Operational Trouble Cases and Preliminary Attempts at Anomaly Detection in J-PARC

（**J-PARC**におけるトラブル事例と異常検知の試み）

加速器･ビーム物理の機械学習ワークショップ2025
8-9 Dec., 2025

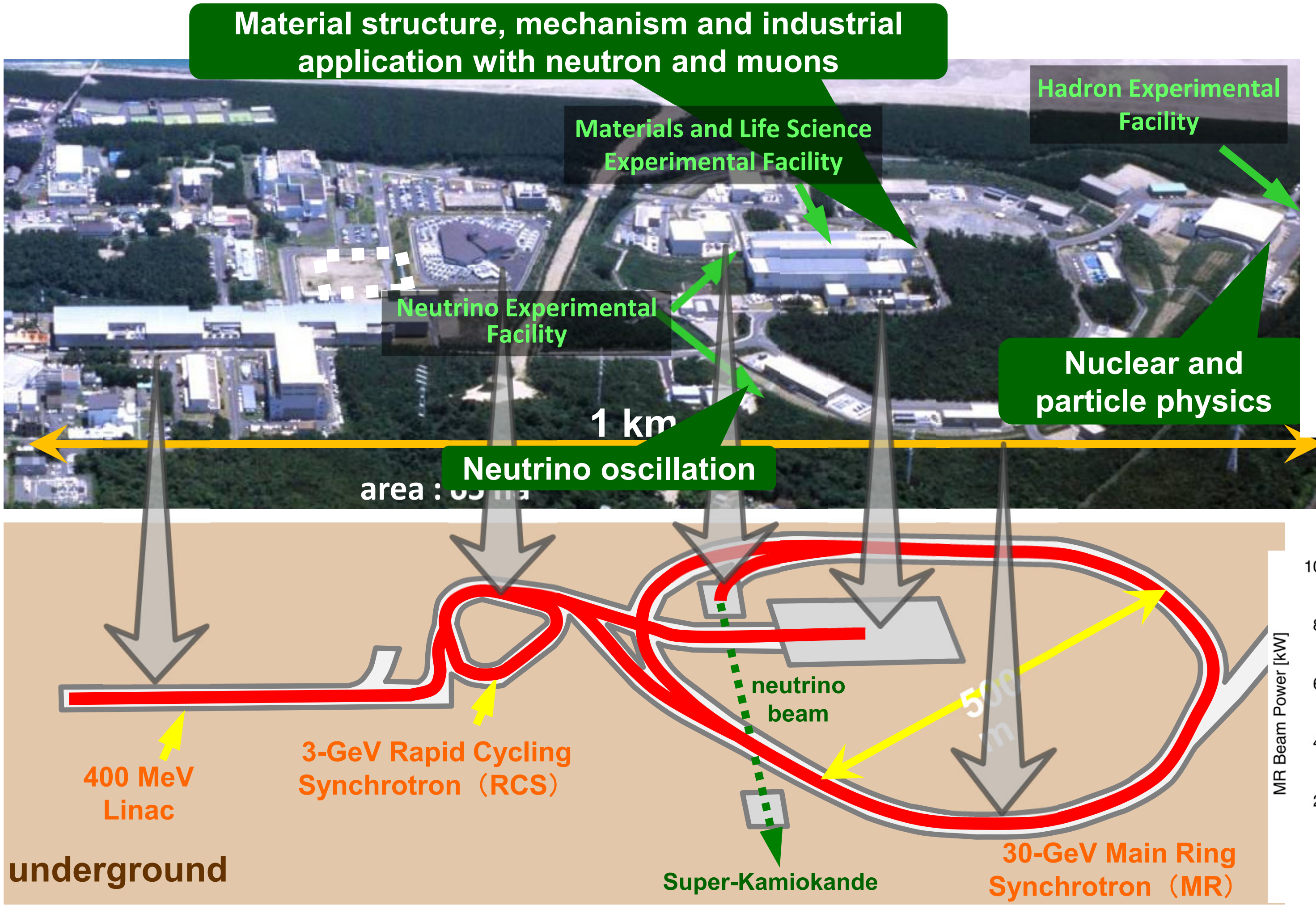
Kazami Yamamoto

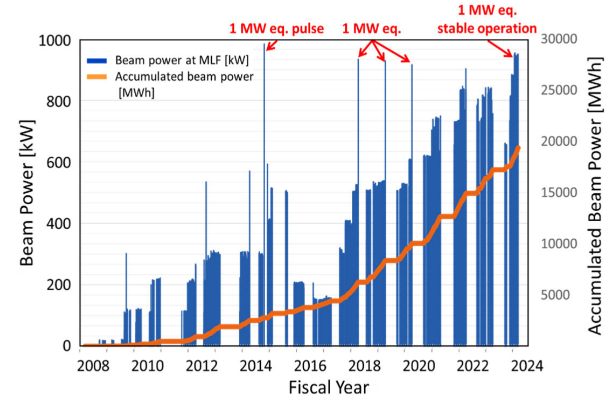
J-PARC Center
Accelerator Division

Contents

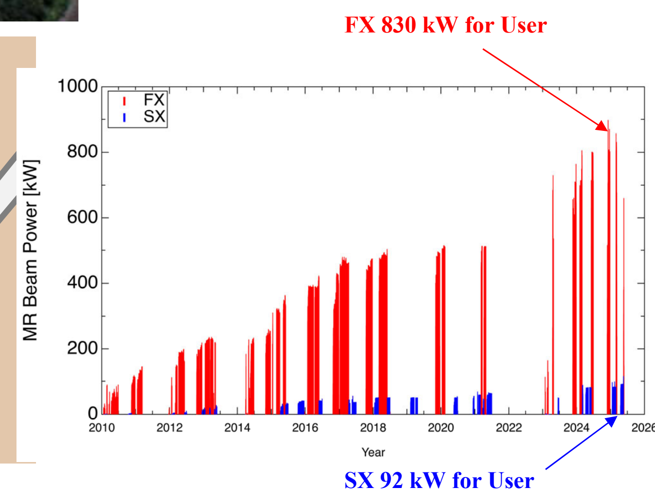
- Introduction and Motivation
- Preparation for Anomaly detection from sounds
- Summary

Japan Proton Accelerator Research Complex





Beam power history at MLF



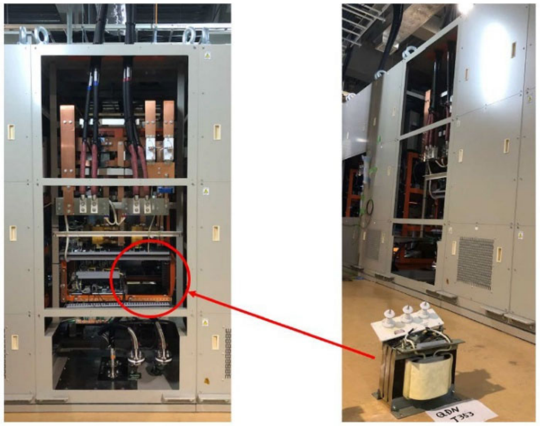
Beam power history of MR

3 proton accelerators and 3 experimental facilities

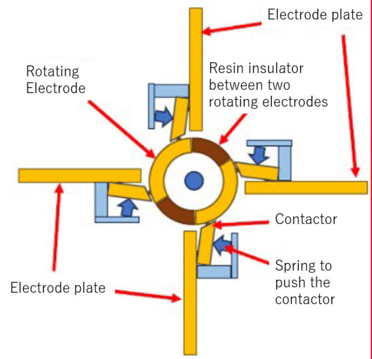
RCS and MR achieved the design beam power (1 MW@RCS, 0.75MW@MR) and pursue more high power! (>1.5 MW@RCS, 1.3 MW@MR)
However, the availability was reduced due to aging of components.

Major troubles

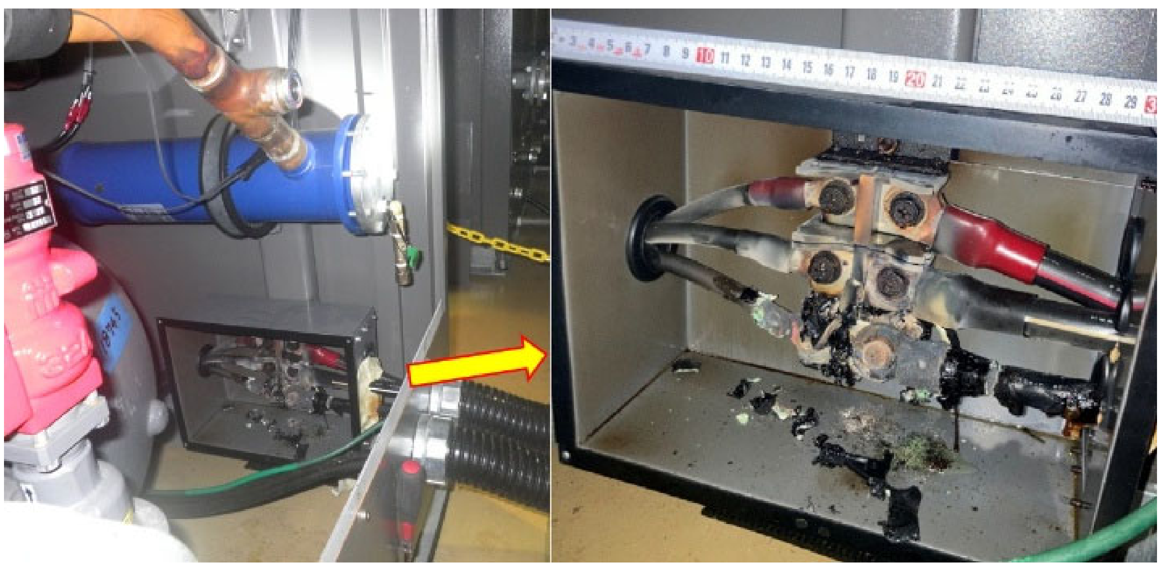
First fire incident : MR, April 25th, 2023



Second fire incident :
Hadron facility, June 22nd, 2023

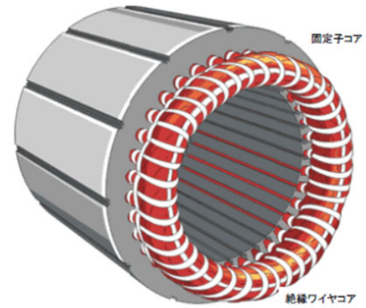


Third fire incident : Linac, July 5th, 2024 (Found)

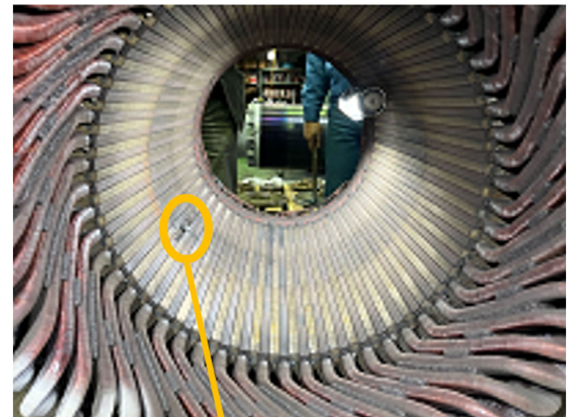


Detect by an odor sensor : future plan

Failure of the cooling water pump



Applied voltage 1000V
: U、V、W
→ 0.3 MΩ



So far, we start to detect this kind of anomaly event from the sound.

