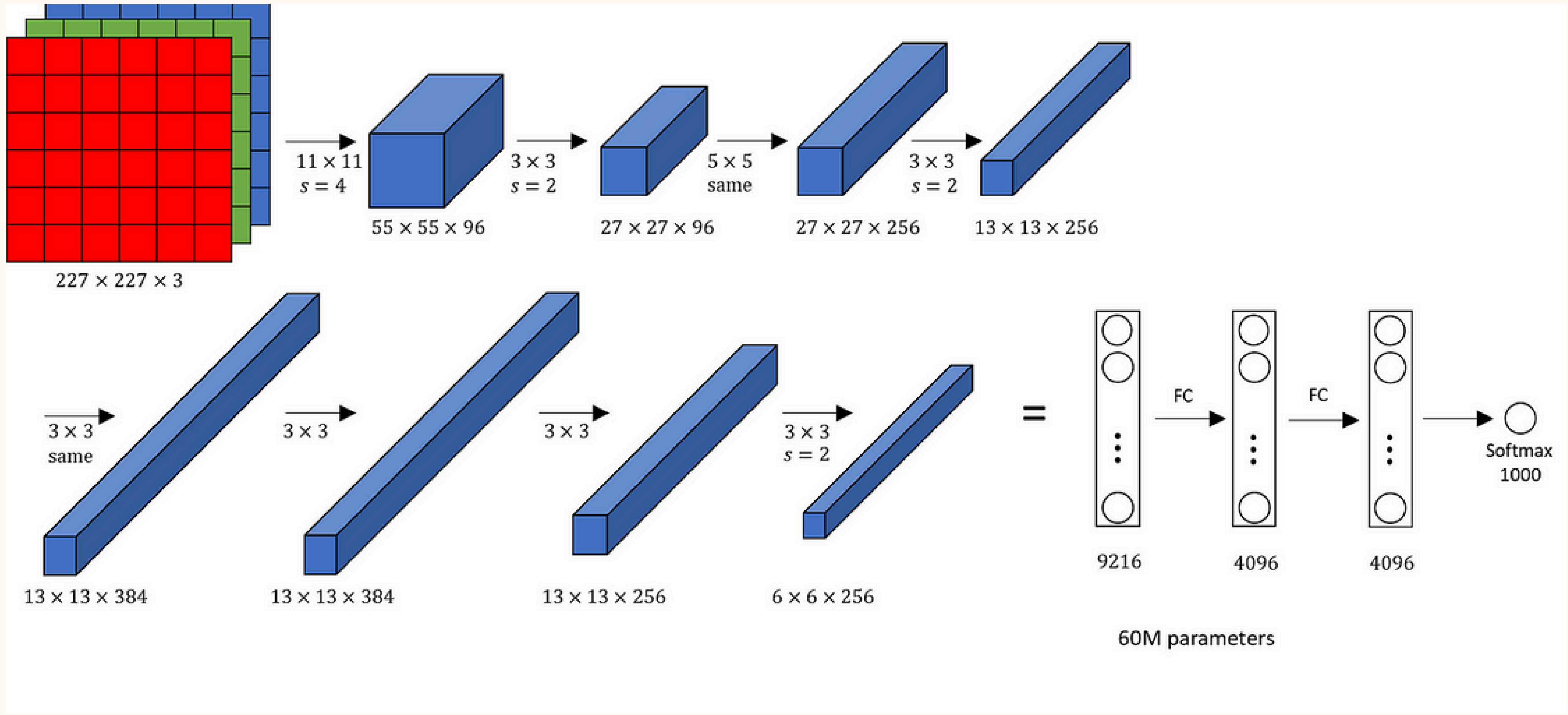
TRADITIONAL METHODS AND THEIR LIMITATIONS:

- Traditional methods include using FAST (Features from Accelerated Segment Test) and SIFT (Scale-Invariant Feature Transform) algorithms, the EMD (Empirical Mode Decomposition) method, and using different transform functions on the vibration signals.
- Although traditional methods work and perform well, there are modern methods to improve the performance of the health monitoring model.
- Integration of AI with mechanical domain knowledge significantly improves the learning and performing capacity of the health monitoring model.
- Based on the literature study, AI can be used for predictive maintenance, anomaly detection, real-time vibration signal analysis, stress detection, and noise filtering.
- So, we are planning to build a machine learning model using mechanical domain knowledge to build a robust, non-destructive health classifier with good accuracy.

DEEP LEARNING ARCHITECTURES EXPLORED:

ALEXNET:

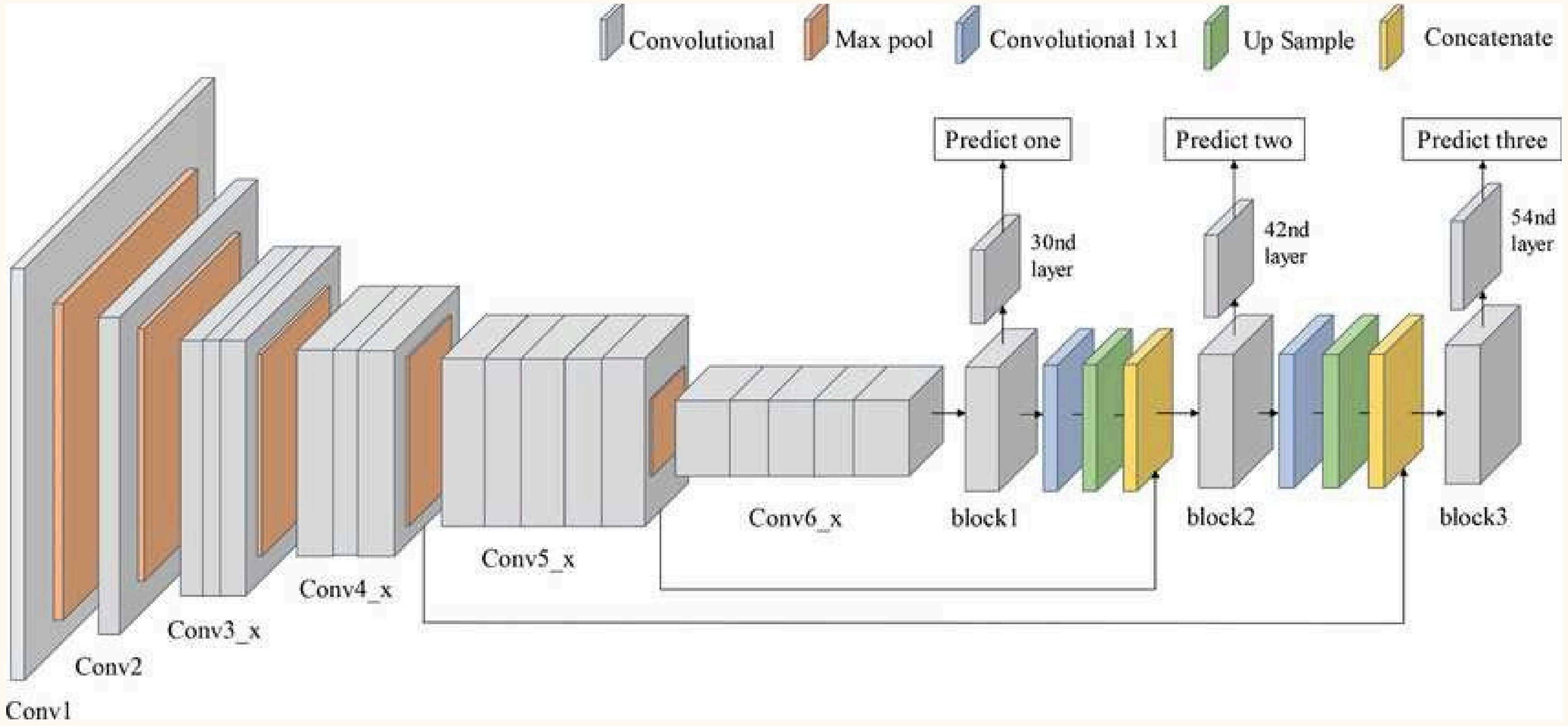
- The custom AlexNet-based Convolutional Neural Network (CNN) was designed and optimized to process grayscale images. .
- It contains Convolutional Layers to extract spatial features, Max-Pooling Layers to Reduce spatial dimensions Fully Connected Layers to Combine high-level features and include dropout to prevent overfitting.



DEEP LEARNING ARCHITECTURES EXPLORED:

DARKNET:

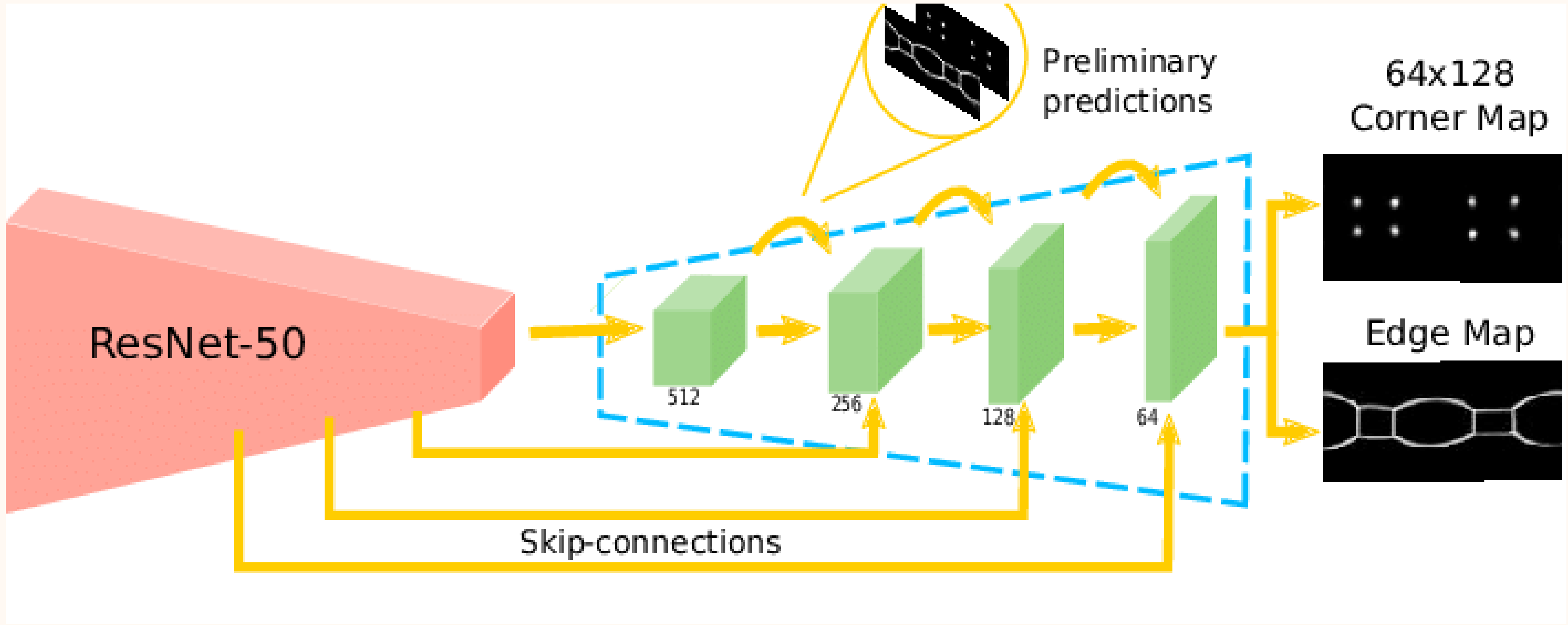
- The custom DarkNet-based Convolutional Neural Network (CNN) was also designed and optimized to process 16×16 grayscale images.
- It also contains Convolutional Layers to extract spatial features and max-Pooling Layers to Reduce spatial dimensions Fully connected layers to combine high-level features and include dropout to prevent overfitting.



