Audio Inpainting for English Spoken Language using Machine Learning

CS 6140: Project Report

Jacob Krucinski
krucinski.j@northeastern.edu

Andrew Sayegh
sayegh.a@northeastern.edu

April 24th, 2025

1 Abstract

Within recorded speech, segments of audio are often lost due to noise, interruptions, or transmission errors. Audio inpainting reconstructs these missing segments by analyzing surrounding audio features. This project focuses on the development of a regression-based approach to estimate and fill in gaps in English language audio data. We utilized, compared, and tested the following three regressive and generative models for this task: CNN + Bidirectional LSTM, Generative Adversarial Networks (GANs), and an auto-regressive model. For training, we utilized the large open-source dataset LibriSpeech, from which we applied spectral and waveform-based pre-processing to extract log magnitude spectrograms and introduce a single 200 ms gap of silence. In the reconstructed spectrograms, we were able to see energy bands corresponding to word phonetics and we achieve an PEMOQ score of 0.9849 for the CNN + LSTM and 0.9557 for the GAN, as well as a PEAQ score of -3.8016 for the CNN + LSTM and -3.9087 for the GAN. All our code can be found at [1].

2 Introduction

Audio inpainting is a form of audio restoration in which segments of an audio that were previously corrupted by recording loss, noise, or transmission loss are "filled-in", hence the term inpainting. Since audio data can be very diverse, from spoken language to music or animal sounds, we will be focusing on English spoken language audio recordings with no background sounds. To further limit the scope and unnecessary pre-processing, we insert only 1 gap at a random valid position, which is given to the model during inference to eliminate the need for the model to learn where gaps are located. More details can be found in section 4.1.

Developing methods to perform audio inpainting is important, as it can allow cleaner, more intelligible audio. Also, since audio and many other digital signals like ECGs or EEGs can be represented in the frequency domain using a spectrogram, audio-visual inpainting approaches can be applied in various domains.

From a personal standpoint, both of us are new to audio processing for machine learning applications, and thus we will be learning the concepts of handling and processing audio. This includes understanding signal processing concepts like Short-Time Fourier Transforms (STFT) and English phonetics concepts such as phonemes. Also, since spoken language being diverse due to accent, gender, tonality, the models will have to learn numerous complex aspects of speech such as phonetic and acoustic variability, as well as speech rhythm.

Audio inpainting has many uses from real-time communication losses to forensics. Wireless audio data transmission can occasionally drop packets, or physical data transmission via copper wire may suffer from electromagnetic interference in copper wire, and hence audio inpainting can reconstruct these signals and ensure smoother and more intelligible conversations. In a related manner, audio recordings are sometimes used in criminal investigations and thus audio inpainting can be used to decipher unintelligible or missing parts and allow investigators to solve the case.

The following sections outline existing machine learning-based audio inpainting models, the dataset and pre-processing steps, model architecture modifications, both spectrogram and audio quality metric results, and lastly a direct comparison between the 3 models.

3 Related Work

3.1 CNN + Bidirectional LSTM Bottleneck

We originally wanted to use the bi-directional LSTM and fully connected architecture as proposed in [2], however were unable to replicate its performance. The input and output spectrograms are of size 257×250 and the architecture contains 3 stacked bi-directional LSTM layers with size 250 and a fully connected layer with input size 834 and output size 417. The lack of performance is likely due to the fact that the dataset

that was used only consisted of 1,000 unique sentence recordings, which meant the model needed to learn less speech features as compared to the 28,539 samples in the LibriSpeech "train-clean-100" dataset.

3.2 GAN

To resolve the dataset mismatch and explore a different inpainting technique, we followed the architecture described in [3]. Within the paper they used a generative adversarial network (GAN), a network consisting of two sub-networks, a generator, which fabricates fake data to appear real and, a discriminator, that attempts to distinguish between the real data and the generator's fake data.

