END-TO-END STREAMING MODEL FOR LOW-LATENCY SPEECH ANONYMIZATION

Waris Quamer, Ricardo Gutierrez-Osuna

Department of Computer Science and Engineering, Texas A&M University, United States
{quamer.waris, rgutier}@tamu.edu

2024 IEEE Spoken Language Technology Workshop (SLT) | 979-8-3503-9225-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/SLT61566.2024.10832303

ABSTRACT

Speaker anonymization aims to conceal cues to speaker identity while preserving linguistic content. Current machine learning based approaches require substantial computational resources, hindering real-time streaming applications. To address these concerns, we propose a streaming model that achieves speaker anonymization with low latency. The system is trained in an end-to-end autoencoder fashion using a lightweight content encoder that extracts HuBERT-like information, a pretrained speaker encoder that extract speaker identity, and a variance encoder that injects pitch and energy information. These three disentangled representations are fed to a decoder that re-synthesizes the speech signal. We present evaluation results from two implementations of our system, a full model that achieves a latency of $230ms$, and a lite version (0.1x in size) that further reduces latency to $66ms$ while maintaining state-of-the-art performance in naturalness, intelligibility, and privacy preservation.

***Index Terms*—** speaker anonymization, voice conversion, voice privacy, speech synthesis

1. INTRODUCTION

The task of speaker anonymization is to transform utterances to hide the identity of the speaker (while preserving their linguistic content). Speaker anonymization provides privacy protection and confidentiality in a range of applications, including customer service interactions, voice-operated virtual assistants, legal proceedings, and medical consultations. Moreover, speaker anonymization addresses ethical and responsible use of speech data, aligning with privacy regulations and safeguarding individuals' rights.

Existing machine learning (ML) based approaches to speaker anonymization follow a cascaded automatic speech recognition (ASR) – text-to-speech (TTS) architecture [1, 2]. An ASR module produces a text transcription that is speaker independent but eliminates emotional cues that may otherwise be of use for downstream applications. Moreover, existing systems for speaker anonymization are computationally heavy, operate in a non-streaming fashion, and/or have high latency on CPU devices as opposed to GPUs. For speech anonymization to be used in the field, it must operate at real time (or faster), exhibit low latency, require minimal future context and be compatible with low-resource devices (e.g., smartphones).

To address these needs, we propose an end-to-end streaming model suitable for low-latency speaker anonymization. Our model draws inspiration from neural audio codecs [3, 4] for audio compression in low-resource streaming settings. Our key strategy that enables streaming is to replace traditional non-causal computationally intensive networks (*e.g.*, ASR or self-supervised learning based models) for encoding linguistic content with a lightweight convolutional neural network (CNN) based architecture. Our proposed architecture consists of: (a) a streaming waveform encoder that generates a speaker-independent content representation from waveforms, (b) a pseudo-speaker generator that produces an anonymized speaker representation (i.e., an embedding) from the input speech, (c) a speaker/variance adapter that adds speaker, pitch and energy information to the content representation, and (d) a streaming decoder that consumes the speaker/variance adapted linguistic representation and the corresponding speaker embedding to generate the final anonymized audio waveform. Our system is trained in an auto-encoder fashion, which reconstructs the input *conditioned* on the speaker embeddings generated using pre-trained speaker encoders [5, 6]. During inference, a pseudo-speaker generator produces a target speaker embedding with cosine distance greater than 0.3 from the source embedding, ensuring that the re-synthesized utterance sounds as if a different (i.e., anonymized) speaker had produced it. Additionally, the speaker/variance adapter is used to modulate pitch and energy values to further enhance privacy and control the similarity of the synthesized speech with the source audio. We show that our lightweight convolutional neural network (CNN) based architecture achieves similar performance as traditional content encoders.

We perform experiments on two versions of our model, a *Base* version that can perform real-time streaming synthesis with a latency of $230ms$ and a *Lite* version (having 0.1x the number of parameters) that further reduces latency to $66ms$ while maintaining state-of-the-art performance on naturalness, intelligibility, privacy and speaker identity transfer[1].

