Variational AutoEncoder
:변이형 오토인코더

강예준

https://youtu.be/nmDXpI8lGOs

한성대학교 HANSUNG UNIVERSITY

CryptoCraft LAB

AutoEncoder

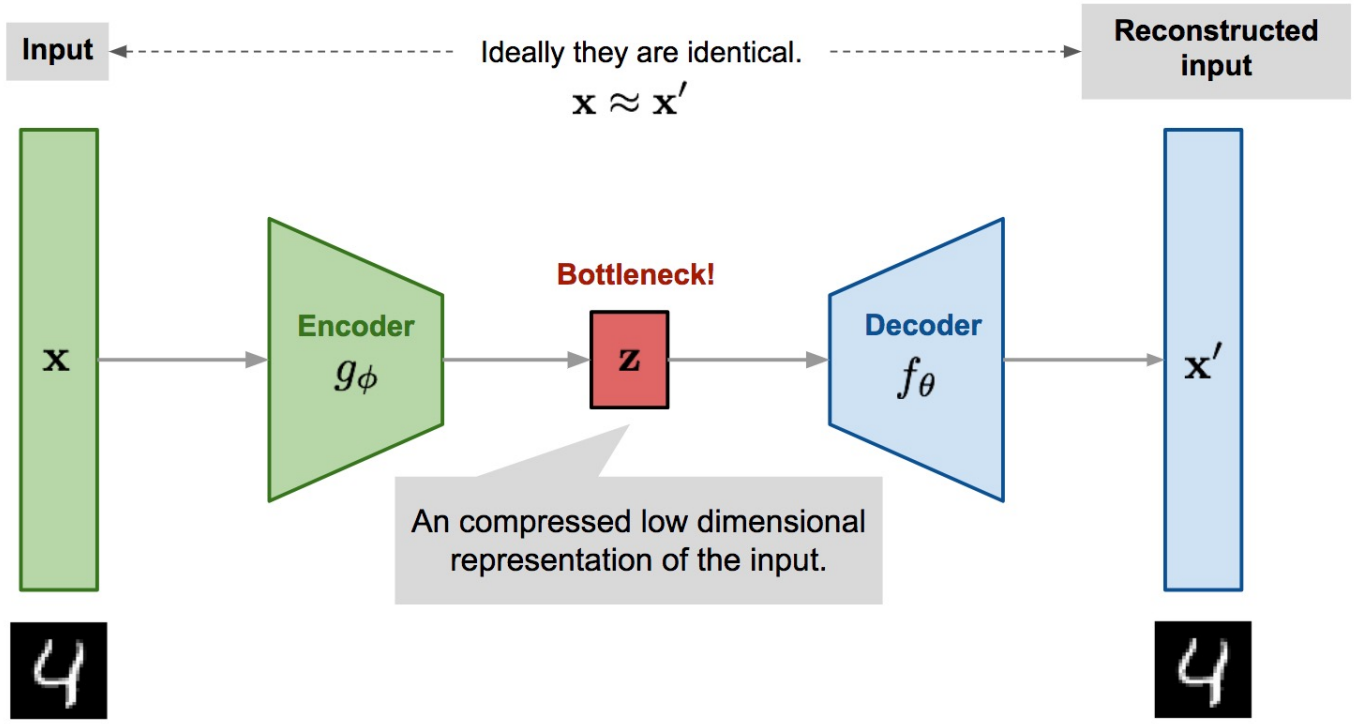
AutoEncoder와 Variational AutoEncoder의 차이점

