Deep Learning

Vazgen Mikayelyan

YSU, Krisp

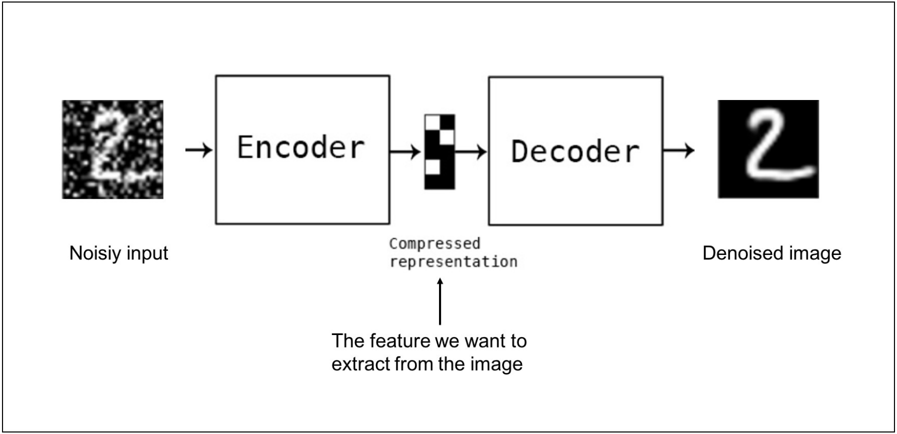
December 3, 2019

Outline

1. Autoencoders

2. Generative Adversarial Networks

Denoising Autoencoders



Denoising Autoencoders

- Denoising autoencoders create a corrupted copy of the input by introducing some noise.

Denoising Autoencoders

- Denoising autoencoders create a corrupted copy of the input by introducing some noise.
- This helps to avoid the autoencoders to copy the input to the output without learning features about the data.

Denoising Autoencoders

- Denoising autoencoders create a corrupted copy of the input by introducing some noise.
- This helps to avoid the autoencoders to copy the input to the output without learning features about the data.
- The model learns a vector field for mapping the input data towards a lower dimensional manifold which describes the natural data to cancel out the added noise.

Denoising Autoencoders

- Denoising autoencoders create a corrupted copy of the input by introducing some noise.
- This helps to avoid the autoencoders to copy the input to the output without learning features about the data.
- The model learns a vector field for mapping the input data towards a lower dimensional manifold which describes the natural data to cancel out the added noise.
- Minimizes the loss function between the output node and the corrupted input.

Variational Autoencoders

- Can we generate content with autoencoders?

Variational Autoencoders

- Can we generate content with autoencoders?
- Variational autoencoder can be defined as being an autoencoder whose training is regularised to avoid overfitting and ensure that the latent space has good properties that enable generative process.

Variational Autoencoders

- Can we generate content with autoencoders?
- Variational autoencoder can be defined as being an autoencoder whose training is regularised to avoid overfitting and ensure that the latent space has good properties that enable generative process.
- Instead of encoding an input as a single point, we encode it as a distribution over the latent space.

Variational Autoencoders

- Denote distribution of latent representations by $p(z)$.

Variational Autoencoders

- Denote distribution of latent representations by $p(z)$.
- Denote distribution of ideal encoder by $p(z|x)$.

Variational Autoencoders

- Denote distribution of latent representations by $p(z)$.
- Denote distribution of ideal encoder by $p(z|x)$.
- We want to find a neural network $q_w(z|x)$ such that

$$q_w(z|x) \approx p(z|x).$$

Variational Autoencoders

- Denote distribution of latent representations by $p(z)$.
- Denote distribution of ideal encoder by $p(z|x)$.
- We want to find a neural network $q_w(z|x)$ such that

$$q_w(z|x) \approx p(z|x).$$

- Let minimize the function

$$L(w) = KL(q_w(z|x) || p(z|x)).$$

Variational Autoencoders

Note that

$$KL\left(q_w\left(z|x\right)||p\left(z|x\right)\right) = \mathbb{E}_{q_w(z|x)}\left[\log \frac{q_w\left(z|x\right)}{p\left(z|x\right)}\right]$$