[1] https://warisqr007.github.io/demos/stream-anonymization/

2. RELATED WORK

2.1. Voice conversion

Speaker anonymization is closely related to voice conversion (VC). However, whereas VC seeks to transform utterances from a source speaker to match the identity of a (known) target speaker, speaker anonymization only requires that the transformed speech be sufficiently different from the source speaker to conceal their identity.

The first step in conventional VC architectures is to disentangle the linguistic content of an utterance from speaker-specific attributes. As an example, cascaded ASR-TTS architectures [7] use an ASR model to transcribe the input utterance into text, followed by a TTS model that converts the text back into speech –conditioned on a speaker embedding. Variants of this approach replace the ASR module with acoustic models that generate a more fine-grained representation than text, such as phonetic posteriorgrams (PPGs) [8]. Recent approaches have also used information bottlenecks to disentangle linguistic content form speaker identity [9]. A major drawbacks of the latter approach is that information bottlenecks must be carefully designed and are sensitive to the dimension of latent space. Other techniques include instance normalization [10], use of mutual information loss [11], vector quantization [11, 12], and adversarial training [13]. To enable streaming, recent VC methods use a streaming ASR to extract PPGs [14] or streaming ASR sub-encoders [15, 16] to generate linguistic content, and then perform VC through causal architectures that require limited future contexts.

2.2. Speaker anonymization

Speaker anonymization approaches can be broadly divided into two categories: digital signal processing (DSP) and machine learning (ML) based. DSP methods include formant-shifting using McAdams coefficients [17], frequency warping [18], or a series of steps consisting of vocal tract length normalization, McAdams transformation and modulation spectrum smoothing [19]. Additionally, modifications to pitch [20] and speaking rate [21] are used. DSP models are significantly smaller (i.e., fewer parameters) than ML models, which results in efficient and speedy execution. However, the types of global transforms used in DSP methods cannot fully remove speaker-dependent cues, making them vulnerable to ML-based speaker verification systems [1].

ML methods for speaker anonymization follow the conventional VC framework of disentangling linguistic content from speaker identity, but then replace the latter with a speaker embedding that is different (anonymized) from the source. Various methods have been proposed to select this anonymized speaker embedding. For example, Srivastava *et al.* [22] generate anonymized embeddings by randomly selecting N speaker vectors from a pool of speakers farthest from the source, using e.g., cosine distance, whereas Perero-Codosero *et al.* [23] use an autoencoder architecture with an adversarial training module that removes speaker, gender, and accent information. Other approaches use look-up tables [24] or generative adversarial networks [1] to generate pseudo-speakers. Our approach follows the latter: we combines a GAN-based pseudo-speaker generator with a streaming model to enable real-time speaker anonymization with low latency.

3. METHOD

The proposed system is illustrated in Figure 1. Anonymization takes place in two steps, (1) generating a fixed (i.e., off-line) anonymized speaker embedding personalized to the source speaker and, (2) using this fixed anonymized speaker embedding and the streaming speech synthesizer to synthesize anonymized speech that only preserves the linguistic content of the source speech. To generate the anonymized embedding, a reference waveform from the source speaker is passed to the pre-trained speaker encoder, which then produces the source speaker embedding. The pseudo speaker generator receives this source speaker embedding and generates the anonymized version (see Figure 1b). To synthesise speech signals, the content encoder receives streaming chunks of waveform and converts it into a hidden representation z that contains the linguistic content disentangled from the speaker representation. The content information z and the anonymized speaker embedding (generated in the previous step) is passed to the speaker/variance adapter. The speaker/variance adapter, first, conditions the anonymized speaker embedding on the content representation z and then adds pitch and energy values. The decoder receives the output of the speaker/variance adapter and the anonymized speaker embedding to generate the final anonymized waveform.

We train two versions of our proposed system, a *base* and a *lite* version. Below, we describe each component of our system and the training procedure in detail.