Today's main theme

Preparation for Anomaly detection from sounds

J-PARC

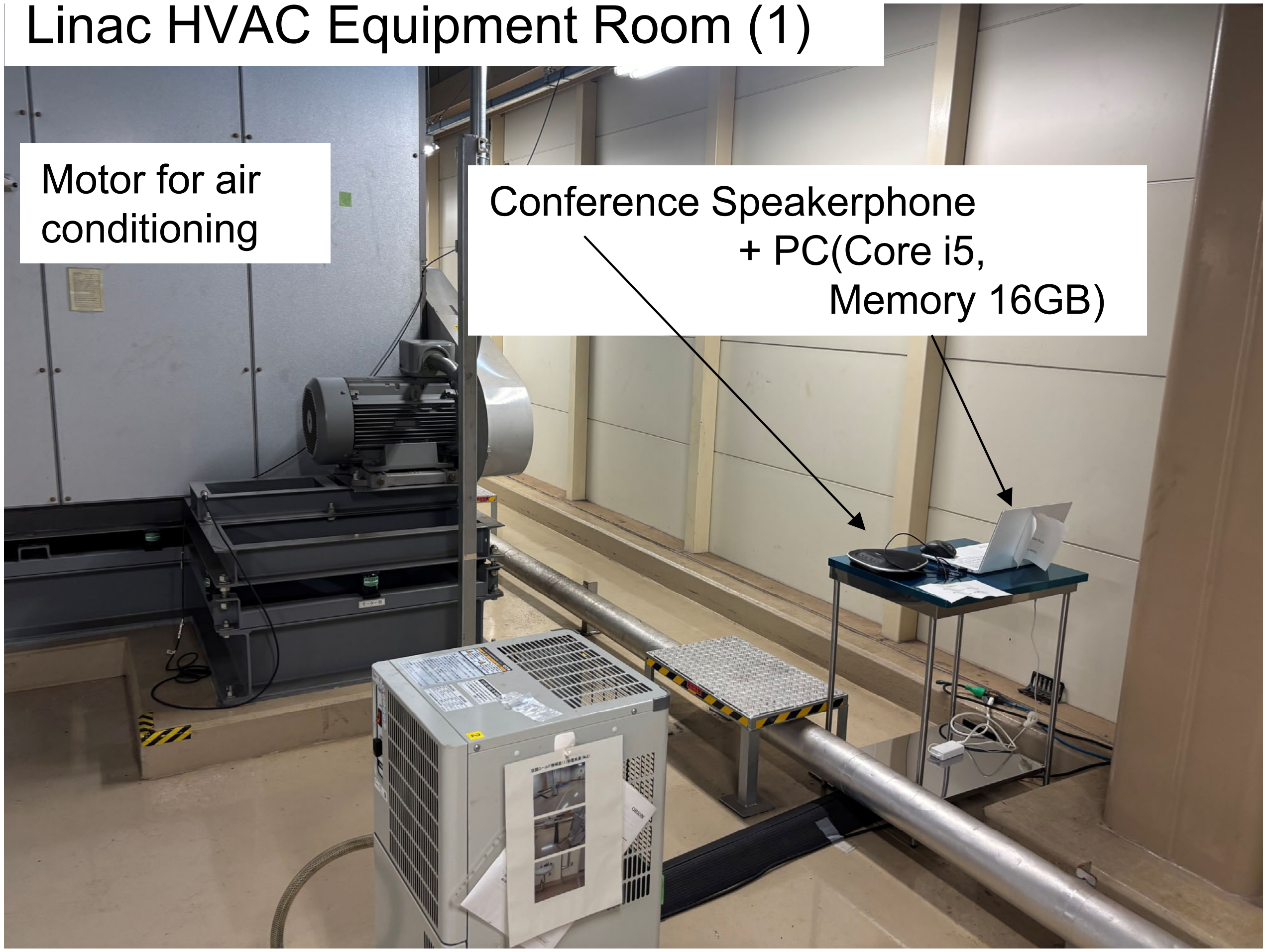
Basic Policy

- Develop machine learning-based anomaly diagnosis by the sounds as **universally applicable** as possible.

- Make the process of anomaly diagnosis as **effortless** as possible by automating data collection, analysis, and reporting.

- Windows-based system
  - ✓ Learning was conducted on the faster machine
  - ✓ WAV recording and analysis on the small notebook by Task scheduler and batch files

Data Aquisition



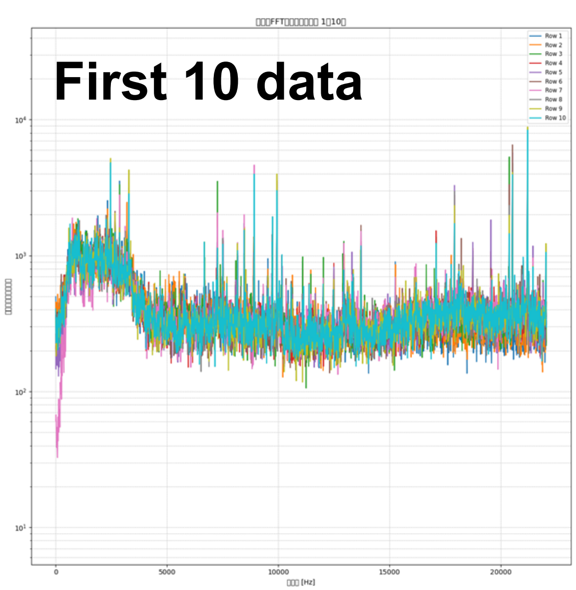
Using **PyAudio** in Python for audio recording

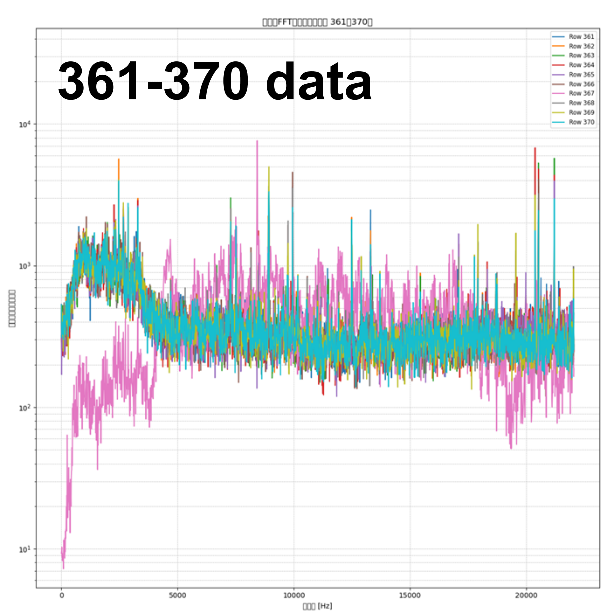
Advantages:
- **The input audio device can be explicitly selected.**
- Multiple microphones can be connected, and each microphone input can be individually specified, enabling audio recording from distinct microphone sources.

A Python recording script is executed within the batch file.
The batch file can be repeatedly executed at arbitrary intervals using the Task Scheduler.
For the purpose of collecting training data, **10-second** audio WAV files are recorded every **2 hours**.

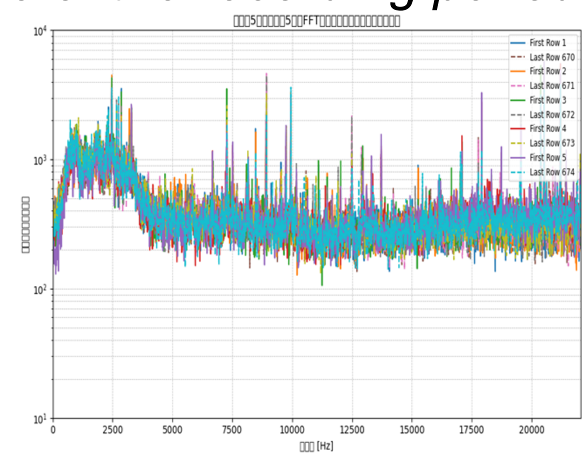
Training Data confirmation

- Data were collected every two hours during the periods from June 13 to July 3 and from July 17 to August 27, resulting in a total of 674 files.
- AT first, an FFT was applied to each entire 10-second sound file to check their spectral characteristics.
- Among the 674 samples, **5 exhibited significantly different spectral shapes**.
- Upon listening to the audio, we understand that all five were affected by temporary broadcast announcements.
- These five samples were therefore excluded, and the remaining 669 samples were used as the reference dataset.





The first and the last five samples were compared (because maintenance occurred in the middle of the recording period.)



Specific Feature

J-PARC

- There are TOO many studies on sound-based anomaly detection—what features should we focus on?"

Recent advancements of signal processing and artificial intelligence in the fault detection of rolling element bearings: a review

A. Anwarsha[1], T. Narendiranath Babu[2]
School of Mechanical Engineering, Vellore Institute of Technology, Vellore, 632 014, Tamil Nadu, India
[2]Corresponding author
E-mail: [1]*anwarsha.a2019@vitstudent.ac.in*, [2]*narendiranathbabu.t@vit.ac.in*

Received 31 December 2021; received in revised form 20 April 2022; accepted 15 May 2022
DOI https://doi.org/10.21595/jve.2022.22366

Check for updates

Abstract. A rolling element bearing is a common component in household and industrial machines. Even a minor fault in this section has a negative impact on the machinery's overall operation. As a result, the industry suffers significant financial losses, and this damage can potentially result in catastrophic failures. Therefore, even a little fault in the rolling element bearings must be recognized and remedied as soon as possible. Many ways for detecting REB defects have been created in recent years, and new methods are being introduced on a daily basis. This article will provide a summary of such methods, with a focus on vibration analysis techniques. The newest advancements in this field will be recognizable to readers of this article. Anyone interested in defect diagnostics of rolling element bearings can utilize this material.

Keywords: rolling element bearings, fault diagnosis, signal processing, vibration analysis, acoustic emission, artificial intelligence, machine learning, deep learning.

Some papers summarize the techniques commonly used in anomaly detection through sound analysis. At first, we have selected the following features from among them.

- **FFT/STFT**
- **MFCC+RMS**
- **Wavelet transform**

RECENT ADVANCEMENTS OF SIGNAL PROCESSING AND ARTIFICIAL INTELLIGENCE IN THE FAULT DETECTION OF ROLLING ELEMENT BEARINGS: A REVIEW. A. ANWARSHA, T. NARENDIRANATH BABU

References	Main objectives	Techniques used	Main findings
[111]	Damage detection of REB	Discrete impulse frequency translation	The technique is applicable to all types of REB
[114]	Diagnosis and monitoring of REB	Power Spectral Density analysis + ANN	Efficient fault identification
[115]	FD of low-speed bearing	Multiclass relevance vector machine	More suitable in the real application, better computational ability
[116]	Multi-fault diagnostics of ball bearing	FFT + wavelet energy entropy means + RMS	Good computational efficiency, better resolution
[117]	Extracting fault characteristics	WPT + Kurtogram	Precisely match the fault characteristics for various noisy signals
[118]	REB fault diagnosis	WPT + manifold learning	Effective and reliable for early-stage fault detection
[119]	Bearing FD	Multi-scale morphology analysis	Can be used to determine damage modes
[120]	Diagnostics of REB defects	Envelope detection + PSO	Capable of identifying all types of faults in incipient stages
[121]	Rolling element bearing fault classification	EMD + Multiscale dynamic time warping	Simple and reliable, necessitate a smaller number of samples, lower the period
[122]	Enrichment of defect signature in REB	Fast non-local means algorithm	Effective bearing diagnostics against different signal-to-noise ratio
[123]	Fault detection of aero-engine bearing	WPT + Rough set theory	Quick detection of bearing faults
[124]	FD of rolling bearing	Discrete Hidden Markov Model	A reliable, on-site diagnosis is possible
[125]	Bearing fault diagnosis	Wavelet packet decomposition + SVM	High recognition accuracy, easy to apply
[126]	Rolling element bearing fault diagnostics	Over-Complete rational dilation WT	Better performance under background noise and speed fluctuations
[127]	Fault diagnostics of REB	Sparsogram + Lempel-Ziv	Better performance in the quantitative diagnostics
[128]	Fault feature extraction from bearings	Improved resonance-based signal sparse decomposition	Better performance compared with conventional spectrum analysis
[129]	REB incipient defect diagnostics	Adapted dictionary free orthogonal matching pursuit	Robust, strong adaptability, effective in analysis
[130]	Incipient defect detection in rolling bearing	WT + Resonance-based sparse signal decomposition	Better fault feature recognition ability than regular RSSD
[131]	Fault diagnostics of rolling bearings	Segment tensor rank decomposition	Extract the fault characteristic frequency accurately, excellent data compression capability
[132]	Bearing fault diagnostics in variable speed conditions	Acoustic spectral imaging	Invariant to fluctuations of shaft speed
[133]	Assessment of remaining useful life of REB	WPT + ANN	Better accuracy in the prediction of remaining useful life
[134]	Automated bearing FD	Self-normalizing CNN	The fault detection rate is faster than traditional CNN

Data preprocessing (Feature extraction)

Data	Application
Fast Fourier Transform (FFT)(Ave., Normalized)	Long-term Variation of Overall Frequency
Short-Time Fourier Transform (STFT)	Short-term Frequency Fluctuation
Mel-frequency cepstrum Component(MFCC) (Ave.)	Feature extraction using a filter based on the Mel scale, which closely approximates human auditory characteristics (relatively low-frequency emphasis)
Mel-frequency cepstrum Component(MFCC) (frame-by-frame)	Short-term Fluctuation of MFCC
Wavelet transform(Ave.)	High-pass (mother wavelet function) extracts high-frequency features, while low-pass (scaling function) extracts low-frequency components (relatively robust to high frequencies, capable of capturing not only frequency but also temporal structure).
Wavelet transform(frame-by-frame)	Short-term Variation of Wavelet Coefficients
FFT+MFCC+Wavelet (Ave.)	Correlation Among Various features (Long term)
STFT+MFCC+Wavelet (frame-by-frame)	Correlation Among Various features (Short term)

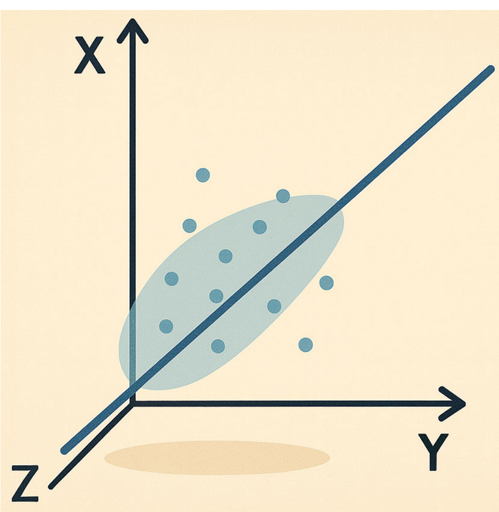
Data preprocessing and learning were carried out by the workstation.
(Ryzen threadripper 3970X 32core, 64 thread, 64GB memory)