Variational Autoencoders

Note that

$$KL\left(q_w\left(z|x\right)||p\left(z|x\right)\right) = \mathbb{E}_{q_w(z|x)}\left[\log \frac{q_w\left(z|x\right)}{p\left(z|x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(z|x\right)\right]$$

Variational Autoencoders

Note that

$$KL\left(q_w\left(z|x\right)||p\left(z|x\right)\right) = \mathbb{E}_{q_w(z|x)}\left[\log\frac{q_w\left(z|x\right)}{p\left(z|x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(z|x\right)\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log\frac{p\left(x|z\right)p\left(z\right)}{p\left(x\right)}\right]$$

Variational Autoencoders

Note that

$$KL\left(q_w\left(z|x\right)||p\left(z|x\right)\right) = \mathbb{E}_{q_w(z|x)}\left[\log\frac{q_w\left(z|x\right)}{p\left(z|x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(z|x\right)\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log\frac{p\left(x|z\right)p\left(z\right)}{p\left(x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log\frac{q_w\left(z|x\right)}{p\left(z\right)}\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(x|z\right)\right] + \mathbb{E}_{q_w(z|x)}\left[\log p\left(x\right)\right]$$

Variational Autoencoders

Note that

$$KL\left(q_w\left(z|x\right)||p\left(z|x\right)\right) = \mathbb{E}_{q_w(z|x)}\left[\log \frac{q_w\left(z|x\right)}{p\left(z|x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(z|x\right)\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log \frac{p\left(x|z\right)p\left(z\right)}{p\left(x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log \frac{q_w\left(z|x\right)}{p\left(z\right)}\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(x|z\right)\right] + \mathbb{E}_{q_w(z|x)}\left[\log p\left(x\right)\right]$$

$$= KL\left(q_w\left(z|x\right)||p\left(z\right)\right) - \mathbb{E}_{q_w(z|x)}\left[\log p\left(x|z\right)\right] + \log p\left(x\right).$$

Note that

$$KL\left(q_w\left(z|x\right)||p\left(z|x\right)\right) = \mathbb{E}_{q_w(z|x)}\left[\log \frac{q_w\left(z|x\right)}{p\left(z|x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(z|x\right)\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log \frac{p\left(x|z\right)p\left(z\right)}{p\left(x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log \frac{q_w\left(z|x\right)}{p\left(z\right)}\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(x|z\right)\right] + \mathbb{E}_{q_w(z|x)}\left[\log p\left(x\right)\right]$$

$$= KL\left(q_w\left(z|x\right)||p\left(z\right)\right) - \mathbb{E}_{q_w(z|x)}\left[\log p\left(x|z\right)\right] + \log p\left(x\right).$$

So we have to model the distribution $p\left(x|z\right)$ too, which will be our decoder:

Note that

$$KL\left(q_w\left(z|x\right)||p\left(z|x\right)\right) = \mathbb{E}_{q_w(z|x)}\left[\log\frac{q_w\left(z|x\right)}{p\left(z|x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(z|x\right)\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log q_w\left(z|x\right)\right] - \mathbb{E}_{q_w(z|x)}\left[\log\frac{p\left(x|z\right)p\left(z\right)}{p\left(x\right)}\right]$$

$$= \mathbb{E}_{q_w(z|x)}\left[\log\frac{q_w\left(z|x\right)}{p\left(z\right)}\right] - \mathbb{E}_{q_w(z|x)}\left[\log p\left(x|z\right)\right] + \mathbb{E}_{q_w(z|x)}\left[\log p\left(x\right)\right]$$

$$= KL\left(q_w\left(z|x\right)||p\left(z\right)\right) - \mathbb{E}_{q_w(z|x)}\left[\log p\left(x|z\right)\right] + \log p\left(x\right).$$

So we have to model the distribution $p\left(x|z\right)$ too, which will be our decoder:

$$\underset{w,w'}{\operatorname{argmin}}\left(KL\left(q_w\left(z|x\right)||p\left(z\right)\right) + \mathbb{E}_{q_w(z|x)}\left[\|x - f_{w'}\left(z\right)\|^2\right]\right)$$

Problem: how to evaluate the term $KL\left(q_w\left(z|x\right)||p\left(z\right)\right)$?

