Department of Electrical, Computer, and Software Engineering

Part IV Research Project

Project Compendium Report

Project Number: 58

The reconstruction of

speech/voice with the use

of noise-cancelling

algorithms and machine

learning

Edward Chan

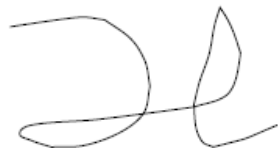Timothy Aguana Cabrera

Catherine Watson

Yusuke Hioka

15/10/2023

Declaration of Originality

This report is my own unaided work and was not copied from nor written in collaboration with any other person.

Name: Edward Chan

Name: Timothy Aguana Cabrera

Abstract: This contribution addresses enhancing speech quality in noisy environments with low signal-to-noise ratios (SNR). The approach consists of two key components. First, it leverages codebooks derived from Gaussian mixture models (GMMs) and a Wasserstein Generative Adversarial Network (WGAN) to generate time-varying spectral candidates that closely resemble noise spectra. While the estimation of noise spectra is accurate, the estimation of speech spectra needs to improve, resulting in distorted and unintelligible Wiener-filtered mixtures.

Low SNR leads to significantly quieter speech, posing a problem in audio signal processing. The paper proposes an iterative learning approach that combines two machine learning algorithms for speech enhancement and classification in noisy mixtures. The first algorithm employs a WGAN for end-to-end learning to generate noise spectra that can be used to create enhanced speech. Simultaneously, considering SNR levels, a convolutional neural network (CNN) uses the spectrogram representation of the estimated clean speech to classify between clean and noisy speech.

The primary objective of this paper is to explore whether improvement to the performance of the codebook-based approach can be used with SNR levels to indicate improved speech quality and intelligibility. The proposed method utilises computational matrices, including perceptual evaluation of speech (PESQ), short-time objective intelligibility (STOI), and signal distortion ratio (SDR).

1. Introduction

The low SNR problem is explored through a journey using a cycling approach through two machine learning algorithms to generate an estimated speech and classify if that speech has enhanced the speech or requires a more aggressive transformation. Working with the more generalised single-channel configuration inferior to the multi-channel configuration, our approach attempts to improve traditional statistical estimators at relatively high SNR while effectively enhancing low SNR mixtures.

This paper investigates existing statistical and neural network techniques used to enhance speech in section 2. Section 3 discusses the research to find the ideal approach to improving speech quality and intelligibility. Section 4 contains the overall method and the journey to enhance the speech. Lastly, section 5 highlights the challenges, reflection, and future considerations.

2. Literature Review

2.1. Low SNR Speech Enhancement

Speech enhancement enhances speech intelligibility or quality through noise reduction algorithms. Current speech enhancement methods have shown outstanding performance at enhancing the quality and intelligibility of noisy speech signals with high signal-to-noise ratios (SNR) above 0dB. These methods consist of statistical estimation or neural networks to predictively estimate the noise spectra to be removed from speech [1, 2]. In comparison, low SNR speech enhancement remains challenging for both methods due to the high presence of noise and the sparsity of speech information. Although utilising a Multi-channel configuration has shown improved results through spatio-temporal information to recover speech spectra [3], many standard communication devices do not have these configurations. Therefore, the extensive development and usage of statistical estimators and neural networks have been applied to the challenging task of low SNR single-channel speech enhancement.

2.2. Statistical Estimators

Traditionally, speech enhancement is done through statistical estimators. These statistical models use certain assumptions, such as statistical stationarity, to derive an average spectral shape of noise to be removed from a noisy mixture [2, 4]. The assumption of statistical stationarity means that for a given random process such as noise, the probability of an occurrence is assumed to be the same throughout time and therefore, an average value can be inferred. However, in situations where noise is non-stationary, the speech enhancement performance of these algorithms degrades as the wrong estimates of noise spectra will lead to a reduction of speech intelligibility due to over-subtraction, residual noise or filter-induced distortion that is especially noticeable with low SNR mixtures [5].

2.3. Neural Networks

As an alternative, neural networks have been recently applied to speech enhancement. Currently, successful methods of speech enhancement neural networks consist of non-causal filtering techniques

such as spectrogram-derived time-frequency masks or direct time-domain synthesis from an input mixture with improved results compared to statistical estimators at all SNRs [2, 6]. However, in contrast to statistical estimators, the filter characteristics of neural networks cannot be mathematically modelled, and their performance is mainly attributed to the choice of architecture and hyperparameters [7].

2.4. Generative Adversarial Networks

Generative adversarial networks (GANs) are a recent advancement in generative models. GANs can generate realistic data through adversarial training between the generator and discriminator of two neural networks. The generator will learn to map a random input to a generated distribution that closely resembles real data by its parameters by classifying its outputs by the discriminator [8]. Typical applications of GANs consist of realistic random or controlled image creation or augmentation from a low-dimensional vector that is randomly generated or feature encoded [9, 10]. Regarding speech enhancement, GANs employ the same non-realisable filter approaches as other neural networks, such as 'end-to-end' enhancement of time domain mixtures [2, 11].

2.5. Convolutional Neural Network for Classification

Image classification is the process of 1) extracting features from an image and 2) categorising the image into its most likely class. Some examples of classification using spectrograms can be to classify honey bee sounds [12], the difference between speech and music [13] and human activities [14]. Convolutional neural networks (CNN) are the most popular technique used for image classification. It works well with collecting information on different images as inputs and output possibilities for all classes depending on the image features.

2.6. Datasets

Existing literature shows datasets that train learning models use noisy and clean speech at different SNR levels [15, 16]. SNR levels range from 10 dB to -10dB for the training phase. One single-channel

speech source dataset is the TIMIT database [17], which includes 630 speakers with approximately 5 hours of recording. VCTK dataset [18] contains 109 speakers with approximately 400 sentences that were 1 to 10 seconds long.

Some of the databases used for noise are as follows: the CHiME [19], the Diverse Environment Multi-channel Acoustic Noise Database (DEMAND) [20], the NOISEX-92 database [21] and the AURORA dataset [22]. The attributes of the noise datasets are shown in Table 1 [23].

Table 1: Noise datasets attributes [23]

Dataset	Total time (h)	Frequency (kHz)	# of microphones	Avg. SNR
CHiME 1	70	16 & 48	2	low
CHiME2	78	16	2	low
CHiME3	48	16	6	low
DEMAND	~ 23	16 & 48	16	high
NOISEX-92	-	-	-	-
AURORA-2	33	8-16	8-16	low

3. Research

3.1. Codebook Driven Wiener Filtering

Codebooks are a collection of spectral candidates used to accurately estimate noise spectra through a superposition of weighted codebook entries. For good speech enhancement performance, it is often the case that a codebook will have many entries to have a high chance of accurately estimating the spectra of noise through the formation of an under-determined system at the cost of computational time [24, 25, 33]. A study [26] concerning low SNR speech enhancement showed that effective low SNR speech enhancement was possible by forming an overdetermined system of only 15 spectral candidates. Further development [16] by Manamperi et al. has shown improvements over the baseline design through a multi-stage filtering approach utilising different codebooks per iteration.

3.2. Gaussian Mixture Modelling of Codebooks

This project used Gaussian mixture models (GMMs) to derive the codebook. To derive the codebook, the mean spectral power of each frequency bin is calculated through the expectation maximisation (EM) algorithm. As no assumptions were made about the dataset distribution statistics, sieving was used. Sieving [30] is a process that iteratively guesses many different parameters in order to assist the model in achieving the global maximum log-likelihood, as the EM algorithm cannot escape local maxima. The assumption used in this project was that the model that achieves the global maxima is the model closest to the true distribution.

3.3. Generative Adversarial Network

Although GANs have been frequently used for realistic data creation, the training process of these networks is very unstable [8]. A GAN aims to transform a latent input vector Z with a distribution into a generated distribution, Pg, that matches the learnt distribution, Pd, by the discriminator. In order to measure the deviation of Pg from Pd, a metric known as the Jensen Shannon divergence (JSD) is used, the gradients of which are used to train both the generator and discriminator [27]. The

GAN aims to minimise the JSD between Pg and Pd. However, a key limitation of the JSD that causes the training instability of GANs is the inability to quantify the divergence where Pg does not overlap with Pd [26]. In these non-overlapping situations, the JSD is a constant value resulting in 0 or small gradients, which impedes the neural network's learning process [27]. Therefore, as an alternative, this project uses a Wasserstein GAN (WGAN) to stabilise the network's training process.

In contrast to the JSD metric, WGANs use an alternative metric of Wasserstein distance. The Wasserstein distance measures the distance between the two distributions, Pg and Pd and allows the WGAN to learn by minimising this distance [28]. This enables the WGAN to be less sensitive to the chosen architecture and hyperparameters as the relationship between these and Pg and Pd are no longer constrained, unlike that of the default GAN [27, 28].