Table 1: Generator (PConvUnet)

Layer	**Channel**	**Kernel**	**Stride**	**Pad**	**Activation**
		Encoder			
Input	1+1	-	-	-	-
Enc 1	64	7×7	2	3	LeakyReLU(0.2)
Enc 2	128	5×5	2	2	LeakyReLU(0.2)
Enc 3	256	5×5	2	2	LeakyReLU(0.2)
Enc 4-7	512	3×3	2	1	LeakyReLU(0.2)
		Decoder (Post upsampling & concatenation)			
Dec 1-3	512+512 $\rightarrow$ 512	3×3	1	1	LeakyReLU(0.2)
Dec 4	512+256 $\rightarrow$ 256	3×3	1	1	LeakyReLU(0.2)
Dec 5	256+128 $\rightarrow$ 128	3×3	1	1	LeakyReLU(0.2)
Dec 6	128+64 $\rightarrow$ 64	3×3	1	1	LeakyReLU(0.2)
Final 1	64+1 $\rightarrow$ 64	3×3	1	1	LeakyReLU(0.2)
Final 2	64 $\rightarrow$ 1	3×3	1	1	Tanh

Table 2: Discriminator (PatchGAN)

Layer	**Ch.**	**Kernel**	**Stride**	**Pad**	**Activation**
Input	1	-	-	-	-
Disc 1	1 $\rightarrow$ 64	4×4	2	1	LeakyReLU(0.2)
Disc 2	64 $\rightarrow$ 128	4×4	2	1	LeakyReLU(0.2)
Disc 3	128 $\rightarrow$ 256	4×4	2	1	LeakyReLU(0.2)
Disc 4	256 $\rightarrow$ 512	4×4	1	1	LeakyReLU(0.2)
Output	512 $\rightarrow$ 1	4×4	1	1	None

3.3 Auto-Regressive Model

Aside from deep learning approaches, we came across an auto-regressive (AR) approach [4]. Auto-regressive models use a linear combination of previous samples to predict the current sample value, and repeat this process to predict multiple samples into the future. A signal $e \in \mathbb{R}^n$ can be modeled as an auto-regressive process as:

$$x_n = \sum_{i=1}^{p+1} a_i x_{N+1-i} +, \quad \forall\, n \in 1, 2, \ldots, N+p \tag{1}$$

where a: AR coefficients; ϵ_n: white noise (mean 0, variance 1)

The novelty presented in this paper is the gap-wise Janssen method. The traditional Janssen method uses an overlapping window approach, which may include unrelated parts of the signal. The gap-wise variant fits an AR model audio samples to the left and right of the gap to learn context. The AR coefficients are then estimated using the Burg algorithm.

4 Methods

4.1 Dataset & Data Pre-Processing

As mentioned previously, we are using the LibriSpeech Automatic Speech Recognition (ASR) dataset which contains recordings from publicly available audio books [5]. Specifically, we are using the "train-clean-100" and "train-clean-360" subsets for training, the CNN/LSTM and GAN models, respectively. The '100" subset contains 100 hours of audio recordings sampled at 16 kHz, and the "360" subset has 360 hours.

For pre-processing, we defined a common length of 5 seconds for each audio file. If the original audio is less than 5 seconds, the end is padding with silence. If it is longer than 5 seconds, the audio is trimmed to 5 seconds. The gap, which is 200 ms of silence, is then added at a random starting position uniformly distributed between 0 and 4.8 seconds. For the CNN/LSTM model specifically, there are multiple training samples for a single audio file, as there are 25 different gap positions with only 1 gap per sample. This mimics the dataset setup in [2] and allows the model to learn reconstructions at multiple points in a single audio file not just one, for better context learning. To keep the dataset relatively small, only 100 audio files were used for a total of 2500 audio samples.

Next, the spectrograms for both the original and corrupted audio are extracted. This is done using the Short-Time Fourier Transform (STFT), which applies the Fourier transform to overlapping, windowed segments of the signal.

It is worth noting here that the spectrogram dimensions are then calculated from the parameters listed above as follows:

$$H \times W = \left(\frac{\text{N\_FFT}}{2} + 1\right) \times \left\lceil \frac{\text{sample\_rate} * \text{audio\_length\_sec}}{\text{hop\_length}} \right\rceil \tag{2}$$

All hyperparameters for the dataset, model, and training can be found in the configuration YAML files within our GitHub repository [1].

However, due to the auto-regressive model being implemented for a max gap length of 80ms, we also trained another variant of our CNN + Bidirectional LSTM and GAN models for 80ms so that metrics between the models can be appropriately compared.

