Deep Generative Model Tutorial
VAE & GAN

Taewook Nam
MLAI, KAIST
namsan@kaist.ac.kr

In this tutorial,

- Demonstrate **VAE & GAN** with toy data (MNIST)
- Describe brief outline of implementation
- You can go in detail with code : github.com/namsan96/1126_vaegan

code & slide :
github.com/namsan96/1126_vaegan

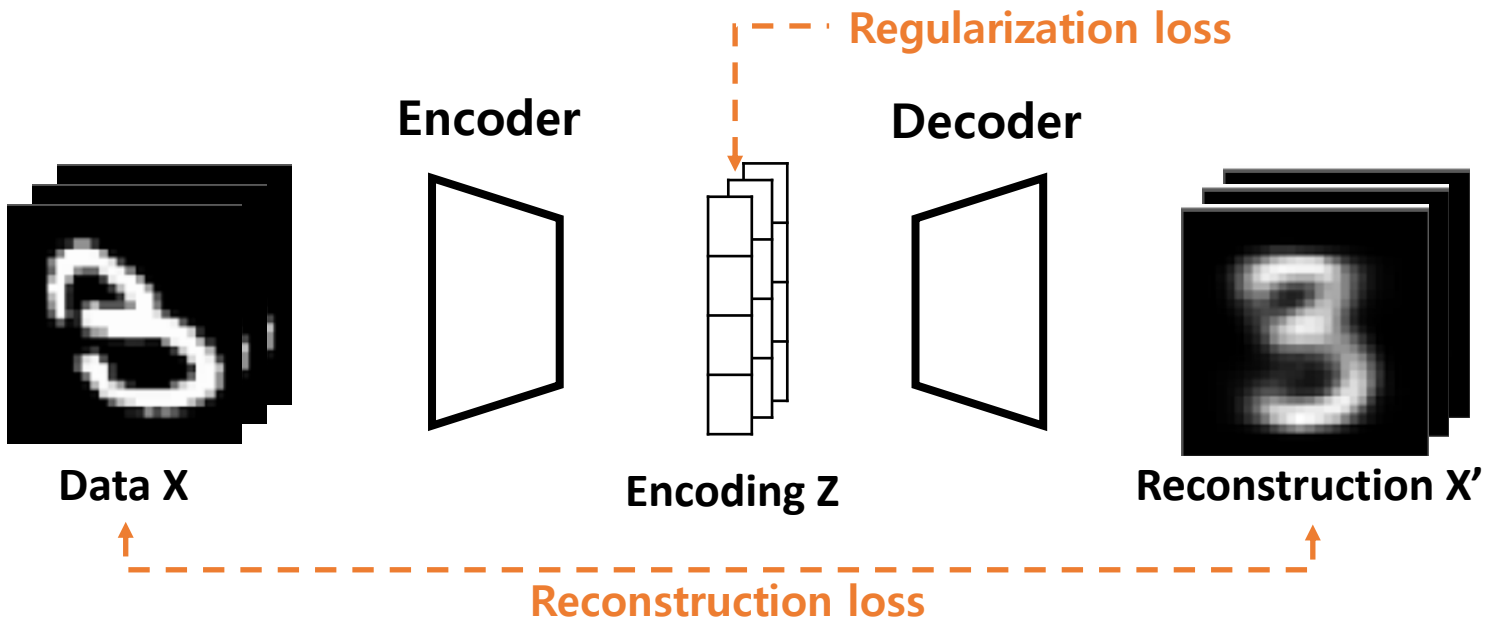
MLAI
Machine Learning & Artificial Intelligence

Variational Auto-Encoder (VAE)

