Listening to Sounds of Silence for Speech Denoising

Ruilin Xu, Rundi Wu, Yuko Ishiwaka, Carl Vondrick, Changxi Zheng

Paper

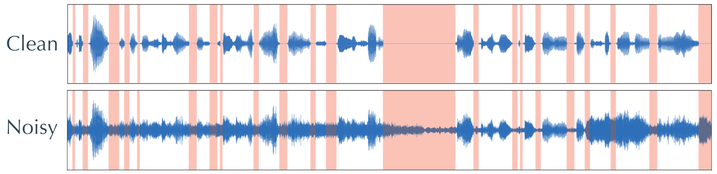
Audio Samples

COLUMBIA UNIVERSITY

SoftBank

Motivation

Observation: In a clean speech signal, there is often a pause between each sentence or word (highlighted in red below). These pauses are exhibited as ***silent intervals***. In a noisy speech signal, silent intervals exposes pure noise. With silent intervals over time, all together they assemble a time-varying picture of background noise, which in turn benefits the denoising process.

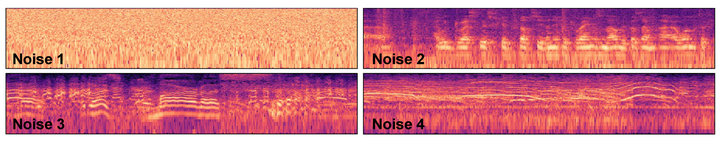


Problem: Even with mild noise, silent intervals become hard to detect.

Dataset & Data Synthesis

Clean speech dataset: We use AVSPEECH, from which we randomly choose 2448 videos and extract their speech audio channels. Among them, we use 2214 videos for training and 234 videos for testing, so the training and testing speeches are fully separate. All these speech videos are in English, selected on purpose: as we show on our project website, our model trained on this dataset can readily denoise speeches in other languages.

Noise dataset: We use two datasets, DEMAND and Google's AudioSet. Both consist of environmental noise, transportation noise, music, and many other types of noises. Our evaluations are conducted on both datasets, separately.

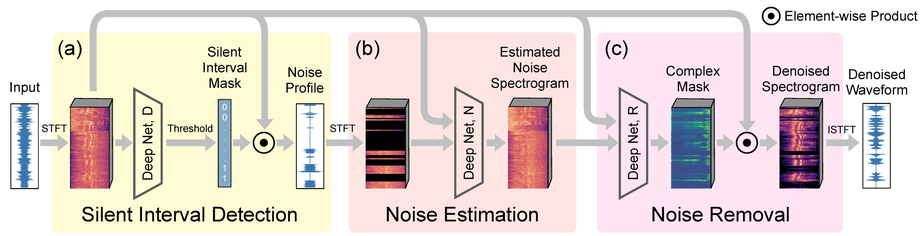


Here are some examples of noise from the noise dataset. Noise 1 is a stationary (white) noise. Noise 2 is a monologue in a meeting. Noise 3 is party noise from people speaking and laughing. Noise 4 is street noise from people shouting with additional traffic noise.

Data synthesis: When synthesizing a noisy input signal, we randomly choose a signal-to-noise ratio (SNR) from seven discrete values: -10dB, -7dB, -3dB, 0dB, 3dB, 7dB, and 10dB; and by mixing the foreground speech with properly scaled noise, we produce a noisy signal with the chosen SNR.

Remarks on creating our own datasets: Unlike many previous models which are trained using existing datasets such as Valentini's VoiceBank-DEMAND, we choose to create our own datasets because, 1) Valentini's dataset has a noise SNR level in [0dB, 15dB], much narrower than what we encounter in real-world recordings; 2) although Valentini's dataset provides several kinds of environmental noise, it lacks the richness of other types of structured noise such as music, making it less ideal for denoising real-world recordings.

Proposed Model



We propose a neural network that harnesses the time distribution of silent intervals for speech denoising. Our model has three components: **(a)** one that detects silent intervals over time, and outputs a noise profile observed from detected silent intervals; **(b)** another that estimates the full noise profile, and **(c)** yet another that cleans up the input signal.

Silent Interval Detection

The first component is dedicated to detecting silent intervals in the input signal. The input to this component is the spectrogram, S_x, of the input (noisy) signal x. The output from this network component is a vector $\boldsymbol{D}(S_x)$. Each element of $\boldsymbol{D}(S_x)$ is a scalar in [0,1], indicating a confidence score of a small time segment being silent.

$\boldsymbol{D}(S_x)$ is then expanded to a longer mask, denoted as $m(x)$. With this mask, the noise profile $\tilde{x}$ exposed by silent intervals are estimated by an element-wise product, namely $\tilde{x} \coloneqq x \odot m(x)$.

