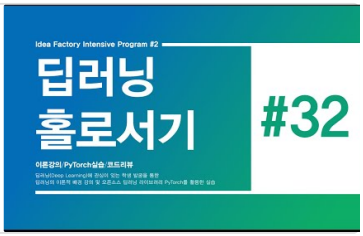
#32 - AutoEncoder and Variational AutoEncoder

주제 : AutoEncoder and Variational AutoEncoder

링크 :

[#32.Lec] AutoEncoder and Variational AutoEncoder - 딥러닝 홀로서기
강의 자료 링크 : https://github.com/heartcored98/Standalone-DeepLearning/blob/master/Lec10/Lec10-A.pdf자료 저장소 링크 : https://github.com/heartcored98/Standalone-DeepLe...
https://youtu.be/54hyK1J4wTc



1. Review of Last Lecture

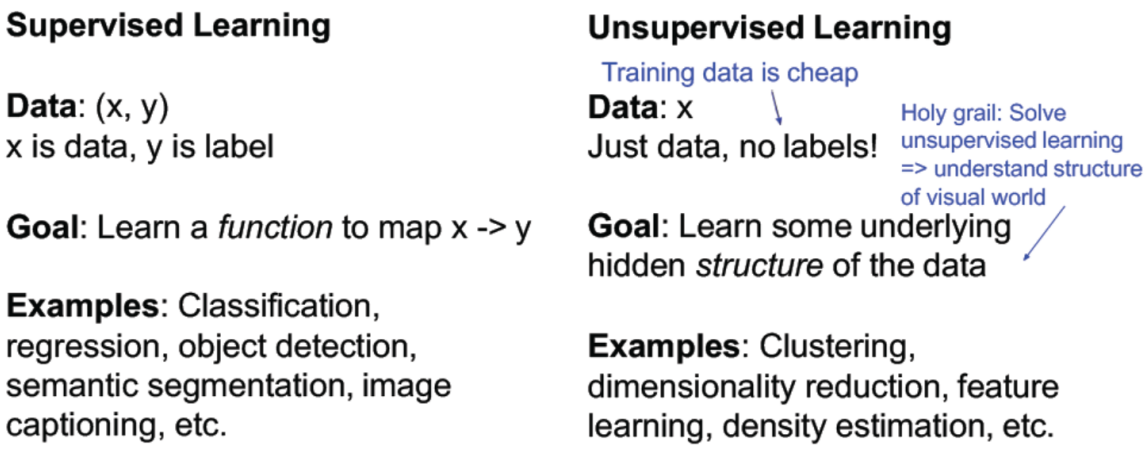
- Graph Convolution 필요성
  - Sequential Data를 다루기 위한 모델 아키텍쳐 CNN, RNN
    - Image : Grid 형태의 Pixel Value가 흑백은 1채널, 컬러는 3채널
    - Sentence : Sequence를 이루는 단어, Character가 배열된 형태
  - Graph Data를 다루기 위한 모델 아키텍쳐 GCN
- Graph Structure의 정의
  - Node와 Edge로 이루어진 형태
  - Adjacency Matrix와 Feature Node Matrix로 표현가능
- CNN Architecture Review
  - 작은 필터가 이미지상을 돌아다니면서, Activation Map을 만들고, 또 다른 필터를 씌워서 Activation Map을 만드는 모델
  - 특징
    - Weight Sharing
    - Learning Local Features
    - Translation Invariance
- CNN의 특징을 Graph로 적용시키는 방식
  - CNN Layer를 하나 거치면 각 Layer의 Activation Map의 Value를 업데이트

    - Activation Map의 Value는 특정 Layer에서의 이미지의 상태임
    - Graph의 구성요소
        - Adjacency Matrix : Node의 Connectivity를 나타냄. 업데이트는 필요 없음
        - Node Feature Matrix : 그래프가 현재 어떤 상태인지에 대한 정보를 담음. 따라서 업데이트가 필요함 → Graph Convolution Layer통과로 업데이트 → CNN의 Activation Map Value 업데이트 방식과 비슷함
    - Weight Sharing + Local Feature
        - 중신 노드와 주변에 있는 노드들만 가져와서 업데이트
        - 각각의 노드 Feature에 동일한 Weight를 곱해줌
        - 행렬 연산으로 구현
- Advanced Techniques
    - Inception : 다양한 Receptive를 가지는 여러개의 Filter를 통과 → Graph Convolution Layer를 두번이상 거침
    - Skip Connection : Size만 잘 맞춰서
    - Gated Skip Connection : 적절한 비율 z를 학습하여 섞자
    - Attention : 근접한 노드와 자기자신 노드의 상관관계 비율을 학습하여 업데이트