3.4. Convolutional Neural Network

The proposed model in Fig. 1 includes convolution layers to extract features from the spectrogram and pooling layers to downsize the image so features can be detected at various resolutions [29]. Using both the convolution and pooling layers for feature extraction, the next step is to use a flattening layer that converts the 2D features from the convolutional layers into a 1D vector to connect the convolutional layers to a fully connected layer. The first dense layer has 1024 neurons with ReLU activation, which tries to understand the complex relationship in the patterns. The final dense layer has eight neurons with softmax activation, which outputs probabilities for each SNR class.
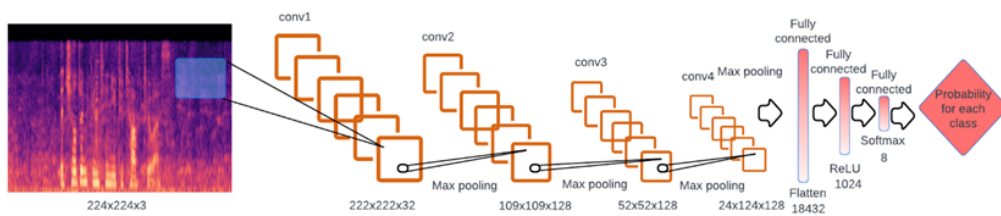

Fig. 1. Overview of the CNN classification model with each layer [29].
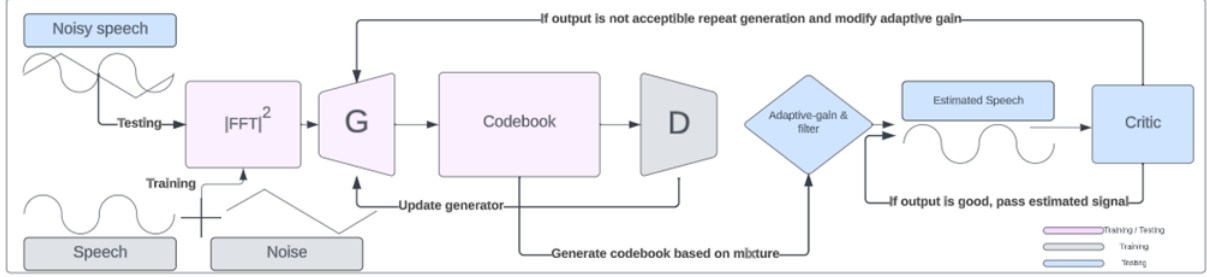
4. Implementation

4.1. Method

Fig. 2. High-level flow chart of the proposed design.

In Fig. 2, our high-level diagram shows the training and testing requirements. Starting the training of the overall system, we train the GAN model using a range of noises to quantify the energy associated with each frequency component of the noise. This comprehensive analysis of frequency components will allow us to fine-tune the GAN architecture and training parameters to ensure that the generated data accurately captures a more accurate collection of noise power spectral densities called the codebook. As mentioned, the generator predicts the codebook when training the GAN model. At the same time, the discriminator aims to distinguish between a less accurate and more accurate codebook, which updates the generator to enhance its prediction accuracy. Testing the system includes a noisy mixture of speech at any SNR level through the GAN model to generate an estimation of the codebook. Based on the noisy mixture, the adaptive gain and filter adjust the frequency amplitude and response based on the input signal. A speech estimation is then generated and critiqued using a CNN classification model to distinguish between different SNR levels for clean or noisy speech. If the critic determines that the generated mixture does not meet the criteria for clean speech, the system initiates a regenerative process. During this process, a modified adaptive gain adjustment control is applied to the mixture signal and continuously iterated until one of the two conditions is met. These conditions are if the system recognises clean speech when the power of the speech signal substantially

exceeds the background noise power signal, resulting in high SNR or when the system reaches a predefined threshold.

4.2. Gaussian Mixture Model (GMM) Derived Codebooks

In order to generate a GMM-derived codebook, the audio files from the DEMAND [20] and VCTK [18] datasets were low-pass filtered using a fifth-order infinite impulse response Butterworth low-pass filter. For an audio sample, the long-term average of the power spectral densities (PSD) calculated per analysis frame is taken and logged as an entry to the dataset from which the GMM will derive a distribution. The spectral power of each frequency bin of the power spectral densities in the dataset is used as an input to a sieving algorithm [30] to derive the assumed true distribution of the spectral power of each frequency bin. The mean of each resulting univariate GMM is used to assemble the codebook entries for speech and noise spectra. An example of the resulting GMM can be seen in Figure 3.
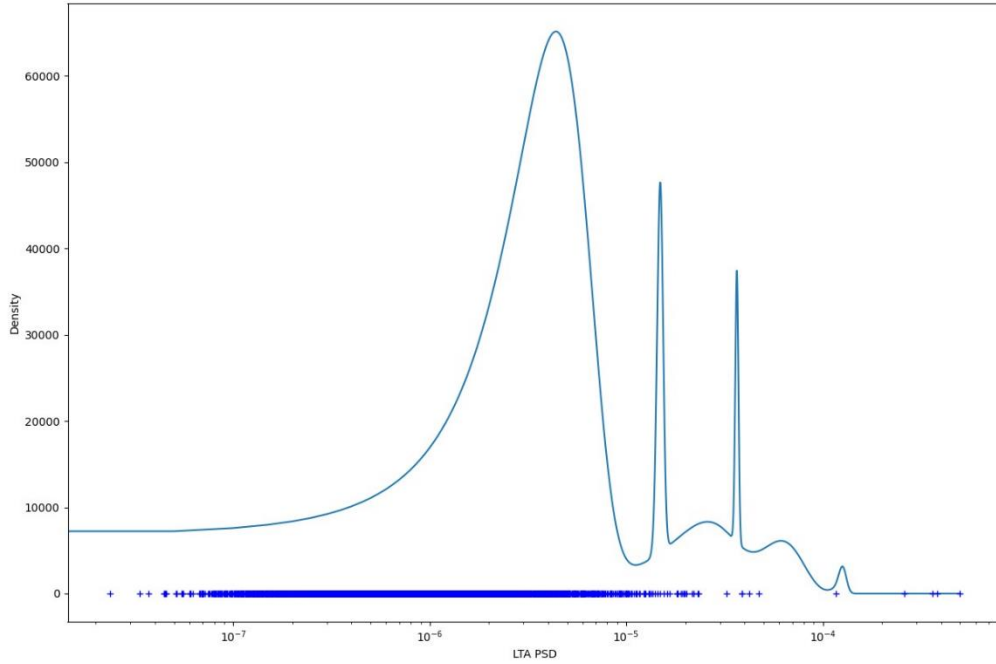
Figure 3: GMM distribution of a frequency bin from the noise codebook. Blue crosses indicate the data points, while the blue line is the resulting probability density function (PDF) of the GMM.

Before training the GMM of speech, Silero-VAD [31] was used to remove the effects of no presence in each audio file. The voice activity detector (VAD) concatenates sections with speech presence and removes the silence in between. This was done to represent the distribution of speech's spectral power accurately. An interesting observation was the convergence failure of certain frequency bins when the GMM was trained with the PSDs of speech. This resulted in no available number (NaN) to be returned. On inspection of the histogram distribution of the non-converging speech frequency bins, it was seen that the resulting laplacian or negative exponential distribution could not be approximated by Gaussians, as seen in Figure 4.
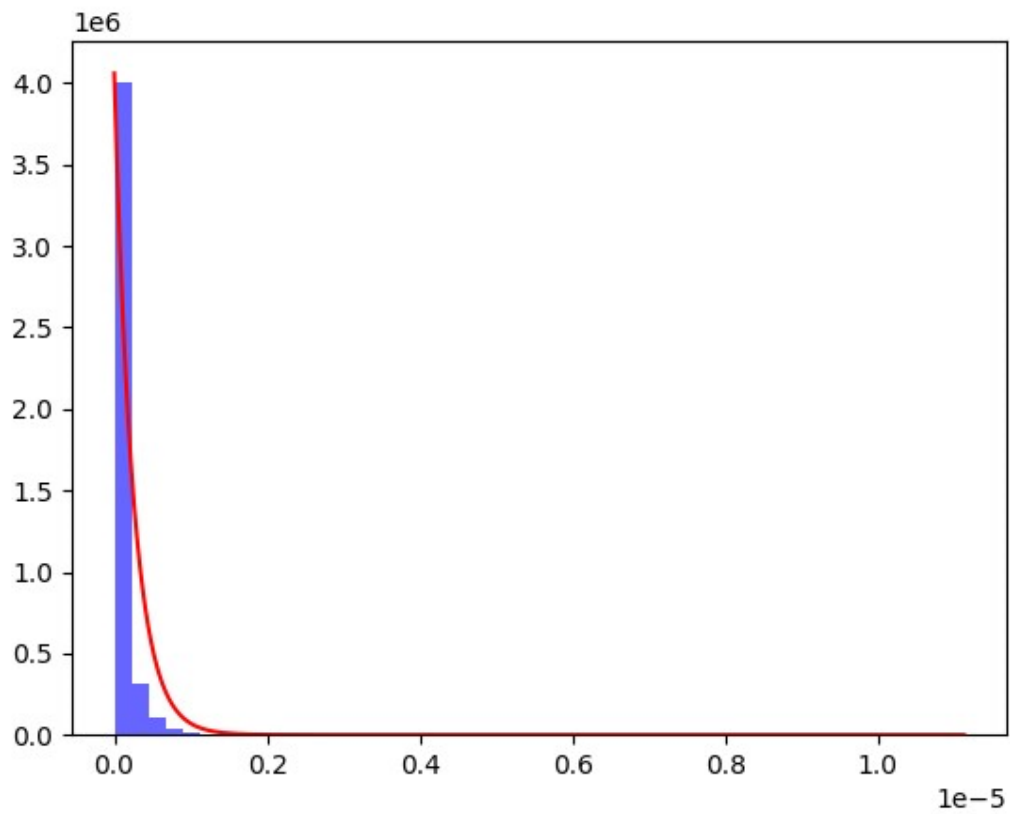
Figure 4: Histogram plot with a fitted negative exponential curve of a frequency bin of speech. The resulting GMM closely resembled a laplacian curve for any converging frequency bin of speech, as seen in Figure 5. This was an intuitive result as the speech PDF closely follows a laplacian or negative exponential.
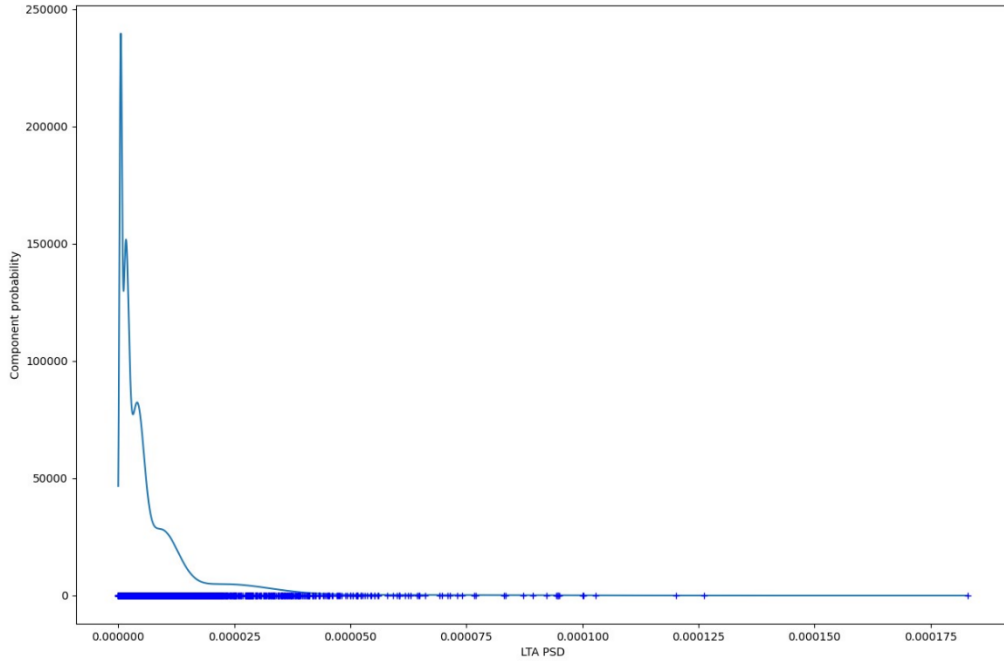
Figure 5: GMM distribution of a frequency bin from the speech codebook. Blue crosses indicate the data points, while the blue line is the resulting probability density function (PDF) of the GMM.

