Lecture 26: Speech Resynthesis from Discrete Disentangled Self-Supervised Representations

Mark Hasegawa-Johnson
All content CC-BY 4.0 unless otherwise specified.

ECE 537, Fall 2022

Outline

1. Speech Resynthesis from Discrete Disentangled Self-Supervised Representations

2. Vector-Quantized Variational Autoencoder

3. Transposed Convolution

4. Why Use Generative Adversarial Networks?

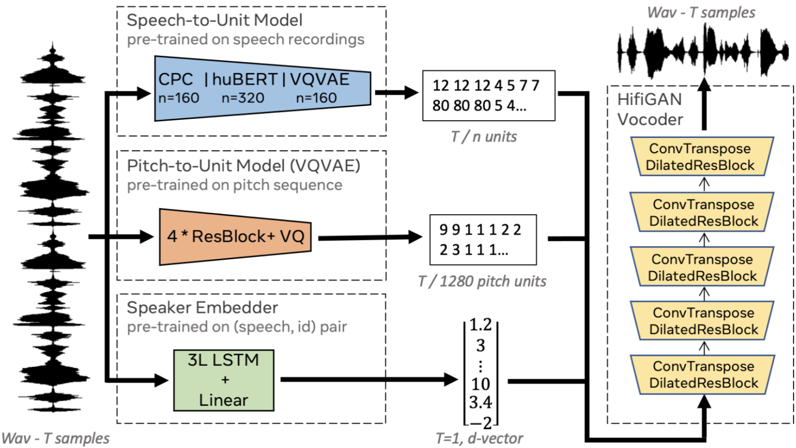
5. Summary

“Disentangled” Representations

The goal is that:

- Self-supervised units (CPC, HuBERT, or VQ-VAE) should represent phonemes (i.e., text, content) and rhythm.
- Pitch VQVAE represents pitch movements.
- Speaker embedder represents speaker identity.
- We can then resynthesize original speech with low bit rate, or perform voice conversion or pitch conversion.

Speech Resynthesis from Discrete Disentangled Self-Supervised Representations (Polyak et al., 2021)



Experimental Tests

- Low-bit-rate speech coding: resynthesize speech using discrete units, with as few bits/second as possible
- Voice Conversion: Synthesize same content, different voice
- Pitch Conversion: Synthesize same content & voice, but different pitch

Background Knowledge Necessary to Understand This Article

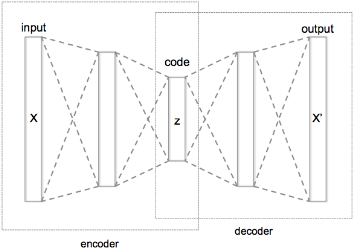
- CPC, HuBERT: you know this
- VQ-VAE
- Transposed convolution (Speech resynthesis)
- Why use generative adversarial networks (GANs)?

Outline

Review: Autoencoder

An autoencoder is trained to reconstruct the data, x, from a low-dimensional latent variable, z. The loss is usually L1 or L2 error (minimum absolute error, MAE, or minimum mean-squared error, MMSE):

$$\mathcal{L}_{\text{MAE}} = E\left[|x - x'|\right]$$
$$\mathcal{L}_{\text{MMSE}} = E\left[\|x - x'\|^2\right]$$



CC-SA 4.0,

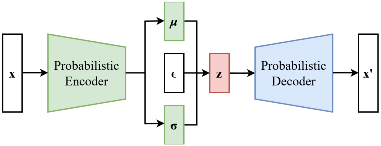
https://commons.wikimedia.org/wiki/File:

Autoencoder_structure.png

Variational Autoencoder (VAE)

A variational autoencoder (VAE) adds a requirement: z must have a known pdf. In this way, it becomes possible to generate novel, unseen data (use the VAE as a data generator). A typical formulation says that z must be Gaussian.

- The encoder computes z's mean and variance as functions of x.
- The decoder computes $x' = g(z)$.
- Training maximizes a lower bound on $p(x)$.