4.2 CNN + Bidirectional LSTM Architecture

Despite the issues in [2], we used that model as inspiration for defining a CNN encoder-decoder style network with an LSTM bottleneck. The input and output are both a 257×417 spectrogram. However, when reconstructing the audio, only the reconstruction for the gap segment is added to the corrupted spectrogram using a binary mask. This post-processing workflow is defined in the flowchart below:

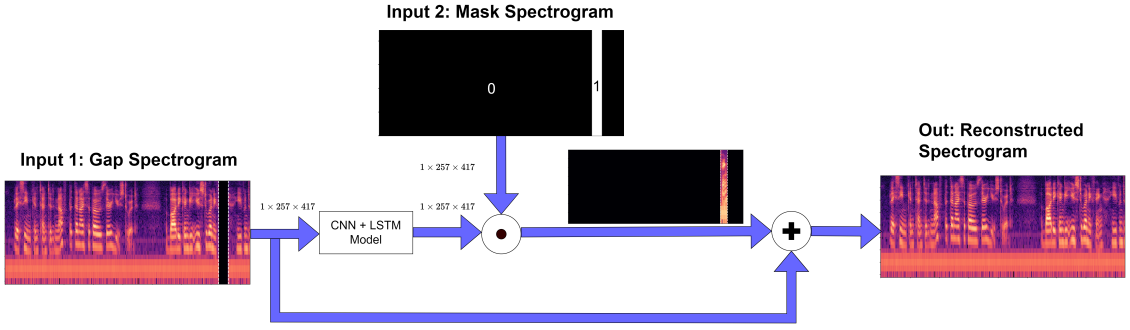

Figure 1: CNN + LSTM Model Spectrogram Reconstruction Workflow

Since our pre-processed data was in the form of a spectrogram, we used the CNN encoder to learn spatial features about common frequency energy patches, which would correspond to specific words or parts of speech. Once these spatial features are learned, they are passed to the bi-direction LSTM bottleneck layer, which learns the temporal relationships between the spatial features. Lastly, the CNN decoder learns to reconstruct the full spectrogram from LSTM cell states.

The full model architecture is shown in the diagram below:

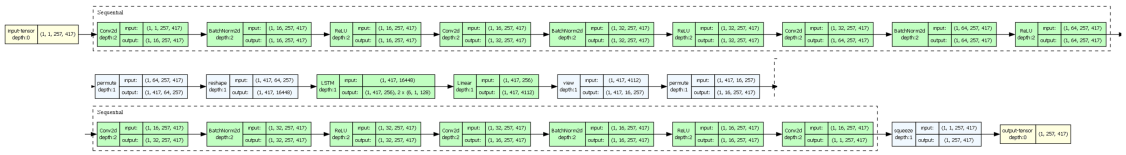

Figure 2: CNN + Bidirectional LSTM Model Architecture

4.3 GAN Architecture

The implementation and architecture of this model was defined earlier. Our approach had to modify some of the architecture to specifically tune it to our needs. Within our implementation, we chose not to use their Two-Strategy Phase Reconstruction Algorithm (TSPRA) for the spectrogram. Instead, we opted to maintain the original phase information for reconstruction, which we found more effective for consistent comparisons of reconstruction along all models tested. The input to the model remained constant with the original design, as both implementations take in the log magnitude spectrogram and apply inpainting over the binary mask regions.

Within implementation there were few details which remained undefined, we innovated and applied such differences. The use of PartialConv2d normalization was noted, yet not defined. We ended up inspiring from [6] using the following equation for the output:

$$\text{output} = \text{output} * \frac{\text{window\_length}}{\text{updated\_mask} + 1\text{e}{-8}} \tag{3}$$

4.4 Auto-Regressive Model

The implementation of this model was discussed previously. However, since this approach work with audio data in the sample domain rather than spectrogram domain, we wrote our own script to load in a single sample pre-processed audio file in the same way the input data structure is defined in the existing implementation. This way, we can compute various quality metrics and compare performance directly between the methods (see Table 3), as opposed to just viewing spectrograms.

4.5 Model Training & Validation

Each model was trained using a configuration indicative of its architectural design, as detailed in the following table:

Table 3: Batch sizes & training

Model	Batch Size	Architecture Details
CNN + LSTM	1×25	Sample rate: 16kHz Spectrogram: 512 FFT, 192 hop length
GAN	8	Sample rate: 16kHz Spectrogram: 512 FFT, 128 hop length
Audio Regressive	1	Sample rate: 16kHz Raw waveform input

As mentioned in section 4.1 we uniquely utilized a 1×25 batch approach, processing a single audio file with 25 different gap positions simultaneously to efficiently capture diverse inpainting scenarios while preserving contextual consistency.