2. Generative Models

- CS231N - Stanford University의 강좌 슬라이드를 활용
- Overview
    - Unsupervised Learning
        - Supervised Learning과의 차이점
        - Generative Model들 중 VAE와 GAN에 대하여

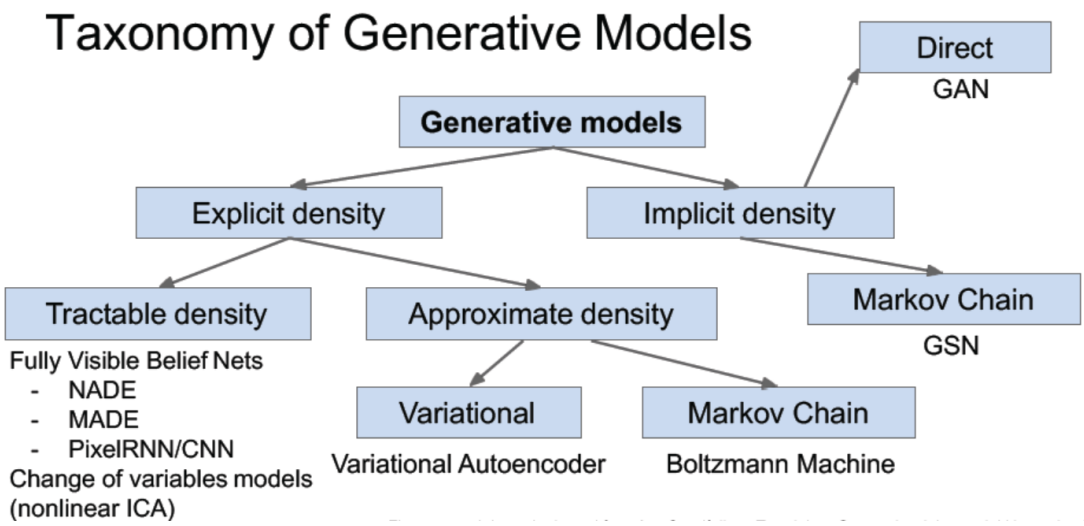
- Supervised vs Unsupervised Learning



- Supervised Learning
  - Data : (x, y)
    - Data x, Label y
    - 위와 같이 Data에 Label이 있는 형태
  - Goal
    - x로부터 y로 가는 Function을 neural network를 통해서 mapping
    - 예) Classification, Object Detection, Semantic Segmentation, Image Captioning
- Unsupervised Learning
  - Data : x
    - Data x만 존재하고 Label이 없는 형태
  - Goal
    - Data속에 내재되어 있는 정보(Feature)를 학습하는 것
    - 예) Clustering, Dimensionality Reduction, Autoencoders(Feature Learning), Density Estimation

- Generative Models
  - 기본 개념
    - 주어진 Training Data Set에서 같은 분포를 가진 새로운 샘플을 생성
      - Data Set에는 존재하지 않지만 해당 Data Set에서 나왔을 법한 새로운 Data를 생성하는 것
      - 예) 자동차, 비행기, 새 이미지를 입력해 주었을 때 새로운 자동차, 비행기, 새 이미지를 도출해 내는 것

- Density Estimation
  - Data들의 확률밀도함수 (Probability Density Function)를 찾아내서 Model의 PDF를 Data의 PDF에 비슷하게 만들자 (=근사시키자)
  - 예시) 서울의 집값 데이터에 대한 PDF
    - $P_{data}(X)$: 서울의 모든 집값에 대한 PDF
    - $P_{model}(X)$: Sample들의 집합으로 P_{data}에 근사한 값
    - 많은 샘플링을 통해서 $P_{data}(X)$와$P_{model}(X)$가 근접을 했다면, $P_{model}(X)$에서 임의로 샘플링한 Data는 $P_{data}(X)$를 대표할 수 있음 (즉, 서울의 집값이라고 이야기 할 수 있음)
  - 종류
    - Explicit Density Estimation
      - $P_{model}(X)$가 무엇인지 정확하게 식으로 알 수 있음
    - Implicit Density Estimation
      - $P_{model}(X)$의 정의된 상태는 모르나, 샘플링은 할 수 있음
- 역할
  - 존재하지 않지만 실사적인 Sample을 Data로부터 얻어낼 수 있음
    - 저화질의 화면을 고화질의 화면으로 변경
    - 스케치에 색을 입힐 수 있음
- 종류