Noise Estimation

Inputs to this component include both the noisy audio signal x and the incomplete noise profile $\tilde{x}$. Both are converted by STFT into spectrograms, denoted as S_x and $S_{\tilde{x}}$, respectively. We view the spectrograms as 2D images. Estimating the full noise from the noise profile is conceptually akin to the image inpainting task in computer vision. We denote this process as $\boldsymbol{N}(S_x, S_{\tilde{x}})$.

Noise Removal

Lastly, we clean up the noise from the input signal x. We use a neural network $\boldsymbol{R}$ that takes as input both the input audio spectrogram S_x and the estimated full noise spectrogram $\boldsymbol{N}(S_x, S_{\tilde{x}})$. The output of this component is a vector with two channels which form the real and imaginary parts of a complex ratio mask $\mathbf{c} \coloneqq \boldsymbol{R}\big(S_x, \boldsymbol{N}(S_x, S_{\tilde{x}})\big)$ in frequency-time domain.

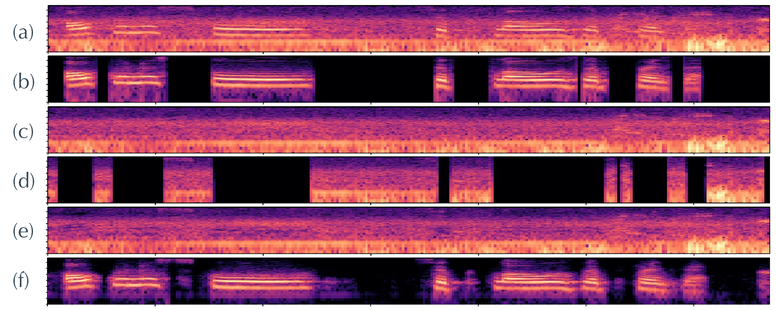
Loss Function

We optimize the following loss function:

$$\mathcal{L}_0 = \mathbb{E}_{x\sim p(x)}\left[\|\boldsymbol{N}(S_x, S_{\tilde{x}}) - S_n^*\|_2 + \beta\left\|S_x \odot \boldsymbol{R}\big(S_x, \boldsymbol{N}(S_x, S_{\tilde{x}})\big) - S_x^*\right\|_2\right]$$

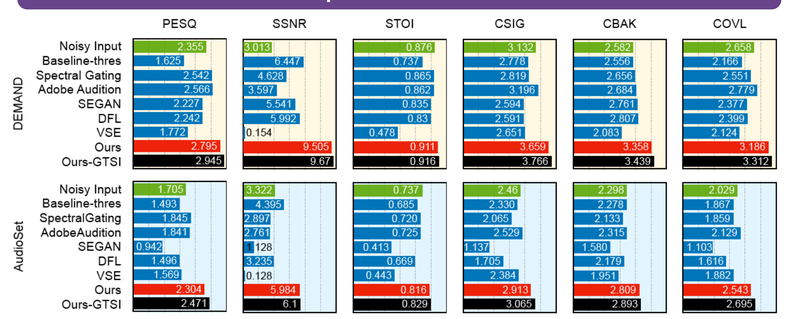
where S_x^* and S_n^* denote the spectrograms of the ground-truth foreground signal and background noise, respectively. The first term penalizes the discrepancy between estimated noise and the ground-truth noise, while the second term accounts for the estimation of foreground signal.

One Example



(a) The spectrogram of a noisy input signal, which is a superposition of a clean speech signal **(b)** and a noise **(c)**. The black regions in **(b)** indicate ground-truth silent intervals. **(d)** The output of the silent interval detection component. **(e)** The estimated noise profile using subfigure **(a)** and **(d)** as the input to the noise estimation component. **(f)** The final denoised spectrogram output.

Comparison Results



The comparisons are conducted using our datasets with noise from DEMAND and AudioSet separately. Ours-GTSI (in black) uses ground-truth silent intervals. The green bar indicates the metric score of the noisy input without any processing.

SOTA Benchmark

Method	PESQ	CSIG	CBAK	COVL	STOI
Noisy Input	1.97	3.35	2.44	2.63	0.91
WaveNet	–	3.62	3.24	2.98	–
SEGAN	2.16	3.48	2.94	2.80	0.93
DFL	2.51	3.79	3.27	3.14	–
MMSE-GAN	2.53	3.80	3.12	3.14	0.93
MetricGAN	2.86	3.99	3.18	3.42	–
SDR-PESQ	3.01	4.09	3.54	3.55	–
T-GSA	3.06	4.18	3.59	3.62	–
Self-adapt. DNN	2.99	4.15	3.42	3.57	–
RDL-Net	3.02	4.38	3.43	3.72	0.94
Ours	3.16	3.96	3.54	3.53	0.98

To compare with SOTA methods, we train our model on Valentini's DEMAND, the same dataset used across all methods.