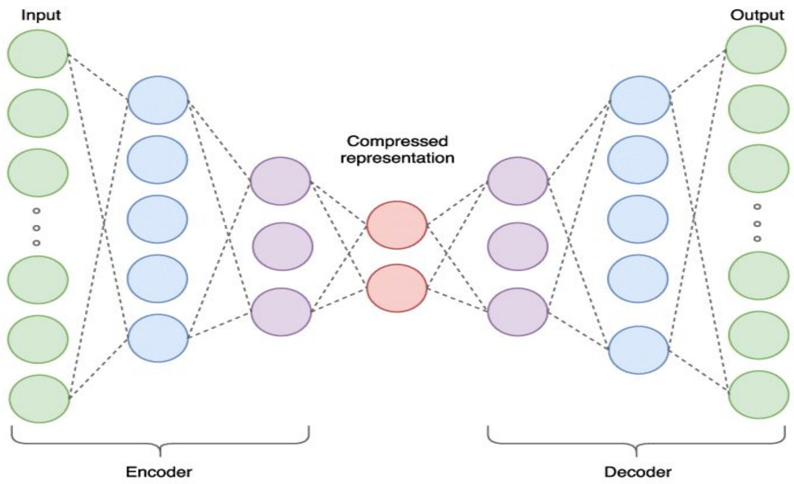
Generative Approaches

- Generate or reconstruct input data
  - Predict future from past
  - Masked from unmasked
  - Original from some corrupted view
    - Autoencoders
    - Variational autoencoders

Autoencoders

- Neural network which reconstructs its own inputs, x
- Learns useful latent representation, z
- Regularized by bottleneck layer – compresses latent representation
- Encoder f (x) → z and decoder g(z) → x
- The low-dimensional representation can be used as the representation of the data in various applications, e.g., high image resolution, image denoising, coloring and missing patch reconstruction

