Interspeech 2022
18-22 September 2022, Incheon, Korea

Application for Real-time Personalized Speaker Extraction

Damien Ronssin[1], *Milos Cernak*[1]

[1] Logitech Europe S.A., 1015, Lausanne, Switzerland
dronssin@logitech.com, mcernak@logitech.com

Abstract

This short paper demonstrates an audio processing desktop application that allows isolating in real-time the voice of a specific speaker from the possibly noisy audio input after a short enrollment phase. The machine learning model embedded in this application suppresses all other sounds than the target voice from the incoming audio stream, including disturbing distractor voices. In the context of a growing need for video-collaboration solutions, personalized speech enhancement enables the use of such technologies in more challenging acoustic environments, i.e., in the presence of near distractor speech. In this situation, classical speech enhancement systems typically fail as they do not filter out any speech, hence the need for personalized methods. The presented application is an all-in-one solution for personalized speech enhancement: it allows the user to enroll and then to apply the effect seamlessly for one-to-one or one-to-many online meetings.

Index Terms: speaker extraction, personalized speech enhancement, real-time audio processing, speech separation

1. Introduction

Most speech enhancement algorithms used in modern telephony or videoconferencing systems are designed to remove all sounds but human speech. This approach is of great help in many situations, such as when there are traffic or typing noises, for example. However, when an unwanted human voice is present in the signal, such algorithms would not filter it as they cannot distinguish between the user's voice and distractor ones. Still, a distractor speaker can significantly impact the communication's quality, specifically intelligibility. Such a scenario is, in fact, very common: for example, when someone has an online meeting at a desk in an open office space where a lot of unrelated speech can occur nearby and eventually be transmitted, or when calling from home from a shared room.

To address this issue, we created an application based on a speaker extraction machine learning model that can isolate the voice of a specific speaker in real-time. The identity of the target speaker is given to the model via a speaker embedding computed on a short recording (≈10-20s) of the target speaker's voice. That embedding is called the voiceprint. The extraction model was trained both with interfering speech and noises and can thus handle both situations described earlier, i.e., both noises and distractor speech are suppressed.

The desktop application we present here allows enrolling the user/speaker, getting the corresponding speaker embedding, and then using the speaker extraction algorithm with one of the previously enrolled speakers. This application can be used to pre-process audio for an online meeting via Zoom, Skype, or Microsoft Teams, for example. It enables better communication than classical speech enhancement methods in noisy environments with near distractor speakers.

In this paper, we will first describe the machine learning models used to perform the speaker extraction. Then we will present the application that runs these models in real-time.

2. Algorithms

The task of retrieving a specific speaker's voice in an audio mixture is referred to as speaker extraction. Recently, different systems have been published, for example VoiceFilter [1], VoiceFilter-Lite [2] and Personalized Percepnet [3]. Those systems work similarly as they all use a speaker embedding as an auxiliary input to identify the target speaker's voice. Note that both Personalized Percepnet and VoiceFilter-Lite are real-time systems while VoiceFilter is not.

Our proposed system's architecture is adapted from that of VoiceFilter that we made causal to enable the real-time capability. Still, our approach differs from VoiceFilter-Lite as the latter does not output audio and is used only as a pre-processing block for an Automatic Speech Recognition (ASR) system. The final extraction system relies on two machine learning models: the voice encoder and the extraction model, which are described below.

2.1. Voice encoder

The voice encoder takes as input an enrollment recording from the target speaker and outputs a fixed size speaker embedding, the voiceprint. Such a model is trained to produce discriminative embeddings for different speakers. For this task, we used as is an open-source implementation[1] of the same voice encoder used in VoiceFilter [4]. The resulting 256-dimensional speaker embedding is then given as auxiliary input to the extraction system.

2.2. Extraction model

2.2.1. Model architecture

The extraction model predicts a sequence of masks to be applied to the input signal's frequency representation, finally getting the isolated voice of the target speaker as output. The 16 kHz incoming audio is first segmented into 40ms frames with a 16ms hop size, and the spectrogram is then obtained by computing a Fast Fourier Transform (FFT) of size 1024 on each frame. The spectrogram frames are then fed to a small convolutional neural network (CNN) operating only on the frequency axis. Subsequently, the speaker embedding is repeatedly concatenated for each frame to the output of the CNN, and the result is propagated into a stack of three LSTM layers. Finally, a few fully connected layers are applied to obtain a mask to be multiplied with the current spectrogram frame. As the CNN operates only on the frequency axis and the LSTM layers are uni-directional, the model is causal and operates in a streaming fashion, i.e., frame by frame.

[1] https://github.com/resemble-ai/Resemblyzer

2.2.2. *Training and results*

The model was trained using synthetic mixtures combining two LibriSpeech [5] utterances from different speakers. Noise or music samples from MUSAN [6] were also randomly added. The relative distractor loudness compared to the target level was also set randomly for each mixture between -25 dB and 10 dB and between -25 dB and -5 dB for noises and music. The loss function is the same as in [1].

