Lecture5: Latent Variable Models

Ki Pyo	완료
선택	VAE

1. Latent Variable Models: Motivation → What & Why

Latent Variable Model

what : 보이지 않지만 X에 영향을 주는 요소를 확률 변수 z(관측되지 않은 원인)로 나타내어 x와 z의 joint probability를 학습하는 모델

why : 보이지 않는 요소 또한 고려하기 위해!

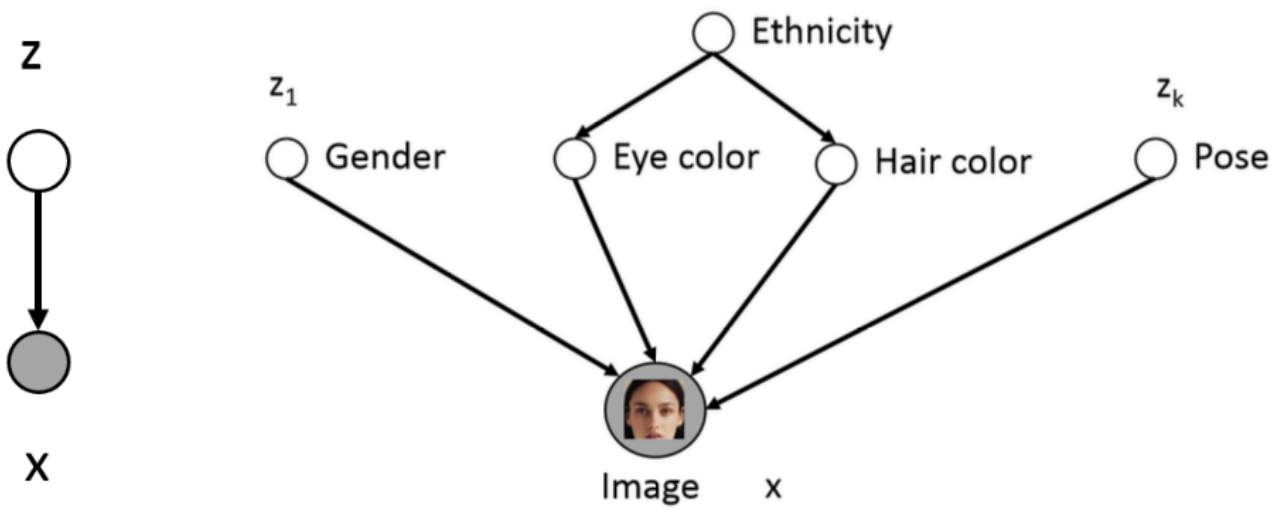
EX)

- 우리가 관측하고 있는 것들, 모집단 P(x)에 대한 샘플 데이터셋은 pixel의 값들임 -> X
- Latent Factors For Variation, z: 관측되지는 않지만 픽셀 값 X에 영향을 미치는 요소들이 많을 것임.

  → 예를 들어 사람들의 사진이라면, 사람의 성별, 눈 색깔 등과 같은 보이지 않는 요소들이 현재 관측된 픽셀 값들에 영향을 줄 수 있음
- Latent Variable Model은 이런 관측되지 않은, X의 variability에 영향을 주는 요소들을 Z라는 잠재변수로 취급하여 X와 Z의 분포를 모델링

2. Latent Variable Models: Motivation: Cont’d

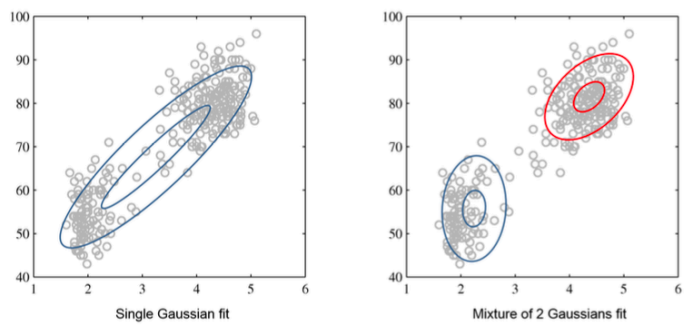
- Latent Variable Model에서 X, Z 둘의 관계를 Bayesian Network로 표현하면 다음과 같다
- z → x의 ordering을 따르기에 joint probability P(x,z)를 P(z)P(x|z)로 모델링하고 각각의 확률 분포를 Learning할 것임
- P(z)는 Learnable한 대상이 될 수도 있고 Fix 할 수도 있음(eg. VAE)



- 잠재적인 요소가 여러개라면 z1~zk는 하나의 Z로 표현될 것임 : Bunch of Latent Factors for variation that will be helpful for describing different types of images that we have access to.