Problem: how to evaluate the term $KL\left(q_w\left(z|x\right)||p\left(z\right)\right)$?
Solution: we will assume that $q_w = \mathcal{N}\left(\mu, \Sigma\right)$ and $p = \mathcal{N}\left(0, I\right)$.

The model is then trained as follows:

- first, the input is encoded as distribution over the latent space,

Variational Autoencoders

The model is then trained as follows:

- first, the input is encoded as distribution over the latent space,
- second, a point from the latent space is sampled from that distribution,

The model is then trained as follows:

- first, the input is encoded as distribution over the latent space,
- second, a point from the latent space is sampled from that distribution,
- third, the sampled point is decoded and the reconstruction error can be computed,

Variational Autoencoders

The model is then trained as follows:

- first, the input is encoded as distribution over the latent space,
- second, a point from the latent space is sampled from that distribution,
- third, the sampled point is decoded and the reconstruction error can be computed,
- finally, the reconstruction error is backpropagated through the network.

Variational Autoencoders

The model is then trained as follows:

- first, the input is encoded as distribution over the latent space,
- second, a point from the latent space is sampled from that distribution,
- third, the sampled point is decoded and the reconstruction error can be computed,
- finally, the reconstruction error is backpropagated through the network.

Assumptions

- In practice, the encoded distributions are chosen to be normal so that the encoder can be trained to return the mean and the covariance matrix that describe these Gaussians.

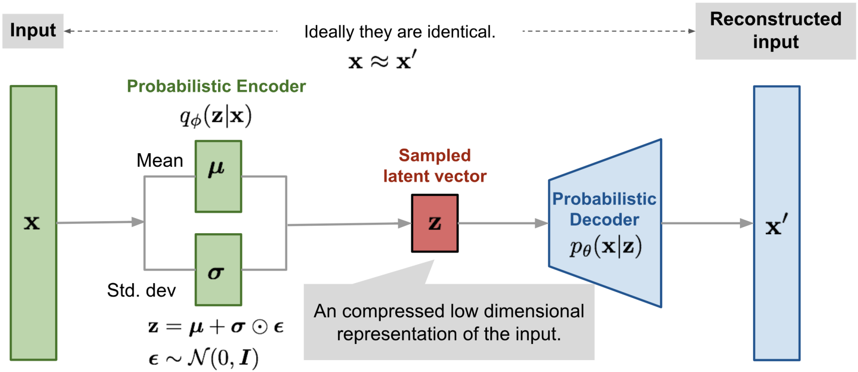
The model is then trained as follows:

- first, the input is encoded as distribution over the latent space,
- second, a point from the latent space is sampled from that distribution,
- third, the sampled point is decoded and the reconstruction error can be computed,
- finally, the reconstruction error is backpropagated through the network.

Assumptions

- In practice, the encoded distributions are chosen to be normal so that the encoder can be trained to return the mean and the covariance matrix that describe these Gaussians.
- The distributions returned by the encoder are enforced to be close to a standard normal distribution.

Variational Autoencoders



Variational Autoencoders

Let f is our encoder, g is the decoder and D is our training dataset. In this case we will minimize the following loss function

$$\sum_{x \in D} L\left(x, g\left(f\left(x\right)\right)\right) + \lambda KL\left(N\left(\mu, \Sigma\right) || N\left(0, I\right)\right).$$

Outline

GANs

- GANs are neural networks that generate synthetic data given certain input data.

GANs

- GANs are neural networks that generate synthetic data given certain input data.
- GANs consist of two models:

GANs

- GANs are neural networks that generate synthetic data given certain input data.
- GANs consist of two models:
  - **Discriminator**
    takes samples of true and generated data and that try to classify them as well as possible.

GANs

- GANs are neural networks that generate synthetic data given certain input data.
- GANs consist of two models:
    - **Discriminator**
      takes samples of true and generated data and that try to classify them as well as possible.
    - **Generator**
      trained to fool the discriminator as much as possible by generating fake data.

