Applications of Deep Learning: Sequence Modelling in Industrial Sound Analytics

A Thesis

submitted to

Indian Institute of Science Education and Research Pune in partial fulfilment of the requirements for the BS-MS Dual Degree Programme

by

Abhishek Choudhary

Indian Institute of Science Education and Research Pune

Dr Homi Bhabha Road,

Pashan, Pune 411008, INDIA.

April 2021

Supervisor: Mr Prabhanjan Borwankar

TAC member – Dr Sourabh Dube

Certificate

This is to certify that this dissertation entitled **Applications of Deep Learning: Sequence Modelling in Industrial Sound Analytics**, towards the partial fulfilment of the BS-MS dual degree programme at the Indian Institute of Science Education and Research, Pune, represents study/work carried out by Abhishek Choudhary at Indian Institute of Science Education and Research under the supervision of Prabhanjan Borwankar, Data Scientist, ASquared IoT Pvt. Ltd, during the academic year 2020-2021.

Supervisor's Name: Prabhanjan Borwankar

Signature:

Committee:

Name of Supervisor: Mr Prabhanjan Borwankar

Name of TAC: Dr Sourabh Dube

This thesis is dedicated to *Mari Abinaya*,

Arpita Misra & Imrankhan;

my backbone, my support

my strength

Declaration

I hereby declare that the matter embodied in the report entitled **Applications of Deep Learning: Sequence Modelling in Industrial Sound Analytics** are the results of the work carried out by me at the Department of IDC, Indian Institute of Science Education and Research, Pune, under the supervision of Prabhanjan Borwankar and the same has not been submitted elsewhere for any other degree.

Name: Abhishek Choudhary

Sign: 

Date: 15-05-2021

Table of Contents

List of figures

Abstract

In today's world of data and information, deep learning has proven to be the ace in data science applications. While we hear much of deep learning concerning computer vision, NLP (Natural Language Processing), audio analysis and its usefulness to ML have been recognised in the past decade.

Audio data analysis analyses and understands audio signals captured by digital devices such as microphones. It has widespread applications, including automatic speech recognition (ASR), digital signal processing, medical diagnostics, music classification, tagging and generation.

Factory machinery is prone to malfunction or breakdown, resulting in a significant loss of time, money and resources.

So, there is a need for an efficient and affordable solution to this problem.

So, there is a rising interest in building deep learning models for monitoring machines. The Industrial Internet of Things (IoT) and data-driven techniques have revolutionised the manufacturing industry.

Many new and innovative approaches have been attempted to build algorithms to monitor machinery.

Many new and innovative approaches have been attempted to build algorithms to monitor machinery.

Sound Classification is one of the most popular applications of Deep Learning. It involves classifying sounds based on various frameworks like music genre prediction, urban sound classification, identification of speaker or tone from sound clips.

It has huge applications, especially in manufacturing industries, like predicting machine failures, spotting faulty pieces of machinery.

Acknowledgements

I am grateful to my supervisor Mr Prabhanjan Borwankar for his endless help and support.

His constant guidance and encouragement have been invaluable in this project.

I am also grateful to Dr Anand Deshpande, who has provided me with the opportunity to work on this project. His originality has been a constant source of suggestion ever since I first interacted with him.

I also want to thank Dr Sourabh Dube for his enormous, kind support throughout my project.

I want to express my deep gratitude to all the professors of IISER Pune for helping me out in my personal and academic life.

I want to thank Dr Aparna Deshpande for her unbounded love, care and support since I first met her. Her constant support and encouragement have been instrumental to me both in my personal and academic life.

I want to thank all my friends for their boundless love, care and support throughout the programme. It is not wrong to say that it would have been impossible for me to reach this stage without their love, care, and support.

I would also like to express gratitude to my family for their unwavering love, care and support throughout all my endeavours.

It is a pleasure to express my sincere gratitude to all associated me at all phases of the programme.

CHAPTER 1: INTRODUCTION

Acoustic data have always been crucial in science and engineering.

Apart from allowing us to measure some property or physical variables, it has also provided us with valuable insights in various fields, including machine interpretation of human speech and animal vocalisations, geophysical imaging structures in the ocean.

However, data analysis is likely to get complicated by several challenges. It includes corrupt or missing data, reverberations, unnecessary data, multiple acoustic arrivals of a single event, etc.

Using conventional methods increases the amount of human effort required to identify acoustic features and rapidly become limiting as the datasets size increases.

Moreover, patterns may exist in the data that are not easily recognised by human cognition.

Machine learning (ML) techniques have made huge advances in automated data processing capabilities and pattern recognition across several fields.

ML in acoustics is not very old and is rapidly developing. Moreover, the increase in parallel computing power and relatively new neural network architectures specifically for sequential data has boosted it even more.

CHAPTER 2: BACKGROUND INFORMATION

2.1 Definitions and Terminologies

DATA SCIENCE

The term Data Science is a broad term that encompasses various topics from a wide range of subjects. It has emerged as an interdisciplinary field that unifies statistics, data analysis, informatics, and related methods to understand, analyse and predict the actual phenomena using data.

ARTIFICIAL INTELLIGENCE (AI)

In simple words, we can define intelligence as the ability to process information such that it can be used to inform a future decision.

Artificial intelligence or AI is a science that focuses on building algorithms that can do precisely this, i.e. to build algorithms to process information such that they can inform future predictions.

Using AI, a computer (and its systems) can learn to perform complex tasks that usually require human intervention. There are numerous applications such as visual perception, speech recognition, language translation and various decision-making tasks.

MACHINE LEARNING (ML)

ML is a subset of AI.

In contrast to traditional and conventional methods, it works on the idea upside down.

Rather than providing the rules for a particular dataset and then getting predictions, in this case, machines are provided with the raw data, and they are left to figure out the pattern by themselves through learning based on training and success criteria.

Currently, the preferred approach to ML is the Artificial Neural Network or ANN.

DEEP LEARNING (DL)

The word deep signifies that each neuron of a layer is connected to every other neuron in the adjacent layers.

NEURAL NETWORK (NN)

The neural network is a subset of DL.

Geoffrey Hinton conceptualised deep learning in the 1980s. At present, we consider him to be the founding father of the field of deep learning.

Hinton's structure was called an artificial neural network (or ANN for short).