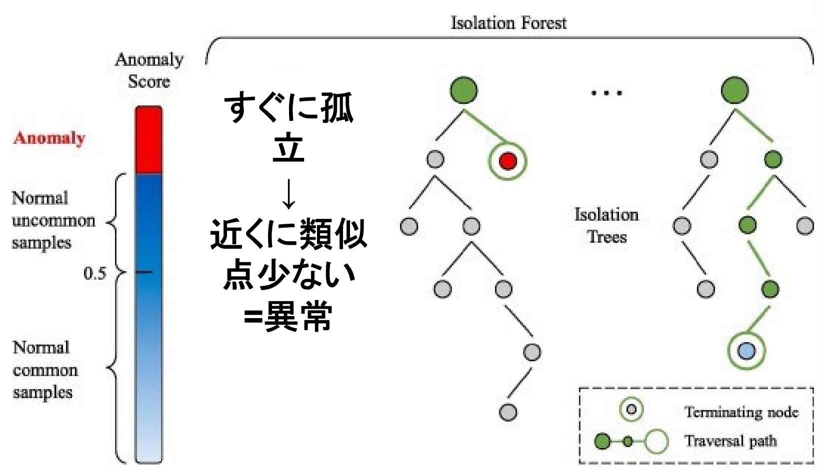
Data preprocessing (Feature extraction)(Cont’d)

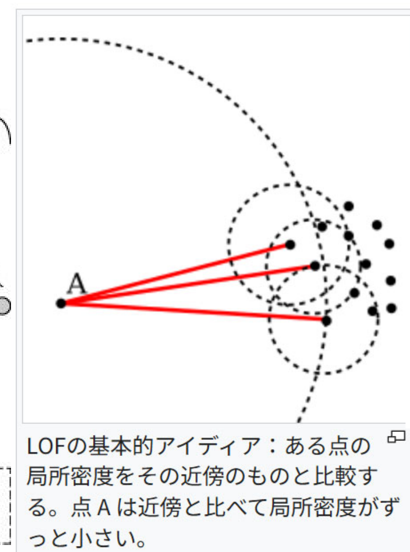
- Describe each data processing step in more detail:
- FFT… Applied the np.fft.fft function from numpy to the entire WAV data and removed frequencies beyond the Nyquist frequency (half of 44.1kHz = 22.05kHz; 22,050 data points). Resampling to 4096 points and normalized by the sum of all 4096 points values, then scaled by multiplying by 10^6. (Default : Rectangular Window)
- STFT... Used scipy's signal.stft. One frame is 2048 points (approximately 0.04644 seconds at 44.1kHz; data points are 2048/2+1 (DC component) per Nyquist's theorem). Overlapping 512 points results in 288 frames for a 10-second WAV file. (Default : HannWindow)
- MFCC … Calculated using librosa's feature.mfcc function. Frequency domain divided into 40 filters (default division: 12 filters up to 1kHz, 20 filters up to 2kHz, dense on the low-frequency side). MFCC are calculated per frame, and added RMS of the wave amplitude to 40 MFCC values (41 elements in total). For the entire 10-second average data, add 7 statistical quantities for 41 frame components: mean, standard deviation, median, max/min, kurtosis, skewness. Total 287 elements.
- Wavelet ... Installed and used pywt. For the high-frequency side, used the mean, standard deviation, kurtosis, skew, energy (amplitude RMS), and Shannon entropy of each 5 components. Same statistical values are calculated by the residual low-frequency side. (36 data points in total). Processed all data and each frame individually using pywt.wavedec.
- For the average dataset (FFT+MFCC+Wavelet), FFT was reduced to 120 dimensions and MFCC to 100 dimensions, compressing the total data points to 256.
- For frame-by-frame data (FFT+MFCC+Wavelet), only STFT was compressed to 100 dimensions, resulting in a total of 177 data points.

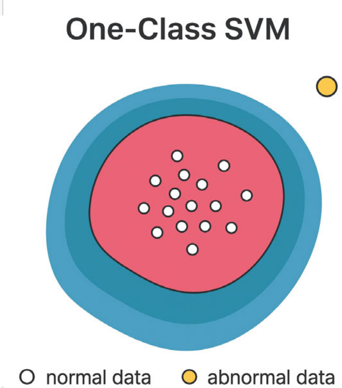
Analysis Model

- There are also many models on sound-based anomaly detection—what models should we focus on?"
- We selected five models to analyze the features.
  1. Principal Component Analysis(PCA)
  2. Auto Encoder(AE)
  3. Isolation Forest(IF)
  4. Local Outlier Factor(LOF)
  5. One-Class Support Vector Machine(OCSVM)
- PCA and AE are used to learn the latent feature space and reconstruction
- IF,LOF,OCSVM learn the distribution or boundary of normal data, then calculate a deviation from it.







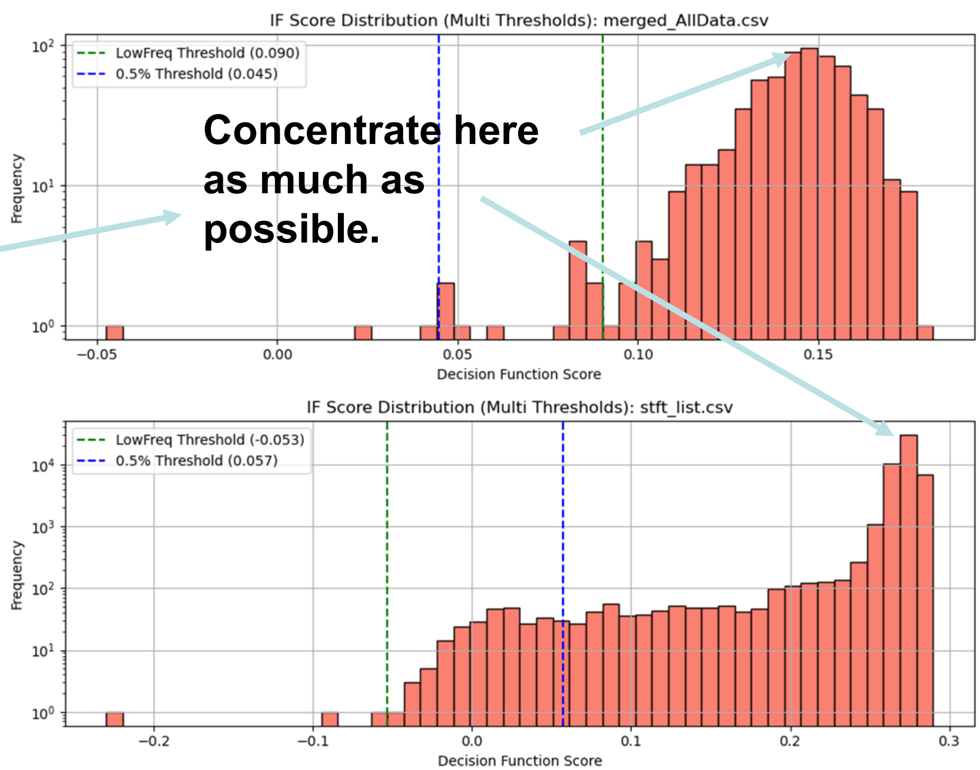


Hyper parameter search

- There are 669 WAV files for training. Assuming all are valid, training is performed using only the valid data.
- Each algorithm first performs hyperparameter search using **Optuna**, then trains using the hyperparameters optimized during that process.

（Optuna... employs TPE (Tree-structured Parzen Estimator) as its optimization algorithm. It estimates the probability distributions of “good regions” and “bad regions” using a Gaussian Mixture Model based on past trial results, prioritizing exploration of more promising parameters）

- Optuna's hyperparameter search was optimized according to the following principles:
- **For PCA**: Number of components achieving **95% contribution rate**
- **For OCSVM, LOF, and IF**: Metric aiming to create a model where **the score distribution of normal data is “high and narrow”**
- **For AE**: Minimized **the standard deviation + average of val_loss** across epochs
- **These guidelines assume that “all training data is normal and highly uniform and stable.”**

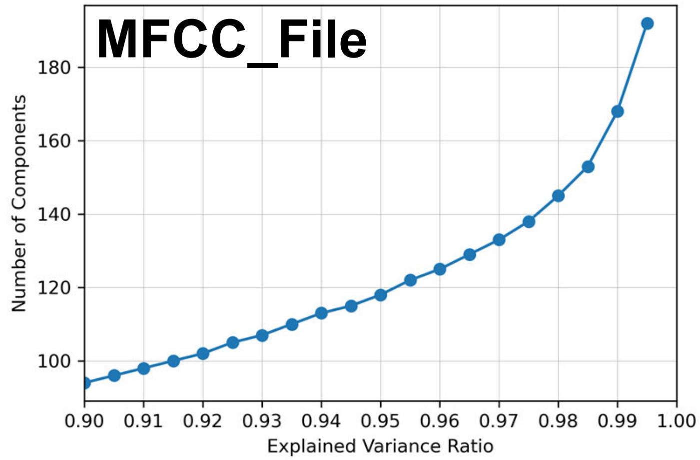


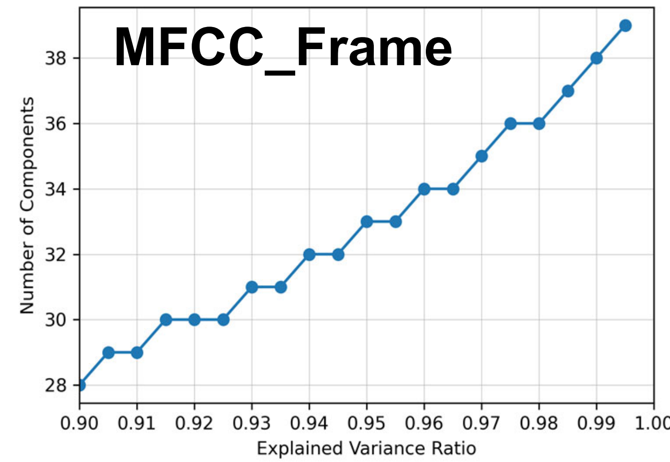
Parameters

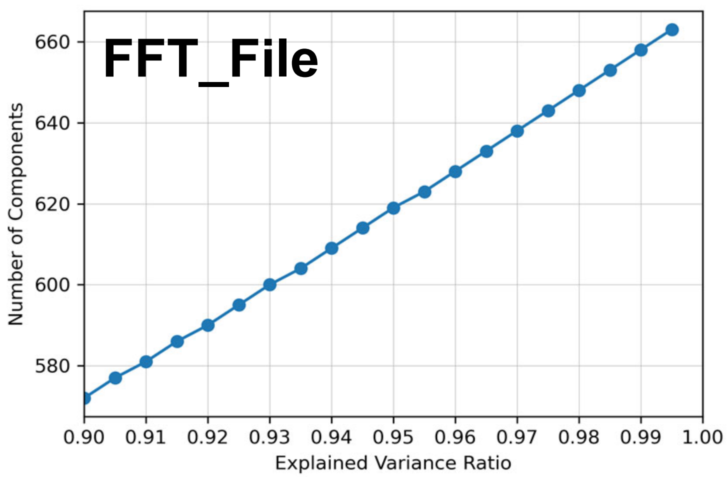
Hyperparameters searched by Optuna.

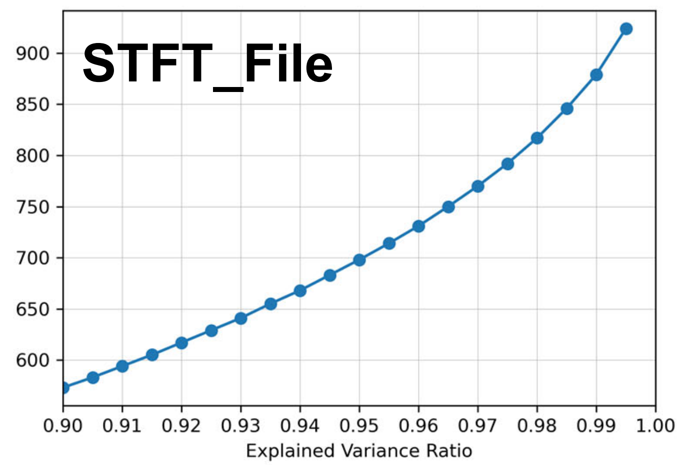
- OCSVM:nu, gamma, tol
- LOF:n_neighbors, leaf_size, metric, contamination
- IF: n_estimators, max_samples, max_features , contamination
- AE:activation_function, latent_dim, hidden_dim, n_hidden_layers, batch_size, final_epoch, lr