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

VAE Overview



code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Implementation TODO

- Encoder & Decoder Network
- Loss function
- Training Loop
- Visualization

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

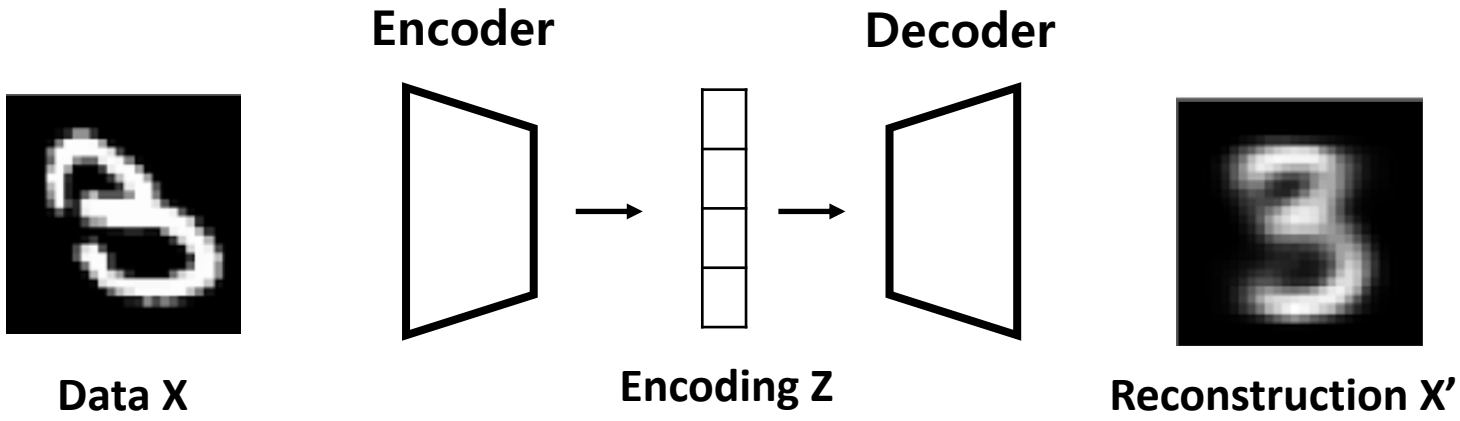
Implementation TODO

- **Encoder & Decoder Network**
- Loss function
