# Neural Discrete Representation Learning

Paper Review

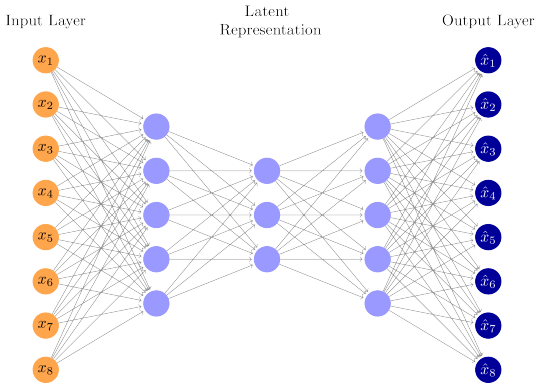
Fred Jeong

February 19, 2025

# Overview

1. Autoencoder
2. Latent Space
3. Variational Autoencoder
4. VQ-VAE
5. Learning

# Recall: Autoencoder



# Latent Space

- Latent variables contain a compressed, low-level representation of data.
- Latent feature representation $\mathbf{h}_i \in \mathbb{R}^q$ of an input $\mathbf{x}_i \in \mathbb{R}^n$ from the encoder function $g$ is defined as

$$\mathbf{h}_i = g(\mathbf{x}_i).$$

- Reconstructed input $\tilde{\mathbf{x}}_i \in \mathbb{R}^n$ from the decoder function $f$ is defined as

$$\tilde{\mathbf{x}}_i = f(\mathbf{h}_i) = f(g(\mathbf{x}_i))$$

- Training an autoencoder means finding the functions $g$ and $f$ such that the reconstruction loss

$$\sum_i \|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|_2$$

is minimised.

Variational Autoencoder

**Variational** Autoencoder

# Latent space

- Autoencoders are deterministic models. They encode discrete, fixed latent variables.
- Variational autoencoders are probabilistic models. They encode **continuous, probabilistic** latent variables.

# Probabilistic Latent Variable

- VAEs encode latent variables of training data not as a fixed discrete value $\boldsymbol{z}$, but as a continuous range of possibilities expressed as a probability distribution $p(\boldsymbol{z})$ (prior).
- For each latent attribute of training data, VAEs encode two different latent vectors: a vector of means $\boldsymbol{\mu}$ and a vector of standard deviations $\boldsymbol{\sigma}$ that represent the range of possibilities for each latent variable and the expected variance within each range of possibilities.
- By randomly sampling from within this range of encoded possibilities, VAEs can synthesize new data samples that resemble the original training data.

# Kullback-Leibler Divergence

- Reconstruction loss alone can result in an irregular encoding of latent space that overfits the training data.
- VAEs incorporate another regularisation term: KL divergence.

# Kullback-Leibler Divergence

- The latent space must exhibit two types of regularity:
  - Continuity: Nearby points in latent space should yield similar content when decoded.
  - Completeness: Any point sampled from the latent sapce should yield meaningful content when decoded.

Minimising only recsontruction loss does not incentivise the model to organise the latent space in any particular way.

# Kullback-Leibler Divergence

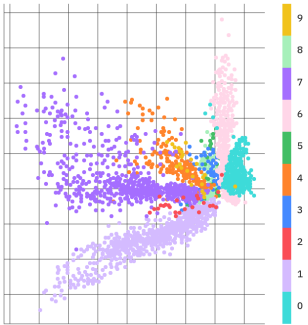
- KL divergence compares two probability distributions:

$$D_{KL}(P\|Q) - \sum_i P(i)\,(\log P(i) - \log Q(i))$$

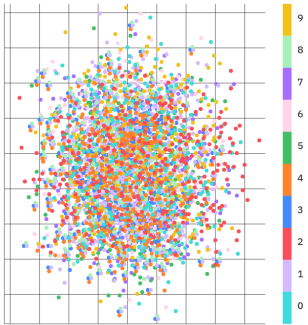
- We minimise the KL divergence between the learned distribution of latent variables and a simple Gaussian distribution forces the learned encoding of latent variables to follow a normal distribution.

# Kullback-Leibler Divergence

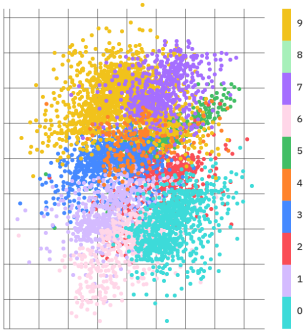
Only reconstruction loss




Only KL divergence




Reconstruction loss and KL divergence



# Introduction

VAE can not only accurately reconstruct the exact original input, but also use variational inference to generate new data samples that resemble the original input data.

# Drawbacks

- Potentially huge variance
- Posterior collapse
- Is our nature really continuous?

# VQ-VAE

# Discrete Latent Variables

We define a latent embedding space $e \in \mathbb{R}^{K \times D}$ where $K$ is the size of the discrete latent space and $D$ is the dimensionality of each latent embedding vector $e_i$ for $i = 1, 2, \cdots, K$.

# Discrete Latent Variables

The model takes an input $x$, that is passed through an encoder, producing $z_e(x)$.

# Discrete Latent Variables

The discrete latent variables $z$ are then calculated by a nearest neighbour look-up using the shared embedding space $e$:

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

# Discrete Latent Variables

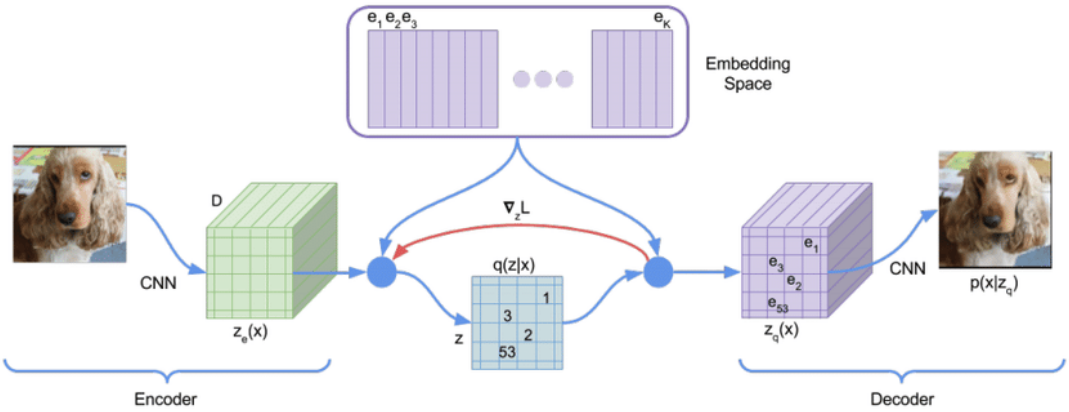
The proposal distribution $q(z = k \mid x)$ is deterministic, and by defining a simple uniform prior over $z$ we have a constant KL divergence ($\log K$).

# Discrete Latent Variables

The representation $z_e(x)$ is passed through the discretisation bottleneck following by mapping onto the nearest element of embedding $e$:

$$z_q(x) = e_k \text{ where } k = \arg\min_j \|z_e(x) - e_j\|_2$$

# Discrete Latent Variables



# Learning

Gradient descent?

# Learning

We just copy gradients from decoder input $z_q(x)$ to encoder output $z_e(x)$.
Subgradients also work.

# Learning

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

# Learning