MULTI-USER VOICEFILTER-LITE VIA ATTENTIVE SPEAKER EMBEDDING

Rajeev Rikhye[*], *Quan Wang*[*], *Qiao Liang, Yanzhang He, Ian McGraw*

Google LLC, USA
{rvrikhye,quanw}@google.com

ABSTRACT

In this paper, we propose a solution to allow speaker conditioned speech models, such as VoiceFilter-Lite, to support an arbitrary number of enrolled users in a single pass. This is achieved via an attention mechanism on multiple speaker embeddings to compute a single attentive embedding, which is then used as a side input to the model. We implemented multi-user VoiceFilter-Lite and evaluated it for three tasks: (1) a streaming automatic speech recognition (ASR) task; (2) a text-independent speaker verification task; and (3) a personalized keyphrase detection task, where ASR has to detect keyphrases from multiple enrolled users in a noisy environment. Our experiments show that, with up to four enrolled users, multi-user VoiceFilter-Lite is able to significantly reduce speech recognition and speaker verification errors when there is overlapping speech, without affecting performance under other acoustic conditions. This attentive speaker embedding approach can also be easily applied to other speaker-conditioned models such as personal VAD and personalized ASR.

Index Terms— VoiceFilter-Lite, speaker embedding, attention mechanism, keyphrase detection

1. INTRODUCTION

Many speech models rely on a target speaker embedding as a side input to achieve either better performance or a more personalized user experience. This target speaker embedding is usually obtained via an offline enrollment process [1, 2], and is directly usable at runtime. We refer to this type of speech models as *speaker-conditioned models*; in comparison, we refer to speech models that do not rely on target speaker embeddings as *generic models*.

For example, speech enhancement or separation models can leverage the target speaker embedding to avoid the complexity of permutation invariant training [3], and directly output the clean waveform of the target speaker. Representative works include DENet [4], SpeakerBeam [5, 6, 7], VoiceFilter [8] and more recently SpEx [9].

VoiceFilter-Lite is a variant of speaker-conditioned speech separation models that is optimized for streaming on-device applications, such as automatic speech recognition (ASR) [10] and text-independent speaker verification (TI-SV) [11]. It directly takes the acoustic frontend features of ASR (*e.g.* stacked log Mel-filterbank energies) as inputs and outputs, instead of operating on the audio waveforms.

Target speaker embedding can also be used for voice activity detection (VAD) and speech recognition. In a personal VAD system [12], an always-on, small-footprint VAD model can be used to reject frames containing either non-speech noise or speech from a non-target speaker. In scenarios where a keyword detector is not preferable, personal VAD largely reduces false triggering rate of more expensive downstream models such as ASR and speaker recognition, thus saving computational cost and battery consumption. In a target-speaker speech ASR system [13, 14], the target speaker embedding helps the ASR model to better recognize the speech from the target speaker under noisy environments with background noise and interfering speakers.

Most of the above mentioned works assume that only a single target speaker embedding is present at runtime. This assumption is usually valid for personal devices, such as mobile phones. However, for shared devices, such as smart home speakers and displays, multiple users can enroll their voices on the device, and thus more than one target speaker embeddings are usually present at runtime. In such cases, we need to either: (1) disable the personalized model and fallback to a generic model; or (2) run the entire system for multiple passes for each enrolled speaker embedding and choose the best output, which can be extremely expensive and unacceptable for on-device applications.

In this paper, we address the multi-user challenge for speaker-conditioned models by leveraging an attention mechanism [15] on the enrolled speaker embeddings. We implemented a multi-user VoiceFilter-Lite model that can significantly improve speech recognition and speaker verification performance when the input audio contains overlapped speech, and can take an arbitrary number of enrolled speaker embeddings as side input. This multi-user VoiceFilter-Lite model is also critical to a personalized keyphrase detection system on shared devices. Apart from VoiceFilter-Lite, we would like to point out that the same mechanism can be easily applied to various other applications, such as waveform-based speech separation, personal VAD and personalized ASR.

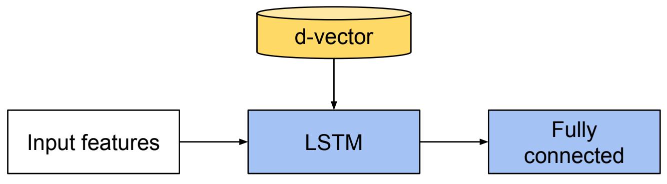
The idea of "attentive speaker embedding" can also be found in other works, where the attention mechanism is mostly used as a pooling layer to aggregate frame-level embeddings in the same utterance [16, 17, 18, 19]. To avoid confusion, we emphasize that in this work, the attentive speaker embedding attends to multiple embeddings from different speakers that are already obtained via a separate offline enrollment process, which is more relevant to the work in [20, 21].