We can represent the relationship between AI, ML, and DL by a Venn diagram

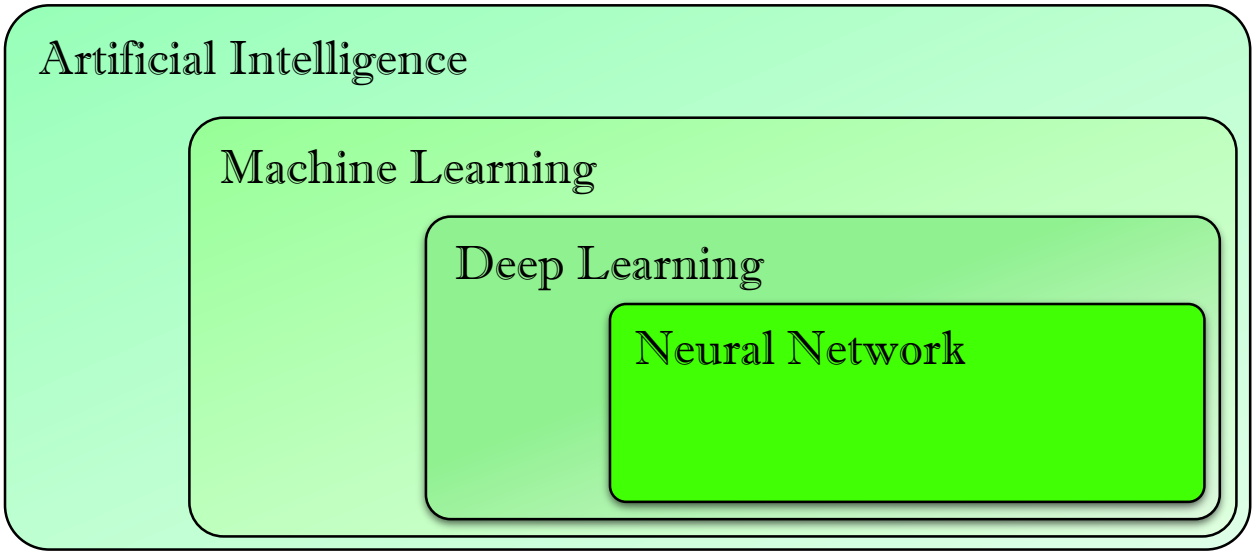


Figure 1

PERCEPTRON – A single neuron

Neurons are the structural building blocks of deep learning and every neural network.

A single neuron, also known as a perceptron, is the node through which data and computations flow.

ACTIVATION FUNCTIONS

The purpose of the activation functions is to introduce non-linearity into the network.

This allows us to approximate arbitrarily complex functions and gives us successful results.

There are mainly four types of activation functions widely in use today.

Threshold functions

Sigmoid functions

Rectifier functions, or ReLUs

Hyperbolic Tangent functions

2.2 Types of Neural Network Architectures

Several kinds of neural network architectures exist based upon the nature of the task it is required to complete.

However, this project requires focus on mainly three types of neural network architectures.

Artificial Neural Network or ANN

ANN is the simplest architecture consisting of an input layer, hidden layers and one output layer.

The following diagram explains the architecture of ANN.

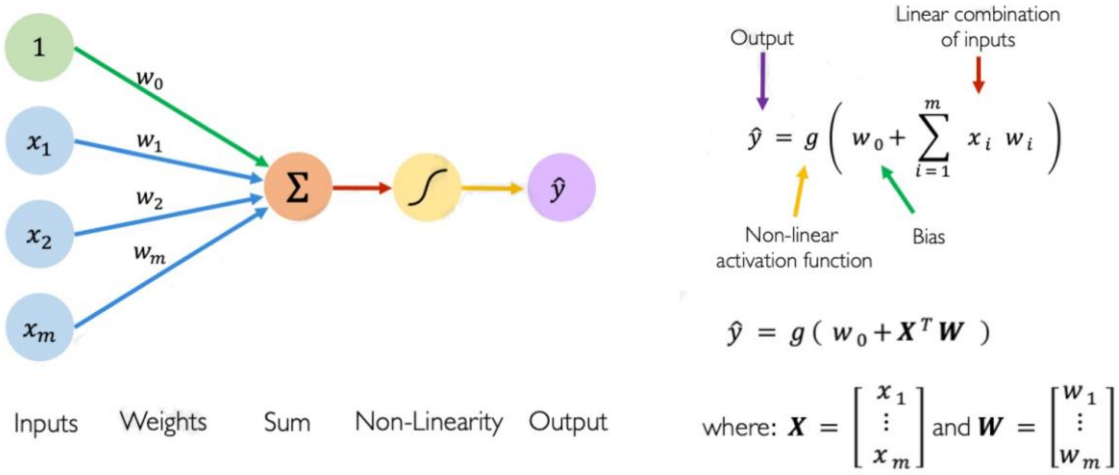


Figure 2

As we can see from the above figure, inputs are fed through the input layer. Then, certain weights are assigned to each neuronal connection before the data is passed to the next layer.

Before passing on to the output layer, the data is passed through an activation function to introduce non-linearity.

And then, we receive the output.

The equations governing the above architecture is given as follows.

We represent the inputs being fed and the weights using column matrices **X** & **W**

$$\mathbf{X} = \begin{pmatrix} X_1 \\ X_2 \\ . \\ . \\ . \\ X_m \end{pmatrix} \qquad \mathbf{W} = \begin{pmatrix} W_1 \\ W_2 \\ . \\ . \\ . \\ W_m \end{pmatrix}$$

The activation is given by

$$\mathbf{a1} = \mathbf{X}^T\mathbf{W} + W_0$$

$$\begin{bmatrix} X1 & X2 & X3 & . & . & . & Xm \end{bmatrix} \begin{pmatrix} W1 \\ W2 \\ . \\ . \\ . \\ Wm \end{pmatrix} + W_0$$

For the next layer, this becomes the input, and certain weights are assigned to each neural connection. The new activation is calculated, and the data is passed on to the next layer.

Convolutional Neural Network or CNN

This architecture is best suited for image classification.

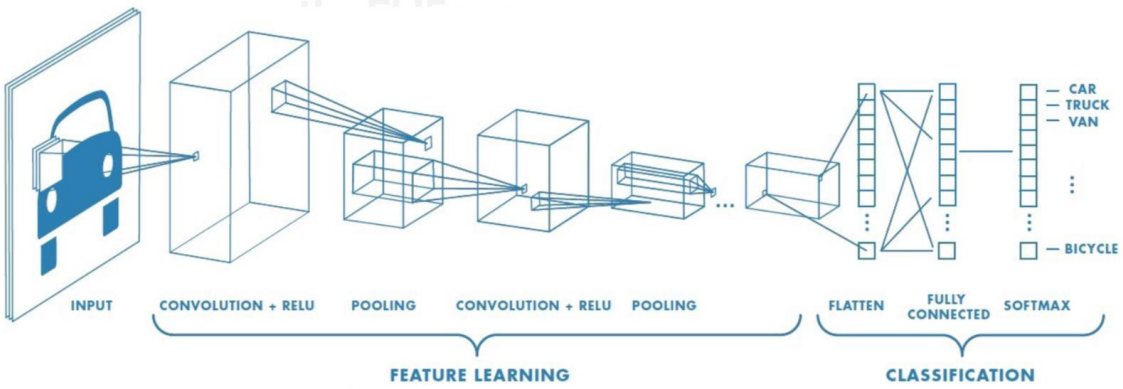


Figure 3

A Convolutional Neural Network or CNN is a deep learning algorithm that is the ace of deep computer vision. It is similar to the other classification algorithms; besides it captures spatial and temporal features of an image by various techniques such as convolution, filtering, and pooling layers.

