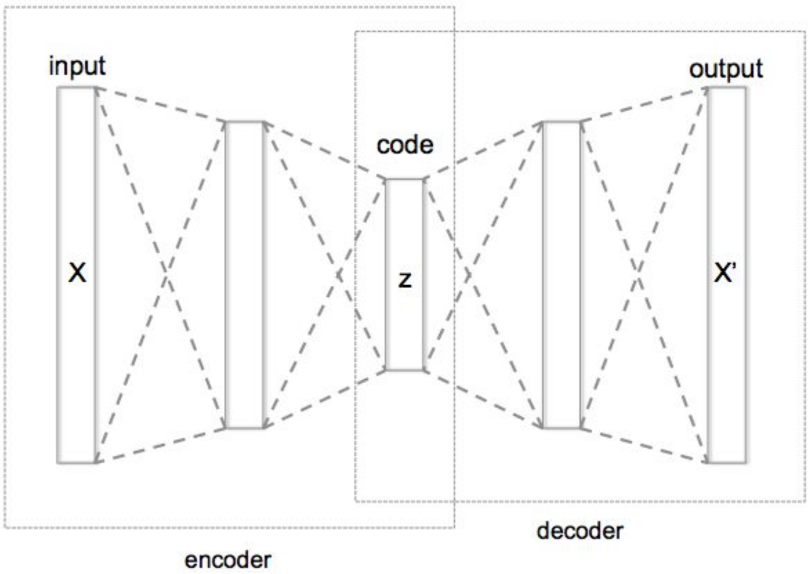
Variational Autoencoder

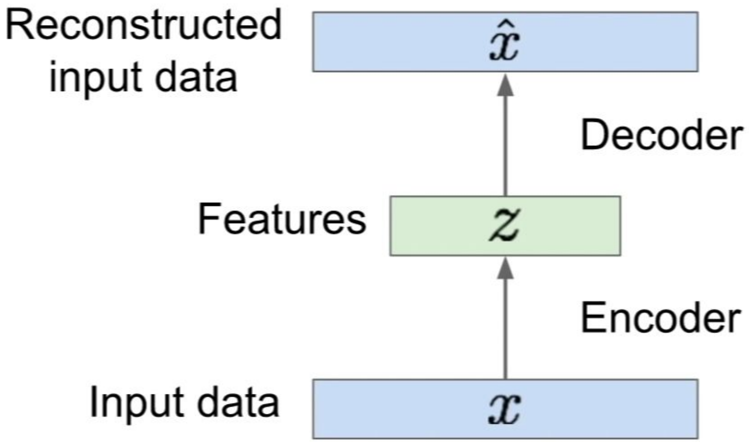
William Hu

Autoencoder

- A type of neural network used to learn efficient data encodings in unsupervised manner.
- Consists of two networks, encoder and decoder.
- Dimensionality reduction
  - Transform data in high-dimensional space to low-dimensional space data.