- Training Loop
- Visualization

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

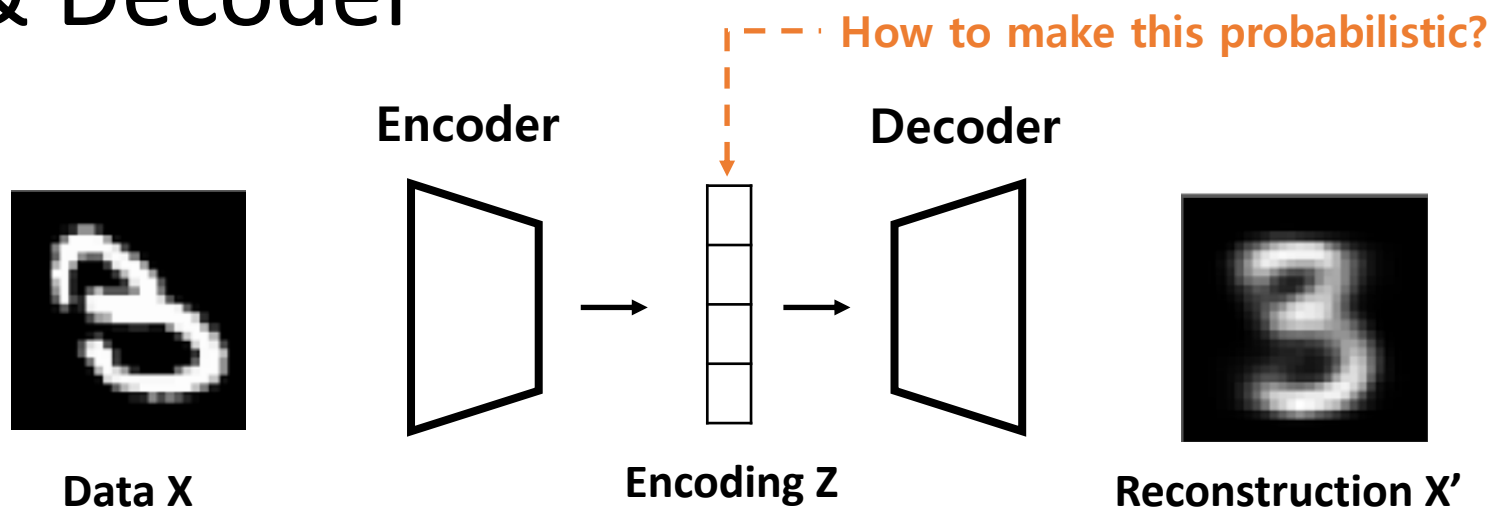
Encoder & Decoder



code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

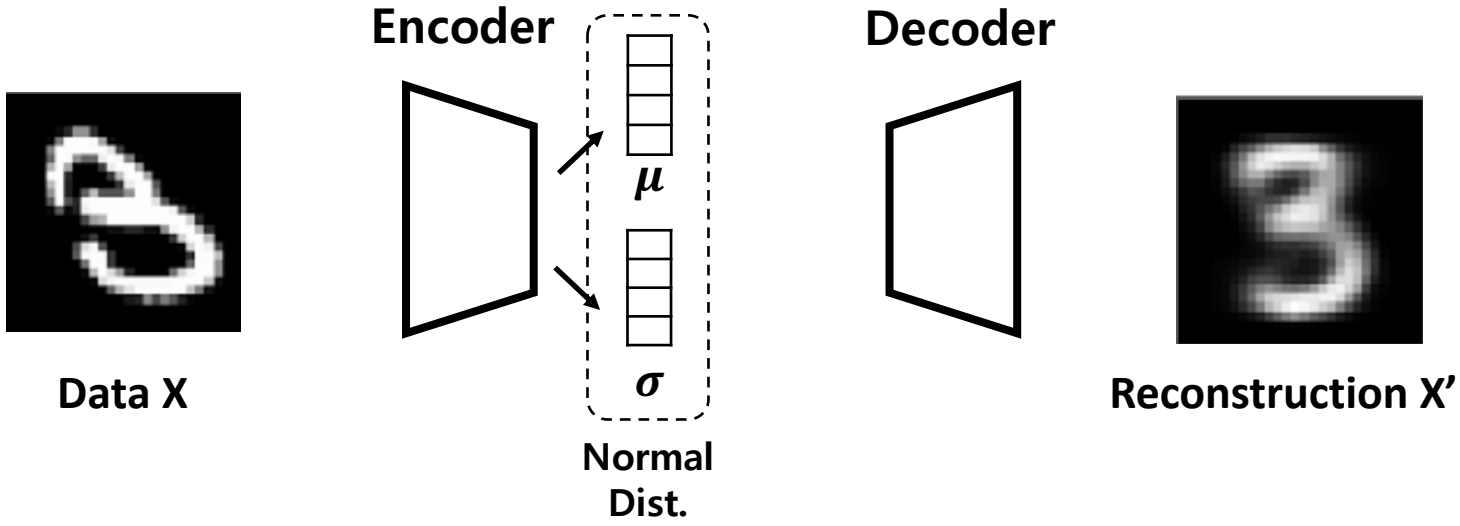
Encoder & Decoder



code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

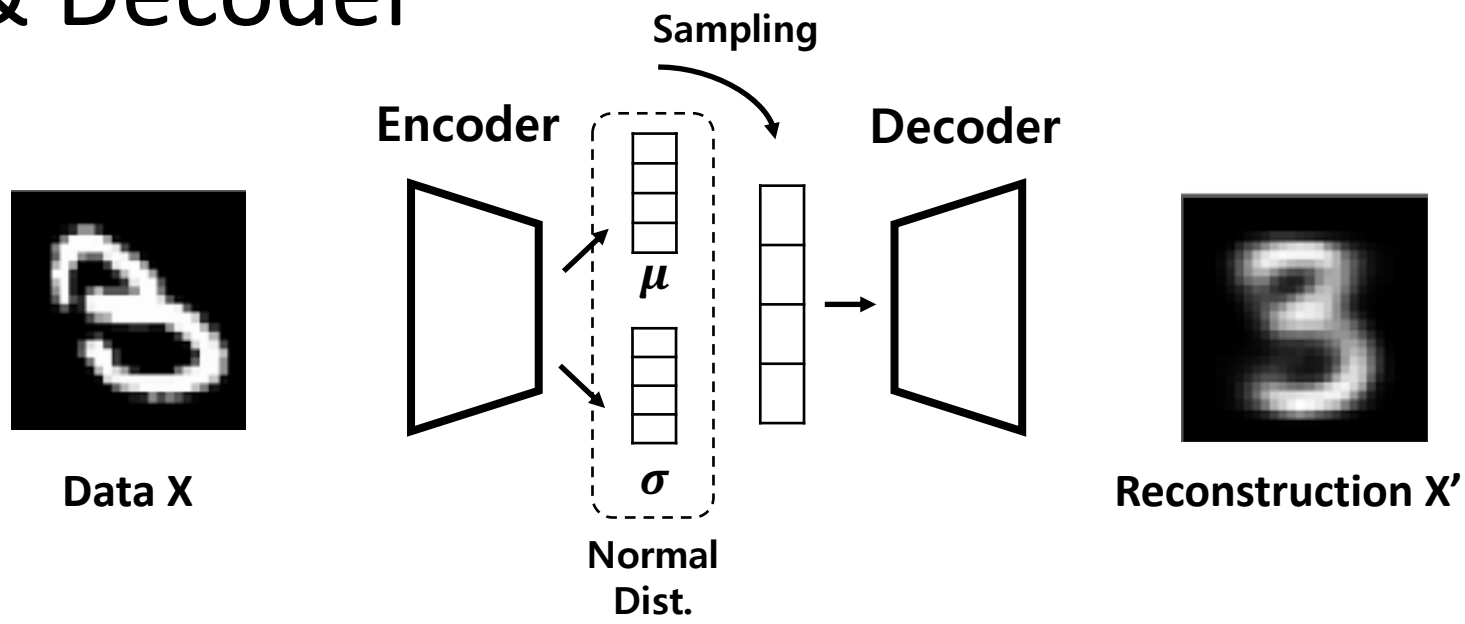
Encoder & Decoder