The architecture performs better on the image dataset due to its explicit assumption that the inputs will be images only. This allows for more freedom, and thus, image specific properties can be encoded in it.

Recurrent Neural Network or RNN

RNN is used where we have a sequence of data in the time domain. It is a specific type of ANN with recurring connections to itself besides having connections to the other layers.

These recurring connections to the self, make it possible to retain data/feature from earlier layers and update the subsequent layers based on the input and previous layers.

The following diagram helps in explaining the working of an RNN

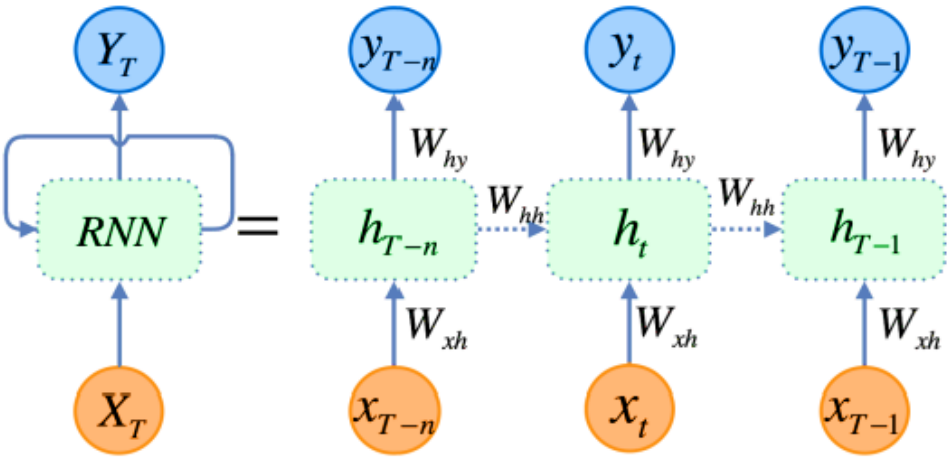


Figure 4

In the above diagram, the cell named “RNN” is unfolded. The current state of cell is represented by h_t at time t. X_t is the input given to the cell at time t. It has certain weights associated with it and is represented by a matrix W_{xh}.

Now as we are working with a sequence of data, we need the information from previous layers. The cell labelled as h_{T-1} represents the state at time T-1. The information from the previous layer is passed on to the next layer at time t by the weights matrix W_{hh}.

Finally, we get the output for time t as y_t with updated weights matrix given by W_{hy}.

RNN suffers from two major issues i.e.

Vanishing and

Exploding gradients.

To overcome this problem, LSTM's are introduced.

Long Short Term Memory or (LSTM)

LSTM overcomes the problem of vanishing gradients by using gates that control the flow of information. So far, LSTM continues to be one of the most effective and innovative solutions. Hochreiter and Schmidhuber developed the LSTM architecture in around 1997.

These gates can learn which data is important to keep and which data is not useful for the task, and then it will allow the flow of data accordingly.

The LSTM network architecture consists of recurrently connected subnets which are also known as memory blocks.

It relies on gated cells to track information throughout many time steps. LSTM modules contain computational blocks that control the flow of information.

Gates optionally let information through, for example, via a sigmoid neural net layer and pointwise multiplication.

Each unit consists of one or more self-connected memory cells and three multiplicative units that provide continuous analogues of write, read and reset operations for the cells.

LSTM works in four basic steps:

Forget, Store, Update, Output

LSTM forgets irrelevant parts of the previous steps and stores relevant new information into the cell state. It selectively updates cell state values. The output gate controls what information is sent to the next step.

These four steps are shown in the figure 5.

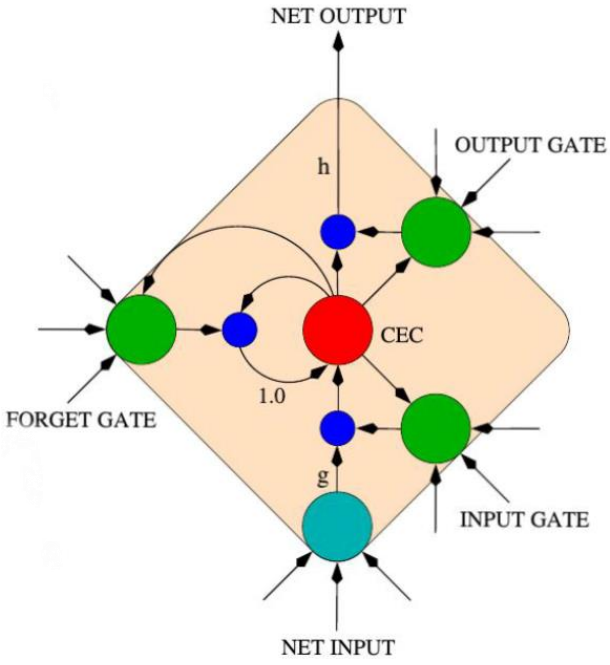


Figure 5

2.3 Metric used to evaluate the performance of a model

CONFUSION MATRIX

A confusion matrix is a type of table construct in a machine learning classification problem, which helps infer the performance of an algorithm on a set of data for that the actual values are known beforehand.

It helps measure the Accuracy, Precision, Specificity, Recall and F-scores.

Most importantly, it is used to measure the AUC-ROC Curve.

Definitions of various terms related to the Confusion matrix

True Positive (TP):

Interpretation: The algorithm predicted positive, and the actual value is true.

True Negative (TN):

Interpretation: The algorithm predicted negative, but the actual value is true.

False Positive (FP): (Type 1 Error)

Interpretation: The algorithm predicted positive, but the actual value is false.

False Negative (FN): (Type 2 Error)

Interpretation: The algorithm predicted negative, and its actual value is false.

Accuracy

Accuracy is a measure of how many correct predictions the model made for the complete test dataset.