2. METHODS

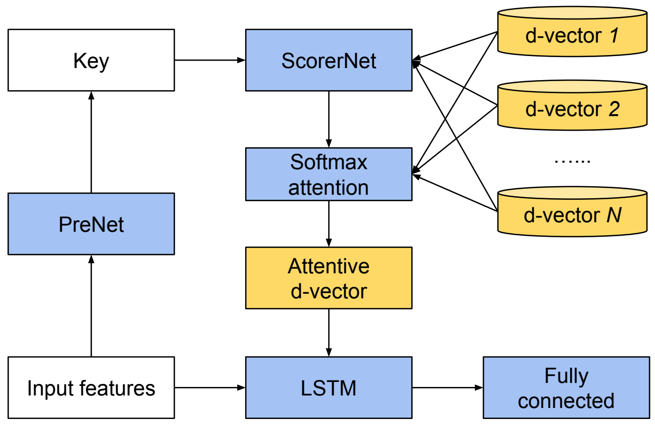
2.1. Review of VoiceFilter-Lite

VoiceFilter-Lite is a targeted voice separation model for streaming on-device speech recognition [10], as well as text-independent speaker verification (TI-SV) [11]. It assumes that the target speaker has completed an offline enrollment process, which uses a speaker recognition model (see Section 3.3) to produce an aggregated embedding vector $\mathbf{e}$ that presents the voice characteristics of this

[*] Equal contribution.



(a)



(b)

Fig. 1. Attentive speaker embedding in VoiceFilter-Lite. (a) Original VoiceFilter-Lite, where the d-vector of the single target speaker is frame-wise concatenated to the input features; (b) Multi-user VoiceFilter-Lite, where we support an arbitrary number N of d-vectors (the target speaker embedding is included in these N d-vectors).

speaker. In this work, we use the d-vector embedding [22] trained with the generalized end-to-end extended-set softmax loss [23].

Let $\mathbf{x}^{(t)}$ be the input feature frame at time t. This feature is first frame-wise concatenated with the d-vector $\mathbf{e}$, then fed into an LSTM network [24] followed by a fully connected neural network to produce a mask $\mathbf{y}^{(t)}$ (see Fig. 1a):

$$\mathbf{y}^{(t)} = \mathrm{FC} \circ \mathrm{LSTM}(\mathrm{Concat}(\mathbf{x}^{(t)}, \mathbf{e})). \tag{1}$$

At runtime, the mask $\mathbf{y}^{(t)}$ is element-wise multiplied to the input $\mathbf{x}^{(t)}$ to produce the final enhanced features.

2.2. Attentive embedding selection

In the multi-user scenario, we have multiple enrolled speaker embeddings $\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_N$. First, we use another small LSTM-based neural network which we refer to as the *PreNet* to compute a *key vector* $\mathbf{k}^{(t)}$ for each frame:

$$\mathbf{k}^{(t)} = \mathrm{PreNet}(\mathbf{x}^{(t)}). \tag{2}$$

We concatenate this key vector with each speaker embedding, and feed it as an input to a fully connected neural network which we refer to as the *ScorerNet* to produce attention scores $s_i^{(t)} \in \mathbb{R}$:

$$s_i^{(t)} = \mathrm{ScorerNet}(\mathrm{Concat}(\mathbf{k}^{(t)}, \mathbf{e}_i)). \tag{3}$$

We use a softmax layer to convert these scores into attention weights $\alpha_i^{(t)} > 0$:

$$\alpha_i^{(t)} = \frac{\exp{(s_i^{(t)})}}{\sum_{j=1}^{N} \exp{(s_j^{(t)})}}. \tag{4}$$

And finally, the attentive speaker embedding $\mathbf{e}_{\text{att}}^{(t)}$ at time t is the weighted sum of all enrolled speaker embeddings:

$$\mathbf{e}_{\text{att}}^{(t)} = \sum_{i=1}^{N} \alpha_i^{(t)} \cdot \mathbf{e}_i. \tag{5}$$

With the attentive speaker embedding, the rest of multi-user VoiceFilter-Lite is similar to the original VoiceFilter-Lite as described in Eq. 1 (see Fig. 1b):

$$\mathbf{y}^{(t)} = \mathrm{FC} \circ \mathrm{LSTM}(\mathrm{Concat}(\mathbf{x}^{(t)}, \mathbf{e}_{\text{att}}^{(t)})). \tag{6}$$

2.3. Loss function

The original VoiceFilter-Lite model is trained with two losses [10]:

1. L_{asym}: an asymmetric L2 loss for signal reconstruction;
2. L_{noise}: a noise type prediction loss for adaptive suppression at runtime.

For multi-user VoiceFilter-Lite, we introduce a new loss that measures how well the attentive embedding selection module attends to the correct target embedding:

$$L_{\text{att}} = \sum_t ||\mathbf{e}_{\text{att}}^{(t)} - \mathbf{e}_{\text{tar}}||^2, \tag{7}$$

where $\mathbf{e}_{\text{tar}} \in \{\mathbf{e}_1, \cdots, \mathbf{e}_N\}$ is the target speaker embedding. The final loss for multi-user VoiceFilter-Lite training is then a linear combination of L_{asym}, L_{noise}, and L_{att}. We use the Adam optimizer [25] to optimize our loss function.