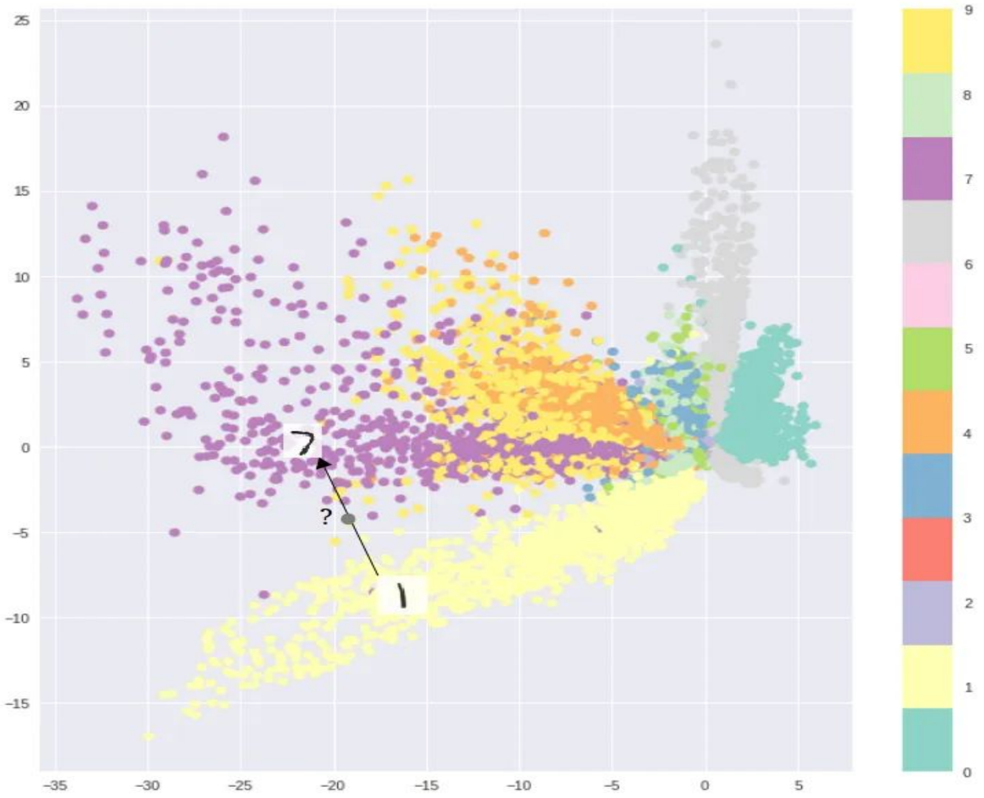


Limitations

- Latent space is discontinuous
- Generation: randomly sample

  from latent space
- Decoder will generate unrealistic

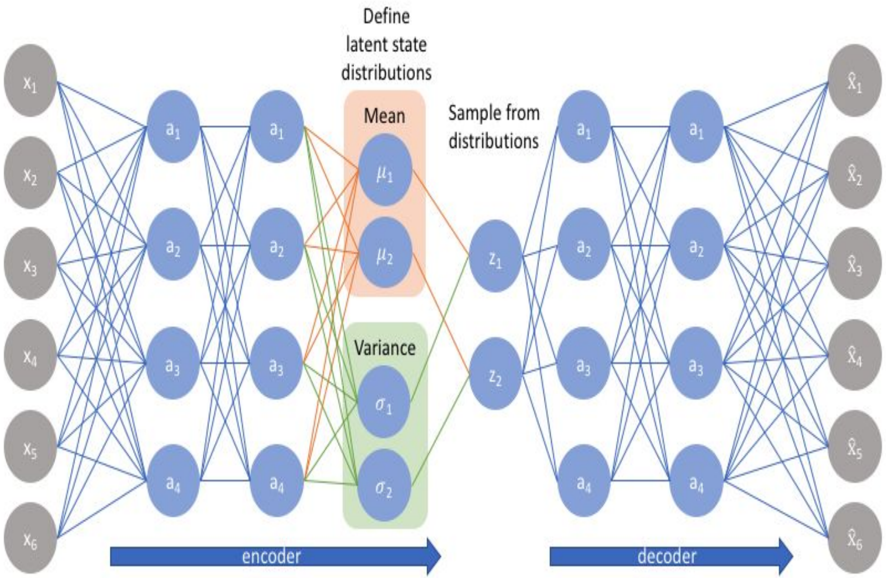
  output



Autoencoder trained on MNIST data set
2D latent representation

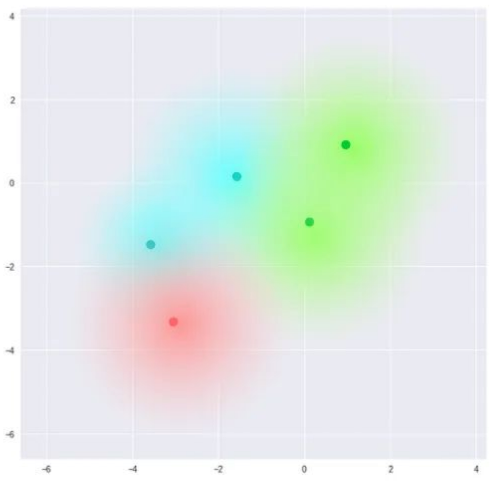
Variational autoencoders

- Latent space is continuous
- Allowing easy to random sampling and interpolation
- For same input mean, variance same but due to sampling encodings will vary

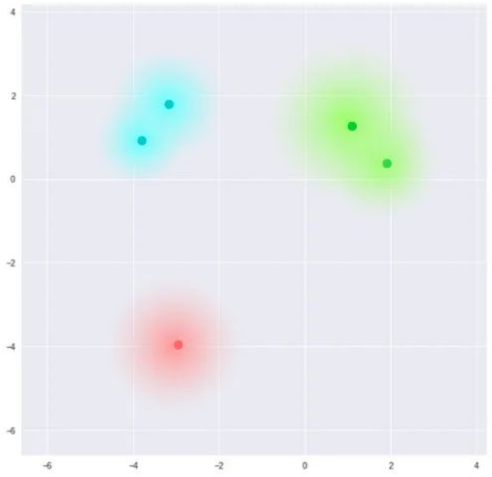


VAE

- Now latent space is smooth
- Want overlap between classes
- Minimize KL-divergence between latent variable, standard normal distribution



What we require

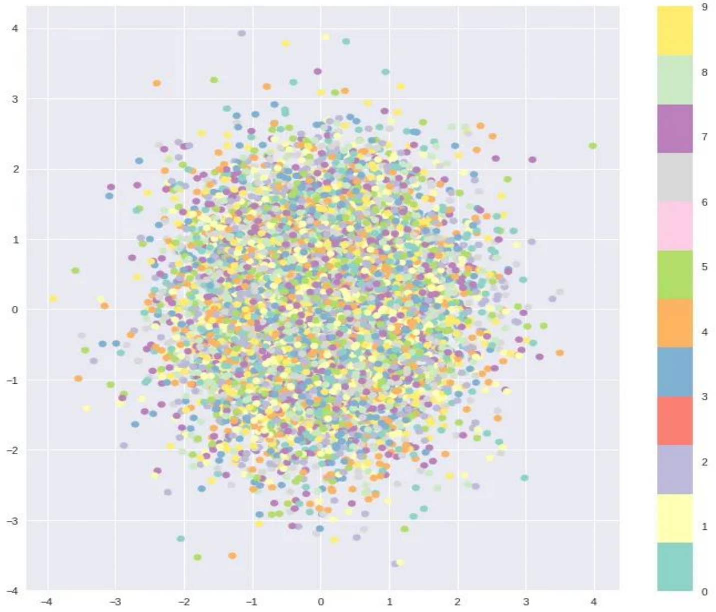


What we may inadvertently end up with

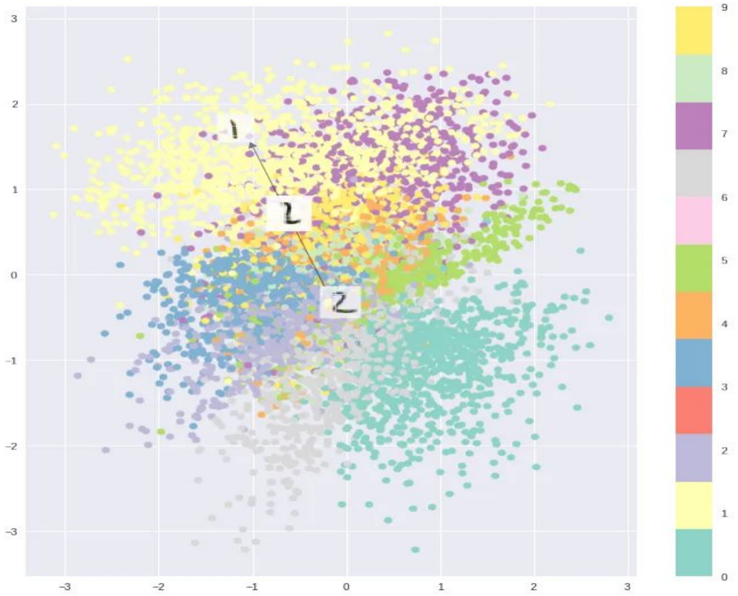
VAE

- Encodings densely placed in center of latent space
- Prior distribution of latent space

