DTU Compute
Department of Applied Mathematics and Computer Science

Real Time Speech Enhancement

Jonas Myhre Schiøtt s204218 | Malthe Ørberg Pedersen s194584 | Viktor Sebastian Petersen s204225

Technical University of Denmark

Introduction

Audio codecs are crucial in tasks such as music-streaming, video-conferences and digital audio playback. Mainly, there are 3 components a good audio codec must satisfy: 1) good compression, 2) low latency and 3) good reconstruction.

In this project, we focus on the noise-removal aspect of this process, hence we want to take a noisy input (say of someone talking and a baby crying in the background), and remove the noise from this signal.

This is done by mixing clean speech sound files with noise files by some signal-to-noise ratio, to simulate a noisy signal. Only then, are we able to compare the result to the original clean speech file.

Model Specifications

Our starting point is the model used by AudioDec [1]. It is a model that takes inspiration from the SoundStream audio codec [2], with some additional tricks. The AudioDec model consists of an encoder, projector, quantizer, codebook and decoder. The bottleneck is configured such that only a certain bitrate (12.8kbps) is allowed to pass through, which is useful for telecommunications. Additionally, both SoundStream and AudioDec achieve models that can work in a real-time streaming scenario.

Initially, we did experiments with continuing training from a pre-trained model from AudioDec. However, these did not yield any satisfactory results and the model did not converge. Therefore, it was decided to eliminate the bitrate limit by excluding the projector, quantizer and codebook from the model. This is still within the purpose of the project, as we are focusing on denoising in real-time rather than balancing that with the needs of telecommunication.

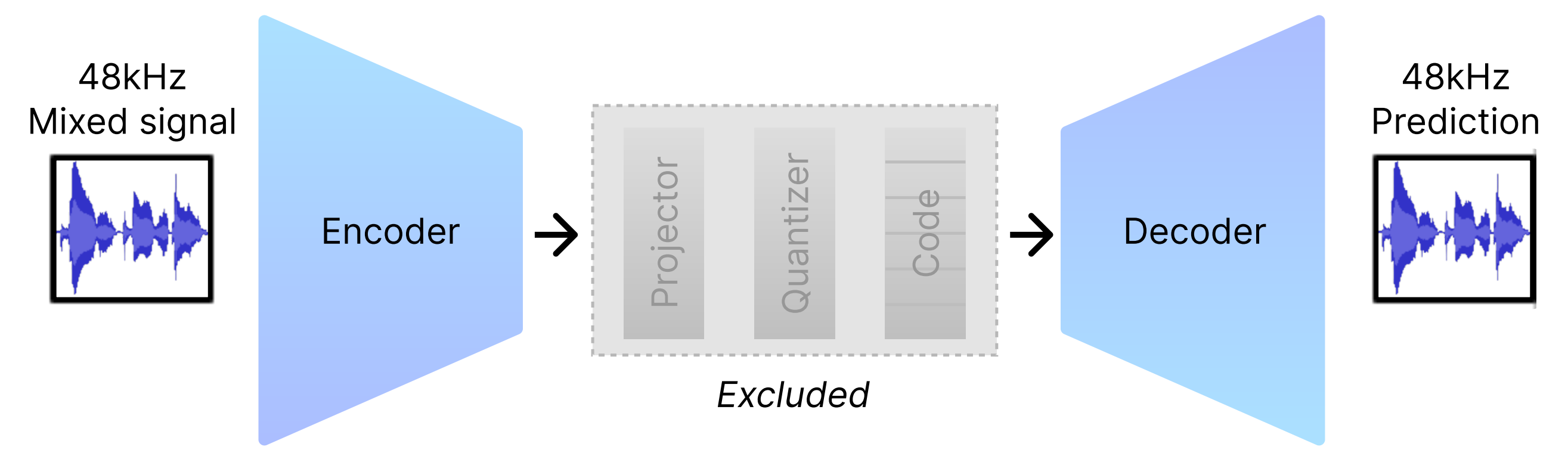

Figure 1: **Model Architecture advanced from the *Soundstream* model architecture.** The middle Bottle neck featured in the *Soundstream* model, was used for compression, and has been excluded in this model.

To elaborate on the structure of the model, the below figure shows a detailed structure of the layers in the Encoder and Decoder which is implemented as described in [2].