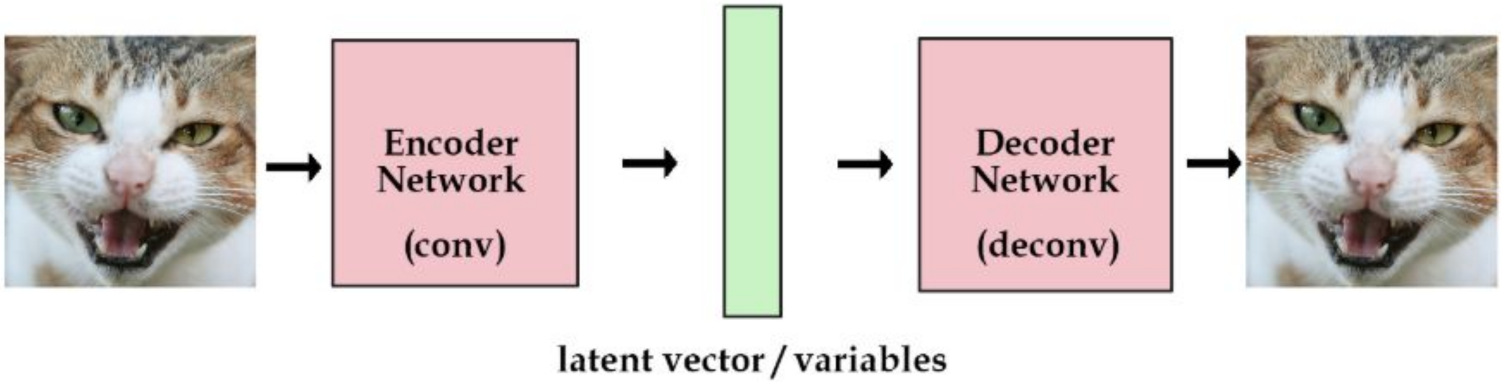


Autoencoder



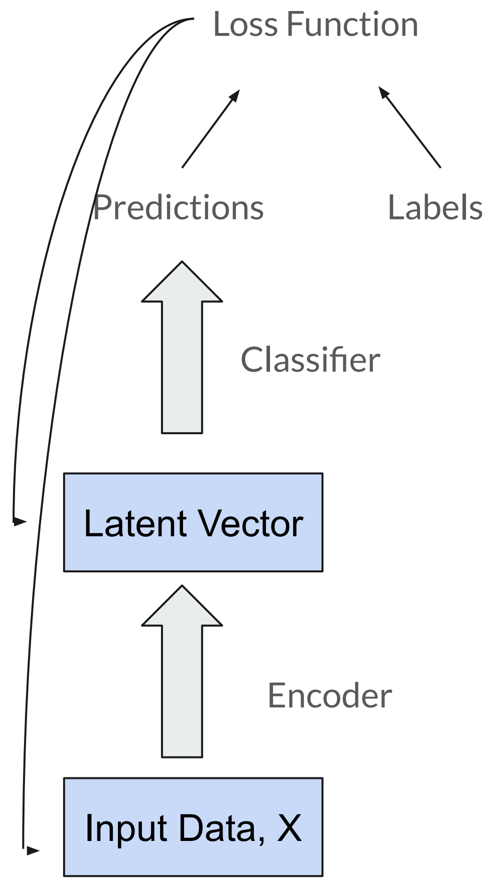
- Input as data in high dimensional space.
- Encoder to reduce its dimension.
- Decoder to reconstruct the original data.
- Possible networks, including:
  - Linear layers connected with nonlinearity (activation functions).
  - Dense, fully connected layers.
  - Conv and DeConv
  - LSTM, RNN, GRU etc.
- L2 loss to measure the difference between the input and the output.
- Can be very useful when we are trying to extract important features.

Autoencoder



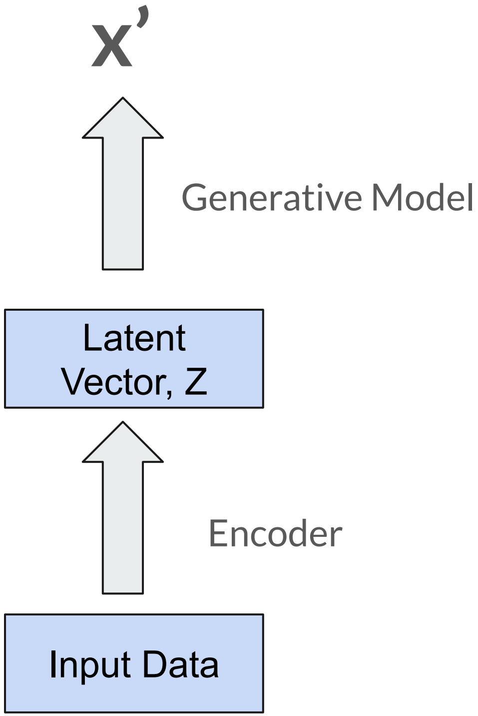
Autoencoder

- Can be also applied in supervised learning problem.
- Remove the decoder part, use only the encoder as feature extractor.
- Combine with supervised models, fine-tune them jointly.
- Large amount of unlabeled data together with labelled data.



What AE is not good at

- What if we want to generate new data, for example, new images?
- Given the input, we generate a latent representation z.
- What we trying to do is to sample x prime from prior z.
  - Z is a latent vector that contains some factors of the desired x.
  - If we are generating human faces, then z might contain the information about the eyebrow, about how high the nose is.