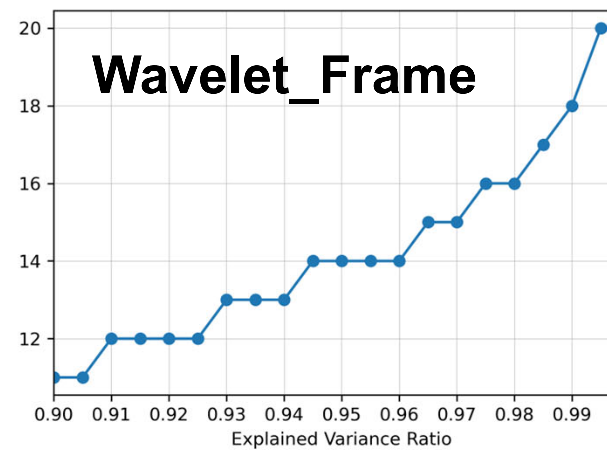
PCA Results

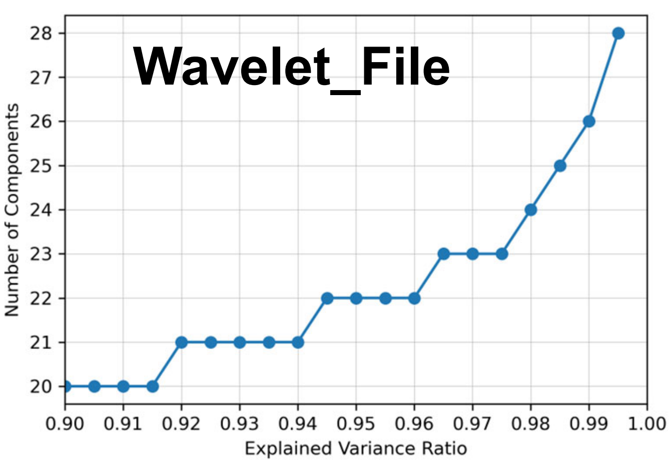

MFCC_File
Number of Components
100
120
140
160
180
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 1.00
Explained Variance Ratio

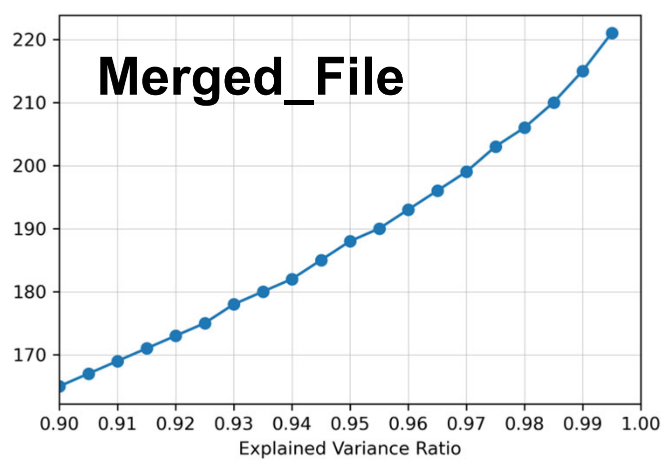

MFCC_Frame
Number of Components
28
30
32
34
36
38
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99
Explained Variance Ratio

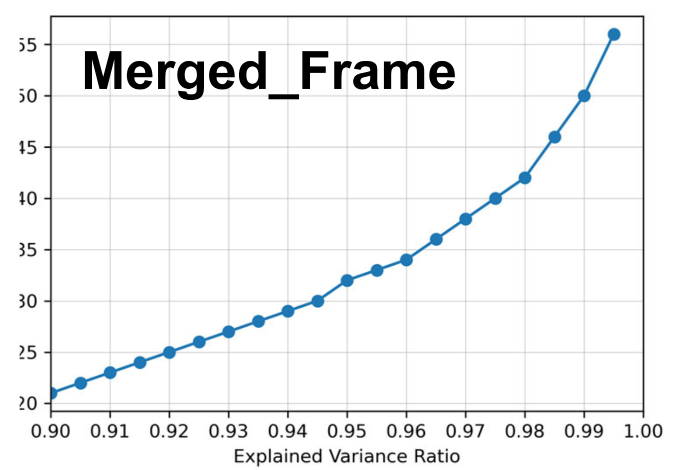

FFT_File
Number of Components
580
600
620
640
660
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 1.00
Explained Variance Ratio


STFT_File
600
650
700
750
800
850
900
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 1.00
Explained Variance Ratio


Wavelet_Frame
12
14
16
18
20
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99
Explained Variance Ratio

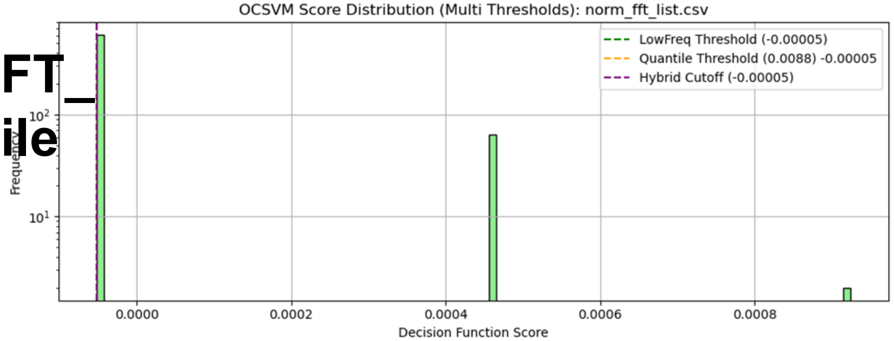

Wavelet_File
Number of Components
20
21
22
23
24
25
26
27
28
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 1.00
Explained Variance Ratio


Merged_File
170
180
190
200
210
220
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 1.00
Explained Variance Ratio

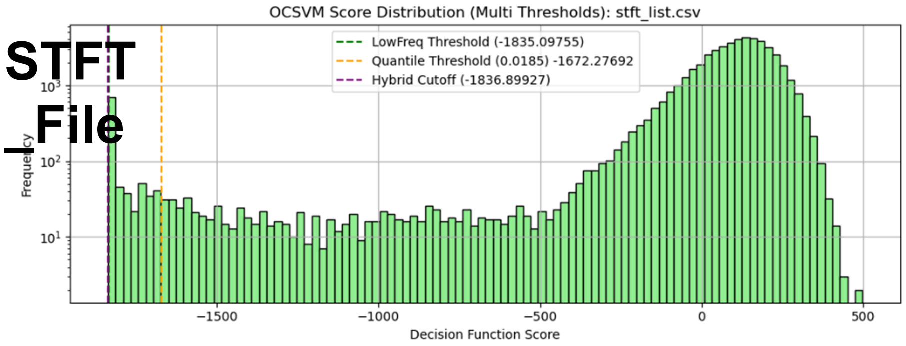

Merged_Frame
0.90 0.91 0.92 0.93 0.94 0.95 0.96 0.97 0.98 0.99 1.00
Explained Variance Ratio