DEEP LEARNING ARCHITECTURES EXPLORED:

RESNET:

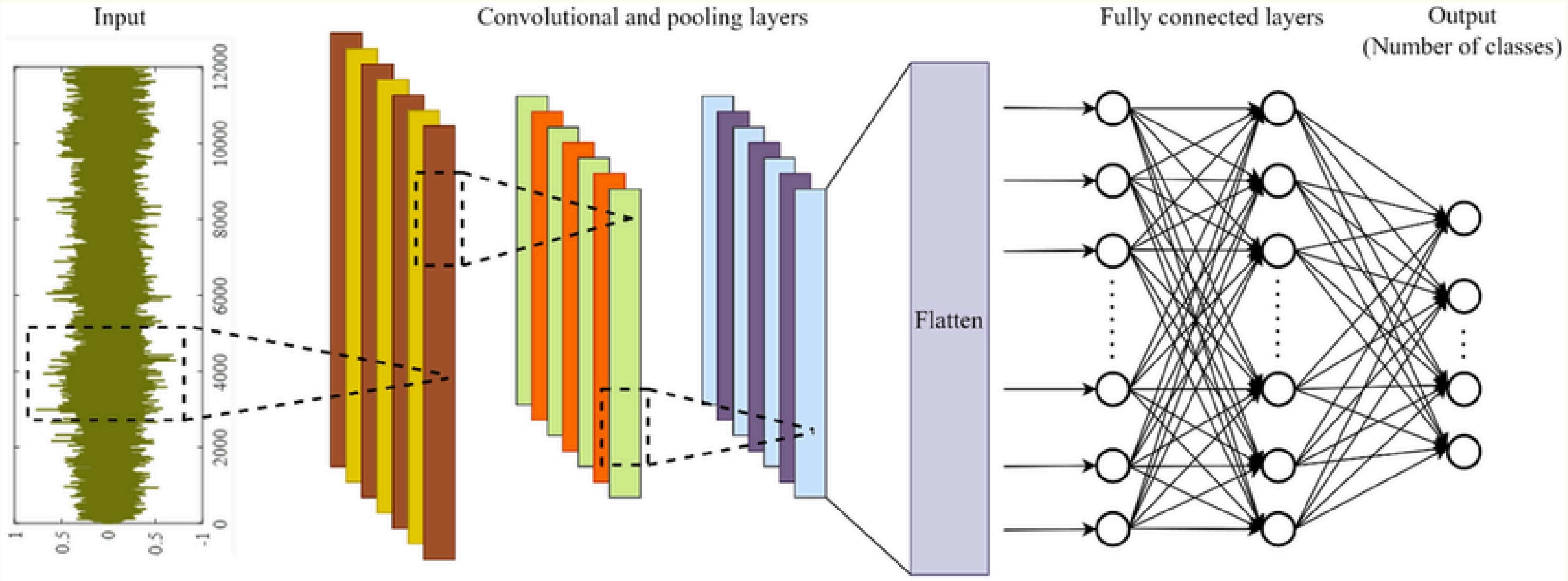
- The custom Resnet-based Convolutional Neural Network (CNN) was also designed and optimized to process grayscale and 3-layered images.
- It also contains Convolutional Layers to extract spatial features and max-Pooling Layers to Reduce spatial dimensions Fully connected layers to combine high-level features and include dropout to prevent overfitting.



DEEP LEARNING ARCHITECTURES EXPLORED:

1D-CNN:

- 1D-Convolutional Neural Network (CNN) was designed and optimized to work on sequential data.
- 1D convolutional layers slide filters over the input to extract local temporal patterns, ReLU activations for non-linearity, max-pooling layers to downsample and reduce dimensionality, dropout layers to prevent overfitting, and fully connected layers that combine extracted features for final decision-making.



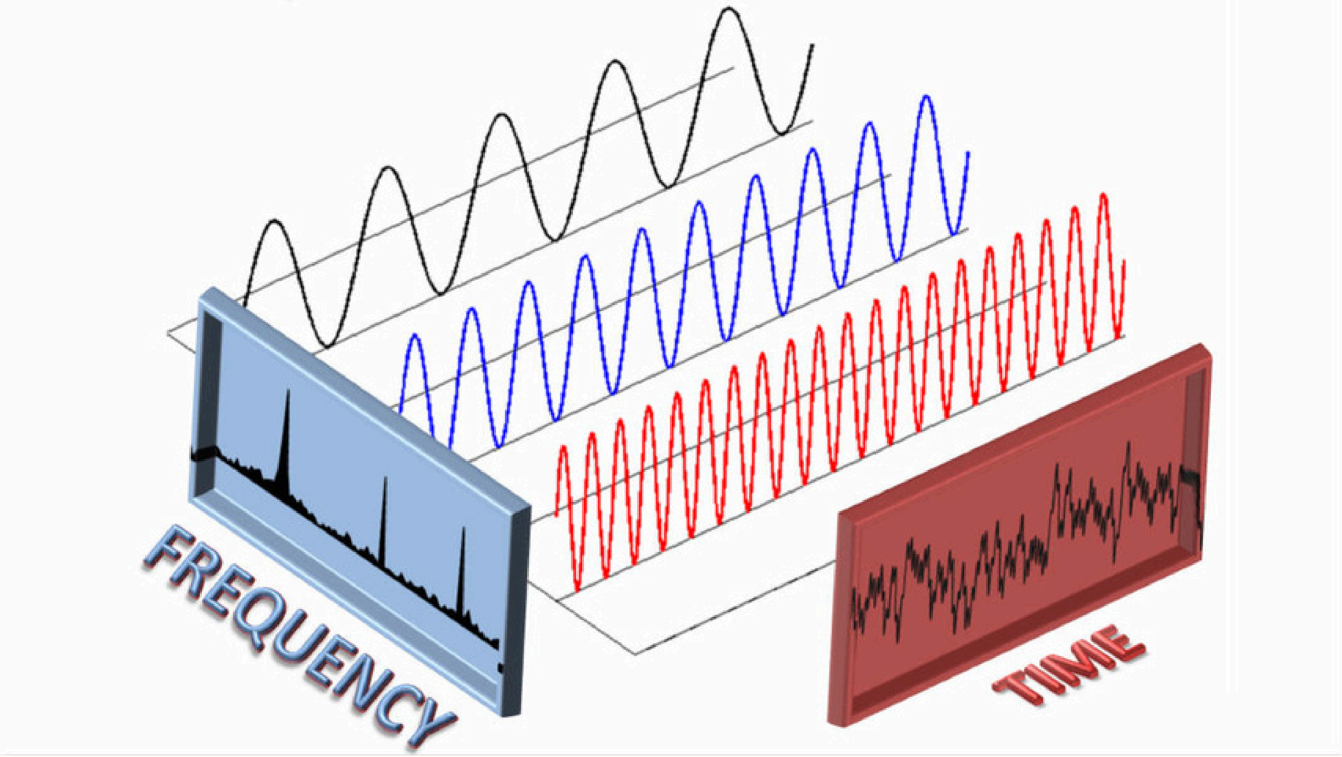
TRANSFORM FUNCTIONS EXPLORED:

FOURIER TRANSFORM:

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{i\omega t} d\omega$$

- The Fourier transform converts a time- or space-domain function f(x) into a frequency-domain representation F(k)
- The inverse transform reconstructs the original function:
- This pair allows analysis of signals by identifying amplitude and phase contributions at different frequencies.
- Applications and Extensions
- Signal Processing: Noise removal, compression, and modulation.
- Image Analysis: 2D Fourier transforms identify patterns in images for tasks like edge detection.

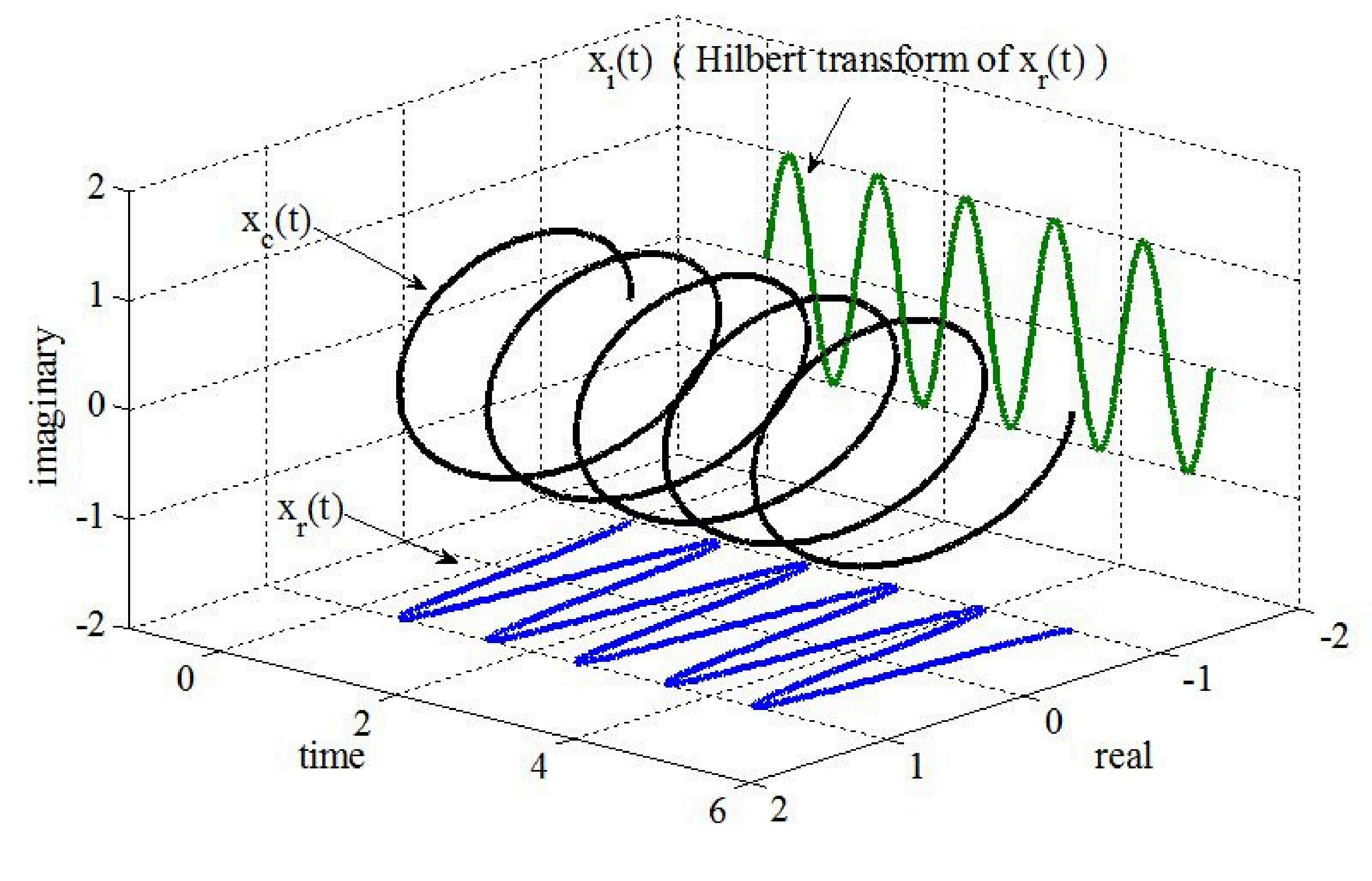


TRANSFORM FUNCTIONS EXPLORED:

HILBERT TRANSFORM:

Definition and Operation: The Hilbert transform of a signal x(t) is defined as the convolution of x(t) with 1/πt, or equivalently, as an operator that shifts the phase of all positive frequency components by -90o and all negative frequency components by +90o.

$$H[u](t) = u(t) * \left(\frac{1}{\pi t}\right)$$

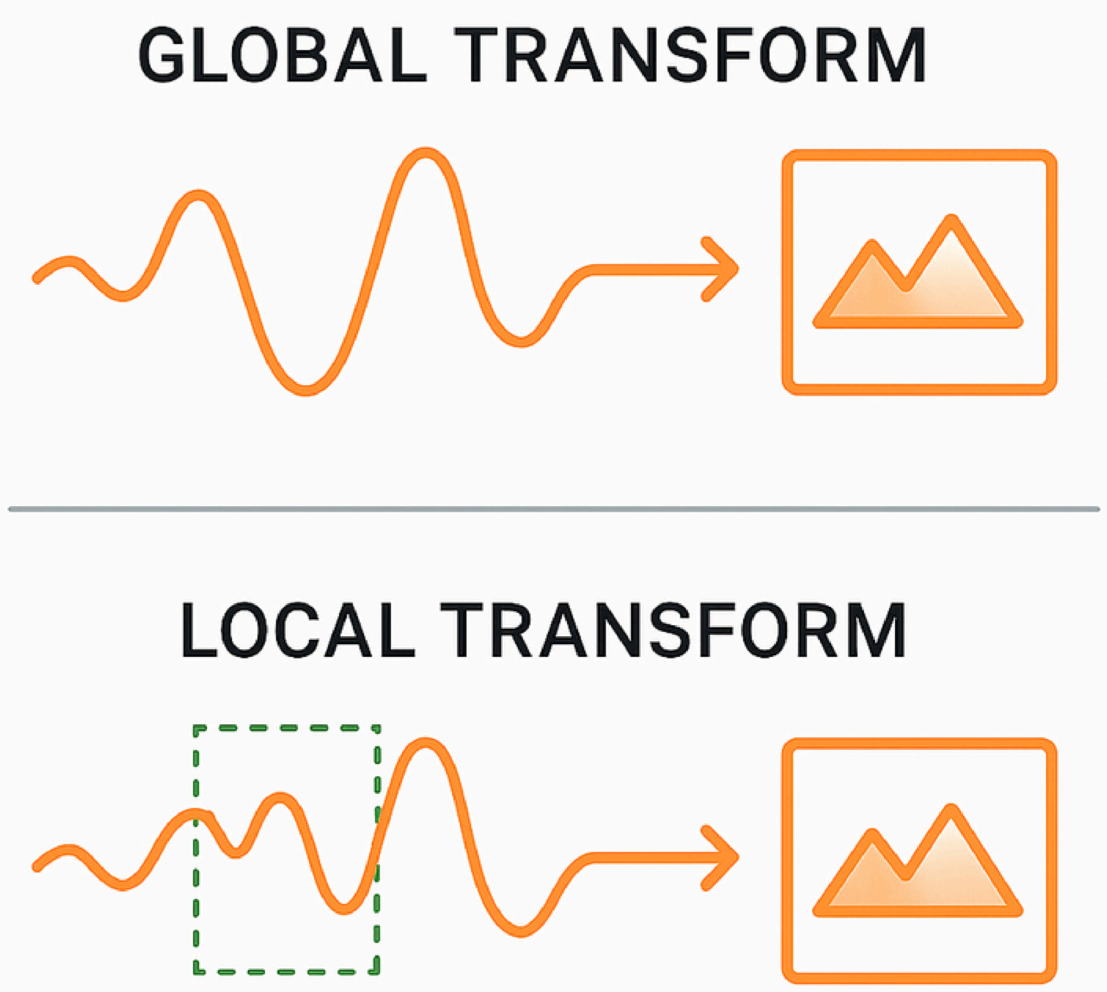


The Hilbert transform at time t is the weighted average of the signal u(τ) at all other times τ, where the weights are given by 1/(t-τ), and the averaging is done carefully to avoid the singularity at t=τ.
where ⋆ denotes convolution.

TRANSFORM FUNCTIONS EXPLORED:

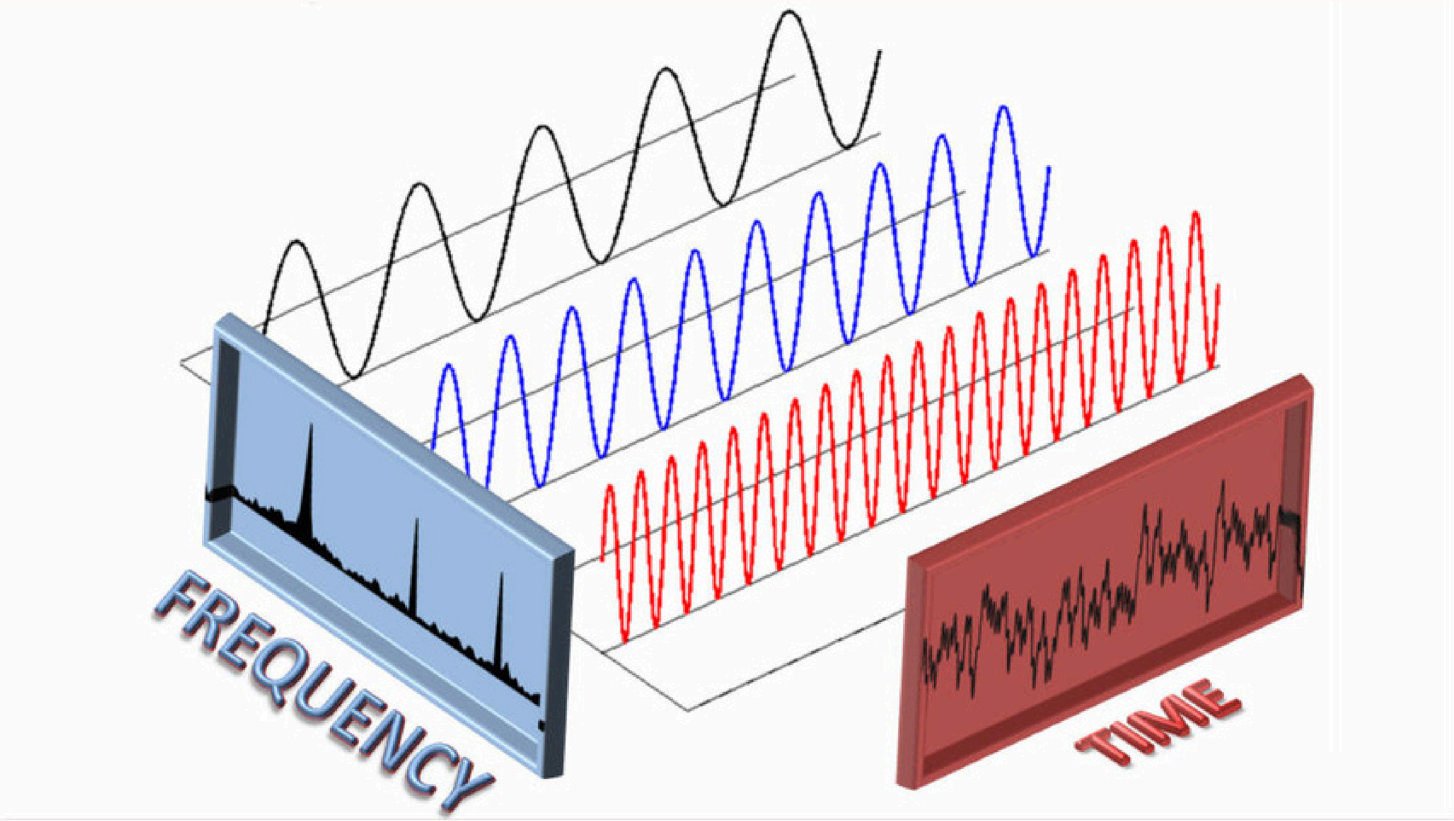
GLOBAL VS LOCAL TRANSFORMS:

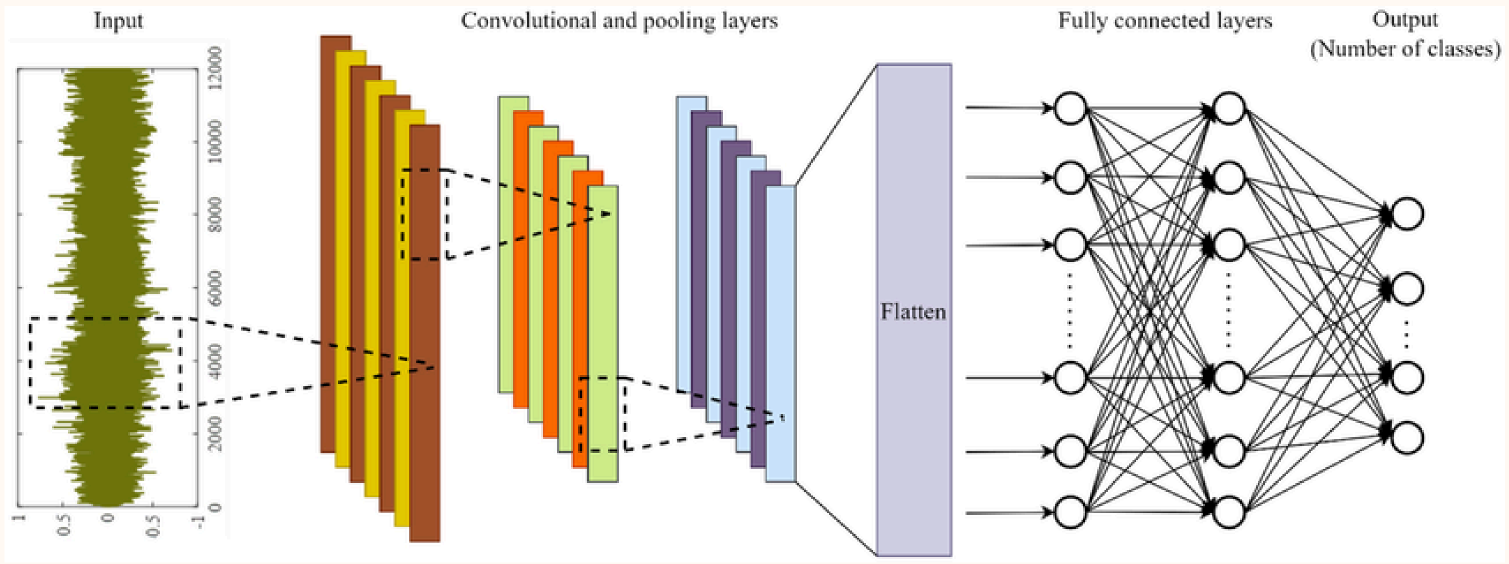
- Scope:
  - Global transforms (e.g., Fourier Transform, DCT) analyze the entire signal or image at once, capturing overall frequency or spatial information.
  - Local transforms (e.g., wavelet transform, short-time Fourier transform) focus on specific regions or segments, capturing localized features in both time/space and frequency.
- Feature Representation:
  - Global transforms offer fixed resolution across the signal/image, lacking adaptability to changes or transients.
  - Ideal for tasks like object recognition or image stitching.
- Computational Considerations:
  - Global transforms are typically computationally lighter and faster, whereas local transforms are more computationally intensive but offer greater robustness.