Optimizing using purely KL-loss

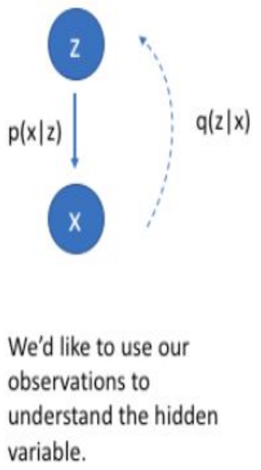


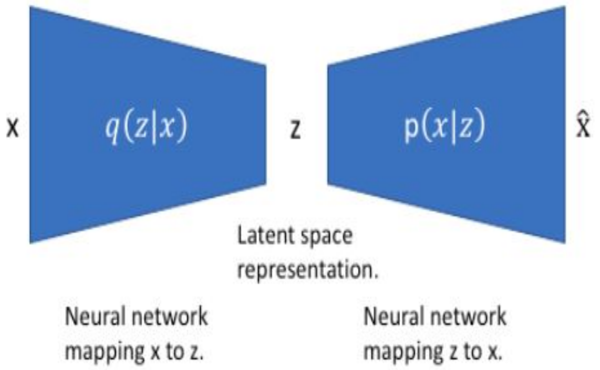
Optimizing using purely KL-loss & reconstruction loss



VAE Loss function

- Infer the characteristics of **Z** from observations **X**
- Intractable distribution
- Approximate by another distribution (family of gaussian)
- Minimize **KL** divergence

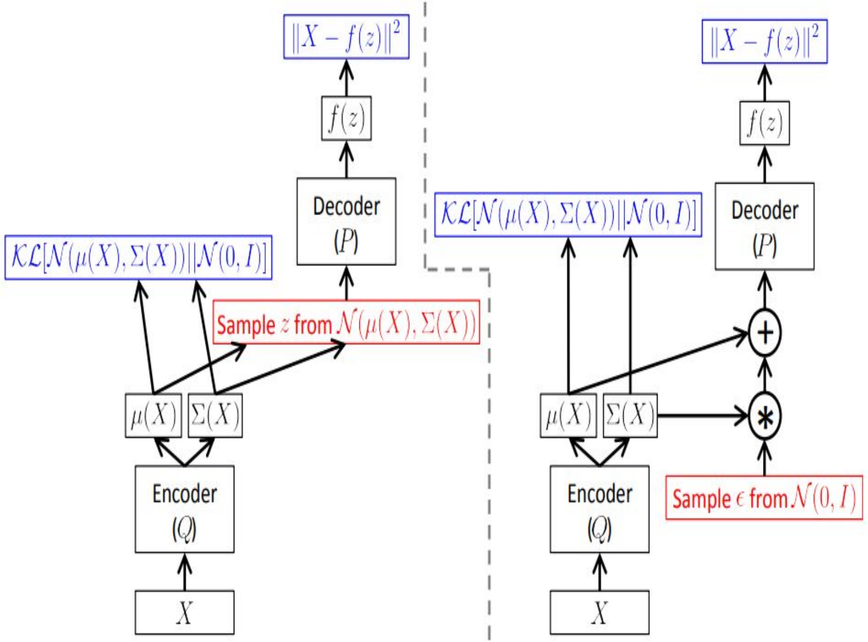




Reparameterization

Issue. we need to back-propagate through a stochastic node

Solution. Reparameterization trick: write z as a deterministic transformation of a simpler random variable, so that all the parameters are in the "deterministic part" of the graph.



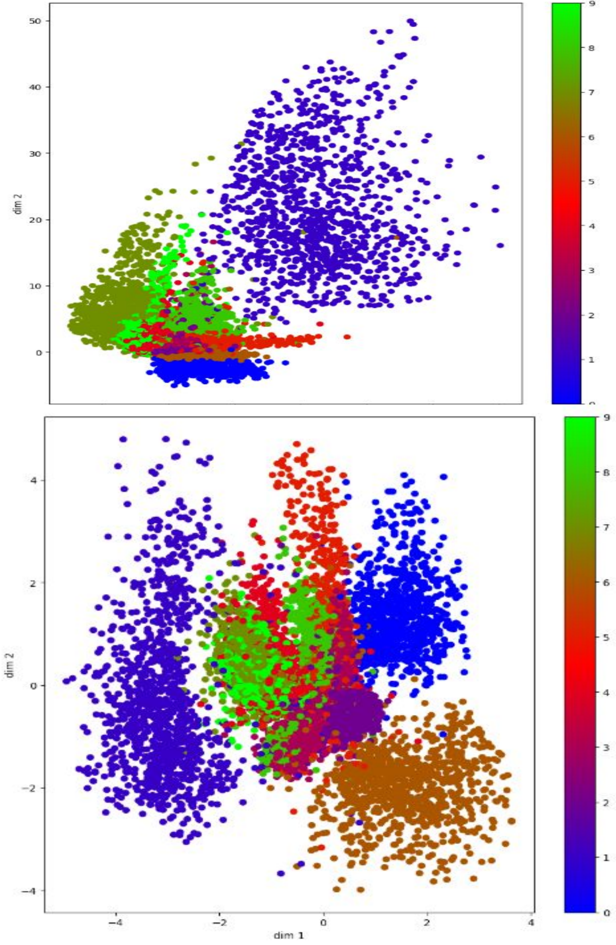
Auto encoders & VAE’s

Autoencoders:

- Learn a compressed representation of data.
- Keep the important features of the data.

