Deep Learning for Audio

Lecture 7

Pavel Severilov

AI Masters

2024

Outline

1. Voice Conversion: basics
2. CycleGAN-VC
3. StarGAN-VC
4. AutoVC

Outline

Voice Conversion: task

- Types:
  - One-to-One
  - Many-to-One
  - Many-to-Many
  - Any-to-Any (Zero Shot)
- Data: parallel/not parallel. Parallel data hard to obtain
- Quality: Naturalness (MOS), Speaker Similarity, Any Other Product Metric

VC old solution

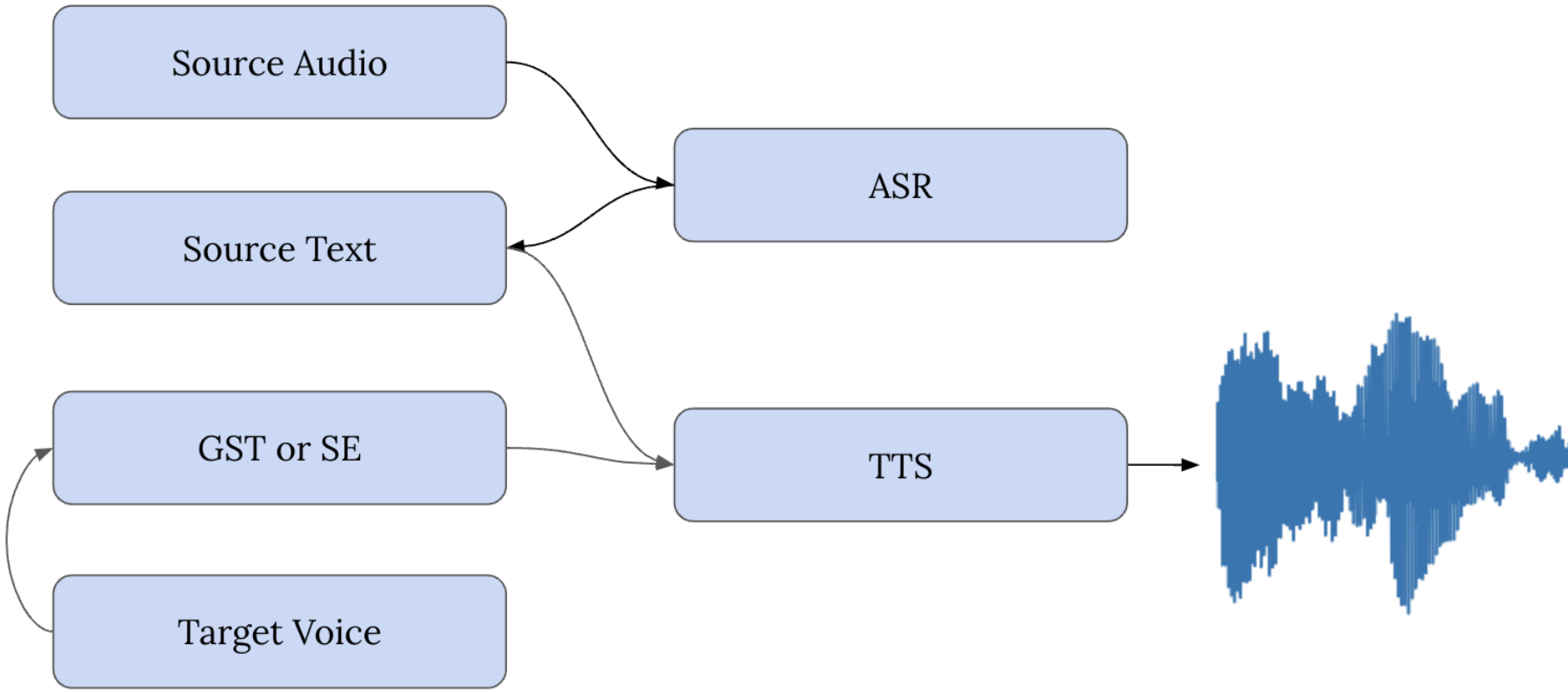


Figure: Straightforward way to solve VC task (has problems)