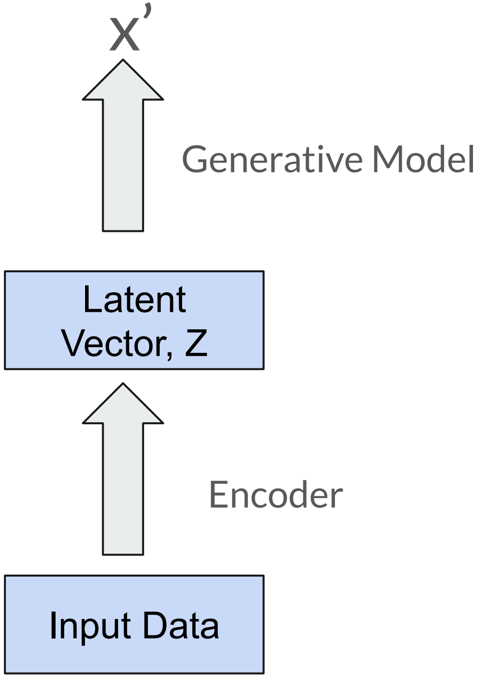


Generative Model using AE

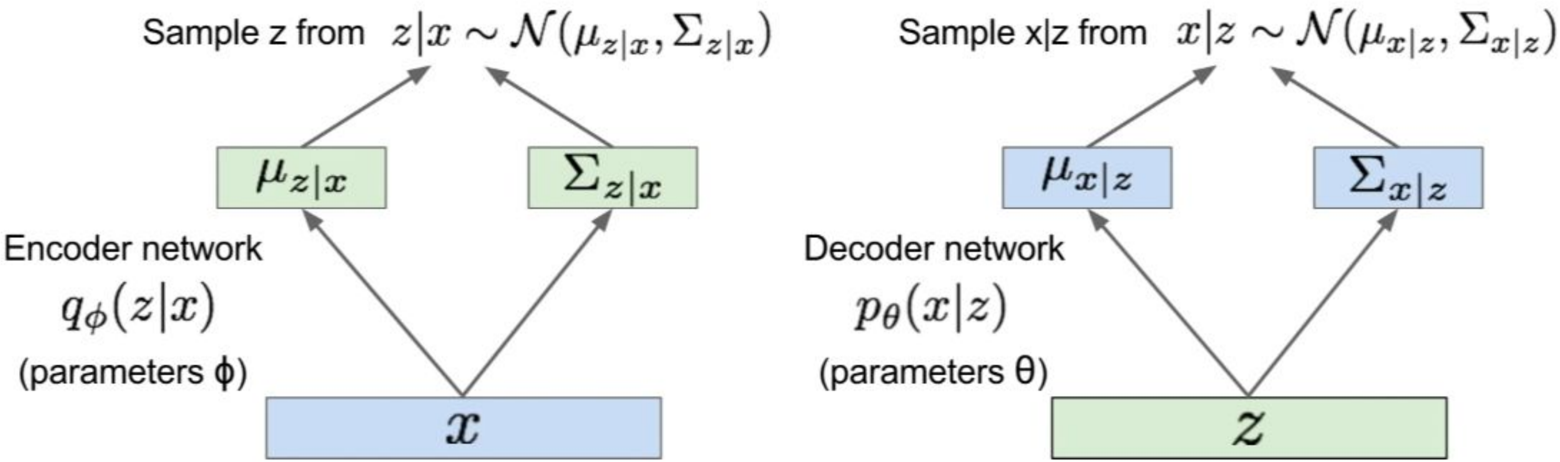
- We want to estimate the true parameter θ^* of this generative model.
- Assume that prior p(z) is just gaussian distribution.
- The conditional probability of p(x'|z), this we will use a network to represent.
- Learn the parameters that maximize the likelihood of the training data.

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

- **Problem:** it is intractable to compute p(x|z) for every z.

