American University of Sharjah

Fall 2023

College of Engineering

Department of Computer Science and Engineering

COE 476 - Neural Networks and Deep Learning

Detection of Anomalous Sounds for Machine Condition Monitoring

Ahmad AlSaleh - b00093749

Yousef Irshaid- b00093447

Hussain Saif - b00088793

Date of Submission: Nov 23, 2023

Submitted to:

Dr. Imran Zualkernan

Mr. Ali Reza Sajun

Table of Contents

0. Table of Contribution

Team member AUS ID	Team member name	Team member's contribution to the project	Percent Effort (out of a total of 100%)
93749	Ahmad Alsaleh	Writing report, running models, YouTube video 10 lit review Writing parts 6 and 7	33%
93447	Yousef Irshaid	Running models and testing 10 lit review Writing parts 4 and 8	33%
88793	Hussain Saif	Writing report for parts1, 3, 5, 10 lit review. Running models.	33%

1. Description of the Problem [/ 10]

Anomaly detection (AD) is the process of identifying patterns, events, or observations that deviate significantly from the expected or normal behavior. The goal of anomaly detection is to highlight abnormal occurrences that may indicate potential problems, errors, or interesting insights about the underlying system. However, since it is almost always impossible to define the meaning of abnormality, anomaly detection systems should be trained on normal samples only, making it an unsupervised problem. That being said, anomaly detection can be applied to numerous industries such as healthcare for health monitoring or in financial transactions to detect fraud. However, the focus of this paper is Anomalous Sound Detection (ASD), particularly for machine sounds using the Malfunctioning Industrial Machine Investigation and Inspection (MIMII) sound dataset. ASD for machine sound monitoring is an interesting field of anomaly detection since it can help in predicting failures in industrial machines, thereby minimizing the costs of periodic machine maintenance in factories. Many intrinsic complexities and challenges are faced by AD in general due to the unique nature of such anomalies. Namely, these challenges are the class imbalance that would result from the infrequency of anomalies used for testing AD systems, as well as the unknownness of these anomalies. These two challenges make a completely supervised solution impractical since the collection of labeled data for AD is infeasible.

2. Previous Approaches [/ 10]

The table below summarizes papers that relate to solving anomaly detection in time series data

	Year	Features	Strengths	Weaknesses
[14]	2022	The survey paper details 6 types of anomaly detection methods: Forecasting: The method learns normal data and predicts the data ahead and compares it which is used to calculate an anomaly score. Reconstruction: The normal behavior is encoded to a latent space and test series are reconstructed from the latent space and compared to the original. Anomalies won't be able to be reconstructed in this way Encoding: similar to reconstruction but they calculate anomaly score based on test data representation in latent space Distance methods: uses specialized distances to find anomalies, Distribution methods: Learns the distribution of the data and anomalies and detected based on their fit of the distribution Isolation trees: builds an ensemble of random trees that partition the samples of the test time series. Anomalies are easier to split and are closer to the root node	Not applicable	Not applicable
[15]	2023	The paper reviews the state-of-the-art deep learning techniques for detecting anomalies in multivariate time series. The paper explains the techniques used for anomaly detection in time series in the following: - LSTM is a type of neural network that has a memory cell and three gates (input, output, and forget) that control the information flow in and out of the cell. LSTM can learn long-term dependencies in sequential data, such as time series, by selectively remembering or forgetting the past states. - Autoencoders are neural networks that learn to reconstruct the input data by compressing it into a lower-dimensional latent space. Autoencoders can	Not applicable	Not applicable

		be used for anomaly detection by measuring the reconstruction error between the input and the output. A high reconstruction error indicates an anomaly. - Variational autoencoders are a type of autoencoder that learns a probabilistic distribution of the latent space instead of a deterministic one. They can detect anomalies by measuring the likelihood of the input data under the learned distribution. A low likelihood indicates an anomaly. - Generative adversarial networks are composed of two neural networks: a generator and a discriminator. The generator tries to produce fake samples that resemble the real data, while the discriminator tries to distinguish between the real and the fake samples. The two networks compete with each other and improve over time. Generative adversarial networks can generate realistic and diverse samples from random noise. They can also detect anomalies by measuring the discriminator's output. A low output indicates an anomaly.		
[17]	2022	The paper provides a survey for deep learning methods of anomaly detection. The key methods they presented were as follows: - DeepAnT: A CNN that predicts future values and compares them with actual values to detect outliers. - LSTM: A LSTM network that predicts confidence intervals for future values and uses nonparametric dynamic thresholding to identify anomalies - HTM: A hierarchical temporal memory network that learns the temporal patterns and predicts the probability distribution of the next value to detect outliers. - DAGMM: A deep autoencoding Gaussian mixture model that learns a low-dimensional representation and a mixture density estimation to measure the reconstruction and energy errors for anomaly detection - Donut: A variational autoencoder (VAE) that models the normal patterns of time series and uses a moving window to extract features and reconstruct the input to detect anomalies	Not applicable	Not applicable

[19]	2022	The paper classifies anomaly detection via 2 methods: - Feature extraction: These methods use pre-trained neural networks to extract low-dimensional features from complex data, and then apply traditional anomaly scoring methods on the latent space. Models such as AlexNet, VGG, ResNet are used here to detect the low dimensional features and SVM for classification. This method is good as there are alot of pretrained models to use but the disjoint nature of this method usually leads to suboptimal results - Learning normality feature representations which captures the regularities of normal data by methods such as data reconstruction, generative modeling, predictability modeling, or self-supervised classification. Autoencoders and GANs and common models used for this method. This method is good as these models are very capable to detect anomalies however they are harder to train. - Anomaly measure-dependent feature learning which aims at learning feature representations that are specifically optimized for one particular existing anomaly measure such as distance-based measure, one-class classification measure, or clustering-based measure. This method is well defined and works for high dimensional data but they require high computation	Not applicable	Not applicable
[20]	2017	The paper proposes the model, AnoGAN, a deep convolutional generative adversarial network that learns a normal distribution of images and detects anomalies based on the anomaly score. The anomaly score is a combination of residual and discrimination losses that measures the dissimilarity between an image and the closest image on the learned manifold, A high score indicates an anomalous image. The model used four fractionally strided convolution layers in the	- The model outperforms alternative approaches based on autoencoders or reference discriminatio n loss.	- The model overfits the data and may mis some anomalies

		generator, and four convolution layers in the discriminator, all filters of sizes 5 × 5.		
[22]	2023	The paper proposes the use of graph based autoencoder neural networks (GWAE and GWVAE) for sound anomaly detection. The encoders use multiple spectral graph wavelet convolutional (SGWConv) layers while the decoders are fully connected layers followed by relu activation.	The method captures multiscale temporal relationships in high-frequency data, transforms raw signals into graph representations, and reconstructs features from latent space.	The model relies on kernel density estimation for threshold selection, and do not consider low-frequency signals
[27]	2017	The authors describe deep autoencoder as a feed-forward multi-layer neural network in which the desired output is the input itself. The authors claim that by allowing many layers of encoders and decoders, a deep autoencoder can effectively represent complicated distributions over the input X.	- RDA's demonstrate better results that normal auto encoders	Not applicable
[28]	2019	Stacked Auto Encoders is a deep learning algorithm that can learn the features of normal data and reconstruct them with minimal error. SAE consists of multiple encoder and decoder parts that feed into one another.	- Results in higher accuracy and stability of results	Not applicable
[29]	2018	The authors propose a stack of traditional autoencoder and online sequential extreme learning machine (OSELM) to extract features and classify the health status of bearings based on vibration signals. They reduce the noise and dimensionality of the raw signals by averaging every five samples The authors train a single hidden layer autoencoder with 4096 input and output nodes and 5 hidden nodes to learn a compressed representation of the filtered signals. They use ReLU activation function and Adam optimizer for training. The authors use	- Got 100% detection accuracy with this method on NASA dataset	Not applicable

		an online version of ELM with 5 input nodes, 10 hidden nodes and 1 output node to classify the bearing health based on the features extracted by the autoencoder. They use a boundary mode and a convergence criterion to train the network with healthy samples and then test it with faulty samples.		