Outline

CycleGAN-VC

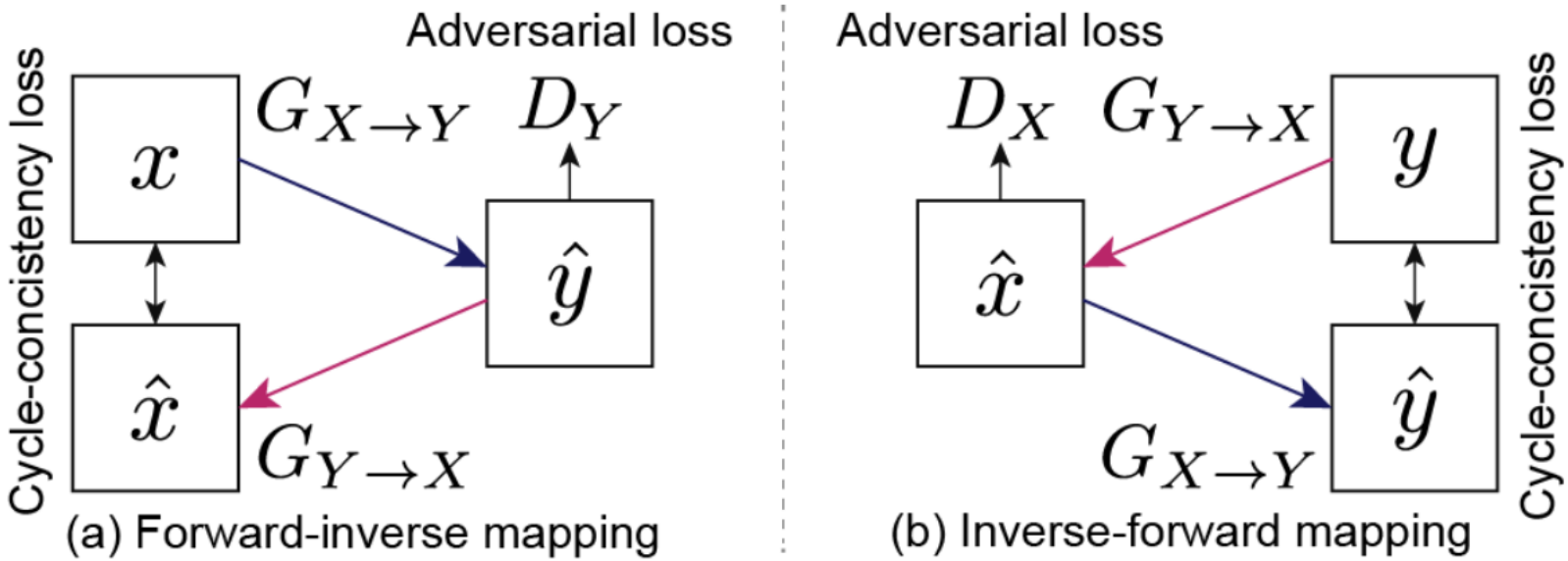


Figure: Adversarial Loss + Cycle-consistency Loss idea for VC

Takuhiro Kaneko et al., Parallel-Data-Free Voice Conversion Using Cycle-Consistent Adversarial Networks, 2017

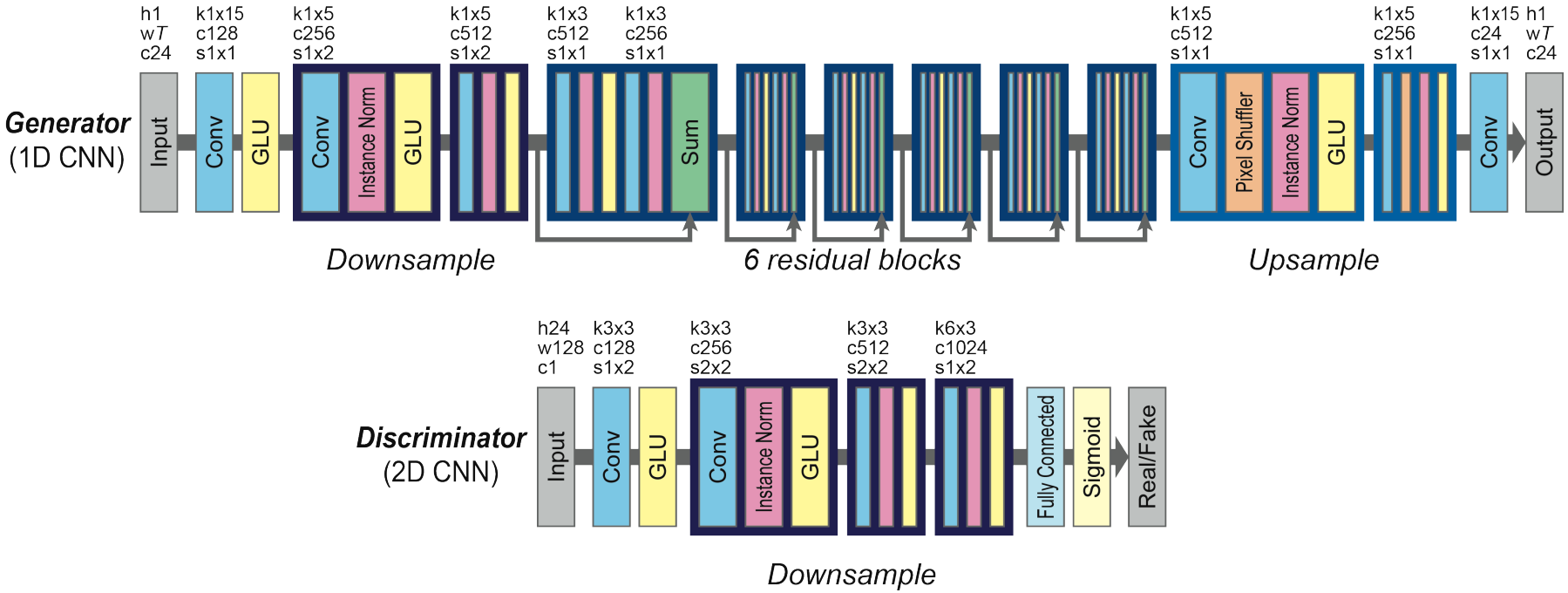
CycleGAN-VC: loss

$$
\begin{aligned}
&\mathcal{L}_{cyc}(G_{X\to Y}, G_{Y\to X}) \\
=&\mathbb{E}_{x\sim P_{\text{Data}}(x)}[\|G_{Y\to X}(G_{X\to Y}(x)) - x\|_1] \\
+&\mathbb{E}_{y\sim P_{\text{Data}}(y)}[\|G_{X\to Y}(G_{Y\to X}(y)) - y\|_1]
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{L}_{id}(G_{X\to Y}, G_{Y\to X}) = &\mathbb{E}_{y\sim P_{\text{Data}}(y)}[\|G_{X\to Y}(y) - y\|_1] \\
&+ \mathbb{E}_{x\sim P_{\text{Data}}(x)}[\|G_{Y\to X}(x) - x\|_1]
\end{aligned}
$$