code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

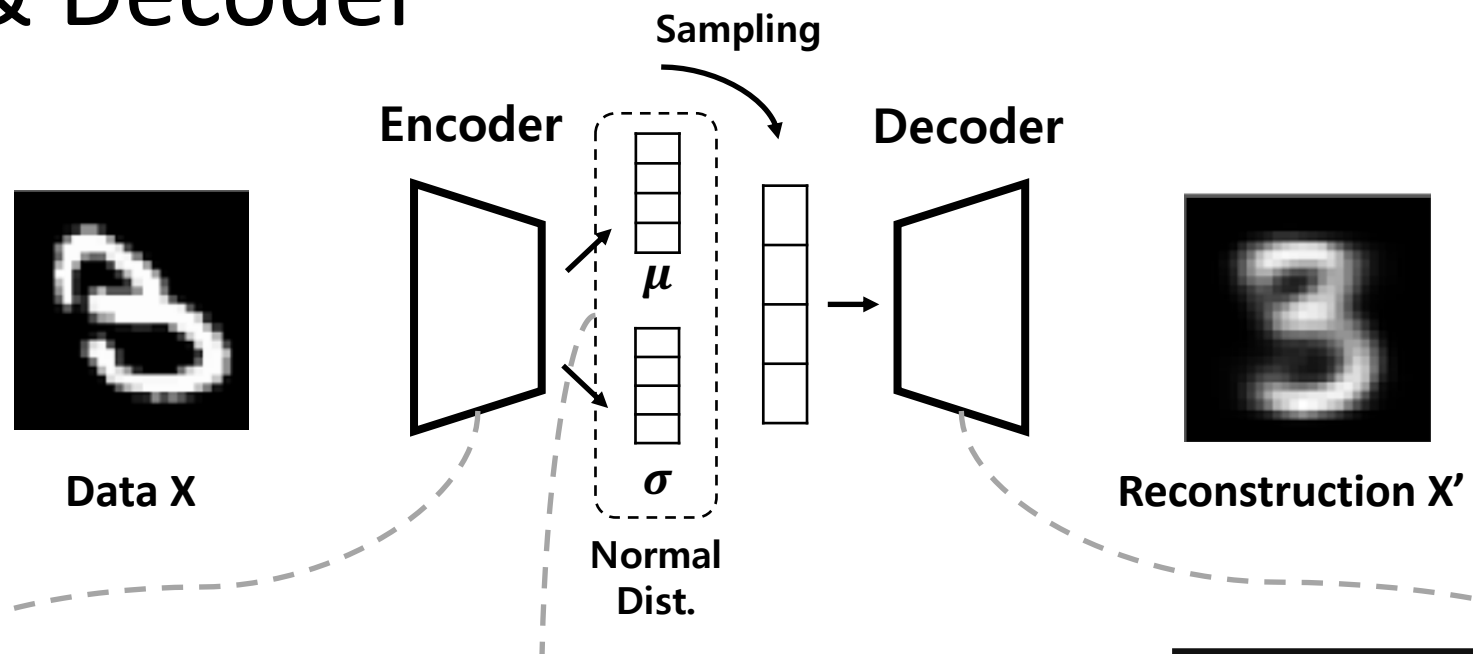
Encoder & Decoder



code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Encoder & Decoder




```python
self.encoder = nn.Sequential(
    nn.Linear(self.x_dim, hidden_dim),
    nn.ReLU(),
    nn.Linear(hidden_dim, hidden_dim),
    nn.ReLU(),
    nn.Linear(hidden_dim, hidden_dim),
    nn.ReLU(),
)
```