The table below summarizes papers using the same dataset as our project

	Year	Features	Strengths	Weaknesses	Metrics
[1]	2019	The paper introduces the MIMII dataset and provides a baseline result using a dense autoencoder with 2 layers in encoder with 64 neurons each, 8 neuron latent space and 2 layers in decoder with 64 neurons reach. RELU activation was used. The encoder and decoder are connected to 320 neuron input and output layers. The preprocessing was done using frame size of 1024, a hop size of 512, and 64 mel filters.	- Simple model	- Trains each machine ID individually which would result in overfitting for that specific machine - Only AUC was reported	AUC: - Fan: 0.67 - Pump: 0.81 - Slider: 0.94 - Valve: 0.90
[2]	2020	A Group Masked Autoencoder that has a fully connected neural network with 9 hidden layers [128, 128, 128, 128, 32, 128, 128, 128, 128]. The input layer has 640 nodes. Last layer has 19200 neurons - 640 input x 10 Gaussian distributions for each input (used to estimate the conditional distribution of each input) x 3 parameters for each distribution (mean,	- Uses image warping techniques to do label augmentation - Uses Group masked autoencoder for density estimation	- Only AUC was provided as a metric - Treats machines of same type differently based on their ID's	AUC - Fan : 70.10, Pump: 75.68, Slider: 93.29, Valve: 89.68

		variance, and mixture component probability).			
[3]	2020	The model has 4 sections, the encoder with 3 hidden layers [128, 64 and 32] with relu activation, the decoder with 4 hidden layers [128, 128, 128, 128] with relu activation and 2 conditioning layers with 16 neurons each and sigmoid activation.	- Removes background factory noise from audio in preprocessing	- Provides AUC and pAUC only. Other metrics were not provided in the paper. - Uses machine IDs to condition their autoencoder	AUC - Fan : 76.90, Pump: 78.45, Slider: 79.28, Valve: 76.26
[4]	2020	Uses a receptive-field-regularized, fully convolutional, residual network (ResNet). Consists of 7 residual blocks which contain two convolutional layers followed by a batch normalization layer. Relu activation function and cross entropy loss was used.	- The raw audio was normalized to zero mean and one standard deviation. - The batches used did not contain data from different datasets which makes their results more comparable with ours	- Provides AUC and pAUC only. Other metrics were not provided in the paper. - Treats machines of same type differently based on their ID's	AUC - Fan : ~93, Pump: ~92, Slider: ~97, Valve: ~92
[5]	2020	Uses a conformer autoencoder consisting of Feed-forward modules, including layer normalization, a linear layer with Swish activation and dropout, and another linear layer with dropout, A multi-head attention layer, and a convolution module, which includes layer normalization, a 1x1 1D convolution layer with GLU activation, and a 1D depth-wise convolution layer. Following the depth-wise convolution,	-Tested three different autoencoder architectures, a feed-forward AE, a Transformer AE, and a conformer (convolution-a ugmented Transformer) AE, of which the conformer	-Makes use of machine ID as an input feature that gets reconstructed with the sound based on the observation that the AE confuses the machine ID when the audio has anomalous sound which might not be applicable in	AUC: -Fan : 86.59% -Pump: 88.83% -Slider: 97.16% -Valve: 99.68%

		there's a 1D batch normalization layer, Swish activation, and a 1D convolution layer with dropout.	is the best performing	real world scenarios.	
[6]	2020	Combines two systems: a pretrained OpenL3 to extract embeddings of audio files and an Interpolation Auto Encoder (IAE) to compute statistical features (e.g.: mean, std, median, etc.) from the reconstruction error of Mel-spectrograms. The concatenation of the outputs of both systems is used to train a GMM with three components. The weighted log probability is used as an anomaly score. The paper proposed a second model that tunes the parameters (e.g.: dimensions of IAE encoder, sampling rate, etc.) for each machine type. Note: only the audio branch of OpenL3 is used to extract the embeddings.	- Makes use of a pre-trained model (OpenL3). - Uses Interpolation Auto Encoder instead of normal Auto Encoders. The IAE masks the middle time-frame and uses it as the output target. - Uses PCA to to decorrelate embeddings obtained from OpenL3.	- Provides AUC and pAUC only. Other metrics were not provided in the paper. - Uses statistical features as a part of the input to the GMM model.	- AUC: - Fan: 79.6 - Pump: 84.8 - Slider: 90.9 - Valve: 99.6
[7]	2020	Uses and architecture inspired from MobileNetV2 and MobileFaceNet to train a machine ID classifier. The feature extraction network takes an input of 1024×32×1 into a conv3x3 then a depthwise conv3x3 and then taken into multiple bottleneck layers to finally end at a conv1x1 then a Global Depthwise Conv. layer (GDConv) and finally to a linear 1x1cov. The embeddings generated from this network are inserted into	-They managed to address the problem of anomaly detection as a classification problem	- Although they manage to address the problem as a classification problem, their approach involves classifying the machine ID, which might not be applicable to the real world.	-AUC: Fan: 88.12% Pump: 91.59% Slider: 99.99% Valve: 92.98%

		an ArcFace layer to create the machine ID classifier.			
[8]	2020	Trains a Siamese Network (SNN) to extract features of the audio file and feeds these features to a KNN anomaly detection algorithm.	- Makes use of an SNN to extract embeddings. However, we can improve on their approach by freezing the SNN after training it and feeding the embeddings to an unfreezed fully connected layer perhaps attached to a classification layer.	- KNN is relatively slow. - KNN is a traditional ML method. More recent anomaly detection methods could be used. - Loss is not clearly mentioned. We can use triple loss or contrastive loss for our project. - Only AUC is mentioned. Other metrics like F1-score are not present in the paper. - Could have made use of augmentation.	- Average AUC: 85% - Average pAUC: 77.1
[9]	2021	3 CNN blocks followed by a convolutionary block attention module (CBAM) which is then followed by an outlier classifier. After each CNN block, a classifier was added and the final classification was the weighted average of each of the classifiers where the later classifiers have more weight. The CBAM block does max pooling and average pooling in parallel and then combines them and then sequentially does max and average pooling on them before passing it	- use the warm restarts gradient descent method to solve the local optimal problem that is prone to occur during gradient descent - They chose to have the front CNN blocks have a larger kernel size and more pooling operations to	- The paper doesn't mention what outlier classifier was used - Treats machines of same type differently based on their ID's	AUC: - Fan: 97.53 - Pump: 97.34 - Slider: 99.04 - Valve: 92.00

		through a convolutional layer.	reduce the feature dimension, while the back CNN blocks to have a smaller kernel size and fewer pooling operations to maintain the resolution of the features.		
[11]	2020	The paper uses ResNet architecture starting with a convolutional layer followed by three stages with four residual blocks and a batch normalization layer each and a global average pooling layer at the end. They used binary cross-entropy loss function and Adam optimizer.	- Audio was resampled to 16kHz which reduced the amount of data per sample.	- No methods were used to prevent the model from overfitting and no information was provided that shows it wasn't overfitted - Treats machines of same type differently based on their ID's	AUC: - Fan: 92 - Pump: 90 - Slider: 99 - Valve: 94
[16]	2023	The authors train an ID constrained Transformer AE (IDC-TransAE) along with a machine ID classifier of machines of the same type. The classifier's inputs are the latent features obtained from the encoder layer. In the end, a composed metric of the classification error using cross-entropy and the reconstruction error of the decoder layer is used to determine the anomaly score.	-The authors propose a new method to solve the AE generalization problem when a normal sound of a machine is anomalous to another machine of the same type by training a classifier and including the classification	-No metrics were provided other than AUC and pAUC -No information that shows the model isn't overfitted - Treats machines of same type differently based on their ID's	AUC: - Fan: 80.44 - Pump: 83.41 - Slider: 96.2 - Valve: 99.6

			error in the anomaly score. This enhances the AE's learning of each machine ID across the machine types		
[10]	2021	The paper uses a non-neural network approach by proposing a Multi Fidelity Bayesian online changepoint detection (MF BOCD) algorithm. It partitions the data and uses abrupt changes in partitions to detect anomalies. Before the BOCD was applied, they passed the data through a 2-layer dense neural network and computed the median anomaly score from that clip	- Requires less computation than a typical NN model	- The 2-layer dense neural for low fidelity observation resulted in unreliable results. - Reported results in mean square error	Mean Square Error was reported: - Fan: 0.0060 - Pump: 0.0195 - Slider: 0.0347 - Valve: 0.1743
[12]	2022	The paper uses a generator and discriminator model. The model uses 2 generators which are 1D convolutional autoencoders. The discriminator is a 1D convolutional classifier that distinguishes between real and fake signals. The model also has a global filter later which is a 1D FFT layer that captures global signals	- The global filter later can learn long-term interactions in the audio	- Only AUC scores were provided for the models - Treats machines of same type differently based on their ID's	AUC for 6db SNR: - Fan: 0.9534 - Pump: 0.9012 - Slider: 0.9110 - Valve: 0.6393 AUC for 0db SNR: - Fan: 0.8662 - Pump: 0.8531 - Slider: 0.8085 - Valve: 0.5701
[13]	2021	The network is called DeSpaWN (Denoising Sparse Wavelet Network) and it mimics the fast discrete wavelet transform (FDWT) with learnable filters and hard-thresholding coefficients. It consists of encoding and decoding	- It can also handle noisy and variable signals without any pre-processing. - There are only a few hundred	- It uses wavelet representation which may not be able to capture non-linear features in the audio	AUC: - Fan: 92.8 - - Pump: 84.5 - Fan: 86.2 - Slider: 91.0