3.1. Content encoder

The content encoder consumes the wav signal to predict discrete speech units produced by discretizing the output speech representation from a pretrained HuBERT model [25] into one of N codewords or pseudo-labels [26]. Our content encoder architecture follows that of HiFiGAN [27], except all transposed convolutions in HiFiGAN are replaced with strided causal convolutions to downsample the input waveform. Additionally, to support streaming applications, we replace all vanilla CNN layers in HiFiGAN with causal CNNs so that the prediction only considers the past context and does not rely on future audio frames. For both versions of our model (base and lite), we use downsampling rates of $[2, 2, 4, 4, 5]$. The residual blocks have kernel sizes as $[3, 7, 11]$ with dilation rates as $[[[1, 1], [3, 1], [5, 1]] * 3]$ (please

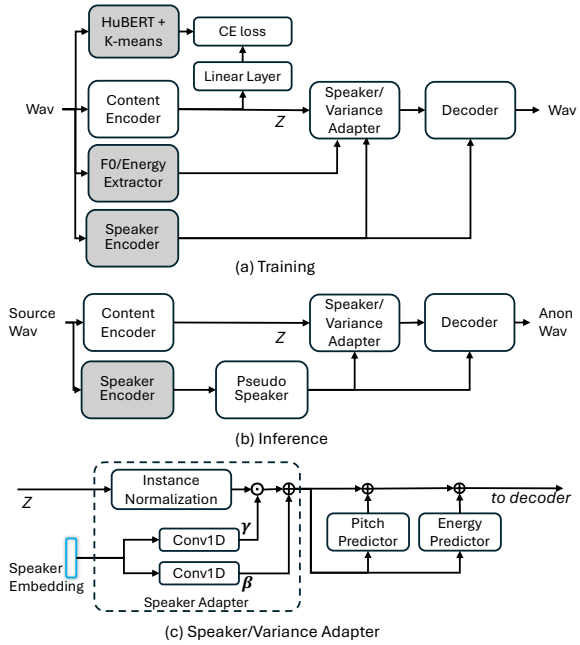

Fig. 1. Block diagram of the proposed anonymization system. (a) training workflow (b) inference workflow (c) speaker/variance adapter.

refer [27] for details). The difference between *base* and the *lite* version is the dimension of the hidden representation z (the output of the encoder): 512 dimensions for the *base* version and 128 dimensions for the *lite* version.

3.2. Speaker encoder and pseudo-speaker generator

Speaker verification or classification systems generally use speaker embeddings to represent the characteristics or timbre of a speaker's voice. Widely used speaker encoders include the GE2E model [28], X-vectors [6] and ECAPA-TDNN [5]. Our system concatenates embeddings generated from X-vectors and ECAPA-TDNN models, since these two models have been shown to complement each other [1].

To perform speaker anonymization, we use a pseudo-speaker generator that takes the original speaker embedding as input and outputs an artificially generated speaker embedding such that the generated anonymized speaker embedding has a cosine distance greater than 0.3 as compared with the original speaker embedding. Our pseudo-speaker generator follows a GAN-based architecture [1] and is trained separately. The generator is trained to receive a random vector sampled from a standard normal distribution $N(0, 1)$ as input and output a vector of the same shape as the original speaker embedding. The discriminator is trained to discriminate w.r.t the quadratic Wasserstein distance and transport cost [29] between the artificial and the original speaker embeddings.

3.3. Speaker/Variance adapter

The speaker/variance adapter aims to add speaker, pitch, and energy (i.e., variance) information to the speaker-independent content representation and provides a way to control them [30]. The speaker/variance adapter consists of three modules: (a) speaker adapter, (b) pitch predictor, and (c) energy predictor (see Figure 1c).

The speaker adapter conditions the speaker embedding on the content representation z, and passes it to the pitch and energy predictors. The speaker adapter is based on adaptive instance normalization (adaIN) [31] and feature-wise linear modulation (FiLM) [32]. The conditioning goes as follows. First, we apply instance normalization to the input feature representation, and then transform it with scale and bias parameters learned through two 1D CNNs that take speaker embeddings as input. The use of instance normalization was motivated by a prior work [10] that showed instance normalization being helpful in removing residual speaker information.

Pitch and energy predictor estimate pitch and energy values based on speaker adapted content representation z. During training, we use the ground-truth pitch and energy values to train the pitch and energy predictors. At inference, the output of the pitch and energy predictor are added to the speaker adapted z. The pitch and energy predictors have similar architecture consisting of a 2-layered 1D causal CNNs (kernal size 3) with ReLU activation, followed by layer normalization and dropout layer and an additional 1D CNN (with kernel size 1) to project pitch and energy values on the latent representation.