2.4. Design philosophy

2.4.1. PreNet

The purpose of the PreNet is to produce a key vector that represents the voice characteristics of the input features, similar to a speaker recognition neural network. Since both the PreNet and the main VoiceFilter-Lite network are based on low-level input features, it's possible to share common low-level layers between these two networks, such as 1-D convolutional layers.

2.4.2. Permutation invariance

The softmax attention mechanism in Eq. 4 and Eq. 5 guarantees that the inference and training process of the model is invariant to permutations of the speaker embeddings. This is critical for efficient training.

An alternative solution we also considered is to directly concatenate all the enrolled speaker embeddings into a single super embedding:

$$\mathbf{e}_{\text{sup}} = \mathrm{Concat}(\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_N), \tag{8}$$

and replace the embedding in Eq. 1 by this super embedding instead of an attentive embedding. This solution has several drawbacks compared with the attentive embedding solution:

1. Because $\mathbf{e}_{\text{sup}}$ is a fixed dimension vector, it requires a fixed number of enrolled speakers N.
2. It is not permutation invariant. In another word, changing the order of two enrolled speaker embeddings will result in a different super embedding. Thus in the training process, each input needs to be transformed to all the $N!$ permutations to make the model robust. This unnecessarily makes the learning problem much harder.

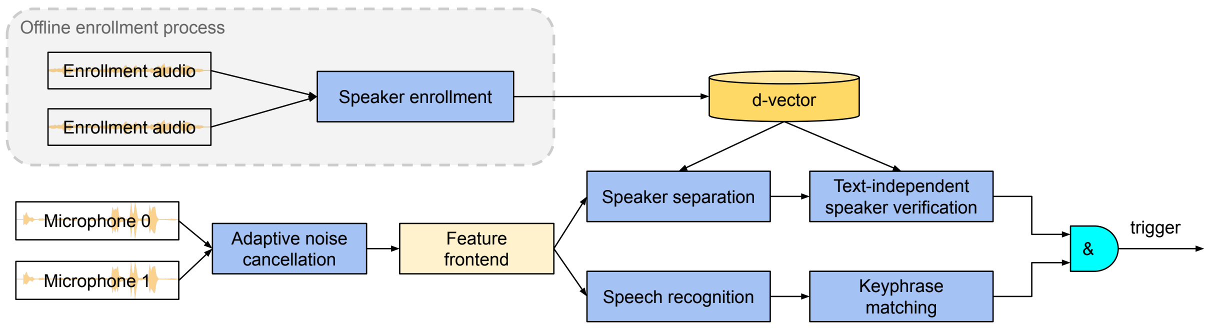


Fig. 2. Diagram of the personalized keyphrase detection system. The d-vector is obtained in a separate offline enrollment process.

2.4.3. *ScorerNet*

In Eq. 3, we use a dedicated small feedforward neural network *ScorerNet* to compute the attention scores. We prove that a simple cosine similarity (or dot-product) attention mechanism cannot be used for this problem.

Proof. Assume we can use a simple cosine similarity attention mechanism:

$$s_i^{(t)} = \cos(\mathbf{k}^{(t)}, \mathbf{e}_i). \tag{9}$$

Assume that the input contains speech from the target speaker A and an interference speaker B. Let $\mathbf{e}_A$ and $\mathbf{e}_B$ be the speaker embeddings of speaker A and speaker B, respectively.

In an effectively working model, because the input contains speech from the target speaker A, the attentive embedding must be close to the embedding of speaker A:

$$\mathbf{e}_{\text{att}}^{(t)} \approx \mathbf{e}_A, \tag{10}$$

thus

$$\cos(\mathbf{k}^{(t)}, \mathbf{e}_A) \approx 1. \tag{11}$$

Because the input also contains speech from speaker B, according to symmetry, we must also have:

$$\cos(\mathbf{k}^{(t)}, \mathbf{e}_B) \approx 1. \tag{12}$$

From Eq. 11 and Eq. 12, we can easily infer that:

$$\cos(\mathbf{e}_A, \mathbf{e}_B) \approx 1. \tag{13}$$

However, because speaker A and speaker B are random different speakers, Eq. 13 apparently does not hold. Thus the hypothesis that a simple cosine similarity attention mechanism can be used is incorrect. $\square$

2.4.4. *Practical considerations in TFLite implementation*

Although the multi-user VoiceFilter-Lite model supports an arbitrary number of enrolled speaker embeddings as side input, when implementing the model in TFLite, there are additional constraints. As TFLite currently does not support inputs with an unknown dimension, we have to pre-define a maximal number of enrolled speaker embeddings, *i.e.* N in our implementation.