3. Advantages of Latent Variable model

1. Less Varaibility in each cluster, clustered by Z

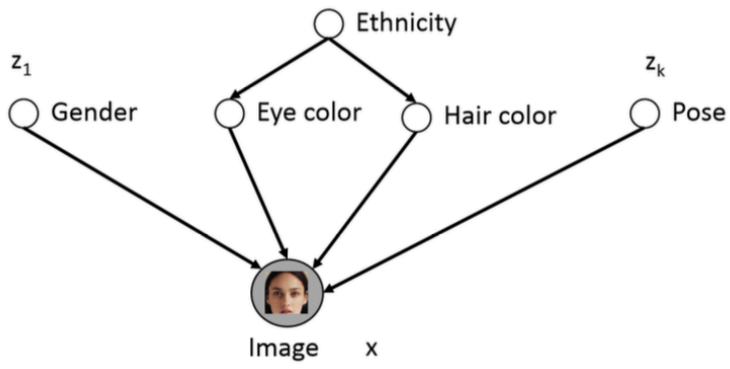


- 이미지들 간의 Cluster를 잠재적으로 잘 구별해내는 잠재적인 feature z를 뽑아낸다면 그때의 각각 Clustering된 X distribution은 capture해야하는 variation이 적기에 더 모델링하기 편함 P(x|z)
  ⇒ "쉽게 말해서 Z로 인해 X들이 Clustering이 될 수 있고 clustering된 X에 대해서 분포를 모델링하는 것이 Latent Variable z없이, Clustering없이 쌩으로 X에 대한 분포 P(x)를 모델링하는 것보다 편함"

2. Unsupervised Representation Learning

- 관측 변수 X에 대해 Label이 없는 상태로 P(x,z)를 학습
- Latent Variable Model을 통해 Z,X의 분포를 잘 모델링 및 학습했다면 z가 X를 구별할 좋은 feature들을 가지고 있을 수도 있음 representation extraction에도 유용할 것임 → Ptheta(x,z)를 모델링하니 Ptheta(z|x)를 구할 수 있음
- P(z|x)로 x에 대한 feature와 같은 역할을 하는 z를 compute할 수 있다

4. Deep Latent Variable Models



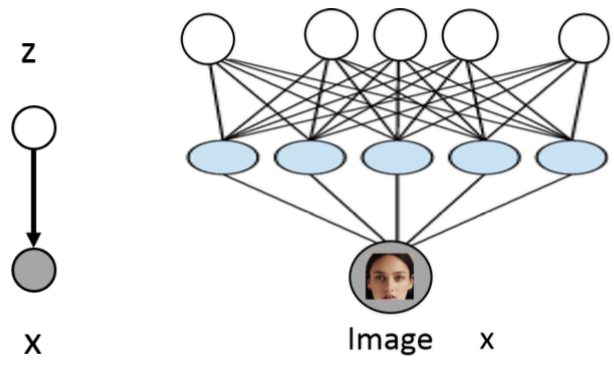
- 만약 위처럼

  (1) 어떤 Latent Factor들이 X의 Variabiliy에 영향을 줄 수 있는지 명확하게 알 수 있고,

  (2) Latent Factor간의 관계도 알 수 있다면,

Bayesian Network로 표현하여 X에 대한 Distribution을 명시적으로 나타낼 수 있음(위의 경우 각 Latent Factor끼리는 서로 어떤 영향을 주지 않음)

→ P(x|z1, z2, ...zk)와 같이 !



1. $\mathbf{z} \sim \mathcal{N}(0, I)$
2. $p(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}\left(\mu_\theta(\mathbf{z}), \Sigma_\theta(\mathbf{z})\right)$ where $\mu_\theta, \Sigma_\theta$ are neural networks

- 그러나 (1), (2)가 어떤지 모르는 경우가 대부분이니

  Z factor개수도 무작위로 설정하고(보통 어떤 분포에서 나왔다고 가정)

  이렇게 샘플링된 Z를 입력으로 받아 NN으로 Functional하게 Z factor간의 관계를 고려하여 X Distribution에 대한 파라미터를 출력하는 방식으로 X에 대한 Distribution을 모델링 및 학습을 진행
- 예를 들어 given z에 대해 X에 대한 분포가 Gaussian Distribution이라면 NN는 mean vector와 covariance matrix를 뽑아내면됨