Figure: Adversarial Loss + Cycle-consistency Loss + Identity Loss

CycleGAN-VC: architecture



- One-to-one
- No parallel data

CycleGAN-VC2

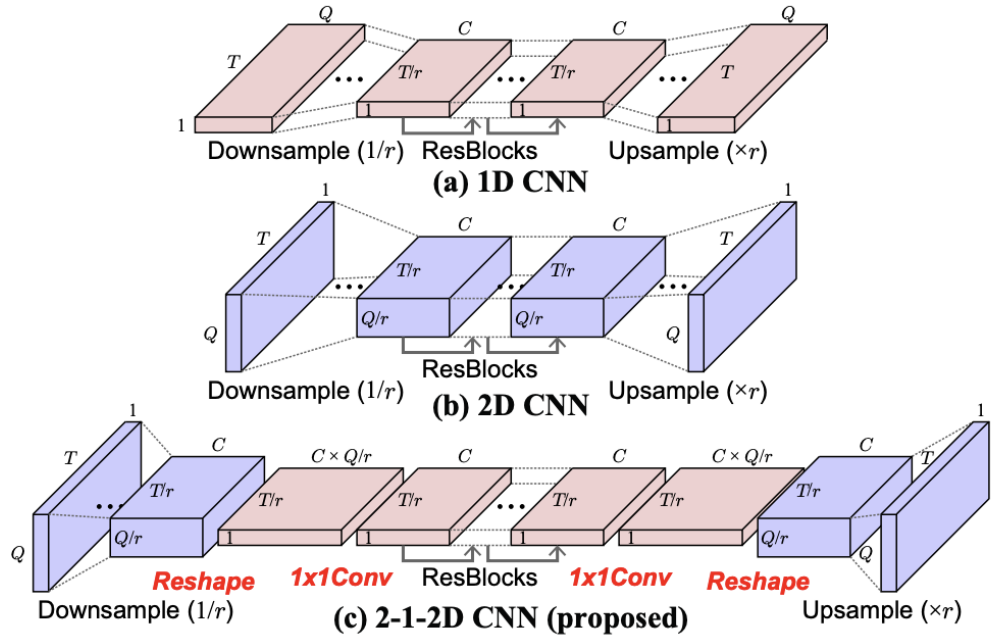


Fig. 2. Comparison of generator network architectures. Red and blue blocks indicate 1D and 2D convolution layers, respectively. r indicates a downsampling or upsampling rate.