Variational AutoEncoder

Variational AutoEncoder in Pytorch

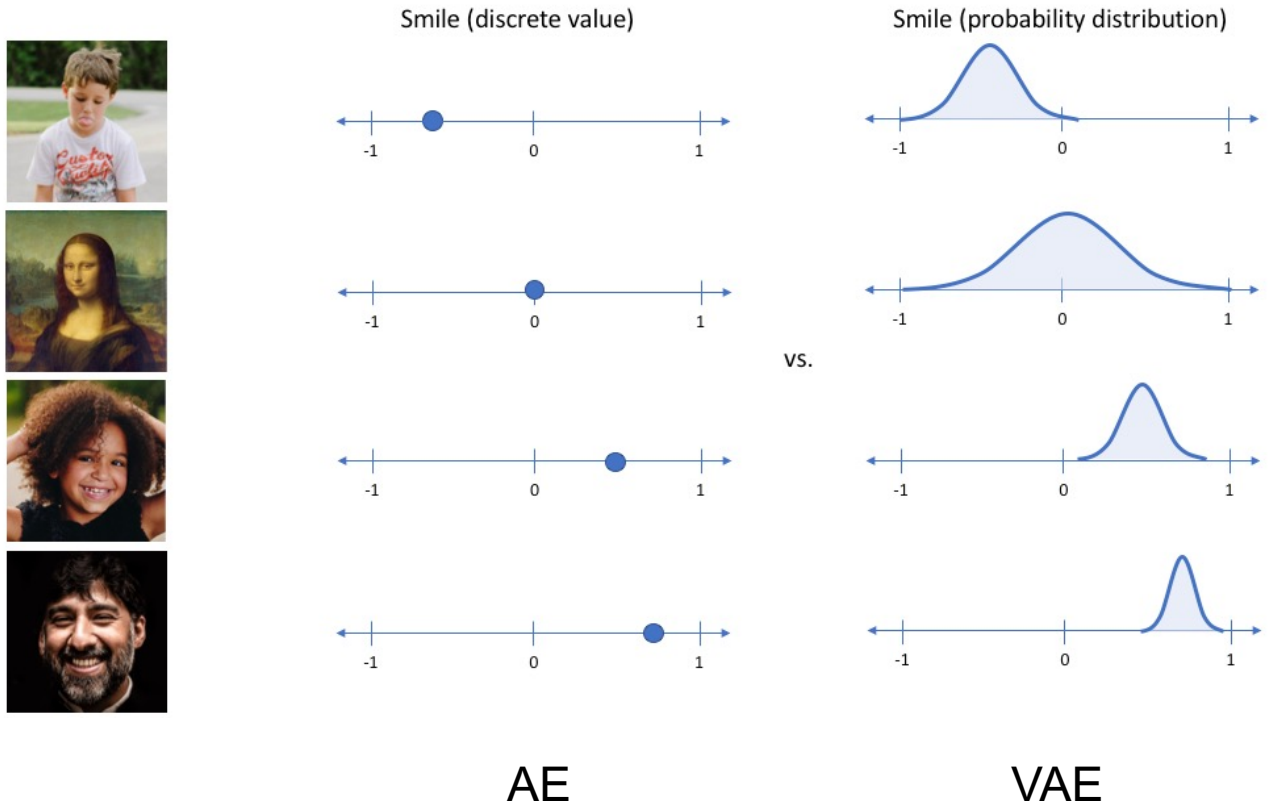
AutoEncoder

- **AutoEncoder**
  - Encoder : 입력한 데이터 x를 특징 값으로 변환하는 신경망
  - Decoder : 인코더 과정을 거쳐 나온 출력값을 디코더에 입력 시, 인코더 과정을 거치기 전 데이터인 x값 출력