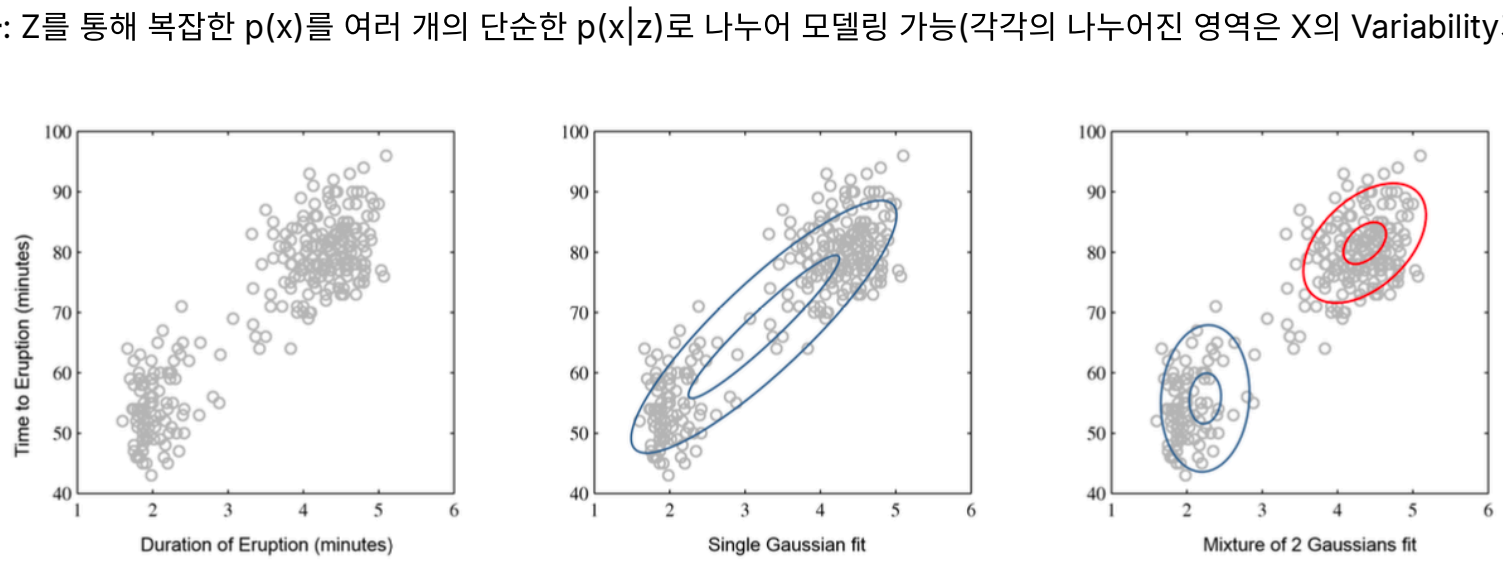
5. Summary

Goal of Latent Variable Model

- X에 대한 분포를 모델링 및 학습하고 싶은 것은 Generative Model의 목적과 같다.
- 다만 보이지 않은 X에 대한 Latent Factor of Variation Z 또한 고려하여 P(x,z)를 학습한다.
- Z → X의 ordering을 고려하여 P(x,z) = P(z)P(x|z)로 분해하여 각각을 학습. X Sampling(generation)은 Z sampling → X|Z sampling 순으로 진행

2 Advantages

- **구조 단순화**: Z를 통해 복잡한 p(x)를 여러 개의 단순한 p(x|z)로 나누어 모델링 가능(각각의 나누어진 영역은 X의 Variability가 적음)



- **표현 학습**: Z는 X를 잘 구분·요약하는 의미 있는 representation이 될 수 있음

6. Mixture Of Gaussians

- z는 이산적인 값을 가진다: Mixture의 type를 의미함

- P(x|z=k)는 각 z에 대한 X의 Distribution이고 Gaussian Distribution이다.
- z→x ordering을 따라 먼저 P(z)에서 z를 샘플링 한 후에 P(x|z)에서 x를 샘플링

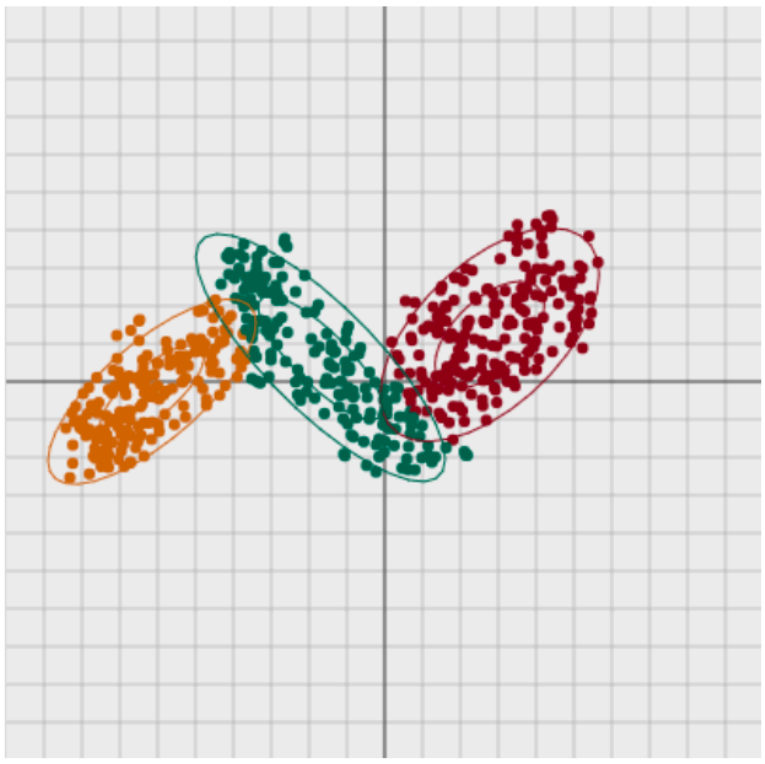
Mixture of Gaussians:

1. $\mathbf{z} \sim \text{Categorical}(1, \cdots, K)$
2. $p(\mathbf{x} \mid \mathbf{z} = k) = \mathcal{N}(\mu_k, \Sigma_k)$

P(z, x)를 위와 같이 모델링

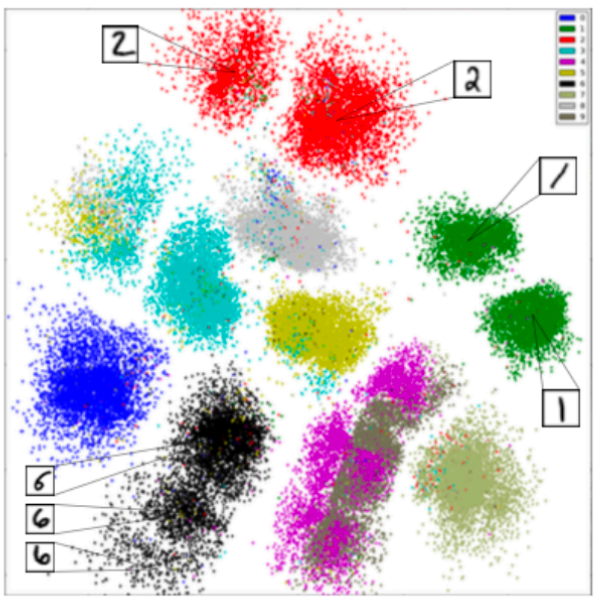
- z를 입력으로 받아 functional 하게 x distribution의 파라미터를 뽑아내지는 않음 -> Deep Neural Network를 사용하지 않아 Shallow Latent Variable Model이라고 불림
- 각 Distribution에 대한 mean vector와 Covariance를 Functional하게 뽑아내지 않으므로 따로 저장해야함 (look-up table)

- Clustering : P(z|x)를 통해 각 x가 어떤 z에서 가장 나올 법한지를 알 수 있음



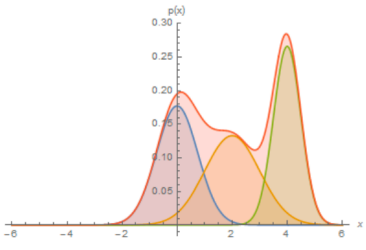
Shown is the posterior probability that a data point was generated by the i-th mixture component, $P(z = i|x)$

- Unsupervised Learning : 잘 훈련된 GMM의 각 z는 X를 잘 구별할 수 있는 feature의 역할을 할 수 있음 -> Z도 cluster를 이룰 것임



7. Marginalization

Alternative motivation: Combine simple models into a more complex and expressive one

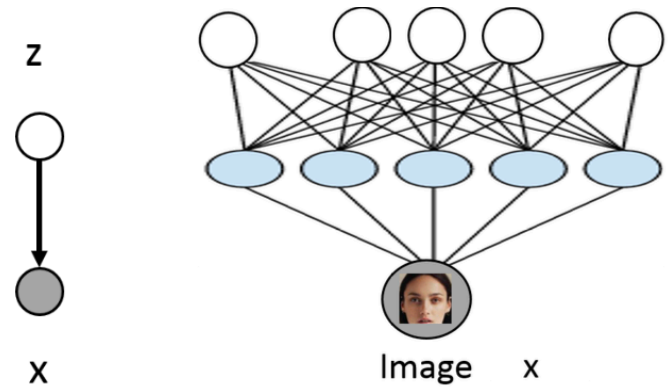


$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) = \sum_{\mathbf{z}} p(\mathbf{z}) p(\mathbf{x} \mid \mathbf{z}) = \sum_{k=1}^{K} p(\mathbf{z} = k) \underbrace{\mathcal{N}(\mathbf{x}; \mu_k, \Sigma_k)}_{\text{component}}$$

- P(x,z)를 학습하는 GMM에서 P(x) desnity는 z marginalization으로 구할 수 있음
- 각 Z에 대한 P(x|z)는 간단한 가우시안 분포이지만, P(x,z)에서 전체적으로 X에 대한 Distribution P(x)은 Complex하고 Flexible한 모양을 띔 → Latent Variable Model의 또다른 강점