PROPOSED SOLUTION

- Our approach is to convert the vibration signals into grayscale images and apply deep learning methods like AlexNet/Darknet to classify them.
- To further improve the accuracy of predictions, we can use preprocessing techniques like Fourier Transform/Hilbert Transform and then apply deep learning methods directly to 1d data instead of converting it into 2d images
- Integrating both deep learning and signal processing techniques are proved to efficient in the literature study.





DATA SET AND ANALYSIS

- We considered a public dataset from GitHub for evaluation of our method:
- MetroDataset: This data set contains both normal and failure datasets of metro bogies. We collected X, Y and Z axis vibration acceleration data of bogies at a sampling frequency of 1024. The data set includes about 50.8 minutes of normal data and 101.4 minutes of failure data. Document description: Two energy-harvesting sensor nodes was used to collect the vibration acceleration data from two metro bogies. One bogie is in normal state and the other is in a failure state. A 3-axis accelerometer is used in each sensor node to collect 3-axis vibration acceleration data.

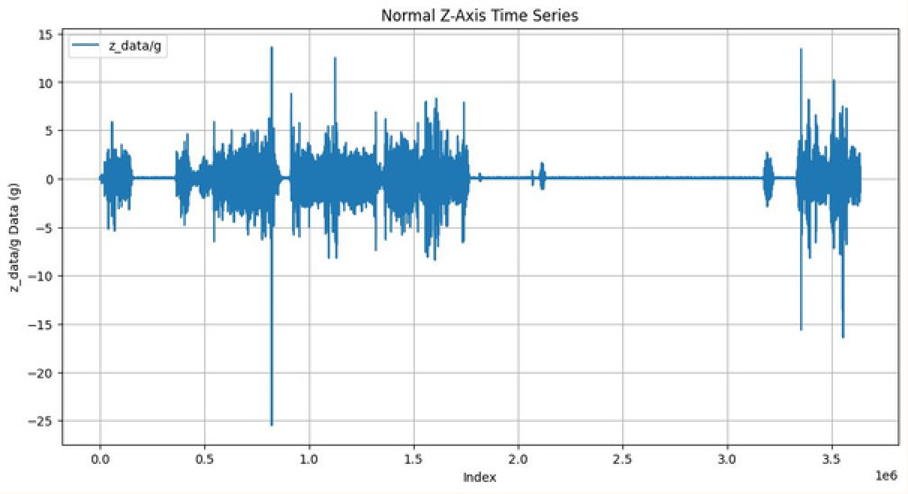
DATA PATTERN:

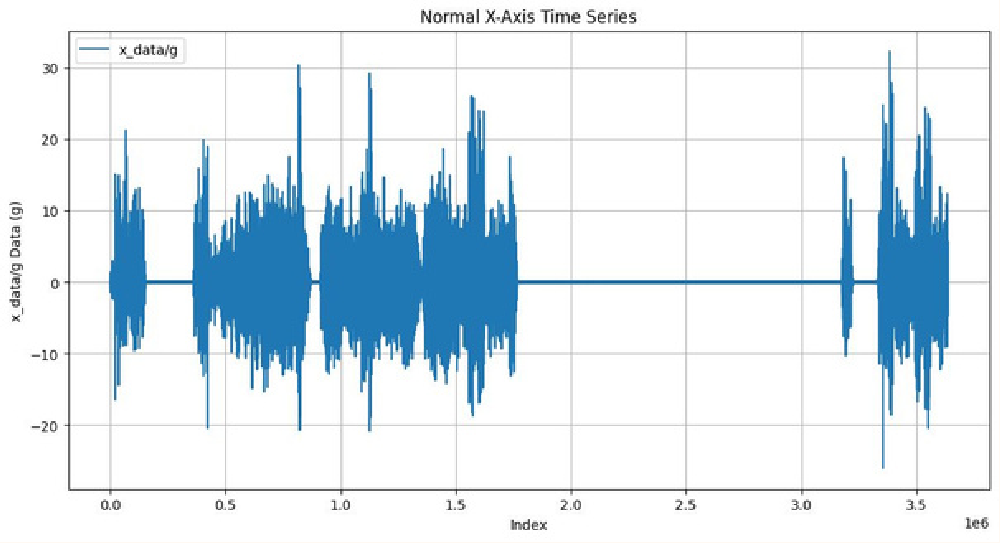
```
[           index  x_data/g
0              0      -0.1
1              1      -0.1
2              2       0.2
3              3       0.3
4              4       0.3
...          ...       ...
3635195  3635195       1.6
3635196  3635196       2.7
3635197  3635197       1.0
3635198  3635198      -0.8
3635199  3635199       0.1

[3635200 rows x 2 columns],
           index  z_data/g
0              0       0.0
1              1       0.1
2              2       0.0
3              3       0.1
4              4       0.1
...          ...       ...
3635195  3635195       0.3
3635196  3635196      -0.8
3635197  3635197       0.3
3635198  3635198       0.3
3635199  3635199      -1.0

[3635200 rows x 2 columns],
           index  y_data/g
0              0      -0.0
1              1      -0.1
2              2       0.0
3              3       0.0
4              4       0.0
...          ...       ...
3635195  3635195      -0.9
3635196  3635196       0.9
3635197  3635197       0.5
3635198  3635198      -0.3
3635199  3635199       1.1

[3635200 rows x 2 columns]]
```

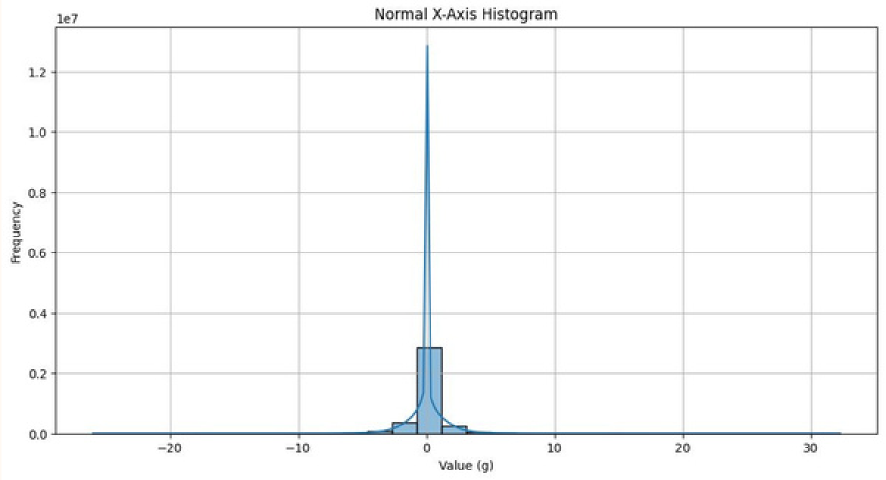
```
[           index  y_data/g
0              0      -0.6
1              1      -0.3
2              2      -0.1
3              3      -0.2
4              4      -0.5
...          ...       ...
6231035  6231035      -0.0
6231036  6231036      -0.0
6231037  6231037       0.0
6231038  6231038      -0.0
6231039  6231039       0.0

[6231040 rows x 2 columns],
           index  z_data/g
0              0      -0.0
1              1      -0.1
2              2      -0.1
3              3       0.1
4              4       0.1
...          ...       ...
6231035  6231035       0.1
6231036  6231036       0.0
6231037  6231037       0.0
6231038  6231038       0.0
6231039  6231039       0.1

[6231040 rows x 2 columns],
           index  x_data/g
0              0      -0.4
1              1      -0.4
2              2      -0.6
3              3      -0.2
4              4       0.3
...          ...       ...
6231035  6231035       0.0
6231036  6231036       0.0
6231037  6231037       0.0
6231038  6231038       0.0
6231039  6231039       0.0

[6231040 rows x 2 columns]]
```

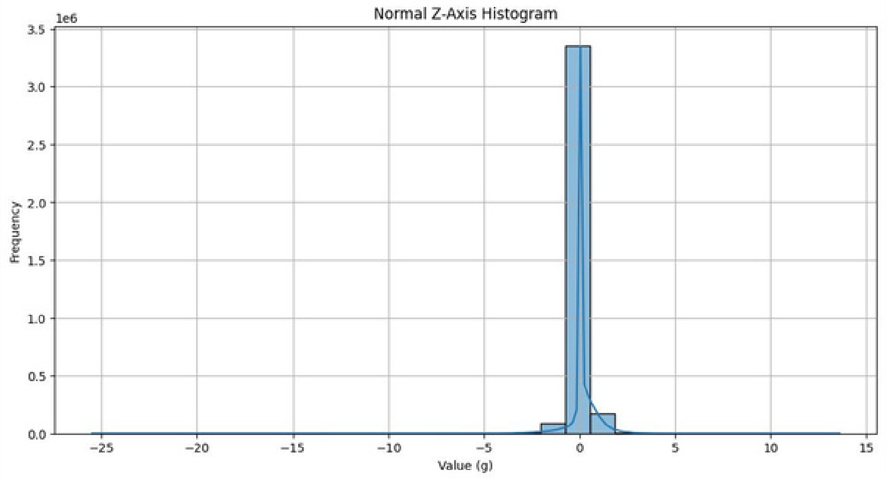
DATA SET AND ANALYSIS

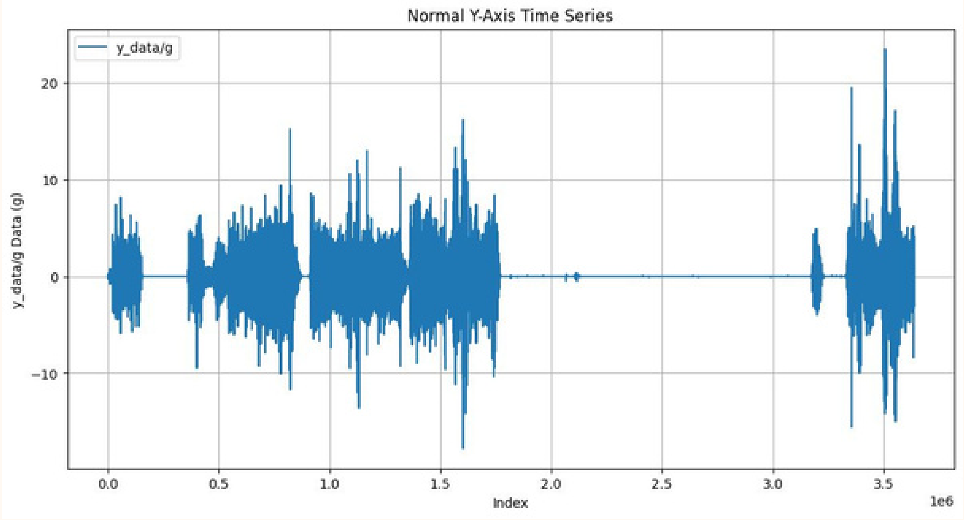
NORMAL DATA REPRESENTATION:

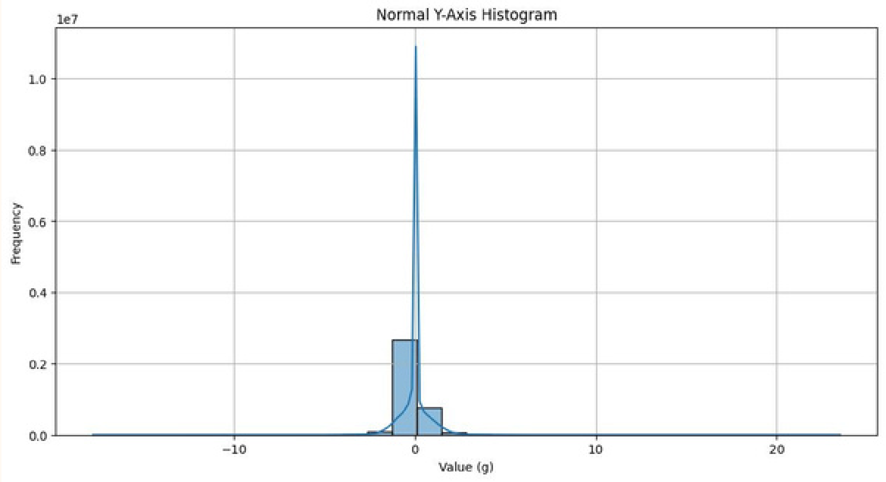








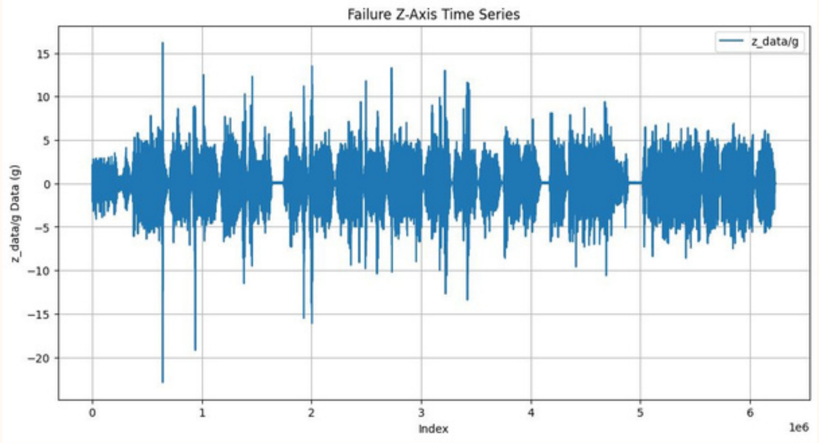




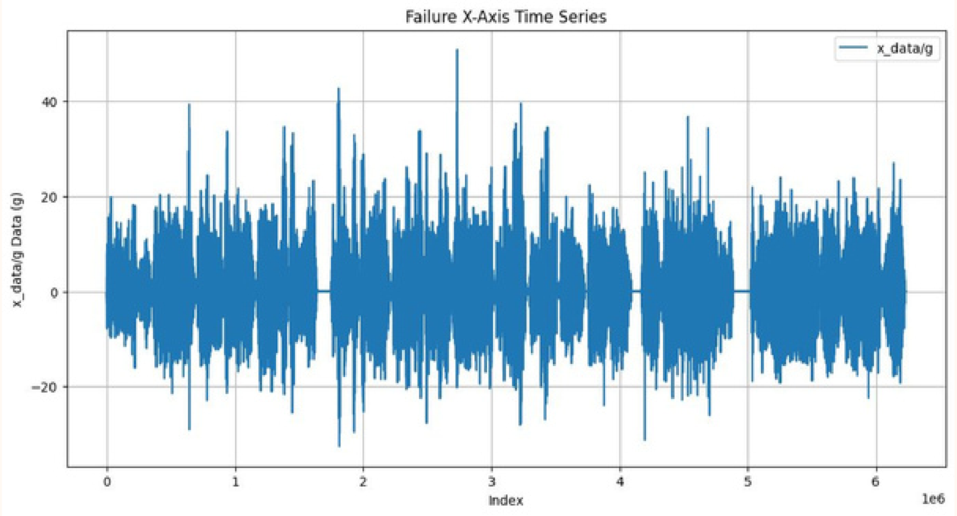
OBSERVATIONS:

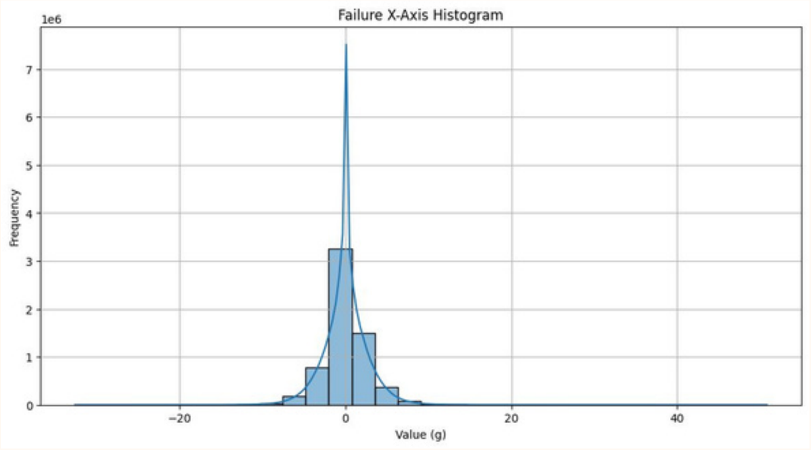
- There are many missing values in the normal data in all axes.
- This missing data will create imbalance in data while training; therefore, we need to find a method to solve this issue.
- There might be some physical or mechanical reason behind missing values which must be fed to the model.

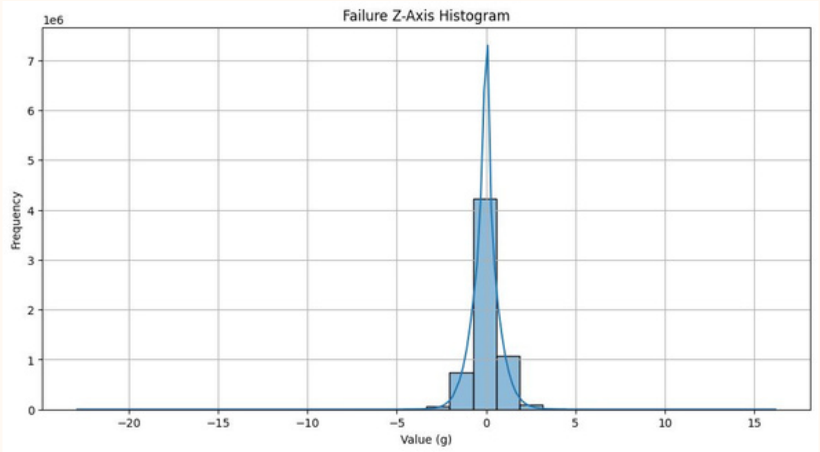
DATA SET AND ANALYSIS

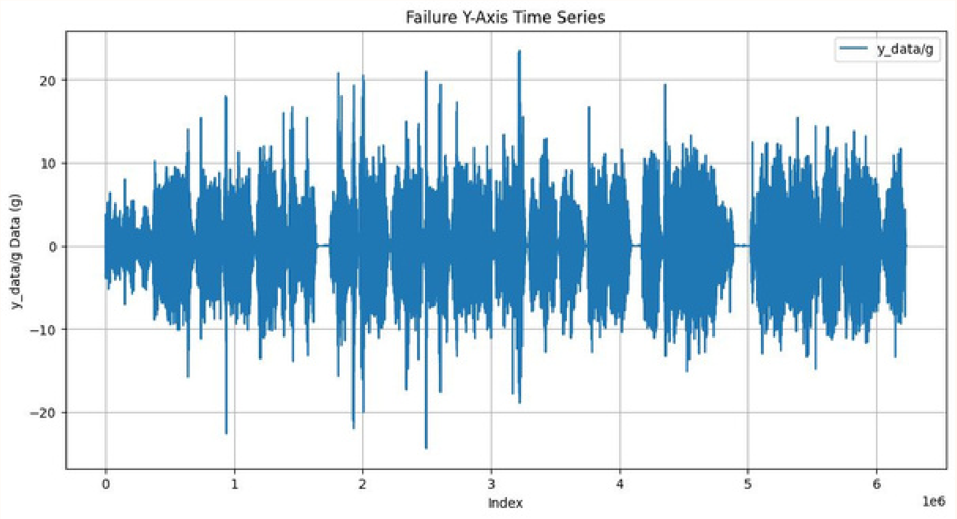


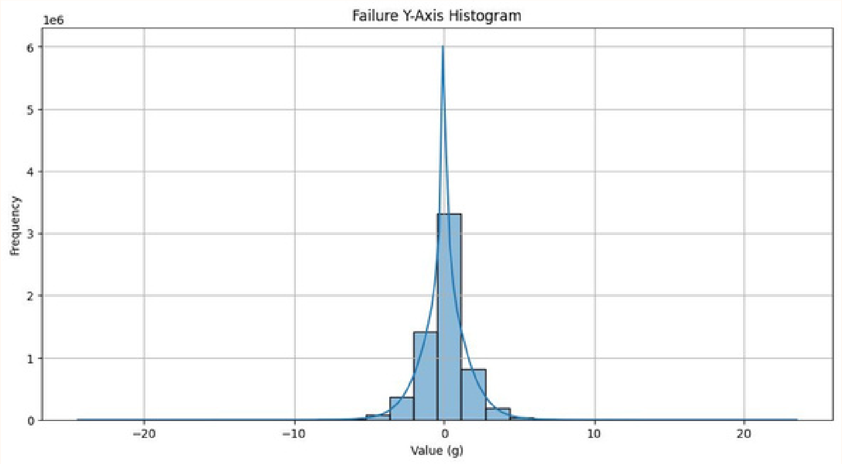
FAILURE DATA REPRESENTATION:









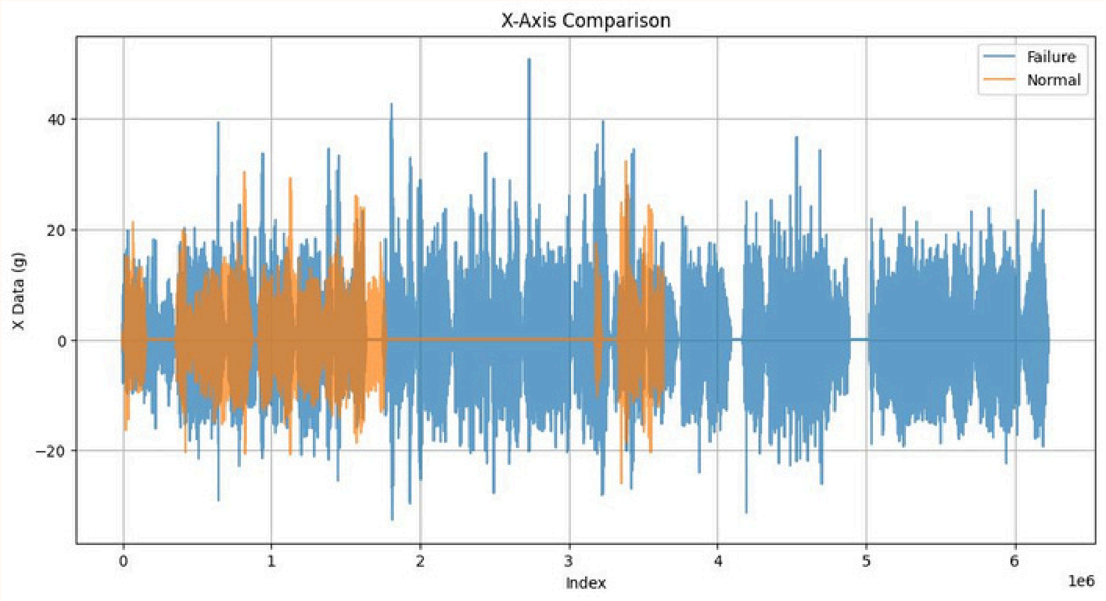


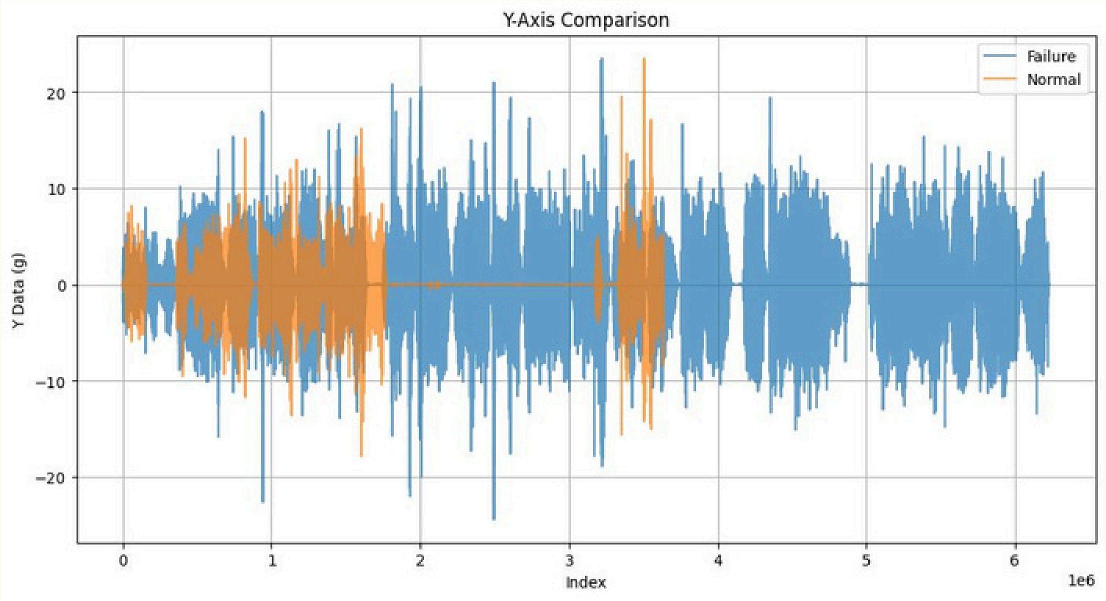
OBSERVATIONS:

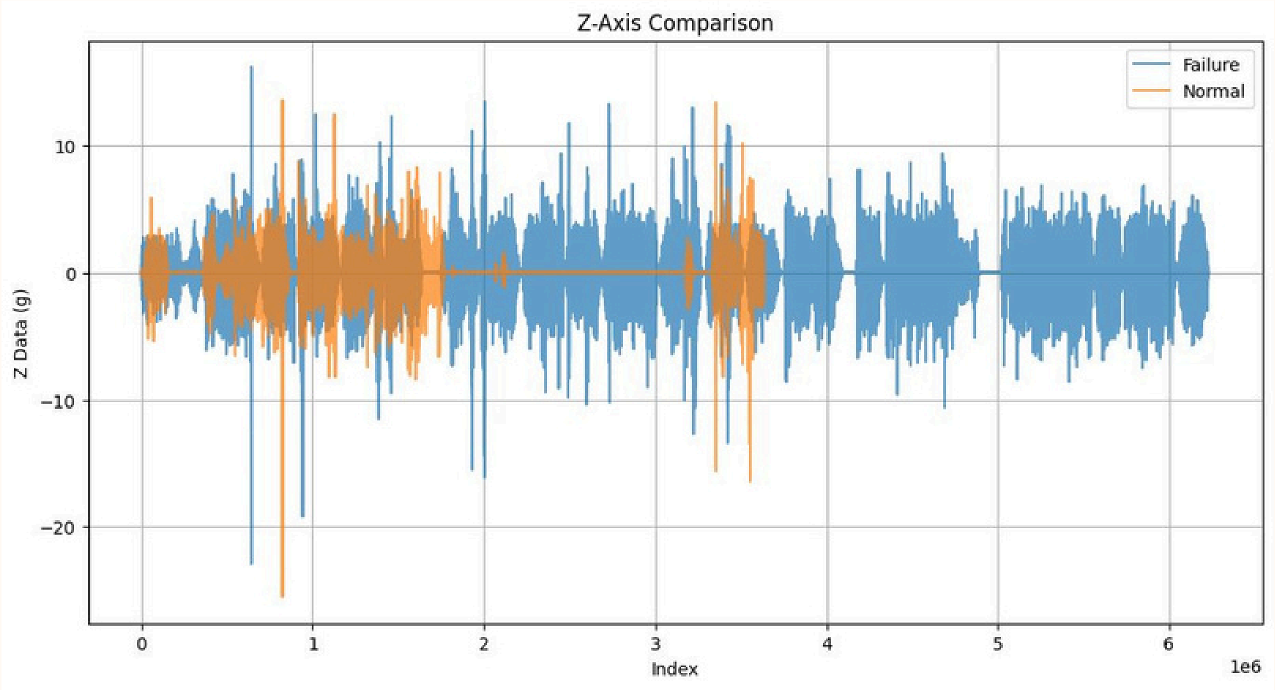
- There are no missing values in the failure data in all axes.
- This means that there is some fault in the bogie that is creating extra vibration signals

DATA SET AND ANALYSIS

COMPARISION :







- There are many missing values in the normal data in all axes but no missing values in failure data.
- This missing data will create imbalance in data while training; therefore, we need to find a method to solve this issue.
- There might be some physical or mechanical reason behind missing values, which must be fed to the model.

RESULTS AND CONCLUSION

1D-CNN MODEL PERFORMANCE:

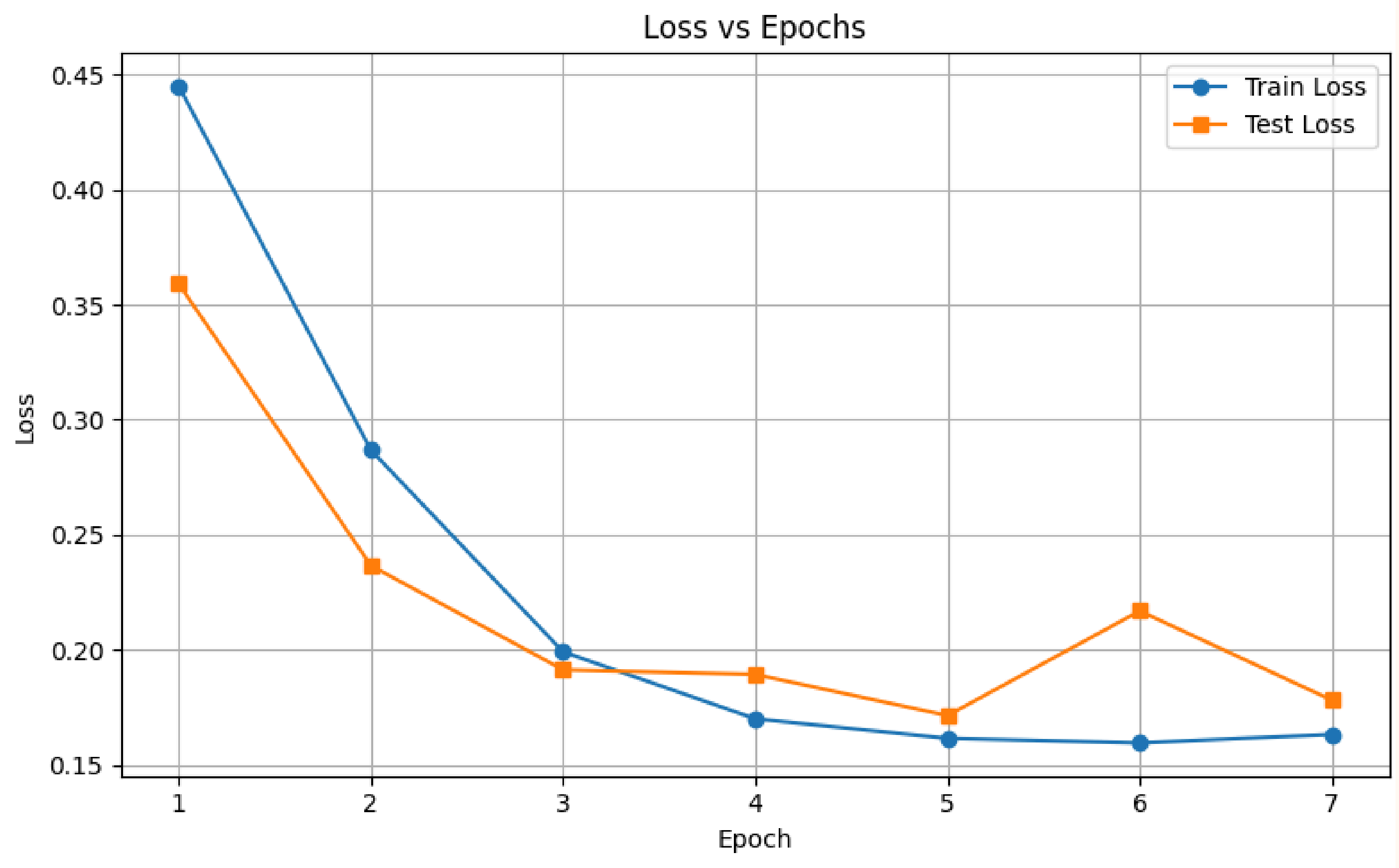
The table below summarizes the training and testing performance over epochs:

Epoch	Train Loss	Train Acc (%)	Test Loss	Test Acc (%)
1	0.4451	79.84	0.3591	83.45
2	0.2868	88.56	0.2356	91.85
3	0.1991	93.27	0.1913	93.40
4	0.1700	94.21	0.1893	93.40
5	0.1615	94.24	0.1714	93.88
6	0.1595	94.28	0.2169	92.30
7	0.1632	94.21	0.1781	93.88

Table 4.1: Training and Testing Accuracy Across Epochs

KEY OBSERVATIONS:

- Performance Improves Quickly: The model achieves high accuracy within just a few epochs.
- Test Accuracy Stabilizes: Test accuracy plateaus around ~93.88%, suggesting good generalization.
- Low Overfitting: Train and test accuracies are close — no major gap, meaning the model isn't overfitting.
- Early Stopping at Epoch 7: Training was halted as performance stopped improving significantly — helps prevent overfitting and saves training time.