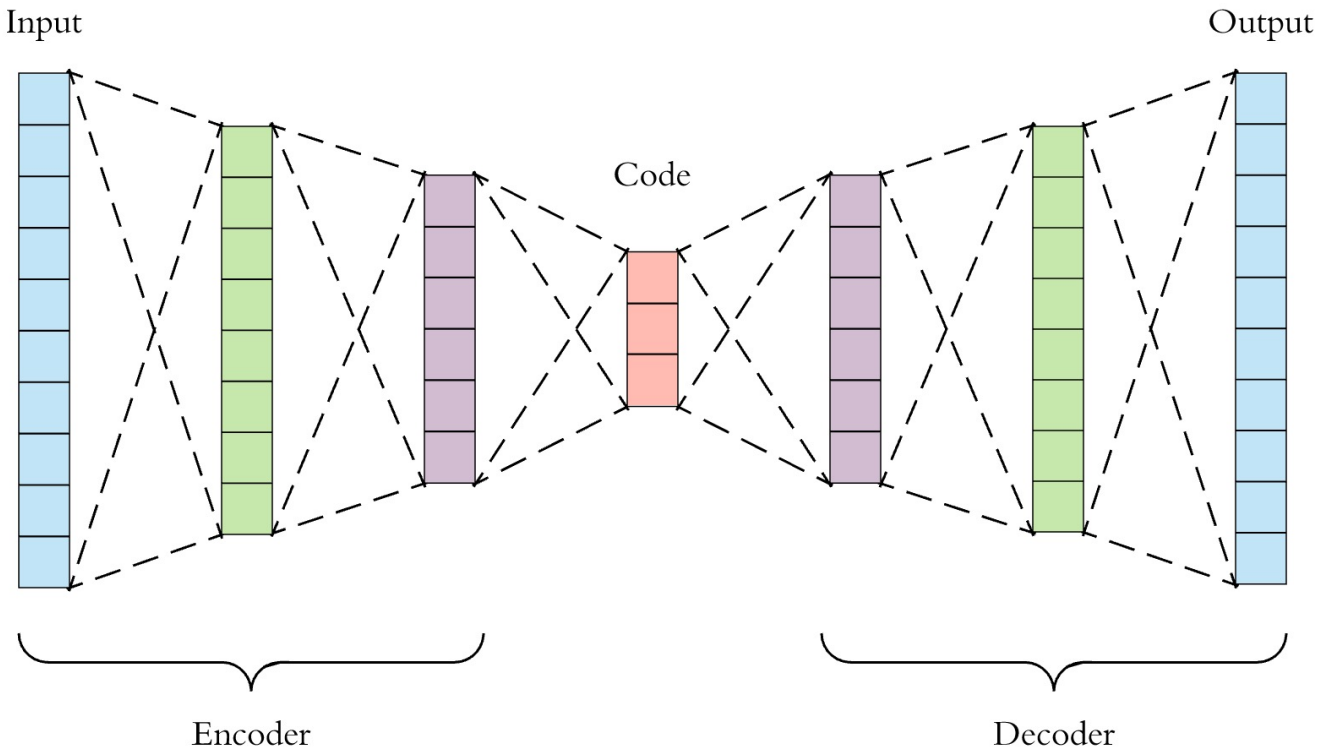
AutoEncoder와 Variational AutoEncoder

- **AutoEncoder와 Variational AutoEncoder의 차이점**
  - AutoEncoder : 잠재변수를 고정된 값(표현 벡터)으로 표현
  - Variational AutoEncoder : 잠재변수를 가우시안 확률분포(정규분포) 값의 범위로 제공