It is calculated by dividing the sum of true positives and true negatives by the total number of instances.

Precision

Precision is concerned with how many of the instances the model classified as positive; were positive. In order to calculate precision, the number of true positives is divided by the sum of false positives and true positives.

Recall

Recall is a measure of the proportion of true positive examples that a machine learning model has classified.

It is calculated by dividing actual positive instances by the sum of false negatives and true positives.

Specificity

It measures the true negative rate or the number of instances the model classified as negative that were truly negative.

Specificity is calculated by dividing the number of instances classified as negative by the sum of false positives and true negatives.

F1-score

We can combine precision and recall in a single metric called F-score.

When equal importance is given to recall and precision, we use F-score to optimise the model.

CHAPTER 3: PROBLEM STATEMENT

The purpose of this project is to try and build a neural network architecture that can classify different sounds accurately. This classification will, in turn, be used to classify the sounds produced by machines at different conditions, like when it is functioning well when it starts to dis-function to the point where it is not at all working what it is required to perform. This particular application of ML is known as Predictive Maintenance of Machines.

Specific to A2IoT, the architecture built will be used to classify sounds from a factory environment among the background industrial noise.

The condition is to detect whether there is a gas leakage or not at the desired location.

The algorithm will classify sounds in two categories i.e. normal air sound and air leakage sound, amidst the background industrial noise.

This classification is of immense importance as it can prevent accidents that may occur due to gas leakages that are flammable or hazardous. Thus, it not only saves the loss of property, capital, resources but can also save lives that cannot be compromised.

I used audio data consisting of situations with a gas leakage sound versus when there is no gas leakage with the normal industrial background noise.

CHAPTER 4: MATERIALS & METHODS

4.1 Strategy Outline

The strategy used to address the problem is to first study the various types of neural network architectures that are already present to classify some common sounds.

This is done as there are many open source datasets that are available, free of cost which otherwise, can cost a huge amount of money to collect it.

Secondly, huge amounts of data are required to train any model and getting such high volumes of data is quite difficult.

So, the method is to first use some open source dataset and classify its sounds using various combinations of neural network architectures.

Then, the second step is to take the models which gives good accuracy and train it on the custom dataset i.e. air leakage versus non-air leakage (normal condition).

After training the chosen models on custom dataset, then test the model on a test dataset which is entirely new for the model.

Finally, compare the results and explain the observed results.
Explain the insights that came after dong the project.

4.2 General Procedure for Sound Classification

This flowchart depicts the typical processing flow for sound classification algorithm

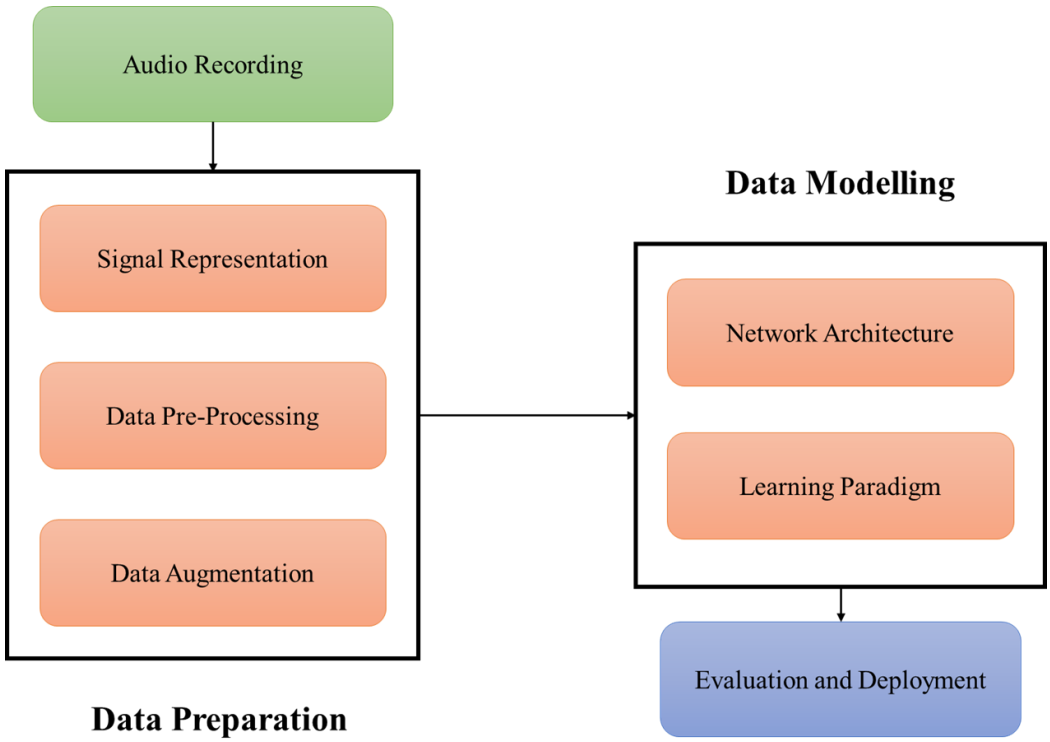


Figure 6

First of all, sound is recorded using various recording machines. Once this is done, the raw audio signals need to be converted and represented in desired physical parameters. The most common representation is Amplitude v/s Time domain.

Then the next step is to pre-process the audio data. It involves converting audio data into desired formats, chopping long audio files into smaller segments, augmenting audio files etc.

Before I go on to describe classification procedure in detail, a little background in sound analysis is needed. So, a brief digression.

Few Concepts and Terminologies related to Audio Data Analysis

In Physics, sound is defined as a disturbance or vibration that propagates as an acoustic wave in an intervening medium like air, glass, water etc.

It propagates as a wave due to oscillations in pressure, stress, particle displacement, velocity etc. in a medium having some internal force like elastic, viscous or the superposition of such propagating oscillations.

When we need to represent sound of an audio signal, we use physical variables such as frequency, wavelength, amplitude, etc.
A typical audio signal can be represented as a function of Amplitude and Time.

The figure below shows a graph of sound of a dog barking in amplitude v/s time domain.

