Noise Supression using Deep Learning

A project report submitted in partial fulfillment of the requirements for B.Tech. Project

Bachelor of Technology Project

by

Kartik Mishra (2019IMG-67)

विश्वजीवनामृतं ज्ञानम्

ABV INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT GWALIOR-474 010

2022

CANDIDATES DECLARATION

I hereby certify that the work, which is being presented in the report, entitled **Nosie Suppression using Deep Learning**, in partial fulfillment of the requirement for the award of the Degree of **Bachelor of Technology** and submitted to the institution is an authentic record of my own work carried out during the period *June 2022* to *September 2022* under the supervision of **Dr. Gaurav Kaushal**. I have also cited the references about the text(s)/figure(s)/table(s) from where they have been taken.

Date: September 19th,2022 Signatures of the Candidate

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date: Signatures of the Research Supervisor

ABSTRACT

The capacity of a platform to block out background noise is vital in the era of the explosion of online video conferencing and virtual communication, as it gives it a competitive edge. In order to give the finest audio quality possible, platforms like Google Meet regularly do noise suppression.

By adding a second speech sound that is intended to cancel the first speech signal, noise suppression is a technique for eliminating unwanted speech sound. By using a secondary source to introduce anti-noise wave, this is accomplished. These auxiliary sources are connected via an electrical device that employs a noise-canceling algorithm. In this method, it is predicted that the undesirable signal will cancel out the desired signal at every instant of time. The most effective way to eliminate noise is via an adaptive filter.

The process of reducing noise from a signal is known as noise reduction. This technique can lessen background noise that is consistent, such as fan noise and hiss. We must choose a section of the waveform that solely contains the unwanted noise. When the noise is very loud, fluctuates, and music isn't much louder than the noise threshold or when the noise frequency is very comparable to that of music or speech, the method is not appropriate. The idea behind noise reduction techniques is to identify in-line noise during silences and subtract it from the speech that follows.

The spectral domain is utilized by current speech enhancement algorithms, or they may take advantage of a more advanced feature. They mainly concentrate on a limited set of noise conditions and then use first-order statistics. Deep networks are being employed more commonly to address these issues due to their ability to learn complex functions from large example sets. In this paper, we propose using generative adversarial networks to improve speech.

ACKNOWLEDGEMENTS

I am profoundly obligated to **Dr. Gaurav Kaushal** and am obliged for providing me the independence of working and trying different approaches, and experimenting with my thoughts. I want to accept this moment to offer my gratitude and significant thanks to him for their academic direction and enthusiasm for my thesis work and consistent help combined with certainty boosting and inspiring meetings that demonstrated productivity and were instrumental in fostering confidence and trust inside me. The supporting and blooming of the current work is primarily because of his essential direction, proposals, ingenious judgment, productive analysis, and an eye for excellence. My mentor consistently addressed many of our questions with grinning benevolence and enormous persistence, never letting us feel that we are learners by continually loaning an ear to our perspectives, acknowledging and improving them, and giving me a free hand in our task. It's as it were because of his staggering interest and accommodating disposition, the current work has accomplished the stage it has.

Last but not least, I want to thank my Institute, seniors, and my family, who continuously motivated and supported me during this challenging time of pandemic and assisted me in every possible way to help complete my thesis work and well-properly in time. I am grateful to have you all in my life

Kartik Mishra

TABLE OF CONTENTS

ABSTRACT **ii**

LIST OF TABLES **v**

LIST OF FIGURES **vi**

1 Introduction **ix**

1.1 Motivation ix
1.2 Objectives x
1.3 Research work flow x

2 Literature review **xi**

2.1 Background xi
2.2 Advantages and Disadvantages of some Noise reduction method xii
2.3 Key related research and Analysis xii
2.4 Research gaps xiii

3 Methodology **xv**

3.1 Proposed hypothesis xv
3.2 Algorithms and Terminologies xv
3.2.1 Convolutional Neural Network xv
3.2.2 Generative adversarial networks xvi
3.2.3 Optimization Algorithm xvi
3.2.4 Layers xvi
3.3 Functioninig xvi
3.3.1 Data loading and visualisation xvii
3.3.2 tf.data.dataset creation for pipelininG xviii
3.3.3 Creating the Model xix

4 Result and Conclusion **xxii**

4.1 Intermediate Outputs xxii
4.2 Handling the Variable Input Shape xxiii

4.3 Rapidness of results xxiv
4.4 Improvement and Enhancement of the TFLite Model xxiv
4.5 Model Inference with TFLite xxiv
4.6 Overall conclusion xxvi
4.7 Future Work xxvi

REFERENCES **xxvi**

LIST OF TABLES

2.1 Benefits and Drawbacks of Some Noise Reduction Techniques(4) . . . xii
2.2 Comparing the noisy signal and the Wiener- and SEGAN-enhanced signals.(19) xiv

LIST OF FIGURES

1.1 Creating the Model x

3.1 Loading Libraries xvii
3.2 Loading and Visualizing Data xvii
3.3 Batch size of 12000 xviii
3.4 Clean Audio xviii
3.5 Noisy Audio xix
3.6 Creating a tf.data.Dataset for pipelining xix
3.7 Creating the Model xx

4.1 Visible Reduction Of Noise xxii
4.2 Overlapped Frames xxiii
4.3 xxiii
4.4 xxiv
4.5 xxiv
4.6 xxv
4.7 xxv
4.8 Plot for TFLite Model xxv

ABBREVIATIONS

SEGAN	Speech Enhancement Generative Adversarial Network
CNN	Convolutional neural networks
ADAM	Adaptive Moment Estimation
GAN	Generative Adversarial Network
RMSProp	Root Mean Square Propagation
SGD	Stochastic Gradient Descent
RNN	Recurrent Neural Network

CHAPTER 1

Introduction

Speech enrichment aims to make speech which has been damaged by additive noise more comprehensible and of greater quality (13). Its primary uses focus around improving mobile conversations in loud settings. Although, we alsodiscover major applications for cochlear implants and hearing aids, where improving the signal prior to amplification can significantly lessen discomfort and improve understanding (24). In voice recognition and speaker identification systems, additionally effective as a preprocessing stage is speech augmentation (7)(1)(15). Spectral subtraction (5), Wiener filtering (12), statistical model-based techniques (8), and subspace algorithms (14)(9) are traditional speech enhancement techniques. Neural networks have also been used to enhance speech since the 1980s(20)(18). The denoising auto-encoder architecture (22) has recently gained popularity.

1.1 Motivation

Humans have always been at war with noise. If it is loud enough, it can interfere with conversations, hinder focus, and even harm the auditory system. The conventional approach to noise suppression relies on energy absorption: by carefully positioning damping material adjacent to the noise source, a significant portion of the sound energy can be converted to heat, thereby lowering the sound level. The walls of dance and studio rooms are covered with a soft, porous, multi-layered material that can dampen most of the noise's intensity. Since they merely passively confine or absorb noise, these techniques are known as "passive noise suppression". Since most foam can minimize noise by up to 35 dB, the outcome might be optimal. However, this kind of noise suppression's drawback is its size. If any valuable sound information is present, it is also filtered out along with all noises in the absorptive frequency range. Additionally more expensive and requiring more difficult instrument installation is passive noise suppression.