In regards to validation, aside from using visual techniques by comparing the original, gap, and restored spectrograms, we also convert the spectrogram back to an audio file using either the inverse STFT (iSTFT) or the Griffin-Lim algorithm. iSTFT is used for the GAN model that outputs the spectrogram phase information in the complex plane, which leads to near lossless reconstruction. However, if only magnitude information (no phase) is reconstructed, as in the CNN + LSTM, the Griffin-Lim algorithm is used to estimate the original phase information using an iterative heuristic-based optimization approach.

5 Experiments & Results

All training was run on a Dell T5820 Tower with an Intel Xeon W-2235 CPU, 64GB of RAM, and an NVIDIA RTX A4000 GPU. To track various metrics during training time without needing to run a separate testing script, we logged all loss metrics, spectrograms (original, gap, and inpainted), and audio files directly to TensorBoard. TensorBoard is a web interface that provides interactive plots of various metrics to track long training experiments which can take hours or even days.

5.1 CNN + Bidirectional LSTM

The CNN + LSTM model was evaluated simply using an L1 loss with a sum reduction, rather than a mean reduction. This was done to penalize all differences in frequency bin energy in the spectrogram, rather than the average difference. In early iterations of training this model, we noticed that using the mean L1 loss led to reconstructed spectrograms that contained energy blurs across many frequency bins, which yielded poor, imperceptible audio when converted back into the audio domain. Switching to the sum reduction allowed the model to learn the intricate speech frequency patterns that showed up in the spectrogram, thus producing more intelligible audio.

The train loss values over batch steps (100 batch steps/epoch) for both the 80ms and 200ms model variants is shown below:

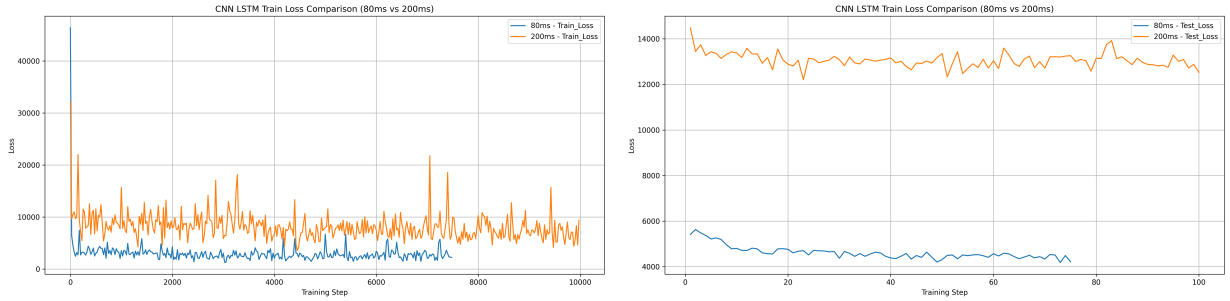

Figure 3: Train (Left) and Test (Right) Loss Curves for CNN + LSTM Model

As a result of the sum reduction described above, these loss values are very high, as there are $257 * 17 = 4,369$ energy values to fill in for the 200ms gap and $257 * 7 = 1,799$ energy values to fill in for the 80ms gap in the spectrogram. The general trend is that the train loss values start very high (around 32,000 for 80ms model and 45,000 for the 200ms model) and stabilize quickly after 500 batches. Despite the loss being noisy, the average loss does decrease over the training to 7,500 for the 200ms model and 3,000 for the 80ms model. This values scale proportionally to the number of energy values calculated above. The testing loss also follows a similar trend, except at different scales. The 200ms model starts at a test loss of 14,500 but decreases slowly to 12,750 by the end of 100 epochs. The 80ms test loss starts at 5,800 and decreases to 4,200 by the end of 75 epochs.

The following spectrograms are the original, with gap, and reconstructed:

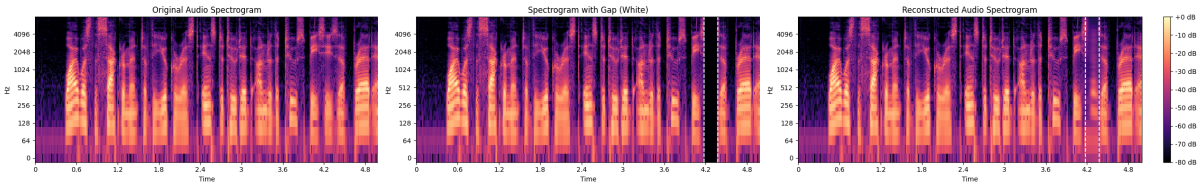

Figure 4: Comparison of spectrograms for original, gapped, and inpainted audio.