  - Explicit Density Model
    - Tractable Density : Density를 실제로 계산해 볼 수 있는 모델

    - Approximate Density : Density의 형태를 가정하여(정규성 등) 근사하는 모델
        - Variational Autoencorder
        - Markov Chain (Boltzmann Machine)
    - Implicit Density Model : Density를 가정하지 않은 상태에서 샘플링만 가능
        - Markov Chain (GSN)
        - Direct (GAN)

3. Variational Autoencoders (VAE)

- 용어의 정의
    - Likelihood : $P_\theta(x)$에 속한 특정한 값
        - 1에서 6까지의 연속된 숫자 중 5를 뽑는 과정을 보자
        - 각 숫자들은 연속된 숫자들이기 때문에 정확히 5가 뽑힐 확률은 $\frac{1}{\infty}$로 수렴하여 0이 되어버림
        - 하지만 실제로는 확률이 0이더라도 해당 숫자는 뽑힐 수 있기때문에 '확률'이라는 지표로는 (연속된)숫자가 뽑힐 가능성을 비교할 수 없음
        - 따라서 확률로는 구할 수 없는 연속된 숫자들(사건)의 가능성은(=Likelihood) 특정 구간에 속할 확률을 구함으로서 간접적으로 그 숫자가 뽑힐 확률에 대한 감을 잡을 수 있음
        - 이것을 설명하는 그래프가 **확률 밀도 함수(Probability Density Function)이며, 확률 밀도 함수의 특정 구간에 속한 넓이 = 특정 구간에 속할 확률로 해석**될 수 있음
        - 가능도의 직관적인 정의 : 확률분포함수의 y값
            - 셀 수 있는 사건: **가능도 = 확률**
            - 연속 사건: **가능도 ≠ 확률, 가능도 = PDF값**

RPubs
https://www.rpubs.com/Statdoc/204928

    - 최대 가능도 추정량 (Maximum Likelihood Estimator)
        - 어떤 사건의 PDF를 찾는다고 했을 때, 각 사건의 확률을 우리는 모를 경우가 대부분
        - 이러한 사건들의 PDF를 근사하는 방법 중 하나가 최대 가능도 추정량임

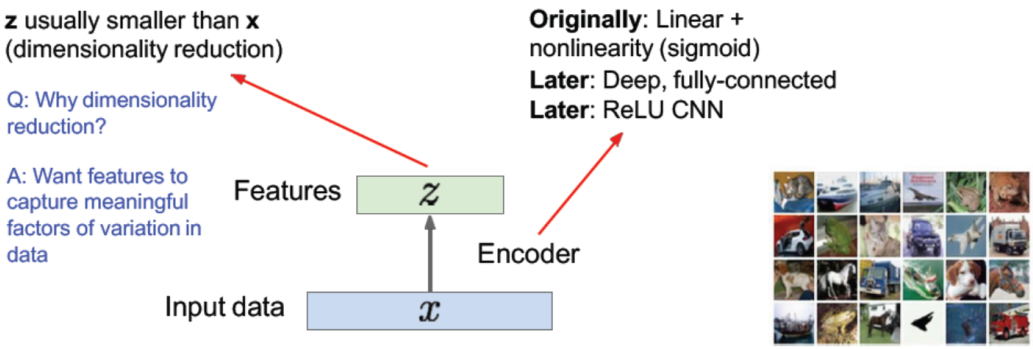
    예) 앞뒤가 나올 확률이 똑같지 않은 구부러진 동전을 던졌을 때의 확률에 대해 알아보자
        - 임의로 1000번 던져봤더니, 앞이 400, 뒤가 600번 나왔음

        - 우리는 이 샘플링으로부터 동전을 1000번 던졌을 때 앞이 400번 나올 가능성의 최대의 확률은 0.4라는 결론을 도출할 수 있음
        - 즉, 가능성(Likelihood)을 Maximize하는 어떤 P를 찾으면 그게 이 시스템을 가장 잘 묘사하는 P임
- 수식
    - VAE는 Latent Vector Z와 함께 다음과 같이 정의할 수 있음

