CS F425: Deep Learning

29 Variational Autoencoder

Dr. Kamlesh Tiwari
Assistant Professor, Department of CSIS,
BITS Pilani, Pilani Campus, Rajasthan-333031 INDIA

April 13, 2023 ON-CAMPUS Campus @ BITS-Pilani [Jan-May 2023]

http://ktiwari.in/dl

Variational Autoencoders

A generative model

- h corresponding to the dataset, would not cover whole space
- Randomly pertered h would be very similar

How to sample a new latent vector from feature space?

It would give something that is not in my database

Variational Autoencoders

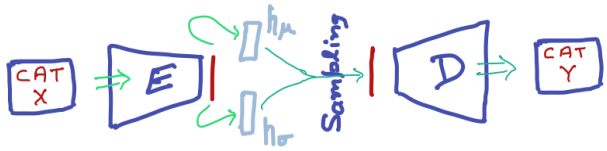
- Distribution of latent space can help sample a new feature vector
- If we don't know the distribution of latent variables,
  → can we force it to be one of our known one?

Regularize the autoencoder

1. Distribution of latent vector shoculd be unit normal. Along with minimum reconstruction loss $||x - \hat{x}||$

- Why Gaussian? simple, has only two parameters μ and σ
- KL-Divergence can be used to measure the distance between two distributions

Variational Autoencoders



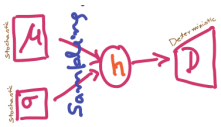
Two loss

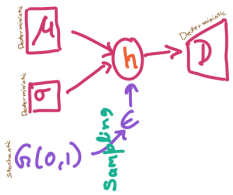
1. Reconstruction loss: $MSE(X, Y)$
2. KL-D: $KLD(G(h_\mu, h_\sigma), N(0, 1))$

- Issue is that now we are unable to do backpropogation
- Solution is re-parametrization

Sampling is Stochastic

- Stochastic sampling blocks backpropogation



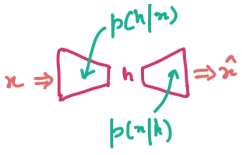


Re-parametrization

- How would you sample given the h_μ and h_σ
  1. Get ϵ from unit normal $N(0, 1)$
  2. Then $\hat{h} = h_\mu + \epsilon \times h_\sigma$

Important thing is that you donot need to learn anything for $N(0, 1)$

Probabilistic view



- Bayes Theorem: recall **conditional probability** of an event

$$p(A|B) = \frac{p(A \cap B)}{p(B)} = \frac{p(B|A)p(A)}{p(B)}$$

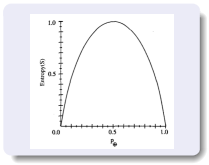
- Event and associated information content

Statement x	Probability $p(x)$	Information $-\log(p(x))$
Sun rise in east	1 (high)	0 (low)
Today is solar eclipse	1/100 (low)	6.64 (high)

Entropy

- Entropy (Shannon's) is expected value of information content of event.

$$-\sum_x p(x)\log(p(x))$$



It measures disorderness and is zero for the pure systems

- Difference between the information content of two distributions p and q can be found as

$$-\sum_x q(x)\log(q(x)) + \sum_x p(x)\log(p(x))$$

- KL-Divergence is almost same except that the expectation is always computed with respect to $p(x)$ when taking KLD of q with respect to p

$$-\sum_x p(x)\log(q(x)) + \sum_x p(x)\log(p(x))$$

KL-Divergence

- KL-Divergence is

$$\begin{aligned} KL-Div(p(x)|q(x)) &= -\sum_x p(x)\log(q(x)) + \sum_x p(x)\log(p(x)) \\ &= -\sum_x p(x)\log\frac{q(x)}{p(x)} \end{aligned}$$

- It is not symmetric
- It is always positive

$$KL-Div(q(h)|p(h|x)) = -\sum q(h)\log\frac{p(h|x)}{q(h)}$$

An example

- Consider database with images of size 100×100
- A pixel taking 256 values (range 0-255), we have $(256)^{10000}$ logically possible images (yes, most of them won't make sense)
- Even 1-Million images in database is sparse (may be localized)
- Joint probability distribution $p(x) = p(x_1, x_2, ..., x_{10000})$
- Can we get a distribution parameter θ such that if we sample from this distribution then the probability of getting an image from out training set is very high.