1.2 Objectives

There are many methods to address the task of noise suppression. This includes Residual Network, GANs etc. Regardless of the method, noise suppression suffers from two main issues.

- Managing audio loops of varying lengths
- Lag is generated by slow processing.

A hearing aid is a portable, low-power device that can capture, process, and give back acoustic data in real-time. Its many technologies, including manufacturing technology and modifying techniques, will affect how hearing is perceived. The noise reduction technique is the main technological factor that affects the effect on hearing. The effectiveness of noise reduction has a significant impact on speech understanding and even the physical and mental health of those with impaired or damaged hearing. Therefore, research into how well hearing aids reduce noise has a very significant impact.

1.3 Research work flow

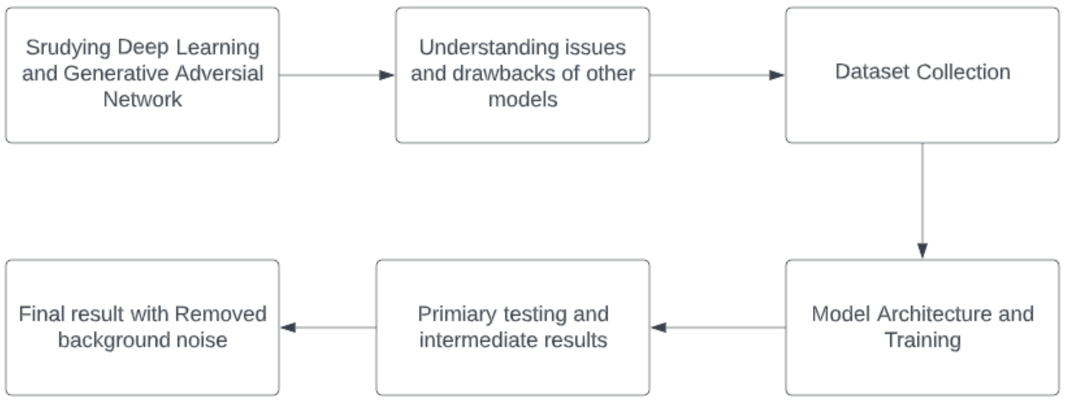

Srudying Deep Learning and Generative Adversial Network
Understanding issues and drawbacks of other models
Dataset Collection
Model Architecture and Training
Primiary testing and intermediate results
Final result with Removed background noise

Figure 1.1: Creating the Model

CHAPTER 2

Literature review

2.1 Background

Background noises come in a wide variety of standard forms. These consist of: Impulse noises, such as pops, that are often loud yet short-lived,Impulse is made up of brief power spikes with a roughly flat frequency response across the targeted spectral range. A voltage increase of at least 12 dB over the r.m.s. Noise for a duration of at most 12 ms is regarded as sufficient. Many things, including the switching of relays in electro-mechanical telephone exchanges, can produce this kind of noise. The human ear can tolerate these "clicks and pops" even if they are irritating. On data or digital circuits, impulsive noise can, however, result in a variety of serious error rate issues.

When a noise is carried throughout a broad frequency range, it is called broadband noise, including buzzing. The PA's generated broadband noise has the potential to reduce the receiver's sensitivity. Better PA design, which may be expensive and difficult to implement, as well as filtering and transmit-receive isolation, are two ways to reduce this form of noise.

Narrow band noises, which are characterised by constant signal noises. A random method is bandpass or narrowband random process if its power spectral density is nonzero solely in an exceedingly tiny neighborhood of some high frequency fc settled signals: outlined by its Fourier remodel Random processes: defined by its power spectral density.

Electrical noises, usually originating from the recording equipment, Electrical hum is related with a frequency-based unwanted electrical signal typically not up to two hundred kilohertz.It should be highlighted that while noise disturbances do not behave like waveforms, they are entirely distinct from harmonic distortions and transients. Electronic devices, control circuits, arcing apparatus, and switch power supplies are examples of noise generators. It's been noted that the results thanks to noises are often created worse once grounding is introduced to the electric power distribution system.

Unusual sounds include everything from talking to traffic to loud noises like rumbling thunder and rain. All of these noises have the potential to ruin your conference or recording. An essential first step in improving your sound is learning to deal with ambient noises and creating a plan of action for them.

2.2 Advantages and Disadvantages of some Noise reduction method

Noise Reduction method	**Advantages**	**Disadvantages**
Linear filters such as Gaussian, wiener filters (17)	Simple conceptually	They deteriorate the edges and fine features of the image. Consequently, a denoised image would be hazy.
Anisotropic nonlinear diffusion (17)	It preserves edges to a greater extent and lessens noise in flat areas.	Adjusting different parameters, such as counting the iterations is a challenging task. It degrades the fine structure, lowering the image's resolution
Wavelet-based methods (6)	These techniques aim to isolate signal from noise without compromising the signal's quality.	Wavelet coefficients might be biased. These methods corrode fine details, especially in highly noisy images
Deep Learning Model	Effective at Producing High-Quality Results	Massive Data Requirement

Table 2.1: Benefits and Drawbacks of Some Noise Reduction Techniques(4)

2.3 Key related research and Analysis

Due to the risk rule's strong theoretical foundation, it has consistently performed the best in these experiments (18), It also appears in the risk rule's status as the most widely employed to fundamental wavelet threshold. In addressing Five separate approaches were examined in the case of white Gaussian noise, and the results showed Clearly, bionic risk threshold surpasses its wavelet cousin in terms of effectiveness. A more rational view of the human has indeed impacted beliefs more about cause of this occurrence. Relative to the wavelet transform, the bionic wavelet transform utilizes the cochlear mechanism. The bionic method, nevertheless, displays sharp production decline if handled via Gaussian noise with the a 10 dB SNR level. The threshold techniques that were utilized were designed as they don't fit with the nonlinear properties of the bionic wavelet transform because they were designed originally for the discrete wavelet transform., that might be plausible explanation of this outcome. In the event that a strong speech signal and a weak noise signal are present, limit struggles to make a distinction between the target signal and the noise.

WaveNet, a deep generative model of audio data that operates directly at the waveform level (2), has been presented in this paper. While representing the long-range temporal dependencies in audio signals, WaveNets, which are auto regressive, combine causal filters with dilated convolutions to allow their receptive fields to grow with depth exponentially. It is showed how WaveNets can be both locally and globally conditioned on other inputs, like speaker identity . When used for Text To Speech, WaveNets produced samples that functioned better in terms of subjective naturalness than the top TTS systems presently available.