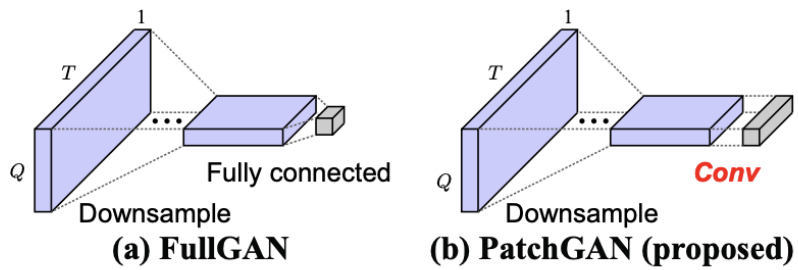


Fig. 3. Comparison of discriminator network architectures

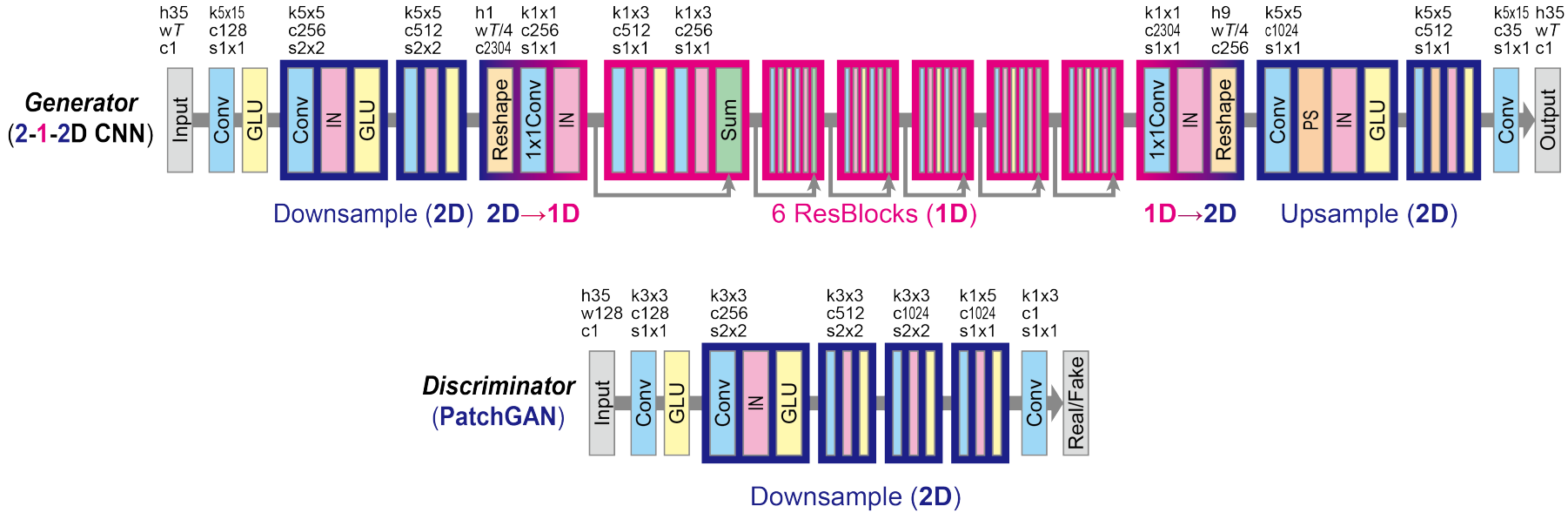
CycleGAN-VC2

$$\mathcal{L}_{adv2}(G_{X\to Y}, G_{Y\to X}, D'_X) = \mathbb{E}_{x\sim P_X(x)}[\log D'_X(x)] + \mathbb{E}_{x\sim P_X(x)}[\log(1 - D'_X(G_{Y\to X}(G_{X\to Y}(x))))]$$

Figure: Additional adv loss: Two-step Adversarial Loss

Takuhiro Kaneko et al., CycleGAN-VC2: Improved CycleGAN-based Non-parallel Voice Conversion, 2019

CycleGAN-VC2: architecture



Outline

StarGAN-VC

$$\mathcal{L}_{\text{adv}}^{D}(D) = -\mathbb{E}_{c\sim p(c),\mathbf{y}\sim p(\mathbf{y}|c)}[\log D(\mathbf{y}, c)] - \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}),c\sim p(c)}[\log(1 - D(G(\mathbf{x}, c), c))]$$

$$\mathcal{L}_{\text{adv}}^{G}(G) = -\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}),c\sim p(c)}[\log D(G(\mathbf{x}, c), c)]$$

$$\mathcal{L}_{\text{cls}}^{C}(C) = -\mathbb{E}_{c\sim p(c),\mathbf{y}\sim p(\mathbf{y}|c)}[\log p_C(c \mid \mathbf{y})]$$

$$\mathcal{L}_{\text{cls}}^{G}(G) = -\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}),c\sim p(c)}[\log p_C(c \mid G(\mathbf{x}, c))]$$

- Based on CycleGAN-VC
- Many-to-many (4 to 4)
- Domain Classification Loss
- No parallel data

Takuhiro Kaneko et al., StarGAN-VC: Non-parallel many-to-many voice conversion with star generative adversarial networks, 2018

CycleGAN-VC vs StarGAN-VC

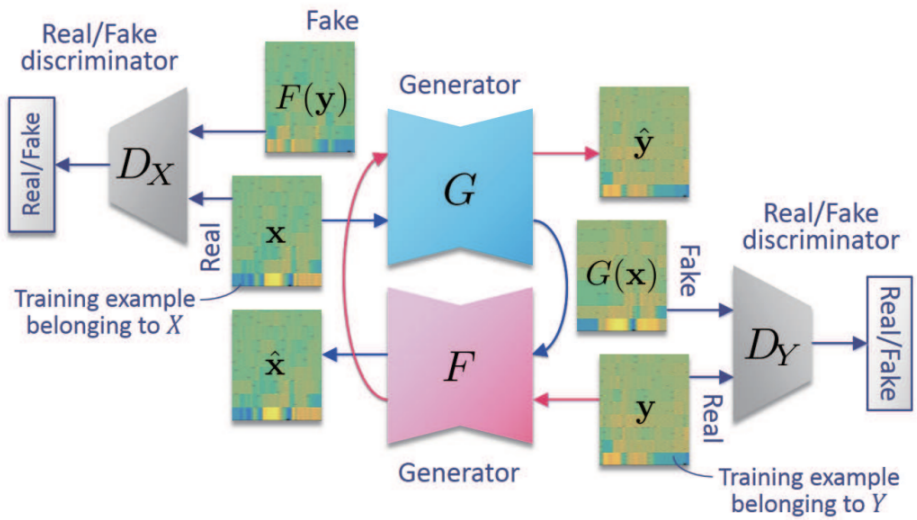


Fig. 1. Concept of CycleGAN training.