      $P_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz$

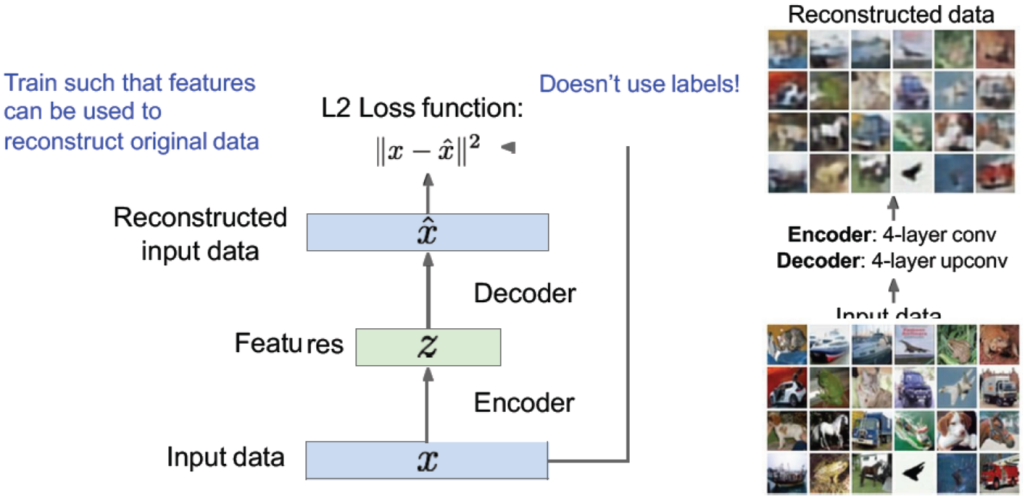
    - $P_\theta(x)$: $P_{data}(x)$에 근사한 모델의 Probability Density Function
        - θ는 Neural Network 속의 Parameter Set이라고도 할 수 있음
    - z : x를 Input으로 입력했을 때 Feature를 표현하는 Vector를 의미함
    - $P_\theta(z)$: z에 대한 PDF
    - $P_\theta(x|z)$: 어떤 특정한 z값일 때 어떤 x가 나올 확률
    - 따라서 $P_\theta(z)P_\theta(x|z)$를 적분한 값을 통해 $P_\theta(x)$를 구할 수 있음
    - PDF값이 Likelihood이기 때문에, Likelihood가 최대치가 되는 방향으로 트레이닝을 시키면 결국 저 θ들이 그 시스템을 잘 표현하는 Optimal θ로 수렴하게 됨

- AutoEncoder
    - 역할
        - Data를 받아서 그보다 훨씬 작은 착원의 Latent Vector를 만들어 내고, 이 Latent Vector가 x에 포함된 중요한 정보들을 담을 수 있도록 하는 것
        - Input x가 Encoder를 거치면 Feature z가 도출되는 형태



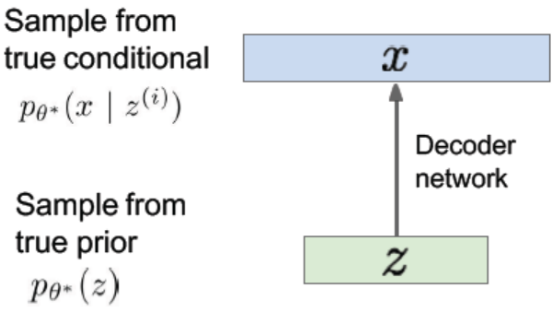
    - 구현방식
        - 전통적인 방식은 MLP + Sigmoid였지만, 최근에는 ReLU CNN으로 하는 추세

    - z는 중요한 정보를 내포하고 있어야 하므로 x보다 적은 차원으로 구현해야함
- Feature 학습 방식
    - Input Data Reconstruction
        - $\hat{x}$를 x와 최대한 비슷하게 만들기 위해서 z에 중요 정보만 담아야함
        - $\hat{x}$와 x가 얼마나 다른지를 정의한 L2 Loss Function을 활용하여 학습
        - 결국 최종적으로 중요한 Feature들만 담은 Latent Factor가 됨
        - Label이 들어가 있지 않으므로 이는 Unsupervised Learning임



- VAE활용 방식
    - VAE로 학습이 완료가 되면 Decoder파트를 제거한 후, Label이 적게 되어있는 Data를 가져와서 학습을 시킴 (Label이 없는 Data의 비율이 많음)
    - 이는 z라는 중요 Feature들로 학습되어 있는 모델이기에, 기존의 CNN모델에보다 훨씬 더 빠르게 학습이 가능함

- Variational Autoencoders
    - 중요한 Feature를 담고 있는 z를 활용하여 기존의 Data에 Label을 채워주는 방식이 아닌, 아예 새로운 Data를 생성하는 방식
    - Unobserved (Latent) z를 활용하여 Training Data x를 생성하는 예