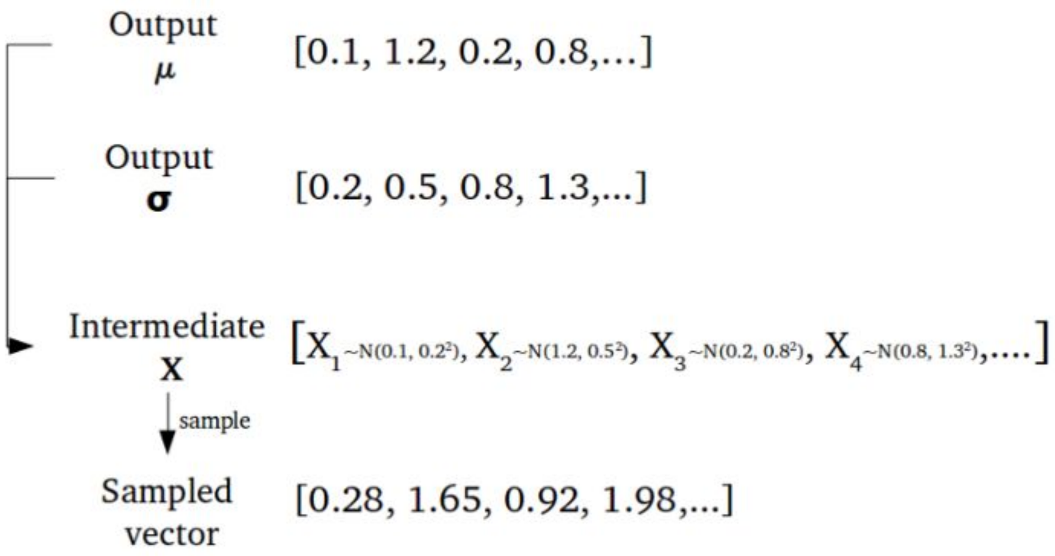


Variational Autoencoder



Variational Autoencoder

- We are sample from the distribution, everytime we will get different x.



Kullback Leibler divergence, also known as KL term, is used to measure the difference of two probability distribution.

How the problem is fixed

$$
\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \qquad (p_\theta(x^{(i)}) \text{ Does not depend on } z) \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Bayes' Rule)} \\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \qquad \text{(Multiply by constant)} \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \qquad \text{(Logarithms)} \\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))
\end{aligned}
$$

Can compute estimate of this term through sampling.

This KL term has nice closed-form solution (between two Gaussian distribution)

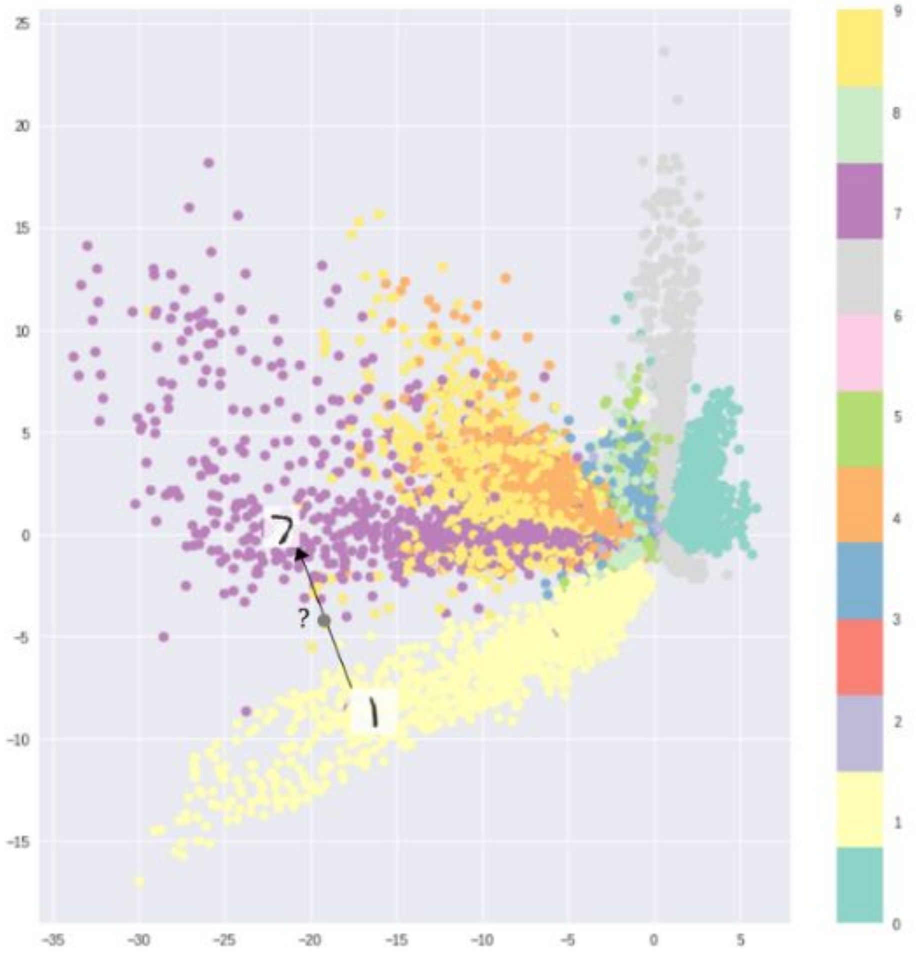
KL term by definition is always greater or equal to 0.