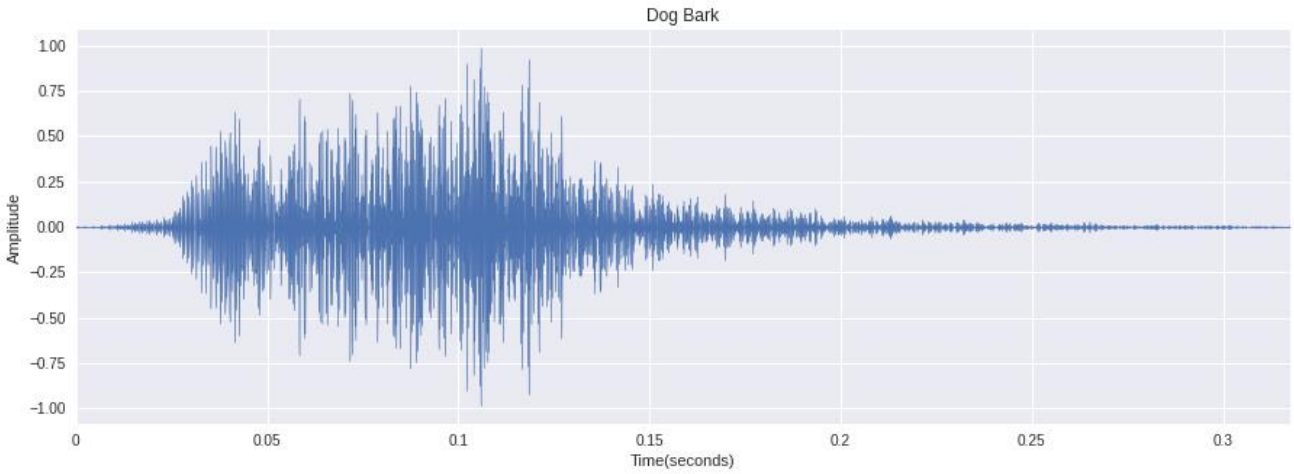


Figure 7

There are devices built that help us catch these sounds and represent them in a computer-readable format. Examples of these formats are:

- wav (Waveform Audio File) format
- mp3 (MPEG-1 Audio Layer 3) format
- WMA (Windows Media Audio) format

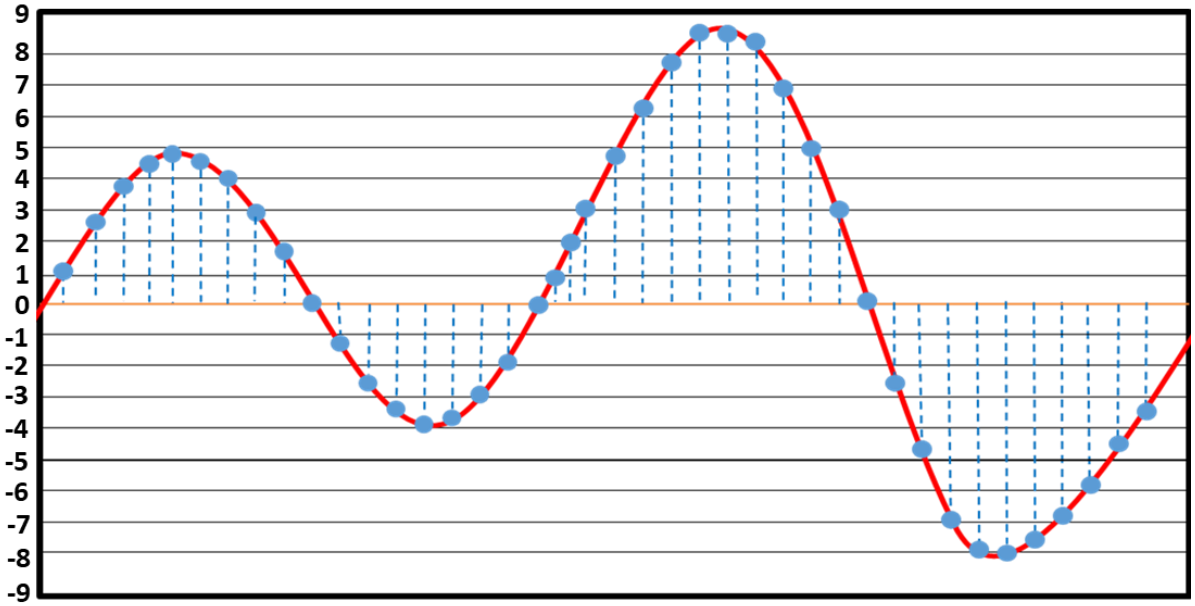


Figure 8 A general sound-wave (in red) and its digital representation (in blue)

Audio Signals: Raw Data

A raw sound signals needs to be converted in a digital form so that it can be used as inputs to feed in the computer program. Audio files are represented in digital form using a time series of numbers that depicts the amplitude at every step.

Sampling and Sampling frequency

In audio signal processing, we need to first transform the audio file into time series of discrete values. It is done so that the computer can read an audio file.

The process of discretising the continuous audio signal is called sampling.

The sampling frequency is defined as the number of samples taken over some fixed time interval. The higher the sampling frequency, the higher is the audio resolution.

Bit-depth

The bit-depth measures the total number of possible values any sample can take while sampling an audio file.
Higher bit-depth implies better audio fidelity.
The typical bit depth is of 16 bit. However higher bit depths like 32 bit, 64 bit also exists.
It means a sample can take 2^{16}= 65,536 range of amplitude values.

The sound excerpts used in this project are digital audio files in .wav format. Sound waves are digitised by sampling them at discrete intervals known as the sampling rate. Generally, the sampling rate of an audio recording is 44.1kHz which means samples are taken 44,100 times in one second.

Each sample consists of the amplitude of the wave for some interval of time.
As discussed earlier, bit depth determines how detailed a sample will be, also known as the signal's dynamic range.

The array generated from the sound signal in figure 5 will look something like

[1, 3, 4, 5, 4, 3, 2, 1, 0, -1, -2, -3, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

This 1-dimensional array of numbers is the digital form of the audio signal which we can feed in our model as input.

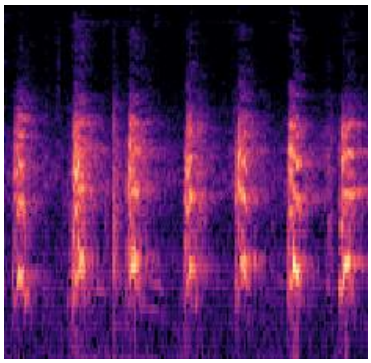
Spectrograms

A spectrogram is used to plot frequency vs time plot.
We take the Spectrums at different instances in time and then join them into a single plot.

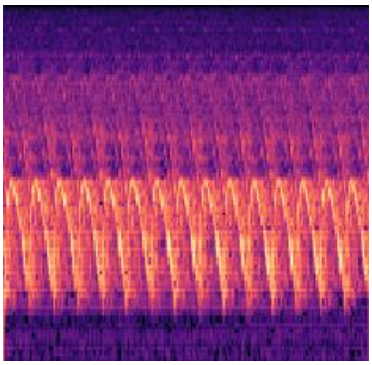
Each vertical slice is essentially the Spectrum of the signal at that instant in time.