3.4. Decoder

The decoder follows the same design and training procedure as HiFiGAN [27] and can be seen as a mirror-image of the content encoder. Similar to the content encoder, all vanilla CNNs are replaced with causal CNNs. The decoder receives speaker/variance adapted latent representation along with speaker embedding and directly generates waveform signal without any intermediate mel-spectrogram generation. We additionally adapt the output of each residual block of the decoder to the speaker embedding. In our experiments, we observed that doing so gave better speaker transfer performance. For both versions of $base$ and $lite$ version of our model, we use upsampling rates of $[5, 4, 4, 2, 2]$. The residual blocks have kernel sizes as $[3, 7, 11]$ with dilation rates as $[[[1, 1], [3, 1], [5, 1]] * 3]$.

3.5. Training

The training workflow is described in Figure 1a. The proposed system is trained end-to-end similar to an autoencoder, reconstucting the same waveform at output that fed as input. The content encoder is trained to predict the pseudo-labels generated through a HuBERT/Kmean module using cross-entropy loss. The pitch and energy predictor in the variance

adapter apply mean-squared error loss for pitch and energy prediction. At the output of the decoder, Following the HiFi-GAN architecture [27], we apply a combination of adversarial losses at the output of the decoder, including feature loss, multi-period discriminator loss, multi-scale discriminator loss, multi-resolution STFT loss [33] and mel-spectrogram reconstruction loss. These discriminators have similar architecture as those in HiFiGAN, including weighting schemes to compute the total decoder loss. The final training loss is the summation of content loss, pitch/energy error loss and decoder losses. We apply a stop-gradient operation to prevent gradient flow (i.e., back-propagation) from the decoder to the encoder, to ensure that the speaker information is not leaked via the content representation. This operation effectively decouples it from the rest of the system; in other words, the encoder and the rest of the system can potentially be trained sequentially as two independent modules.

4. EXPERIMENTAL SETUP

We trained our system on the LibriTTS corpus [34] following guidelines for the Voice Privacy Challenge 2022 (VPC22) [35]. All our experimental results are presented on the LibriTTS dev and test set, which were not part of the training. We use a pretrained HuBERT-*base*[2] and extract the the output from its 9th layer. We set the number of cluster centroids to 200. For all our experiments, we use a sampling rate of 16 kHz and batch size of 16 with the AdamW optimizer with a learning rate of $2 * 10^{-4}$ annealed down to 10^{-5} by exponential scheduling. The encoder is first pretrained for 300k steps (for training stability), and then trained together with the decoder for an additional 800k steps. The pretrained speaker encoders were taken from speechbrain [36]. The pseudo speaker embedding generator follows the training procedure described in [1] and trained on VoxCeleb 1 and 2 [37, 38]. All our models are trained using two NVIDIA Tesla V100 GPUs for approximately two weeks.

5. RESULTS

We evaluated our system on a series of subjective and objective measures of synthesis latency, synthesis quality, privacy as well as speaker transfer ability. We compare our results against five baselines: three state-of-the-art VC models (VQMIVC[3] [11], QuickVC[4] [39], and DiffVC[5] [40]) and two speaker anonymization models[6], a DSP-based model [17] (baseline B2 from VPC22) and a ML-based model (to which we refer as B3) that uses a transformer-based ASR and a Fastspeech2-based TTS with a WGAN-based anonymizer [1]. The five baseline models are trained on the same dataset as our proposed system, and we use their pretrained checkpoints obtained from their corresponding github repositories. We could not find any open-source streaming speech synthesis model and hence were unable to include them as baselines. We evaluate our model using the same Libri-TTS evaluation split as VPC22. For the VC baselines we randomly select a speaker from the CMU Arctic corpus [41] as the target speaker.

5.1. Synthesis Latency

Our pretrained HuBERT model produces speech frames at $50Hz$, so the smallest chunk size that our model can process is $20ms$. In this section, we present the synthesis latency and the real-time factors (RTF) for the base and lite versions for our model, for various chunk sizes between $20ms$ and $140ms$ on both CPU and GPU devices. Latency is defined as the sum of the chunk size and the average time the model takes to synthesize that chunk. RTF is the ratio of the system's average processing time to the chunk size. For a system to be real-time, the latency should be less than twice the chunk size, meaning RTF should be less than 1. Results are summarized in Table 1. On GPU, our base version can operate in real-time for the chunk size of $40ms$ with a latency of $64ms$, while on CPU the base model can be real-time for chunk size of $120ms$ with a latency of $230ms$. In case of the lite version, the model is real-time for chunk size of $20ms$ with a latency of $38ms$ on GPU and can operate in real-time for $40ms$ with latency of $66ms$ on the CPU. For our test set of experiments we set the chunk size of $120ms$ and $40ms$ for the base and lite versions respectively.