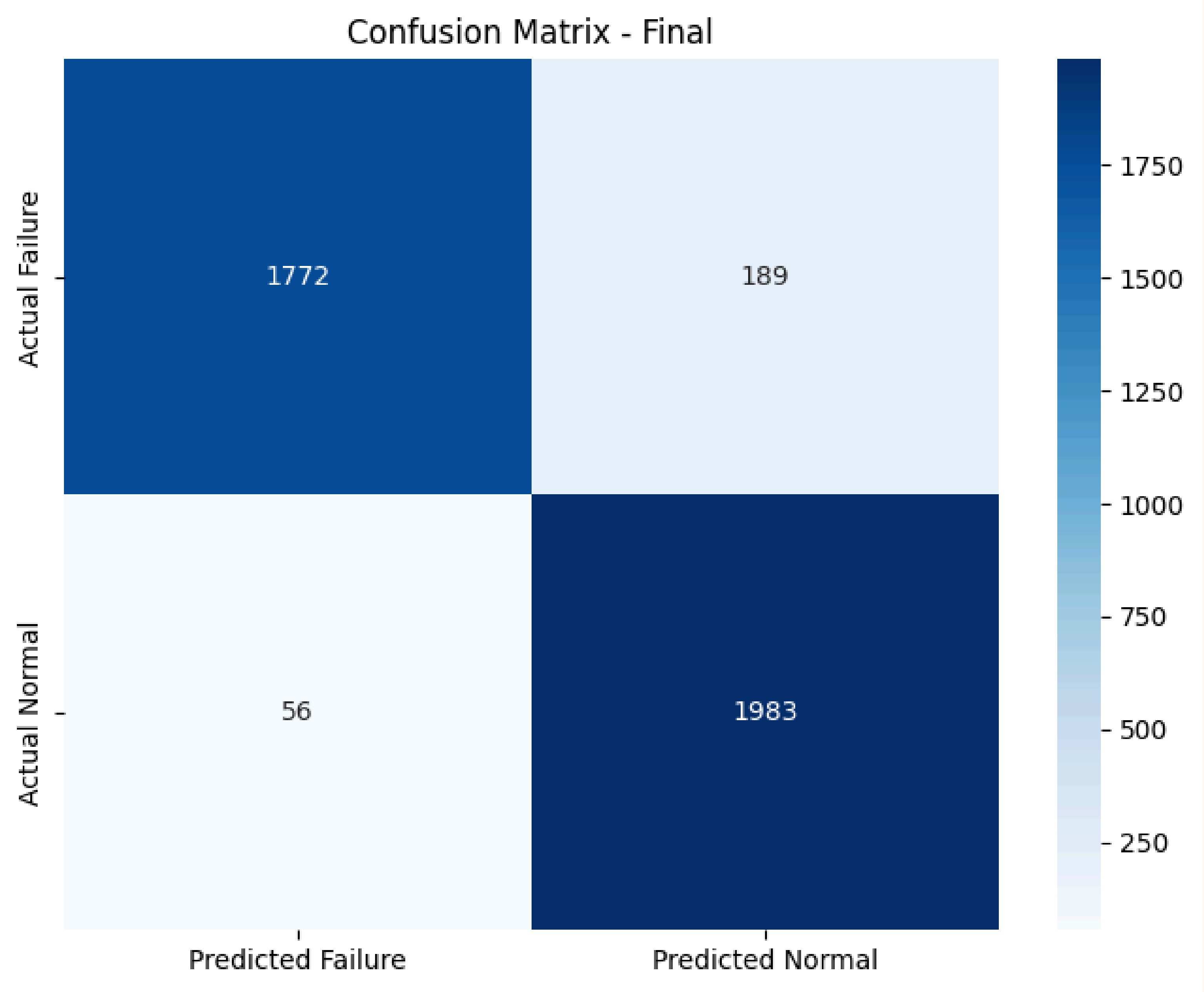


RESULTS AND CONCLUSION

1D-CNN MODEL PERFORMANCE:

KEY OBSERVATIONS:

- High Overall Accuracy:
- The model correctly classified 1772 failures and 1983 normals, showing strong predictive performance.
- Slightly Higher False Negatives:
- 189 actual failures were incorrectly predicted as normal — this could be critical in safety-critical applications like rail monitoring.
- Low False Positives:
- Only 56 normal cases were misclassified as failures, indicating good precision for detecting normal conditions.
- Balanced Class Performance:
- Both classes (failure and normal) are handled well, suggesting no major class imbalance issue or bias in model predictions.



RESULTS AND CONCLUSION

1D-CNN MODEL USING HILBERT PERFORMANCE:

KEY OBSERVATIONS:

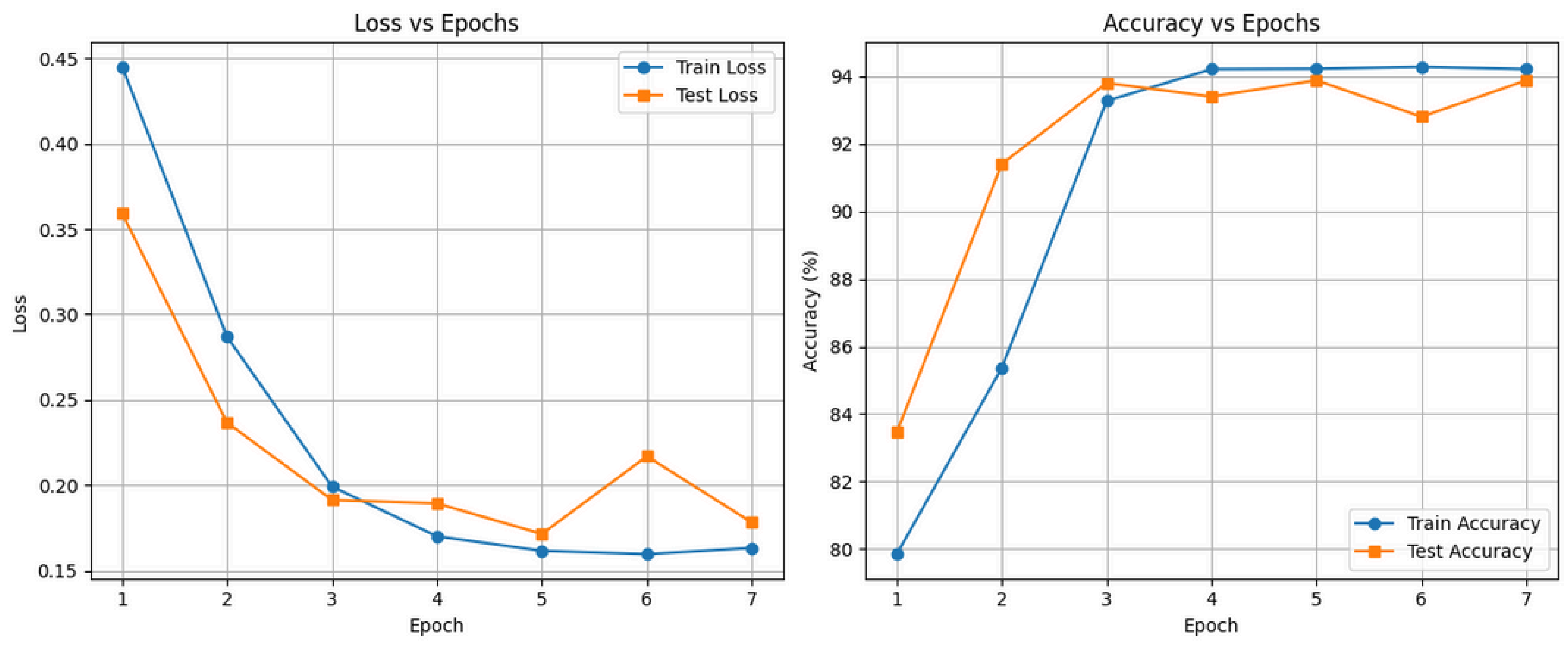
- Strong Learning Curve: Both training and test accuracy improve rapidly over epochs.
- Highest Test Accuracy: 93.97% achieved at Epoch 6.
- Slight Overfitting Risk: Small dip in test accuracy at Epoch 5 (despite high train accuracy) suggests early signs of overfitting.
- Early Stopping at Epoch 6: Triggered to prevent overfitting and maintain optimal generalization performance.

The table below summarizes the training and testing performance of the model across epochs:

Epoch	Train Loss	Train Acc (%)	Test Loss	Test Acc (%)
1	0.6317	80.37	0.2971	87.53
2	0.2360	91.44	0.1999	92.97
3	0.1878	93.41	0.1792	93.80
4	0.1510	94.74	0.1735	94.50
5	0.1473	94.91	0.3080	90.62
6	0.1373	95.19	0.1789	93.97

TABLE 4.2: Model Training and Testing Accuracy Across Epochs

Note: Early stopping was triggered at epoch 6.

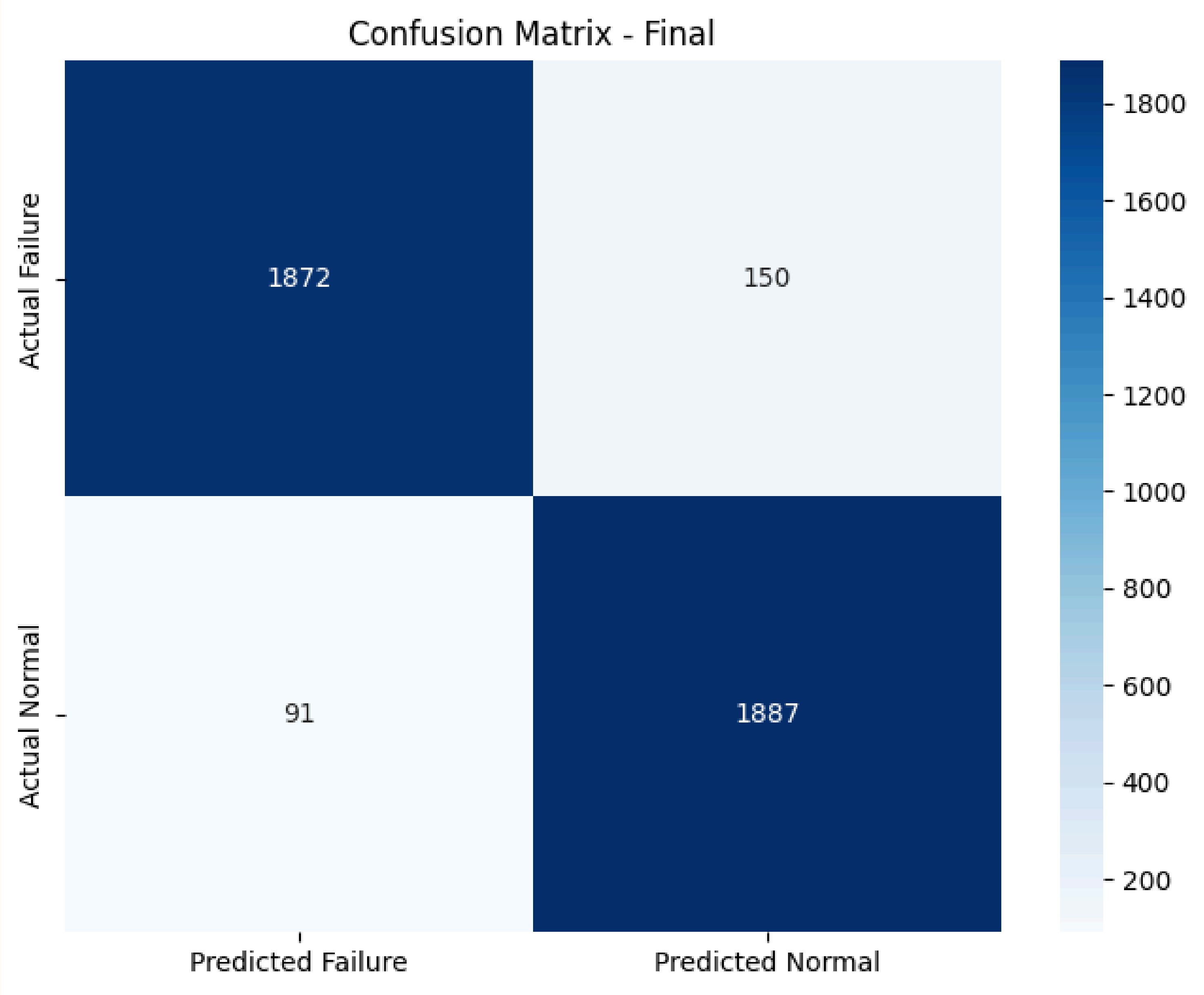


RESULTS AND CONCLUSION

1D-CNN MODEL USING HILBERT PERFORMANCE:

KEY OBSERVATIONS:

- Improved Overall Accuracy:
- The model correctly predicted 1872 failures and 1887 normals, indicating strong and balanced performance across both classes.
- Reduced False Negatives:
- Only 150 actual failures were misclassified as normal — an improvement over the earlier model, which had 189 false negatives.
- Slight Increase in False Positives and Balanced Detection:
- 91 normal cases were misclassified as failures (compared to 56 in the earlier model), showing a small trade-off for better fault detection.
- Near-symmetric performance across both classes shows that the model is not biased toward any class and is robust in binary classification.