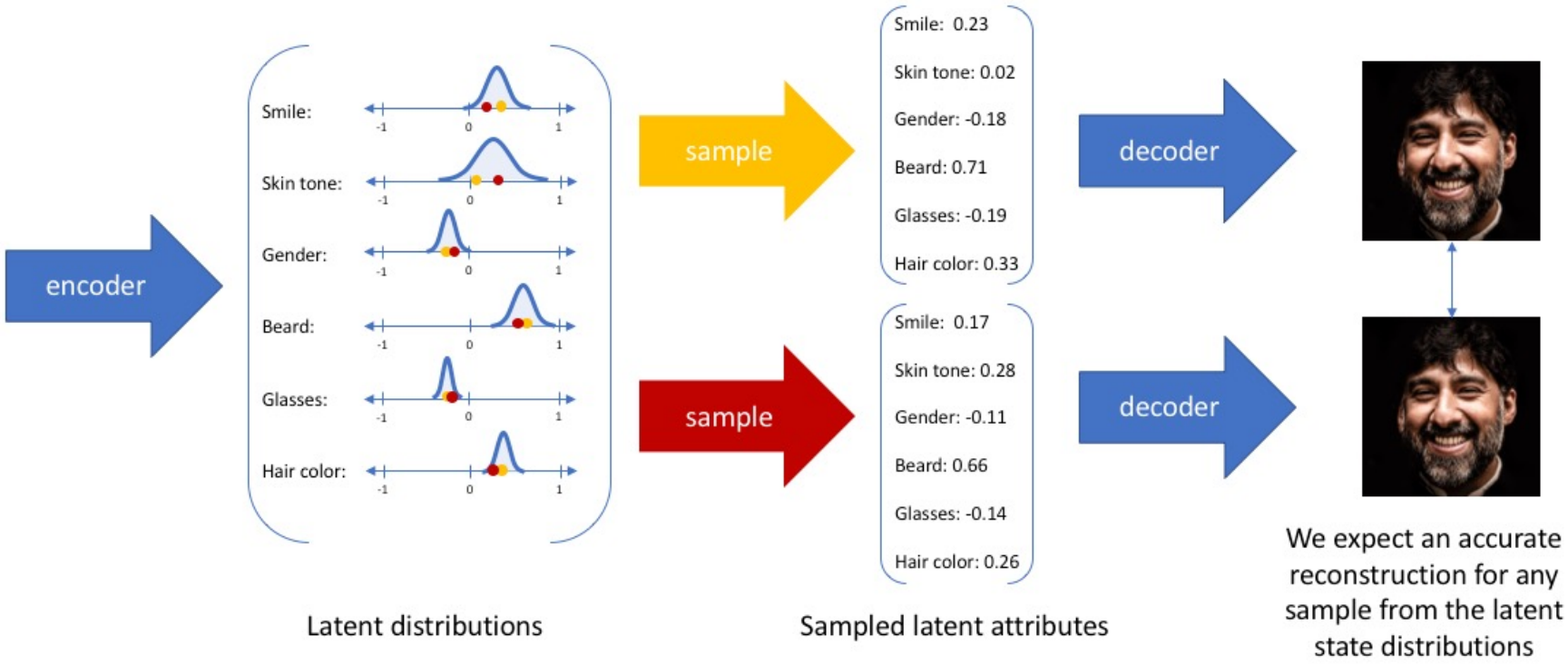
AutoEncoder와 Variational AutoEncoder

- **AutoEncoder와 Variational AutoEncoder의 차이점**
  - AutoEncoder의 목적 : 입력 데이터의 압축을 통해, 데이터의 중요한 **특징을 추출** (Manifold Learning)
  - Variational Auto Encoder의 목적 : 디코더를 학습 시켜 새로운 **이미지를 생성** (Generative Model)



Variational AutoEncoder

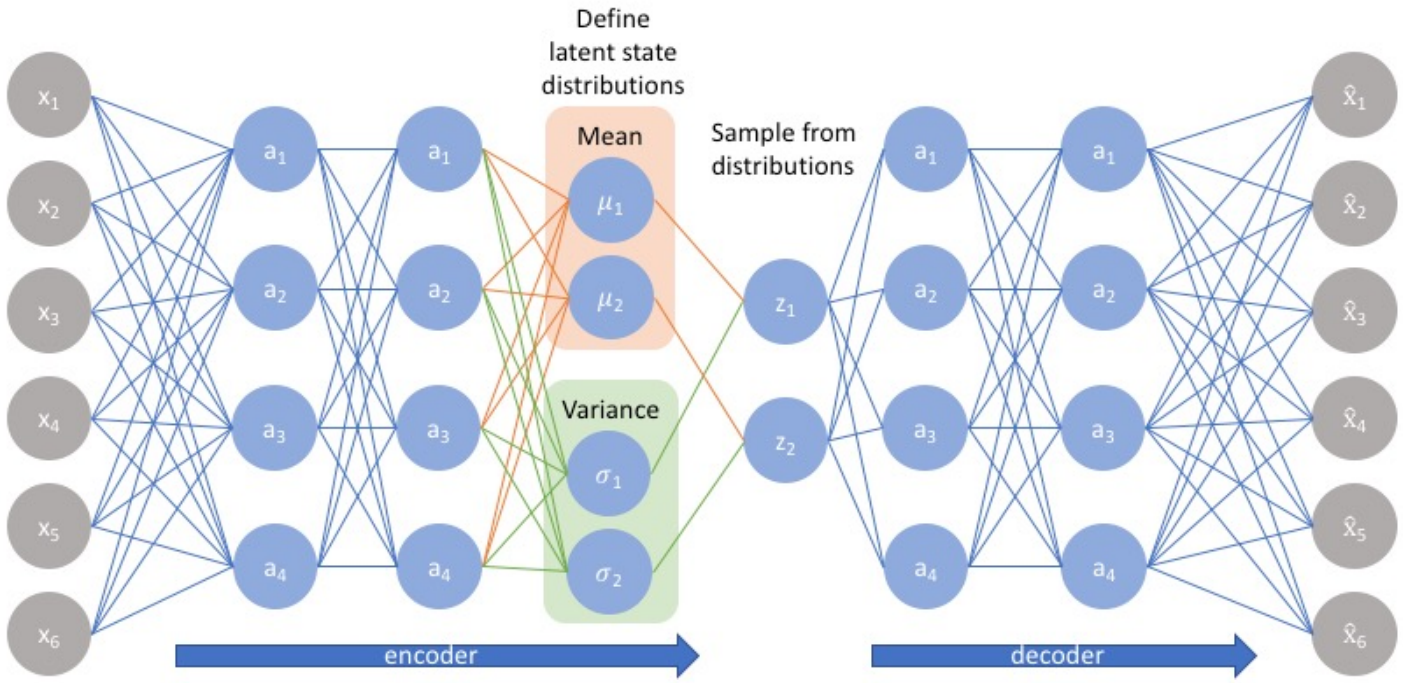
- **Variational AutoEncoder**
  - Encoder : 입력한 데이터 x를 정규분포(평균,표준편차)로 변환하는 신경망
  - Decoder : 인코더 과정을 거쳐 나온 출력값을 디코더에 입력 시, 인코더 과정을 거치기 전 데이터인 x값 출력



Variational AutoEncoder

- **Variational AutoEncoder**
  - Encoder가 만들어낸 z의 평균과 표준편차를 모수로 하는 **정규분포**로부터 랜덤으로 샘플링하여 잠재변수 Z를 생성
  - 잠재변수 z를 통해, 디코더가 새로운 데이터를 생성 (생성형 모델)

    → 대표적인 생성형 모델 GAN과 비교하였을 때, 출력이 흐릿하지만 평가기준이 명확하며 학습이 안정적



Loss Function of VAE

- **Loss Function of Decoder (Reconstruction Error)**
  - 원본 이미지와 생성된 이미지와의 오차
  - 잠재변수 z값으로 얼마나 원본 이미지와 유사한 이미지를 잘 생성하는가?