		blocks, each with two convolutional layers and a hard-thresholding activation function.	parameters to learn so it is very computationall y light		
[19]	2023	The architecture applies the MobileFaceNet model with ArcFace loss as the main classifier of their architecture. However, they also use a simple 1D CNN to extract temporal features from audio to use as input features to the main model. The CNN has a Conv1D with a large kernel as the first layer followed by 3 Conv1D layers with layer normalization and Leaky Relu activations with smaller kernel sizes. The model is then trained on a fusion between the spectrogram and the temporal features (spectral-temporal features). The fusion of the two features is explained by the log-Mel filtering out high-frequency components of the anomaly sound where distinct features may reside.	-The authors propose new input features to be fed to the NN along with the log Mel spectrogram, which are the temporal features of the audio. The temporal features are extracted using a CNN network (TgramNet).	- Only AUC scores were provided - The model uses a simple architecture - No reported information about overfitting - Treats machines of the same type differently based on their ID's	AUC: - Fan: 94.04 - Pump: 91.94 - Slider: 99.55 - Valve: 94.44
[21]	2021	The architecture used is an Autoencoder model which comprises of fully-connected layers with 320-64-32-32-64-320 units in each of the layers. In addition to that a 1D-CNN-based scene classification model (S-Net) is used which comprises of 4-layers. The first layer is convolution layer with 16 filters, each of length 64, followed by a global average pooling layer, dense layer	The S-Net classifies the data under different noisy conditions by using a scene-aware framework which can be utilized to select threshold based on the surrounding	Results only give AUC values - Treats machines of same type differently based on their ID's	AUC values - Fan: 0.92 - Pump: 0.86 - Valve: 0.75 - Slider: 0.93

		with 64 units and classification layer with 3 neurons.	noisy conditions		
[23]	2023	The authors claim that the models that are trained for AD claiming to be unsupervised, are not truly unsupervised but “semi-unsupervised.” This is because in a case similar to the MIMII dataset, the data is previously labeled for clean training. Therefore, in their paper, the authors adopt a truly unsupervised approach by following an unsupervised data refinement (USDR) approach to train an autoencoder and principal component analysis (PCA) models.	- The authors assume a normal dataset contaminated with abnormalities and assume no labels prior to training making their approach truly unsupervised as opposed to labeling the anomaly-free and anomaly data making their findings the most realistic.	- No details about the architectures used. - Only provide average precision as a metric - No reported information about overfitting	Precision(6 dB): Auto Encoder: - Fan: 0.98 - Pump: 0.98 - Slider: 0.97 - Valve: 0.90 PCA: - Fan: 1.00 - Pump: 1.00 - Slider: 0.94 - Valve: 0.71
[24]	2020	The authors use an autoencoder; however, they realize that the auto-encoders can still effectively reconstruct abnormal samples–hence the reconstruction error cannot be used as an affective anomaly score. As a result, the authors used an encoder followed by a variant of the x-vector model used for speaker recognition capped with a classification layer.	- Uses a different architecture for each machine, this allows for better predictive scores. - Takes the highest value of two scores as an anomaly score.	- Encodes the ID of the machine in the input of the NN. This makes the model ungeneralizable to new IDs in the future. - Lists AUC only without other metrics like F1 score.	AUC: - Fan: 0.9303 - Pump: 0.9398 - Slider: 0.9888 - Valve: 0.9680
[25]	2021	The paper used an autoencoder with dense layers of size 64 64, 8, 64, 64, and 320, with a 320-element input vector. Each layer is followed by a ReLU activation function	- The model is simple which can be used for edge devices	- no description of the preprocessing was given	Accuracy: 0.70

		except for the last one which has linear activations.			
[26]	2023	Their model is an encoder front end which generates embeddings for a backend anomaly evaluator. The encoder consists of conformer blocks which includes two feed-forward networks that sandwich a multi-head self-attention module followed by a convolution module. The embeddings then feed into 3 networks. The first being an inlier properties learner which learns properties based on the machine ID using arcface loss. The next is an outlier decision boundary learner which uses BCE loss. The third is a augmentation classifier (the data is augmented to get more data) to help reduce overfitting	- Takes measures to reduce overfitting	- Only AUC was provided as metrics - Treats machines of same type differently based on their ID's	AUC: - Fan: 88.80 - Pump: 94.12 - Slider 100.00 - Valve:96.52

In summary, most models used autoencoders and it worked well for this task. All different types of autoencoder implementations were found such as dense autoencoders, convolutional autoencoders, and transformer-based autoencoders. For the anomaly score, many papers used the reconstruction loss of the decoder, some fit the latent space to Gaussian mixture models (GMM) which is similar to variational autoencoders in concept, while others used the latent space representation to find the anomaly score.

There were also other successful methods such as GANs [12] [16], SNN [8], and various types of CNNs such as mobile net-inspired architectures [7] [19], ResNet [4] [11], and a CBAM-based CNN (Convolutional Block Attention Module)[9]. There were also successful non-neural network approaches found such as BOCD (Bayesian online changepoint detection) [10].

However we would like to note, most of the papers that used this dataset were participating in the DCASE challenge. In that, they were given 3 datasets, training, additional training and evaluation. According to the website, the training consisted of sounds from all machine types machines from IDs 1-4 and the additional training consisted of sounds for all machine types with IDs 5-7. And the evaluation dataset contained sounds from machines of ID's from 5-7. We noticed that the papers with the best results augmented their architecture by using the machine ID to either affect the loss (which affects latent space representation) or to help in the anomaly score, which in our opinion invalidates their results. This is because it results in overfitting for the specific machines given in the training set and wouldn't generalize for all the machines of the same type that are not part of the dataset.

The metrics provided were mostly AUC from which, the best results were from [23] with average AUC of 95.75. However, if we only include the papers that didn't use ID to augment their model, the highest results were from [3] with average AUC of 86
For processing, most papers used mel spectrograms to process the audio before passing into the neural network. There was no mention of window sizes in the papers which leads us to believe that the full 10 seconds were used.

On a related note, as part of this literature review, we tried to replicate the baseline model built by the MIMII authors [1]. The model was a simple dense autoencoder with the first two layers of the encoder having 64 neurons, and then the latent space layer is a dense layer of 8 neurons. The decoder also consists of two dense layers of 64 neurons and the output has the number of neurons the same as the input dimension. While this architecture is simple the authors reported AUC above 90% for some machines. However, after inspecting their code and running a clone of it, it turned out that they were calculating their AUC correctly, which resulted in high AUC. The way they do it is by finding the AUC between y_true (zeros and one for anomaly and normal) and the reconstruction error of the decoder.

3. Data Selection [/ 10]

With the increase of automation in every aspect of manufacturing, there is a demand for automating the process of inspecting the quality of the machines and whether or not it is working properly. For a machine to be considered as working normally, it should sound like it is working normally. This is the basic principle that the authors of [1] had in mind when they created the MIMII dataset. This dataset contains different types of machine sounds in real factory environments. The authors recorded sounds for 4 different types of machine:

- Fans - industrial fans, which are used to provide a continuous flow or gas of air in factories
- Pumps - water pumps, which drained water from a pool and discharged water to the pool continuously
- Slide rails - linear slide systems, which consist of a moving platform and a stage base
- Valves - solenoid valves that are repeatedly opened and closed.

For each machine type, there are four different models of the machine (denoted by ids 00, 02, 04, 06). The recording process was to place the microphones 50 cm away from each machine and 10 second clips were recorded. In addition to that, background noise from real factory environments were recorded and mixed in the dataset. The background noise was added in different amounts to create 3 versions of the dataset, 6dB SNR, 0dB SNR and -6dB SNR. With these recording conditions, both normal and abnormal conditions were recorded. For this paper, we have chosen the 6dB SNR version of this dataset.

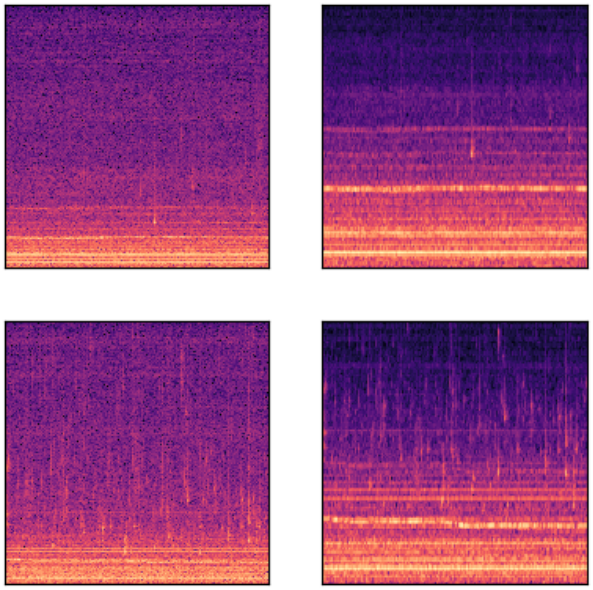
This dataset contains 14,719 10 seconds of normal machine sounds that can be used to train unsupervised neural networks that learn the distribution of normal sounds to create anomaly detectors. The approximately 3600 samples of data per machine type gives us 1 hour of normal sound data per machine type which should be enough to train neural network models that need to detect anomalies in 10 second clips. In addition, there are 700-1000 samples for every model which helps ensure our model is not fitted to only one instance of machines.

This dataset contains sounds from real machines recorded in realistic scenarios (because of the background factory noise) which makes our work here directly applicable in real life industrial ASD scenarios. This dataset also records some anomalous sound conditions which can be used to test our models to ensure they work correctly.