Variational Autoencoders (VAEs):

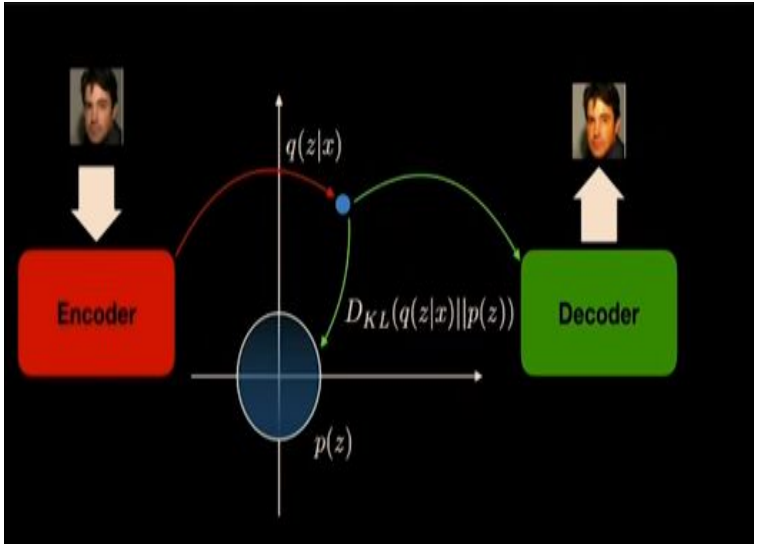
- Can generate new data.
- Model uncertainty, which is helpful for generating data.



2D-latent space

Limitations

- Posterior Collapse
  - The latent code become independent of data samples
  - It results in limited diversity and poor quality in the generated samples
- Static Prior
  - Lack of flexibility
  - Mismatch between prior and data distribution
  - Inability to capture data dynamics

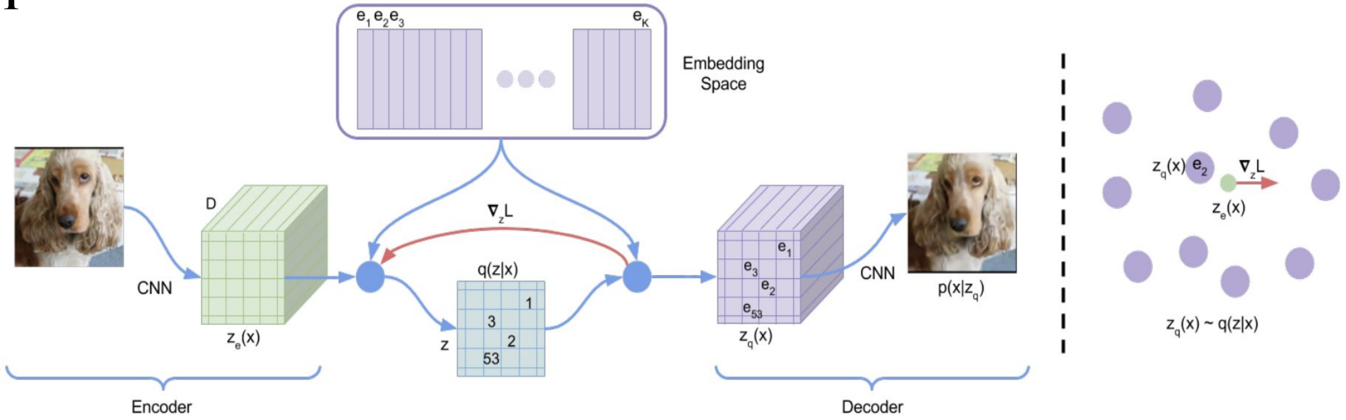


$$\mathcal{L}(x;\theta,\phi) = \underbrace{\mathbb{E}_{q_\phi}[\log p_\theta(x|z)]}_{\text{Reconstruction Likelihood}} - \underbrace{\mathrm{KL}(q_\phi(z|x)||p(z))}_{\text{Divergence from Prior}}$$

Discrete Spaces:

- Lot of the data we encounter in the real world favors a discrete representation
    - Example: Human speech is well represented by discrete phonemes
- Effective use of latent space
    - Opposed to focusing or spending capacity on noise and imperceptible details which are local
    - Model important features that usually span many dimensions in data space
        - Example: phonemes in speech, objects in image
- Number of algorithms are designed to work on discrete data

Algorithm-phase 1



Step I: Input **x** is encoded into continuous **Ze(x)**

Step II: Transforming into **Z--** discrete variable over **K** categories

We define a latent embedding space $e \in R^{K \times D}$

(D is the dimensionality of each latent embedding vector)