5.2. Synthesis Quality

We use DNSMOS [42] as an objective measure of naturalness for our experiments. DNSMOS provides three ratings for quality of speech (SIG), noise (BAK), and overall (OVRL). Additionally, we assess the intelligibility of synthesized speech through Word Error Rate (WER). We calculated WER using an ASR consisting of a CRDNN based acoustic model and a transformer-based language model that uses CTC and attention decoders. Table 2, summarizes results for the five baselines and the proposed systems. In terms of DNSMOS, our models achieve comparable ratings as Diff-VC, QuickVC, and B3 across the three measures, and comparable or better than the source speech. In terms of intelligibility, our systems achieve comparable WER to that of B3, and superior to the rest, even though our models operate in a causal fashion with far more limited context.

We verified the synthesis quality of our two models through listening tests on Amazon Mechanical Turk (AMT). Namely, participants (N=20) were asked to rate the acoustic quality of utterances using a standard 5-point scale mean

[2]https://github.com/facebookresearch/fairseq

[3]https://github.com/Wendison/VQMIVC

[4]https://github.com/quickvc/QuickVC-VoiceConversion

[5]https://github.com/huawei-noah/Speech-Backbones/tree/main/DiffVC

[6]https://github.com/Voice-Privacy-Challenge/Voice-Privacy-Challenge-2024

Table 1. Synthesis latency and real-time factors (RTF) on CPU and GPU devices.

Chunk Size	**CPU** (ms)				**GPU** (ms)			
(ms)	**base**		**lite**		**base**		**lite**	
	latency	RTF	latency	RTF	latency	RTF	latency	RTF
20	83.82	3.19	43.01	1.15	44.61	1.23	38.36	0.92
40	115.23	1.88	66.27	0.66	64.38	0.61	58.70	0.47
60	170.93	1.85	88.03	0.47	84.03	0.40	79.55	0.33
100	230.23	1.30	136.60	0.37	124.19	0.24	118.92	0.19
120	229.75	0.91	159.22	0.33	144.16	0.20	139.41	0.16
140	249.38	0.78	173.49	0.24	163.90	0.17	158.90	0.14

Table 2. DNS MOS, Word Error Rates (WER) and speaker similarity scores (SSS) for baselines and the proposed model.

	SIG	**BAK**	**OVRL**	**WER**	**SSS**
VQMIVC	3.35	3.78	2.98	26.88	0.72
Quick-VC	3.55	4.06	3.28	6.30	0.68
Diff-VC	3.62	4.17	3.40	7.64	0.82
VPC22 B2	2.85	3.44	2.50	12.02	-
B3	3.57	4.03	3.28	4.65	-
base	3.53	3.99	3.31	5.12	0.85
lite	3.48	3.93	3.22	6.47	0.81
Source	3.58	3.99	3.26	2.98	0.89

Table 3. Subjective MOS for naturalness.

	Source	**base**	**lite**
MOS	3.54 ± 0.56	3.57 ± 0.59	3.47 ± 0.62

opinion score (MOS) [rating, speech quality, level of distortion]: [5, excellent, imperceptible] — [4, good, just perceptible but not annoying] — [3, fair, perceptible but slightly annoying] — [2, poor, annoying but not objectionable] — [1, bad, very annoying and objectionable]. Each listener rated utterances synthesized using the base and lite models, as well as original utterances (20 for each). Results are shown in Table 3. Both systems (base and lite) obtained comparable ratings of MOS as the original utterances. We see a difference of 0.1 between the MOS score of base and lite versions, but we didn't find them to be significant. It is noteworthy that while the lite version has 0.1x number of parameters, it achieves nearly the same synthesis quality as the base version.