At runtime, when the actual number of enrolled speakers is smaller than N, we use **all-zero vectors** as the embeddings of the missing speakers. At training time, we also randomly replace some actual speaker embeddings by all-zero vectors to simulate this scenario. As all valid speaker embeddings are $\mathcal{L}2$-normalized unit-length vectors, an all-zero vector will not conflict with any actual speaker in the embedding space.

3. SYSTEM

3.1. Personalized keyphrase detection

We use VoiceFilter-Lite as part of our streaming personalized keyphrase detection system, which is illustrated in Fig. 2. This system can be easily customized to accurately detect any phrase composed of words from a large vocabulary. The system is implemented with an end-to-end trained ASR model [26] and a text-independent speaker verification model [22]. To address the challenge of detecting keyphrases under various noisy conditions, VoiceFilter-Lite is used as a speaker separation module in the feature frontend of the speaker verification model. In addition, an adaptive noise cancellation (ANC) algorithm [27, 28] is included in the feature frontend of both ASR and speaker separation, which exploit cross-microphone noise coherence to reduce background noise. For further details, we refer the readers to [11].

3.2. Feature frontend

In our experiment, a shared feature frontend is used for all speech models including speech recognition, speaker recognition, and VoiceFilter-Lite. This frontend first applies automatic gain control [29] to the input audio, then extracts 32ms-long Hanning-windowed frames with a step of 10ms. For each frame, 128-dimensional log Mel-filterbank energies (LFBE) are computed in the range between 125Hz and 7500Hz. These filterbank energies are then stacked by 4 frames and subsampled by 3 frames, resulting in final features of 512 dimensions with a frame rate of 30ms.

3.3. Speaker recognition

The speaker embeddings, *a.k.a.* d-vectors, are computed using a text-independent speaker recognition model trained with the generalized end-to-end extended-set softmax loss [22, 23]. Most of our training data are from a vendor collected multi-language speech query dataset covering 37 locales. We also added public datasets including LibriVox, CN-Celeb [30], TIMIT [31], and VCTK [32] to the training data for domain robustness. Multi-style training (MTR) [33, 34, 35] with SNR ranging from 3dB to 15dB is applied during the training process for noise robustness. The speaker recognition model has 3 LSTM layers each with 768 nodes and a projection size of 256. The output of the last LSTM layer is then linearly transformed to the final 256-dimension d-vector.

Table 1. Word Error Rate (%) of streaming ASR system with different VoiceFilter-Lite (VFL) models and different number of enrolled users. The ASR model is trained on multi-domain data, and evaluated on vendor-collected speech queries.

Model	Enrolled users	Clean	Non-speech noise (1∼10 dB)		Speech noise (1∼10 dB)		Size
			Additive	Reverb	Additive	Reverb	
No VFL		15.3	21.1	29.1	56.0	54.5	N/A
Single-user VFL	1	15.3	21.1	29.2	28.8	37.3	2.2 MB
Multi-user VFL	1	15.2	21.1	29.1	32.8	39.1	2.8 MB
	2	15.3	21.2	29.2	34.6	40.6	
	3	15.4	21.2	29.1	35.5	41.5	
	4	15.4	21.2	29.2	36.6	41.8	

3.4. VoiceFilter-Lite

The VoiceFilter-Lite model has 3 LSTM layers, each with 256 nodes, and a final fully connected layer with sigmoid activation. It also has another two feedforward layers, each with 64 nodes for predicting the noise type. In the multi-user setup, the PreNet has 3 LSTM layers, each with 128 nodes; the ScorerNet has two feedforward layers, each with 128 nodes. The new layers in PreNet and ScorerNet increased the quantized [36, 37] model size from the original 2.2MB to 2.8MB.

The training data of the VoiceFilter-Lite model consist of: (1) The LibriSpeech training set [38]; and (2) a vendor-collected dataset of realistic English speech queries. For more details on VoiceFilter-Lite, please refer to [10] and [11].

4. EXPERIMENTS

4.1. Multi-talker speech recognition

Our first group of experiments focuses on using VoiceFilter-Lite to improve streaming on-device ASR, especially for overlapped speech. Similar evaluations had been done in [10] for single-user cases. We use Word Error Rate (WER) as our metrics of ASR.

4.1.1. Datasets

For easier comparisons with previous work, instead of using the ASR model from the keyphrase detection system, here we use the same ASR model as the one described in [10]. This is an on-device streaming RNN-transducer model [39, 26] trained with data from multiple domains including YouTube and anonymized voice search. We evaluate this ASR model on a vendor-collected dataset of realistic speech queries, which consists of about 20,000 utterances from 230 speakers.

During evaluation, we apply different noise sources and room configurations to the data. We use “Clean” to denote the original non-noisified data, although they could be quite noisy already. The non-speech noise source consists of ambient noises recorded in cafes, vehicles, and quiet environments, as well as audio clips of music and sound effects downloaded from Getty Images[1]. The speech noise source is a distinct development set without overlapping speakers from the testing set. We evaluate with two types of room conditions: “Additive” means directly adding the noise waveform to the clean waveform; “Reverb” consists of 3 million convolutional room impulse responses (RIR) generated by a room simulator [35]. Each noise source was then applied to each utterance with an SNR that was drawn from a uniform distribution from 1dB to 10dB for both additive and reverberant conditions.

