Group 2 Challenge 2

Anomalous Sound Detection

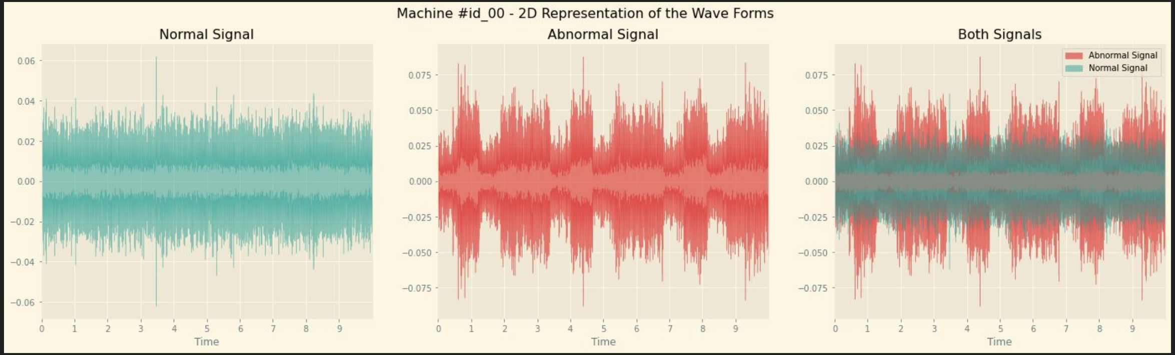
August Birkeland, Dario Dekovic, Santhosh Shanmugam & Tiril Syslak

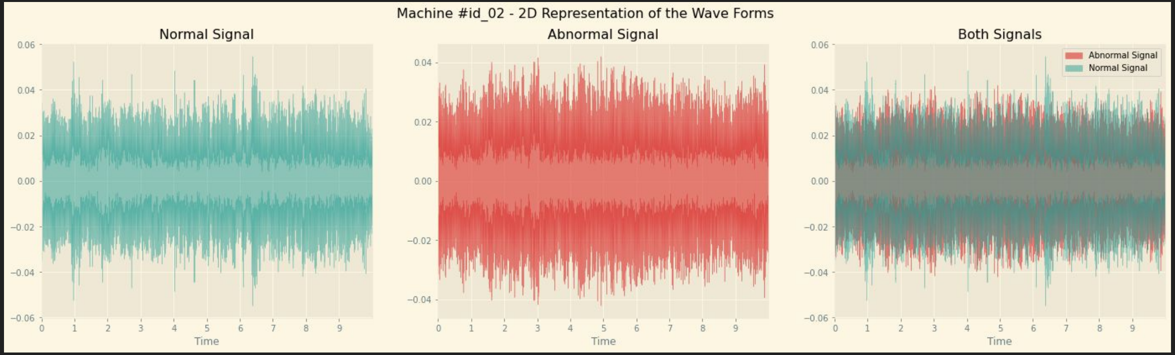
Data Analysis Objectives

- Find out the impact of the difference between the six machines
- Visualize how different an anomaly is compared to a normal sample
- Make use of different audio feature extraction tools and observe which ones are able to differentiate the best between anomaly and normal sounds

