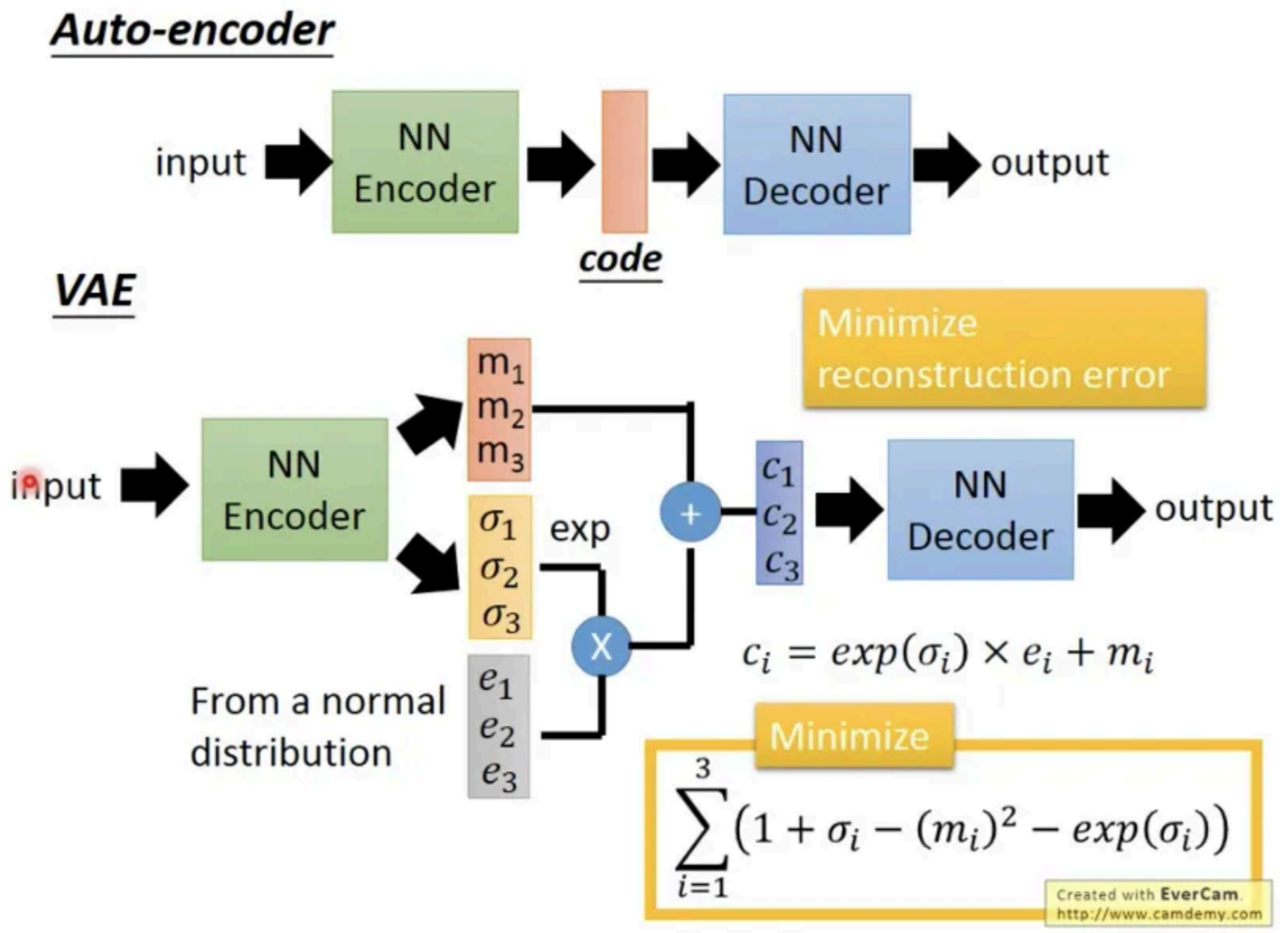
1. Variational Autoencoder

Advantage:
VAE is a generative statical model, while AE can be viewed as a data compressor and decompressor. So when you want to generate new data, VAE is a better choice.

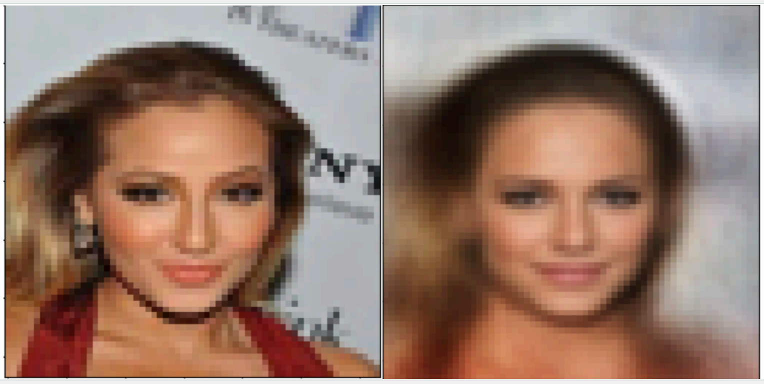
Disadvantage:
Compared to AE, VAE adds complexity and computational cost associated with training and inference.