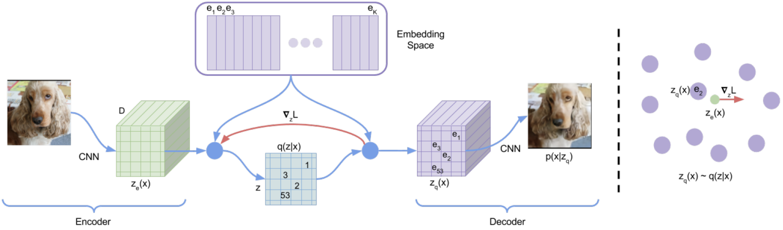
CC-SA 4.0,

https://commons.wikimedia.org/wiki/File:Reparameterized_Variational_Autoencoder.png

Vector-Quantized Variational Autoencoder (VQ-VAE)

In a VQ-VAE, the encoded input, $z_e(x)$, is quantized to the nearest codevector, e_j:

$$z_q(x) = \underset{j}{\operatorname{argmin}} \|z_e(x) - e_j\|_2$$

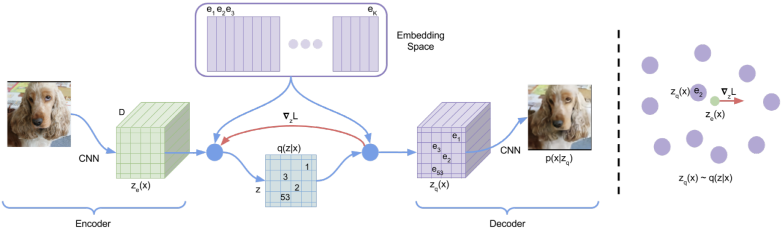


Oord et al., *Neural Discrete Representation Learning*, 2017

Vector-Quantized Variational Autoencoder (VQ-VAE)

During training, the gradient w.r.t. $z_e(x)$ is assumed to be equal to the gradient w.r.t. $z_q(x)$:

$$\nabla_{z_e(x)}\mathcal{L} \approx \nabla_{z_q(x)}\mathcal{L}$$



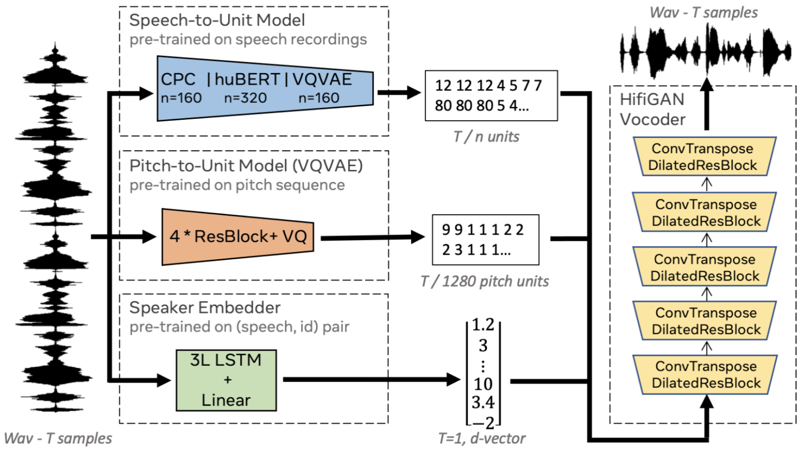
Oord et al., *Neural Discrete Representation Learning*, 2017

The SRDDSR Encoder

The SRDDSR encoder computes three separate latent variables $\mathbf{z}_c$, $\mathbf{z}_{F_0}$, and z_{spk}:

- $\mathbf{z}_c = (z_{c,1}, \ldots, z_{c,L})$ is a sequence of discrete content codes, $z_{c,i} \in \{0, 1, \ldots, K\}$
- $\mathbf{z}_{F_0} = (z_{F_0,1}, \ldots, z_{F_0,L})$ is a sequence of discrete F0 codes, $z_{F_0,i} \in \{0, 1, \ldots, K'\}$.
- $z_{\text{spk}} \in \Re^{256}$ is a speaker code. The speaker embedding network is pre-trained in a speaker verification task.