OCSVM Results

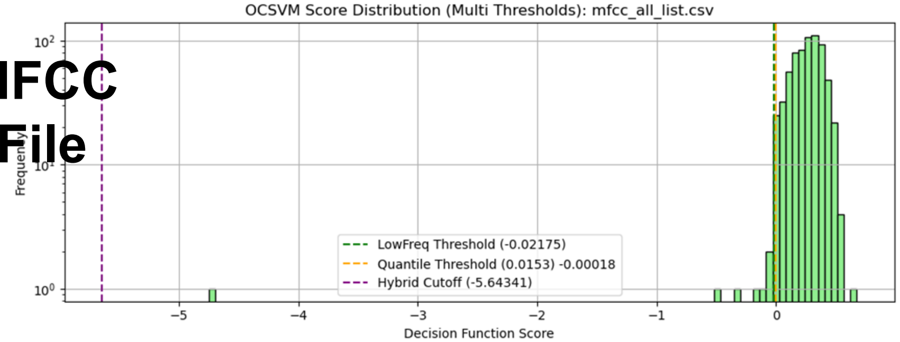
FFT_File



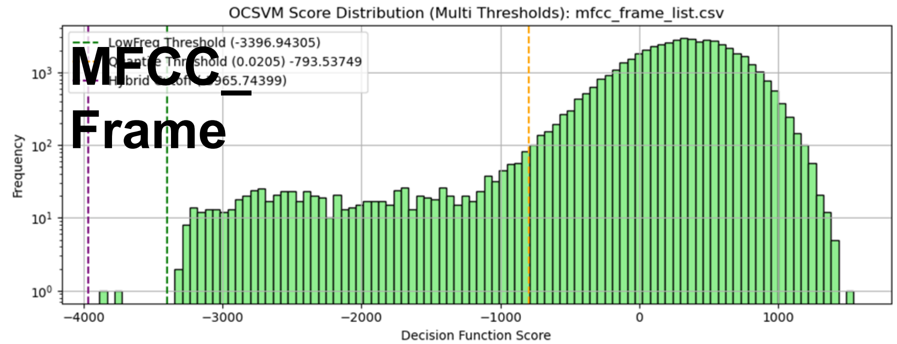
STFT_File



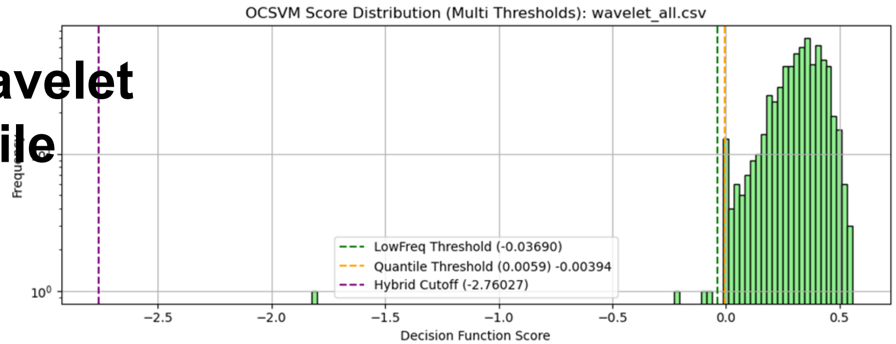
MFCC_File



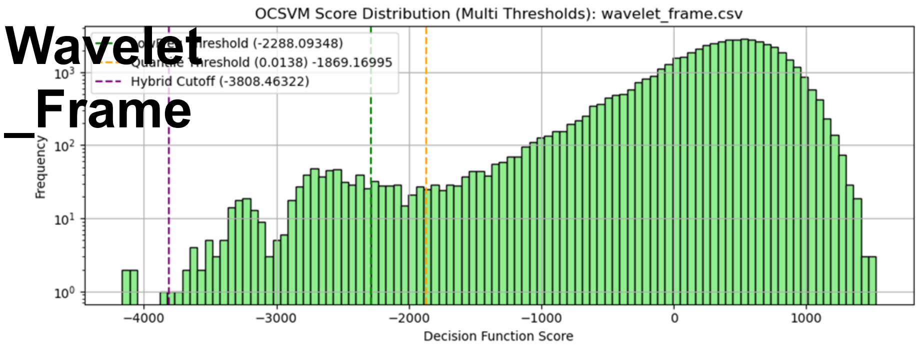
MFCC_Frame



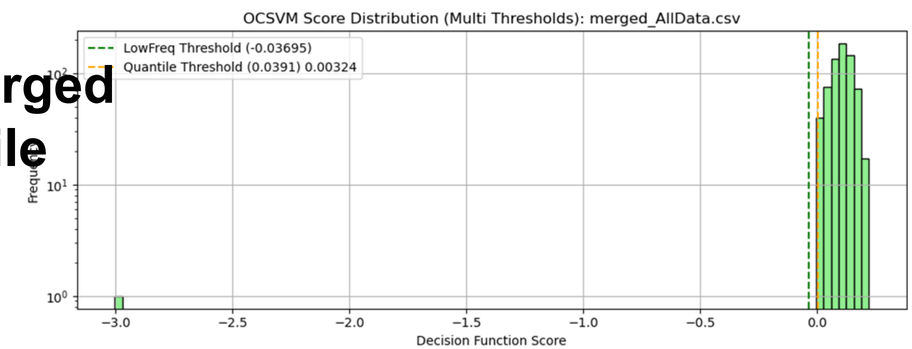
Wavelet_File



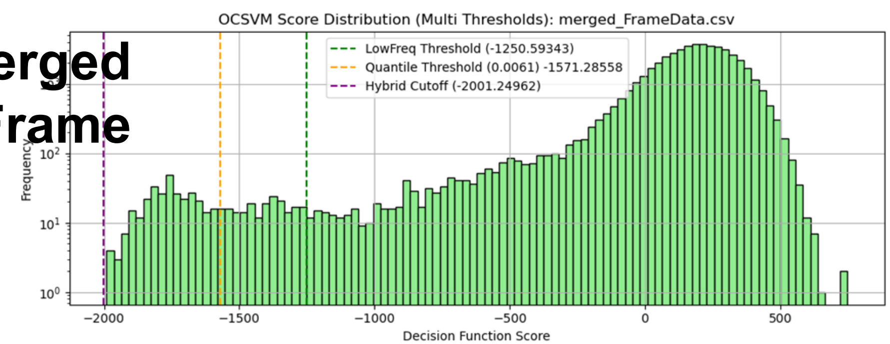
Wavelet_Frame



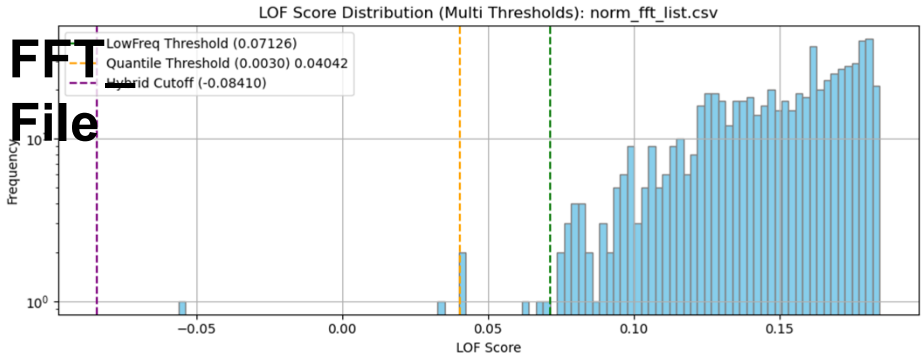
Merged_File

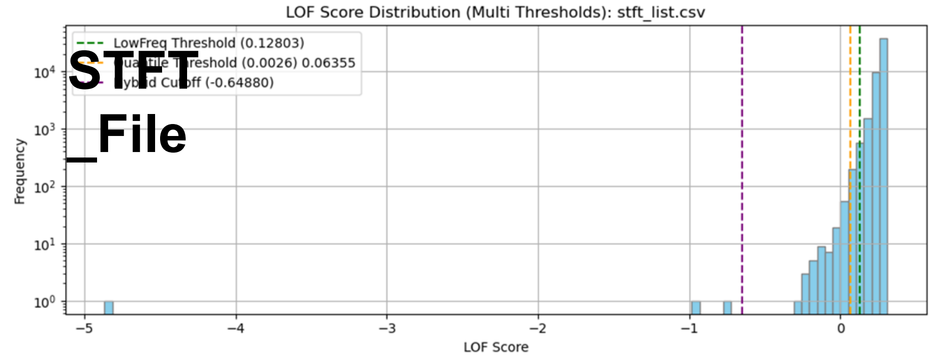


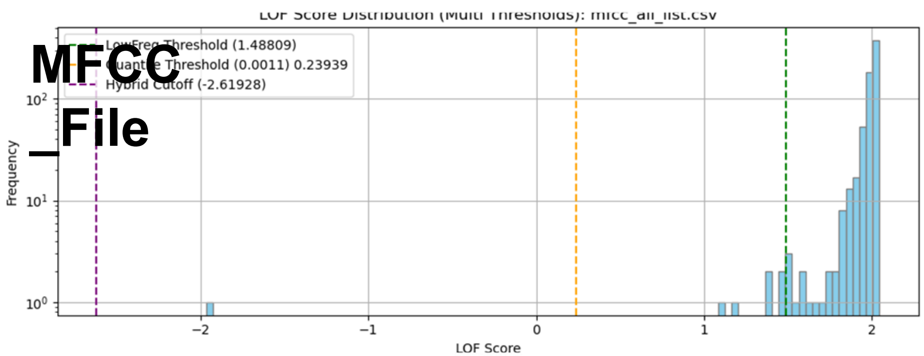
Merged_Frame

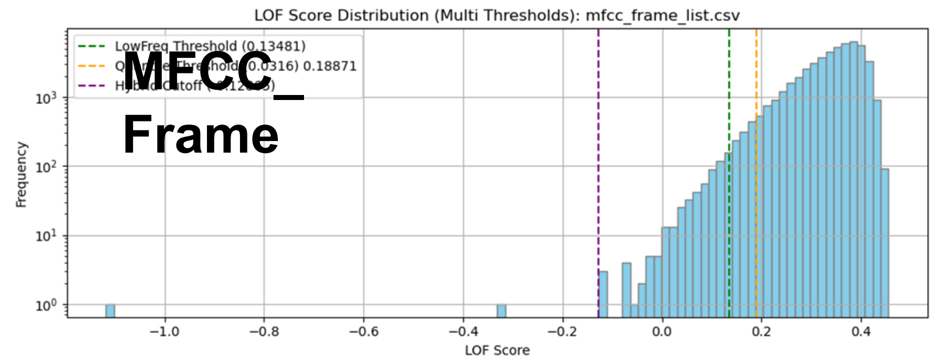


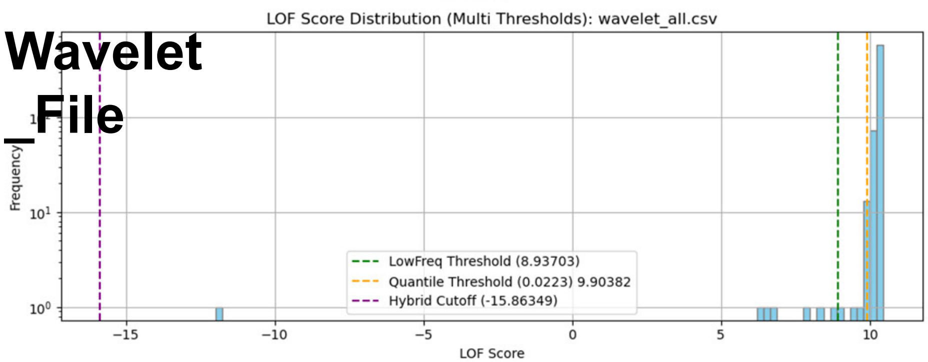
LOF Results

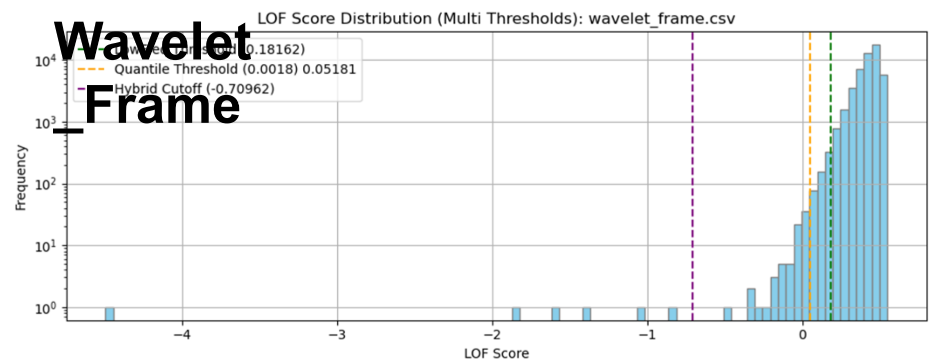

FFT_File
LOF Score Distribution (Multi Thresholds): norm_fft_list.csv
LowFreq Threshold (0.07126)
Quantile Threshold (0.0030) 0.04042
Hybrid Cutoff (-0.08410)
Frequency
LOF Score

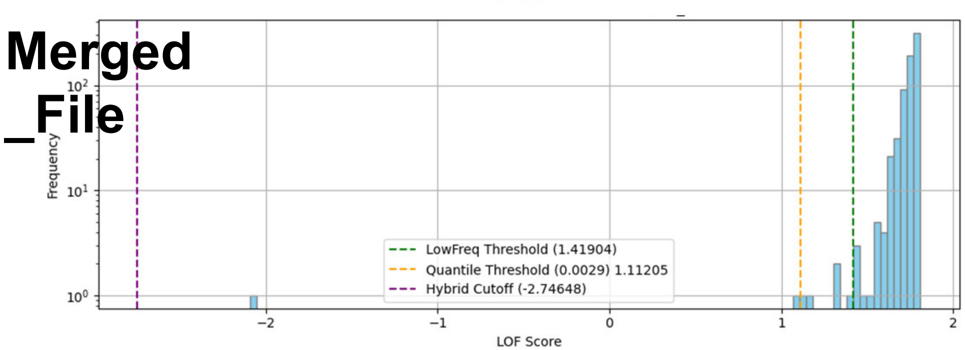

STFT_File
LOF Score Distribution (Multi Thresholds): stft_list.csv
LowFreq Threshold (0.12803)
Quantile Threshold (0.0026) 0.06355
Hybrid Cutoff (-0.64880)
Frequency
LOF Score

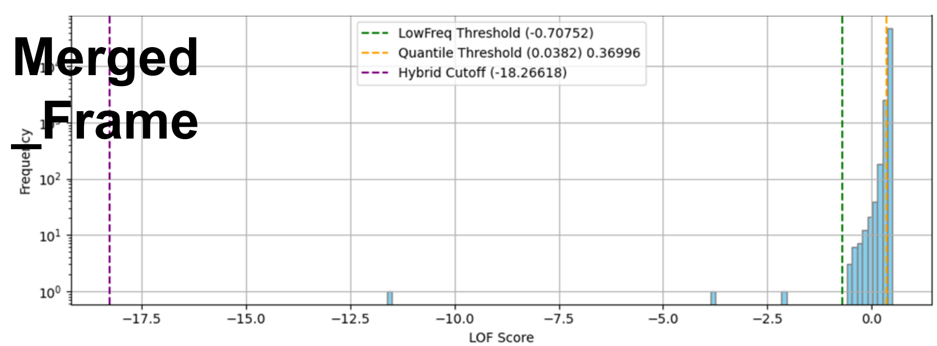

MFCC_File
LOF Score Distribution (Multi Thresholds): mfcc_all_list.csv
LowFreq Threshold (1.48809)
Quantile Threshold (0.0011) 0.23939
Hybrid Cutoff (-2.61928)
Frequency
LOF Score


MFCC_Frame
LOF Score Distribution (Multi Thresholds): mfcc_frame_list.csv
LowFreq Threshold (0.13481)
Quantile Threshold (0.0316) 0.18871
Frequency
LOF Score


Wavelet_File
LOF Score Distribution (Multi Thresholds): wavelet_all.csv
LowFreq Threshold (8.93703)
Quantile Threshold (0.0223) 9.90382
Hybrid Cutoff (-15.86349)
Frequency
LOF Score

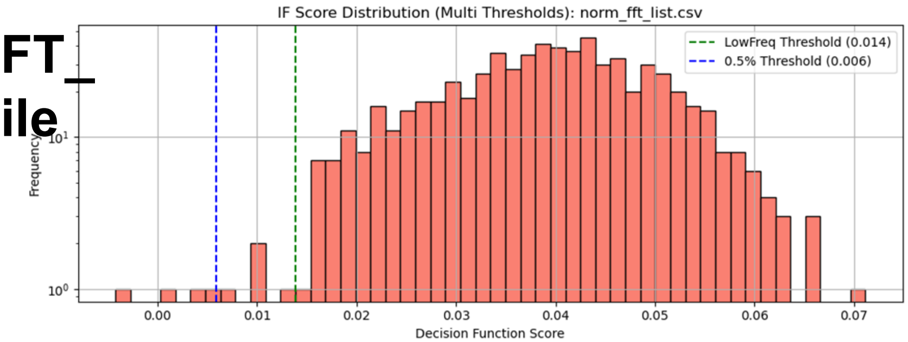

Wavelet_Frame
LOF Score Distribution (Multi Thresholds): wavelet_frame.csv
Quantile Threshold (0.0018) 0.05181
Hybrid Cutoff (-0.70962)
Frequency
LOF Score


Merged_File
LowFreq Threshold (1.41904)
Quantile Threshold (0.0029) 1.11205
Hybrid Cutoff (-2.74648)
Frequency
LOF Score

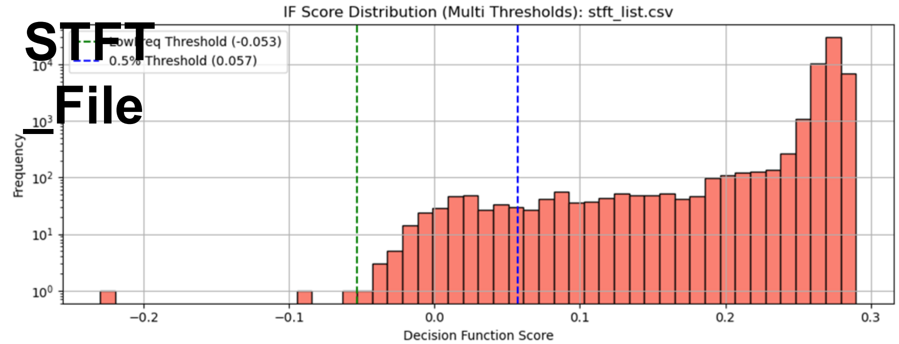

Merged_Frame
LowFreq Threshold (-0.70752)
Quantile Threshold (0.0382) 0.36996
Hybrid Cutoff (-18.26618)
LOF Score