A more intuition way of thinking math

- Is to think through graph.

Autoencoder

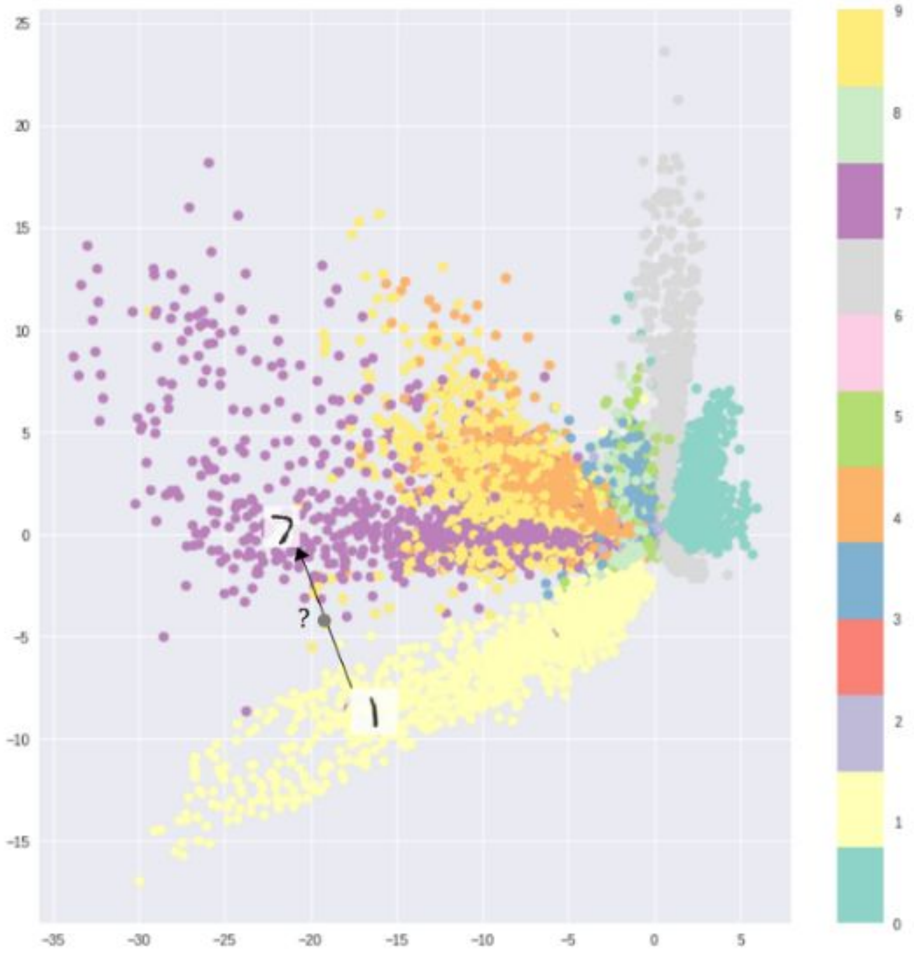
- The latent space of autoencoder may not be continuous, or allow easy interpolation.
- That is a problem for generation.