Different types of GANs

- Simple GANs

Different types of GANs

- Simple GANs
- Deep Convolutional GANs (DCGANs)

Different types of GANs

- Simple GANs
- Deep Convolutional GANs (DCGANs)
- Conditional GANs (cGANs)

Different types of GANs

- Simple GANs
- Deep Convolutional GANs (DCGANs)
- Conditional GANs (cGANs)
- CycleGANs

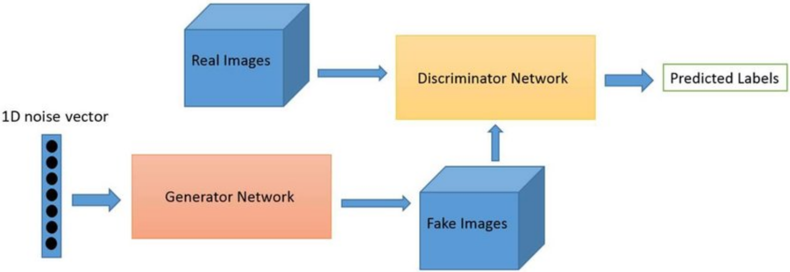
Different types of GANs

- Simple GANs
- Deep Convolutional GANs (DCGANs)
- Conditional GANs (cGANs)
- CycleGANs
- Wasserstein GANs (WGANs)

Different types of GANs

- Simple GANs
- Deep Convolutional GANs (DCGANs)
- Conditional GANs (cGANs)
- CycleGANs
- Wasserstein GANs (WGANs)
- Discover Cross-Domain Relations with Generative Adversarial Networks (Disco GANs)

Simple GANs


Real Images
Discriminator Network
Predicted Labels
1D noise vector
Generator Network
Fake Images

The Loss Function

Let p_z and p_{data} be respectively the distributions of input noise and our data and let D and G be respectively discriminator and generator.

The Loss Function

Let p_z and p_{data} be respectively the distributions of input noise and our data and let D and G be respectively discriminator and generator. We will do the following optimization

$$\min_G \max_D \left(\mathbb{E}_{x \sim p_{data}} \left[\log D\left(x\right) \right] + \mathbb{E}_{z \sim p_z} \left[\log\left(1 - D\left(G\left(z\right)\right)\right) \right] \right)$$

Optimization Algorithm for GANs

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right].$$

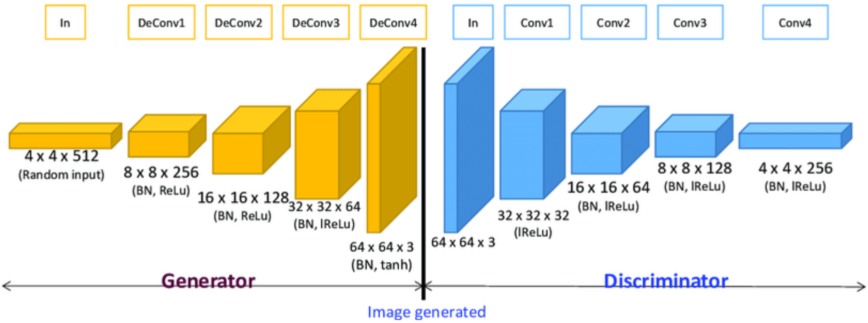
end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

DCGANs



In
DeConv1
DeConv2
DeConv3
DeConv4
In
Conv1
Conv2
Conv3
Conv4
4 x 4 x 512
(Random input)
8 x 8 x 256
(BN, ReLu)
16 x 16 x 128
(BN, ReLu)
32 x 32 x 64
(BN, lReLu)
64 x 64 x 3
(BN, tanh)
64 x 64 x 3
32 x 32 x 32
(lReLu)
16 x 16 x 64
(BN, lReLu)
8 x 8 x 128
(BN, lReLu)
4 x 4 x 256
(BN, lReLu)
Generator
Discriminator
Image generated

cGANs

These GANs use extra label information and result in better quality images and are able to control how generated images will look.

cGANs

These GANs use extra label information and result in better quality images and are able to control how generated images will look. In this case we will do the following optimization

$$\min_{G}\max_{D}\left(\mathbb{E}_{x\sim p_{data}}\left[\log D\left(x|y\right)\right]+\mathbb{E}_{z\sim p_{z}}\left[\log\left(1-D\left(G\left(z|y\right)\right)\right)\right]\right)$$

cGANs