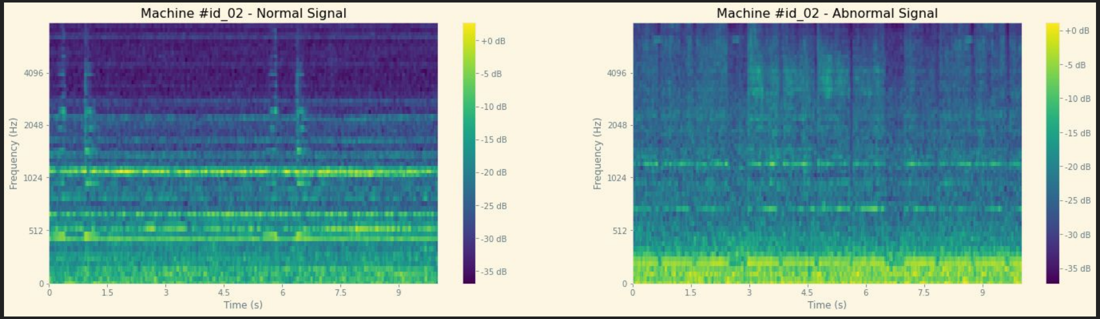
Data Analysis - Machine Analysis

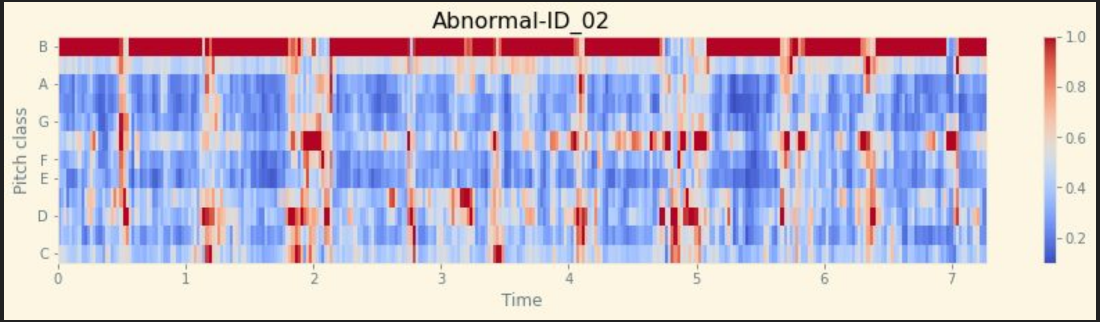
- First, we manually listened to a few of each of the machines’ (00, 02 and 04) normal and anomalous sounds and were able to hear that the machines were highly different, as well as that normal sounds for the same machine can differ quite substantially. In addition we listened to the machines (01, 03 and 05) in the “eval” set and checked how they compared to the machines in the “dev” set
- Then, we tested the feature extraction tools Root-mean-square (RMS), MCFF, Zero Crossing, logMEL and Chroma to visualize the differences between the samples
- By doing this we observed that logMEL and Chroma were better at highlighting the difference between normal and anomalous sound than the rest of the feature extraction tools, as can be seen on the next slides

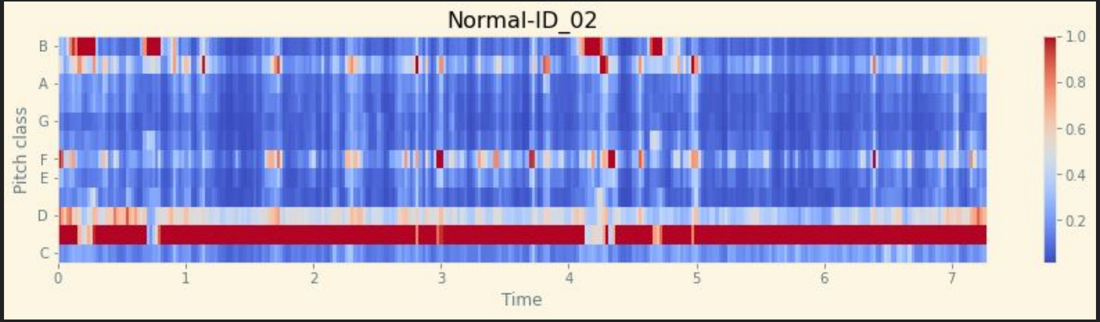




- Here we can observe a normal signal and an anomalous signal of both Machine00 and Machine02
- It is apparent that anomalies can be both different and similar to a normal signal waveform







- We used the same anomalous and normal signals of Machine02 as the last slide
- The visualization of the logMEL indicates that the differences between the spectrograms are not that clear
- The visualization of the Chroma feature extraction indicates a better distinction between the signals