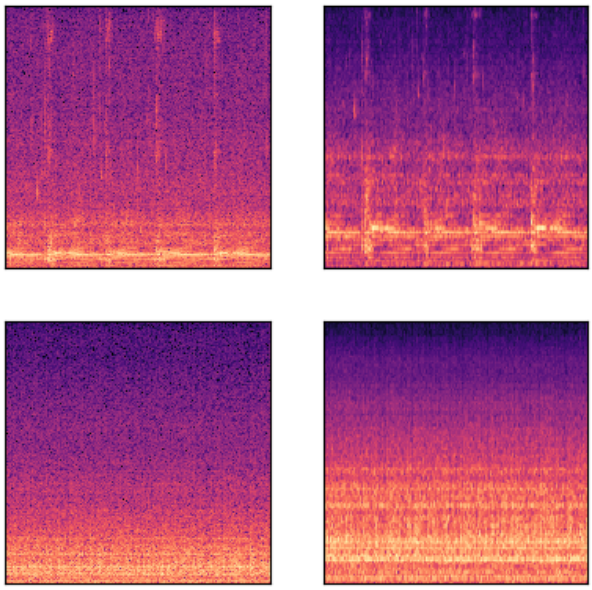
4. Data Cleaning and Feature Engineering [/10]

The data in the MIMII dataset is given such that all the audio data are in 10 second clips, single channel sampled at 16 kHz. So the data is clean and we can use it as is. However, for neural networks to process audio data such as ours, we need to split the samples into training and testing sets and then convert them to a spectrogram.

We can either use STFT and stop there or further process it to the mel scale. We plotted both and found that the mel spectrogram shows more information. For demonstration, below are 4 images of normal fan samples. On the left side STFT spectrograms are shown and on the right mel spectrograms of the same samples are shown.

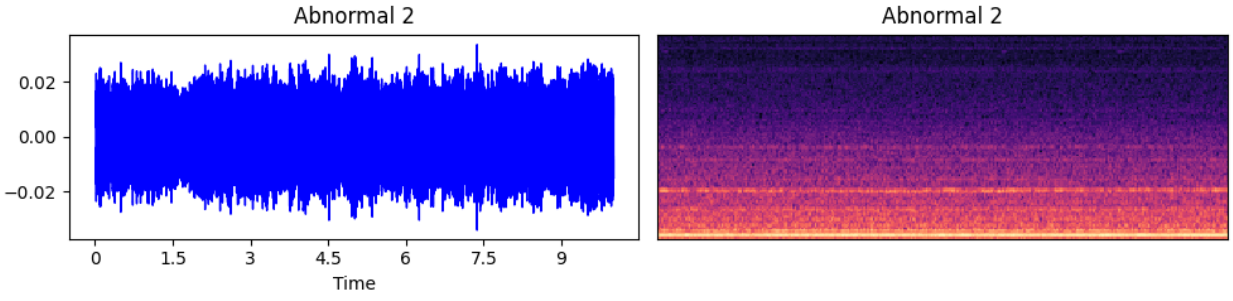


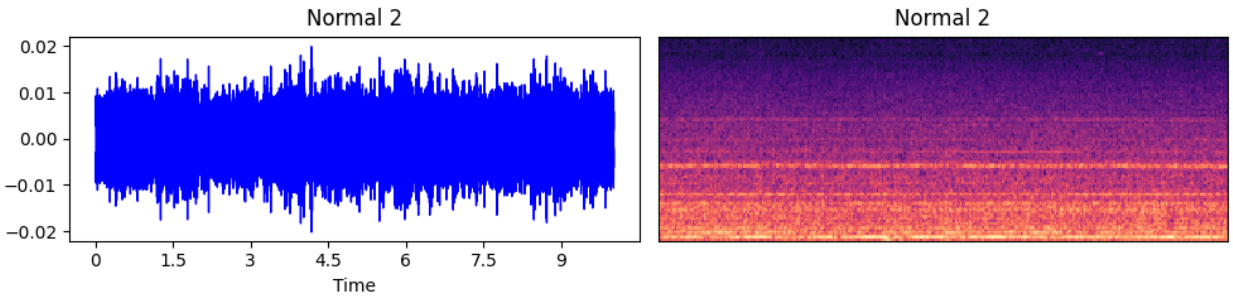
We ran the same test for abnormal fans samples as seen below and found similar results that the mel spectrogram (on the right) shows more information.



Hence we choose to process it further to a mel spectrogram before passing it into our model. We then reshaped and scaled our data using min max scaling to get all the values between 0 and 1. This was done to prevent exploding gradients. We ensure that we only use the min and max values of the training data to scale all the data (training and testing). This was done because min max scaling is sensitive to outliers and abnormal data is likely to have outliers.

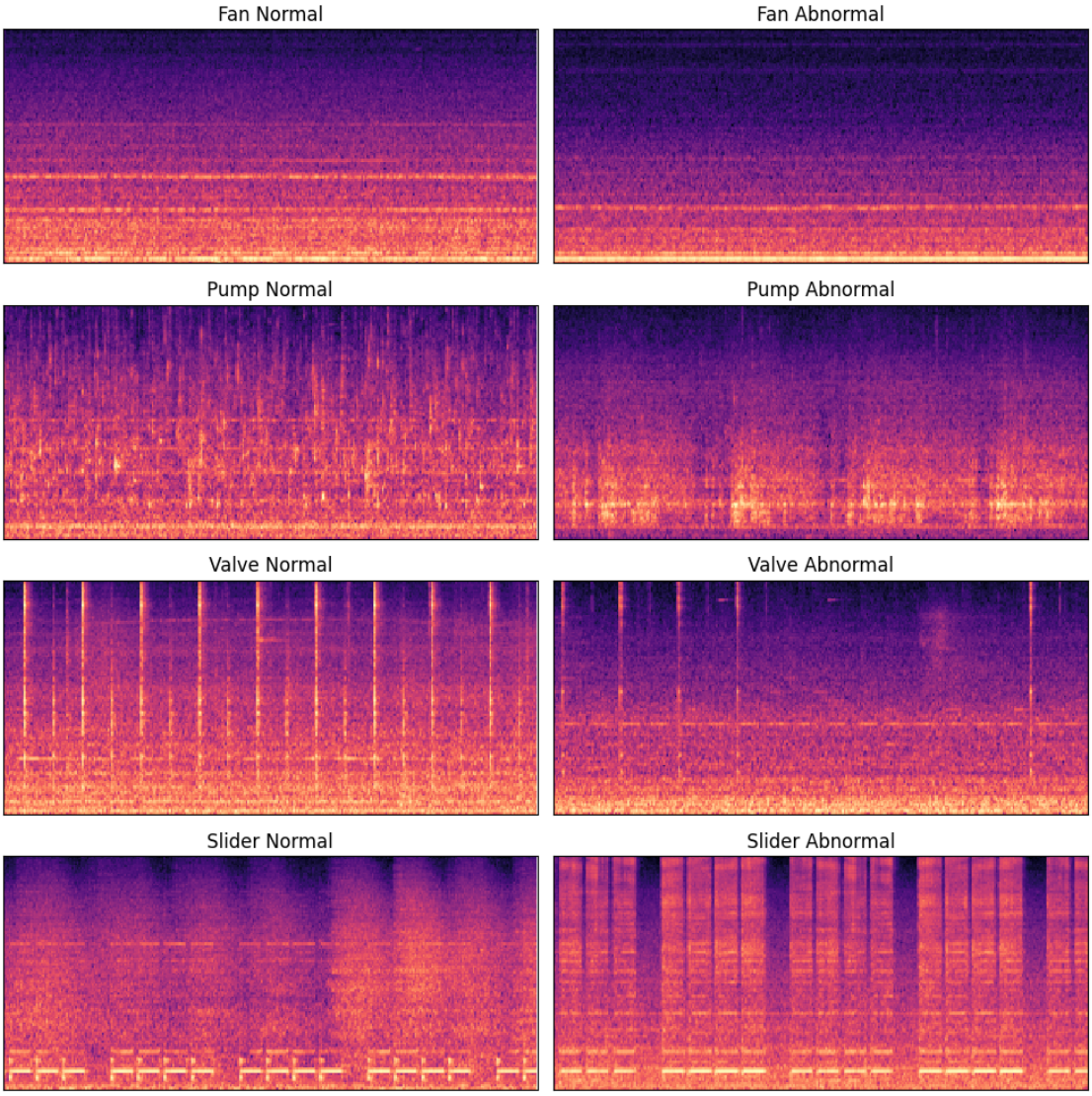
We also chose not to split the data into window sizes as it could lead to some sub samples of abnormal being misclassified as normal samples and vice versa. Below are the time signals and its mel spec for demonstration. The 4.5 to 6 seconds are quite similar which can lead to misclassification. However, we are still aware that having a larger window size will have an averaging effect on the anomaly data since most of the audio signal might be normal.