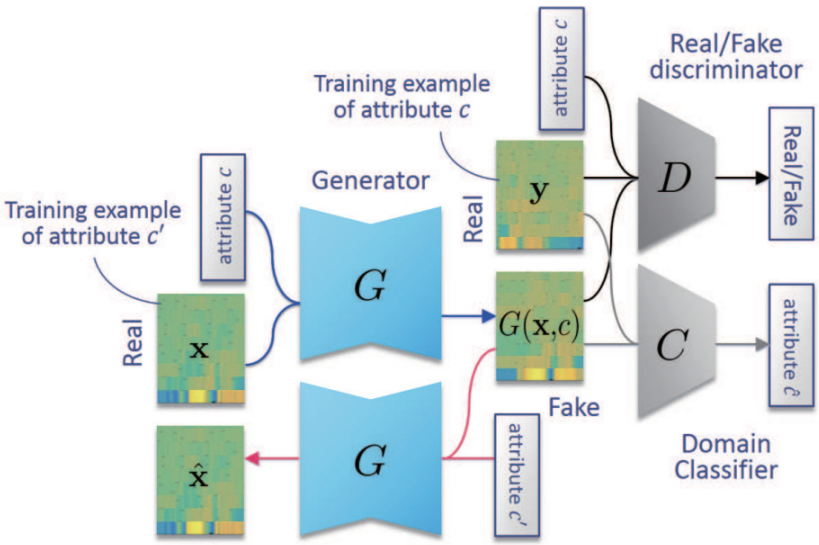
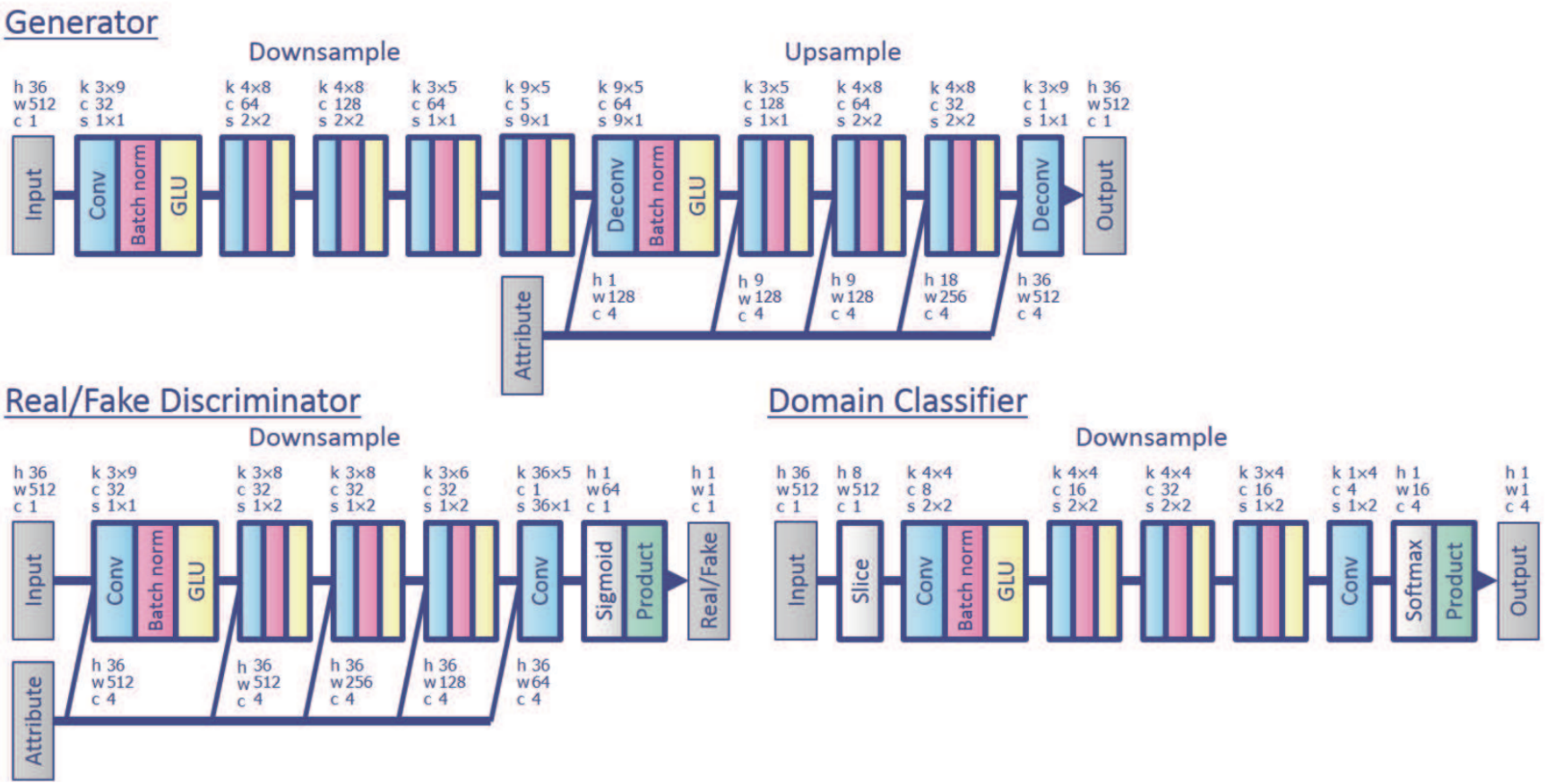


Fig. 2. Concept of StarGAN training.