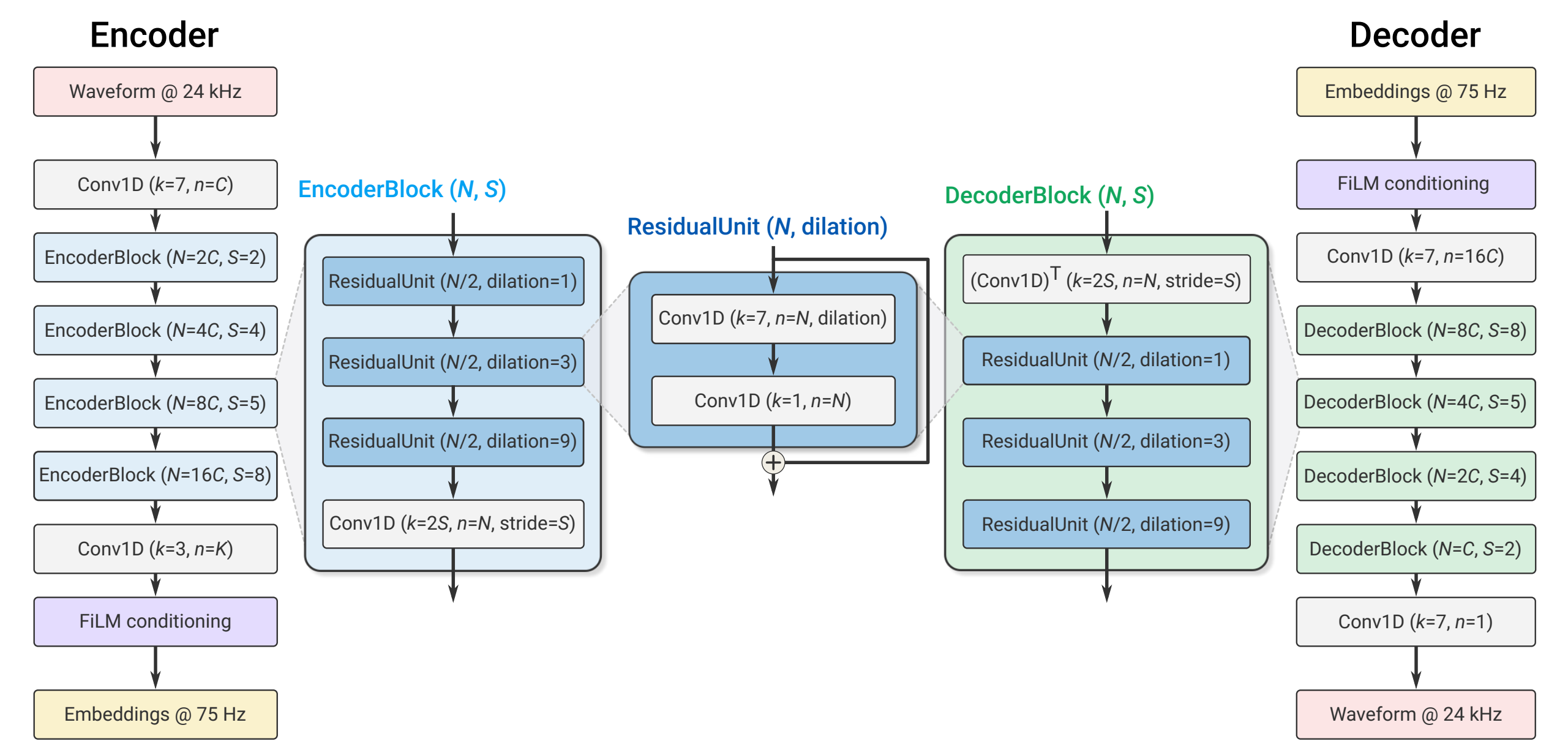

Figure 2: **Specification of layers in encoder/decoder.** The structure of the *EncoderBlocks* of the encoder, as well as the *DecoderBlocks* of the decoder is shown in detail, and both of these use the *ResidualUnit* that is shown in the centre. For our model. $C = 32$ is used. *Image Source: [2]*

For our results, we have used four losses: L1-loss, mel-spectrogram loss, discriminator loss and an adversarial loss. The last three losses are the same as eq 5, 6 and 7 in [1], except a stop gradient operator is not used, as both the encoder and decoder are being trained.

$$\mathcal{L}_{L1} = \mathbb{E}\Big[\,||x - \mathcal{G}(x)||_1\Big], \tag{1}$$

$$\mathcal{L}_{mel} = \mathbb{E}\Big[\,||mel(x) - mel(\mathcal{G}(x))||_1\Big], \tag{2}$$

$$\mathcal{L}_{D} = \mathbb{E}_x\Big[(1 - D(x))^2 + D(\mathcal{G}(x))^2\Big], \tag{3}$$

$$\mathcal{L}_{adv} = \mathbb{E}\Big[(1 - D(\mathcal{G}(x)))^2\Big], \tag{4}$$

where x denotes the true signal, $\mathcal{G}(x)$ is the output of the model, $D(x)$ is the discriminator, and $mel()$ is a function that computes the mel-spectrogram.