To discretize **Ze(x)** : calculate a nearest neighbour in the embedding space

$$k = \text{argmin}_j \|z_e(x) - e_j\|_2$$

The posterior categorical distribution $q(z|x)$ -- deterministic

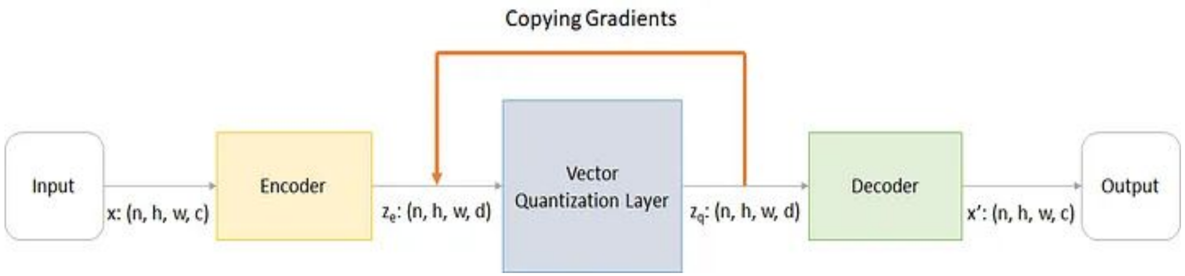
$$q(z = k|x) = \begin{cases} 1 & \text{for } \mathrm{k} = \text{argmin}_j \|z_e(x) - e_j\|_2 \\ 0 & \text{otherwise} \end{cases}$$

Step III: use $z_q(x)$ as input to the decoder

$$z_q(x) = e_k, \quad \text{where} \quad k = \text{argmin}_j \|z_e(x) - e_j\|_2$$

Algorithm-phase 1

Backpropagation



Reconstruction loss: $\boxed{\log p(x|z_q(x))}$

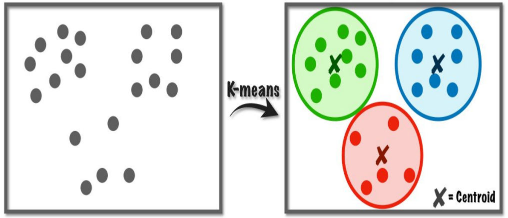
Just copy gradients from decoder $z_q(x)$ input to encoder output **Ze(x)** (straight-through estimator)

Main idea: Gradients from decoder contain information for how the encoder has to change its output to lower the reconstruction loss

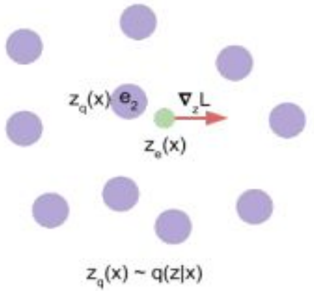
Code book update

- Embedding don't get gradient from reconstruction loss
- Use L2 error to move the embedding vectors e_i towards $z_e(x)$

- **Embedding loss** = $\boxed{\|\text{sg}[z_e(x)] - e\|_2^2}$
  - sg = stop gradient operator



Commitment loss= $\boxed{\beta\|z_e(x) - \text{sg}[e]\|_2^2}$



To make sure encoder commits to an embedding and its out does not grow

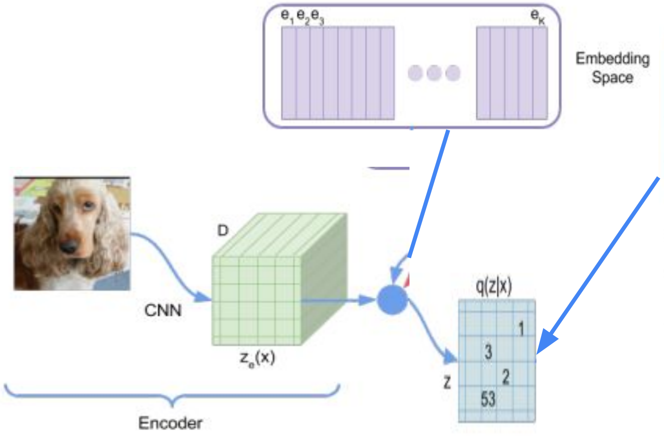
$$L = \log p(x|z_q(x)) + \|\text{sg}[z_e(x)] - e\|_2^2 + \beta\|z_e(x) - \text{sg}[e]\|_2^2$$

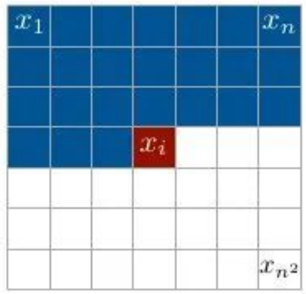
<u>Algorithm -phase 2</u>

Prior Learning

- Train PixelCNN on the discrete latent space. Sample from PixelCNN, decode with VQ-VAE decoder.