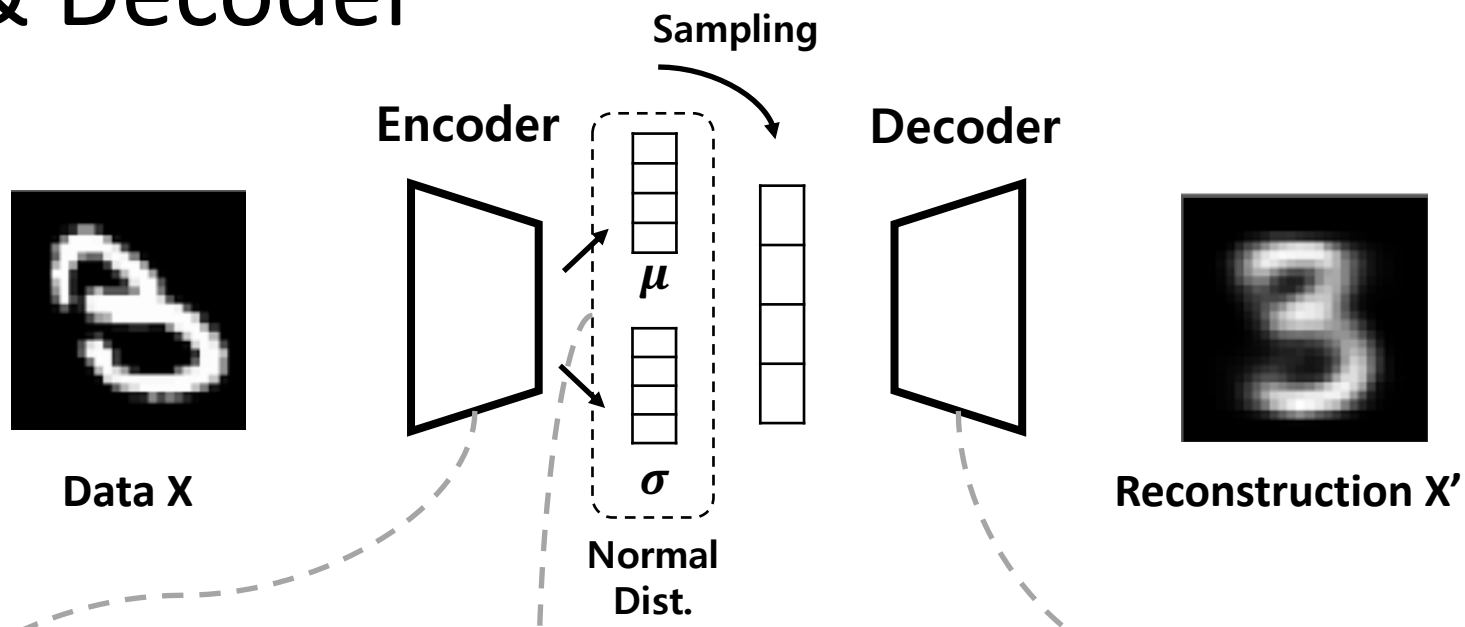
```python
self.mu_net = nn.Sequential(
    nn.Linear(hidden_dim, z_dim)
)
self.sigma_net = nn.Sequential(
    nn.Linear(hidden_dim, z_dim),
    nn.Softplus()
)
```

```python
self.decoder = nn.Sequential(
    nn.Linear(z_dim, hidden_dim),
    nn.ReLU(),
    nn.Linear(hidden_dim, hidden_dim),
    nn.ReLU(),
    nn.Linear(hidden_dim, self.x_dim),
    nn.Sigmoid(),
)
```

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Encoder & Decoder




```python
def encode(self, x):
    x = x.view(len(x), self.x_dim)
    h = self.encoder(x)
    z_mu = self.mu_net(h)
    z_sigma = self.sigma_net(h)
    return dist.Normal(z_mu, z_sigma)
```

```python
z_dist = vae.encode(x)
z = z_dist.rsample()
x_reconst = vae.decode(z)
```

```python
def decode(self, z):
    return self.decoder(z).view(len(z), *self.x_shape)
```