Experiments

AudioDec provides a suite of scripts for training, testing and loading models. This, however, is with the assumption that the data set is already stored in files on disk. As the dataset is quite large, it was decided that the mixing of clean and noisy samples should be done just in time, thus rendering the suite difficult to use. Therefore, we introduced our suite of scripts that only uses parts of the AudioDec source code that are necessary, while allowing us more freedom to experiment, i.e., with loss functions. As mentioned before, the continuation of training the AudioDec did not yield a model that converged, so the bitrate limiting bottleneck was excluded from the model. This allows the model to be more flexible, as the representations out of the encoder do not have to be quantized and limited.

Following this, we did different combinations and weighting of the loss functions shown in the previous section. After adding the losses, $\mathcal{L}_{L1}$ and $\mathcal{L}_{mel}$ with a weighting of 45 and 35 respectively, the average epoch loss shows that the model converges steadily and after around 100k iterations, it was possible to hear parts of the clean speech. With this model, we added the adversarial losses, $\mathcal{L}_D$ and $\mathcal{L}_{adv}$ with a weighting of 1, and continued training for around 25k iterations. However, it did not seem like the model was learning anything based on the curves for loss. Most likely, this is due to the metric losses overpowering the adversarial losses.

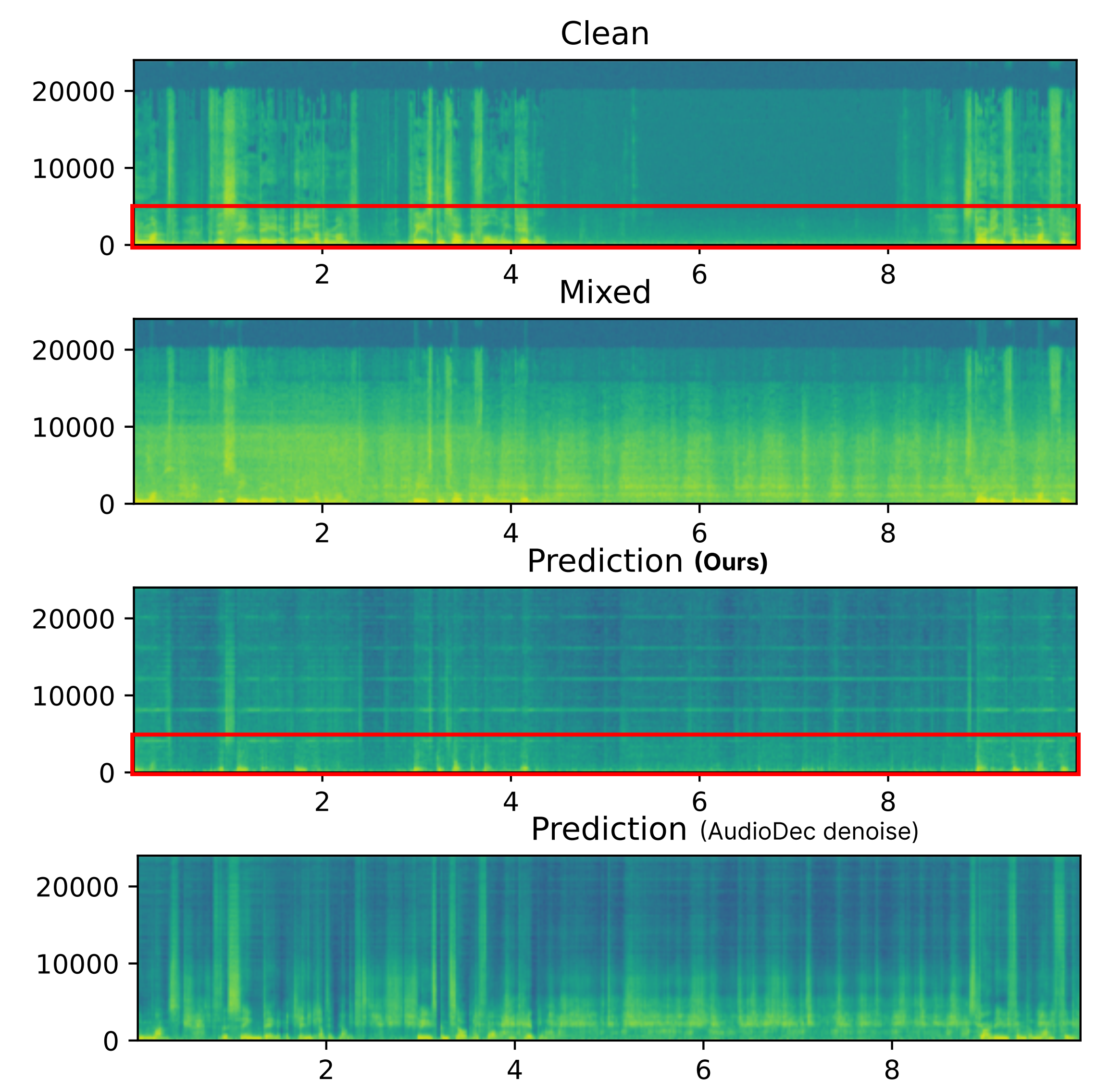

Figure 3: **Mel spectrograms of a random mixed signal.** First: The clean speech sample. Second: The mixed (noisy) sample. Third: The predicted clean speech generated by our model. Fourth: The predicted clean speech using *AudioDec denoise.*

Model Comparison

We have compared our model to the standard AudioDec model in Figure 3. We see that both models have horizontal lines going through the mel-spectrogram, although our lines are more visible and further spread. AudioDecs model produces stronger vertical lines, that are very obstructive for the overall quality of the audio. Unfortunately, the horizontal lines introduce a constant pitch noise and make the voice very robot-like as well. The red boxes highlight the low frequencies, and we see that our model can capture these frequencies.

To further evaluate the performance of the models there exist different measures. Initially, we will use the DNSMOS as defined in [3]. This measure is produced by a neural network that has been trained to predict the Mean Opinion Score (MOS) based on data sourced from subjective human ratings. The DNSMOS can be found in Table 1 for both models.

Table 1: The measure used to evaluate the model is the P808_MOS based on the GitHub repo: https://github.com/microsoft/DNS-Challenge/tree/master/DNSMOS and the paper [3]

	AudioDec Decode	Proposed Model
DNSMOS	2.85	2.45