4.3. Implementation of the Overdetermined System

On implementing the aforementioned overdetermined system using the codebooks derived from the GMMs, it can be seen that the resulting estimates are poor and resulted in distortions, as seen in Figure 6. Note that, compared to the study [26] conducted by Chehresa et al, the short-time Fourier transform parameters of this project deviate and are as follows: NFFT = 1024, Samples per windowed segment = 512, Noverlap/Hop size = 384/128, Window = 'Hann'.
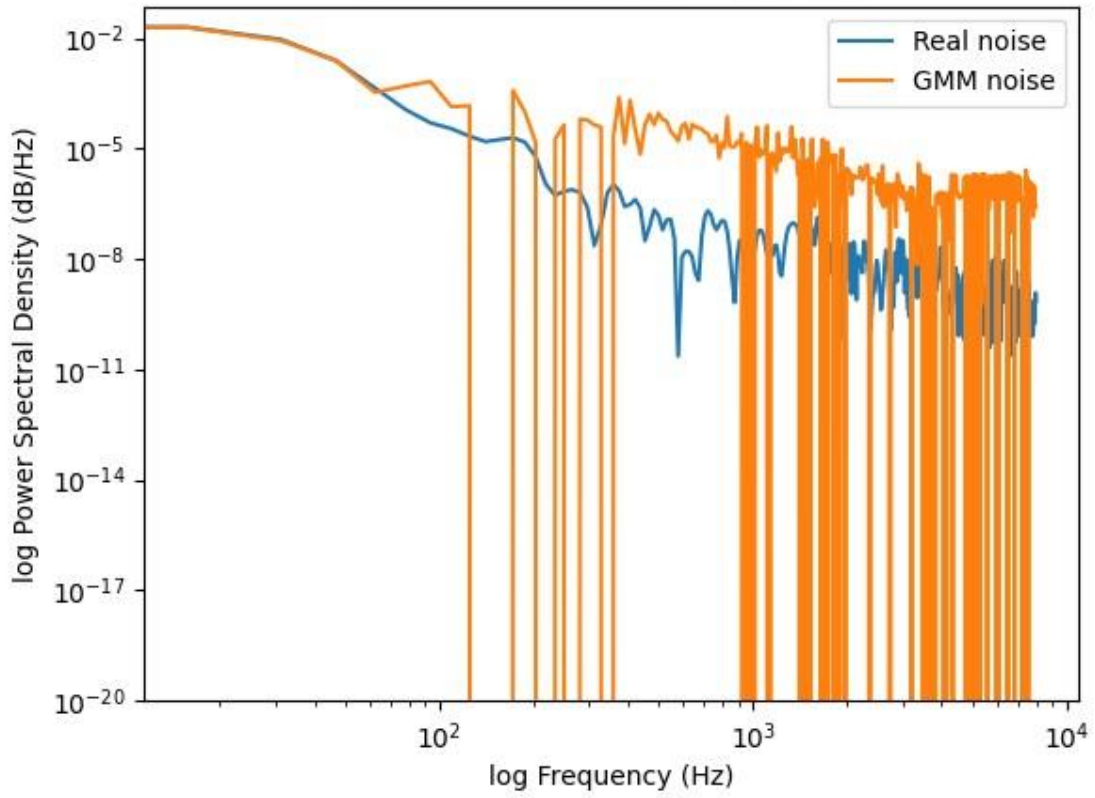
Figure 6: Noise PSD estimate of the overdetermined system.

It was suspected that the resulting codebook could have been suboptimal. However, non-tracking

and distorted estimates were observed from integrating the proposed system with a developed
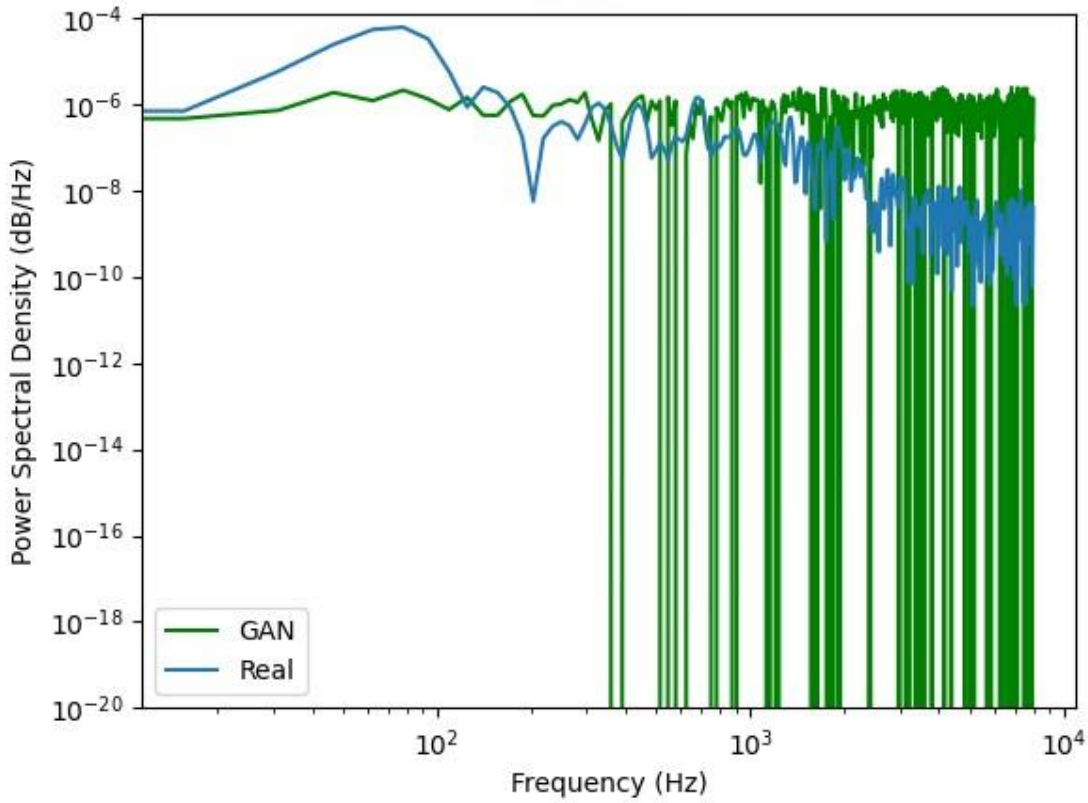
WGAN, as seen in Figure 7.

Figure 7: Integration of the overdetermined system with a GAN and the resulting estimate of the

noise PSD.

4.4. Reformulation and Derived Models

To reformulate, instead of calculating the weights applied to each spectrum, the desired

functionality was then to dynamically adjust the gain applied to each frequency bin of each

codebook entry.

The overdetermined model [26] and the derivation of the estimates are as follows:

$A \equiv$ the concatenated observation vector of both speech and noise spectral candidates. This study

keeps the same number of noise and speech spectral candidates. This vector contains 15 spectral

candidates, 9 of which are potential noise spectra, and the remaining are speech spectra. The

dimensions of this vector are (NFFT,15).

$$M(f,t) \equiv Periodogram\ of\ the\ mixture\ at\ the\ current\ frame.$$

$$x \equiv scalar\ vector\ of\ dimension\ (15,1).$$

$$Ax = M(f,t) \tag{1}$$

$$x = M(f,t)A^{\dagger} \tag{2}$$

$$M(f,t) = \sum_{K=1}^{9} A_{noise,K} x_{noise,k} + \sum_{K=1}^{6} A_{speech,K} x_{speech,K} \tag{3}$$

This is essentially performing the following calculation:

$$|Noise(\omega)|^2 = x_1 \text{spectral candidate}_1(f) + \cdots x_9 \text{spectral candidate}_9(f) \tag{4}$$

$$|Speech(\omega)|^2 = x_{10} \text{spectral candidate}_{10}(f) + \cdots x_{15} \text{spectral candidate}_{15}(f) \tag{5}$$

The observation vector's Moore-Penrose inverse (Pseudo-inverse) is taken to obtain the

overdetermined system's least squares solution. The evaluation of equations 4 and 5 will result in

the estimated periodograms of both speech and noise.