We can observe that the model is applying context to the right of the gap, yielding 3 higher energy bands approximately 200, 380, and 500 Hz. However, in the higher frequencies (due to the y axis log scale), the frequency bins are thinner and usually lower energy, but the model does not appear to have learned this, as the high frequencies are a constant low energy value.

5.2 GAN

The performance of the GAN model was evaluated using adversarial, perceptual and reconstruction losses. Training progress was monitored for two gap time frames: 80ms and 200ms. The loss curves for both the generator and discriminator are as follows:

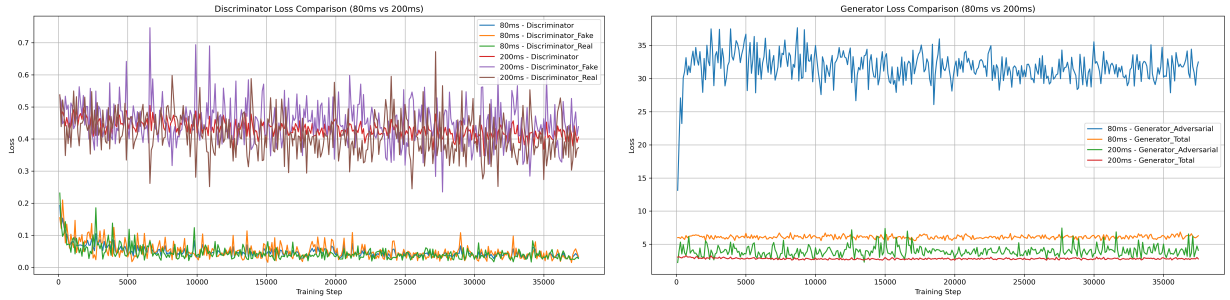

Figure 5: Losses for the Generator & Discriminator

For the 80ms gap, the discriminator loss rapidly decreased during early training (first 5,000 iterations) and stabilized after ∼15,000 training steps, with values below 0.1. This suggests effective learning and convergence. For the 200ms gap, the discriminator losses (real, fake, and overall) showed some improvement, gradually decreasing from approximately 0.5 to 0.4 on average over throughout the 35,000+ steps. However, this improvement came at a much slower rate and remained significantly higher than the 80ms case, with persistent fluctuations throughout training. While some learning occurred, the discriminator struggled to confidently differentiate between real and generated data with the larger gap size.

The generator loss curves further show this disparity. In the 80ms case, the generator's adversarial loss quickly rose to approximately 30-35 and remained high throughout training. This high loss actually indicates a positive training, as it reflects challenge from an effective discriminator that forces the generator to improve. The generator total loss stabilized around 6, suggesting consistent reconstruction performance by the VGG19 stylistic and perceptual metrics. For the 200ms, both the adversarial loss (approximately 3-5) and the total loss (approximately 2.5) remained comparatively low and flat, indicating limited improvement in learning, despite the modest gains in the the discriminator performance.

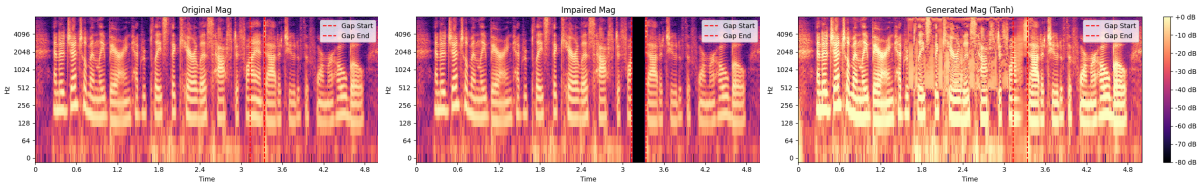

Figure 6: Comparison of spectrograms for original, gapped, and inpainted audio.

We can observe the model's inference through its reconstruction of the spectrogram, which successfully preserves the overall spectral shape and temporal rhythm. The noticeably brighter appearance of the inpainted spectrogram likely results from normalization or scaling artifacts in the reconstruction pipeline. Specifically, we suspect that an unintended application of logarithmic dB scaling may have amplified the apparent spectral energy making a visual overestimation of magnitude in the spectrogram. Sample recordings can also be found on our GitHub [1].