- 정의
  - x는 이미지이고, z는 x를 생성하는 데 사용되는 잠재적 Feature들임: 속성, 방향 등
- 목표
  - Generative model의 true Parameter θ^*를 예측하고자 함
- 어떻게 이 모델을 표현할 수 있을까?
  - P(z)가 가우시안 분포를 따른다고 가정 → 예) 행복함을 나타내는 속성일 경우 →포커페이스가 평균이고 울거나 웃는 얼굴은 양 끝단
  - 새로운 이미지를 생성하려면 P(z)로부터 P(x|z)로의 맵핑을 해야하고 이는 복잡한 구조를 가지고 있기에 (가우시안 분포가 아닐 수도 있음) Neural Network를 통하여 맵핑하여 근사시키도록 함
- 어떻게 모델을 학습 시킬 것인가?
  - 즉, $P_{\theta^*}(z)$와$P_{\theta^*}(x|z)$를 어떻게 실제 $P_\theta(z)$와$P_\theta(x|z)$에 근사시킬 수 있을 것인가?
  - 이를 위해서 Maximum Likelihood Estimation의 개념을 도입하기로 함
    - Generative model의 식 $P_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz$ 과 같이 적분계산을 통해서 maximize likelihood값을 계산해야 함
  - 하지만 여기서 문제는 전체 z 스페이스에 대한 적분 계산을 할 수가 없다
    - P(x|z)를 구하기 위해서의 x는 어떤 이미지를 의미하는데, 여기서 이미지는 아직 생성된 것이 아니므로 (샘플링을 해봐야 알 수 있는 값이마, 전체 z 스페이스에 대한 샘플링도 무한대로 할 수 없음) 결론적으로는 likelihood를 계산할 수 없음
  - 그럼 이 값이 Maximize는 할 수 있는지 수학적으로 분석해보자
    - $P_\theta(z)$는 가우시안 분포이며, $P_\theta(x|z)$는 Decoder neural network로 표현 (샘플을 알 수 있음)
    - 단, 적분 계산은 불가능 → 모든 z 스페이스에 대해서 샘플링을 무한번 하는 것이 불가능 하므로 → 따라서 $P_\theta(x)$를 알 수 없음

Data likelihood: $p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$

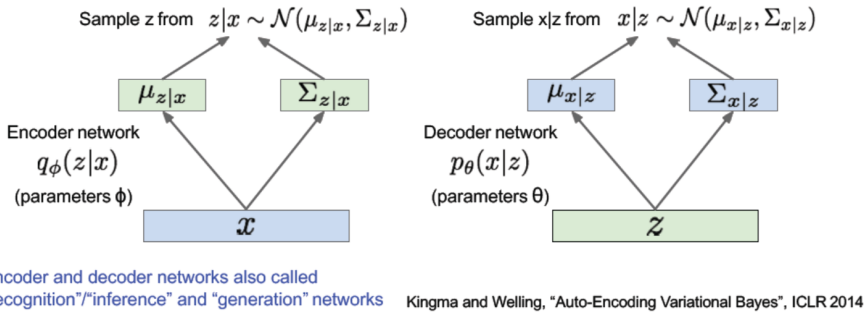
Posterior density also intractable: $p_\theta(z|x) = p_\theta(x|z) p_\theta(z) / p_\theta(x)$

Solution: In addition to decoder network modeling $p_\theta(x|z)$, define additional encoder network $q_\phi(z|x)$ that approximates $p_\theta(z|x)$

Will see that this allows us to derive a lower bound on the data likelihood that is tractable, which we can optimize

- 베이지안 룰에 따라, $P_\theta(z|x) = P_\theta(x|z)P_\theta(z)/P_\theta(x)$로 표현할 수 있으며, 앞의 이유로 $P_\theta(z|x)$ 또한 알 수 없음
- 아까 z로부터 x를 구하는 것이 decoder라 했으니 $P_\theta(z|x)$는 encoder라 부르고 이는 계산이 안되니 Neural Network를 활용하여 approximation 하도록 하자. 그리고 실제로 이 Likelihood가 Maximize될 수 있는지 확인해보자
- Likelihood의 Maximization 확인 방법
  - $P_\theta(x)$의 함수의 Lower Bound를 계속 높여서 Maximize할 수 있는지 알 수 있음 → 일단 Lower Bound가 실제로 존재하는지를 확인해야 함