[1]https://www.gettyimages.com/about-music

4.1.2. Results

The evaluation results are shown in Table 1. From the table, we can see that multi-user VoiceFilter-Lite has similar performance as single-user VoiceFilter-Lite: under clean and non-speech noise conditions, there is no degradation in ASR performance; whereas under speech noise conditions, WER is significantly reduced. Unlike single-user VoiceFilter-Lite, multi-user VoiceFilter-Lite is able to work when there are multiple enrolled users. However, we observed that increasing the number of enrolled users decreased the WER improvement under speech noise conditions. This is because selecting the correct enrolled speaker in the presence of speech background noise is a hard problem.

4.1.3. Other observations

The PreNet and ScorerNet modules in Fig. 1b can be viewed as a student network that attempts to distill some knowledge [40] from the speaker recognition network described in Section 3.3. However, VoiceFilter-Lite applies to each frame of the input, but it can be very difficult to recognize the correct speaker on early frames of the input. During the training process, we observed that the values of most of the attention weights $\alpha_i^{(t)}$ are around $1/N$, suggesting that the attention mechanism learns to utilize all user embeddings by averaging them.

4.2. Multi-talker speaker verification

Our second group of experiments focuses on addressing the multi-talker speaker verification challenge, especially for the *multi-user multi-talker* case. For example, when both speaker A and speaker B enroll their voices on the device (multi-user), and at runtime, speaker A and speaker C speaks at the same time (multi-talker), we expect the speaker verification system to accept the input because it contains speech from one of the enrolled users (speaker A).

We evaluate the standard speaker verification task under various noise conditions with: (1) no VoiceFilter-Lite model; (2) single-user VoiceFilter-Lite model; (3) multi-user VoiceFilter-Lite model for single enrolled user; (4) multi-user VoiceFilter-Lite model for multiple enrolled users. Similar to the experiments in Section 4.1, the noise source can be either non-speech noise or an interfering speaker, and the room condition can be either additive or reverberant to simulate both near-field and far-field devices. The speaker verification model being evaluated is the same model which we used to generate the d-vectors for VoiceFilter-Lite, as described in Section 3.3.

Table 2. Equal Error Rate (%) of our speaker verification system with different VoiceFilter-Lite (VFL) models and different number of enrolled users. Note that the number of users in this table is only for VFL, not for speaker verification. Speaker verification uses a predefined list of enrollment utterances.

Noise source	Room	SNR (dB)	EER (%)					
			No VFL	Single-user VFL	Multi-user VFL			
				1 user	1 user	2 users	3 users	4 users
Clean			0.71	0.71	0.71	0.72	0.73	0.73
Non-speech	Additive	-5	5.05	4.94	4.98	5.02	5.02	5.02
		0	2.20	2.18	2.20	2.22	2.22	2.22
		5	1.47	1.47	1.48	1.49	1.50	1.51
	Reverb	-5	7.76	7.63	7.67	7.71	7.65	7.73
		0	3.66	3.58	3.64	3.64	3.66	3.67
		5	2.09	2.05	2.06	2.08	2.11	2.09
Speech	Additive	-5	12.27	3.61	6.36	7.97	9.09	9.75
		0	8.21	2.18	3.31	4.06	4.58	5.00
		5	5.15	1.51	1.98	2.27	2.44	2.57
	Reverb	-5	17.08	6.15	9.56	11.61	12.54	13.35
		0	10.87	3.50	5.07	6.11	6.69	7.18
		5	6.42	2.21	2.90	3.33	3.55	3.73

4.2.1. Datasets

For this speaker verification evaluation, we use a vendor-provided English speech query dataset. The enrollment list comprises 8,069 utterances from 1,434 speakers, while the test list comprises 194,890 utterances from 1,241 speakers. The interfering speech are from a separate English dev-set consisting of 220,092 utterances from 958 speakers. The non-speech noise and room impulse responses are the same as described in Section 4.1.1.

4.2.2. Results

The evaluation results are shown in Table 2. From the table, we see that for clean and non-speech noise conditions, the EER of the speaker verification system does not change much when we apply single-user or multi-user VoiceFilter-Lite in the feature frontend.

When there is speech noise, single-user VoiceFilter-Lite provides the biggest improvement. The improvement of multi-user VoiceFilter-Lite is smaller than single-user model, but still quite significant compared to the no VoiceFilter-Lite case. Similar to what we previously observed in Section 4.1.2, when the number of enrolled users increases, the performance of multi-user VoiceFilter-Lite degrades as the problem becomes harder.