Instead of the proposed model above, what is desired is the following functionality:

$$|Noise(\omega)|^2 = \sum_{n=1}^{NFFT} \sum_{n=1}^{9} [F_1, F_2, \ldots F_n] \odot [\theta_1, \theta_2, \ldots \theta_n] \tag{6}$$

From experimentation, it was found that there were two approaches. An intuitive approach used the

GMM-derived codebooks and a WGAN to generate scalar parameters of the same dimensions as

the codebook, as seen in equation 6. Another approach was experimentally derived with no physical meaning.

The experimental approach consisted of replacing the matrix multiplication operator of the proposed system with the Hadamard product (element-wise multiplication). The transpose of the Hadamard product, as seen in equation 7, between the mixture and the pseudo-inverse of A is taken to match the dimensions of x such that the dimensions of x match that of A and the summation, as outlined by equation 6, can take place.

$$x = (A^{\dagger} \odot M(f, t))^{\top} \tag{7}$$

From initial results, the experimental approach resulted in distortionless estimates, as seen in Figure 8, using the GMM-derived codebooks. The WGAN implementation of the experimental approach consisted of generating codebooks that satisfied the constraints set by equation 8 and resulted in accurate estimates of the noise PSD.
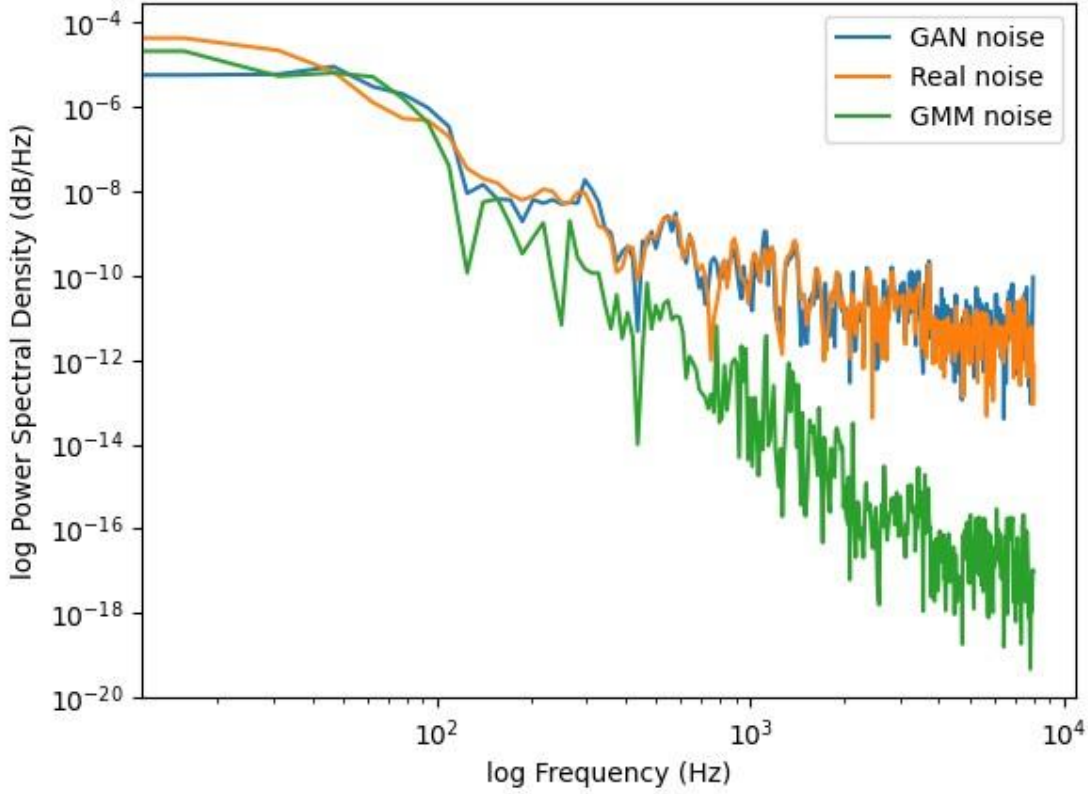
Figure 8: Resulting estimates of the GAN and GMM noise PSD estimates against the ground truth

of a -9dB mixture.

In addition, the WGAN implementation consisting of the GMM-derived codebooks also resulted in

the accurate estimation of the spectral shape of noise. However, a key difference between the two

proposed models was that the GMM-WGAN model required the estimate to be amplified by a

factor of a billion compared to the experimental model, as seen in equation 9, which only needed to
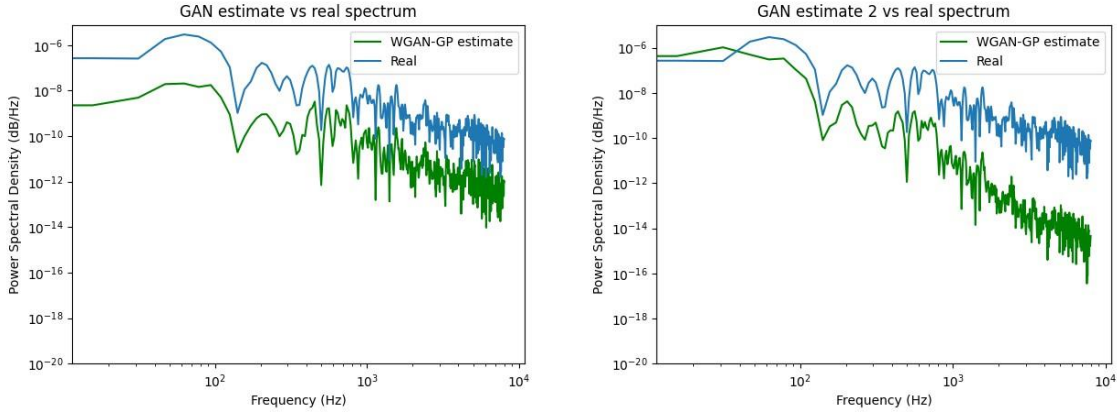
be amplified by a hundred.

Figure 9: Noise spectral shape estimates of the experimental model (left) and the GMM-WGAN model (right).

4.5. WGAN Implementation

To implement the desired WGAN, an already existing WGAN-Gradient Penalty (WGAN-GP) implementation [32] was further developed and adapted to match the requirements of this project. The desired functionality is as follows: The generator learns to generate either the parameters for the GMM-WGAN model or a codebook for the experimental model.

Figure 10 shows dense layers with an expanding shape to encode the mixture into optimal parameters or codebooks to achieve the desired functionality.

```
def build_generator():
    model = keras.Sequential()
    model.add(layers.Dense(1024, input_shape=(1024,)))
    model.add(layers.Dense(2048, activation='relu'))
    model.add(layers.Dense(4096, activation='relu'))
    model.add(layers.Dense(8192, activation='relu'))
    model.add(layers.Dense(16384, activation='relu'))
    model.add(layers.Dense(1024 * 9))
    model.add(layers.Reshape((1024, 9)))
    return model
```

Figure 10: Dense layer generator architecture.

The following decoding block or necessary mathematical transformations were implemented to

convert the resulting parameters into PSDs according to the proposed models, as seen in Figure 11.


```python
def transform_fn(samples, noisemix):
    Numpy_generated_sample = samples.numpy()
    Numpy_noisemix_sample = noisemix.numpy()
    Tensor_size = np.shape(Numpy_generated_sample)[0]
    Estimated_noise_PSD = np.zeros((Tensor_size, 1024))
    for Tensor in range(0, Tensor_size):
        Generated_sample = Numpy_generated_sample[Tensor, :, :]
        Generated_sample = Generated_sample.reshape((1024, 9))
        noise_sample = Numpy_noisemix_sample[Tensor, :]
        noise_sample = noise_sample.reshape(1024)
        Noise_inverse = np.linalg.pinv(np.abs(Generated_sample), rcond=1e-15)
        Noise_coeffs = np.transpose(Noise_inverse * noise_sample)
        Projection = (Noise_coeffs * np.abs(Generated_sample))
        Projection = np.asarray(Projection).clip(min=0)
        for Freq_bin in range(0, 1024):
            Estimated_noise_PSD[Tensor, Freq_bin] = np.sum(Projection[Freq_bin, :])

    transformed_samples = tf.cast(Estimated_noise_PSD, tf.float32)

    return transformed_samples
```

```python
def transform_fn(samples, noisemix, observations):
    Numpy_generated_sample = samples.numpy()
    Numpy_noisemix_sample = noisemix.numpy()
    mean_vector = observations.numpy()
    Tensor_size = np.shape(Numpy_generated_sample)[0]
    Estimated_noise_PSD = np.zeros((Tensor_size, 1024))
    for Tensor in range(0, Tensor_size):
        Generated_sample = Numpy_generated_sample[Tensor, :, :]
        Generated_sample = Generated_sample.reshape((1024, 9))
        Generated_sample = np.abs(Generated_sample)
        noise_sample = Numpy_noisemix_sample[Tensor, :]
        noise_sample = noise_sample.reshape(1024,1)
        Effect_on_mean = noise_sample*mean_vector
        Projection = Effect_on_mean*Generated_sample
        for Freq_bin in range(0, 1024):
            Estimated_noise_PSD[Tensor, Freq_bin] = np.sum(Projection[Freq_bin, :])
    transformed_samples = tf.cast(Estimated_noise_PSD, tf.float32)
    return transformed_samples
```

Figure 11: Decoder/transform implementation of the experimental model (left) and the GMM-GAN

model (right).