Optimizing purely for reconstruction loss

Autoencoder

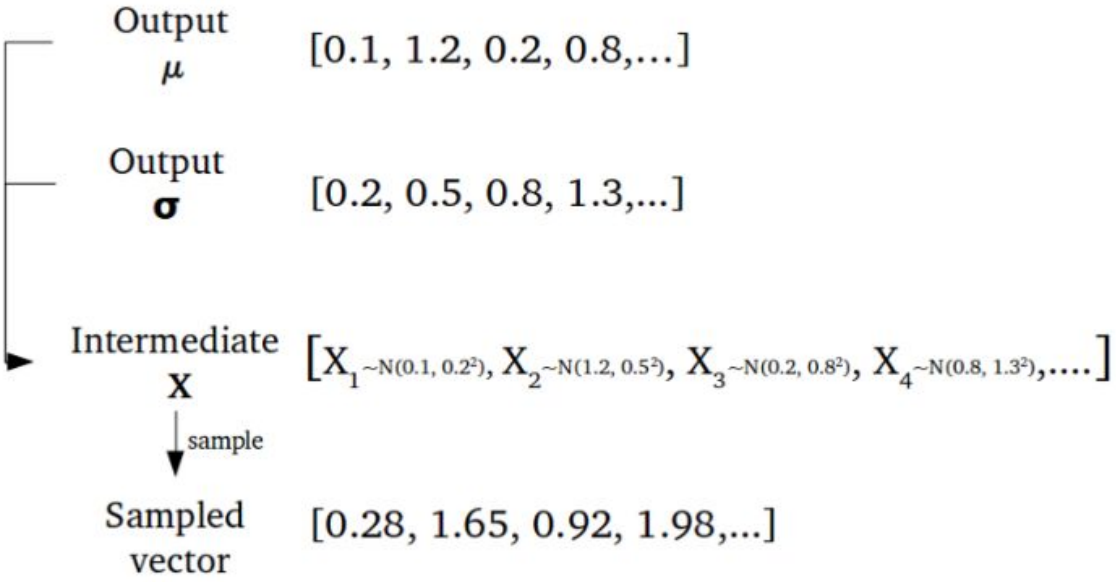
- If you generate from gap area, your generative network has no idea what to generate



Optimizing purely for reconstruction loss

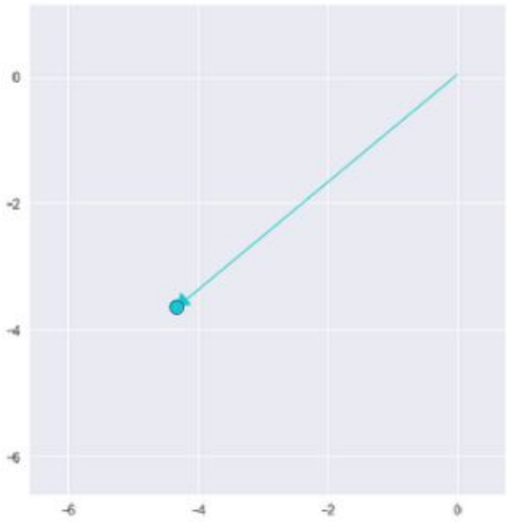
Variational Autoencoder

- Encoder network is going to give two vector of size n, one is the mean, and the other is standard deviation/variance.
- Stochastica generation, for the same input, mean and variance is the same, the latent vector is still different due to sampling.

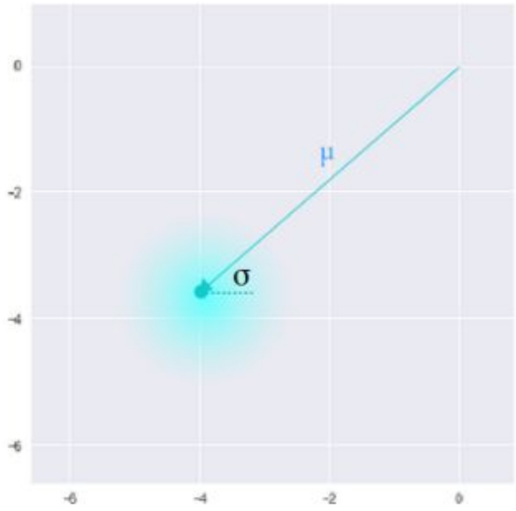


VAE VS AE

- Sample our latent space vector from a distribution.
- Less gap between each cluster.