It shows how the signal strength is distributed among the frequency range found in the signal at that instant.
Colour-code is used to indicate the amplitude or strength of each frequency.

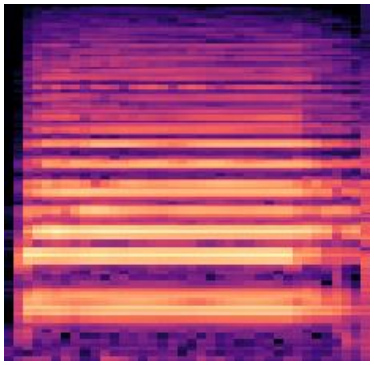
Few spectrogram images of various sounds



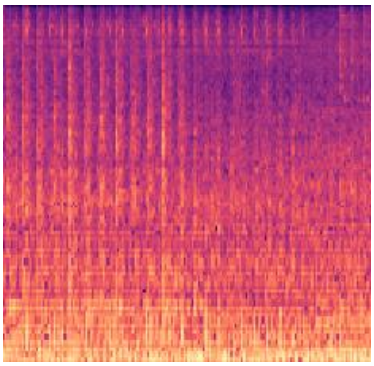
Dog bark



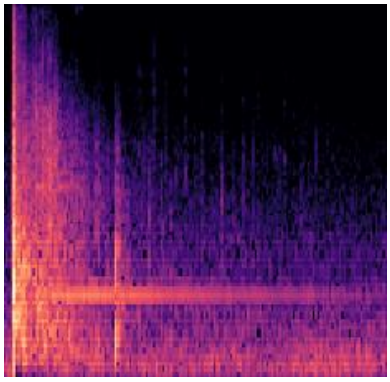
Siren



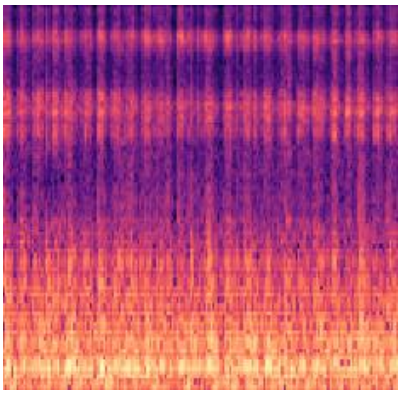
Car horn



Jackhammer



Gunshots



Drilling

Figure 9

Recording audio and creating a dataset

This is the starting point where audio data is gathered. This is usually a field work and is done by recording machines as per the requirement from appropriate place.

Audio Pre-Processing

Once the audio files are recorded, it needs to be processed, checked for errors, missing or inappropriate recordings etc. Afterwards, it is converted into different formats as per the requirement. For ex. wav format is preferred as it takes very less amount of space to store and is easy to share.

(This is not done by me. All the datasets I received were already in wav format with metadata for each of the three dataset is provided with the data).

Feature Extraction

Generally, audio processing involves feature extraction relevant to the task in question, followed by decision-making schemes, including detection, classification, evaluation.
For many of the above processes, there exist python libraries that do the job.

The training data, which are raw audio files are converted into correct formats which are compatible with the model. Generally, any standard format will work provided the program is written such that it can take that particular format as input.
This audio pre-processing takes place while reading and loading the audio data.

As the entire dataset can occupy huge memory, all the data are not converted into readable formats at once.

During runtime, when we train the model, audio files are converted in proper formats one batch at a time. We apply all the series of transforms to the audio files which are present in the loaded batch.

Loading and reading the audio files

The audio files are loaded and read in the model using various Python Libraries such as Librosa and PyTorch, PyAudio etc.

For this project, each audio file in all the datasets are in ".wav" format and PyTorch is mainly used for loading and reading audio files.

Converting sound files to the same channel

Some of the sound files are mono (i.e. one audio channel), while most of them are stereo (i.e. two channels). Since the model needs to feed data of the same dimensions, all the mono files are converted into stereo by duplicating the first channel to the second.

Standardising sampling rate

Few audio files have a sampling rate of 48kHz, while most of them have 44.1kHz.

In order to convert all audio files into arrays having the same dimensions, we need to standardise the sampling rate.

Resizing the audio files

Different audio files need not be of the same length. We resize the files by extending their duration via padding them with silence or truncating it. The method for doing it is kept in the audio utility class.

Generating Mel Spectrograms

Spectrograms are generated by doing a Fourier Transformation on the audio data. However, there exist many Python libraries such as Librosa and PyTorch that does the job for us.

For deep learning models, Mel-spectrograms are used.

Data Augmentation

In case if we do not have sufficient data, then a technique known as data augmentation is used to increase the diversity of the dataset.

Several transformations are done on the existing data like rotation, reflection, cropping to modify the data.

A2IoT data preparation (Factory custom data)

First, the entire data of 30 minutes is divided into clips of 3 second each.

After that, the metadata file is created and is saved as a CSV file having two classes

Normal_sound

Air_leakage_sound

Then the dataset is split into training and testing using train_test_split method in 80:20 ratio.

CHAPTER 5: DESCRIPTION OF DATASETS USED IN THIS PROJECT

5.1 UrbanSound8K Dataset

This Dataset contains 8732 labelled audio files (<=4s) of urban sounds from 10 different classes:

air_conditioner, car_horn, children_playing, dog_bark, drilling, enginge_idling, gun_shot, jackhammer, siren, and street_music.

All these categories of sounds are drawn from the urban sound taxonomy, and each excerpt is taken from field recordings.

The audio files are contained in a folder named audio. The wav files are divided into ten sub-folders, named fold1 to fold10.

Another folder named metadata contains the 'UrbanSound8K.csv' file. The CSV file contains all the relevant information of each of the audio file in the Dataset.

The class label is a numeric ID from 0–9 for each of the ten classes.

A numeric identifier of the sound class:

0 = air_conditioner
1 = car_horn
2 = children_playing
3 = dog_bark
4 = drilling
5 = engine_idling
6 = gun_shot
7 = jackhammer
8 = siren
9 = street_music

All the audio files are in **.wav** format.
The sampling rate, bit depth, and the number of channels are the same as those of the original file uploaded to the website, Freesound.

It is available on the internet as a .zip file of size around 6 GB.