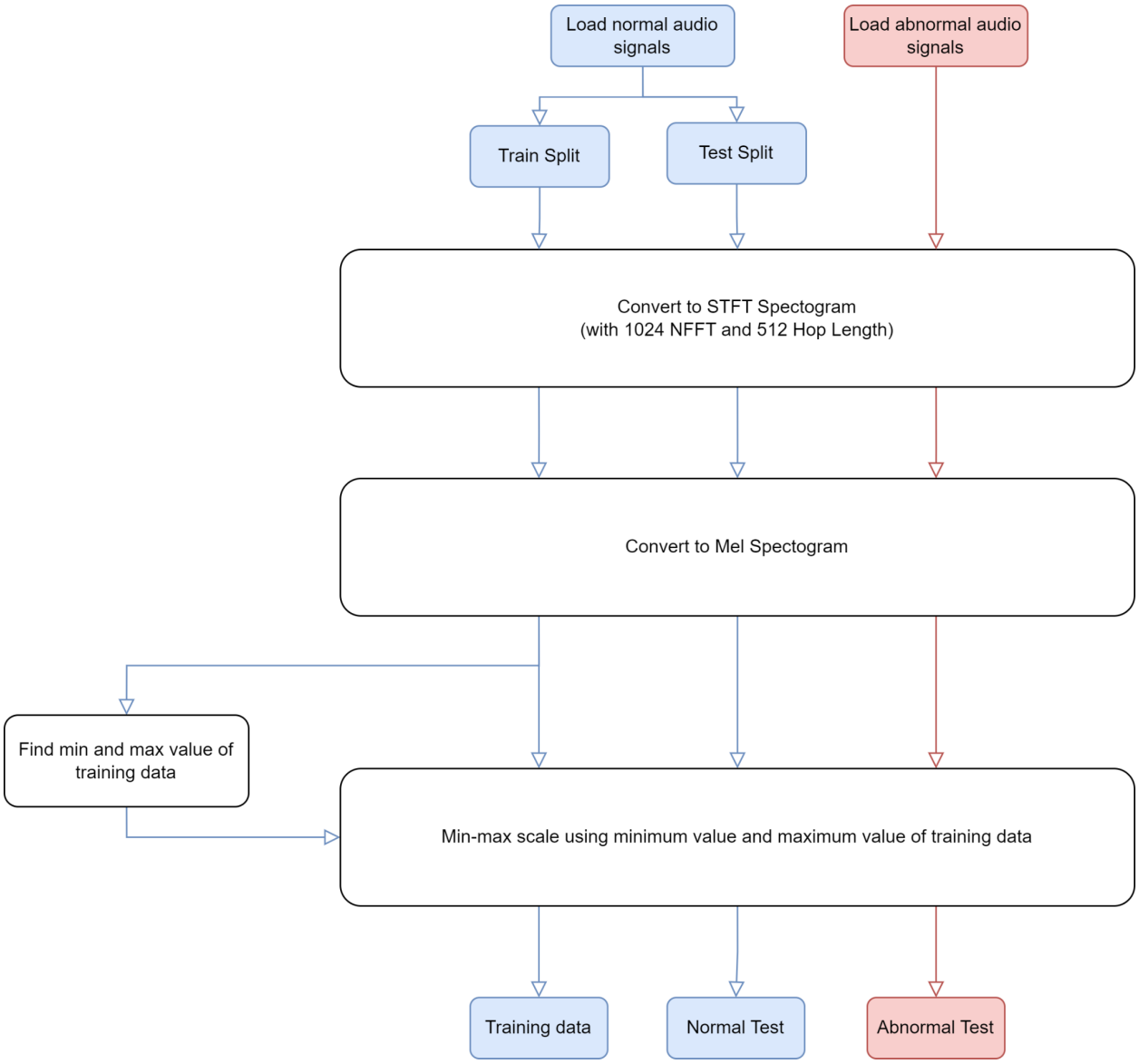


Other problems caused by using a smaller window size is that we will have much more data points than we have with 10s, and training the NNs would be time consuming with the computational power we have.

Our final processed data is shown below



Below is a diagram representing the flowchart of our data cleaning and feature extraction



5. Neural Network Architecture Selection [/ 10]

Model	Features
2D CNN Classifier	- 2D Convolutional layers followed by dense layers - Trained in a supervised manner - Anomaly detected via the classification - Chosen to be used as a proof of concept so that we know that the problem can be solved using neural networks.
Dense AutoEncoder	- Consists of 2 parts, the encoder and the decoder. - The encoder consists of dense layers with lesser and lesser number of neurons that reduce the size of the data and compress it into a latent space - The decoder consists of dense layers with more and more number of neurons that increase the size of the data and reconstruct the input from the latent space - The spectrogram is flattened before passing it into the network. - Anomaly is detected via the reconstruction loss - This style of architecture was used by the baseline model [1] (the basic model released by the authors) and by papers [3][25]
Deep Convolutional AutoEncoder	- Consists of 2 parts, the encoder and the decoder. - The encoder consists of sequential blocks 2D convolutional layers followed by pooling layers to reduce the size of the data and compress it into a latent space - The decoder consists of sequential blocks of 2D convolutional layers followed by upsampling layers to increase the size of the data and reconstruct the data from the latent space - This architecture is similar to the dense auto encoder but since our input is spectrograms (which can be treated as images), a 2D convolutional network makes sense to extract features from the spectrograms - Anomaly is detected via the reconstruction loss
Deep Convolutional AutoEncoder with classifier	- Consists of 3 parts, the encoder and the decoder, and the classifier - The encoder consists of sequential blocks 2D convolutional layers followed by pooling layers to reduce the size of the data and compress it into a latent space - The decoder consists of sequential blocks of 2D convolutional layers followed by upsampling layers to increase the size of the data and reconstruct the data from the latent space - The classifier is attached to the output of the latent space and classifies a latent space encoded point. Any unsupervised classifier can work but

	we chose to use unsupervised nearest neighbors - This architecture was chosen because we noticed that the latent space was separating between normal and abnormal but the reconstruction loss difference was low, so we classify based on the latent space embeddings of the data. - Anomaly is detected via unsupervised classifier prediction - The concept of using latent space for anomaly detection was used in [26]

6. Why do chosen NN Architectures Work [/ 10]

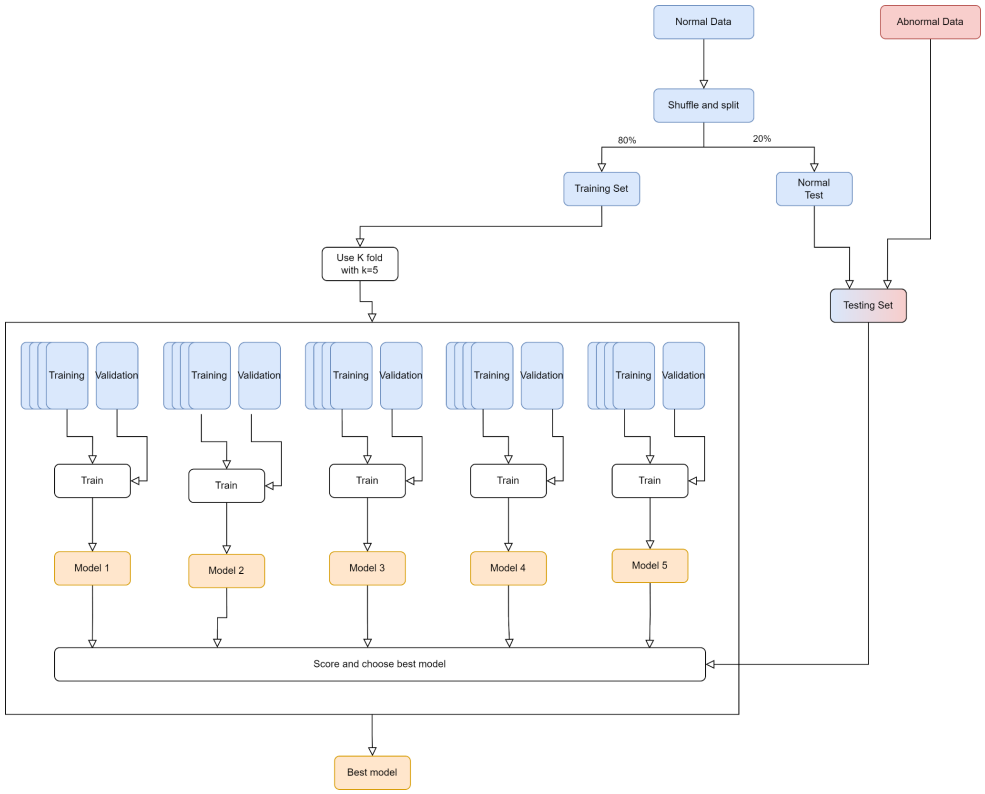
Clearly explain why the various NN architectures are expected to work well for your problem. What was the basic idea behind each neural network architecture being proposed? Summarize your findings in the form of a table.

Model	Reasons for working
2D CNN Classifier	- Our data is passed as spectrograms which can be treated like images - For image classification, 2D CNNs work very well as they can extract features across the image - And the dense layers attached to the feature maps can classify the data into normal and abnormal
Dense AutoEncoder	- Autoencoders for anomaly detection work on the basic principle that during training, the reconstruction of normal data is learnt, after which, when abnormal data is passed, it will not be able to reconstruct it from the latent space resulting in high reconstruction loss
Deep Convolutional AutoEncoder	- Since our input is spectrograms, we can use 2D convolutional layers instead of dense layers to extract features from the images. The idea is the same as the dense auto encoder, during training, the reconstruction of normal data is learnt and when data is given that is very different from the normal, it leads to high reconstruction error.
Deep Convolutional AutoEncoder with classifier	- The idea behind this version of autoencoders is that the 2D convolutional layers are used because of spectrograms as explained above but it relies on the fact that the normal and abnormal data are separated in the latent space. Regardless of high reconstruction loss or not, as long as the latent space can separate between the normal and abnormal, anomalous samples can be found by using a classifier.