Standard Autoencoder
(direct encoding coordinates)

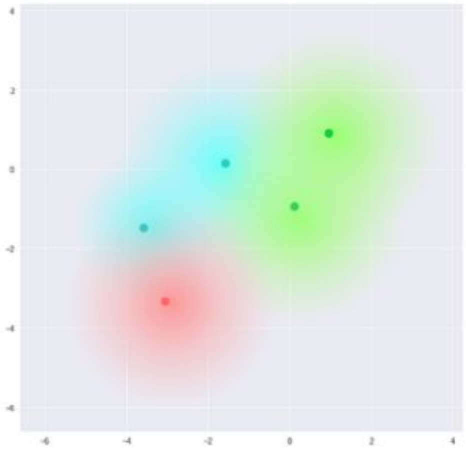


Variational Autoencoder
(μ and σ initialize a probability distribution)

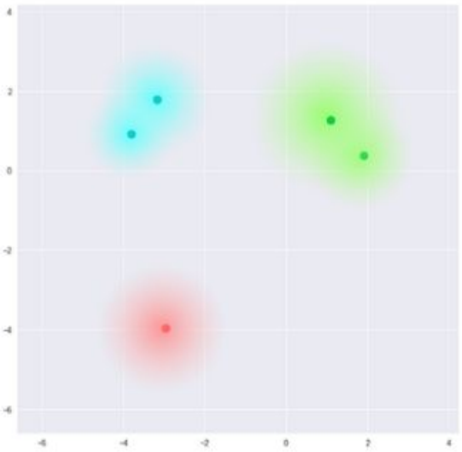
Still problem

- More smooth latent space on local scale.
- We overlap between samples that are not very similar so we can interpolate between classes.

- Discrete clusters, still have gap
- Still chance that network does not know what to generate.



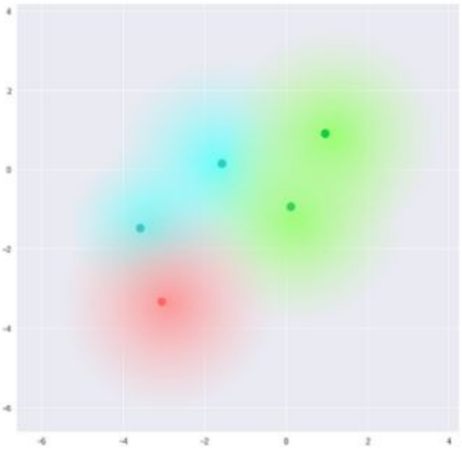
What we require



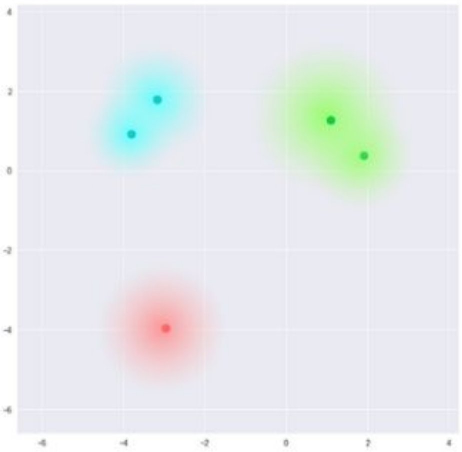
What we may inadvertently end up with

Still problem

- No limitations on mean and variance.
- The encoder can learn to generate very different mean for different classes, and then minimize the variance.
- Less uncertainty for the decoder network.