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Implementation TODO

- Encoder & Decoder Network
- **Loss function**
- Training Loop
- Visualization

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

ELBO

$$\max_{\theta,\phi} \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\phi)}[\log p(\boldsymbol{x}|\boldsymbol{z},\theta)] - KL[q(\boldsymbol{z}|\boldsymbol{x},\phi)||p(\boldsymbol{z})]$$

$$\log p(\boldsymbol{x}|\mathbf{z},\theta) = \log \prod_{d=1}^{D_x} Ber(x_d|x'_d)$$
$$= \sum_{d=1}^{D} \log(x'_d)^{x_d}(1-x'_d)^{x'_d}$$
$$= \sum_{d=1}^{D} x_d \log x'_d + (1-x_d)\log(1-x'_d)$$

$$KL[q(\boldsymbol{z}|\boldsymbol{x},\phi)||p(\boldsymbol{z})] = \sum_{d=1}^{D_z} KL[N(z_d|\mu_d,\sigma_d^2)||N(z_d|0,1)]$$
$$= \sum_{d=1}^{D_z} \mu_d^2 + \sigma_d^2 - \log\sigma_d^2 - 1$$

```python
def compute_elbo(x, x_reconst, z_posterior, z_prior):
    log_likelihood = (x*torch.log(x_reconst + 1e-6) + (1-x)*torch.log(1 - x_reconst + 1e-6)).sum((1, 2, 3))
    kl = dist.kl_divergence(z_dist, z_prior).sum(1)
    return (log_likelihood - kl).mean()
```

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Implementation TODO

- Encoder & Decoder Network
- Loss function
- **Training Loop**
- Visualization

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Training Loop

```python
for epoch_i in trange(n_epoch):
    for x, _ in loader:
        x = x.to(device)

        z_dist = vae.encode(x)
        z = z_dist.rsample()
        x_reconst = vae.decode(z)

        elbo = compute_elbo(x, x_reconst, z_dist, z_prior)
        loss = -elbo

        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
```

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Implementation TODO

- Encoder & Decoder Network
- Loss function
- Training Loop
- **Visualization**

code & slide :
github.com/namsan96/1126_vaegan

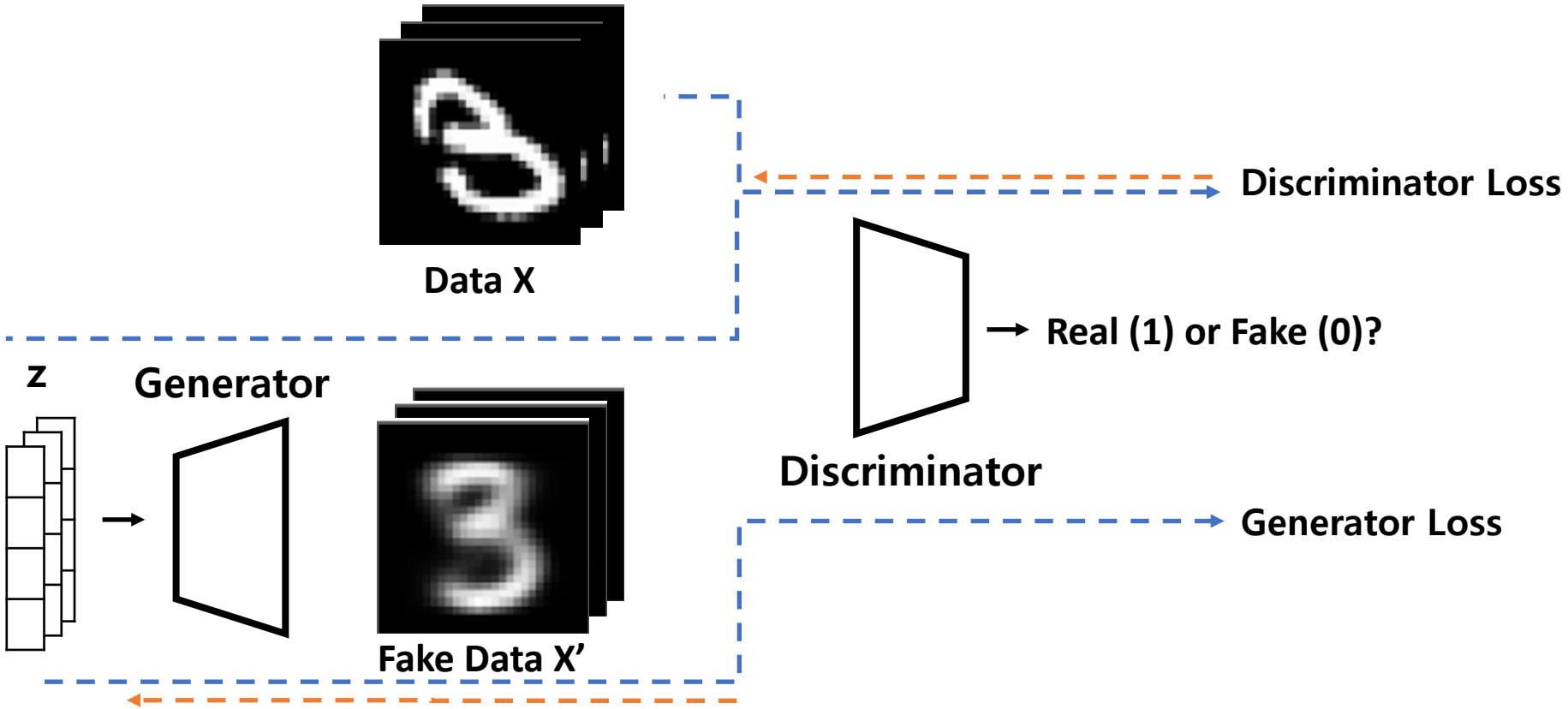
MLAI
Machine Learning & Artificial Intelligence