The composition of the discriminator network is the same as that of the generator, as seen in Figure

12. The discriminator or critic adjusts the parameters of the discriminator based on the transformed

generations.


```python
def build_discriminator():
    model = keras.Sequential()
    model.add(layers.Reshape((1024,), input_shape=(1024,)))
    model.add(layers.Dense(512, activation='relu'))
    model.add(layers.Dense(256, activation='relu'))
    model.add(layers.Dense(128, activation='relu'))
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(32, activation='relu'))
    model.add(layers.Dense(16, activation='relu'))
    model.add(layers.Dense(1))
    return model
```

Figure 12: Dense layer discriminator architecture.

The training consisted of a paired mixture and real data counterpart to teach the network to generate

accurate noise PSDs.

4.6. CNN for Generating Enhanced Speech

An interesting method to enhance speech was to use an image-image regression algorithm to extract the features of noisy speech by comparing the spectrogram to the clean speech spectrogram. Using the CNN to train this process resulted in a training loss and mean squared error of close to zero after 300 epochs, as shown in Fig. 13. This suggests that the model is performing well and is validated through Fig. 14, which shows that the predicted spectrograms show similar results to the real spectrograms. However, when testing the model with unseen data, the output resulted in a similar shape in the clean representation, as seen in Fig. 15, but loss in both the noise and speech resulted in unintelligible speech. Therefore, instead of using spectrograms for a CNN model to predict clean speech from a noisy mixture, the model is good at classifying the characteristics of clean and noisy speech.
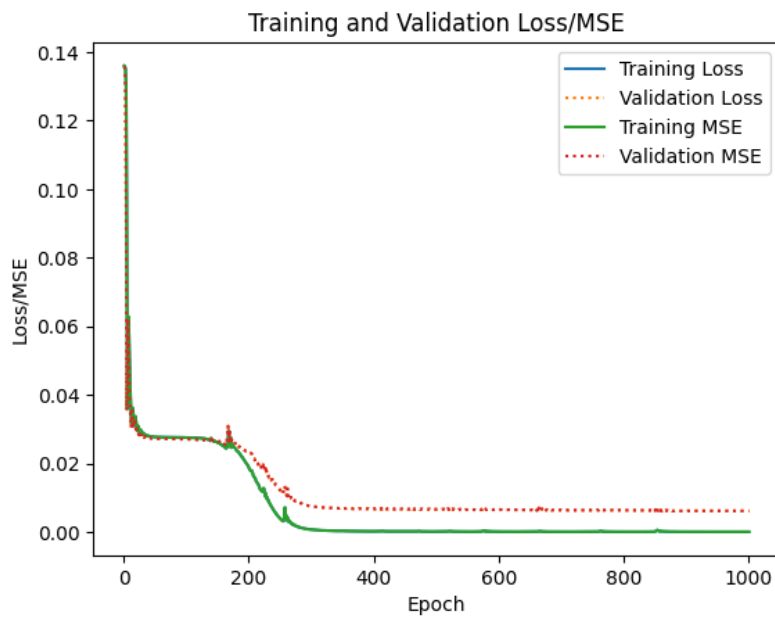

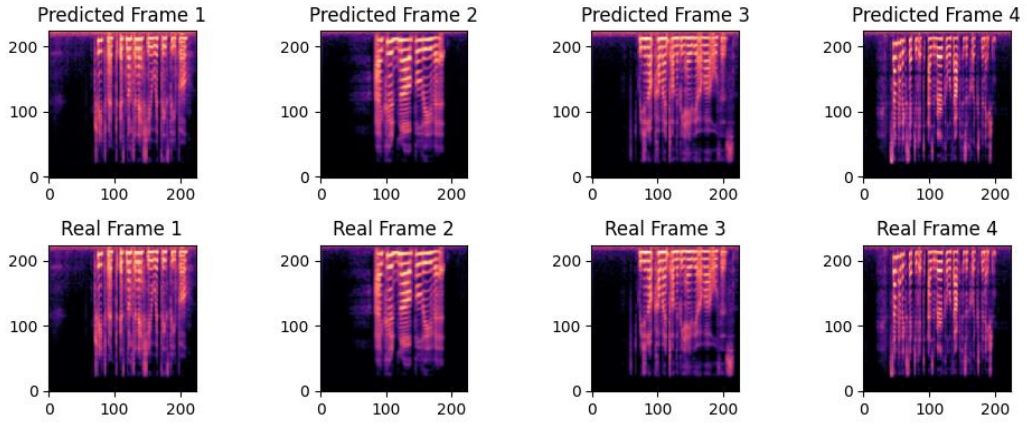Fig. 13. Training and validation loss/MSE over 1000 epochs.

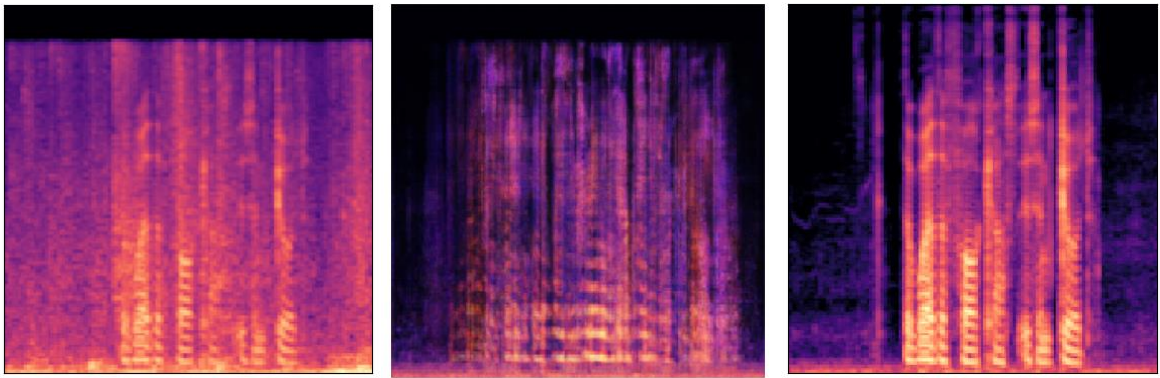Fig. 14: Validation predictions and the clean speech representation.

Fig. 15. The original noisy speech, the predicted speech using the CNN model and clean speech

representation.

4.7. CNN Classification Model

4.7.1. Classes to Classify

Training the CNN classification model with two classes of noisy and clean speech results in any amount of noise being considered noisy speech, which makes the requirements of speech enhancement extremely strict and, in all cases, never classify perceived improvement to quality and intelligibility as good enough speech unless no noise is obtained. To counteract this issue of perceiving improved intelligibility and speech quality, SNR levels are used to perceive if the speech signal is louder than the noise signal. Using a range of 0 to -9 with increments of 1 results in abysmal results as, in most cases, the model is trained spectrograms with minor differences, resulting in the

class being classified as the same, as seen in Fig. 16. The approach resulted in poor training performance of 10% accuracy over ten epochs.
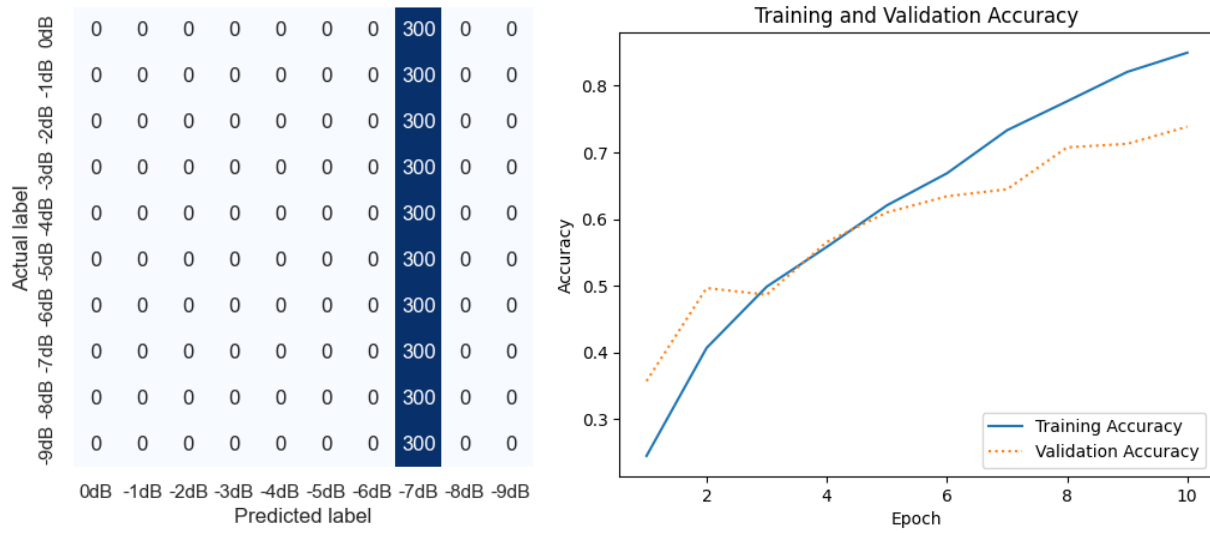

Fig. 16. The first method's confusion matrix of the testing set and training and validation accuracy over ten epochs.