8. VAE

1. $\mathbf{z} \sim \mathcal{N}(0, I)$
2. $p(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}(\mu_\theta(\mathbf{z}), \Sigma_\theta(\mathbf{z}))$ where μ_θ,Σ_θ are neural networks
   - $\mu_\theta(\mathbf{z}) = \sigma(A\mathbf{z} + c) = (\sigma(a_1\mathbf{z} + c_1), \sigma(a_2\mathbf{z} + c_2)) = (\mu_1(\mathbf{z}), \mu_2(\mathbf{z}))$
   - $\Sigma_\theta(\mathbf{z}) = diag(\exp(\sigma(B\mathbf{z} + d))) = \begin{pmatrix} \exp(\sigma(b_1\mathbf{z}+d_1)) & 0 \\ 0 & \exp(\sigma(b_2\mathbf{z}+d_2)) \end{pmatrix}$
   - $\theta = (A, B, c, d)$
3. Even though $p(\mathbf{x} \mid \mathbf{z})$ is simple, the marginal $p(\mathbf{x})$ is very complex/flexible



- GMM은 Z가 이산적인 값을 가진 것에 비해 VAE에서는 Z에 대한 Prior가 연속확률분포를 따름 N(0, I). Z가 discrete한 scalar가 아니라 Z가 여러 개의 factor로 이루어져 있는 continual vector임.
- P(z)를 따르는 따라서 무수히 많은 Z vector 에 대해 Decoder가 P(x|z) Gaussian Distribution의 파라미터를 뱉어내는 구조임. GMM과는 다르게 NN을 이용하여 sampling된 Z의 각 factor 간의 관계를 고려하여 X의 distribution parameter를 출력함 → Mixture of an infinite number of Gaussians, Deep Latent Variable Model
- Decoder Output : mean vector and diagonal covariance matrix
- 이 역시 각각의 P(x|z)는 간단한 Gaussian Distribution으로 모델링하지만 P(x,z)에서 marginalization으로 구한 P(x) distribution은 complex & flexible함.

9. What's the limit of Latent Variable Model

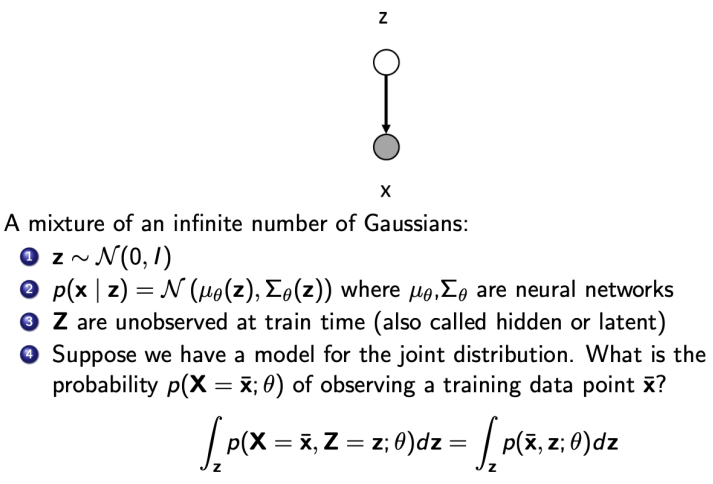
그러나 Training 과정 중에 X에 대한 Liklihood Estimation 계산 과정이 매우 복잡하고 intracible할 수도 있다. P(x,z)에서 z marginalization 때문에(z는 관측되지 않았기에, x 생성에 관여한 z의 모든 가능한 값을 check해야함) fully observed된 X만을 다루는 autoregressive model보다 likelihood estimation이 복잡함.

10. Marginal Likelihood

- 이미지의 절반은 관측된 값 X이고 절반은 관측되지 않은 알 수 없는 Z값일 때 이에 대한 Latent generative model을 모델링한다고 하면 P(x,z)를 학습하게 될 것이다.
- 학습된 P(x,z)에서 P(x)를 구하려면 z에 대한 marginalization이 필요하다.
- 위에서는 예시로 이미지의 가려진 알 수 없는 부분을 z라고 표현하였는데, 이때 marginalization을 한다는 것은 가능한 모든 z에 대해 P(x,z)를 계산 후 더하는 것과 같다.
  z는 all white인 경우도 있을 테고 z는 all black인 경우도 있을 테고 등등 (Need to consider all possible ways to complete the image)