IF Results

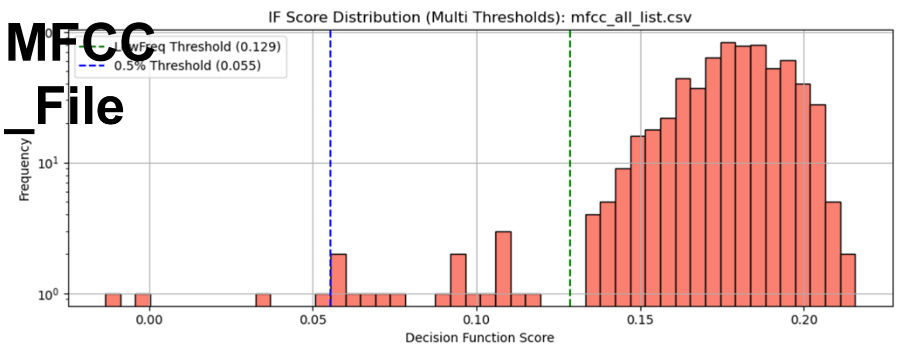
FFT_File



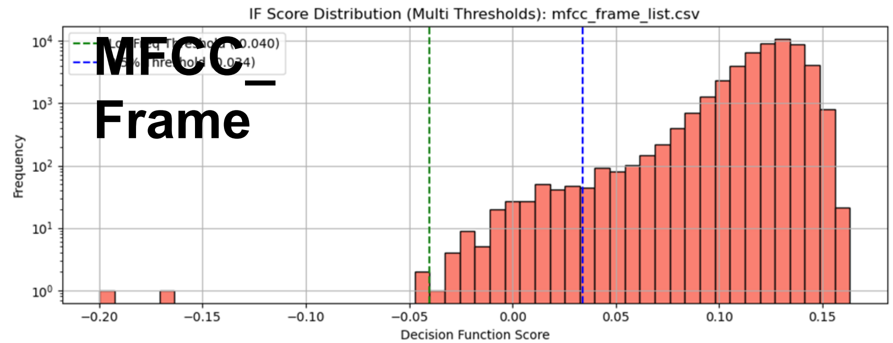
STFT_File



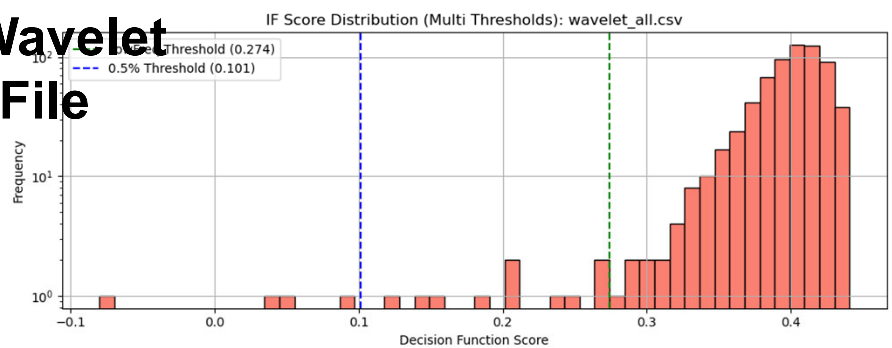
MFCC_File



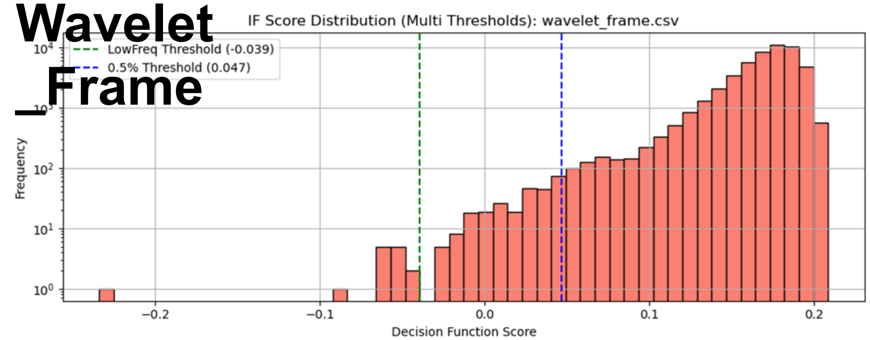
MFCC_Frame



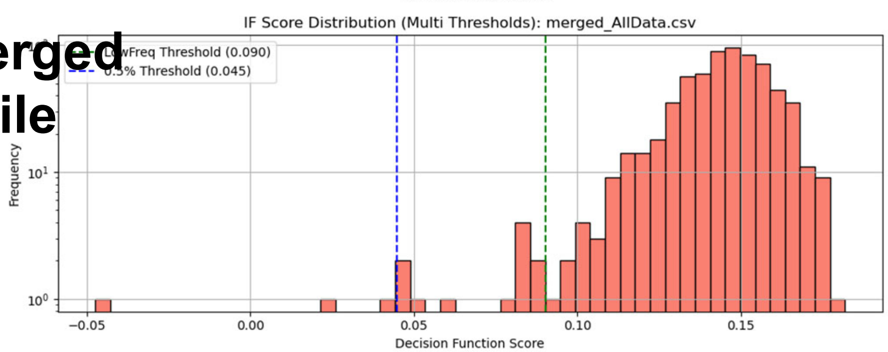
Wavelet_File



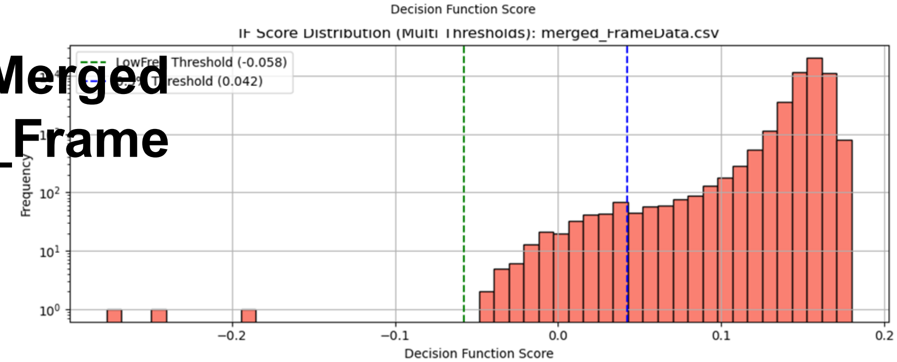
Wavelet_Frame



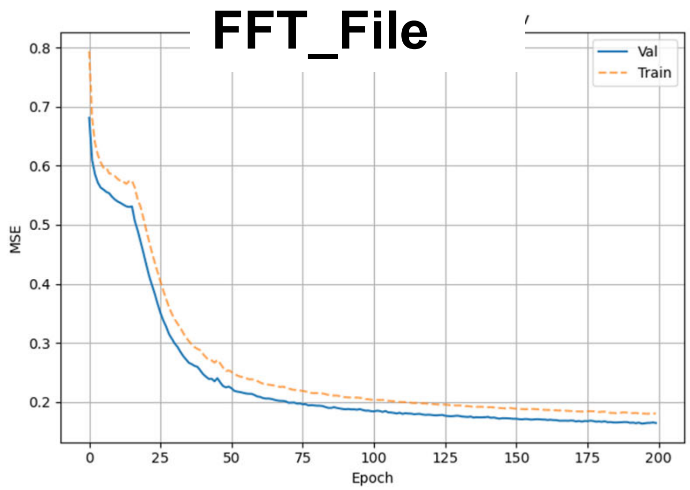
Merged_File

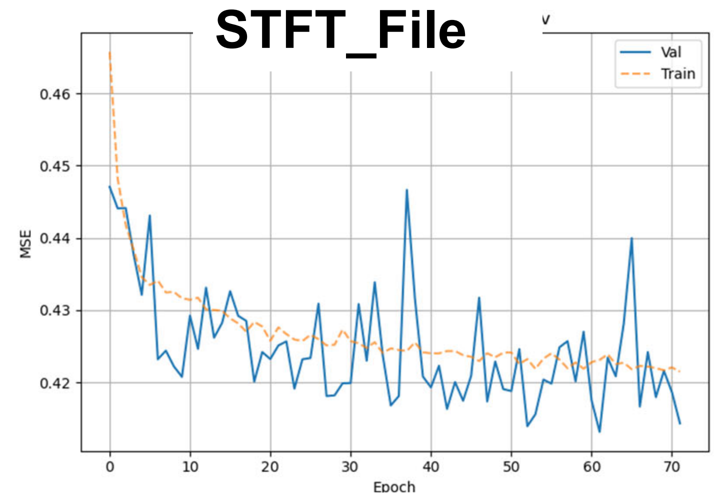


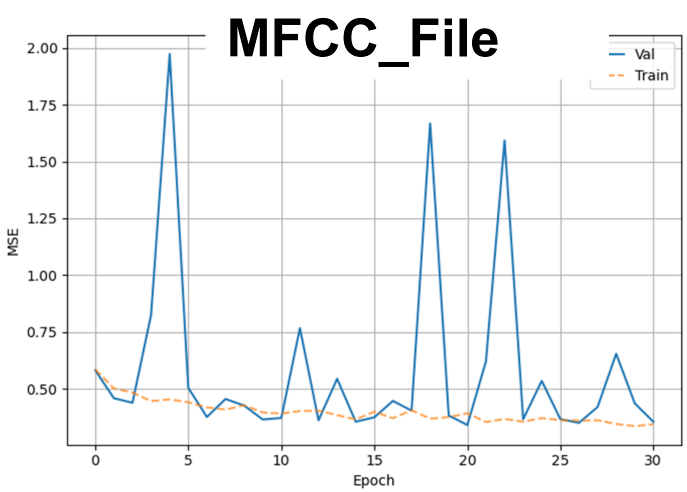
Merged_Frame

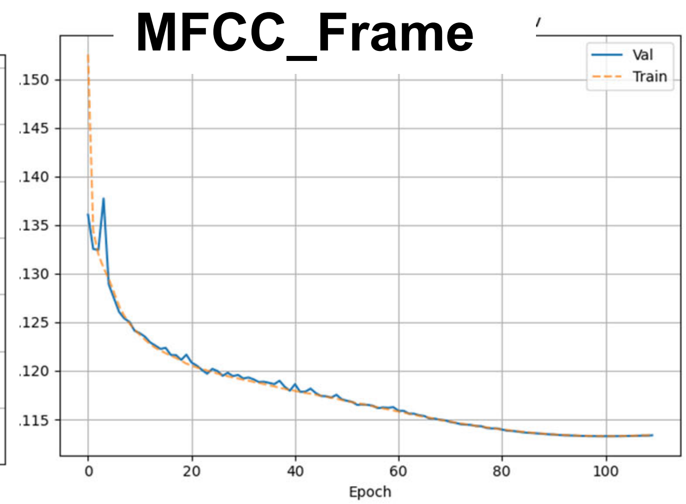


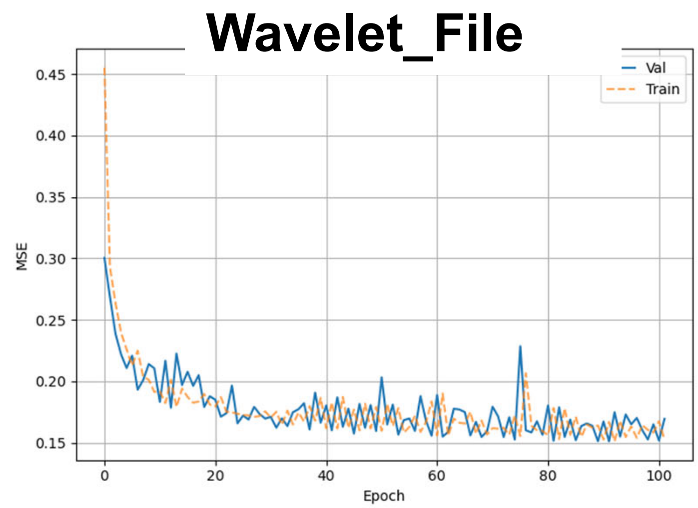
AE Results

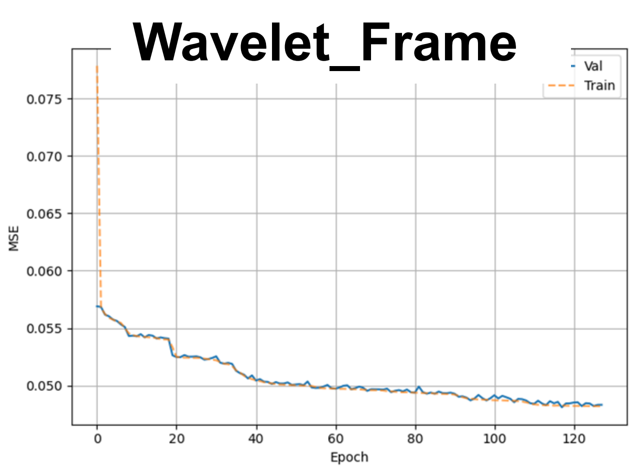

FFT_File
Val
Train
MSE
Epoch

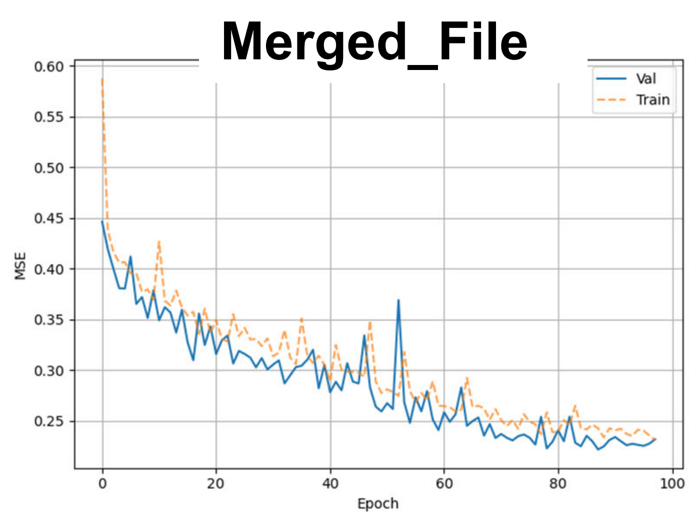

STFT_File
Val
Train
MSE
Epoch

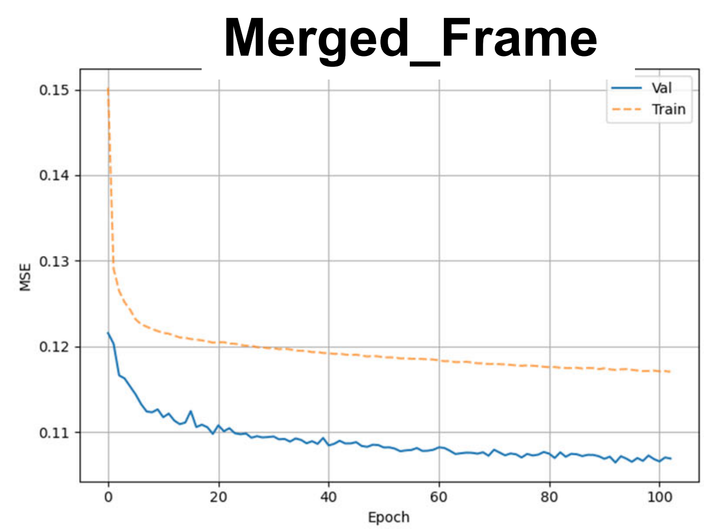

MFCC_File
Val
Train
MSE
Epoch


MFCC_Frame
Val
Train
Epoch


Wavelet_File
Val
Train
MSE
Epoch


Wavelet_Frame
Val
Train
MSE
Epoch

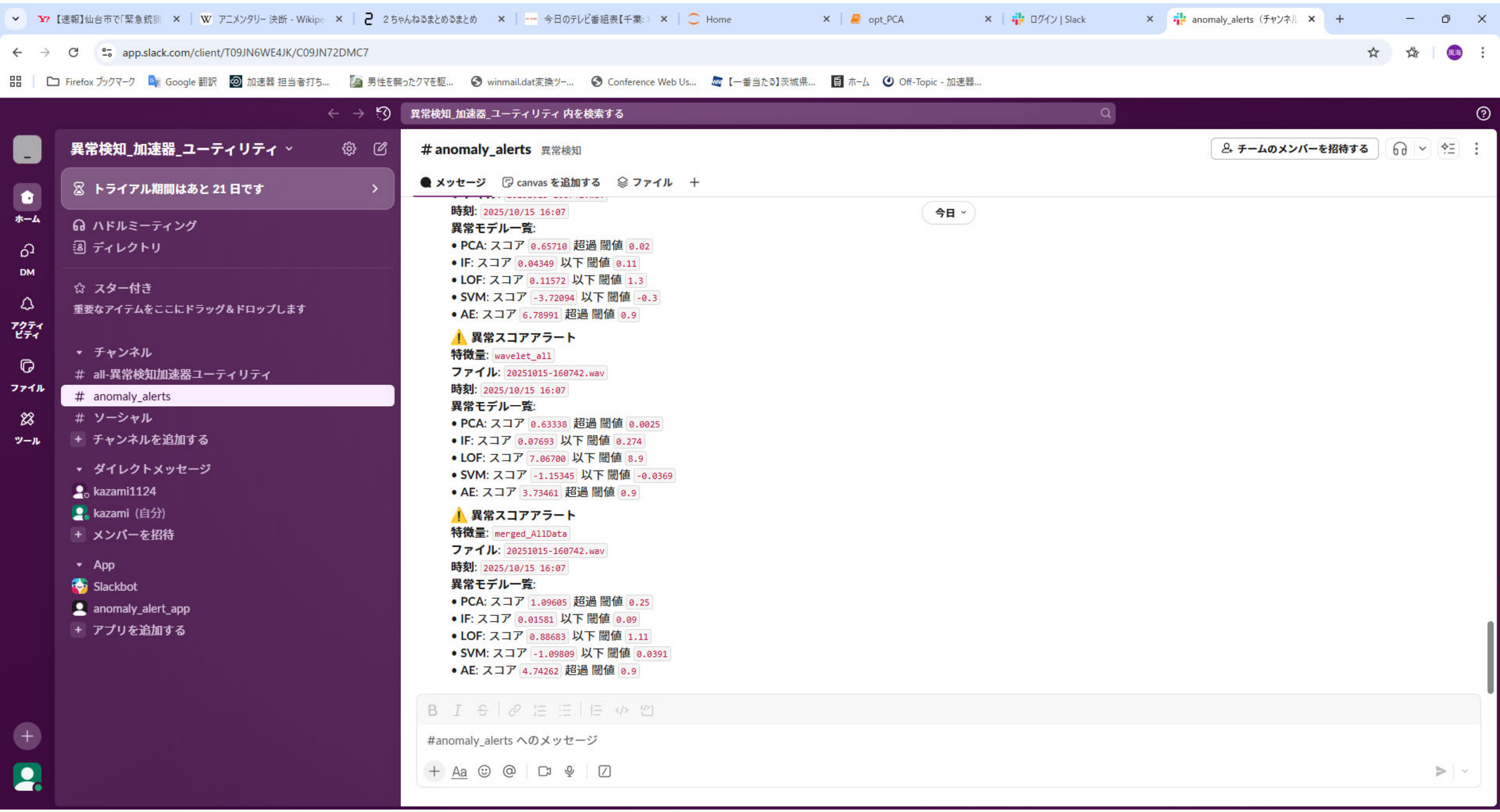

Merged_File
Val
Train
MSE
Epoch


Merged_Frame
Val
Train
MSE
Epoch

Trial of anomaly detection using Trained Model

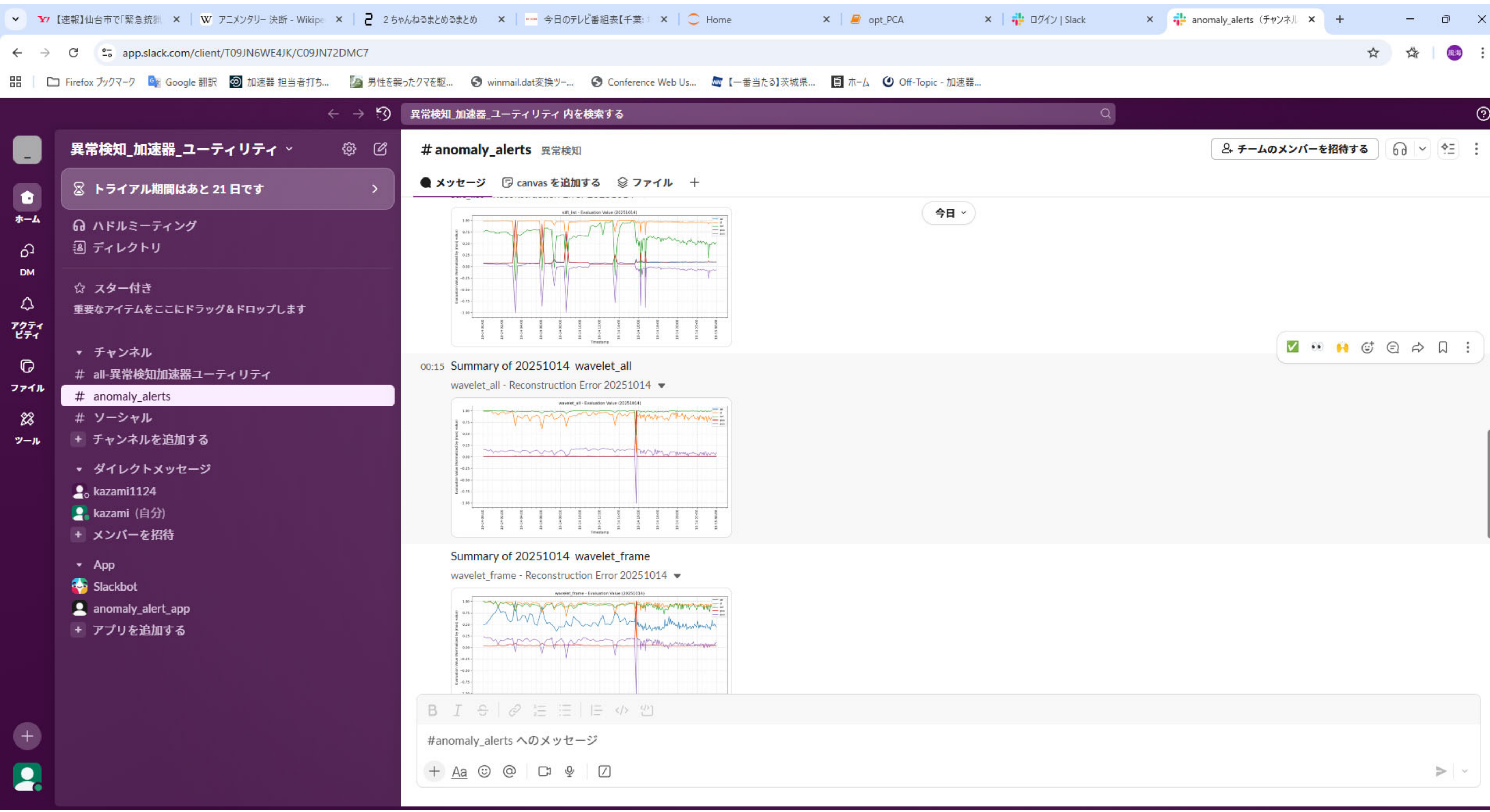