5.3. Speaker Anonymization

To assess speaker-anonymization performance, we report Equal Error Rate (EER) on the speaker verification model (ASV) in the VoicePrivacy 2024 Challenge github (see section 4). ASV tests are conducted for the following two scenarios, (a) ignorant, where we only anonymize the trial data (O-A), or (b) lazy-informed, where we anonymize both enrollment and trial data but use different target speakers (A-A). Results are shown in Table 4. For both the ignorant and lazy-informed scenario, our models achieves similar performance as B3 and outperforms VPC22 B2. Although our base model performs slightly worse than B3, the differences are not significantly different between them ($p = 0.13$).

Table 4. Equal Error Rate (EER) as a privacy metric. The higher the better.

		VPC22 B2	**B3**	**base**	**lite**
O-A	M	25.10	44.24	43.83	42.57
	F	37.42	47.78	46.87	45.31
A-A	M	11.03	42.63	41.43	39.20
	F	15.03	43.23	42.03	41.16

To corroborate these results, we conducted an AB listening test on AMT. Participants were presented with two audio samples, one from a speaker in the enrollment set, and the second sample from one of three options: (a) a different utterance from the same speaker from the trial set, (b) an utterance from a different speaker from the trial set, or (c) another utterance of the same speaker from the trial set but anonymized through our lite version of the system. Then, participants had to decide if both samples were from the same speaker, and rate the confidence in their decision using a 7-point scale (7: extremely confident; 5: quite a bit confident; 3: somewhat confident; 1: not confident at all). Each listener rated 20 AB pairs per scenario. Results are summarized in Table 5. In settings (a) and (b), listeners could easily identify whether the recording were from the same or different speakers (81 %) with high confidence (5.71). In setting (c), however, the anonymized trial data obtained obtained similar rating as in (b), indicating that proposed system was able to anonymize the trial recordings.

Table 5. Subjective speaker verifiability scores for the proposed model.

speaker	anon	Verifiability	Confidence Rating
same	no	81.5%	5.71
different	no	17.75%	2.50
same	yes	14.5%	2.37

5.4. Speaker Identity transfer

In a final step, we evaluated our models' ability to capture the voice of a target speaker. For this purpose, we used an objective score of speaker similarity based on the cosine similarity between speaker embeddings of the target and the synthesized utterances obtained from a ASV system[7]. We compare our model against the three VC baselines (VQMIVC [11], QuickVC [39], DiffVC [40]) using the same settings as those in section 5.2) to generate VC samples. Results are summarized in the rightmost column of Table 2 (SSS). As a guideline, pairing two utterances from the same speaker yields an average cosine similarity of 0.89. As shown, our base model outperforms the three baselines, achieving cosine similarity that is close to the average within-speaker similarity of 0.89.

6. DISCUSSION

Most existing speaker anonymization methods do not operate in low-latency streaming mode, preventing their use in field operations. In this paper, we present an end-to-end streaming model that operates with low latency and achieves anonymization by mapping speaker embedding into an artificially generated pseudo speaker in a causal fashion (i.e., no future context). The pseudo-speaker generator can produce speaker embeddings that are very close to a real person in the corpus. Although we train the pseudo-speaker generator on a different corpus than the speech anonymization system to guard against this possibility, we could also test if a pseudo-speaker is too close to one on the corpus or outside the space of speakers in the training corpus, and generate new ones until a valid one is generated. While there exists a quality-latency tradeoff, our system can achieve latency as low as $66ms$ while maintaining state-of-the-art naturalness, intelligibility and privacy preservation. Our lite version is roughly 10MB and can potentially be deployed on mobile devices to support real-time field applications. Accent can carry speaker related cues [43] and in future work, we aim to add accent conversion to this pipeline. Other research direction is to add the control of emotion while synthesizing speech signals.

7. ACKNOWLEDGEMENTS

This work was funded by NSF award 619212 and 1623750.

[7]Computed using: https://github.com/resemble-ai/Resemblyzer

8. REFERENCES

[1] Sarina Meyer, Pascal Tilli, Pavel Denisov, Florian Lux, Julia Koch, and Ngoc Thang Vu, "Anonymizing speech with generative adversarial networks to preserve speaker privacy," in *2022 IEEE SLT Workshop*, 2023, pp. 912–919.