StarGAN-VC: architecture



StarGAN-VC2: ST Adversarial Loss

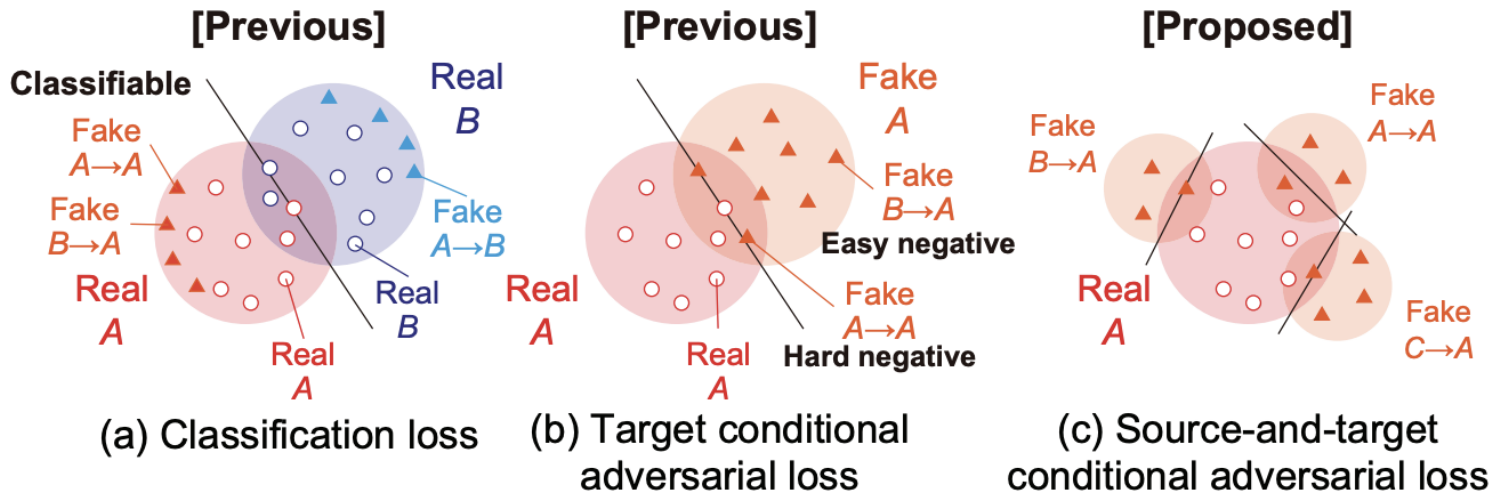


Figure: Comparison of conditional methods in training objectives. "A" and "B" denote the domain codes

$$\mathcal{L}_{st-adv} = \mathbb{E}_{(\boldsymbol{x},c)\sim P(\boldsymbol{x},c),c'\sim P(c')}\left[\log D\left(\boldsymbol{x}, c', c\right)\right] + \mathbb{E}_{(\boldsymbol{x},c)\sim P(\boldsymbol{x},c),c'\sim P(c')}\left[\log D\left(G\left(\boldsymbol{x}, c, c'\right), c, c'\right)\right]$$

Takuhiro Kaneko et al., StarGAN-VC2: Rethinking Conditional Methods for StarGAN-Based Voice Conversion, 2019