A DNN-based framework for speech enhancement is recommended in this paper(23). To comprehend the complicated structure of the mapping function between noisy and clean speech data, that is one of the many DNN implementations, a substantial training set is needed. It was discovered that including more acoustic context information improves system performance and decreases discontinuities in the enhanced speech. Furthermore, adaptation to unfamiliar noise environments can be done successfully via multi-condition training with a range of noise sources. As a conclusion, the proposed DNN framework has the capability to efficiently deal with non-stationary noises in real life.

2.4 Research gaps

Since this hearing model had been using using the Morlet wavelet to obtain the optimal temporal and resolution of frequency compromise, bionic wavelet transform was originally derived from this method . Although employing 22 analytical scales yields in the same level of computation as Ephraim Malah filtering, this method can only be performed by numerical integration because the effective discrete wavelet transform is not supported by the Morlet wavelet.

Despite Wave Net can make high speech waveforms, it is too extremely slow, and the front-end errors may have a consequence on the synthesis effect.

Without any non-linearities or authorizations, a neural network's single linear layer is nearly identical to a linear model, it should be emphasized up ahead. We are speaking regarding DNN here, which has many layers and activation functions. Second, multiple layers and nonlinearities create an error space that really is nonconvex and often quite complex, resulting in a significant number of local minimums to which the deep neural network's training can converge. To locate a point in the error space where the error is sufficiently small for the model to be effective, a lot of hyperparameters should be set.Numerous hyperparameters, with numbers ranging from 10-40 or 50, are addressed by Gaussian processes are used in bayesian optimization, though this does not assure greater outcomes. Their training is extremely slow, and when their hyperparameters were adjusted on top of that, it gets even worse. In comparison, the linear model would train significantly more fast. A substantial cost tradeoff is now established. Weights exist in a trained linear model that is clear and provide the data scientist with helpful information on how various features assist to the task at hand.

Metric	**Noisy**	**Wiener**	**SEGAN**
PESQ	1.97	2.22	2.16
CSIG	3.35	3.23	3.48
CBAK	2.44	2.68	2.94
COVL	2.63	2.67	2.80
SSNR	1.68	5.07	7.73

Table 2.2: Comparing the noisy signal and the Wiener- and SEGAN-enhanced signals.(19)

CHAPTER 3

Methodology

3.1 Proposed hypothesis

There are various methods to tackle the task of noise suppression. This might include the residual networks,GANs and many more. Regardless of the approach, noise suppression struggles from two major problems, handling audio sequences of various lengths,Lag produced through slow processing. We'll explore some basic solutions to these problems in this thesis, such as how to deal with these problems

3.2 Algorithms and Terminologies

3.2.1 Convolutional Neural Network

- **CNN:** It is a deep learning neural network built to handle structure arrays of information, like graphics, is termed as a convolutional neural network, or CNN. CNN is particularly good in gathering information about objects like lines, gradients, circles, or even eyes and faces. Convolutional neural networks are exceptionally efficient for computer vision due to this feature. CNN doesn't really need any preprocessing and can be directly applied to an underdone image. A intake neural network with up to 20 layer upon layer is a CNN.The convolutional layer, a certain type of layer, is just what lends convolutional neural networks its effectiveness. Each of the many convolutional layers which compose CNN is able to recognize more intricate forms. Such layers are placed on top of each other.

3.2.2 Generative adversarial networks

- **GAN:** It is a recent advancement in the field of deep learning generative modelling(21) . In the realm of computer vision, GANs have had reasonable success in generating realistic images and summarising effectively to pixel-wise, complex (high-dimensional) distributions(11)(10)(16). In our opinion, it is the first method to generate speech signals using the adversarial framework, as GANs have not yet been used to any tasks requiring speech synthesis or augmentation.

3.2.3 Optimization Algorithm

- **Adam:** Adam can be regarded as an RMSprop and stochastic gradient descent (SGD) with momentum combination(3). It adjusts the learning rate using squared gradients, comparable to RMSprop, and utilizes momentum by using the gradient's moving average instead of gradient itself, similarly to SGD with momentum.

3.2.4 Layers

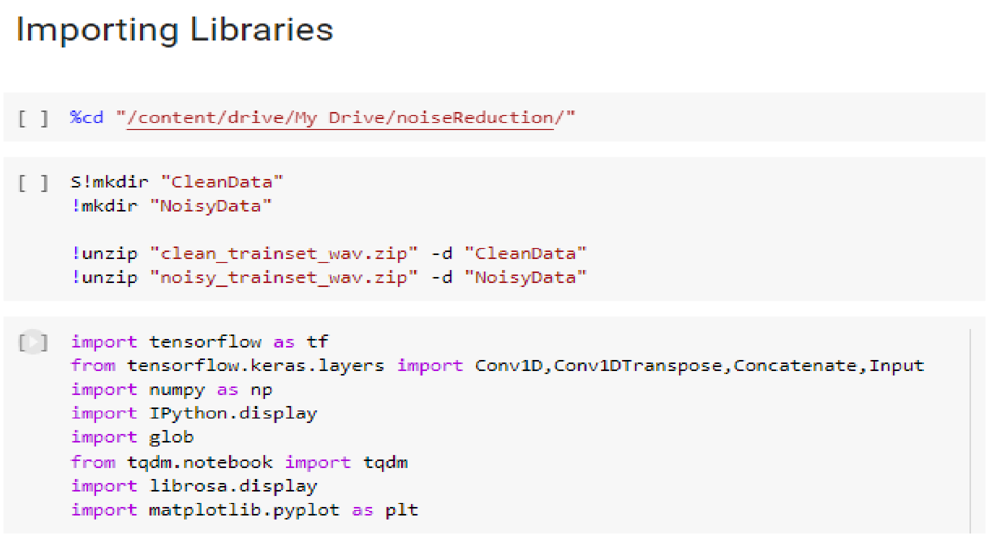
- **Concatenation Layer:** This layer aids in the concatenation of many inserts. A list of tensors with the same shape is acceptable as input and outputs the concatenations of all inputs along the chosen axis. This layer was utilised to concatenate diverse inputs in our model.

- **Transpose Layer:** The necessity generally for transposed convolutions results from need a transformation to be used that ends up heading the other way from that of a pretty normal convolution, that is from something that transforms from something with the shape of a convolution's output to something with the shape of its input while preserving a connectivity pattern that is appropriate for the convolution in question.

- **1D convolution :** In order to create a tensor of outputs, layer generates a convolutional kernels that is combined over a single spatial (or temporal) dimension with the layer input. A bias vector is generated and appended to the outputs if use bias is set to True. Finally and not least, activating is also applied to the outputs if it is not None.

3.3 Functioninig

Our libraries will be loaded first. We will utilize Tensorflow.