Speech Resynthesis from Discrete Disentangled Self-Supervised Representations (Polyak et al., 2021)



The SRDDSR Decoder

The SRDDSR decoder replaces $z_{c,i}$ and $z_{F_0,i}$ with real-valued decoder embeddings (from a lookup table or LUT), concatenates them with z_{spk}, and passes the result to a transposed-convolution synthesizer:

$$\begin{aligned}\mathbf{z} &= [z_1, \ldots, z_L] \\ &= \left[\left[\begin{array}{c} \text{LUT}(z_{c,1}) \\ \text{LUT}(z_{F_0,1}) \\ z_{\text{spk}} \end{array} \right], \cdots, \left[\begin{array}{c} \text{LUT}(z_{c,L}) \\ \text{LUT}(z_{F_0,L}) \\ z_{\text{spk}} \end{array} \right], \right]\end{aligned}$$

Outline

Transposed Convolution

Transposed convolution repeats the following steps across several layers:

1. Upsample the input, e.g.,

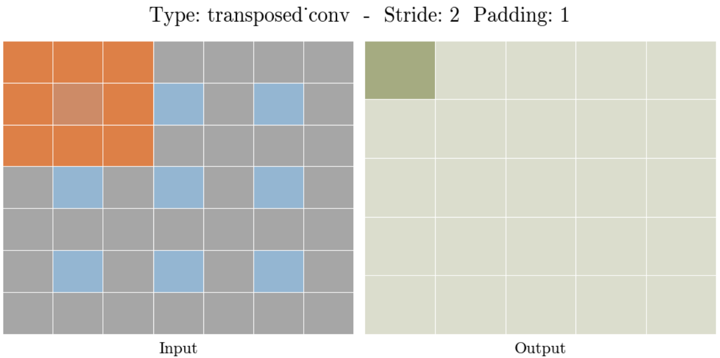
$$\begin{aligned}\mathbf{h} &= [h_1, h_2, h_3, h_4, h_5, \ldots, h_{2L-1}, h_{2L}, h_{2L+1}] \\ &= [0, z_1, 0, z_2, 0, \ldots, 0, z_L, 0]\end{aligned}$$

2. Convolution:

$$\mathbf{y} = [y_1, \ldots, y_{2L+1}]$$
$$y_i = \sum_{j=-D}^{D} K_j h_{i+j},$$

where $K_j \in \Re^{|y|\times|h|}$ is the weight matrix connecting output vector y_i to input vector h_{i+j}.

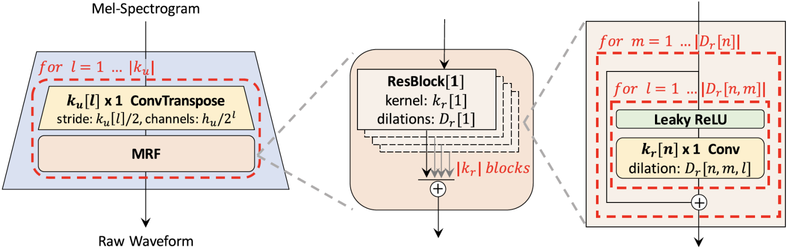
2D Transposed Convolution Example



Aqeel Anwar, `https://github.com/aqeelanwar/conv_layers_animation`

Multi-Receptive-Field

The HiFi-GAN synthesizer follows each transposed convolution with a multi-receptive-field (MRF) module. MRF is a bank of $|k_r|$ different dilated convolutions in parallel, each with a different dilation:



Kong, Kim & Bae, "HiFi-GAN," 2020

SRDDSR Speech Synthesis from Transposed Convolution

The final speech signal, $\hat{x}[n]$, is created by another transposed convolution layer, but this time, $k_j \in \Re^{|h|}$ is a vector:

$$\hat{\mathbf{x}} = [\hat{x}[1], \ldots, \hat{x}[N]]$$

$$\hat{x}[n] = \sum_{j=-D}^{D} k_j^T h_{n+j}$$

SRDDSR Speech Synthesis from Transposed Convolution

The weights of the entire network are then trained in order to minimize a loss term that computes the mel-spectrogram of the input, $\phi(\mathbf{x})$, and the mel-spectrogram of the output, $\phi(\hat{\mathbf{x}})$, and compares them:

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^{L} \|\phi(\mathbf{x}) - \phi(\hat{\mathbf{x}})\|_1$$

Outline

Why is "Regression" called "Regression"?

- The word "regression" comes from Galton's *Regression towards mediocrity in hereditary stature*.
- It refers to the fact that a regression estimate is closer to average (more "mediocre," less "extreme") than the variable it estimates.
- Example:

$$\begin{bmatrix} x \\ z \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}\right)$$

  Then the MMSE estimate of x, $\hat{x} = \rho z$, is more mediocre than x:

$$E\left[|\hat{x}|^2\right] \leq E\left[|x|^2\right]$$

Why is this a problem for autoencoders?