- Use the **model files trained for each algorithm (AE: .pth, others: .pkl)**, the **preprocessing models (Scalers, etc.)** and the **limit values** along with the diagnostic code to the note PC.
- The Python code for anomaly detection consists of:
  A) Recording section
  B) Preprocessing section
  C) Evaluation of the scores based on each learned model
  D) Determination of normal/anomaly flag
  E) Log results and notify Slack.
- A windows batch file runs the Python code at regular intervals.
- **Two kind of thresholds** were prepared: **the soft threshold (closer to the central score) triggers only logging**, while **the hard threshold (further from the central score) reports an anomaly to Slack.**
- **All features and evaluation values** calculated for each processing step are **saved in Excel**. The number of triggered data is also logged.
- For PCA and AE, the top 10 components with the largest errors are recorded with those values.

Report on Slack



Daily Report

- Created a daily report program. It runs only once a day, after midnight.
- Read the evaluation values calculated for each process from Excel, plot the daily trend, and send them to Slack.

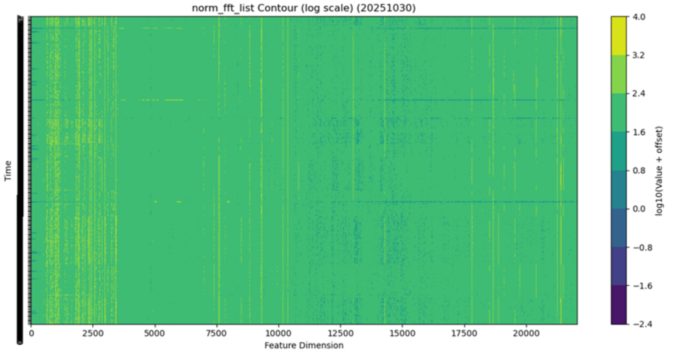


Daily Report(Cont'd)