4.3. Keyphrase detection in the presence of ambient noise

We previously demonstrated that one application of speaker separation using VoiceFilter-Lite is to reduce false rejects in a personalized keyphrase detection system when the background contains speech noise [11]. Specifically, by placing VoiceFilter-Lite in the feature frontend of speaker verification, we can mitigate speaker identification errors due to interfering speech, and as a result reduce false rejections. Thus, in our last group of experiments, we evaluate the overall performance of the personalized keyphrase detection system with multiple enrolled speakers under various background noise conditions. Specifically, we evaluate based on the following two metrics: (1) the number of false acceptance per hour (FA/h), which measures how many keyphrases are incorrectly accepted by the system; and (2) the false rejection rate (FRR), which measures the percentage of true keyphrases that are ignored by the system.

4.3.1. Datasets

To evaluate FA/h, we used a dataset consisting of 156 hours of English speech from curated and hand-annotated YouTube videos [41]. This dataset is designed to mimic television/radio noise, and contains no true keyphrases. As such, phrases in this dataset that trigger the detection system are considered false accepts.

To evaluate FRR, we used a set of vendor-provided keyphrases consisting of 61,555 utterances from 250 speakers with an average of 240 utterances per speaker. This dataset contained commonly used keyphrases such as “remind me to set an alarm”, “turn off the lights”, and “set a timer”.

For both evaluations, we first selected four speakers at random from the vendor-provided dataset. Next, for each speaker, we generated d-vectors from four standard enrollment utterances (*e.g.* “Hey Google, remind me to water my plants”). The number of speakers enrolled however varied depending on the model. For example, to test a 4-user VoiceFilter-Lite, we enrolled all four speakers, while to test the 1-user VoiceFilter-Lite model, we enrolled just a single speaker. FRR evaluation was performed on utterances from all four speakers, such that all models were evaluated on the same number of utterances. We repeated this process for all 4-speaker combinations in our dataset. In order to mimic ambient conditions, we augmented the dataset with either speech or non-speech background noise with reverberation at three different SNR levels using MTR. The “Clean” condition in Table 3 refers to data that is not augmented.

4.3.2. Results

The overall end-to-end performance of our keyphrase detection system is shown in Table 3. We observed that including speaker verification with either a single-user or a multi-user VoiceFilter-Lite model with one enrolled user significantly decreased FA/h from 0.275 to 0.00985 (rel. **96.4%**). With more than one enrolled speaker, there is a greater likelihood of falsely identifying the speaker on the YouTube negative dataset, leading to a marginal increase in the FA/h to 0.0346 in the 4-user case. Despite this increase, speaker verification with VoiceFilter-Lite in its feature frontend is sufficient to significantly reduce false triggering.

As we have demonstrated previously [11], adding speaker verification to the keyphrase detection system increased FRR by 30.8%

Table 3. End-to-end performance of our keyphrase detection on a variety of datasets with single- and multi-user VoiceFilter-Lite (VFL) models. Only the reverberant room condition is considered. The number of users are the number of enrolled speakers.

Noise source	**SNR (dB)**	**No Speaker Verification**	**No VFL**		**Single-user VFL**	**Multi-user VFL**			
			1 user	**4 users**	**1 user**	**1 user**	**2 users**	**3 users**	**4 users**
Vendor-provided keyphrase queries False Rejection Rate (%)									
Clean		4.23	4.23	4.36	4.23	4.23	4.31	4.42	4.42
Non-speech	-5	21.8	23.6	24.6	23.6	23.8	24.1	24.5	24.5
	0	7.62	8.16	8.58	8.16	8.16	8.22	8.52	8.52
	5	3.63	3.72	3.90	3.72	3.75	3.84	3.84	3.84
Speech	-5	82.1	93.7	94.1	82.1	83.5	84.1	87.4	90.4
	0	60.8	78.5	79.5	60.8	61.1	63.8	70.4	71.8
	5	22.6	28.4	29.5	22.6	22.8	23.1	23.4	23.8
YouTube (no queries) False Acceptance / hour									
		0.275	0.00985	0.0346	0.00985	0.00985	0.0197	0.0328	0.0346

in the multi-talker and 7.09% in the non-speech case (SNR = 0dB). This was primarily due to incorrect speaker verification, rather than incorrect speech recognition. This increase in FRR was larger when four speakers were enrolled. Adding a single-user VoiceFilter-Lite model to the feature frontend of speaker verification reduced the FRR from 79.5% to 60.8%, resulting in a **23.5%** reduction in FRR in the SNR = 0dB multi-talker case relative to the model with only speaker verification. Similarly, when we enroll only one user, the multi-user VoiceFilter-Lite model reduces FRR to 61.1% (rel. **23.1%**). This suggests that the single-user and a multi-user VoiceFilter-Lite model with a single enrolled user perform similarly under these conditions. Furthermore, we observe that both the single-user and multi-user VoiceFilter-Lite models perform similarly in the presence of non-speech background noise.