- Learn an autoregressive prior over discrete **z**
  - PixelCNN for images
  - WaveNet for raw audio

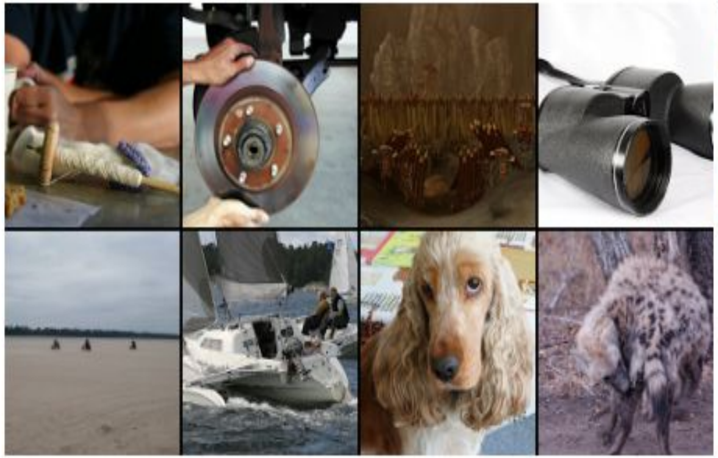




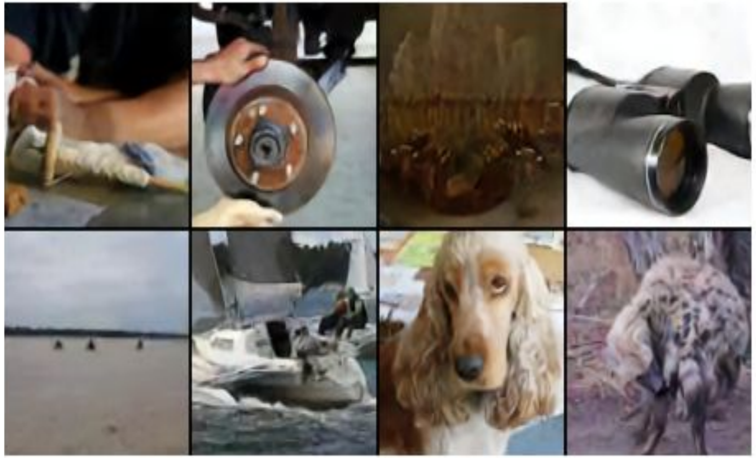
Pixel CNN

Experiments

128x128x3 images ⟺ 32x32x1 discrete latent space (K=512)



Original



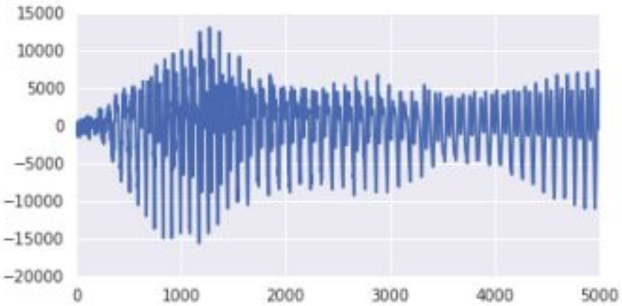
Reconstructed

128x128x3x(8 bits per pixel) / 32x32x(9 bits to index a vector)= 42.6 times compression in bits

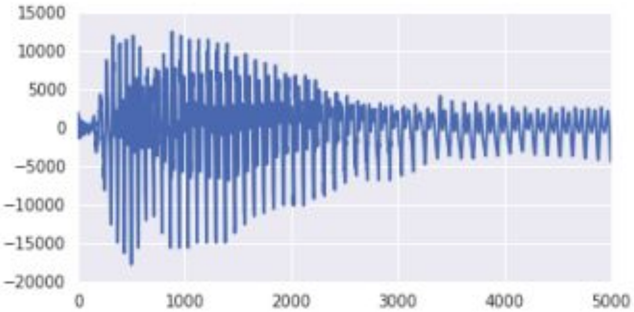
Generation

Microwave pickup tiger beetle coral reef brown bear

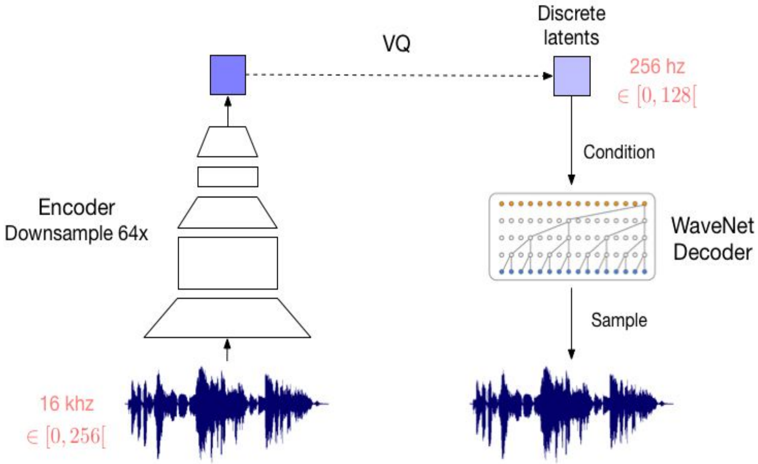
Audio



original



Reconstructed with same speaker id



- Same text content
- Waveform is quite different and prosody in the voice is altered
- Without any form of linguist supervision, only encodes the content of the speech

Realtime Audio Variational autoEncoder (RAVE)

Stage 1: Representation Learning:

Trains a variational autoencoder (VAE) to learn a compact latent representation of audio waveforms.
Focuses on balancing reconstruction fidelity and representation compactness in the learned latent space.

Stage 2: Adversarial Fine-Tuning:

Utilizes adversarial training techniques, potentially drawing from GANs.
Aims to refine the learned representation for enhanced audio synthesis quality.