Suppose that $\hat{\mathbf{x}}$ is estimated in order to minimize

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^{L} \|\phi(\mathbf{x}) - \phi(\hat{\mathbf{x}})\|_1,$$

The law of "regression towards the mean" dictates that a neural net trained to minimize $\mathcal{L}_{\text{recon}}$ will produce a $\hat{\mathbf{x}}$ that is closer to average than $\mathbf{x}$, and that will therefore sound sort of "smoothed out."

Generative Adversarial Network

A GAN is a pair of networks, trained to be one another's adversaries.

- The **generator**, G, tries to generate realistic speech signals.
- The **discriminator**, D, tries to discriminate real vs. generated speech.

Generative Adversarial Network

Suppose $\mathbf{x}$ is a real speech signal, and $\hat{\mathbf{x}}(\mathbf{z})$ is a generated speech signal.

- The **discriminator**, D, tries to generate a score $D(\mathbf{x})$ that is close to 1 for real speech, and close to 0 for fake speech:

$$\mathcal{L}_D(G,D) = E_{\mathbf{x}}\left[f\left(1 - D(\mathbf{x})\right)\right] + E_{\mathbf{z}}\left[f\left(D(\hat{\mathbf{x}}(\mathbf{z}))\right)\right]$$

- The **generator**, G, tries to generate speech that fools the discriminator:

$$\mathcal{L}_{\text{adv}}(G,D) = E_{\mathbf{z}}\left[f\left(1 - D(\hat{\mathbf{x}}(\mathbf{z}))\right)\right]$$

The function $f(D)$ can be D, D^2, $\ln(D)$, or other (SRDDSR uses D^2).

Why might GANs be better than VAE?

- The optimum discriminator: if $p(\mathbf{x}|\text{fake}) > p(\mathbf{x}|\text{real})$, then call it "fake," otherwise call it "real."
- The optimum generator: make sure that the distribution of fake speech is exactly the same as the distribution of real speech, $p(\mathbf{x}|\text{fake}) = p(\mathbf{x}|\text{real})$.
- Thus a GAN converges to a solution where $\hat{\mathbf{x}}(\mathbf{z})$ has the same distribution as $\mathbf{x}$, not just the same average value.
- In order to match the distribution, the generator must generate speech samples that are not smoothed toward the mean; the discriminator must think they sound realistic.

GAN Loss in HiFi-GAN and SRDDSR

- HiFi-GAN uses J different types of discriminators: multi-scale (several different scales of CNNs at input), and multi-rate (several different dilation rates).
- SRDDSR adds the VQ-VAE reconstruction loss, a "feature matching" loss $\mathcal{L}_{fm}(G, D_j)$, and the VAE reconstruction loss, thus

$$\mathcal{L}_G(D, G) = \sum_{j=1}^{J} [\mathcal{L}_{\text{adv}}(G, D_j) + \lambda_{fm}\mathcal{L}_{fm}(G, D_j)] + \lambda_r \mathcal{L}_{\text{recon}}(G)$$

Outline

Summary: Discrete Disentangled Self-Supervised Representations

- Content is encoded using CPC, HuBERT, or VQ-VAE.
- Pitch is encoded using a VQ-VAE:

$$z_q(x) = \underset{j}{\operatorname{argmin}} \|z_e(x) - e_j\|_2$$

$$\nabla_{z_e(x)}\mathcal{L} \approx \nabla_{z_q(x)}\mathcal{L}$$

- Speaker ID is encoded using a neural net trained to perform speaker discrimination.

Summary: Speech Resynthesis

- Speech resynthesis uses transposed convolution:

$$[h_1, \ldots, h_{2L+1}] = [0, z_1, 0, z_2, 0, \ldots, 0, z_L, 0]$$

$$y_i = \sum_{j=-D}^{D} K_j h_{i+j}$$

- Reconstruction minimizes absolute error of the mel-frequency spectrogram:

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^{L} \|\phi(\mathbf{x}) - \phi(\hat{\mathbf{x}})\|_1,$$

- GAN is used to avoid "regression to the mean," to make sure speech sounds good:

$$\mathcal{L}_G(D, G) = \sum_{j=1}^{J} [\mathcal{L}_{\text{adv}}(G, D_j) + \lambda_{fm}\mathcal{L}_{fm}(G, D_j)] + \lambda_r \mathcal{L}_{\text{recon}}(G)$$