Importing Libraries

```
[ ] %cd "/content/drive/My Drive/noiseReduction/"
```

```
[ ] S!mkdir "CleanData"
    !mkdir "NoisyData"

    !unzip "clean_trainset_wav.zip" -d "CleanData"
    !unzip "noisy_trainset_wav.zip" -d "NoisyData"
```

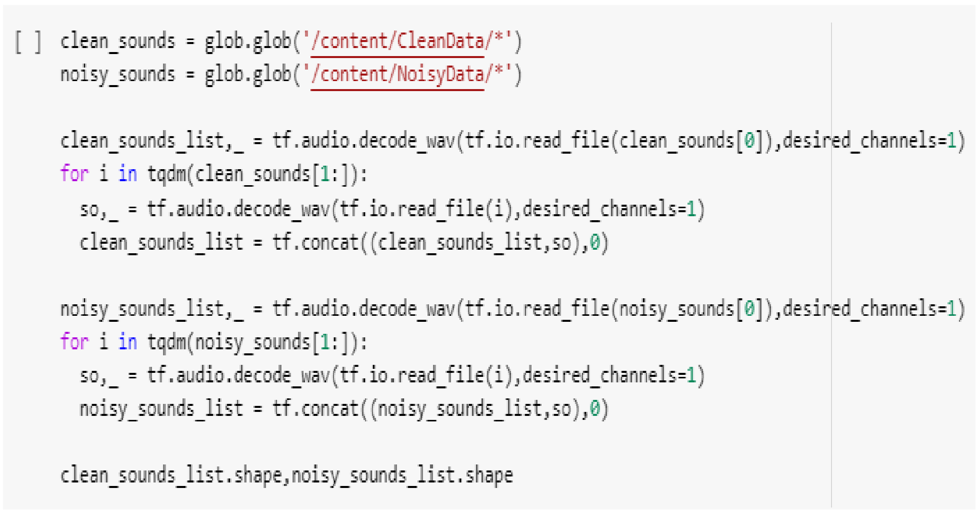
```
[ ] import tensorflow as tf
    from tensorflow.keras.layers import Conv1D,Conv1DTranspose,Concatenate,Input
    import numpy as np
    import IPython.display
    import glob
    from tqdm.notebook import tqdm
    import librosa.display
    import matplotlib.pyplot as plt
```

Figure 3.1: Loading Libraries

A range of different Clean and Noisy audio samples comprise up the data we'll be using. The documentation is gathered from the University of Edinburgh.

3.3.1 Data loading and visualisation

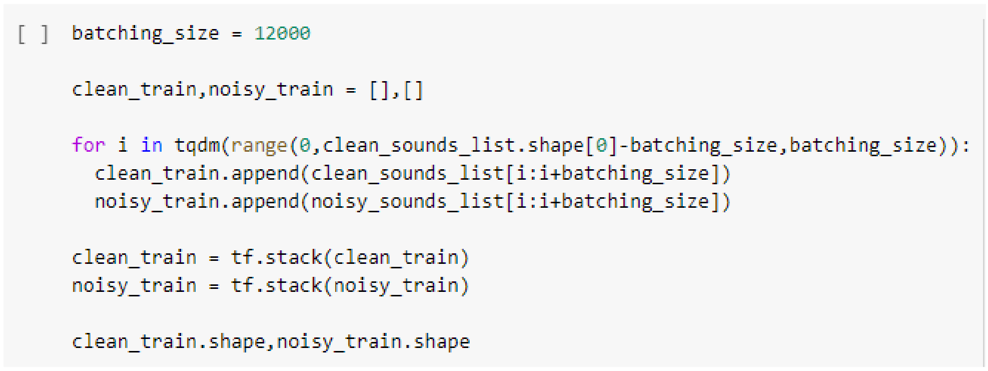
To load our data, we'll utilize the tf.audio module within Tensorflow. Due to tensorflow's utilisation of the GPU, we utilize "tf . audio" and "tf . io.read file" produced in 51 percent more rapid loading times than "librosa.load".



```
[ ] clean_sounds = glob.glob('/content/CleanData/*')
    noisy_sounds = glob.glob('/content/NoisyData/*')

    clean_sounds_list,_ = tf.audio.decode_wav(tf.io.read_file(clean_sounds[0]),desired_channels=1)
    for i in tqdm(clean_sounds[1:]):
      so,_ = tf.audio.decode_wav(tf.io.read_file(i),desired_channels=1)
      clean_sounds_list = tf.concat((clean_sounds_list,so),0)

    noisy_sounds_list,_ = tf.audio.decode_wav(tf.io.read_file(noisy_sounds[0]),desired_channels=1)
    for i in tqdm(noisy_sounds[1:]):
      so,_ = tf.audio.decode_wav(tf.io.read_file(i),desired_channels=1)
      noisy_sounds_list = tf.concat((noisy_sounds_list,so),0)

    clean_sounds_list.shape,noisy_sounds_list.shape
```

Figure 3.2: Loading and Visualizing Data

Next, we utilize tf.audio.decode wav() to load each of our individual audio files, and afterwards we concatenate it and make 2 tensors labeled noise sound list and clean sound list. The tqdm loading bar serves as a visual representation of this process, and took approximately 3–4 minutes to complete.



```
[ ]  batching_size = 12000

     clean_train,noisy_train = [],[]

     for i in tqdm(range(0,clean_sounds_list.shape[0]-batching_size,batching_size)):
       clean_train.append(clean_sounds_list[i:i+batching_size])
       noisy_train.append(noisy_sounds_list[i:i+batching_size])

     clean_train = tf.stack(clean_train)
     noisy_train = tf.stack(noisy_train)

     clean_train.shape,noisy_train.shape
```

Figure 3.3: Batch size of 12000

Once loading is finished, we should split the single massive audio waveform in equal proportions. The core objective to modify this model into a "tflite" model, something that does not currently support inputs with varied lengths, while doing so is not necessary. We selected 12000 as arbitrary batching size choice.

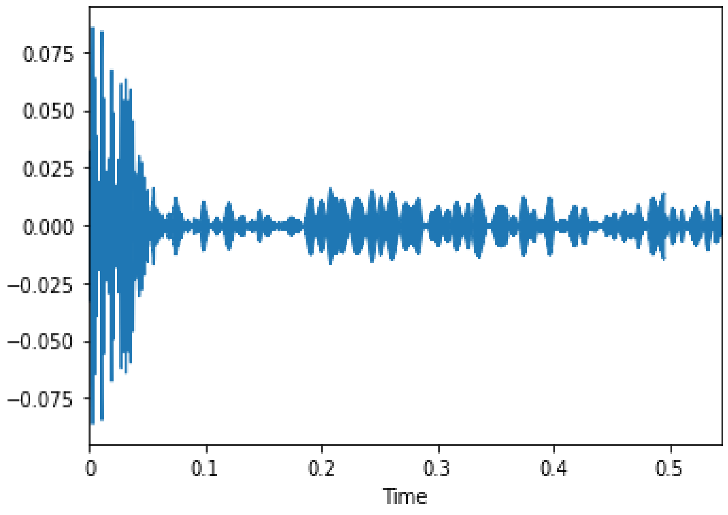


Figure 3.4: Clean Audio

For visualization techniques, we implement librosa's display component, which essentially displays the information using matplotlib on the backend. Whenever the information are plotted as in the instance above, the noise is clearly visible. Dishwashing noises, passing cars, and onlookers may all contribute to the loudness.