It consists of a corpus of urban sounds from 10 different classes as given in the table:

Number of files in each of ten classes in UrbanSound8K dataset

dtype: int64

CLASS	Number of wav files
Children_playing	1000
Street_music	1000
Jackhammer	1000
Air_conditioner	1000
Dog_bark	1000
Drilling	1000
Engine_idling	1000
Siren	929
Car_horn	429
Gun_shot	374

5.2 Set of Factory data (A2IoT)

This set of data is recorded by the company ASquared (A2IoT) IoT Pvt. Ltd. in an environment of industrial work like welding, hammering, cutting etc. is going on. The data mainly focuses on two classes of sounds namely one with gas leakage sound and another with no gas leakage sound with industrial work in the background.

There are two sets of data each consisting of a single .wav audio file of approximately 30 minutes. The metadata for both the audio clips is also provided as a CSV file.

This set of data is shared with me as a .zip file each having a size of around 2 GB.

Hereafter, this set of data will be called as “Set of Factory Data (A2IoT)”.

Set of Factory data (A2IoT) – 01

As described above, the first dataset consists of a single 30 min audio file in .wav format. This one long audio file is to be divided into equal segments using any suitable python program. I divided this 30 min long audio file into small clips of 3 seconds each. Each audio clip is identified and labelled as either air-leakage sound or no air leakage sound using the metadata which is already provided.

The metadata of the audio files are saved as a CSV file.

Number of audio files in each of the two classes

CLASS	Number of audio files
Normal_sound	312
Air leakage_sound	265

Set of Factory data (A2IoT) – 02

The second set of data also consists of a single 30 min audio file in wav format with the metadata provided in a CSV file.

Similar to the first set of data, it is required to clip this long audio file into equal segments. I divided and made small clips of 3 seconds each.

Their metadata file is saved as a CSV file.

Number of wav files in each of the two classes

CLASS	Number of wav files
Normal_sound	317
Air_leakage_sound	289

CHAPTER 6: RESULTS

6.1 Various approaches for Sound Classification

The most common method for the classification of sound is the CNN RNN architecture.

RNN is used to parse the temporal sound data, and then we create Mel-spectrograms.

Then CNN is used as a fingerprint to recognise various kinds of sounds and classify them.

In this project, I have classified sounds using a combination of neural network architectures described in the section 2.2

The first model (M1) is a simple CNN based architecture.

The second model (M2) is based on CNN RNN architecture.

The architecture of third model (M3) is a combination of CNN, RNN and LSTM.

Here, I have written down all the results which I have obtained after running each of the three models on all the datasets mentioned in chapter 5.

The way I have written my observations here is for each model, I have written the observations and results for all the three datasets one by one. Then, I went on to my second model and has written the results obtained after running model 2 on each of the three dataset one by one.

For example- I started with my first model which is M1, mentioned the dataset on which I run this model (M1) i.e. UrbanSound8K dataset.

Then, in a table, I have shown the number of audio files used for training purpose and the number used for testing the model.

Now moving to second set of data i.e. Set of Factory Data (A2IoT)-01. In a table, again mentioned the number of audio files used for training purposes and for testing the model.

MODEL 1 (M1)

It is based on CNN architecture.

DATASET: UrbanSound8K

Dataset split	Number of wav files
Training	6985
Testing	1747

The above table shows the split of audio files which are used for training the model and testing it.

Here, 6985 audio files are used to train the model. Afterwards, the model is tested on testing data which consists of 1747 audio files.

METRICS

Train Epochs: 90

METRICS	Training	Testing
Accuracy	0.9782	1.1047
Loss	0.0681	0.8369

The above table shows the results obtained after running the model (M1) on UrbanSound8K dataset. I ran the program for 90 epochs or cycles.

We see that we get a high accuracy while training. During testing we get 100% accuracy.

DATASET: Set of Factory data (A2IoT) – 01

Dataset split	Number of wav files
Training	368
Testing	116

Now I repeated the same process again but with a different dataset. Here, I used the set of data given to me by the company.

The total number of files were 484 in which a 75-25 split is performed and 368 audio files are kept for training and the rest, 116 audio files are for testing the model.

METRICS

Train Epochs: 90

METRICS	Training	Testing
Accuracy	1.000	0.9483
Loss	2.0892e-05	0.2833

Confusion Matrix

		TRUE LABEL	
		Air-Leakage	No Air-Leakage
PREDICTED LABEL	Air-Leakage	47 TP	4 FP
	No Air-Leakage	2 FN	63 TN

Accuracy = 0.9483

Precision = 0.9215

Recall = 0.9591

F1-Score = 0.9399

It is pretty good accuracy for testing but the 100% accuracy for training might be a result of too less number of data and also it is used for binary classification in which case the algorithm might have learnt few specific features while training which made it biased to look only for those features during testing, resulting in 100% accuracy.

The above table is the confusion matrix which is discussed in section 2.3

We observe that out of 116 audio files, the model has classified 110 audio files correctly and 6 files incorrectly.

However, the false negative which is 2 here, is quite low. This is a good sign as in this case, if there is a wrong classification being air-leakage when in reality, it is not the case, then it will not cost much. While if there is actually an air leakage which is falsely classified, then it may lead to accidents and may cause great loss.

DATASET: Set of Factory data (A2IoT) – 02

This is the second set of data given by the company which is mentioned in chapter 5. This dataset is used only for testing the model (M2).

Number of audio files: 606

METRICS	Testing
Accuracy	0.7980
Loss	0.6383

Confusion Matrix

		TRUE LABEL	
		Air-Leakage	No Air-Leakage
PREDICTED LABEL	Air-Leakage	230 TP	71 FP
	No Air-Leakage	52 FN	253 TN

Accuracy = 0.7980

Precision = 0.7641

Recall = 0.8127

F1-Score = 0.7876

The model is being tested on this entire data which is new to the model. Here we see a sharp drop in accuracy which is expected as the model is trained on a very small number of data. So, it may not be able to learn enough features specific to the air leakage sound which will aid the binary classification.

MODEL 2 (M2) This model is based on a CNN RNN architecture.

Now I ran my second model on all the three datasets, starting with the UrbanSound8K dataset.

DATASET: UrbanSound8K

Dataset split	Number of wav files
Training	6985
Testing	1747

Here, the number of audio files used for training is 6985 and for testing is 1747. The model is iterated for 100 epochs.

METRICS

Train Epoch: 100

METRICS	Training	Testing
Accuracy	0.8993	0.92
Loss	0.4883	0.38

We observe a pretty good accuracy for both training as well as testing. This model performs better than the previous one.

DATASET: Set of Factory data (A2IoT) – 01

Dataset split	Number of wav files
Training	461
Testing	116

Now ran the second model (M2) on the set of data from company.