7. Validation Methodology [/ 10]

The MIMII datasets consist of normal and abnormal machine sounds of four types of machines mentioned previously. For each of the four machines, we take all the IDs from that machine and we split normal data from abnormal data. The abnormal data is kept for testing because it doesn't make sense to be included in training for an anomaly detection task trained in an unsupervised way. We, however, will still use this data in a supervised manner to train classifiers just to make sure that normals and abnormals are distinguishable from the mel spectrograms and from their latent space mappings. As for the normal data, we apply an 80/20 train test split. Therefore, we now have normal and abnormal data for testing.

The training part (normal only) is used for cross-validation using 5 folds. This means 4 segments of the training data will be used for training and one segment will be used for validation in each of the 5 training iterations. The best performing model out of the five folds is then stored and used for the rest of the run. The flowchart below shows out validation methodology



Grid search could have been used to refine the results we obtained in many instances, but mainly we could have used it to find the best dimensions for our latent space, the number of neighbors in the KNN, and the threshold for the anomaly score. However, since we were limited in terms of computation and time, we didn't manage to do that. As for the metrics, we intend to report accuracy, precision, recall, F1 score, ROC curves, and the AUC for the normal and abnormal classes for each machine. The precision is the proportion of correctly predicted positives out of all the predicted positive instances of the class. The recall the proportion of the correctly predicted positives out of all the actual positive instances of the class. The f1 score is the harmonic mean of the precision and recall. The ROC curve is a plot of the true positive rate against the false positive throughout different variations of the model. Lastly, AUC is the area under the ROC curve

8. Results [/ 20]

Baseline Model:

We took the same model as the baseline which was a dense autoencoder and had 2 64 neuron dense layers, 8 neuron latent space, two 64 neuron dense layers and the output. The baseline model only reported AUC so we did the same.

- Average AUC:
- Fan : 87.9%
- Valve: 44.9%
- Slider: 79.6%
- Pump: 83.1%

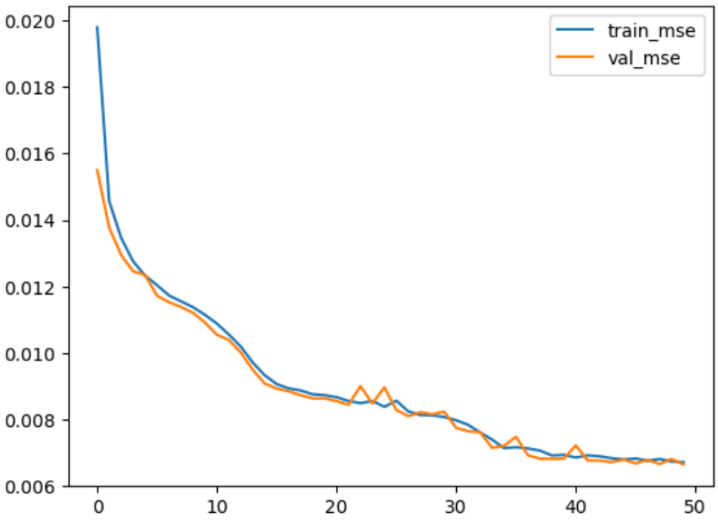
Simple Classifier

We ran a simple supervised classifier model trained on normal and abnormal data of fans just to know if the problem can be solved with neural networks. The precision, recall, f1 score, and accuracy of that model was the following:

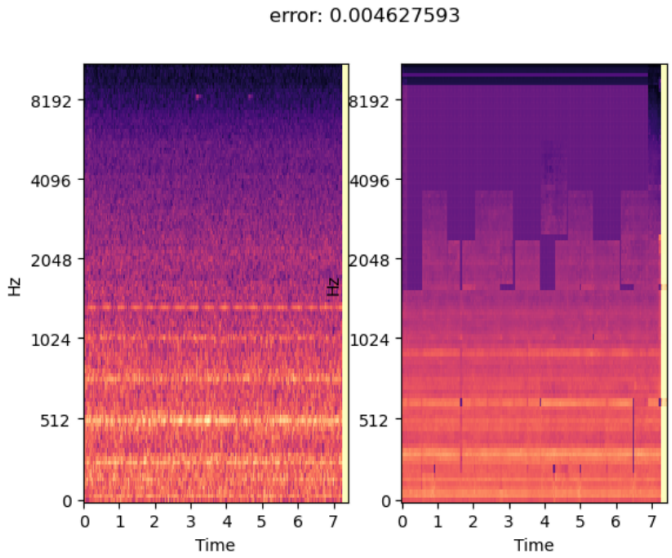
- accuracy : 0.98
- abnormal : precision = 0.99, recall= 0.94, f1 score = 0.96
- normal : precision = 0.99, recall= 1.0, f1 score = 0.99

Deep Convolutional AutoEncoder

We tried many different models of deep convolutional auto encoders and used their reconstruction loss to find anomalies. The first one is a 4 pairs of conv2d with average pooling encoder, a dense 480 neuron latent space and 4 conv2d transpose for the decoder. After training the loss (MSE) curves looked like this

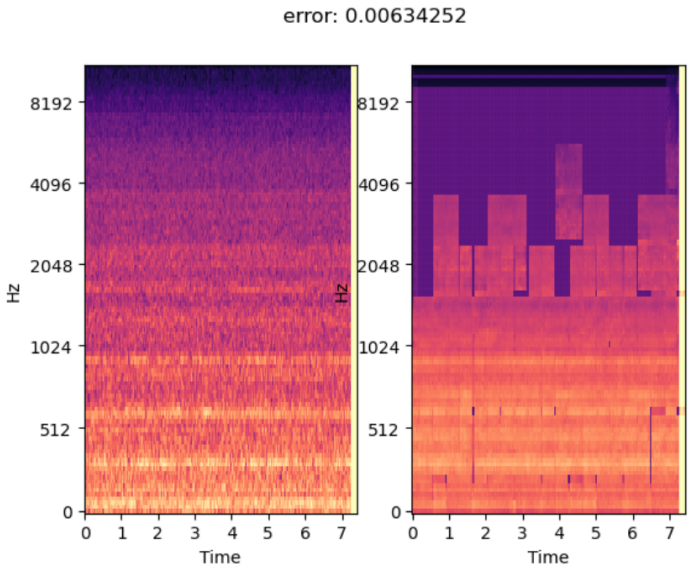


This model gave us averaged reconstruction error for normal data: 0.006719695. If we plot the results, we can see it did a sub par reconstruction for normal data

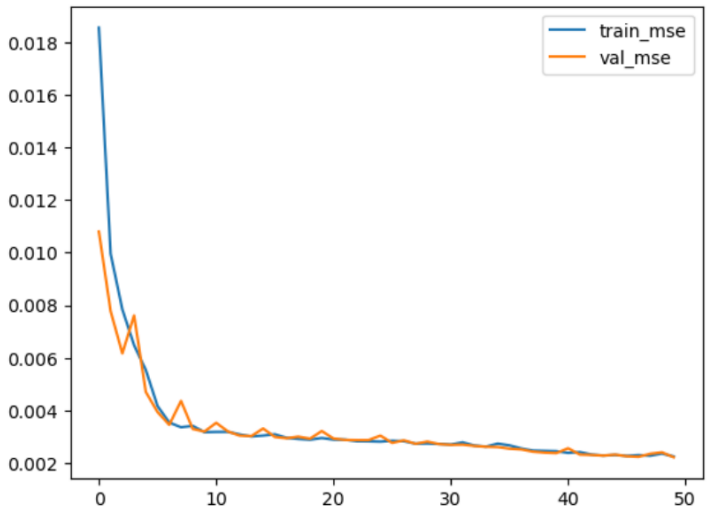


For abnormal data it gave us an averaged error: 0.0073565105

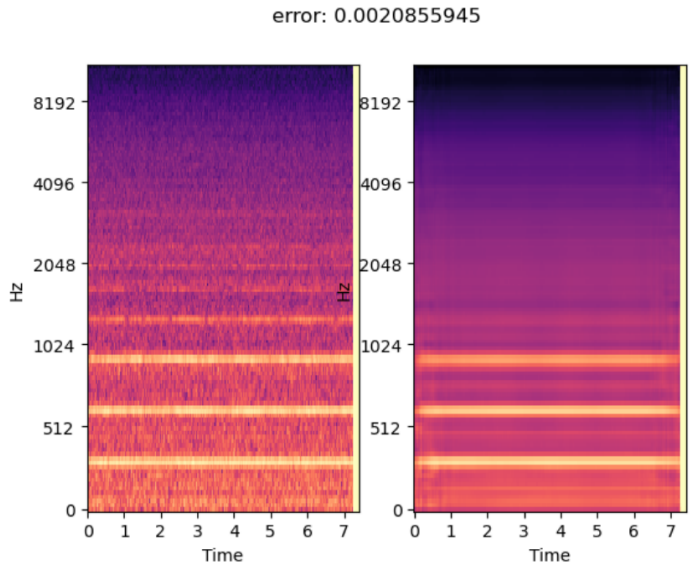
And if we plot the reconstruction, we can see it did a similarly sub par reconstruction