[2] Champion Pierre, Anthony Larcher, and Denis Jouvet, "Are disentangled representations all you need to build speaker anonymization systems?," in *Proc. Interspeech*, 2022, pp. 2793–2797.

[3] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM TASLP*, vol. 30, pp. 495–507, 2021.

[4] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[5] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck, "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," in *Proc. INTERSPEECH*, 2020, pp. 3830–3834.

[6] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *Proc. ICASSP*, 2018, pp. 5329–5333.

[7] Wen-Chin Huang, Tomoki Hayashi, Shinji Watanabe, and Tomoki Toda, "The sequence-to-sequence baseline for the voice conversion challenge 2020: Cascading asr and tts," *arXiv preprint arXiv:2010.02434*, 2020.

[8] Songxiang Liu, Yuewen Cao, Disong Wang, Xixin Wu, Xunying Liu, and Helen Meng, "Any-to-many voice conversion with location-relative sequence-to-sequence modeling," *IEEE/ACM TASLP*, vol. 29, pp. 1717–1728, 2021.

[9] Chak Ho Chan, Kaizhi Qian, Yang Zhang, and Mark Hasegawa-Johnson, "Speechsplit2. 0: Unsupervised speech disentanglement for voice conversion without tuning autoencoder bottlenecks," in *Proc. ICASSP*, 2022, pp. 6332–6336.

[10] Yen-Hao Chen, Da-Yi Wu, Tsung-Han Wu, and Hung-yi Lee, "Again-vc: A one-shot voice conversion using activation guidance and adaptive instance normalization," in *Proc. ICASSP*, 2021, pp. 5954–5958.

[11] Disong Wang, Liqun Deng, Yu Ting Yeung, Xiao Chen, Xunying Liu, and Helen Meng, "VQMIVC: Vector

Quantization and Mutual Information-Based Unsupervised Speech Representation Disentanglement for One-Shot Voice Conversion," in *Proc. Interspeech*, 2021, pp. 1344–1348.

[12] Waris Quamer, Anurag Das, and Ricardo Gutierrez-Osuna, "Decoupling Segmental and Prosodic Cues of Non-native Speech through Vector Quantization," in *Proc. INTERSPEECH*, 2023, pp. 2083–2087.

[13] Shaojin Ding, Guanlong Zhao, and Ricardo Gutierrez-Osuna, "Improving the speaker identity of non-parallel many-to-many voice conversion with adversarial speaker recognition.," in *Proc. INTERSPEECH*, 2020.

[14] Tho Nguyen Duc Tran, The Chuong Chu, Vu Hoang, Trung Huu Bui, and Hung Quoc Truong, "An Efficient and High Fidelity Vietnamese Streaming End-to-End Speech Synthesis," in *Proc. Interspeech*, 2022, pp. 466–470.

[15] Haoquan Yang, Liqun Deng, Yu Ting Yeung, Nianzu Zheng, and Yong Xu, "Streamable speech representation disentanglement and multi-level prosody modeling for live one-shot voice conversion.," in *Proc. INTERSPEECH*, 2022, pp. 2578–2582.

[16] Yuanzhe Chen, Ming Tu, Tang Li, Xin Li, Qiuqiang Kong, Jiaxin Li, Zhichao Wang, Qiao Tian, Yuping Wang, and Yuxuan Wang, "Streaming voice conversion via intermediate bottleneck features and non-streaming teacher guidance," in *Proc. ICASSP*, 2023, pp. 1–5.

[17] Jose Patino, Natalia Tomashenko, Massimiliano Todisco, Andreas Nautsch, and Nicholas Evans, "Speaker Anonymisation Using the McAdams Coefficient," in *Proc. Interspeech*, 2021, pp. 1099–1103.

[18] Jianwei Qian, Haohua Du, Jiahui Hou, Linlin Chen, Taeho Jung, and Xiang-Yang Li, "Speech sanitizer: Speech content desensitization and voice anonymization," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 6, pp. 2631–2642, 2019.

[19] Hiroto Kai, Shinnosuke Takamichi, Sayaka Shiota, and Hitoshi Kiya, "Lightweight and irreversible speech pseudonymization based on data-driven optimization of cascaded voice modification modules," *Computer Speech & Language*, vol. 72, pp. 101315, 2022.