Auto-encoder
input
NN Encoder
code
NN Decoder
output
VAE
input
NN Encoder
m_1 m_2 m_3
σ_1 σ_2 σ_3
exp
e_1 e_2 e_3
From a normal distribution
c_1 c_2 c_3
NN Decoder
output
Minimize reconstruction error
$c_i = exp(\sigma_i) \times e_i + m_i$
Minimize
$\sum_{i=1}^{3}(1 + \sigma_i - (m_i)^2 - exp(\sigma_i))$
Created with EverCam http://www.camdemy.com

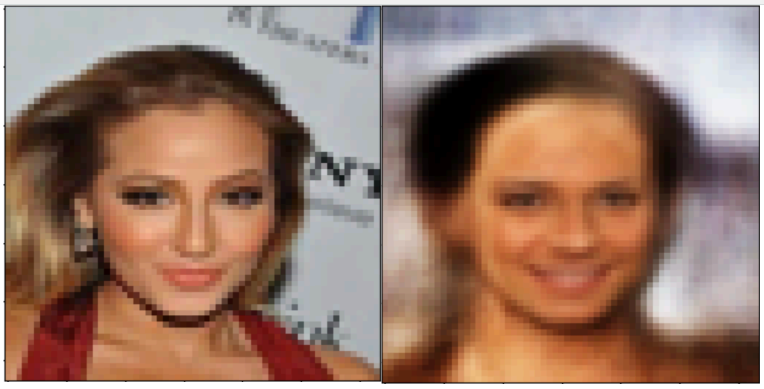
Paper: https://arxiv.org/abs/1312.6114
Reference: https://medium.com/ml-note/autoencoder-—-認識與理解-725854ab25e8

2. model在下一頁

```
# reference: https://github.com/yuting-NTU/NTU-ML2022-Spring/blob/main/r10922196_hw8/ML2022Spring_HW8.ipynb
sample = train_dataset[1000]
sample = sample.reshape(1,3,64,64)
sample_ori = np.array(sample[0].permute(1,2,0))
sample_ori = (sample_ori + 1)/2*255
plt.imshow(sample_ori.astype(np.uint8))
plt.show()
model.eval()
with torch.no_grad():
    img = sample.cuda()
    img = img.reshape(img.shape[0], -1)
    x = model.encoder(img)
    x[0][1] = x[0][1]*2
    output = model.decoder(x)
    output = output.reshape(3,64,64)
    output = output.cpu().permute(1,2,0)
    output = np.array(output)
    output = (output + 1)/2*255
    plt.imshow(output.astype(np.uint8))
```



```
# reference: https://github.com/yuting-NTU/NTU-ML2022-Spring/blob/main/r10922196_hw8/ML2022Spring_HW8.ipynb
sample = train_dataset[1000]
sample = sample.reshape(1,3,64,64)
sample_ori = np.array(sample[0].permute(1,2,0))
sample_ori = (sample_ori + 1)/2*255
plt.imshow(sample_ori.astype(np.uint8))
plt.show()
model.eval()
with torch.no_grad():
    img = sample.cuda()
    img = img.reshape(img.shape[0], -1)
    x = model.encoder(img)
    x[0][0] = x[0][0]*10
    output = model.decoder(x)
    output = output.reshape(3,64,64)
    output = output.cpu().permute(1,2,0)
    output = np.array(output)
    output = (output + 1)/2*255
    plt.imshow(output.astype(np.uint8))
```



上面那張把第1維*2後幾乎沒影響（至少我看不出來）。
下面那張把第0維*10後看起來像露齒笑的毛澤東，很多原本的特徵不見了。

```python
class fcn_autoencoder(nn.Module):
    def __init__(self):
        super(fcn_autoencoder, self).__init__()
        self.encoder = nn.Sequential(
            nn.Linear(64 * 64 * 3, 2048),
            nn.ReLU(),
            nn.Linear(2048, 512),
            nn.ReLU(),
            nn.Linear(512, 128),
        )

        self.decoder = nn.Sequential(
            nn.Linear(128, 512),
            nn.ReLU(),
            nn.Linear(512, 2048),
            nn.ReLU(),
            nn.Linear(2048, 64 * 64 * 3),
            nn.Tanh()
        )

    def forward(self, x):
        x = self.encoder(x)
        x = self.decoder(x)
        return x
```