5.3 Auto-Regressive Model

With the auto-regressive model applied in the audio, not spectrogram, domain, the results were solely evaluated on qualitative metrics, which are described below in section 5.4.

The main auto-regressive script performed a sweep the AR model order parameter $p \in [256, 512, 1024, 2048, 3072]$ and AR coefficient learning being either Burg or LPC, for a total of 10 runs. For each run, the AR coefficients are iteratively updated over 20 iterations.

According to the authors of [4], the best model was $p = 2048$ with the Burg learning method. Examining these results, the Signal-to-Distortion (SDR) ratio started poorly near 0, and rose to 1.5 after the 20 iterations. The Perceptual Evaluation Measure of Objective Quality (PEMO-Q), however, stayed near constant at 0.9985, suggesting that a small gap doesn't change the overall intelligibility of the audio very much. Lastly, the Perceptual Evaluation of Audio Quality (PEAQ) strangely gets worse, decreasing from -1 to -1.15. The full details and ranges of these metrics is described below in section 5.4

5.4 Model Comparison

As mentioned in section 4.4, we extracted the reconstructed audio data for a single audio file for all 3 models, and then computed the SDR, PEMO-Q, and PEAQ metrics and arranged them in a table as shown below:

Table 4: Audio Quality Metrics Comparison

Model	SDR	PEMOQ	PEAQ
CNN_LSTM	2.1241	0.9849	-3.8016
GAN	1.3946	0.9557	-3.9087
AutoRegressive	1.5468	0.9965	-1.7313

The domain for each of these metrics is:

- SDR: 0 dB (poor) to 20+ dB (excellent), with 10-15 dB for professional applications[7].
- PEMOQ: Scale of [0,1], higher values indicating better quality[8].
- PEAQ: Range from -4 (poor) to 0 (excellent)[9].

Below are the reconstructed spectrograms for all models side-by-side

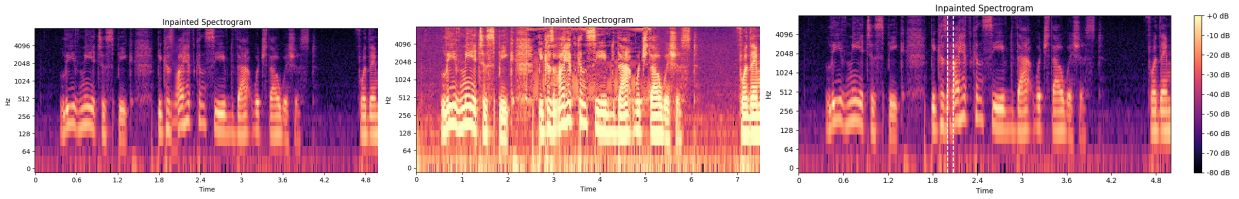

Figure 7: Comparison of reconstructed audio for 80ms gap

6 Discussion

6.1 CNN + Bidirectional LSTM

The CNN + LSTM showed that it was capable of learning some context to the left and right of the gaps, both via energy patterns in the spectrogram and via decreasing raining/testing losses. However, it either lacks large hidden dimensions or data exposure during training to perform as well as the auto-regressive model for short 80ms gaps. This could be related to the number of parameters between the two deep learning models, with the GAN generator having 25.8M parameters and this model only having 18.8M, which is 28% less. However, with the gap-only reconstruction that is defined in Figure 1, this is not a major issue.

The quantitative performance metrics used in Table 4 provide us with a confirmation of our observations from the spectrograms, with the CNN achieving an SDR of 2.1241, a PEMOQ score of 0.9849, and a PEAQ score of -3.8016. The SDR was the highest of the 3 models, likely due to lossless spectrogram reconstruction in the non-gapped audio segments thanks to the gap masking workflow described in Figure 1. Additionally, the PEMOQ score was 0.0292 higher than the GAN model, which is a 3% improvement, and the PEAQ was also slightly better than the GAN model, through still in the poor range. This suggests that while the CNN + LSTM model preserves more of the signal integrity than the other approaches, there is still room for improvement in terms of perceptual audio quality.