To overcome this issue, a more spread-out approach is implemented with the SNR level ranging from 9 to -9dB with increments of 3. This method resulted in better performance than the first method due to having significant features to distinguish between when performing classification. The second method is shown in Fig. 17, with a training accuracy of approximately 95% after 20 epochs and a testing accuracy of 80%.

Fig. 17. The second method's confusion matrix of the testing set and training and validation accuracy over ten epochs.

As seen in Fig. 18, increasing the number of epochs after 20 to train the model further increases the training accuracy. However, the validation accuracy remains the same, and this is due to the model overfitting to the dataset of 1,500 utterances.
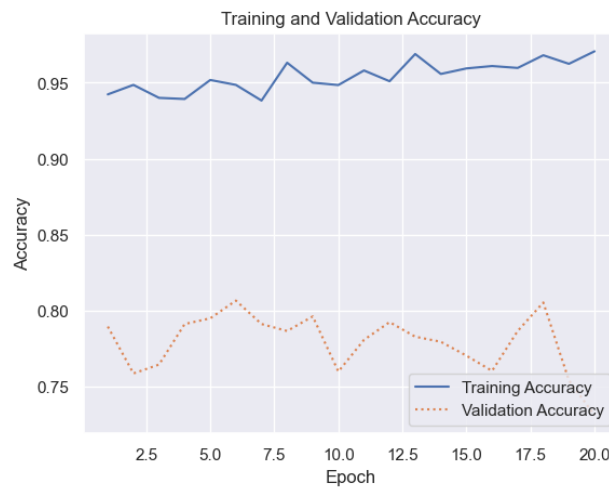

Fig. 18. Training and validation accuracy another 20 epochs.

4.7.2. Spectrograms

Using a wideband rather than a narrowband spectrogram would result in a better representation of differentiation between speech and noise within the spectrogram, as the acoustic features are better

represented. This is evident in Fig. 19, which is the wideband, and Fig. 20 is the narrowband spectrograms, as the former shows more apparent individual pitch harmonics resolved in frequency than the latter [35]. To test the difference in performance, the proposed dataset is transformed into its wideband spectrogram representation with Short Time Fourier Transform parameters [36] of N-FFT = 1024, window function = Hann, hop length = 80 samples and sample per window = 320 samples. As seen in Fig. 21, the wideband spectrogram training accuracy stayed the same. However, the validation accuracy increased by 5% compared to the narrowband spectrogram shown in Fig. 3, proving that wideband spectrograms represent speech better.
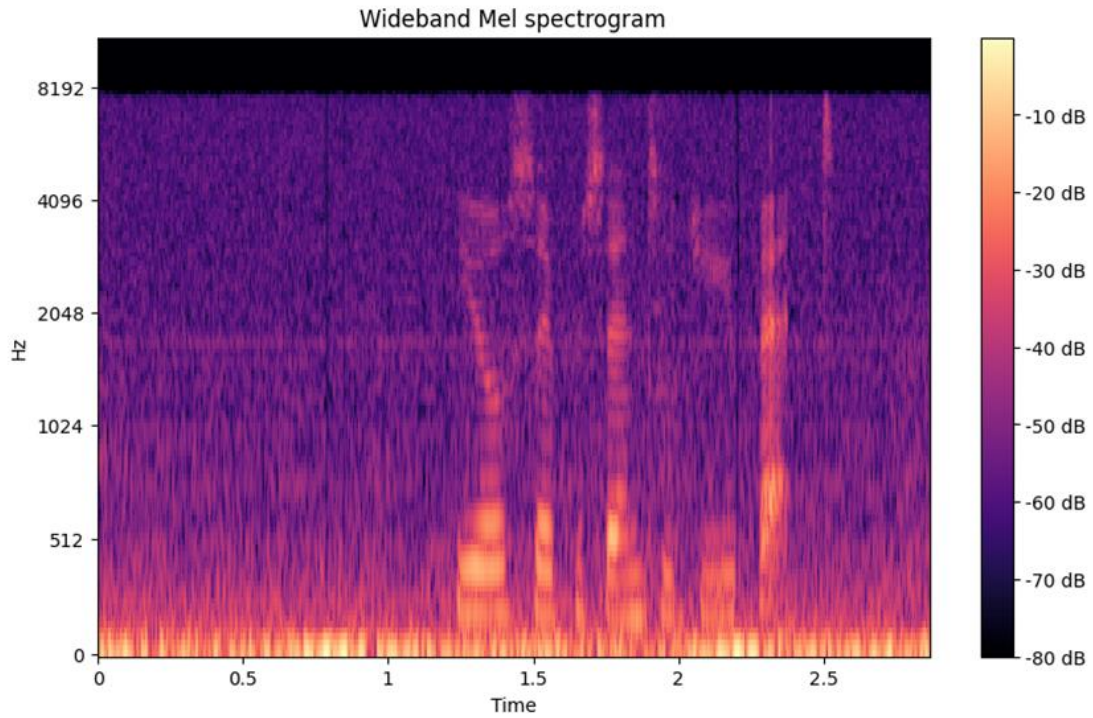

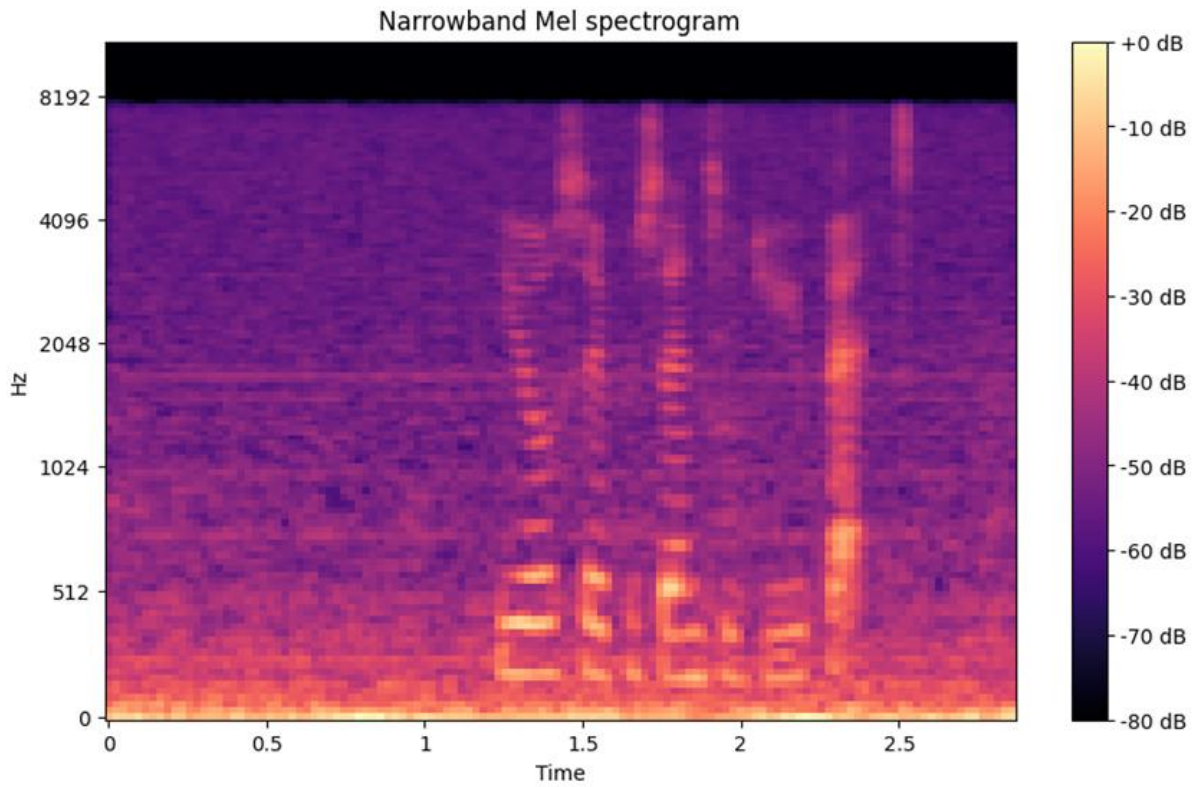Fig. 19. Wideband spectrogram representation.
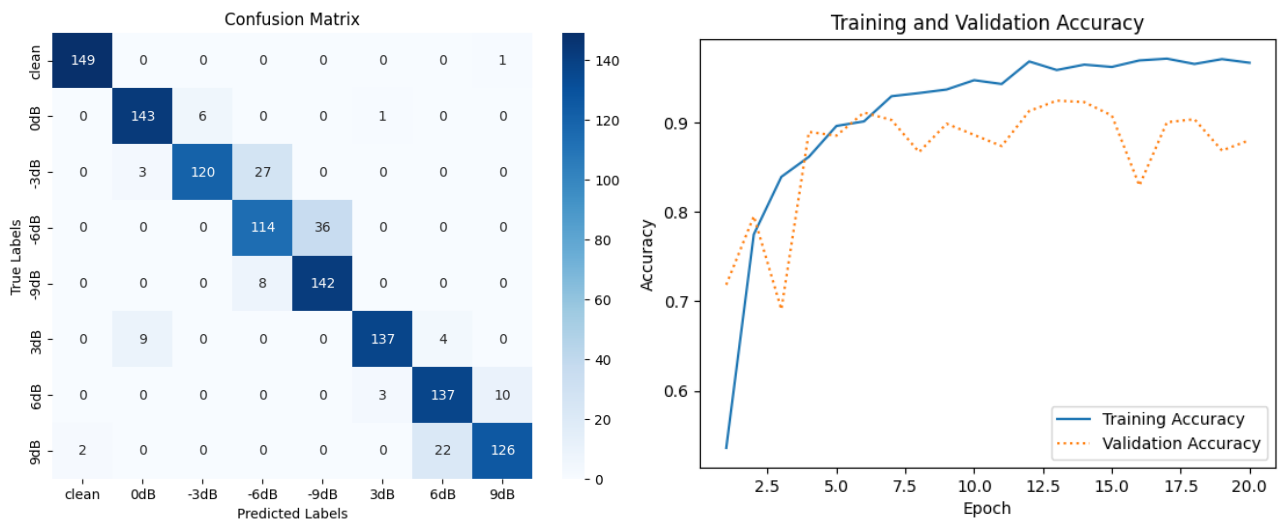
Fig. 20. Narrowband spectrogram representation.