RESULTS AND CONCLUSION

1D-CNN MODEL USING FOURIER TRANSFORM PERFORMANCE:

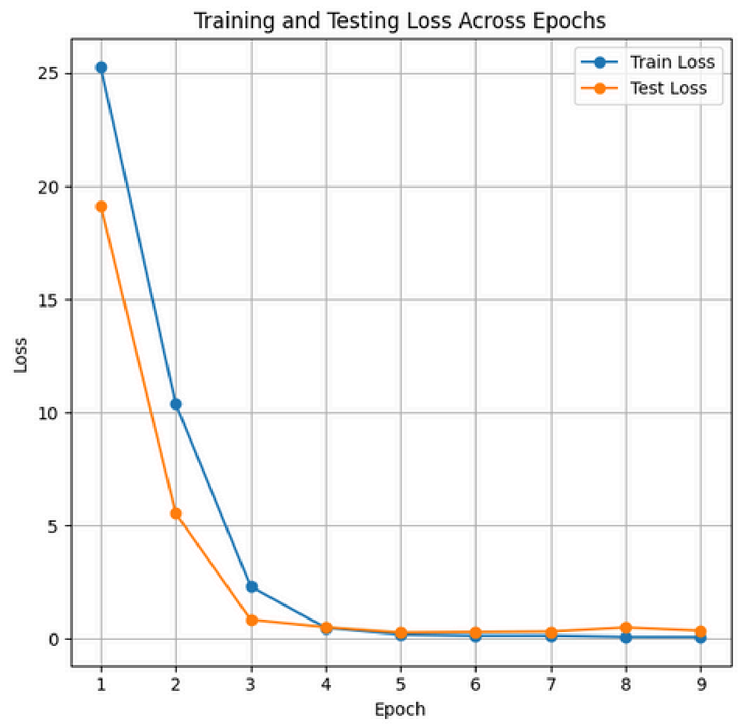
KEY OBSERVATIONS:

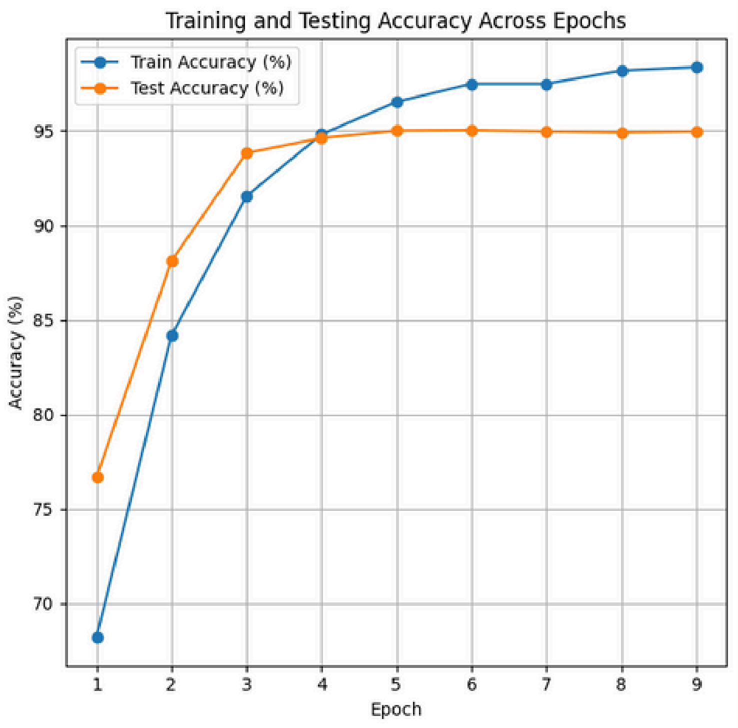
- Steep Learning Curve: Rapid improvement over first few epochs; stabilizes after epoch 5.
- Best Generalization: Test accuracy of 95.67% (Epoch 7) confirms Fourier features enhance model effectiveness.
- Overfitting Controlled: Despite very high train accuracy (~98%), test accuracy remains consistent, suggesting good generalization.
- Early Stopping at Epoch 9: Prevents degradation due to overtraining and locks in optimal performance.

Epoch	Train Loss	Train Acc (%)	Test Loss	Test Acc (%)
1	25.2586	68.21	19.1476	76.67
2	10.3837	84.17	5.5487	88.12
3	2.3168	91.54	0.8514	93.83
4	0.4926	94.81	0.5187	94.62
5	0.1933	96.53	0.2798	95.00
6	0.1485	97.49	0.3053	95.03
7	0.1213	97.84	0.3274	95.67
8	0.0959	98.19	0.5064	94.92
9	0.0935	98.37	0.3657	94.97

TABLE 4.3: Fourier Transform Model: Training and Testing Accuracy Across Epochs

Note: Early stopping was triggered at epoch 9.



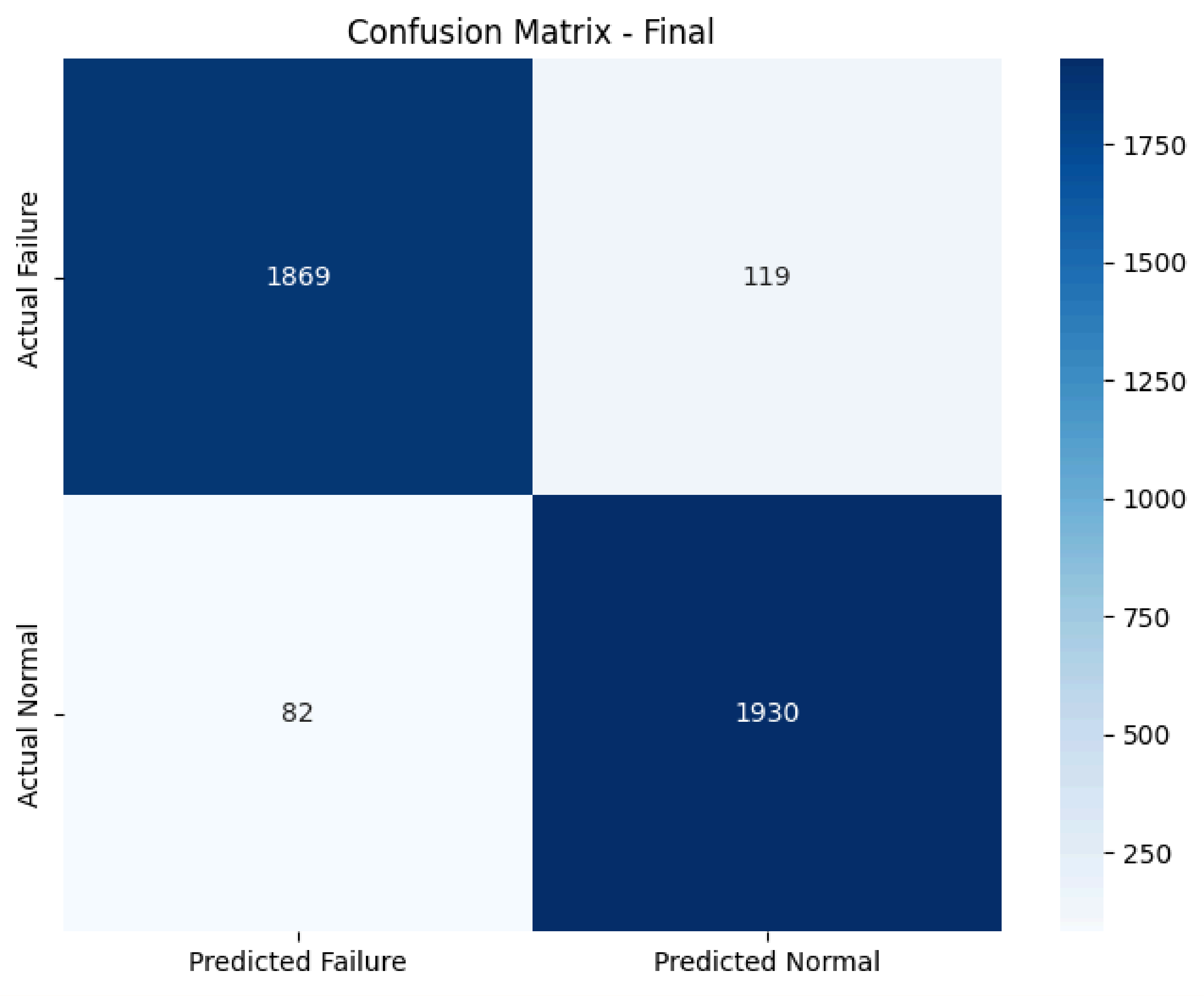


RESULTS AND CONCLUSION

1D-CNN MODEL USING FOURIER TRANSFORM PERFORMANCE:

KEY OBSERVATIONS:

- Highest Accuracy Observed:
- The model correctly classified 1869 failures and 1930 normal cases, indicating strong performance on both classes.
- Lowest False Negative Count:
- Only 119 actual failures were misclassified as normal — this is the best among all previous models, making it the most reliable for fault detection.
- Balanced Misclassifications and Best Precision-Recall Tradeoff:
- False positives (82) and false negatives (119) are both low and relatively balanced, showing the model generalizes well without favoring one class.
- The matrix reflects high precision for failure detection and excellent recall for normal conditions — making it ideal for safety-critical monitoring.



CONCLUSION AND FUTURE WORKS

To evaluate the performance of the three approaches-Normal 1D CNN (without transform), Hilbert Transform, and Fourier Transform-we analyzed their confusion matrices and calculated their overall accuracies:

Model	True Failure	False Normal	False Failure	True Normal	Accuracy (%)
Normal 1D CNN	1772	189	56	1983	93.88
Hilbert Transform	1872	150	91	1887	93.98
Fourier Transform	1869	119	82	1930	94.98

TABLE 5.1: Comparison of Confusion Matrix Results and Accuracies

Key Observations:

The Fourier Transform-based model achieved the highest accuracy, demonstrating superior ability to correctly classify both failure and normal cases.

The Hilbert Transform model also outperformed the baseline 1D CNN, showing that applying signal transforms before classification can enhance model performance.

The Normal 1D CNN (without transforms) had the lowest accuracy and slightly higher misclassification rates, particularly more false negatives (failures predicted as normal).

CONCLUSION AND FUTURE WORKS

- Takeaways:
  - Accuracy is increasing when when we combine deep learning methods with traditional methods
  - DIfferent transforms are giving different accuracies when combined with neural networks
  - Transforms are capturing patterns that are not naturally captured using normal data as input to the AI model.

- Scope in Future:
- We can try out different transforms for preprocessing and find the best transform that works.
- We can use RNNs and LSTMs instead of 1D-CNNs to better capture sequential data because they are known for handling sequential data.
- During data modelling, we can feed a merge of both raw data and transformed data to the model which may lead to better capturing.

THANK YOU