What we require



What we may inadvertently end up with

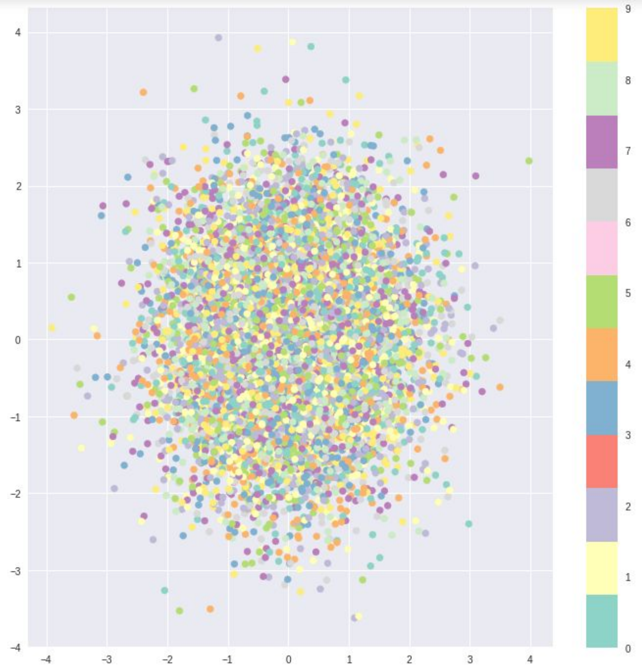
$$\sum_{i=1}^{n} \sigma_i^2 + \mu_i^2 - \log(\sigma_i) - 1$$

KL Divergence

- Measure the difference of two probability distribution.
- Optimize the KL divergence means to optimize probability distribution parameters to closely resemble that of the target distribution.
- KL divergence of component $X_i \sim N(\mu_i, \sigma_i^2)$ in X, and the standard normal.

KL Divergence

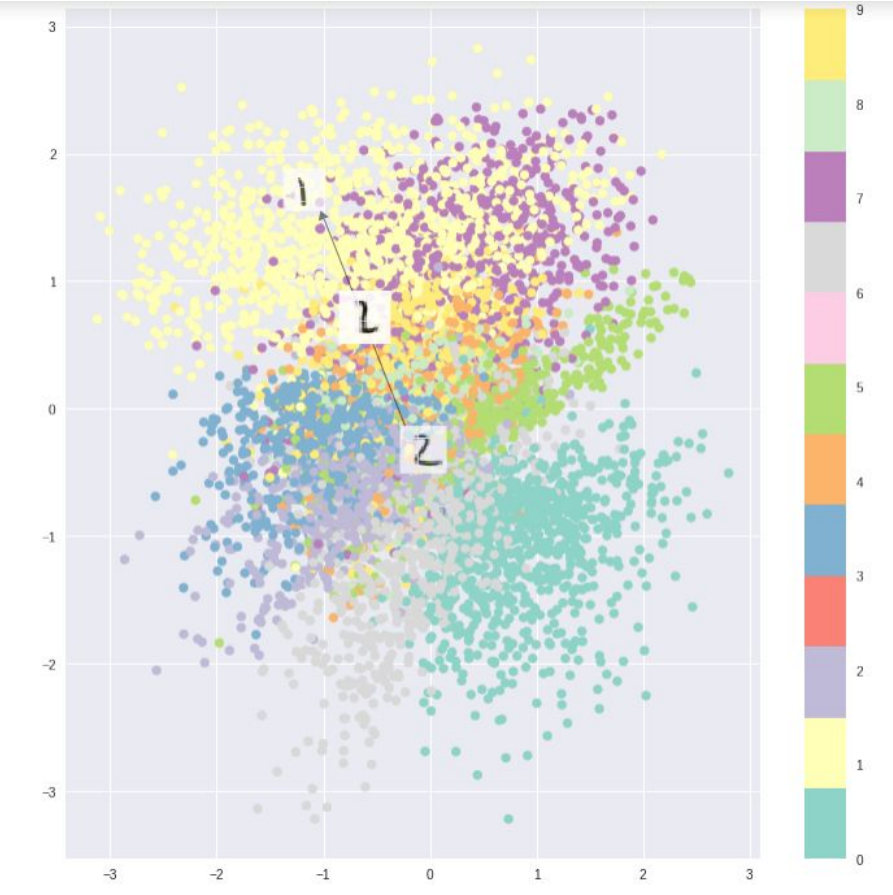
- Encourage the encoder to distribute all encodings evenly around the center of the latent space.
- No difference between different classes, no similarity within the same class.



Optimizing using pure KL divergence loss

KL + Reconstruction loss

- cluster-forming nature of the reconstruction loss
- dense packing nature of the KL loss
- no sudden gaps between cluster, will be a mixture of different features that the decoder can understand.



Optimizing using both reconstruction loss and KL divergence loss

VAE

VAE

- Celebrity face generation.