When listening to the output of the models, we do not seem to have completed our goal. The audio is still somewhat noisy, but you can hear that some noise has been reduced and it sounds better than AudioDec's model. However, this is not reflected by the DNSMOS measure in Table 1, which we think could be a consequence of the horizontal artefacts seen in Figure 3. Besides the quality of the audio reconstruction, the latency of our model is low enough so it can run in real-time. In Table 2 we show the processing time.

Table 2: Live demo runtime (ms) run on a CPU AMD Ryzen 7 6800HS Creator Edition, 3.20GHz and 4 threads

	AudioDec Decode	Proposed Model
Encoder processing time	34.96 ± 3.76	17.35 ± 1.41
Decoder processing time	48.09 ± 5.30	20.09 ± 1.70
System latency	69.13 ± 36.10	75.02 ± 4.87

Issues and Future Work

There are several issues with the model we have now. Firstly, it contains very high frequencies that the clean speech doesn't. Secondly, we introduce artefacts to the signal, such as constant pitch noise. Our model also produces a robot-like voice, which is very unfortunate.

To try and fix the problem with high frequency, we will amplify the generative adversarial network (GAN). The reason is that, according to AudioDec, the model can learn the low-frequency components alone from metric losses, but waveform-detail, high-frequency components and phase synchronization can be improved with a GAN which requires multiple discriminators [1]. In our experiment, the loss value regarding the GAN was very low, so increasing the importance of this loss might help the model with the high frequencies.

Conclusion

AudioDec [1] has come with a great model that is able to reconstruct sounds in 48 kHz with very high speed, which makes it ideal as an audio codec. When trying to use their model for speech enhancement, the results are fine at best, but improving on these results should be possible. We have tried to relax their model, by removing the projector, quantizer and codebook, which we hoped would give us better results. Unfortunately, we haven't been able to improve much on AudioDecs denoising model. Our audio signal includes very high frequencies, constant pitch noise and robotic voices. Although it is possible to hear that some noise gets removed and that we have improved audio quality over AudioDecs model, it still isn't very good unfortunately.

Even with the bad results, the model is very fast, and thus we are able to use the model in real-time. This will be very useful if we manage to fix the problems in our model without introducing further latency, as we then will have a real-time speech-enhancement model.

References

1. Wu, Y.-C., Gebru, I. D., Marković, D. & Richard, A. *Audiodec: An Open-Source Streaming High-Fidelity Neural Audio Codec* June 2023. `http://dx.doi.org/10.1109/ICASSP49357.2023.10096509`.
2. Zeghidour, N., Luebs, A., Omran, A., Skoglund, J. & Tagliasacchi, M. *SoundStream: An End-to-End Neural Audio Codec* 2021. arXiv: `2107.03312 [cs.SD]`.
3. Reddy, C. K., Gopal, V. & Cutler, R. *Dnsmos: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors* in *ICASSP 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2021), 6493–6497.