$$\theta^* = \arg\max_\theta p(x \in D)$$

- If one knows parameters of this distribution he could do Generative Modeling

Issue is

- Issue is that the pixel values are not independent

$$p(x) = p(x_1, x_2, ...x_{10000}) = p(x_1) \times p(x_2|x_1) \times p(x_3|x_1, x_2).....$$

- Due to this the functions became intractable
- However, observe that there is huge redundancy in the data
- There could be some $z_1, z_2, ..., z_{100}$ hidden variables that directly influence $p(x)$
- Distribution of the x is directly influenced by marginalization of z

$$p(x) = \sum_z p(x,z) = \sum_z p(x|z).p(z)$$

- Finding this is also difficult as number of parameters are still large.

Issue is

- As $p(z|x) = p(z,x)/p(x)$ so we cannot even estimate $p(z|x)$
- Let us approximate $p(z|x)$ via $q(z) \in L\{\text{tractable function family}\}$

$$\begin{aligned} q^*(z) &= \arg\min_{q(z)\in L} KL\text{–}Div(q(z)|p(z|x)) = -\sum_z q(z)\log\frac{p(z|x)}{q(z)} \\ &= -\sum_z q(z)\log\frac{p(z,x)}{p(x)q(z)} = \sum_z q(z)\Big[\log\frac{p(z,x)}{q(z)} - log(p(x))\Big] \\ &= -\sum_z q(z)\log\frac{p(z,x)}{q(z)} + log(p(x))\sum_z q(z) \\ &= -\sum_z q(z)\log\frac{p(z,x)}{q(z)} + log(p(x)) \\ log(p(x)) &= KL\text{–}Div(q(z)|p(z|x)) + \sum_z q(z)\log\frac{p(z,x)}{q(z)} \\ A &= B + C \end{aligned}$$

- Maximize ELBO (Evidence Lower bound)

ELBO Maximization

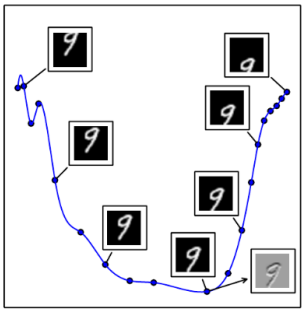
- ELBO is equivalent

$$\begin{aligned} \sum_z q(z)\log\frac{p(z,x)}{q(z)} &= \sum_z q(z)\log\frac{p(x|z).p(z)}{q(z)} \\ &= \sum_z q(z)\log(p(x|z)) + \sum_z q(z)\log\frac{p(z)}{q(z)} \\ &= E_{q_\phi(z|x)}\log(p_\theta(x|z)) - KLD(q(z)|p(z)) \\ &= -MSE - KLD \end{aligned}$$

VAE minimizes MSE and KLD

Learning Manifolds with Autoencoders

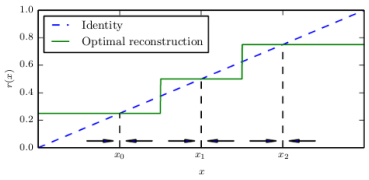
- Manifolds are hidden structures in the data
- Its tangent planes specify how one can change infinitesimally while staying on the manifold
- Regularization penalty and reconstruction loss with respect to small h both play role in manifold learning



Variations tangent to the manifold around correspond to changes. Hence the encoder learns a mapping (from the input to representation space) that is only sensitive to changes along the manifold directions, but that is insensitive to changes orthogonal to the manifold.

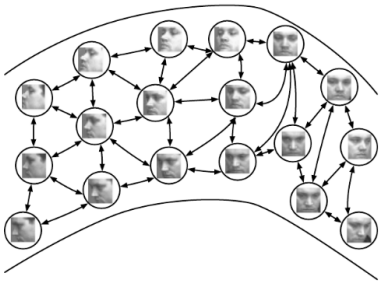
Learning Manifolds with Autoencoders

- By making the reconstruction function insensitive to perturbations of the input around the data points, we cause the autoencoder to recover the manifold structure.



Difficulty arises if the manifolds are not very smooth, one may need a very large number of training examples to cover each one of these variations

Learning Manifolds with Autoencoders



- Nonparametric manifold learning could build a nearest neighbor graph. Various procedures can thus obtain the tangent plane associated with a neighborhood of the graph as well as a coordinate system that associates each training example.

Thank You!

Thank you very much for your attention!