3.3.2 tf.data.dataset creation for pipelininG

Then we build Get dataset which is a very basic assistance function, to construct a "tf.data". For testing, 5000 samples are selected, andwe pick a different 40.000to be trained

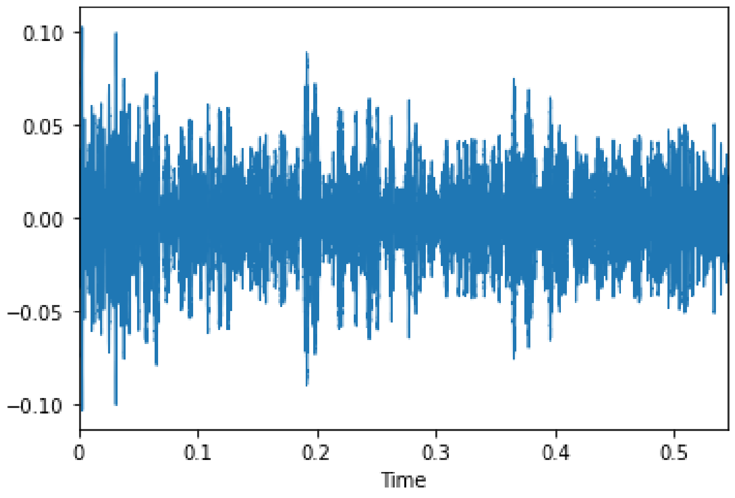
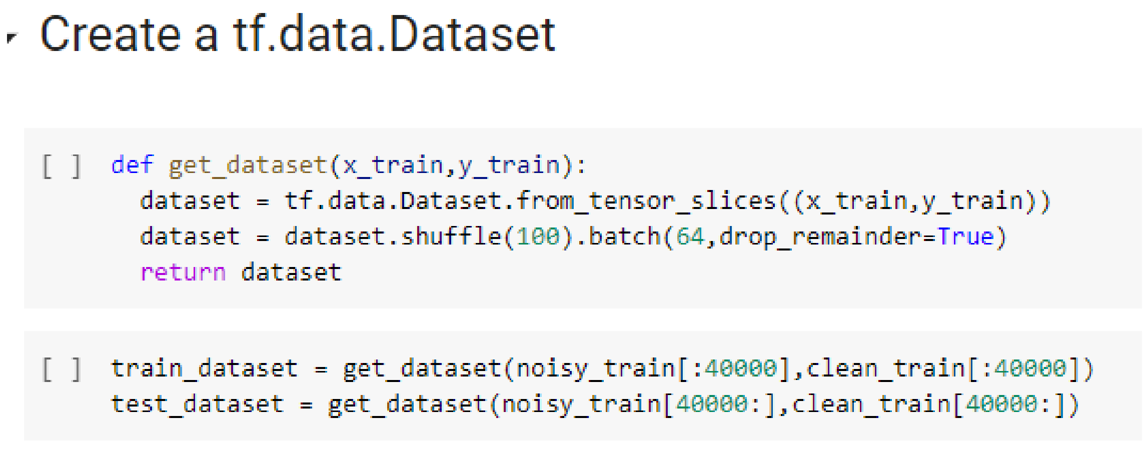


Figure 3.5: Noisy Audio



Create a tf.data.Dataset

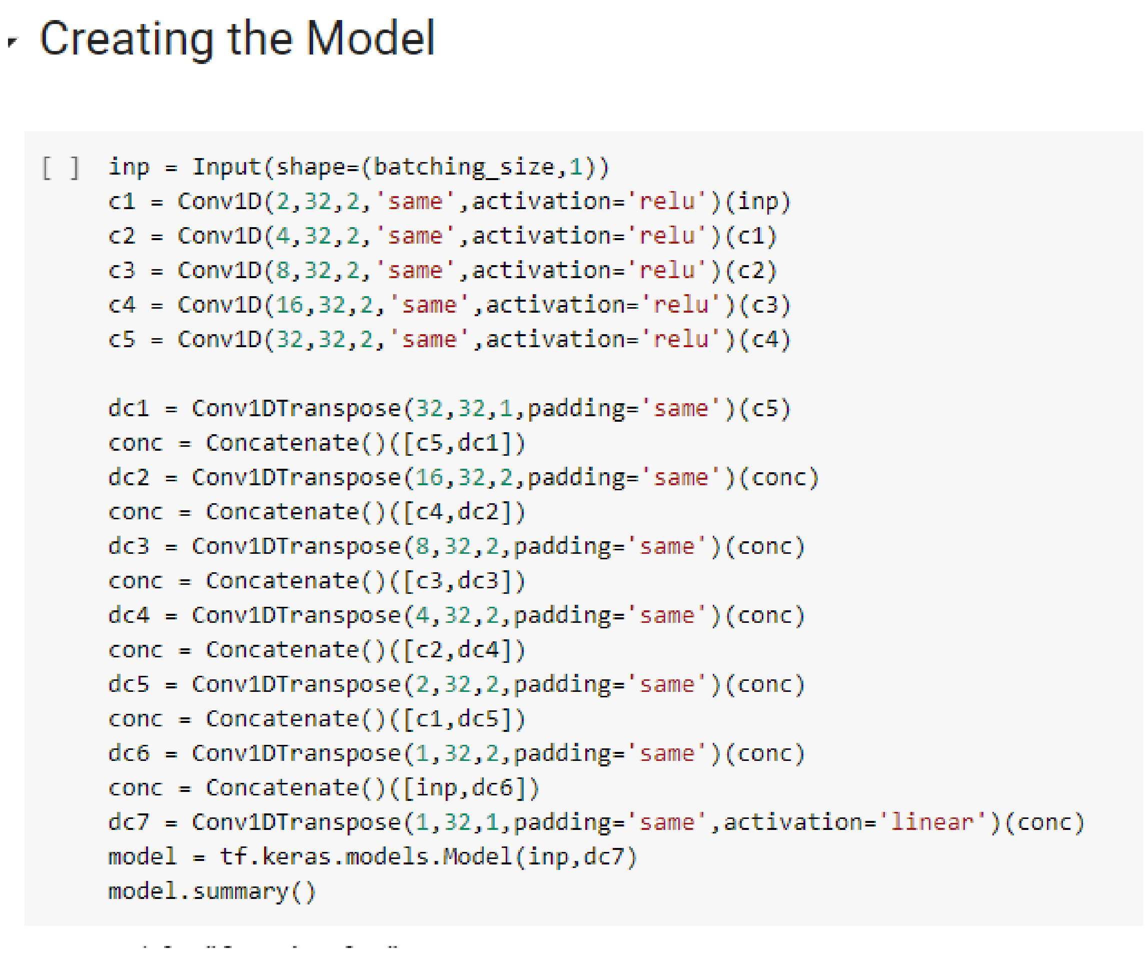
```
def get_dataset(x_train,y_train):
  dataset = tf.data.Dataset.from_tensor_slices((x_train,y_train))
  dataset = dataset.shuffle(100).batch(64,drop_remainder=True)
  return dataset
```

```
train_dataset = get_dataset(noisy_train[:40000],clean_train[:40000])
test_dataset = get_dataset(noisy_train[40000:],clean_train[40000:])
```