StarGAN-VC2: Conditional IN

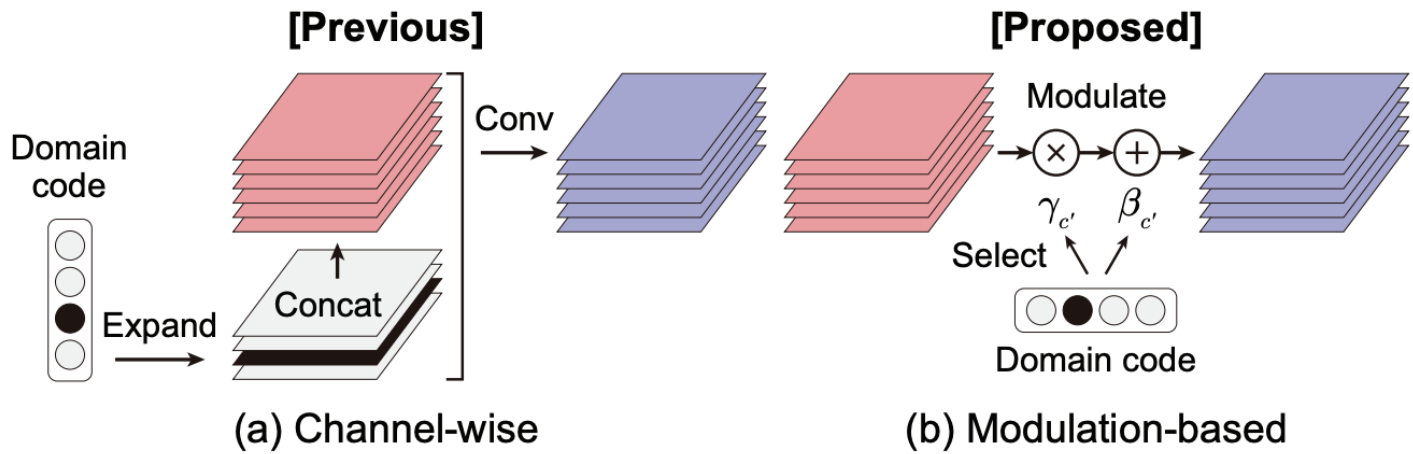
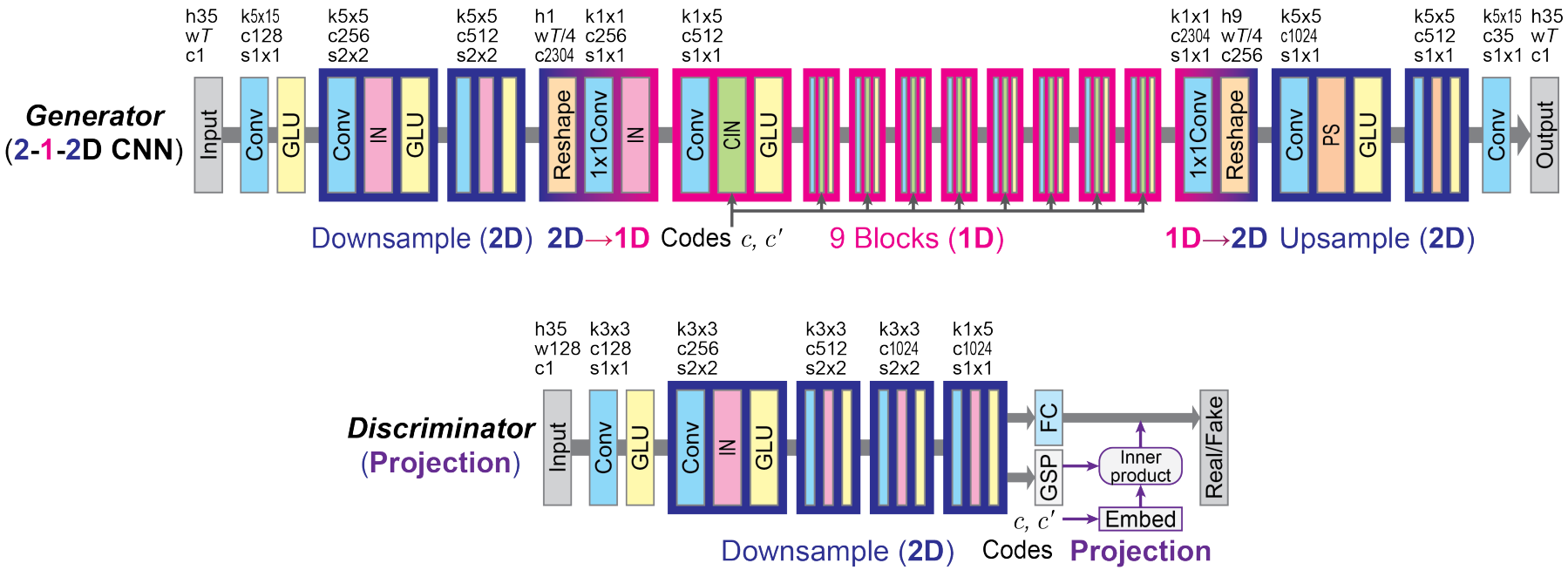


Figure: Comparison of conditional methods in generator networks

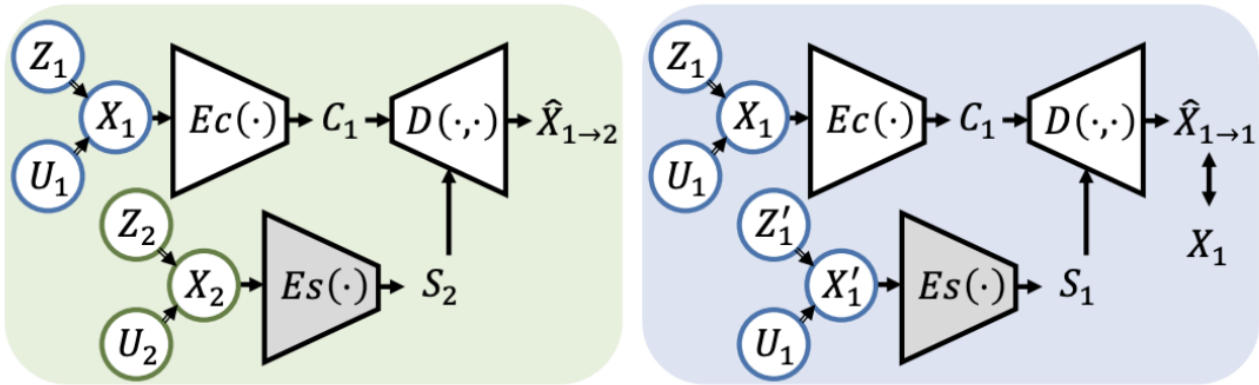
$$\mathrm{CIN}(\boldsymbol{f}; c') = \gamma_{c'} \left(\frac{\boldsymbol{f} - \mu(\boldsymbol{f})}{\sigma(\boldsymbol{f})} \right) + \beta_c$$

StarGAN-VC2:architecture



Outline

AutoVC



$$L_{\text{recon\_post}} = \left\| \hat{X}_{1\to 1} - X_1 \right\|_2^2$$

$$L_{\text{content}} = \left\| E_c\left(\hat{X}_{1\to 1}\right) - C_1 \right\|_1$$

$$L_{\text{recon\_pre}} = \left\| \tilde{X}_{1\to 1} - X_1 \right\|_2^2$$

$$L = L_{\text{recon\_post}} + \mu L_{\text{recon\_pre}} + \lambda L_{\text{content}}$$

Figure: The style transfer autoencoder framework. The grey boxes denote pre-trained modules. During training, the content encoder and the decoder minimize the self-reconstruction error.