Generative Adversarial Networks (GAN)

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

GAN Overview



code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Implementation TODO

- Generator & Discriminator Network
- Training Loop
- Visualization

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

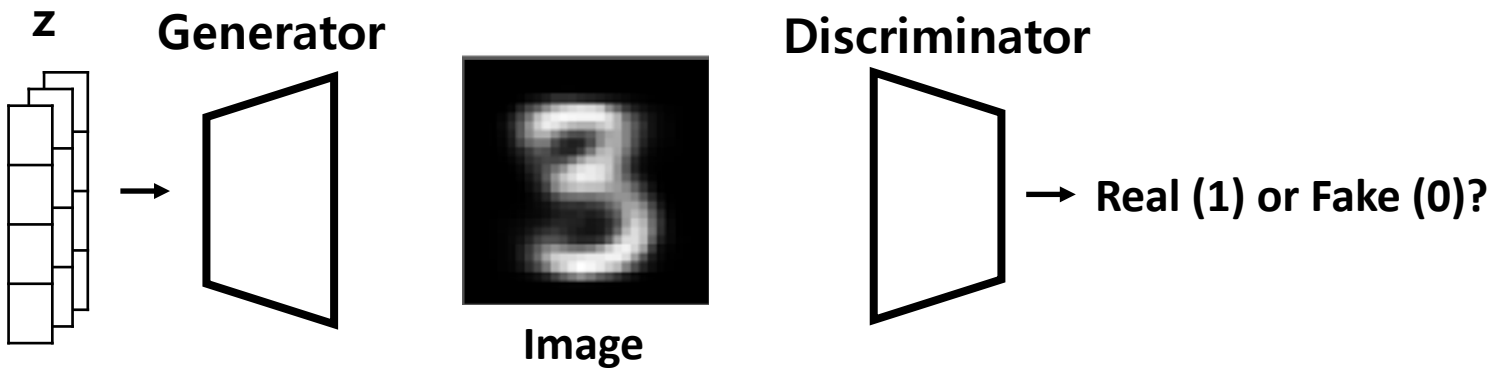
Implementation TODO

- **Generator & Discriminator Network**
- Training Loop
- Visualization

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

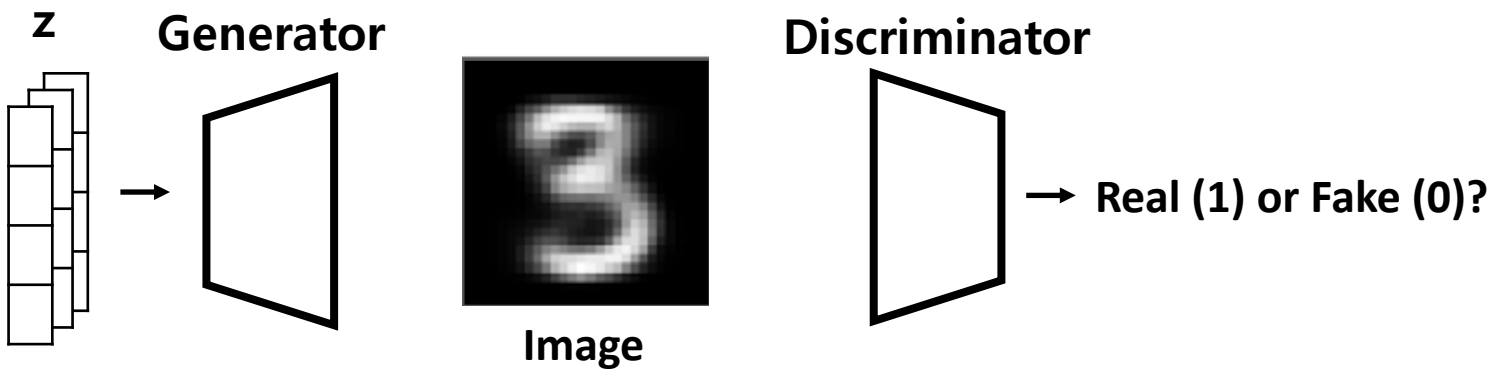
Generator & Discriminator



code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Generator & Discriminator