Figure 3.6: Creating a tf.data.Dataset for pipelining

3.3.3 Creating the Model

The model architecture's code is as shown in:



Creating the Model

```
[ ] inp = Input(shape=(batching_size,1))
    c1 = Conv1D(2,32,2,'same',activation='relu')(inp)
    c2 = Conv1D(4,32,2,'same',activation='relu')(c1)
    c3 = Conv1D(8,32,2,'same',activation='relu')(c2)
    c4 = Conv1D(16,32,2,'same',activation='relu')(c3)
    c5 = Conv1D(32,32,2,'same',activation='relu')(c4)

    dc1 = Conv1DTranspose(32,32,1,padding='same')(c5)
    conc = Concatenate()([c5,dc1])
    dc2 = Conv1DTranspose(16,32,2,padding='same')(conc)
    conc = Concatenate()([c4,dc2])
    dc3 = Conv1DTranspose(8,32,2,padding='same')(conc)
    conc = Concatenate()([c3,dc3])
    dc4 = Conv1DTranspose(4,32,2,padding='same')(conc)
    conc = Concatenate()([c2,dc4])
    dc5 = Conv1DTranspose(2,32,2,padding='same')(conc)
    conc = Concatenate()([c1,dc5])
    dc6 = Conv1DTranspose(1,32,2,padding='same')(conc)
    conc = Concatenate()([inp,dc6])
    dc7 = Conv1DTranspose(1,32,1,padding='same',activation='linear')(conc)
    model = tf.keras.models.Model(inp,dc7)
    model.summary()
```

Figure 3.7: Creating the Model

It's simply a convolutional model. Designing a tonne of filters is the goal in order to decrease the background noise. In order to put the original audio sample in this, we offer residual connections. This model's idea was inspired by Santiago Pascual's SEGAN network implementation. The notion is based on the fact that a convolutional network by itself can handle a range of input forms with ease, resulting in more flexibility. The model should focus on the close temporal correlations throughout due to the convolutional nature of the data. The convolutions and the de-convolutions of the model can be divided into two components (or upsampling layers).After N levels, the depthwise convolutional layers behave like an auto-encoder and generate a compact representation of the input. To generate a a more accurate illustration of the noisy insert, the de-convolution carry out exactly the opposite deliberate methods. The skip connections give the deconvolution layers the contextual information at each stage, enhancing the overall outcomes.

The Mean Absolute Loss method was used to build the model.

$$Abs(Y_{true} + Y_{pred})$$

Selecting an optimizer IS difficult since Adam, SGD, and RMSprop all were reasonably competitive. Just because Adam is slightly more robust, We ultimately decided to move forward with it. we were able to attain a good learning rate of 0.002 after some hyperparameter tweaking.

CHAPTER 4

Result and Conclusion

4.1 Intermediate Outputs

- Train loss - 0.0117
- Test Loss - 0.0117

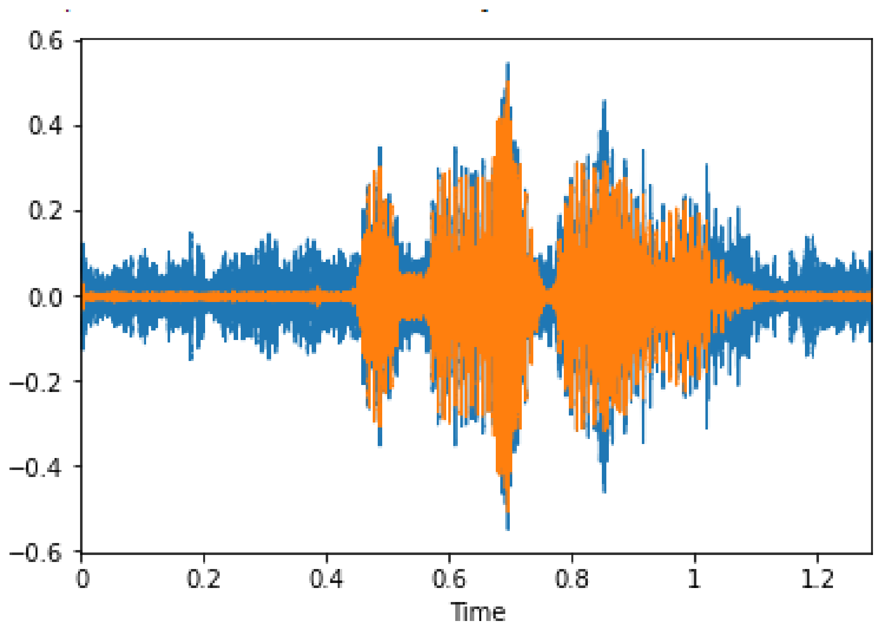


Figure 4.1: Visible Reduction Of Noise

Although The outcomes are really satisfying, it is not yet finished. It is also required manage the inference process for inputs of different sizes. Then we carry out that.

4.2 Handling the Variable Input Shape

A extremely precise input form which rely on the batching-size is utilized for the model. A uncomplicated method will work really well to handle various shape inputs.

Up till the (n-1)th split, we run our model through each split. Think more about situation where your audio array does have the shape and batching size is 12000. (37500,). Waveform was split into minimum [(37500/12000) = 3 splits] in this case. Array's parts will all have shape (1500,). To address this problem, we sample from the array's back end for a second frame. similar to this

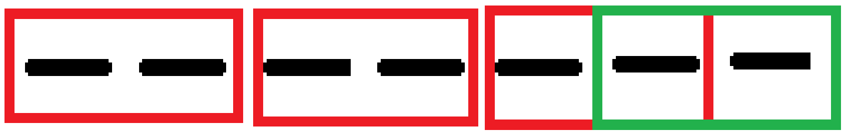
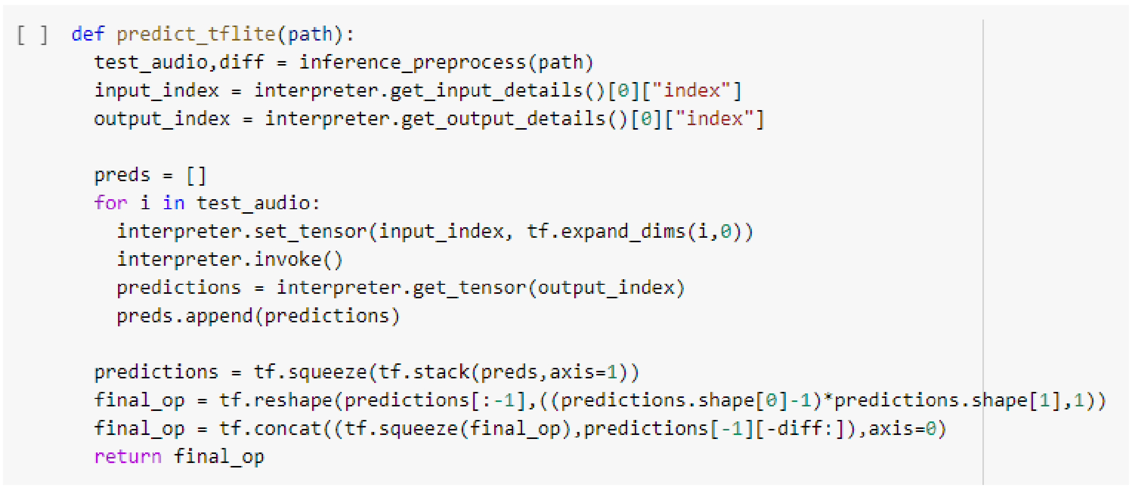


Figure 4.2: Overlapped Frames

- In order to provide distinct predictions, we first test the model on each of the 4 splits.
- The output forecasts' first three frames are taken as-is, and the final frame is chopped down to simply the remaining part.