METRICS

Train Epochs: 100

METRICS	Training	Testing
Accuracy	0.9210	0.9050
Loss	0.29	0.39

Confusion Matrix

		TRUE LABEL	
		Air-Leakage	No Air-Leakage
PREDICTED LABEL	Air-Leakage	48 TP	8 FP
	No Air-Leakage	4 FN	56 TN

Accuracy = 0.9050

Precision = 0.8572

Recall = 0.9230

F1-Score = 0.8889

This architecture performs significantly better than the earlier one. It has pretty good accuracy for both training and testing.

This is reasonable and is expected as CNN RNN architectures gives us very good accuracy.

DATASET: Set of Factory data (A2IoT) – 02

TEST 606

Confusion Matrix

		TRUE LABEL	
		Air-Leakage	No Air-Leakage
PREDICTED LABEL	Air-Leakage	233 TP	61 FP
	No Air-Leakage	56 FN	256 TN

Accuracy = 0.8069

Precision = 0.7925

Recall = 0.8062

F1-Score = 0.7992

When tested on entire new set of data, accuracy drops down but it performed better than the previous model (M1).

From the above table, we see that the number of false positives is quite low as compared to previous model which is a very crucial especially in positive-negative type binary classification.

MODEL 3 (M3)

This is the most complicated, large and model.

It is a combination of all the three architectures namely CNN, RNN & LSTM.

DATASET: UrbanSound8K

Dataset split	Number of wav files
Training	6985
Testing	1747

As earlier, the above table gives the number of audio files used for training and testing the model.

METRICS

Train Epoch: 100

METRICS	Training	Validation
Accuracy	0.898982	0.7426
Loss	0.296555	0.9319

Here, we see reasonably good accuracy for training but the validation accuracy is very low.

This might be due to less number of data.

DATASET: Factory Dataset (A2IoT) – 01

Dataset split	Number of wav files
Training	461
Testing	116

Confusion Matrix

PREDICTED LABEL	TRUE LABEL: Air-Leakage	TRUE LABEL: No Air-Leakage
Air-Leakage	53 TP	9 FP
No Air-Leakage	7 FN	47 TN

Accuracy = 0.8612

Precision = 0.8548

Recall = 0.8833

F1-Score = 0.8688

Here, we see that model M3 has performed significantly good on test dataset.

DATASET: Factory Dataset (A2IoT) – 02

TEST 606

Confusion Matrix

PREDICTED LABEL	TRUE LABEL: Air-Leakage	TRUE LABEL: No Air-Leakage
Air-Leakage	239 TP	52 FP
No Air-Leakage	49 FN	266 TN

Accuracy = 0.8403

Precision = 0.8209

Recall = 0.8467

F1-Score = 0.8336

On testing this model on entire new set of data, the performance drops down as expected. But we see that this model gives the best results as compared to the other two models.

All the metrics used to measure performance is the highest among all the models. is highest among all the three models.

CHAPTER 7: DISCUSSION

We can infer from the above results that a combination of CNN, RNN and LSTM performs better in classifying sounds.

Further, the accuracy can be increased if we tune the model for an optimum number of RNN & LSTM cells.

However, it is observed from the above results that after training the models, when testing is done on an entire new set of data (A2IoT-02), the performance drops down significantly.

This trend is observed in all of the three models. So we can conclude that when models are tested on completely new data, the performance seems to drop down.

This indicates that we need to train our model on more number of data to learn the parameters that will increase the performance.

The table below shows the accuracy and F1 Scores of the three models when tested on company data i.e. A2IoT -02 (section 5.2)

	M1	M2	M3
Accuracy	0.7980	0.8069	0.8403
F1 Score	0.7876	0.7992	0.8336

In binary classification, there are a lot of metrics that we can use to compare the performance. Accuracy is the simplest one. However, accuracy does not take into account the subtleties in class imbalances or uneven distribution of data.

As an alternative, we use F1 Score. It is a better measure when we want to balance the unevenness of data. F1 Score is the geometric mean of the Precision and Recall. Hence, it combines both and seeks to restore the balance between them.

As seen from the table above, the F1-Scores of M3 is highest.

Hence we can conclude that the Model 3 (M3), based on CNN + RNN + LSTM architecture, performs better than the other two architectures.

We can infer that the best result will come from a model with a combination of CNN, RNN and LSTM neural network architecture.

Once the neural networks are trained on a given dataset, they can be extended to perform other similar tasks such as speech recognition and speaker recognition.

Summary of Important Points of Results

The usual neural network architecture (RNN) is good in classifying sounds. However, from this project, we can conclude that a combination of CNN, RNN and LSTM performs better than architecture having only one type of neural network.

- All the metrics like accuracy, precision, F1-Score etc., improved when the set of company data is tested on model 3 (M3).

- We observed a significant drop in performance for all three models when tested on an entirely new set of data.

- One of the reasons for this drop can be a very small number of data, very little variation among different audio files, which significantly affected the performance.

- Further, the set of factory data is not even, which can lead the algorithm to get biased and rely on some specific features for classification to give poor results when tested on an entirely new set of data.

- A little increment in performance can be observed from optimising the number of perceptrons used in the neural network architecture.

- Other alternatives apart from CNN, RNN and LSTM may be tried like GRU, Transformers etc.

References

- Online courses on Coursera
  - Python – Programming for Everybody (Getting started with Python).
  - Deep Learning Specialisation which includes five courses.
  - Tensorflow- Deeplearning.AI Professional Certificate

- Documentation of
  - Python- https://docs.python.org/release/3.9.1/
  - TensorFlow- https://www.tensorflow.org/guide
  - Librosa - https://librosa.org/doc/latest/index.html
  - PyTorch - https://pytorch.org/docs/stable/index.html

- Tutorials given on the official website of TensorFlow
  https://www.tensorflow.org/tutorials

- Deep Learning for Audio Signal Processing
  https://arxiv.org/abs/1905.00078

- Various blogs and articles on Medium, Towards Data Science and Deep Learning

- Convolutional Recurrent Neural Networks for Urban Sound Classification using Raw Waveforms
  DOI: 10.23919/EUSIPCO.2018.8553247

- Urban Sound Classification using Long Short-Term Memory Neural Network
  DOI: 10.15439/2019F185

- Machine learning in acoustics: Theory and applications
  The Journal of the Acoustical Society of America 146, 3590 (2019);
  https://doi.org/10.1121/1.5133944

- UrbanSound8K Dataset

  This Dataset is compiled by Justin Salamon, Christopher Jacoby and Juan Pablo Bello. The UrbanSound and UrbanSound8K datasets are offered free of charge for non-commercial use only under the terms of the Creative Commons Attribution Non-commercial License (by-NC). A seed grant-supported creating the datasets by NYU's Center for Urban Science and Progress (CUSP).