Interestingly, as we enroll more users with the multi-user VoiceFilter-Lite model, we notice an increase in FRR relative to the single-user VoiceFilter-Lite model for both speech and non-speech noise conditions. This is because the attention mechanism now has to select between multiple speakers in the presence of background noise, which is a fundamentally difficult problem especially if the speakers have similar vocal patterns. However, despite this degradation in performance, the multi-user VoiceFilter-Lite with 4 enrolled users has a lower FRR (rel. **9.68%**, SNR = 0dB speech case) than the 4-user system without VoiceFilter-Lite.

Altogether, we demonstrate that by using an attention mechanism to select between N user embeddings, we can extend the benefit of having a VoiceFilter-Lite speaker separation model in the speaker verification feature frontend to multiple enrolled users. This system will greatly reduce both false triggering and false rejections in the presence of ambient noise.

5. CONCLUSIONS

In this paper, we describe a generic solution to allow speaker-conditioned speech models to support scenarios where multiple users have enrolled their voices. This is a very common use case for shared devices, such as smart home speakers. The core idea is to build a permutation-invariant attentive speaker embedding from all enrolled speaker embeddings. We implemented this idea for VoiceFilter-Lite, a streaming target voice separation model for on-device ASR and speaker verification. Experiments show that our multi-user VoiceFilter-Lite model is able to significantly reduce multi-talker speech recognition and speaker verification errors with up to four enrolled users. At the same time, the improvement becomes less when the number of enrolled users increases, as the problem also becomes more difficult.

One future research direction would be to explore alternative implementations of the attention mechanism to better match the multi-user VoiceFilter-Lite performance with single-user models. Besides, the same multi-user solution for VoiceFilter-Lite can be easily used for other speaker-conditioned speech models, such as personal VAD and personalized ASR.

6. REFERENCES

[1] Natasha Jensen, "More ways to fine tune Google Assistant for you," Google Assistant Blog, April 2020.

[2] Quan Wang and Ignacio Lopez Moreno, "Version control of speaker recognition systems," *arXiv preprint arXiv:2007.12069*, 2020.

[3] Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *ICASSP*. IEEE, 2017, pp. 241–245.

[4] Jun Wang, Jie Chen, Dan Su, Lianwu Chen, Meng Yu, Yanmin Qian, and Dong Yu, "Deep extractor network for target speaker recovery from single channel speech mixtures," in *Proc. Interspeech*, 2018, pp. 307–311.

[5] Katerina Zmolikova, Marc Delcroix, Keisuke Kinoshita, Takuya Higuchi, Atsunori Ogawa, and Tomohiro Nakatani, "Speaker-aware neural network based beamformer for speaker extraction in speech mixtures," in *Proc. Interspeech*, 2017.

[6] Kateřina Žmolíková, Marc Delcroix, Keisuke Kinoshita, Takuya Higuchi, Atsunori Ogawa, and Tomohiro Nakatani, "Learning speaker representation for neural network based multichannel speaker extraction," in *ASRU workshop*. IEEE, 2017, pp. 8–15.

[7] Marc Delcroix, Katerina Zmolikova, Keisuke Kinoshita, Atsunori Ogawa, and Tomohiro Nakatani, "Single channel target speaker extraction and recognition with speaker beam," in *ICASSP*. IEEE, 2018, pp. 5554–5558.

[8] Quan Wang, Hannah Muckenhirn, Kevin Wilson, Prashant Sridhar, Zelin Wu, John R. Hershey, Rif A. Saurous, Ron J. Weiss, Ye Jia, and Ignacio Lopez Moreno, "VoiceFilter: Targeted voice separation by speaker-conditioned spectrogram masking," in *Proc. Interspeech*, 2019, pp. 2728–2732.

[9] Chenglin Xu, Wei Rao, Eng Siong Chng, and Haizhou Li, "Spex: Multi-scale time domain speaker extraction network," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1370–1384, 2020.

[10] Quan Wang, Ignacio Lopez Moreno, Mert Saglam, Kevin Wilson, Alan Chiao, Renjie Liu, Yanzhang He, Wei Li, Jason Pelecanos, Marily Nika, and Alexander Gruenstein, "VoiceFilter-Lite: Streaming targeted voice separation for on-device speech recognition," in *Proc. Interspeech*, 2020, pp. 2677–2681.

[11] Rajeev Rikhye, Quan Wang, Qiao Liang, Yanzhang He, Ding Zhao, Arun Narayanan, Ian McGraw, et al., "Personalized keyphrase detection using speaker and environment information," in *Proc. Interspeech*, 2021.

[12] Shaojin Ding, Quan Wang, Shuo-yiin Chang, Li Wan, and Ignacio Lopez Moreno, "Personal VAD: Speaker-conditioned voice activity detection," in *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, 2020.

[13] Pavel Denisov and Ngoc Thang Vu, "End-to-end multi-speaker speech recognition using speaker embeddings and transfer learning," in *Proc. Interspeech*, 2019, pp. 4425–4429.