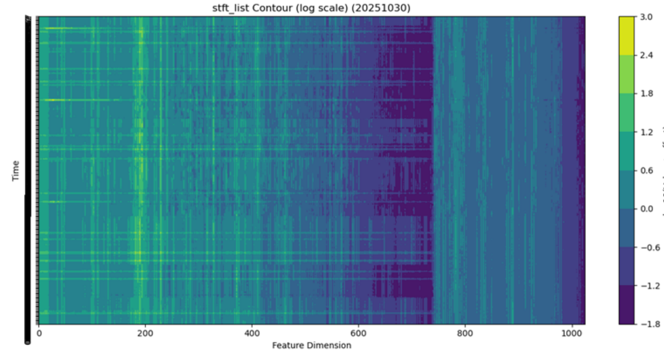
J-PARC

- **Daily report also makes the contour plots of extracted feature trends.**

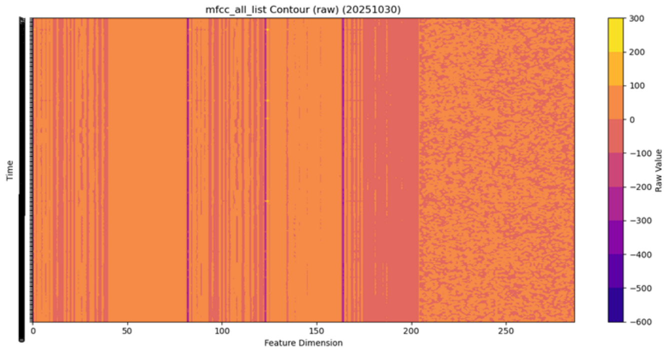
FFT_File



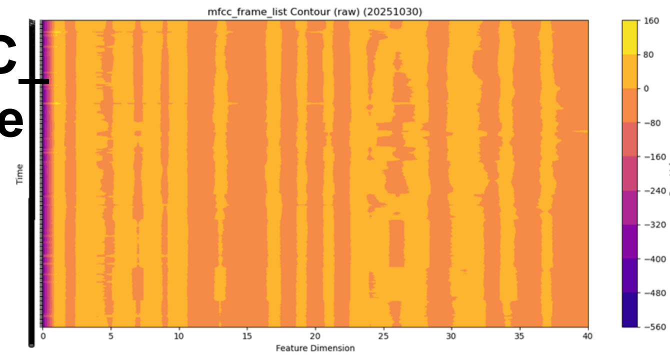
STFT_File



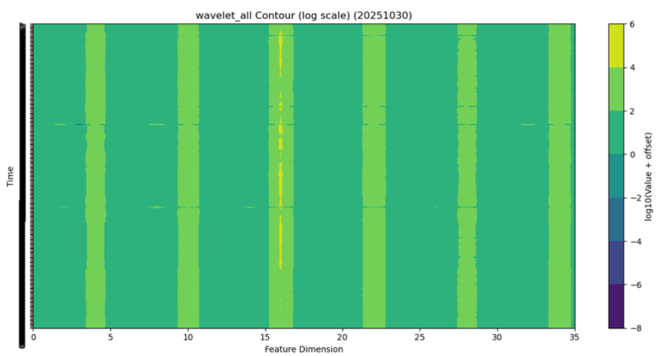
MFCC_File



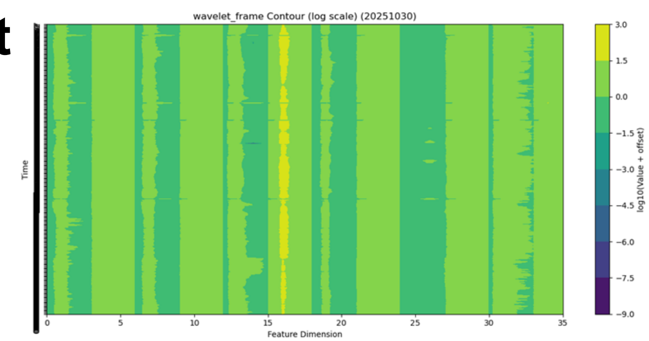
MFCC_Frame



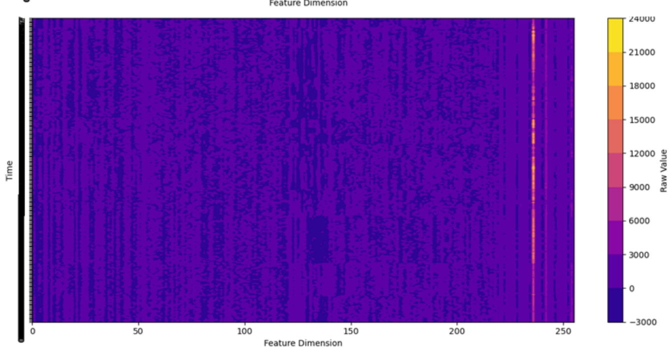
Wavelet_File



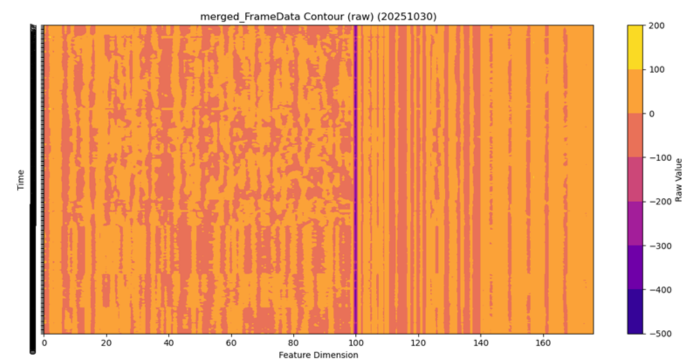
Wavelet_Frame



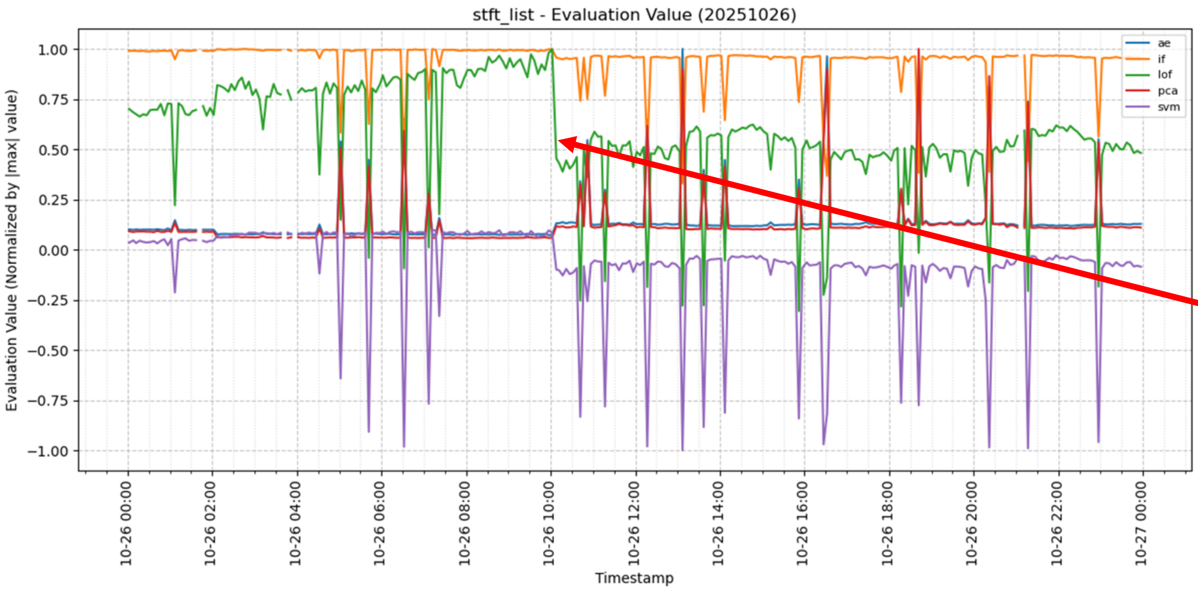
Merged_File

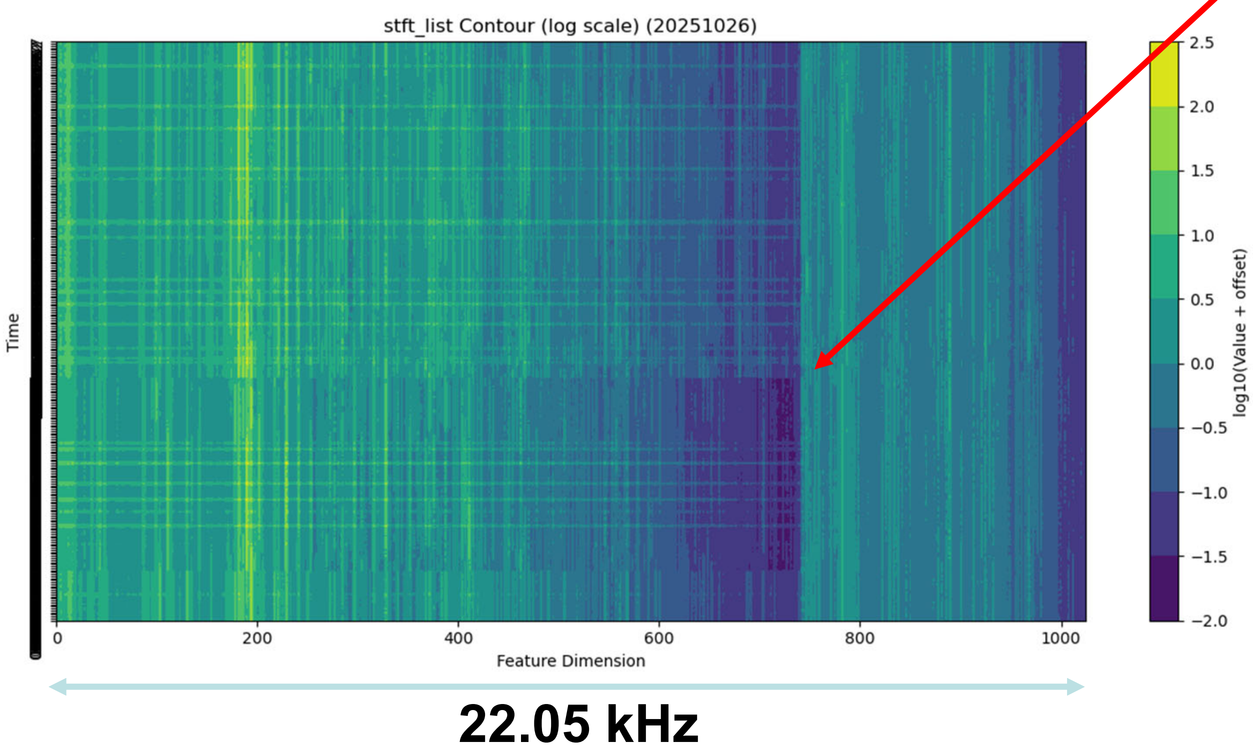


Merged_Frame



Preliminary result





- This data was on 10/26. The baseline for each score fluctuated around 10:00.

(Spikes indicate broadcast)

- The STFT contours shows changes in the low-frequency spectrum (16 kHz and below).
- This is due to switch the operation mode of the air conditioning system, so retraining is underway using data that includes this effect.

Summary

- We have started the development of the anomaly-detection system and initiated its operation.
- By copying all required components—the anomaly-detection code, daily reporting scripts, trained models, and related files—to a Windows machine with Python installed, and registering a batch file with the task scheduler, the system can be executed in any environment.
- Since beginning operation, several findings have emerged:

  - The dataset based on 10-second averaged values contained only 669 training samples. After allocating 10% for validation (as done during training), the remaining data was clearly insufficient for effective learning. We are currently collecting data at 5-minute intervals and have been retraining the models.

  - Some feature–algorithm combination exhibits “bad food combination”.

  - Broadcast audio has been included in the input stream, which has resulted in a large number of anomaly alerts whenever such audio is present.
- For the time being, we will continue collecting data from the linac equipment room and monitor system behavior.

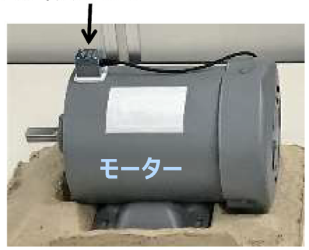
Future Work

Include another sensors (Acoustic Sensor / Odor sensor)

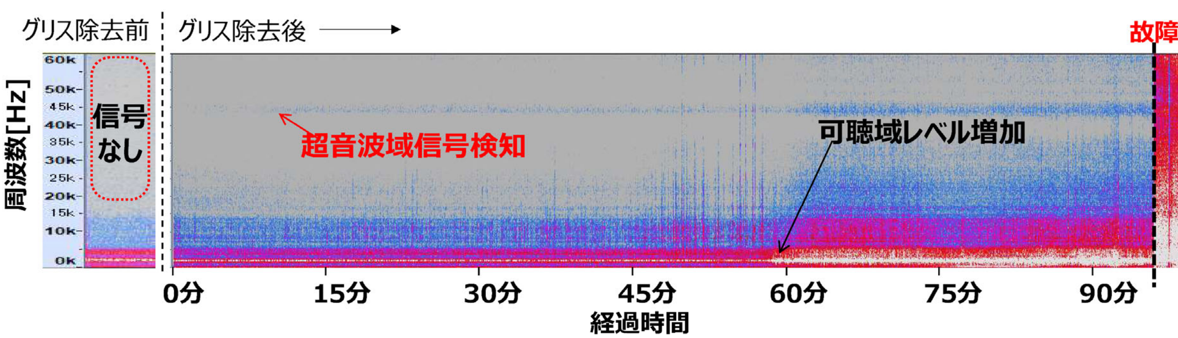
アコースティックセンサー

超音波モニタリングの有効性検証

◆ 実験方法
(寿命を短縮するために)モーター軸受け内のグリスを抜きベアリング損傷と摩耗過程の動作音をアコースティックセンサーで測定



◆ 結果（スペクトラム遷移）



可聴域レベル増加の前に超音波域で予兆信号を検知している。
より早い予兆検知には**超音波域モニタリング**が有効

測定概要

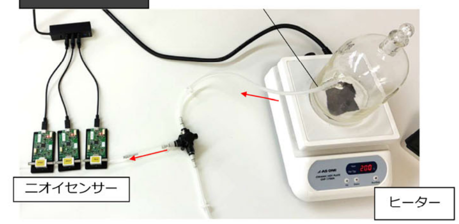
目的：
ニオイセンサーによる発火前の異常検知可能性検証

測定対象：
塩ビケーブル, P板, 木片

測定条件

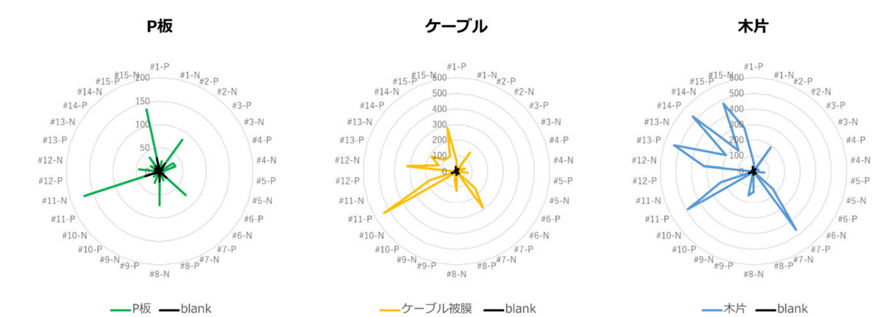
使用センサー	：	5C-SSM (FS0101, FS0200, FS0300)
サンプル	：	P板, 木片, 塩ビ被覆ケーブル
加熱温度	：	300℃ (ホットプレート)

計測環境



解析結果例：レーダーチャート (Potential difference)

@300℃



サンプルごとに異なる応答パターン → 異なる焦げ臭成分を示唆