Post-Training Analysis of Latent Space:

Involves analyzing and manipulating the learned latent representations.
Enables direct control over various attributes of synthesized audio (e.g., pitch, tempo, style).
Facilitates finding a trade-off between fidelity and compactness for desired audio synthesis outcomes.

Variational Autoencoders (VAEs) are powerful generative models that learn rich, compressed representations of data. Their encoder-decoder structure enables both efficient data encoding and generation of new, meaningful data samples.

VAEs as Analysis-Synthesis Frameworks:

- Enable generation of new data while preserving meaningful features.
- Flexible architecture for learning representations without strict feature constraints.
- Challenges in Latent Dimensionality for VAEs:

Incorrect dimensionality:

- Too low: Leads to poor reconstruction and distorted data.
- Too high: Introduces noise or redundancy in the latent space.
- Balancing act needed for meaningful, informative representations.

Issue:
In order to address the dimensionality of the learned representation, we introduce a novel method to split the latent space between informative and uninformative parts

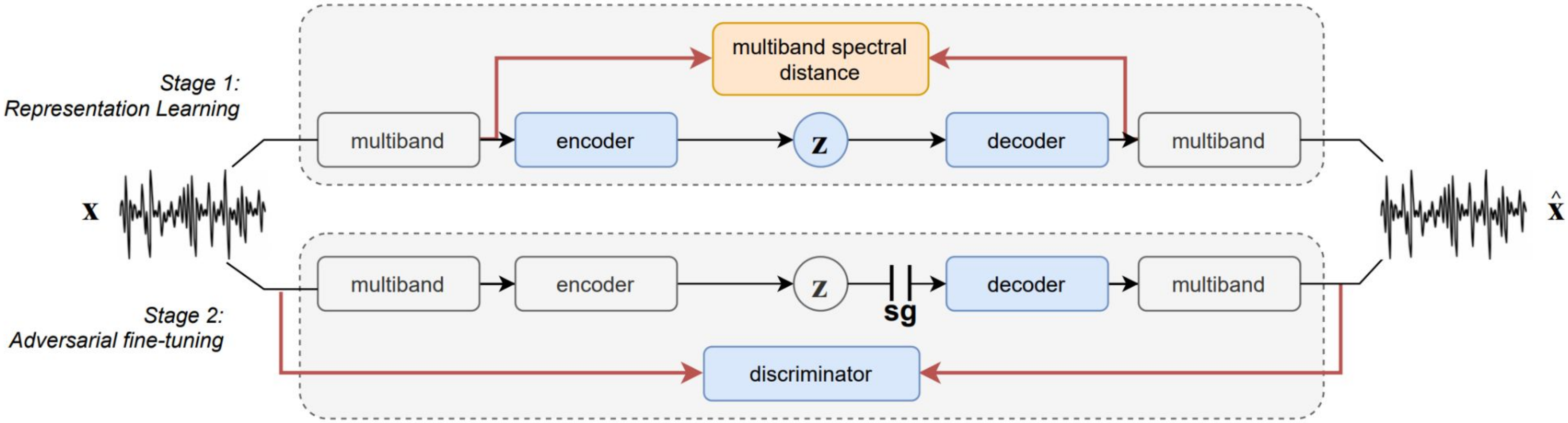
Multiband Audio Processing:

- Divides audio signals into distinct frequency bands.
- Involves segmenting the spectrum into different ranges (e.g., bass, midrange, treble).

Uses in Deep Learning:

- Enhanced Representation: Provides diverse spectral information for richer feature extraction.
- Robustness to Noise: Helps in noise reduction by analyzing different frequency components separately.
- Task-Specific Analysis: Enables focused analysis of frequency bands for specific audio tasks (e.g., speech recognition, music analysis).
- Improved Model Performance: Offers more comprehensive and informative input for deep learning models handling audio data.

Overall architecture of the proposed approach



Stage 1: Representation learning:

$$S(\mathbf{x},\mathbf{y}) = \sum_{n\in\mathcal{N}} \left[\frac{\|\mathrm{STFT}_n(\mathbf{x}) - \mathrm{STFT}_n(\mathbf{y})\|_F}{\|\mathrm{STFT}_n(\mathbf{x})\|_F} + \log\left(\|\mathrm{STFT}_n(\mathbf{x}) - \mathrm{STFT}_n(\mathbf{y})\|_1\right) \right]$$

Frobenius Norm Term:

- Measures overall difference between real and synthesized waveforms at a specific scale.
- Normalizes the difference by original magnitude, indicating relative change in spectral content.
- Offers insight into the overall magnitude change between real and synthesized spectra.

L1 Norm Term:

- Computes absolute differences between real and synthesized waveforms at a given scale.
- Logarithmic scaling emphasizes smaller differences, complementing the Frobenius norm term.
- Focuses on absolute discrepancies, potentially highlighting finer spectral differences.

Stage 2: Adversarial fine-tuning:

The second training stage aims at improving the synthesized audio quality and naturalness. As we consider that the learned representation has reached a satisfactory state at this point, we freeze the encoder and only train the decoder using an adversarial objective.

$$\mathcal{L}_{\text{dis}}(\mathbf{x}, \mathbf{z}) = \max(0, 1 - D(\mathbf{x})) + \mathbb{E}_{\hat{\mathbf{x}} \sim p(\mathbf{x}|\mathbf{z})}[\max(0, 1 + D(\hat{\mathbf{x}}))],$$
$$\mathcal{L}_{\text{gen}}(\mathbf{z}) = -\mathbb{E}_{\hat{\mathbf{x}} \sim p(\mathbf{x}|\mathbf{z})}[D(\hat{\mathbf{x}})].$$

(1 - D(x)) Term:

- Measures how much the discriminator thinks a real sample (x) is fake.
- If D(x) is close to 1 (indicating high confidence that (x) is real), (1 - D(x) is close to 0 (indicating low belief that (x) is fake).