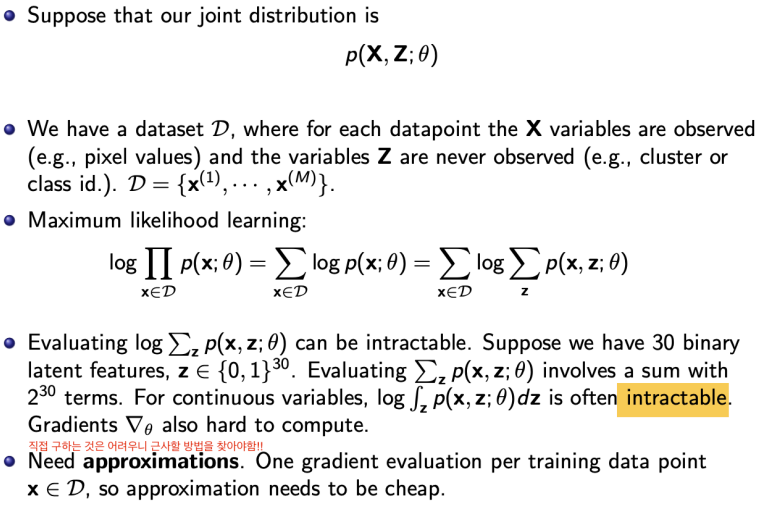
  → 심지어 z가 highdimensional한 feature라면 + continuous한 값이라면.. marginalization이 어려워 질 수 있음

11. Marginal Likelihood of X in VAE



VAE에서는 z가 연속형 백터이므로 하나의 데이터 포인트 X에 대한 VAE의 marginal likelihood 계산은 위와 같고, Z에 대해 Marginalize하기 위한 computational cost가 엄청날 것임, Intractible함.

12. MLE Learning for Latent Variable Models



Intractible한 Marginal Likelihood of X값을 근사할 방법이 필요함

13. Approximation 1: Naive Monte Carlo From Unform Distribution of Z

z를 uniform한 분포로부터 sampling하여 근사하는 방법 -> 안좋음

Likelihood function $p_\theta(\mathbf{x})$ for Partially Observed Data is hard to compute:

$$p_\theta(\mathbf{x}) = \sum_{\text{All values of } \mathbf{z}} p_\theta(\mathbf{x}, \mathbf{z}) = |\mathcal{Z}| \sum_{\mathbf{z} \in \mathcal{Z}} \frac{1}{|\mathcal{Z}|} p_\theta(\mathbf{x}, \mathbf{z}) = |\mathcal{Z}| \mathbb{E}_{\mathbf{z} \sim Uniform(\mathcal{Z})} [p_\theta(\mathbf{x}, \mathbf{z})]$$

z가 가질 수 있는 모든 값에 대한 Uniform Distribution

We can think of it as an (intractable) expectation. Monte Carlo to the rescue:

1. Sample $\mathbf{z}^{(1)}, \cdots, \mathbf{z}^{(k)}$ uniformly at random
2. Approximate expectation with sample average

$$\sum_{\mathbf{z}} p_\theta(\mathbf{x}, \mathbf{z}) \approx |\mathcal{Z}| \frac{1}{k} \sum_{j=1}^{k} p_\theta(\mathbf{x}, \mathbf{z}^{(j)})$$

Works in theory but not in practice. For most $\mathbf{z}$, $p_\theta(\mathbf{x}, \mathbf{z})$ is very low (most completions don't make sense). Some completions have large $p_\theta(\mathbf{x}, \mathbf{z})$ but we will never "hit" likely completions by uniform random sampling. Need a clever way to select $\mathbf{z}^{(j)}$ to reduce variance of the estimator.

Intractible한 expextation인 이유 : Z에 대한 Uniform Distribution이지만 Z가 연속형이기 때문

Monte Carlo 방식으로 Uniform Distribution에 대한 Expectation을 Approximation하여 X Marginal Likelihood를 Approximation할 수 있음

Estimator's Variance Issue

Uniform Sampling에서 Monte Carlo를 통한 Estimation의 문제점:대부분의 z 경우에 P(x,z)의 값이 작을 것임 (무수히 많은 z 경우에 대해 x와 크게 연관이 있는 경우는 상대적으로 적으니)

이에 따라 z에 대해 uniform sampling하는 방식으로 추정지를 구하면 추정치에 대한 분산이 커짐. (대부분의 z에 대해 P(x,z)가 대부분 작은 값을 가지다가 갑자기 큰 값을 가지게 되는 경우도 생기니까! 그런데 이런 경우가 1/N의 확률로 나타나니->작은 값을 가질 확률도 1/N이고 큰 값을 가질 확률도 1/N이니 추정치의 분산이 커질 수 밖에 없음) ⇒ 실제 log-marginal likelihood에 대해 불안정한 예측(high variance)이므로 unstable한 estimator임.