6.2 GAN

Among the models evaluated, the GAN model demonstrated a mixed performance that was dependent on the size of the temporal gap. For shorter gaps as demonstrated with 80ms, training showed clear signs of effective adversarial learning. The rapid decrease and subsequent stabilization of discriminator loss indicates the network successfully learned to distinguish between real and generated samples. The generator maintained a high adversarial loss, which at first seemed off, but looking at the total loss we understood this as positive training towards the GAN. As it indicates that despite continuous improvement, the generator faced challenge from an effective discriminator. GAN training revolves around this dynamic. Where the generator rigorously attempts to fool the discriminator. The generator total loss stable around 6 helps confirm that the reconstruction metrics remained consistent throughout training.

The performance metrics we compared the models with provided a quantitative confirmation of our observations, with the GAN achieving an SDR of 1.3946, a PEMOQ score of 0.9557, and a PEAQ score of -3.9087. The PEMOQ score approaching 1.0 indicates that perceptually, the reconstructed audio retained significant quality. However, the relatively low SDR value and the negative PEAQ score (near the bottom of the scale) suggest that while the model successfully preserved some of the perceptual qualities, there were still significant distortions in the reconstructed signal compared to the ground truth. This aligns with the visual evidence from the spectrograms and the loss patterns from our results.

6.3 Auto-Regressive

The auto-regressive model shows good performance in the model comparison in Table 4, but likely does not scale well to gaps longer than 80ms, which is where our deep learning-based model could be more effective. Also, with sample input audio files in [4] consisting of musical instruments, which likely exhibit simpler and/or cleaner tones, this auto-regressive may not be suited to English spoken language audio.
Referring back to the quantitative metrics in Table 4, this model had the 2nd best SDR and best PEMOQ and PEAQ values, with the PEAQ being in the only the slighly annoying/perceptible range as opposed to annoying. This could be due to these methods operating in the audio domain, and with phase being reconstructed for the CNN + LSTM, the conversion to the audio domain with resampling to 48 kHz may have introduced distortions.

7 Conclusion

In this project, we implemented and compared three distinct approaches to audio inpainting for English spoken language: a CNN + Bidirectional LSTM model, a Generative Adversarial Network, and an Auto-Regressive model. Each approach had their own strengths and weaknesses across the various evaluation metrics and perceptual measurements we contrasted. The CNN + LSTM model achieved the best signal-to-distortion ratio. The auto-regressive model has the best perceptual quality scores, particularly for shorter gaps, while the GAN showed promising perceptual results but generally underperformed compared to the other two models. The work demonstrates audio quality whilst easy to interpret, it is essential for the audio to remain intelligible.

8 Future Work

Due to the additional overhead of understanding the basics of audio and digital signal processing, as well as retraining the CNN and GAN models from scratch, we were unable to explore data conditioning. Since the transcripts for all of these recordings was included in the dataset, we would like to use the transcript word embeddings as conditional inputs so This follows a similar technique as described in [2], where video recordings of speakers' faces were used to guide the audio restoration, which proved to increase performance metrics.

References

[1] J. K. Andrew S, "Ml audio inpainting," https://github.com/savage-hacker14/ml-audio-inpainting, 2024.

[2] G. Morrone, D. Michelsanti, Z.-H. Tan, and J. Jensen, "Audio-visual speech inpainting with deep learning," in *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6653–6657.

[3] H. Zhao, "A GAN speech inpainting model for audio editing software," in *Interspeech 2023*, 2023, pp. 5127–5131.

[4] O. Mokrý and P. Rajmic, "On the use of autoregressive methods for audio inpainting," 2024. [Online]. Available: https://arxiv.org/abs/2403.04433

[5] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Apr. 2015, p. 5206–5210. [Online]. Available: http://dx.doi.org/10.1109/ICASSP.2015.7178964

[6] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," *arXiv preprint arXiv:1804.07723*, 2018.

[7] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "Sdr–half-baked or well done?" *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 626–630, 2019.

[8] R. Huber and B. Kollmeier, "Pemo-q—a new method for objective audio quality assessment using a model of auditory perception," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 6, pp. 1902–1911, 2006.

[9] T. Thiede, W. C. Treurniet, R. Bitto, C. Schmidmer, T. Sporer, J. G. Beerends, and C. Colomes, "Peaq - the itu standard for objective measurement of perceived audio quality," *Journal of the Audio Engineering Society*, vol. 48, no. 1/2, pp. 3–29, 2000.