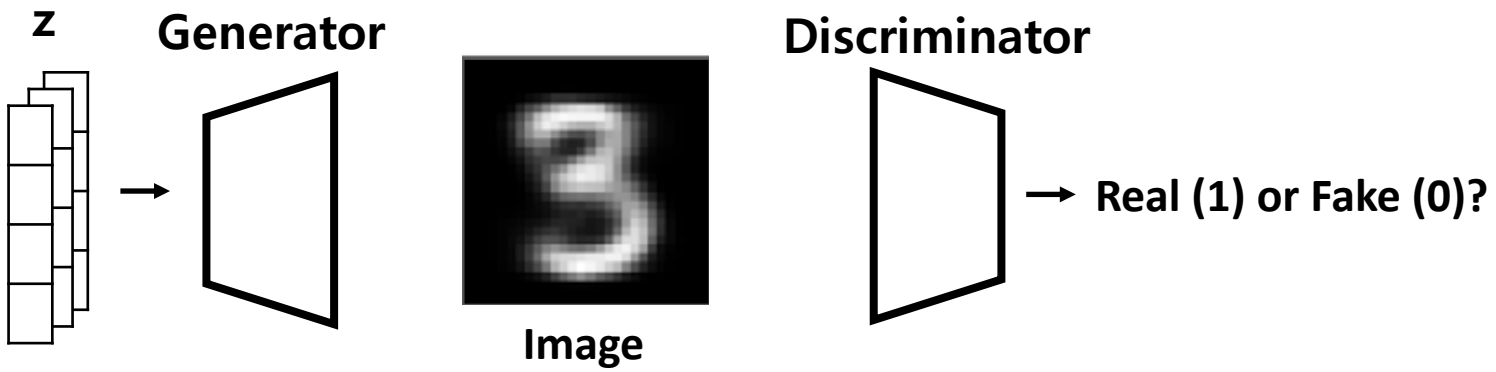



```python
class G(nn.Module):
    def __init__(self, z_dim, x_shape, hidden_dim=500):
        super().__init__()

        self.x_shape = x_shape
        self.x_dim = np.prod(x_shape)
        self.z_dim = z_dim

        self.net = nn.Sequential(
            nn.Linear(z_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, self.x_dim),
            nn.Sigmoid()
        )

    def forward(self, input):
        return self.net(input).view(len(input), *self.x_shape)
```

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Generator & Discriminator




```python
class D(nn.Module):
    def __init__(self, x_shape, hidden_dim=500):
        super().__init__()

        self.x_dim = np.prod(x_shape)

        self.net = nn.Sequential(
            nn.Linear(self.x_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, 1),
            nn.Sigmoid()
        )

    def forward(self, input):
        return self.net(input.view(len(input), self.x_dim)).squeeze(-1)
```

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Implementation TODO

- Generator & Discriminator Network
- **Training Loop**
- Visualization

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Training Loop

```
for epoch_i in trange(n_epoch):
    for real_x, _ in loader:
```

```
        d.zero_grad()

        # discriminator for real data
        real_x = real_x.to(device)
        real_pred_y = d(real_x)
        real_y = make_real_y(len(real_x)).to(device)

        real_d_loss = criterion(real_pred_y, real_y)
        real_d_loss.backward()

        # discriminator for fake data
        z = torch.randn(len(real_x), z_dim, device=device)
        fake_x = g(z)
        fake_pred_y = d(fake_x.detach())
        fake_y = make_fake_y(len(real_x)).to(device)

        fake_d_loss = criterion(fake_pred_y, fake_y)
        fake_d_loss.backward()

        d_optimizer.step()
```

```
        # generator
        g.zero_grad()
        fake_pred_y = d(fake_x).view(-1)
        g_loss = criterion(fake_pred_y, real_y)
        g_loss.backward()
        g_optimizer.step()
```

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Implementation TODO

- Generator & Discriminator Network
- Training Loop
- **Visualization**

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

In this tutorial,

- Demonstrate **VAE & GAN** with toy data (MNIST)
- Describe brief outline of implementation
- You can go in detail with code : github.com/namsan96/1126_vaegan

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

Q&A

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence

code & slide :
github.com/namsan96/1126_vaegan

MLAI
Machine Learning & Artificial Intelligence