Anomalous Sound Detection using Unsupervised machine learning.

EEE 598: Speech Processing and Perception, Fall- 2022, Srikar Reddy Sai Reddy, and Neeraj Borade.

Abstract~ Anomalous sound detection (ASD) is the task of identifying whether the sound emitted from a machine is normal or anomalous. This is important in industries such as manufacturing, where early detection of anomalies can prevent disruptions to the production line and reduce maintenance costs. Machine learning algorithms for ASD often use features extracted from the sound signal to train and make predictions. One common method for extracting these features is called Mel-frequency cepstral coefficients (MFCCs), which capture the important characteristics of the sound. Autoencoders, a type of neural network, can then be used to train a machine learning model for ASD using the MFCCs as input. This approach allows for real-time monitoring of machine health and performance.

1. INTRODUCTION

Anomalous sound detection (ASD) is the task of identifying whether the sound emitted from a machine is normal or anomalous. It is important because it can help identify potential issues with machines before, they lead to costly downtime or failure. In industries such as manufacturing, where machines are a critical part of the production process, early detection of anomalies can help prevent disruptions to the production line and reduce the overall cost of maintaining the machines.

Furthermore, many machines produce sounds that can indicate their condition and performance. By using machine learning algorithms to automatically detect anomalous sounds, it is possible to monitor the health and performance of machines in real-time. In order to train an effective machine learning algorithm for anomalous sound detection, it is important to have a large and diverse dataset of normal and anomalous sounds. This will allow the algorithm to learn the patterns of normal sounds and accurately identify anomalies.

2. METHODOLOGY AND CONCEPTS

Machine learning algorithms for anomalous sound detection often rely on features extracted from the sound signal to train and make predictions. One common method for extracting features from audio is called Mel-frequency cepstral coefficients (MFCCs). MFCCs are a representation of the short-term power spectrum of a sound and capture the important characteristics of the sound. Once the MFCCs have been extracted from the sound, they can be used as input to a machine learning model. Autoencoders are one type of machine learning model that can be used for anomalous sound detection. They are a type of neural networks that can learn to compress and reconstruct input data, such as MFCCs.

When training an autoencoder for anomalous sound detection, the model is first trained on a large dataset of normal sounds. The autoencoder learns to compress and reconstruct the normal sounds accurately. Then, when presented with a new sound, the autoencoder will reconstruct it using the learned patterns of normal sounds. If the reconstruction error is above a certain threshold, the sound is considered anomalous. In summary, MFCCs are a useful tool for extracting features from audio data, and autoencoders can be used to train a machine learning model for anomalous sound detection. This can help identify potential issues with machines in industries such as manufacturing.

2.1 ABOUT THE DATASET

The data used in this task is a combination of parts of the ToyADMOS [1] and MIMII datasets [2], which contain normal and anomalous operating sounds of six different types of toy and real machines. The anomalous sounds in these datasets were collected by intentionally damaging the target machines. The six types of machines included in the task are: Toy-car and Toy-conveyor from

ToyADMOS, and valves, pumps, fans, and slide rails from the MIMII dataset. All recordings are single-channel and were captured using a fixed microphone. Each recording is approximately 10 seconds long and includes both the machine's operating sounds and ambient noise. The sampling rate for all signals has been reduced to 16 kHz.

For each machine type and machine ID, the development dataset includes around 4 sets of 1000 samples of normal sounds for training, as well as 4 sets of 100-200 samples of both normal and anomalous sounds for testing. [3]

2.1.1 READING THE DATA TO MATLAB

Our code uses the *dir* function in MATLAB to find all files in the current directory that match the pattern *normal_id_01_*.wav*. The **.wav* part of the pattern means that the function will match any file with the .wav extension, while the *normal_id_01_* part of the pattern means that the function will only match files that have that exact string at the beginning of the file name. The *dir* function returns a structure array containing information about the matching files, including the file names.

The code then creates two cell arrays, *wave*, and *fs*, which will be used to store the audio data and sampling rates, respectively, for each of the matching files. The names cell array is used to store the file names. Next, the code uses a for loop to iterate over each of the file names in the names array. For each iteration, the *audioread* function is used to read the audio data and sampling rate for the corresponding file. The audio data is stored in the wave cell array, and the sampling rate is stored in the fs cell array.

Further, we create a new matrix called *toy_car_train* that concatenates the audio data from the *wave* cell array into it. A for loop iterates over the elements in the *wave* cell array, and for each iteration, the audio data for that element is appended as a new column in the *toy_car_train* matrix. This results in a matrix containing all of the audio data from the wave cell array, with each column representing the audio data from a single *.mat* file.

Similar *.mat* files are made for the anomalous and normal sounds provided in the test data. These data files can now be easily used to extract features and deploy a machine learning model.

2.2 FEATURE EXTRACTION

Feature extraction from audio is the process of extracting meaningful and relevant information from audio signals in order to facilitate further analysis. This involves identifying and extracting specific characteristics of the audio signal. The extracted features can then be used as input for various machine learning algorithms, allowing them to accurately classify and analyze the audio.