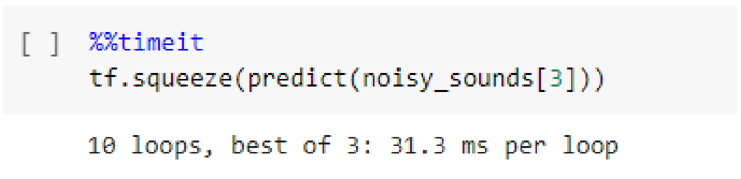
Clarity will be increased at this point with a few clip of code.



```
[ ]  def predict_tflite(path):
       test_audio,diff = inference_preprocess(path)
       input_index = interpreter.get_input_details()[0]["index"]
       output_index = interpreter.get_output_details()[0]["index"]

       preds = []
       for i in test_audio:
         interpreter.set_tensor(input_index, tf.expand_dims(i,0))
         interpreter.invoke()
         predictions = interpreter.get_tensor(output_index)
         preds.append(predictions)

       predictions = tf.squeeze(tf.stack(preds,axis=1))
       final_op = tf.reshape(predictions[:-1],((predictions.shape[0]-1)*predictions.shape[1],1))
       final_op = tf.concat((tf.squeeze(final_op),predictions[-1][-diff:]),axis=0)
       return final_op
```

Figure 4.3:

4.3 Rapidness of results



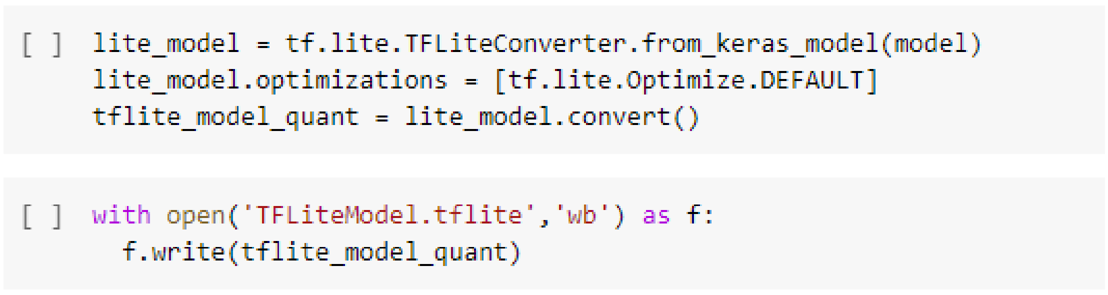
```
[ ] %%timeit
    tf.squeeze(predict(noisy_sounds[3]))

    10 loops, best of 3: 31.3 ms per loop
```

Figure 4.4:

A tensor with varying lengths can be given to this model, which yields even quicker outcomes, if the model's input form is defined [None,1].It is needed to quantize the model to accommodate many endpoints.

4.4 Improvement and Enhancement of the TFLite Model

The TFLiteConverter() has a fairly simple interface. The modified version of the model is saved for storage as a binary file for subsequent use after already being given to Keras along with an optimization strategy.



```
[ ] lite_model = tf.lite.TFLiteConverter.from_keras_model(model)
    lite_model.optimizations = [tf.lite.Optimize.DEFAULT]
    tflite_model_quant = lite_model.convert()
```

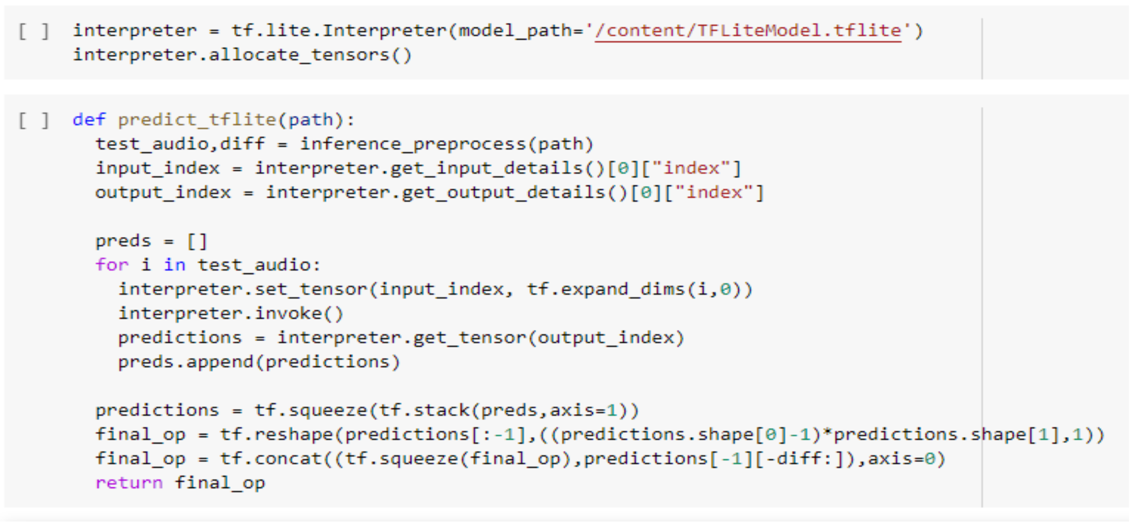
```
[ ] with open('TFLiteModel.tflite','wb') as f:
      f.write(tflite_model_quant)
```

Figure 4.5:

4.5 Model Inference with TFLite

The error-checking is identical to the "Keras model", except We employ a python's for loop to go across all divides in order to not find any data on TFLite model batching. The code which follows shows how to build the Interpreter, assign tensors to it, and afterwards call it to generate our results.