Kaizhi Qian et al., AUTOVC: Zero-Shot Voice Style Transfer with Only Autoencoder Loss, 2019

AutoVC

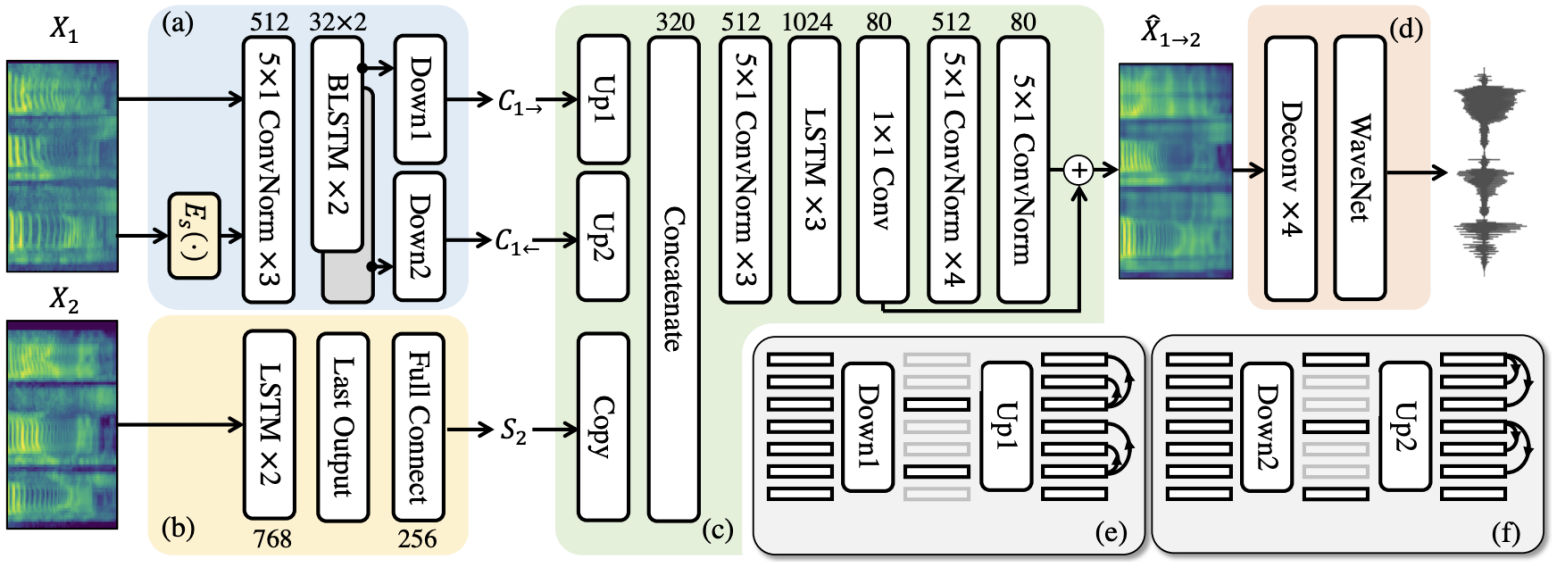


Figure: The style transfer autoencoder framework. The grey boxes denote pre-trained modules. During training, the content encoder and the decoder minimize the self-reconstruction error.

- Any-to-any (first zero-shot VC)
- No parallel data
- Speaker encoder trains to maximize the embedding similarity among different utterances of the same speaker, and minimize the similarity among different speakers (GE2E loss)

ConVoice

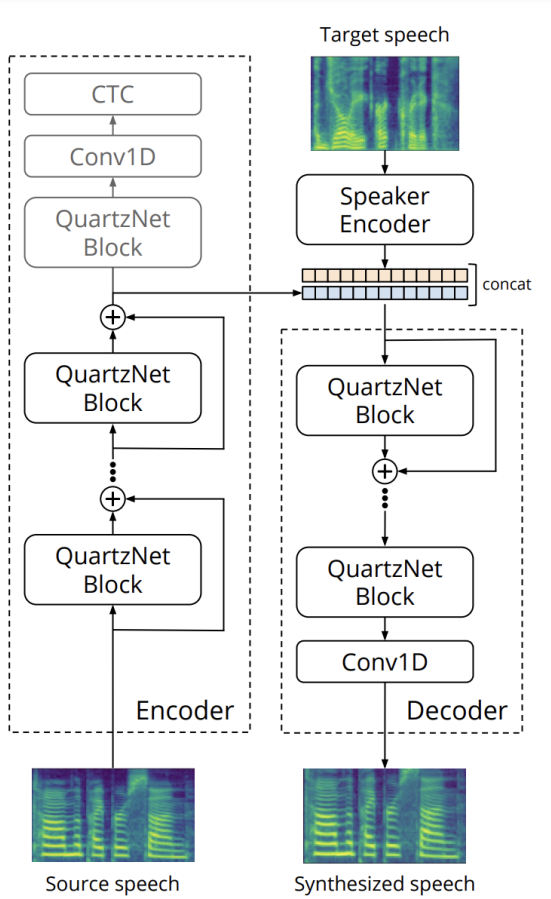


Figure: The pre-trained ASR model extracts acoustic features from the source utterance and the pre-trained speaker encoder produce the target speaker embedding from the target audio. Embeddings are concatenated and fed into the decoder that generates the melscale log spectrogram

Rebryk et al., ConVoice: Real-Time Zero-Shot Voice Style Transfer with Convolutional Network, 2020