Feature extraction is an important step in the field of audio analysis, as it allows for the effective and efficient processing of large amounts of audio data. By extracting only, the most relevant and informative features of the audio, algorithms can make more accurate predictions and classifications.

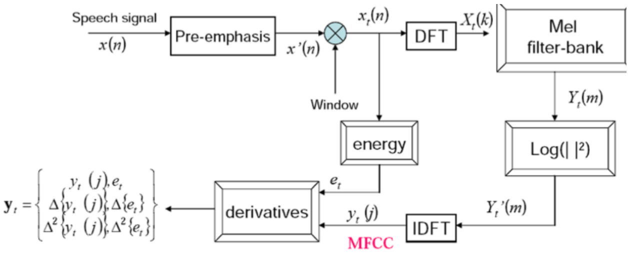
In addition, feature extraction can also help reduce the dimensionality of the data, making it easier to work with and more computationally efficient. This is particularly important when dealing with large amounts of audio data, as it allows for faster and more efficient analysis. Here we extracted MFCC features from each of the audio provided and reduced the dimensionality quite a lot. (Note: average of the extracted features was not taken, all the 12 features have been given as inputs to the autoencoders. But the number of sub-frames were capped.)

2.2.1 MFCC FEATURES

Mel-frequency cepstral coefficients (MFCCs) are a set of coefficients that represent the spectral envelope of an audio signal. They are derived from the mel-scaled spectrum, which is a representation of the spectral power of the signal in terms of mel-frequency bins. The mel-scaled spectrum is obtained by applying a nonlinear transformation to the power spectrum of the signal, which more closely approximates the human auditory system's response to sound.

The MFCCs are derived from the mel-scaled spectrum by taking the discrete cosine transform (DCT) of the log-power spectrum. This results in a set of coefficients that capture the spectral envelope of the signal in a compact and efficient form. MFCCs are commonly used in speech and music processing, as they provide a robust and compact representation of the spectral characteristics of the signal.

We used the code provided by the professor to extract 12 Mel-frequency cepstral coefficients (MFCC) features for each audio file. These features were then reshaped into a single array, with the first 12 elements representing the MFCC features of the first audio file, followed by the features of the second audio file, and so on. It is important to note that we did not take the average of the extracted features; all extracted features were given as input to the autoencoders in order to improve the accuracy of our model.



Speech signal
x(n)
Pre-emphasis
x'(n)
Window
$x_t(n)$
DFT
$X_t(k)$
Mel filter-bank
$Y_t(m)$
energy
Log(| |²)
e_t
derivatives
$y_t(j)$
MFCC
IDFT
$Y_t'(m)$
$\mathbf{y}_t = \left\{ \begin{matrix} y_t(j), e_t \\ \Delta\{y_t(j)\}, \Delta\{e_t\} \\ \Delta^2\{y_t(j)\}, \Delta^2\{e_t\} \end{matrix} \right\}$

Fig 1: *Block diagram of the MFCC feature extraction.*

2.3 AUTOENCODERS

Autoencoders are a type of artificial neural network that are used for unsupervised learning. They consist of an encoder network that maps the input data onto a lower-dimensional latent space, and a decoder network that maps the latent representation back to the original input space. The objective of an autoencoder is to learn a compact, low-dimensional representation of the input data that captures the essential characteristics of the data.

The encoder and decoder networks are typically trained jointly, with the objective of minimizing the reconstruction error between the original input and the output of the decoder. This is typically achieved through the use of an objective function, such as the mean squared error or binary cross-entropy loss. The encoder network learns to compress the input data into a latent representation that captures the essential characteristics of the data, while the decoder network learns to reconstruct the input data from the latent representation.

They can also be regularized using techniques such as weight decay or dropout, in order to improve their generalization ability and prevent overfitting.

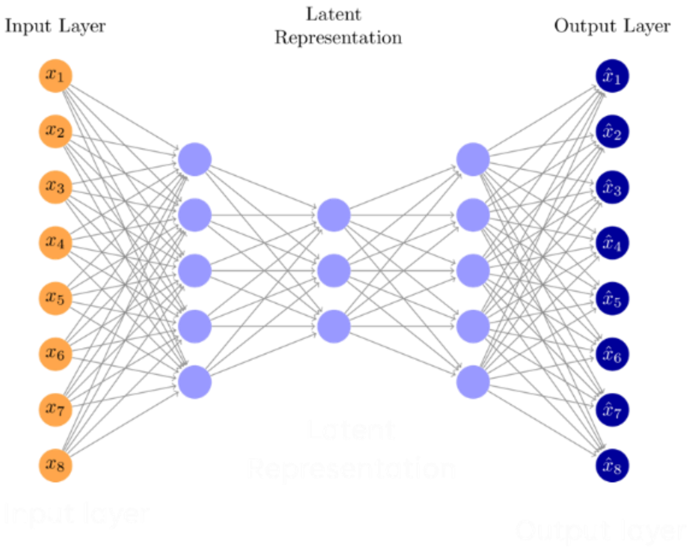


Fig 2: *Typical Autoencoder representation.*

2.3.1 TRAINING MODEL

While traditional autoencoders do not utilize temporal features in the data, we use a long short-term memory (LSTM) autoencoder to incorporate this information. LSTM is a type of recurrent neural network (RNN) that is well-suited for processing sequential or time series data. Recurrent neural networks (RNNs) are a type of artificial neural network that are well-suited for processing sequential or time series data. Unlike traditional feedforward neural networks, which only have a fixed number of inputs and outputs, RNNs have loops in their architecture that allow them to operate on sequences of arbitrary length. This makes RNNs particularly useful for tasks such as language modeling, speech recognition, and time series forecasting.