The model achieves 8.5 dB Scale-Invariant Signal to Distortion Ratio improvement (SI-SDRi) on mixtures generated with LibriSpeech clean test set utterances. There is, of course, some variability in cases where the model performs better or worse. For example, the model is very robust when the target voice is dissimilar to the distractor one. It is also the case when the target voice is louder than the distractor. On the other hand, the extraction quality is impacted when the target and distractor voices are very similar, and the distractor level is high. Still, in a call setting, distractor speech rarely reaches the target speech level as the target speaker is usually significantly closer to the microphone.

3. Desktop application

The desktop application allows to take audio input from a microphone and process it with the extraction model. The resulting audio can then be routed to any audio/video conferencing application using a virtual microphone such as VB-CABLE[2]. The application is implemented using the JUCE framework[3] and works both on Mac and Windows platforms. The model inference is performed using the ONNXRuntime library [7]. All the audio processing, including downsampling followed by FFT, model inference, iFFT, overlap-add, and finally upsampling, is done on a single CPU core. When running, the application uses less than 10% of an Intel Core i5-7300U CPU running at 3.5 GHz. That number could be significantly decreased further by quantizing the weights of the model. All the data associated with the voice identity of a user, i.e., the recording and the voiceprint, is stored on the user's computer and can be deleted at any time.

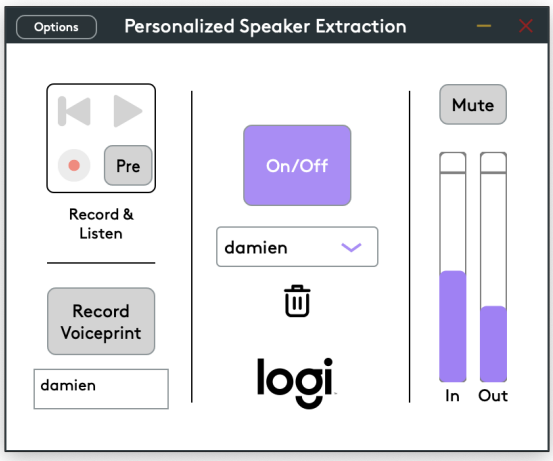

Figure 1: *Application User Interface*

[2] https://vb-audio.com/Cable/
[3] https://juce.com/

The application's interface is shown in Figure 1. As it can be seen on the bottom left, a voiceprint with a custom name entered by the user can be stored. The voiceprint to use can be selected from a list in the middle, and the effect can then be turned on. The currently selected voiceprint can be deleted simply by clicking the bin icon. On the right, input and output level meters are displayed so that the amount of filtering can be monitored at any time. Also, on the top left, a recording and playback feature can be used to hear the difference between before and after the effect (Pre/Post). The total latency is approximately 50 ms: the model lookahead is 40 ms (1 frame) plus small resamplers lookahead and some processing time. This short delay does not affect the communication's quality.

4. Conclusion

This paper has presented a desktop application that filters out all distractor speech and sounds but the user's voice. The usage requires a short enrollment phase. The application embeds a speaker extraction model performing joint denoising and interfering speaker removal. It works by predicting masks to apply to incoming audio's spectrogram. The proposed method provides the SI-SDR improvement of 8.5 dB in the presence of distractor speech, a situation that most speech enhancement systems can't address due to the impossibility of distinguishing between the target voice and distractor ones. The described application can pre-process audio for one-to-one or one-to-many meetings via any videoconferencing service (Zoom, Skype, Microsoft Teams, . . .) and enhance the communication experience when calling from crowded, noisy spaces.

The presented application is a research prototype that will be improved in the near future. Especially the current training dataset only contains English speech and should soon include more languages. Also, the model currently processes 16 kHz audio while the trend is toward wider band audio. Finally, to reduce the application's CPU usage and energy footprint, the extraction model should be optimized for inference especially quantized and maybe pruned. These challenges pave our future work.

5. References

[1] Q. Wang, H. Muckenhirn, K. Wilson, P. Sridhar, Z. Wu, J. Hershey, R. A. Saurous, R. J. Weiss, Y. Jia, and I. L. Moreno, "Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking," *arXiv preprint arXiv:1810.04826*, 2018.

[2] Q. Wang, I. L. Moreno, M. Saglam, K. Wilson, A. Chiao, R. Liu, Y. He, W. Li, J. Pelecanos, M. Nika *et al.*, "Voicefilter-lite: Streaming targeted voice separation for on-device speech recognition," *arXiv preprint arXiv:2009.04323*, 2020.

[3] R. Giri, S. Venkataramani, J.-M. Valin, U. Isik, and A. Krishnaswamy, "Personalized percepnet: Real-time, low-complexity target voice separation and enhancement," *arXiv preprint arXiv:2106.04129*, 2021.

[4] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4879–4883.

[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, 2015, pp. 5206–5210.

[6] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[7] ONNX Runtime developers, "Onnx runtime," https://onnxruntime.ai/, 2021, version: 1.10.0.