14. Approximation 2: Importance Sampling

Likelihood function $p_\theta(\mathbf{x})$ for Partially Observed Data is hard to compute:

$$p_\theta(\mathbf{x}) = \sum_{\text{All possible values of } \mathbf{z}} p_\theta(\mathbf{x}, \mathbf{z}) = \sum_{\mathbf{z} \in \mathcal{Z}} \frac{q(\mathbf{z})}{q(\mathbf{z})} p_\theta(\mathbf{x}, \mathbf{z}) = \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z})} \left[\frac{p_\theta(\mathbf{x}, \mathbf{z})}{q(\mathbf{z})} \right]$$

Monte Carlo to the rescue:

1. Sample $\mathbf{z}^{(1)}, \cdots, \mathbf{z}^{(k)}$ from $q(\mathbf{z})$
2. Approximate expectation with sample average

$$p_\theta(\mathbf{x}) \approx \frac{1}{k} \sum_{j=1}^{k} \frac{p_\theta(\mathbf{x}, \mathbf{z}^{(j)})}{q(\mathbf{z}^{(j)})}$$

What is a good choice for $q(\mathbf{z})$? Intuitively, frequently sample $\mathbf{z}$ (completions) that are likely given $\mathbf{x}$ under $p_\theta(\mathbf{x}, \mathbf{z})$.

- Uniform Distribution말고 다른 Distribution에서 Z를 Sampling하는 방식으로 Marginal Likelihood에 대한 근사값을 구하려고함.
- Naive하게 uniform한 분포로부터 sampling하지 말고,
  q(z)라는 arbitrary한 분포로부터 importance sampling(각 x에 대해 importance한 z를 더 often하게 sampling할 수 있도록)을 하자

What is a good choice for $q(\mathbf{z})$? Intuitively, frequently sample $\mathbf{z}$ (completions) that are likely given $\mathbf{x}$ under $p_\theta(\mathbf{x}, \mathbf{z})$.

** X's Marginal Likelihood Estimation with q(z) monte carlo is unbiased

3. This is an unbiased estimator of $p_\theta(\mathbf{x})$

$$\mathbb{E}_{\mathbf{z}^{(j)})\sim q(\mathbf{z})}\left[\frac{1}{k}\sum_{j=1}^{k}\frac{p_\theta(\mathbf{x},\mathbf{z}^{(j)})}{q(\mathbf{z}^{(j)})}\right] = p_\theta(\mathbf{x})$$

→ Estimator(Monte Carlo Approximation with q(z))의 Expectation 값은 실제 target값인 marginal likelihood of X이다 : Unbiased Estimator

16. Log(X's Marginal Likelihood) Estimation with q(z) monte carlo is biased

Estimating log-likelihoods

Likelihood function $p_\theta(\mathbf{x})$ for Partially Observed Data is hard to compute:

$$p_\theta(\mathbf{x}) = \sum_{\text{All possible values of } \mathbf{z}} p_\theta(\mathbf{x},\mathbf{z}) = \sum_{\mathbf{z}\in\mathcal{Z}} \frac{q(\mathbf{z})}{q(\mathbf{z})} p_\theta(\mathbf{x},\mathbf{z}) = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}\right]$$

Monte Carlo to the rescue:

1. Sample $\mathbf{z}^{(1)}, \cdots, \mathbf{z}^{(k)}$ from $q(\mathbf{z})$
2. Approximate expectation with sample average (*unbiased* estimator):

Log가 붙지 않으면 실제 타겟값에 대해 unbiased estimator

$$p_\theta(\mathbf{x}) \approx \frac{1}{k}\sum_{j=1}^{k}\frac{p_\theta(\mathbf{x},\mathbf{z}^{(j)})}{q(\mathbf{z}^{(j)})}$$

Recall that for training, we need the *log*-likelihood $\log(p_\theta(\mathbf{x}))$. We could estimate it as:

$$\log(p_\theta(\mathbf{x})) \approx \log\left(\frac{1}{k}\sum_{j=1}^{k}\frac{p_\theta(\mathbf{x},\mathbf{z}^{(j)})}{q(\mathbf{z}^{(j)})}\right) \overset{k=1}{\approx} \log\left(\frac{p_\theta(\mathbf{x},\mathbf{z}^{(1)})}{q(\mathbf{z}^{(1)})}\right)$$

추정치의 기댓값 / 실제 타겟값

However, it's clear that $\mathbb{E}_{\mathbf{z}^{(1)}\sim q(\mathbf{z})}\left[\log\left(\frac{p_\theta(\mathbf{x},\mathbf{z}^{(1)})}{q(\mathbf{z}^{(1)})}\right)\right] \neq \log\left(\mathbb{E}_{\mathbf{z}^{(1)}\sim q(\mathbf{z})}\left[\frac{p_\theta(\mathbf{x},\mathbf{z}^{(1)})}{q(\mathbf{z}^{(1)})}\right]\right)$

- MLE Learning을 위해서는 모델의 (1) X에 대한 그냥 likelihood estimation이 아니라 (2) Log likelihood estimation이 필요함
- (1)에 대한 Monte Carlo Estimation은 unbiased하지만
- (2)에 대한 Monte Carlo Estimation은 biased! → 로그 값에 대한 추정은 biased

의견공유 하고 싶어요!!