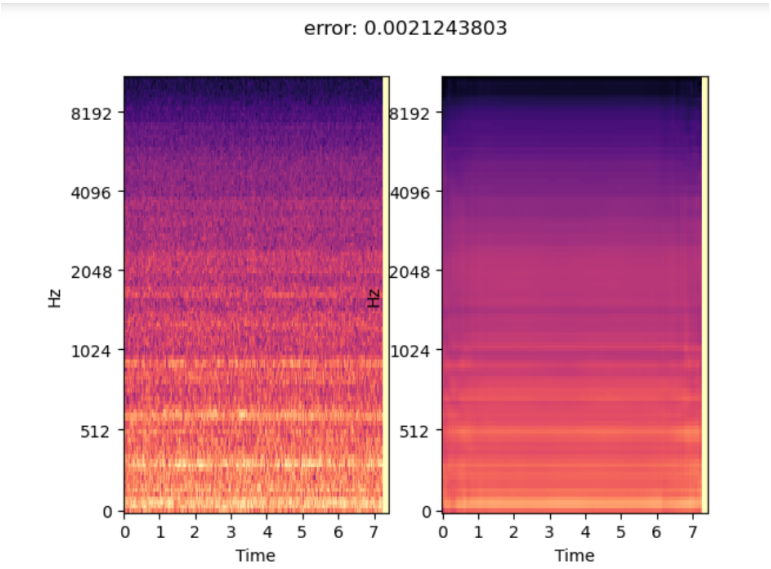
Since we did not get good results with dense latent space, we tried with a volume latent space. The next model we used was 4 pairs of conv2d with max pooling encoder, and 4 conv2d and upsampling pairs for the decoder. The latent space was a volume with (8, 20, 32) . After training the loss curves looked like this



If we plot the reconstruction it gives a image with no noise



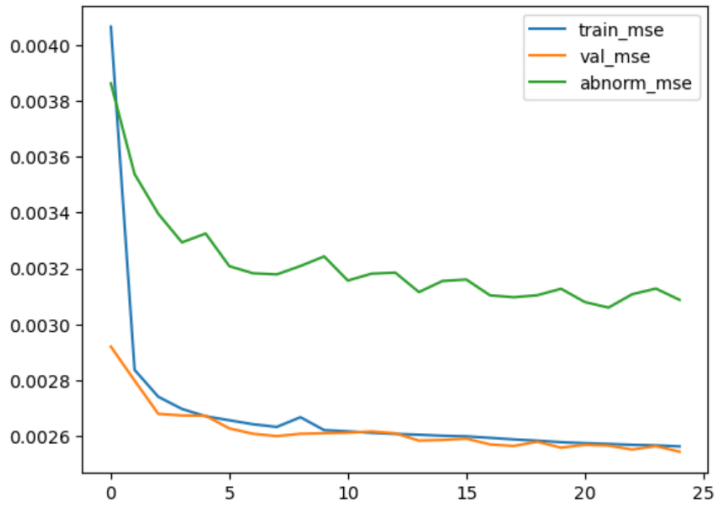
This was similar for abnormal data



The averaged error for normal test data is 0.0022148688 while the averaged error for abnormal data was 0.0021914856.

Since the convolutional autoencoders were not giving high reconstruction loss difference, we tried a model with a window size of 0.5.The model was 3 pairs of conv2d with average pooling

encoder, a dense 160 neuron latent space and 3 conv2d transpose for the decoder. After training the loss curves looked like this

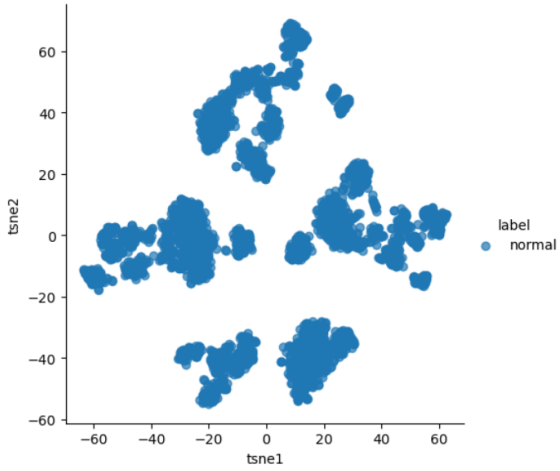


The mean reconstruction loss for normal data was: 0.00254 (sd= 0.001127) while the mean reconstruction loss for abnormal data was: 0.003100 (with sd=0.002146)

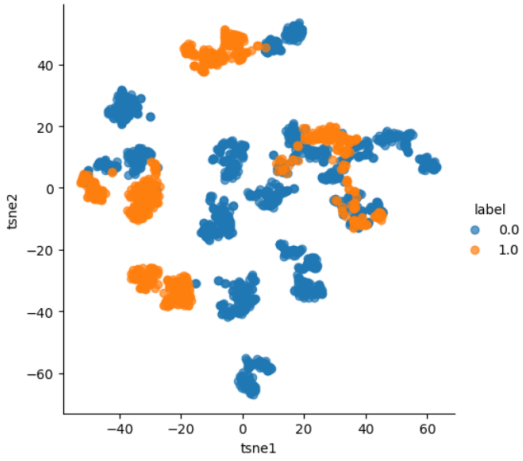
Deep Convolutional Auto Encoders with classifier

For testing we ran a model with only 1 of the 4 ids. The model was 4 Convolutional 2D layers followed by max pooling layers which resulted in a latent space of size (8, 20, 64). Then we used 4 2d Convolutional layers followed by upsampling to get the decoder output. For the classifier we used KNN with 65 neighbors and PSNR metric.

TSNE plot of the flattened latent space of the training data:

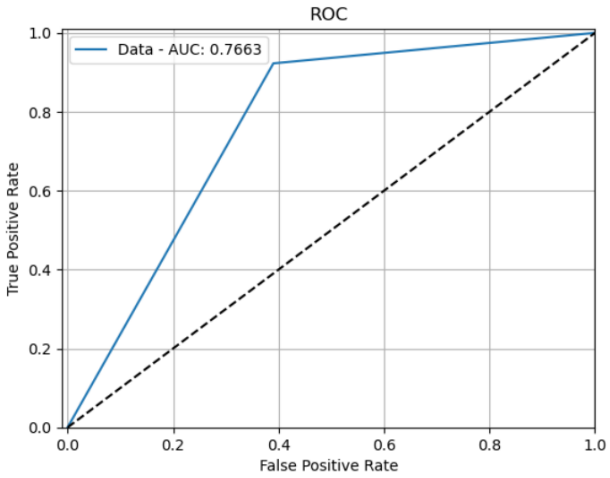


The TSNE plot of the testing data:

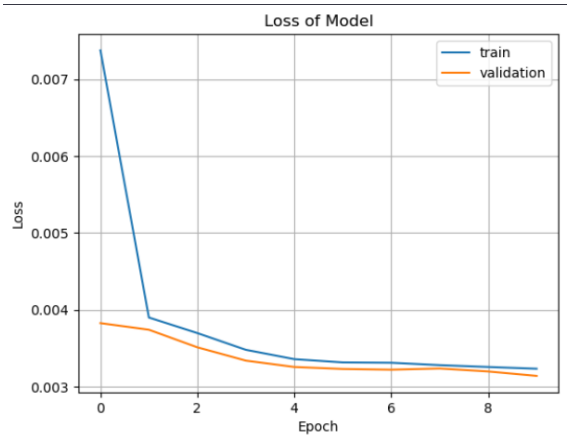


- Accuracy : 0.81
- Abnormal: precision: 0.81, recall: 0.92, f1 score: 0.86
- Normal: precision: 0.81, recall: 0.61, f1 score: 0.70

Average AUC and ROC curve:

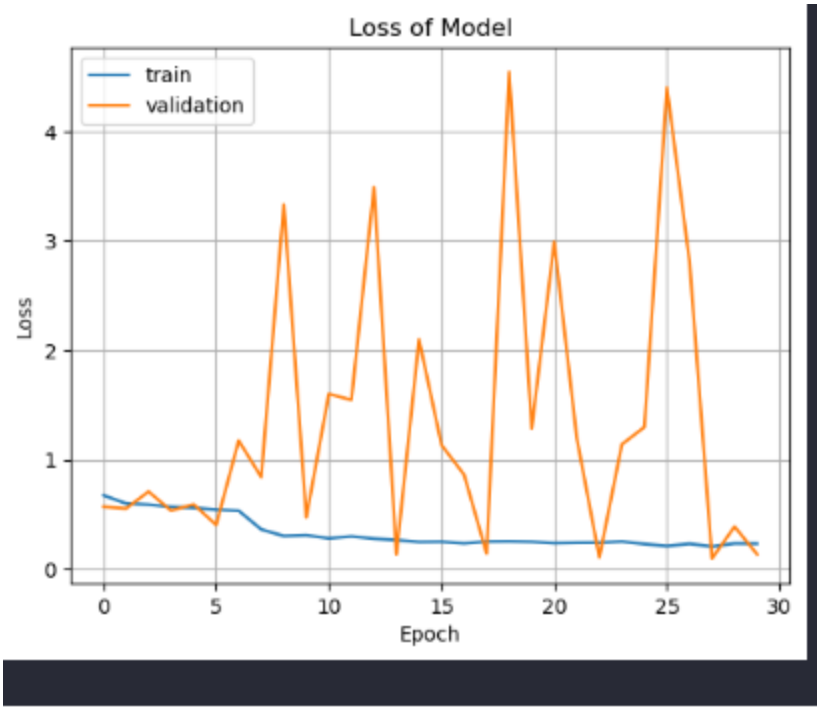


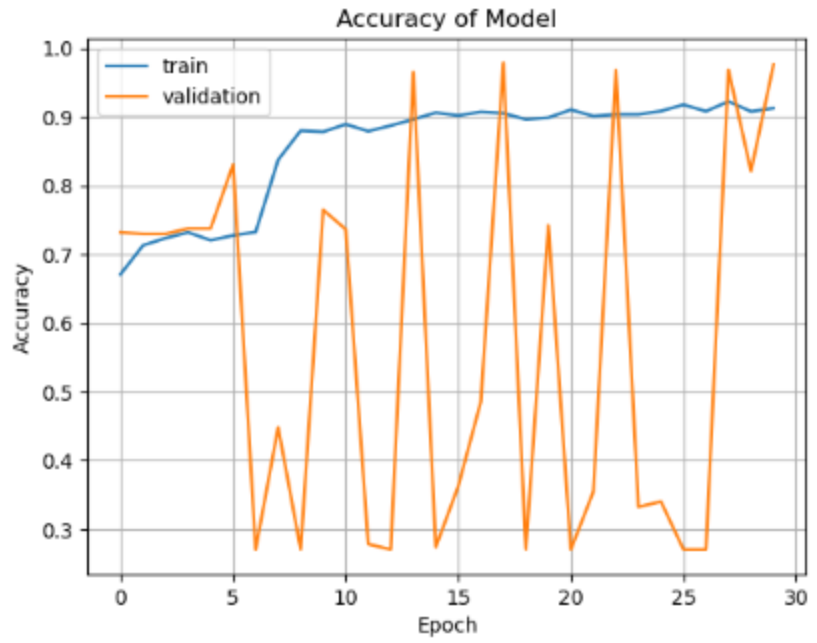
For this model we used a similar architecture to the above but with mse metric for the KNN. We also used kfold cross validation for this, so the TSNE will look different.



From the loss curve, the training of the autoencoder is not overfitting.

We also trained a dense classifier based on the latent space given by the encoder and the results are as follows:





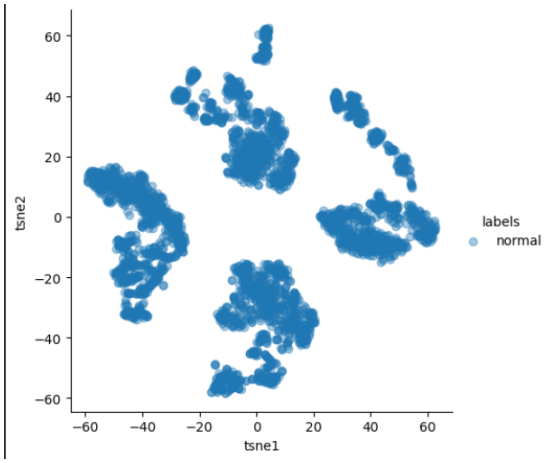
The training curves are exploding.

Confusion matrix of the:

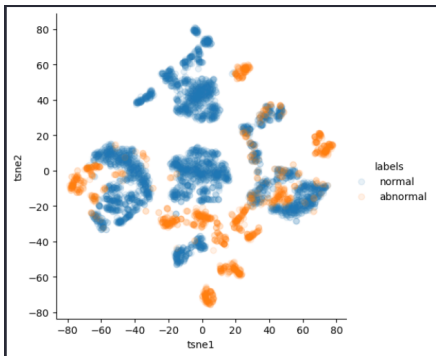
	normal	abnormal
normal	789	26
abnormal	3	292

- Accuracy: 0.97
- Normal: precision: 1.00, recall: 0.97, f1 score: 0.98
- Abnormal: precision: 0.92, recall: 0.99, f1 score: 0.95

TSNE for flattened latent space of training data:

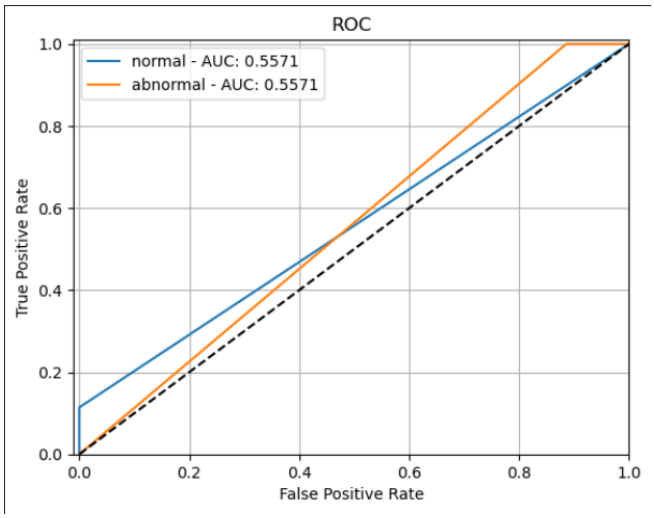


TSNE of all the data points:



- Accuracy : 0.68
- normal: precision: 1.00, recall: 0.11 f1 score: 0.20
- abnormal: precision: 0.67 recall: 1.00 f1 score: 0.80

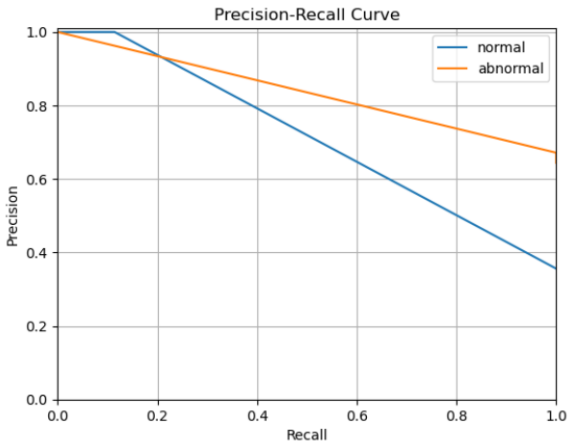
ROC curves and AUC for normal and abnormal



Confusion matrix:

	normal	abnormal
normal	93	722
abnormal	0	1475

Precision/recall curve:



Architecture	Parameters and Conditions	Results
Dense AutoEncoder	Dense: 64-64-8-64-64 parameters : 5, 177, 736	- Average AUC: - Fan : 87.9% - Valve: 44.9% - Slider: 79.6% - Pump: 83.1%
Supervised CNN classifier	3 Convolutional layers followed by dense layer of 3 with softmax Parameters: 53362	- accuracy : 0.98 - abnormal : precision = 0.99, recall= 0.94, f1 score = 0.96 - normal : precision = 0.99, recall= 1.0, f1 score = 0.99
Deep Convolutional Auto Encoders with classifier	4 convolutional layers →latent space → 4 conv + upsampling layers	- Accuracy : 0.68 - normal: precision: 1.00, recall: 0.11 f1 score: 0.20 - abnormal: precision: 0.67 recall: 1.00 f1 score: 0.80

9. Discussion and Future Work [/ 10]

In general, the results we got were sub par compared to the general results obtained from the DCASE challenge, however, we would like to note that almost all of the models proposed in DCASE used machine ID's to augment their learning which we believe results in their models not being useful outside the context of the challenge. For our model selection, we plan to use more complex architectures in the future, including variational autoencoders, stacked autoencoders and GANS. These are models that we found in our literature review and we are suggested to believe that they work.

Our supervised models worked really well which gives us an indication that the neural network can solve the problem. Our reproducing the baseline results resulted in us getting similar results to the baseline. Deep convoluted autoencoders performed worse than expected. During training we could see the separation in loss between the normal and abnormal data however the latent spaces did not show much separation and we were not able to extract anomaly scores from them.

For our Deep Convolutional AutoEncoder with classifier, the model we spent the most time on, we mostly used KNN as our classifier. We would also like to try using other unsupervised classifiers such as one class SVM. Also, for our KNN, we would like to optimize our n_neibourghs and threshold even more such that it results in less false positives and negatives. We would also like to experiment with other distance metrics with KNN to see which gives us the best results.

Also, we would like to train the models and compute the metrics for all the 4 types of machines given in the dataset. Given the time constraint and limited computational resources, we were only able to produce results mainly for the fan data. We don't expect to change our model architecture for each machine but we do believe the results will be different because of the different sound patterns.

10. Link to a YouTube Video

https://youtu.be/HCLPRen9ZCg