Data Analysis - Machine Analysis Conclusions

- Compared to the output of RMS, MCFF and Zero Crossing, the output plots of logMEL and Chroma were better at distinguishing between the anomalous and normal samples. However, we did not get the time to make an ensemble of both Chroma and logMEL and therefore only made use of the logMEL spectrograms in our training
- To be clear, MELs are spectrograms where the frequencies are converted to the MEL scale, making it more similar to how humans perceive sound. Since we managed to hear the differences between the normal and anomalous sounds, supplying these spectrograms to the training will help the model

Data Preprocessing Objectives

- Utilize the additional training set
- Rewrite the Dataset class

Data Preprocessing - Utilizing Additional Data

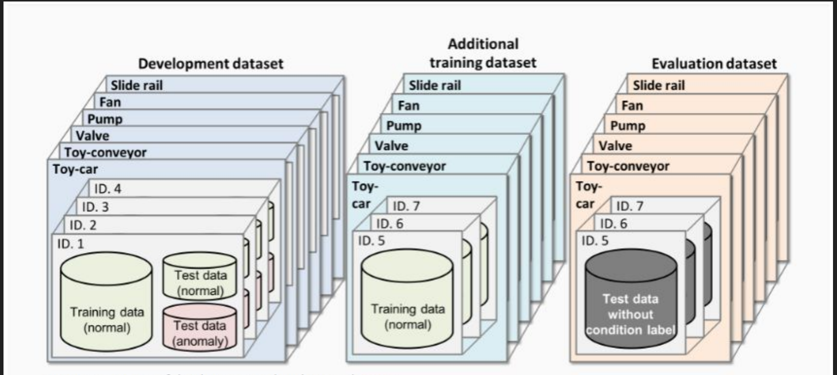


Figure 2: Overview of development and evaluation datasets.

- We were presented different datasets, where two of them could be used for training
- As seen in the figure we could use the training development dataset (from ID 1, 2, 3, and 4) and training evaluation dataset (from Machine ID 5, 6, 7)

Data Preprocessing - Utilizing Additional Data

- The “eval” test set utilizes the same machines as the additional train set. From our data analysis we have gathered that the different machines are able produce substantially different sound samples. A difference in what machines are used in the training set can therefore have a great impact on the training itself
- Therefore we wanted to train the model both with the eval set alone, and the two sets together

Data Preprocessing - Rewriting the Dataset class

Since we utilized CNN architectures in some of our experiments we had to rewrite our dataset class in two steps:

1. We utilized a given method to generate a 2D logMEL spectrogram instead of 1D logMEL spectrograms
2. In the 2D setting it was impossible to precompute and load all the data in the constructor because the memory footprint would be too large. For that reason we changed the code to generate spectrograms during runtime

Modelling Objectives

- Convolutional Auto Encoder
- Three Model Pipeline
- Optimizing Parameters
- Self Supervised Classification
- Baseline VAE

Model Selection - Convolutional AE

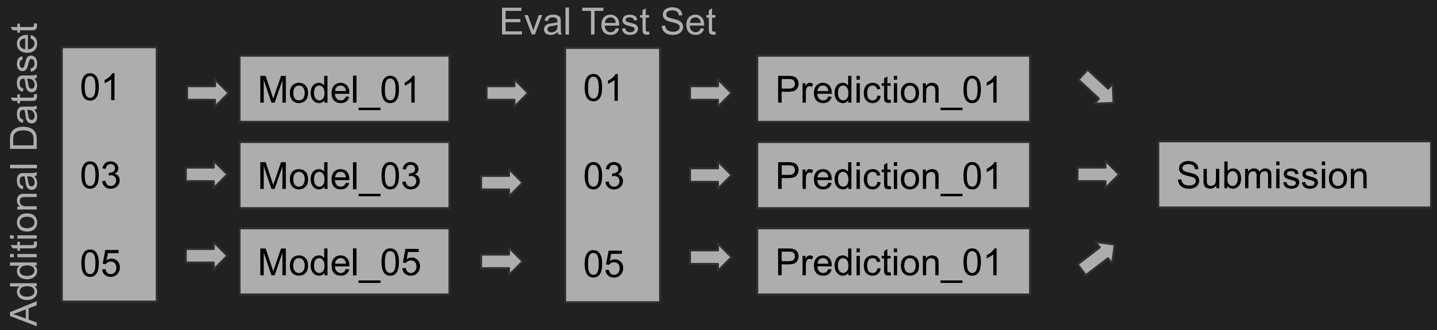
- Our first step on attempting to improve the baseline model was utilizing the CAE described in the github repository. We ran experiments for different crop sizes of logMEL spectrograms and obtained the following results:

	Crop size 64x64	Crop size 128x128	Crop size 256x256
AUC (test)	0.52	0.665	0.677

- This score was obtained by training only on the dev train set and as the scores were lower we chose not to train it on the eval train set

Model Selection - Three VAE Pipeline

- Our data analysis showed us that the machines had substantial differences in their waveforms and logMEL spectrograms. This gave us the idea of training three models on each of the tree machines in the additional dataset (01, 03 and 05). After that each specific model would be used to predict their corresponding samples (e.g. model_05 would be used to predict the samples with ID_05). However, this did not result in any better results



Optimizing Parameters - Optuna Tuning

- We thought there was a lot to gain by trying out other network configurations (params) than those used in the baseline code. We therefore spent some time on implementing parameter optimization with Optuna
- The tuned parameters included learning_rate, h_dim, z_dim and batch_size
  - `"lr": trial.suggest_loguniform('learning_rate', 1e-5, 1e-1)`
  - `"batch_size" : trial.suggest_int("h_unit", 200, 1000)`
  - `"h_dim": trial.suggest_int("h_unit", 200, 500)`
  - `"z_dim": trial.suggest_int("n_unit", 4, 40)`
- Even so, we did not manage to beat our best solution with the original baseline code with this approach

Model Selection - Self supervised classification

Following the practice outlined in one of the best submissions to the original challenge [1] we implemented a self supervised classification approach. We trained two different classification models for classifying the machine ids and then applied 1 - Softmax(output) to obtain the anomaly score. The idea is that when learning classification the model implicitly learns meaningful features of the dataset.

AUC (test)	ResNet-50	MobNetv2
Dev + eval data	0.805	/
Eval data	0.819	0.696

Model Selection - Baseline VAE

- Since the baseline model gave a reasonably good result, one of our approaches was to further improve the existing VAE
- We did this by making use of different amounts of epochs, both for the additional training set alone and the concatenated set with the additional and the original training set. A summary of these attempts can be seen in the table:

Attempt no.	No. of Epochs	Development dataset?	Additional dataset?	Score
1	100	Yes	No	0.757
3	50	Yes	Yes	0.812
2	100	No	Yes	0.833
4	100	Yes	Yes	0.839

Conclusions

- After trying different models and parameters, we still got the best score with the baseline code with VAE, training with both datasets and 100 epochs
- We can conclude that it was indeed a huge improvement to use the additional dataset, because they had the same machine IDs as the test dataset
- In our opinion VAE performed superiorly to other methods we tried because it attempts to learn the underlying distribution of the normal dataset as well as meaningful feature representation while, on the other hand, other methods we tried only attempt to learn meaningful feature representation

Future work

- Include Chroma feature extraction, by making an ensemble model with the model using logMEL
- Do more data analysis on the datasets to find outliers (e.g. maybe there are sound-files labeled normal that actually are really anomalous). This could be done by defining some criteria for a normal sound sample (e.g. 20 000 zero crossings), and drop every sound sample that are labeled normal but deviates largely from this
- Implementing more advanced density estimator, like Group MADE (Masked Autoencoder Density Encoder)