- **Loss Function of Encoder (Regularization)**
  - 원본 이미지 확률분포와 잠재변수 확률분포 사이의 오차
  - 잠재변수 z값이 원본 이미지의 확률분포와 얼마나 유사한가?

$$L_i(\phi, \theta, x_i) = \underbrace{-\mathbb{E}_{q_\phi(z|x_i)}\left[\log\left(p_\theta(x_i|z)\right)\right]}_{\text{Reconstruction Error}} + \underbrace{KL\left(q_\phi(z|x_i)||p(z)\right)}_{\text{Regularization}}$$

Variational AutoEncoder in Pytorch

```python
# Model
class VAE(nn.Module):
    def __init__(self,latent_dim=20,hidden_dim=500):
        super(VAE,self).__init__()
        self.fc_e = nn.Linear(784,hidden_dim)
        self.fc_mean = nn.Linear(hidden_dim,latent_dim)
        self.fc_logvar = nn.Linear(hidden_dim,latent_dim)
        self.fc_d1 = nn.Linear(latent_dim,hidden_dim)
        self.fc_d2 = nn.Linear(hidden_dim,784)

    def encoder(self,x_in):
        x = F.relu(self.fc_e(x_in.view(-1,784)))
        mean = self.fc_mean(x)
        logvar = self.fc_logvar(x)
        return mean, logvar

    def decoder(self,z):
        z = F.relu(self.fc_d1(z))
        x_out = F.sigmoid(self.fc_d2(z))
        return x_out.view(-1,1,28,28)

    def sample_normal(self,mean,logvar):
        # Using torch.normal(means,sds) returns a stochastic tensor which we cannot backpropogate through.
        # Instead we utilize the 'reparameterization trick'.
        # http://stats.stackexchange.com/a/205336
        # http://dpkingma.com/wordpress/wp-content/uploads/2015/12/talk_nips_workshop_2015.pdf
        sd = torch.exp(logvar*0.5)
        e = Variable(torch.randn(sd.size())) # Sample from standard normal
        z = e.mul(sd).add_(mean)
        return z

    def forward(self,x_in):
        z_mean, z_logvar = self.encoder(x_in)
        z = self.sample_normal(z_mean,z_logvar)
        x_out = self.decoder(z)
        return x_out, z_mean, z_logvar

model = VAE()
```

➡ 인코더의 출력 : 평균, 표준편차

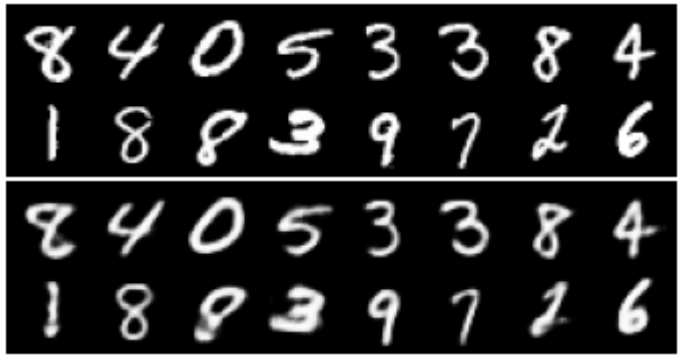
➡ 오토인코더와 달리 정규분포로부터 잠재변수 z 생성

https://github.com/nitarshan/variational-autoencoder

Variational AutoEncoder in Pytorch

```python
# Loss function
def criterion(x_out,x_in,z_mu,z_logvar):
    bce_loss = F.binary_cross_entropy(x_out,x_in,size_average=False)
    kld_loss = -0.5 * torch.sum(1 + z_logvar - (z_mu ** 2) - torch.exp(z_logvar))
    loss = (bce_loss + kld_loss) / x_out.size(0) # normalize by batch size
    return loss
```

Loss Function of VAE



VAE의 입력과 출력값

https://github.com/nitarshan/variational-autoencoder

Q & A