- In simpler terms, (1 - D(x)) tells us how much the discriminator doubts the authenticity of a real sample. When the discriminator is confident in a real sample being real, (1 - D(x)) becomes low.

Datasets:

VCTK Dataset Overview:

- Contains 44 hours of raw audio data.
- Recorded at a sampling rate of 48kHz.
- Features 110 speakers with diverse accent

Strings.

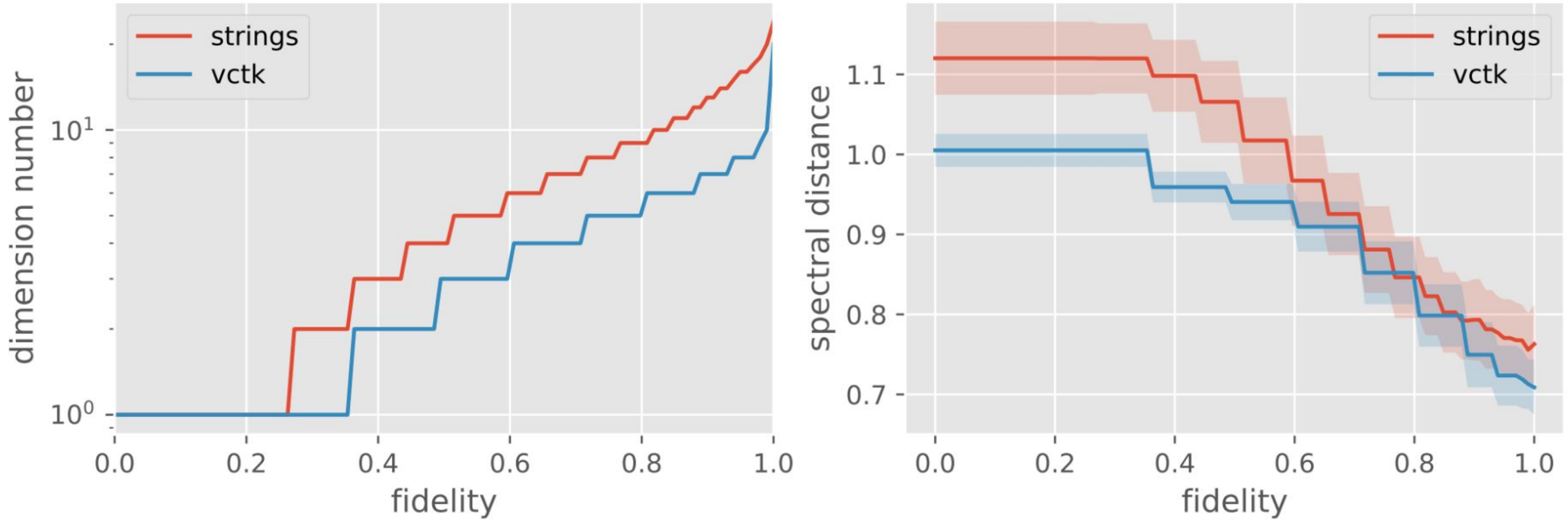
- Dataset Overview:
  - Comprised of approximately 30 hours of raw recordings featuring strings.
  - Encompasses various configurations: monophonic solos and polyphonic group performances in different styles and recording setups.
- Audio Characteristics:
  - Recorded at a high sampling rate of 48kHz, ensuring quality audio data.
- Data Division:
  - Follows a standard 90/10 train/test split for training and evaluating models.

Second Term Explanation:

- It calculates the average assessment of how real these generated samples appear to the discriminator.
- For each generated sample, it checks how much the discriminator perceives it as real or fake.
- It penalizes the discriminator if it mistakenly thinks a generated sample is real.

The loss function for the generator in a Generative Adversarial Network (GAN):

- The generator aims to minimize how much the discriminator thinks its generated samples are real.
- It does this by adjusting its parameters to produce samples that deceive the discriminator into believing they are real data.
- Minimizing this loss encourages the generator to generate more convincing samples.



Estimated latent space dimensionality according to the fidelity parameter and its corresponding influence on the reconstruction quality.

Contribution of Each Team Members

- Contributors:
  - Chunarkar Sumeeth Kumar
  - Velma Dhatri Reddy
- Contribution:
  - Conducted experiments and research related to Variational Autoencoders (VAEs).
  - Contributions focused on implementing VAEs within the RAVE system for audio synthesis or analysis.

- Contributors:
  - Meka Nani
  - Gujjula Samara
- Contributions:
  - Conducted analysis of VQ-VAE (Vector Quantized Variational Autoencoder)

  - Performed a detailed literature survey of RAVE's applications and explored various models for audio synthesis or analysis.

Disclaimer

Most of the content is from following blogs and tutorials

https://www.jeremyjordan.me/variational-autoencoders/

https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

https://www.youtube.com/watch?v=uaaqyVS9-rM&ref=jeremyjordan.me

https://arxiv.org/pdf/2111.05011.pdf