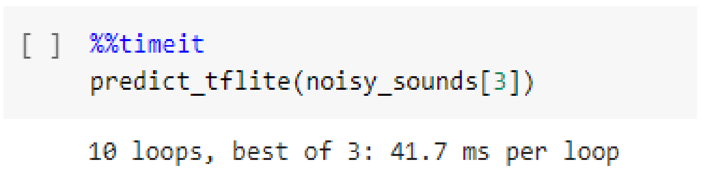
The problem now is, under what ways is this better to the Keras model, That has a complicated answer. The overall inference time was affected since we were unable to process all batches in one go, but the TFLite Model is quicker than the Keras Model by itself.



```
[ ]  interpreter = tf.lite.Interpreter(model_path='/content/TFLiteModel.tflite')
     interpreter.allocate_tensors()
```

```
[ ]  def predict_tflite(path):
       test_audio,diff = inference_preprocess(path)
       input_index = interpreter.get_input_details()[0]["index"]
       output_index = interpreter.get_output_details()[0]["index"]

       preds = []
       for i in test_audio:
         interpreter.set_tensor(input_index, tf.expand_dims(i,0))
         interpreter.invoke()
         predictions = interpreter.get_tensor(output_index)
         preds.append(predictions)

       predictions = tf.squeeze(tf.stack(preds,axis=1))
       final_op = tf.reshape(predictions[:-1],((predictions.shape[0]-1)*predictions.shape[1],1))
       final_op = tf.concat((tf.squeeze(final_op),predictions[-1][-diff:]),axis=0)
       return final_op
```

Figure 4.6:



```
[ ]  %%timeit
     predict_tflite(noisy_sounds[3])

     10 loops, best of 3: 41.7 ms per loop
```

Figure 4.7:

Cross-platform deployment is the TFLite format's biggest advantage, In addition to being significantly easier to adapt than the Keras model is that the latest model is also remarkably small—only 346 kB.

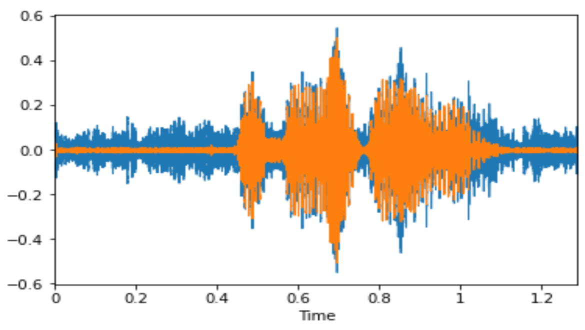


Figure 4.8: Plot for TFLite Model

4.6 Overall conclusion

In this work, a generative adversarial framework with an end-to-end voice improvement mechanism has been implemented. The suggested speech enhancement GAN's (SEGAN's) primary benefits encompass:

- It provides a fast augmentation process.No recursive operation such in RNNs exists as causality is not required.
- It functions smoothly with the raw audio. Since no artificially generated attributes are extracted, no explicit assumptions about the raw data are generated as a result.
- It incorporates what it gains knowledge from various speakers and noise patterns. into one parametrization that it utilizes for all speakers and noise types. This makes the method simple to comprehend and relevant across those dimensions.

As we have seen the proposed model is much easier to adapt than the keras model and has a compact size of 346 kB

4.7 Future Work

Future research should concentrate on strengthening existing convolutional structures and integrating perceptual weightings during adversarial training to reduce any high frequency artefacts that the current model might generate. To compare SEGAN with competitor strategies, more research is required.

REFERENCES

[1] T. M. O'Neil O. Vinyals P. Nguyen A. L. Maas, Q. V. Le and A. Y. Ng. Recurrent neural networks for noise reduction in robust asr. page 22–25., 2012.

[2] H. Zen K. Simonyan O. Vinyals A. Graves N. Kalchbrenner A. Senior A. van den Oord, S. Dieleman and K. Kavukcuoglu. Wavenet: A generative model for raw audio. 2016.

[3] Mohammed Alom. Adam optimization algorithm. 2021.

[4] M.A. Balafar. Review of noise reducing algorithms for brain mri images. 2011.

[5] M. Berouti, R. Schwartz, and J. Makhoul. Enhancement of speech corrupted by acoustic noise. In *ICASSP '79. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages 208–211, 1979.

[6] J.S. Sahambi C.S. Anand. Mri denoising using bilateral filter in redundant wavelet domain. *IEEE Region 10 Conference TENCON*, 2008.

[7] J. Droppo J. Wu Y. Gong D. Yu, L. Deng and A. Acero. A minimum-mean-square-error noise reduction algorithm on melfrequency cepstra for robust speech recognition. *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, page 4041–4044., 2008.

[8] Y. Ephraim. Statistical-model-based speech enhancement systems. *Proceedings of the IEEE*, 80(10):1526–1555, 1992.

[9] Y. Ephraim and H.L. Van Trees. A signal subspace approach for speech enhancement. *IEEE Transactions on Speech and Audio Processing*, 3(4):251–266, 1995.

[10] M. Mirza B. Xu D. WardeFarley S. Ozair A. Courville I. Goodfellow, J. Pouget-Abadie and Y. Bengio. Generative adversarial nets," in advances in neural information processing systems (nips). page 2672–2680, 2014.

[11] K. W´ojcicki K. Paliwal and B. Shannon. The importance of phase in speech enhancement. *Speech Communication*, 53:465 – 494, 2011.

[12] J. Lim and A. Oppenheim. All-pole modeling of degraded speech. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 26:197–210, 2005.

[13] P. C. Loizou. *Speech Enhancement: Theory and Practice, 2nd ed. Boca Raton*. USA: CRC Press, Inc., 2 edition, 2013.

[14] S. Bakamidis M. Dendrinos and G. Carayannis. Speech enhancement from noise: A regenerative approach. *Speech Communication*, 10:45–57, 2010.

[15] J. Ortega-Garcia and J. Gonzalez-Rodriguez. Overview of speech enhancement techniques for automatic speaker recognition. *n Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, 2:929–932, 2005.

[16] T. Zhou P. Isola, J.-Y. Zhu and A. A. Efros. Image-toimage translation with conditional adversarial networks. 2016.

[17] J. Malik P. Perona. Scale space and edge detection using anisotropic diffusion. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.

[18] S. Parveen and P. Green. Speech enhancement with missing data techniques using recurrent neural networks. *Proc. ofthe IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, page 733–736, 2004.

[19] Joan Serrà Santiago Pascual, Antonio Bonafonte. Segan: Speech enhancement generative adversarial network. 2017.

[20] S. Tamura and A. Waibel. Noise reduction using connectionist models. *Proc. ofthe IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, page 553–556., 1988.

[21] D. Wang and J. Lim. The unimportance of phase in speech enhancement. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 30:679–681, 2005.

[22] S. Matsuda X. Lu, Y. Tsao and C. Hori. Speech enhancement based on deep denoising autoencoder. *Proc. of INTERSPEECH*, page 436–440, 2013.

[23] Yong Xu, Jun Du, Li-Rong Dai, and Chin-Hui Lee. A regression approach to speech enhancement based on deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(1):7–19, 2015.

[24] L.-P. Yang and Q.-J. Fu. Spectral subtraction-based speech enhancement for cochlear implant patients in background noise. *The Journal ofthe Acoustical Society ofAmerica*, 117:1001–1004, 2005.