Fig. 21. Using wideband spectrogram confusion matrix of the testing set and training and validation

accuracy over ten epochs.

5. Conclusion

5.1. Conclusion

In conclusion, our study focused on enhancing speech quality in low Signal-to-Noise Ratio (SNR) conditions. We employed a novel approach that involved cycling a low SNR signal through a learning machine until it achieved high SNR status, resulting in enhanced speech. We made several important observations and contributions to the field throughout our investigation. Firstly, we found that the accurate estimation of noise and speech Power Spectral Density (PSD) is crucial for achieving optimal results in speech enhancement. While previous research often emphasised the importance of precise noise estimation, our study highlighted that the quality of speech estimation is equally significant in enhancing the intelligibility of the resulting signal. This insight challenges the common assumption in Wiener filtering studies that accurate noise estimation guarantees improved intelligibility.

Additionally, our proposed approach sheds light on the filter characteristics of the WGAN method by examining augmented codebook entries and the estimated spectra. This comprehensive analysis provides a deeper understanding of the underlying mechanisms in our approach. Furthermore, our experiments demonstrated that, when using the speech enhancement algorithm, speech contaminated with various background noises at different SNR levels slightly improved intelligibility and quality. It is important to note that overcycling through the CNN classification model can have unintended consequences and should be carefully considered.

5.2. Challenges and Reflection

During our study, we encountered several challenges and complexities during this research. One of these challenges is the limited high-quality and diverse datasets required for generalised models capable of working in any environment. Although our models exhibit promising results, we also encountered the challenge of developing and training deep learning models with our computational

setup, resulting in long wait times. Lastly, the challenge of creating a GAN, to begin with, was a struggle, as using a model that is well-known for image generation for speech enhancement resulted in much time spent creating. Overall, for these reasons, something that would have made this project easier would be to have just improved another approach created for speech enhancement instead of creating our new system.

5.3. Future Considerations

In light of these findings, future work in the field could focus on developing more accurate noise and speech spectra estimation techniques, possibly integrating the proposed model with other speech estimators to further improve intelligibility in low SNR mixtures. Additionally, efforts can be directed towards refining the classification model by including unintelligible speech and exploring predictive models for signal recovery at low SNR. Extending the training dataset with realistic and simulated noise data and harnessing improved computational power for longer training durations can significantly enhance the models' performance and capabilities, thereby contributing to the advancement of speech enhancement technology.

Instructions to Setup

Creating Dataset 🔗

- Go to Creating-Dataset
- Open create-dataset.ipynb
- Download the speech dataset and noise dataset
- Extract these folders place them in Creating-Dataset (Create a new folder called "noisy" and extract the noisy dataset in the folder)
- In Step 5 change the number of samples to create to you desired value `n = 1500`
- Then Run All

Generating CNN Classification Model 🔗

- Go to Classification-Setup
- Open class_model.ipynb
- In Step 4 change the batch_size and epoch number to your desired value `batch_size=10, epochs=20`
- Change model name in Step 6 to your desired name `model.save(model_path + 'my_model.h5')`
- Run from Step 1 to Step 6 to save model (Step 7 loads the model and predicts the SNR levels for all WAV files in `path_test_dataset = "../../Dataset/predict_dataset/"`)

Generating GAN Speech Enhancement Model 🔗

- Go to GAN-Setup. Important!! Run this using an IDE. Comments are left in the files regarding directories and functionality!!
- After the steps outlines in Creating Dataset, run the Silero-VAD file to apply VAD SNR. This will generate a new audio files in Dataset/VAD_SNR.
- After the new audio files have been created in (Dataset/VAD_SNR), please copy paste the folder containing clean speech from (Dataset/dataset) to (Dataset/VAD_SNR)
- To prepare the data for the WGAN go to GAN-Setup and run both Generate_Noise_PSDs and Generate_Speech_PSDs. This will create npy files with data. Adjust segments desired.
- After data creation, run the Noise-WGAN-GP-model and Noise-WGAN-GP-models. Models are saved in (Models/GAN-Models). Modify save intervals/batch size as required.

Gaussian Mixture Modelling 🔗

- Go to GMM-Setup
- After the initial dataset made by create-dataset.ipynb is created, run the VAD_Merge. Important!! Run this using an IDE. Comments are left in the files regarding directories and functionality!!
- After the step above, derive the codebooks by running the GMM_Codebook Python script.

Perform Evaluation 🔗

- Go to Evaluation
- Open performing_evaluation.ipynb
- If need be change directory for CNN model in Step 1 `cnn_model = tf.keras.models.load_model('../Models/Classification Models/my_model.h5')`
- If need be change directory for GAN model in Step 1 `Noise_gan = tf.saved_model.load('../Models/GAN Models/Full_Curriculum_4_generator')`
- You can also change the directory for where the evaluation is saved in Step 3 `path_eval = "Evaluation1"`
- Run Step 1 to 3 to create a directory for where the evaluation will be performed
- With the directory created add your desired noisy mixture and clean speech to folders in `path_eval = "Evaluation1"` named `clean_path = path_eval + "/clean"` and `noisy_path = path_eval + "/noisy"`
- Make sure the name of the audio added is speech_"number".WAV the same as the create_dataset convention (With the text inside "" being changed to a number)
- Run Step 4

Fig. 2. Instructions to setup the experiment [34]

List of Files Submitted

- Creating-Dataset: Contains a Python program used to generate the dataset to train and test models

    o create_dataset.ipynb

- Dataset: Where the training dataset is generated after using Creating-Dataset and where evaluation wav files should be placed

    o README.md

- Evaluation: Contains a Python program used to perform speech enhancement and generate the evaluation of one or more audio WAV files

    o performing_evaluation.ipynb

- Exhibition-Day-Presentation: Contains the exhibition presentation and poster

    o Poster_58.pdf

    o Slides.pptx

- Models-Setup: Contains Python code for training both the GAN and the CNN models

    o Classification-Setup

        ▪ Class_model.ipynb

    o GAN-Setup

        ▪ Generate_Noise_PSD.py

        ▪ Generate_Speech_PSD.py

        ▪ Noise-WGAN-GP-model.py

        ▪ Noise_WGAN-GP_model_method_2.py

        ▪ Silero-VAD.py

        ▪ Speech-WGAN-GP-model.py

    o GMM-Setup

- GMM_Codebook.py

- VAD-Merge.py

- Models: Where models generated by Models-Setup are kept and loaded from

  - Classification-Models

    - README.md

  - GAN-Models

    - README.md

- Others: Stores progression documentation, presentation and flowcharts

  - Creating-Figures

    - Making_Spectrograms.ipynb

    - Non_converging_laplacian_figures.py

    - Plot_GMMs.py

    - Training_data_evaluator.py

  - Documentation

    - Documentation.docx

  - Flowcharts

    - Initial flowchart of overall system.png

    - Updated flowchart of overall system.png

  - Other-Presentations

    - 02_05_23 Explaining GANS.pptx

    - Low SNR Optimum Speech Enhancement.pptx

    - The mathematical model used_Progress so far.pptx

- Project-Compendium-Report: Contains the report of the overall compendium

  - Report.docx

- o Report.pdf
- Results: CNN classification results, GAN results and the overall system evaluation results
  - o CNN-classification-results
    - Testing confusion matrix.png
    - Training and validation graph.png
  - o GAN-results
    - Spectrogram_maker
      - Spectrograms images
      - Spectrogram_maker.py
    - GAN_Method_1
      - Contain PNG files of LOG representation of GAN estimate vs real spectrum
    - GAN_Method_2
      - Contain PNG files of LOG representation of GAN estimate vs real spectrum
  - o Overall-system-results
    - Evaluation results.xlsx
- Seminar-Presentation: The presentation slides for the mid-year seminar
  - o Speech_Enhancement_Presentation.pptx
- README.md
- requirements.txt

Seminar and Exhibition

A presentation of a very early prototype of the GAN filter was held for the seminar. This GAN had very poor performance. Overall, the seminar presentation presented the project's progress at that time, where the main goal was to receive feedback from peers regarding the presentation and contribution of the research project. This presentation can be seen in the "Seminar-Presentation" folder in the compendium.

For the exhibition presentation, we plan to present the fully developed model and the results. We have also included a more substantial explanation of the theory required to understand the background of the undertaken research project. This presentation can be seen in the "Exhibition-Day-Presentation" folder in the compendium.

Contributions

Edward's contribution includes the generation of the CNN classification model, the creation of the dataset, and the evaluation of the overall system. Timothy's contribution includes generating enhanced speech using the GAN model and GMM.

References

[1] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," IEEE Trans. Acoust. Speech Signal Process., vol. 27, no. 2, pp. 113–120, Apr. 1979, doi: 10.1109/TASSP.1979.1163209.