In this work, we train an LSTM autoencoder model using only the MFCC features extracted for normal machine sounds and no anomalous sounds from the training dataset. This allows the model to minimize the reconstruction error of the normal training data. This approach is based on the assumption that autoencoders cannot accurately reconstruct sounds that are unlike those used in training, so unknown anomalous test sounds will tend to have higher reconstruction errors than normal test sounds.

2.3.2 TRAINING PARAMETERS

Our LSTM Autoencoder model:

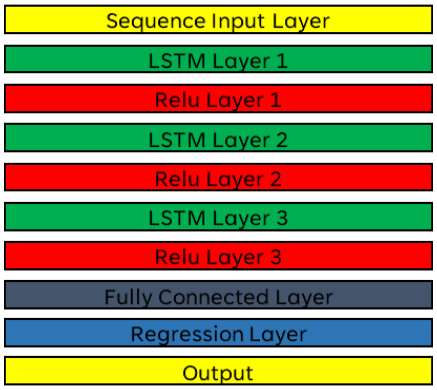


Fig 3: *Model representation.*

- Activation function - ReLu
- Optimizer - Adam
- Mini-batch size - 500
- Iterations: 4000
- training time: 186 minutes

3. TESTING AND RESULTS

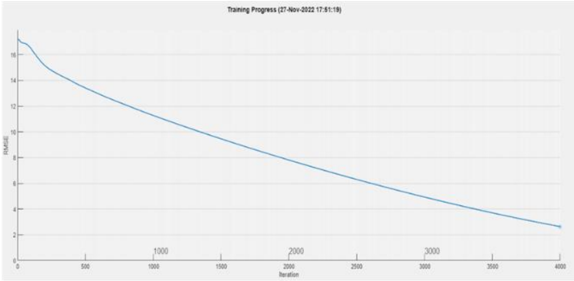
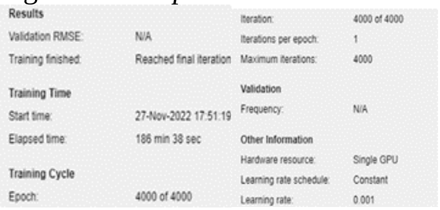


Fig 4: *loss vs epochs curve.*



Results
Validation RMSE: N/A
Training finished: Reached final iteration

Training Time
Start time: 27-Nov-2022 17:51:19
Elapsed time: 186 min 38 sec

Training Cycle
Epoch: 4000 of 4000

Iteration: 4000 of 4000
Iterations per epoch: 1
Maximum iterations: 4000

Validation
Frequency: N/A

Other Information
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.001

Fig 5: *Model results .*

After the model is trained only on the normal sounds in the training dataset. The model is tested using the testing dataset(MFCCs of test audios). It contains both the normal sounds and anomalous sounds of the machines. The reconstruction loss for normal sounds is less as the model is trained on normal sounds. But the reconstruction error for anomalous sounds is high.

We tried many thresholds for the reconstruction error. A threshold of **0.5*(mean of the reconstruction error of the test dataset)** gives optimal results based on our experiments with the threshold.

3.1 PREDICTION AND ACCURACY

We used the aforementioned threshold in our experiments and achieved the following results: **80.97% of anomalous sounds were correctly identified as anomalous**, and **92.26% of normal sounds were correctly classified as normal**. The threshold can be adjusted according to the desired False Positive and Missed detection values, as these may vary depending on the user and the specific application. Additionally, the accuracy of the model can be improved by increasing the number of training epochs, as the error will continue to decrease. An optimal threshold, independent of the testing errors, would also work well on previously unseen data.

REFERANCES

[1] Y. Koizumi, S. Saito, H. Uematsu, N. Harada, and K. Imoto,“ToyADMOS: A Dataset of Miniature-Machine OperatingSounds for Anomalous Sound Detection,” Proc. of the Workshop on Appl. of Signal Process. to Audio and Acoust. (WASPAA), 2019.

[2] H. Purohit, R. Tanabe, K. Ichige, T. Endo, Y. Nikaido, K. Suefusa, and Y. Kawaguchi, “MIMII Dataset: Sound Dataset forMalfunctioning Industrial Machine Investigation and Inspection,” Proc. of DCASE, 2019.

[3] Koizumi, Y. et al. (2020) Description and discussion on DCASE2020 challenge Task2: Unsupervised ... Available at: https://dcase.community/documents/workshop2020/proceedings/DCASE2020Workshop_Koizumi_3.pdf (Accessed: November 28, 2022).