Name of CodaLab account: Group26
Group Members:

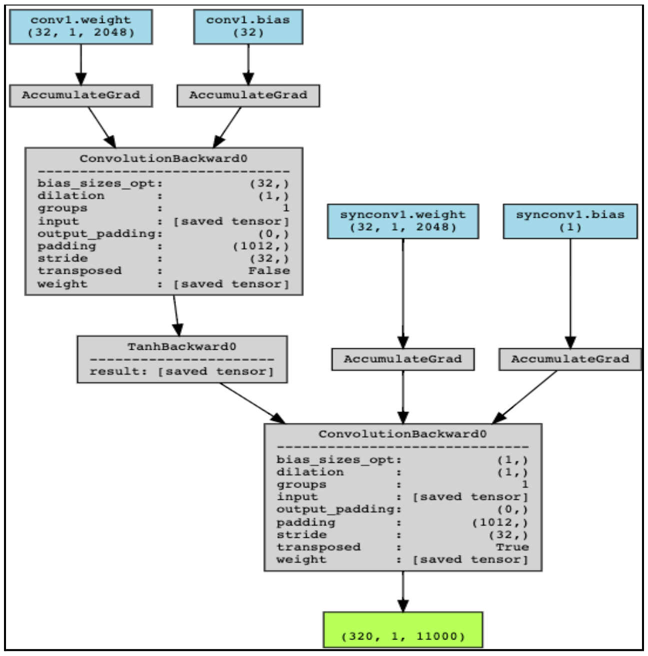
- Djourdan Gomes-Johnson; 2033032
- Ndivhuwo Nyase; 2047606

Description

This report describes the task of using a deep learning model to denoise audio files. Most speech processing techniques use magnitude spectrograms as front-end and are therefore by default discarding part of the signal. We propose a convolutional autoencoder as an end-to-end learning method for speech denoising to recover clean speech signals from a noisy acoustic environment. The goal for the assignment was to build and implement a deep learning model for audio denoising and apply it to a given testset. The dataset was noisy and consisted of short audio fragments of people reading out loud.

Model Architecture

The proposed model is a Pytorch-Implementation of a Convolutional Autoencoder. The goal of the Autoencoder is to reconstruct the clean version of the noisy data. Hence, the model is trained on both the clean dataset and the noisy dataset. The loss function is determined using MSE on the clean signal and the predicted signal. The encoder consists of a Conv1D layer with one input channel and 32 output channels and a tanh layer. The encoder layer is essentially a filterbank that derives analysis from the clean signal. The decoder consists of a Convolutional Transpose1D layer and a tanh layer. The decoder attempts to synthesize clean audio signals from the noisy audio signals.

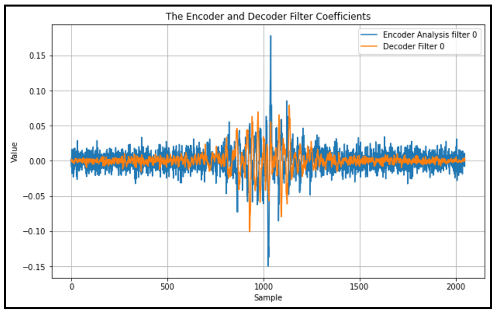


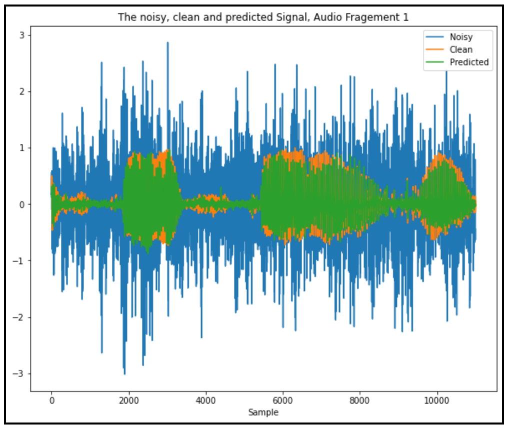
Procedure

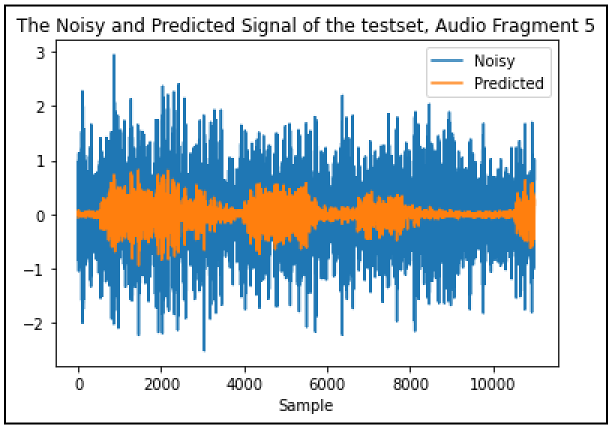
- Step 1: Upsampling the noisy dataset by a factor of two. This is done by resampling the signal along the given axis using polyphase filtering.
- Step 2: Convert the dataset signal vector (mono audio signal), into a 3d Tensor that Conv1D of Pytorch expects for Conv1D input.
- Step 4 : The loss function utilized is the MSE. The model ran to 4000 epochs. The optimizer was Adam.
- Step 5: We use the model with the trained weights to generate clean audio fragments from the test dataset which is noisy.
- Step 6: We implemented a waveform plot of the clean audio fragments against the noisy audio fragments and the predicted audio fragments.
- Step 7: We plotted the waveform of the test (noisy) audio fragments against the predicted audio fragments.

Discussion and Conclusion

- The padding for both the encoder and the decoder was difficult. This changed the sample rate of the predicted values to either be above or below 11 000. We had to examine different padding values because until we gave it the value of 1012.
- Uploading the answer.txt file to codelab provoked many errors. We had to evaluate the error messages and come up with different solutions through trial and error.
- Our method downsampled the audio fragments by a factor of 2 and transmitted them to a noisy channel. We had to make padding=1012 so that the shape of the Ypred could be the shape (9600, 1, 11000).
- Future work would build on this by either adding different layers, or different autoencoder arcitetures or adding an additional audio fragment dataset as training data. Our group attempted to implement a Variational Autoencoder (VAE) however there were many challenges.


The Encoder and Decoder Filter Coefficients
Encoder Analysis filter 0
Decoder Filter 0
Value
Sample


The noisy, clean and predicted Signal, Audio Fragement 1
Noisy
Clean
Predicted
Sample


The Noisy and Predicted Signal of the testset, Audio Fragment 5
Noisy
Predicted
Sample