[2] J. Su, Z. Jin, and A. Finkelstein, "HiFi-GAN-2: Studio-Quality Speech Enhancement via Generative Adversarial Networks Conditioned on Acoustic Features," in 2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), New Paltz, NY, USA: IEEE, Oct. 2021, pp. 166–170. doi: 10.1109/WASPAA52581.2021.9632770.

[3] Y. Hioka, M. Kingan, G. Schmid and K. A. Stol, "Speech enhancement using a microphone array mounted on an unmanned aerial vehicle," 2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC), Xi'an, China, 2016, pp. 1-5, doi: 10.1109/IWAENC.2016.7602937.

[4] N. Virag, "Single channel speech enhancement based on masking properties of the human auditory system," IEEE Trans. Speech Audio Process., vol. 7, no. 2, pp. 126–137, Mar. 1999, doi: 10.1109/89.748118.

[5] P. C. Loizou and G. Kim, "Reasons why Current Speech-Enhancement Algorithms do not Improve Speech Intelligibility and Suggested Solutions," IEEE Trans. Audio Speech Lang. Process., vol. 19, no. 1, pp. 47–56, Jan. 2011, doi: 10.1109/TASL.2010.2045180.

[6] B. Chen, H. Wang, Y. Wei, and R. H. Y. So, "Truth-to-Estimate Ratio Mask: A Post-Processing Method for Speech Enhancement Direct at Low Signal-to-Noise Ratios," in ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain: IEEE, May 2020, pp. 7509–7513. doi: 10.1109/ICASSP40776.2020.9052919.

[7] S. A. Nossier, J. Wall, M. Moniri, C. Glackin, and N. Cannings, "An Experimental Analysis of Deep Learning Architectures for Supervised Speech Enhancement," Electronics, vol. 10, no. 1, Art. no. 1, Jan. 2021, doi: 10.3390/electronics10010017.

[8] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative Adversarial Networks: An Overview," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 53–65, Jan. 2018, doi: 10.1109/MSP.2017.2765202.

[9] P. Esser, R. Rombach and B. Ommer, "Taming Transformers for High-Resolution Image Synthesis," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 2021, pp. 12868-12878, doi: 10.1109/CVPR46437.2021.01268.

[10] X. Xin, Y. Shen, R. Xiong, X. Lin, M. Yan and W. Jiang, "Automatic Image Generation of Peking Opera Face using StyleGAN2," 2022 International Conference on Culture-Oriented Science and Technology (CoST), Lanzhou, China, 2022, pp. 99-103, doi: 10.1109/CoST57098.2022.00030

[11] G. Liu, K. Gong, X. Liang, and Z. Chen, "CP-GAN: Context Pyramid Generative Adversarial Network for Speech Enhancement," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain: IEEE, May 2020, pp. 6624–6628. doi: 10.1109/ICASSP40776.2020.9054060.

[12] P. Mekha, N. Teeyasuksaet, T. Sompowloy and K. Osathanunkul, "Honey Bee Sound Classification Using Spectrogram Image Features," 2022 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON), Chiang Rai, Thailand, 2022, pp. 205-209, doi: 10.1109/ECTIDAMTNCON53731.2022.9720352.

[13] P. Neammalai, S. Phimoltares and C. Lursinsap, "Speech and music classification using hybrid Form of spectrogram and fourier transformation," Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific, Siem Reap, Cambodia, 2014, pp. 1-6, doi: 10.1109/APSIPA.2014.7041658.

[14] Y. Jia, R. Song, G. Wang, C. Yan, Y. Guo and X. Zhong, "Human Activity Classification with Multi-frequency Spectrogram Fusion and Deep Learning," 2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP), Wuxi, China, 2019, pp. 117-121, doi: 10.1109/SIPROCESS.2019.8868830.

[15] Rethage, D., Pons, J., & Serra, X. (2018). A wavenet for speech denoising. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2018-April*, 5069–5073. https://doi.org/10.1109/ICASSP.2018.8462417

[16]   Manamperi, W., Samarasinghe, P. N., Abhayapala, T. D., & Zhang, J. (2022). GMM Based Multi-Stage Wiener Filtering for Low SNR Speech Enhancement. *International Workshop on Acoustic Signal Enhancement, IWAENC 2022 - Proceedings*, 0–4. https://doi.org/10.1109/IWAENC53105.2022.9914707

[17]   Garofolo, J. S., Lamel, L. F., Fischer, W. M., Fiscus, J. G., Pallett, D. S., & Dahlgren, N. L. (1986). The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus CD-ROM. *Nist*, 1–94. Retrieved from https://perso.limsi.fr/lamel/TIMIT_NISTIR4930.pdf%0Ahttp://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Darpa+timit#7%0Ahttp://perso.limsi.fr/lamel/TIMIT_NISTIR4930.pdf

[18]   Veaux, C., Yamagishi, J., & King, S. (2013). The voice bank corpus: Design, collection and data analysis of a large regional accent speech database. *2013 International Conference Oriental COCOSDA Held Jointly with 2013 Conference on Asian Spoken Language Research and Evaluation, O-COCOSDA/CASLRE 2013*. https://doi.org/10.1109/ICSDA.2013.6709856

[19]   Wang, D., & Chen, J. (2018). Supervised speech separation based on deep learning: An overview. *IEEE/ACM Transactions on Audio Speech and Language Processing*, *26*(10), 1702–1726. https://doi.org/10.1109/TASLP.2018.2842159

[20]   Thiemann, J., Ito, N., & Vincent, E. (2013). The Diverse Environments Multi-Channel Acoustic Noise Database (DEMAND): A database of multi-channel environmental noise recordings. *Proceedings of Meetings on Acoustics*, *19*, 1–6. https://doi.org/10.1121/1.4799597

[21]   Varga, A., & Steeneken, H. J. M. (1993). Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech Communication*, *12*(3), 247–251. https://doi.org/https://doi.org/10.1016/0167-6393(93)90095-3

[22]   Pearce, D., & Hirsch, H. G. (2000). The AURORA experimental framework for the performance evaluation of speech recognition systems under noisy conditions. *6th International Conference on Spoken Language Processing, ICSLP 2000*, (May). https://doi.org/10.21437/icslp.2000-743

[23]   J. Le Roux and E. Vincent. (2014). A categorisation of robust speech processing datasets. Mitsubishi Electric Research Laboratories Technical Report, TR2014-116. Retrieved from https://wiki.inria.fr/rosp/Datasets

[24]   S. Chehresa and M. H. Savoji, "Codebook constrained iterative and Parametric Wiener filter speech enhancement," in 2009 IEEE International Conference on Signal and Image Processing Applications, Nov. 2009, pp. 548–553. doi: 10.1109/ICSIPA.2009.5478717.

[25]   S. Srinivasan, J. Samuelsson, and W. B. Kleijn, "Codebook-Based Bayesian Speech Enhancement for Nonstationary Environments," *IEEE Trans. Audio Speech Lang. Process.*, vol. 15, no. 2, pp. 441–452, Feb. 2007, doi: 10.1109/TASL.2006.881696.

[26]   S. Chehresa and M. H. Savoji, "MMSE speech enhancement based on GMM and solving an over-determined system of equations," in *2011 IEEE 7th International Symposium on Intelligent Signal Processing*, Sep. 2011, pp. 1–5. doi: 10.1109/WISP.2011.6051692.

[27]   Y.-J. Cao *et al.*, "Recent Advances of Generative Adversarial Networks in Computer Vision," *IEEE Access*, vol. 7, pp. 14985–15006, 2019, doi: 10.1109/ACCESS.2018.2886814.\

[28]   M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein Generative Adversarial Networks," in *Proceedings of the 34th International Conference on Machine Learning*, PMLR, Jul. 2017, pp. 214–223. Accessed: Oct. 11, 2023. [Online]. Available: https://proceedings.mlr.press/v70/arjovsky17a.html

[29]   *Deep-Learning/Audio Classification (CNN).ipynb at master · jeffprosise/Deep-Learning · GitHub*. (n.d.). Retrieved October 10, 2023, from https://github.com/jeffprosise/Deep-Learning/blob/master/Audio%20Classification%20(CNN).ipynb

[30]   Machine Learning & Simulation, "Sieving & Convergence Analysis for the EM of the Gaussian Mixture Model in Python," www.youtube.com. https://www.youtube.com/watch?v=Vj4b4xojPMw (accessed Oct. 12, 2023).

[31]   A. Veysov, "Silero VAD," GitHub, Mar. 04, 2022. https://github.com/snakers4/silero-vad

[32]   C.-H. Wu, "WGAN-GP Tensorflow 2.0," GitHub, Aug. 10, 2023. https://github.com/henry32144/wgan-gp-tensorflow (accessed Oct. 14, 2023).

[33]   S. Srinivasan, J. Samuelsson, and W. B. Kleijn, "Codebook driven short-term predictor parameter estimation for speech enhancement," IEEE Trans. Audio Speech Lang. Process., vol. 14, no. 1, pp. 163–176, Jan. 2006, doi: 10.1109/TSA.2005.854113. 31

[34]   *echa548/Speech-enhancement-through-a-CNN-driven-codebook-GAN*. (n.d.). Retrieved October 15, 2023, from https://github.com/echa548/Speech-enhancement-through-a-CNN-driven-codebook-GAN

[35]   A. V. Oppenheim, "Speech spectrograms using the fast Fourier transform," in IEEE Spectrum, vol. 7, no. 8, pp. 57-62, Aug. 1970, doi: 10.1109/MSPEC.1970.5213512.

[36]   *librosa.feature.melspectrogram — librosa 0.10.1 documentation*. (2013). https://librosa.org/doc/main/generated/librosa.feature.melspectrogram.html