** Monte Carlo Estimation한 값 또한 확률 변수임 샘플링을 매번 할때마다 가지는 샘플들이 다르기 때문 이런 MCE에 대한 기댓값이 실제 값과 일치하다면 unbiased, 일치하지 않다면 biased(Jensen's inequality로 인해 항상 실제값보다 작은 값으로 biased된다고 합니다!)

- 따라서 log-likelihood에 대한 직접적인 monte carlo estimation은 biased함. 올바르지 않은 estimation
- 다른 방식으로의 안정된 estimation이 필요함

17. ELBO(Evidence Lower Bound) Estimation

Evidence Lower Bound

Log-Likelihood function for Partially Observed Data is hard to compute:

$$\log\left(\sum_{\mathbf{z}\in\mathcal{Z}} p_\theta(\mathbf{x},\mathbf{z})\right) = \log\left(\sum_{\mathbf{z}\in\mathcal{Z}} \frac{q(\mathbf{z})}{q(\mathbf{z})} p_\theta(\mathbf{x},\mathbf{z})\right) = \log\left(\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}\right]\right)$$

- log() is a concave function. $\log(px + (1-p)x') \geq p\log(x) + (1-p)\log(x')$.
- Idea: use Jensen Inequality (for concave functions)

$$\log(\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[f(\mathbf{z})]) = \log(\sum_{\mathbf{z}} q(\mathbf{z}) f(\mathbf{z})) \geq \sum_{\mathbf{z}} q(\mathbf{z}) \log f(\mathbf{z}) = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[\log f(\mathbf{z})]$$

Choosing $f(\mathbf{z}) = \frac{p_\theta(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}$

$$\log\left(\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}\right]\right) \geq \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}\left[\log\left(\frac{p_\theta(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}\right)\right]$$

Called Evidence Lower Bound (**ELBO**).

→ Jensen's Inequality로 인한 Log-Likelihood의 하한인 ELBO에 대한 Monte Carlo Estimation(Approximation)은 unbiased함. ELBO항은 앞에 log가 붙지 않음

- ELBO는 실제 marginal likelihood의 하한임은 보장
- ELBO값은 불편추정을 할 수 있음
- 이 ELBO에 대한 Estimation을 구하는 방식으로 ELBO 값이 실제 Log-Marginal Likelihood와 가까워 지면서도(E step),이 ELBO값이 커지도록 업데이트를 하여 간접적으로 Log-Marginal Likelihood를 크게하자(M step) → EM Algorithmn
- z를 샘플링할 q를 어떻게 고르냐에 따라 lower bound의 tight 정도가 결정됨 (하한인 ELBO값이 실제 log-marginal probability랑 얼마나 가까운가)

18. Variational Inference

- Suppose $q(\mathbf{z})$ is **any** probability distribution over the hidden variables
- **Evidence lower bound** (ELBO) holds for any q

$$\begin{aligned}\log p(\mathbf{x};\theta) &\geq \sum_{\mathbf{z}} q(\mathbf{z}) \log\left(\frac{p_\theta(\mathbf{x},\mathbf{z})}{q(\mathbf{z})}\right) \\ &= \sum_{\mathbf{z}} q(\mathbf{z}) \log p_\theta(\mathbf{x},\mathbf{z}) \underbrace{- \sum_{\mathbf{z}} q(\mathbf{z}) \log q(\mathbf{z})}_{\text{Entropy } H(q) \text{ of } q} \\ &= \sum_{\mathbf{z}} q(\mathbf{z}) \log p_\theta(\mathbf{x},\mathbf{z}) + H(q)\end{aligned}$$

E step : Lower bound를 어떻게 우리가 실제로 관심이 있는 log-marginal likelihood에 tight하게 맞출까?(best approximation?
=> “training”시에 z 샘플링에 이용되는 Q를 model이 관측값 x에 대해 뱉어내는 z의 distribution인 posterior distribution으로 만들면됨)

- Equality holds if $q = p(\mathbf{z}|\mathbf{x};\theta)$

$$\log p(\mathbf{x};\theta) = \sum_{\mathbf{z}} q(\mathbf{z}) \log p(\mathbf{z},\mathbf{x};\theta) + H(q)$$

- (Aside: This is what we compute in the E-step of the EM algorithm)

E step : Lower bound를 어떻게 우리가 실제로 관심이 있는 log-marginal likelihood에 tight하게 맞출까?(best approximation?

→ When doing ELBO Approximation, Best way for inferring Z variable(deciding the q distribution for z sampling) is to use true posterior distribution! Q(z)를 실제 모델에서 정의되는 Ptheta(z|x) distribution(True Posterior Distribution)으로 맞추어주면됨.

→ 그렇게 하면 부등식에서의 equality를 취함(ELBO값이 Log marginal likelihood값과 같아짐)

→ 그러나 이 Ptheta(z|x)를 구하는 것은 어렵다고 함.(Intractible)