[14] Jiatong Shi, Chunlei Zhang, Chao Weng, Shinji Watanabe, Meng Yu, and Dong Yu, "Improving rnn transducer with target speaker extraction and neural uncertainty estimation," in *ICASSP*. IEEE, 2021, pp. 6908–6912.

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[16] Qiongqiong Wang, Koji Okabe, Kong Aik Lee, Hitoshi Yamamoto, and Takafumi Koshinaka, "Attention mechanism in speaker recognition: What does it learn in deep speaker embedding?," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 1052–1059.

[17] FA Rezaur rahman Chowdhury, Quan Wang, Ignacio Lopez Moreno, and Li Wan, "Attention-based models for text-dependent speaker verification," in *ICASSP*. IEEE, 2018, pp. 5359–5363.

[18] Nguyen Nang An, Nguyen Quang Thanh, and Yanbing Liu, "Deep cnns with self-attention for speaker identification," *IEEE Access*, vol. 7, pp. 85327–85337, 2019.

[19] Chenglin Xu, Wei Rao, Jibin Wu, and Haizhou Li, "Target speaker verification with selective auditory attention for single and multi-talker speech," *arXiv preprint arXiv:2103.16269*, 2021.

[20] Jia Pan, Diyuan Liu, Genshun Wan, Jun Du, Qingfeng Liu, and Zhongfu Ye, "Online speaker adaptation for lvcsr based on attention mechanism," in *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2018, pp. 183–186.

[21] Genshun Wan, Jia Pan, Qingran Wang, Jianqing Gao, and Zhongfu Ye, "Speaker adaptive training for speech recognition based on attention-over-attention mechanism.," in *Prod. Interspeech*, 2020, pp. 1251–1255.

[22] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno, "Generalized end-to-end loss for speaker verification," in *ICASSP*. IEEE, 2018, pp. 4879–4883.

[23] Jason Pelecanos, Quan Wang, and Ignacio Lopez Moreno, "Dr-Vectors: Decision residual networks and an improved loss for speaker recognition," in *Proc. Interspeech*, 2021.

[24] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[25] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[26] Yanzhang He, Tara N Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, et al., "Streaming end-to-end speech recognition for mobile devices," in *ICASSP*. IEEE, 2019, pp. 6381–6385.

[27] Bernard Widrow, John R Glover, John M McCool, John Kaunitz, Charles S Williams, Robert H Hearn, James R Zeidler, JR Eugene Dong, and Robert C Goodlin, "Adaptive noise cancelling: Principles and applications," *Proceedings of the IEEE*, vol. 63, no. 12, pp. 1692–1716, 1975.

[28] Yiteng Arden Huang, Turaj Z Shabestary, and Alexander Gruenstein, "Hotword cleaner: dual-microphone adaptive noise cancellation with deferred filter coefficients for robust keyword spotting," in *ICASSP*. IEEE, 2019, pp. 6346–6350.

[29] Rohit Prabhavalkar, Raziel Alvarez, Carolina Parada, Preetum Nakkiran, and Tara N Sainath, "Automatic gain control and

multi-style training for robust small-footprint keyword spotting with deep neural networks," in *ICASSP*. IEEE, 2015, pp. 4704–4708.

[30] Yue Fan, JW Kang, LT Li, KC Li, HL Chen, ST Cheng, PY Zhang, ZY Zhou, YQ Cai, and Dong Wang, "CN-CELEB: a challenging Chinese speaker recognition dataset," in *ICASSP*. IEEE, 2020, pp. 7604–7608.

[31] John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, and David S Pallett, "Darpa TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1," *NASA STI/Recon technical report n*, vol. 93, pp. 27403, 1993.

[32] Junichi Yamagishi, Christophe Veaux, Kirsten MacDonald, et al., "CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit," 2019.

[33] Richard Lippmann, Edward Martin, and D Paul, "Multi-style training for robust isolated-word speech recognition," in *ICASSP*. IEEE, 1987, vol. 12, pp. 705–708.

[34] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L Seltzer, and Sanjeev Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *ICASSP*. IEEE, 2017, pp. 5220–5224.

[35] Chanwoo Kim, Ananya Misra, Kean Chin, Thad Hughes, Arun Narayanan, Tara Sainath, and Michiel Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home," in *Proc. Interspeech*, 2017.

[36] Raziel Alvarez, Rohit Prabhavalkar, and Anton Bakhtin, "On the efficient representation and execution of deep acoustic models," *arXiv preprint arXiv:1607.04683*, 2016.

[37] Yuan Shangguan, Jian Li, Qiao Liang, Raziel Alvarez, and Ian McGraw, "Optimizing speech recognition for the edge," in *Conference on Machine Learning and Systems (MLSys)*, 2020.

[38] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *ICASSP*. IEEE, 2015, pp. 5206–5210.

[39] Alex Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[40] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[41] A. Narayanan, R. Prabhavalkar, C.C. Chiu, D. Rybach, T.N. Sainath, and T. Strohman, "Recognizing long-form speech using streaming end-to-end models," in *ASRU workshop*. IEEE, 2019.