[20] Lauri Tavi, Tomi Kinnunen, and Rosa González Hautamäki, "Improving speaker de-identification with functional data analysis of f0 trajectories," *Speech Communication*, vol. 140, pp. 1–10, 2022.

[21] S Pavankumar Dubagunta, Rob JJH van Son, and Mathew Magimai Doss, "Adjustable deterministic pseudonymization of speech," *Computer Speech & Language*, vol. 72, pp. 101284, 2022.

[22] Brij Mohan Lal Srivastava, N. Tomashenko, Xin Wang, Emmanuel Vincent, Junichi Yamagishi, Mohamed Maouche, Aurélien Bellet, and Marc Tommasi, "Design Choices for X-Vector Based Speaker Anonymization," in *Proc. Interspeech*, 2020, pp. 1713–1717.

[23] Juan M Perero-Codosero, Fernando M Espinoza-Cuadros, and Luis A Hernández-Gómez, "X-vector anonymization using autoencoders and adversarial training for preserving speech privacy," *Computer Speech & Language*, vol. 74, pp. 101351, 2022.

[24] Jixun Yao, Qing Wang, Li Zhang, Pengcheng Guo, Yuhao Liang, and Lei Xie, "NWPU-ASLP System for the VoicePrivacy 2022 Challenge," in *Proc. 2nd Symposium on Security and Privacy in Speech Communication*, 2022.

[25] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM TASLP*, vol. 29, pp. 3451–3460, 2021.

[26] Benjamin van Niekerk, Marc-André Carbonneau, Julian Zaïdi, Matthew Baas, Hugo Seuté, and Herman Kamper, "A comparison of discrete and soft speech units for improved voice conversion," in *Proc. ICASSP*, 2022, pp. 6562–6566.

[27] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Proc NeurIPS*, vol. 33, pp. 17022–17033, 2020.

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

[29] Huidong Liu, Xianfeng Gu, and Dimitris Samaras, "Wasserstein gan with quadratic transport cost," in *Proc. CVF*, 2019, pp. 4832–4841.

[30] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," *arXiv preprint arXiv:2006.04558*, 2020.

[31] Xun Huang and Serge Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proc. ICCV*, 2017, pp. 1501–1510.

[32] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville, "Film: Visual reasoning with a general conditioning layer," in *Proc. AAAI*, 2018, vol. 32.

[33] Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *Proc. ICASSP*, 2020, pp. 6199–6203.

[34] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu, "Libritts: A corpus derived from librispeech for text-to-speech," *arXiv preprint arXiv:1904.02882*, 2019.

[35] Natalia Tomashenko, Xin Wang, Xiaoxiao Miao, Hubert Nourtel, Pierre Champion, Massimiliano Todisco, Emmanuel Vincent, Nicholas Evans, Junichi Yamagishi, and Jean-François Bonastre, "The voiceprivacy 2022 challenge evaluation plan," *arXiv preprint arXiv:2203.12468*, 2022.

[36] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, et al., "Speechbrain: A general-purpose speech toolkit," *arXiv preprint arXiv:2106.04624*, 2021.

[37] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Proc. INTERSPEECH*, 2017.

[38] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Proc. INTERSPEECH*, 2018.

[39] Houjian Guo, Chaoran Liu, Carlos Toshinori Ishi, and Hiroshi Ishiguro, "Quickvc: A lightweight vits-based any-to-many voice conversion model using istft for faster conversion," in *Proc. ASRU*, 2023, pp. 1–7.

[40] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, Mikhail Kudinov, and Jiansheng Wei, "Diffusion-based voice conversion with fast maximum likelihood sampling scheme," in *Proc. ICLR*, 2022.

[41] John Kominek and Alan W Black, "The cmu arctic speech databases," in *Fifth ISCA workshop on speech synthesis*, 2004.

[42] Chandan KA Reddy, Vishak Gopal, and Ross Cutler, "Dnsmos: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors," in *Proc. ICASSP*, 2021, pp. 6493–6497.

[43] Anurag Das, Guanlong Zhao, John Levis, Evgeny Chukharev-Hudilainen, and Ricardo Gutierrez-Osuna, "Understanding the effect of voice quality and accent on talker similarity.," in *Proc. INTERSPEECH*, 2020, pp. 1763–1767.