- Encoder Network와 Decoder Network
  - Encoder Network
    - Input x를 입력받아 z로 맵핑하는 역할 → 확률적으로(Probabilistic) 맵핑
    - 즉, 가우시안 분포를 가진 z의 평균과 공분산을 뽑아내는 네트워크를 말함
  - Decoder Network
    - 어떤 z로부터 고차원 이미지를 표현하는 평균과 공분산을 뽑아내는 네트워크
  - 따라서 이론적으로 Encoder Network와 Decoder Network는 서로 이어져 있어야 하는데, Output과 Input이 서로 다른 문제가 발생
    - 원인 : Encoder Network의 Output은 값 두개, Decoder Network의 Input은 Vector형태
    - 해결방법 : 가우시안 분포를 특정하는(평균과 공분산을 통해) 랜덤 샘플링을 통해 Vector를 만들어 Decoder의 Input으로 입력하여 해결



- Lower bound 확인방법
  - 손쉬운 계산을 위해서 Log likelihood로 계산

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

Taking expectation wrt. z (using encoder network) will come in handy later

  - $z \sim q_\phi(z|x^{(i)})$: Encoder Network가 표현하는 z의 가우시안 분포로부터 샘플한 z
  - $E[\log p_\theta(x^{(i)})]$: $\log p_\theta(x^{(i)})$의 적분한 값
  - 베이지안 룰에 의하여 위의 식을 풀어서 쓰면 다음과 같음

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

  - 분모와 분자에 $q_\phi(z|x^{(i)})$를 곱해줌

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

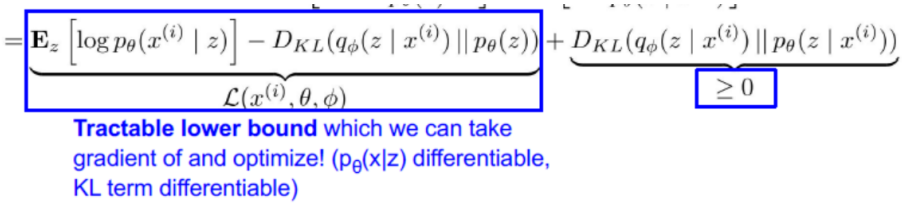
  - 그럼 Log의 성질에 의하여 각 항을 분해할 수 있음

$$\begin{aligned}
\log p_\theta(x^{(i)}) &= \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] && (p_\theta(x^{(i)}) \text{ Does not depend on } z)\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] && \text{(Bayes' Rule)}\\
&= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] && \text{(Multiply by constant)}\\
&= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] && \text{(Logarithms)}
\end{aligned}$$

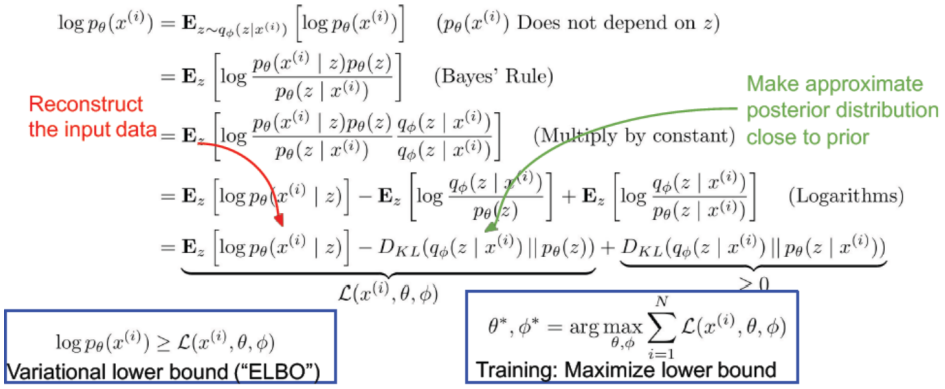
  - D_{KL} : KL 다이버전스

- 어떤 두 가지 PDF, P와 Q가 있을때 두 PDF가 얼마나 비슷한지를 나타내는 함수
  - 두 PDF의 분포가 다를 때 : $D_{KL}(P||Q)$값이 높음
  - 두 PDF의 분포가 같을 때 : $D_{KL}(P||Q)$값이 낮음
- 여기서는 $E_z[\log \frac{q_\theta(z|x^{(i)})}{P_\theta(z)}]$가 D_{KL}의 정의와 같음
- D_{KL} 의 특징은 항상 0이상의 값임 (양수이거나 0임)
  - 똑같을 때 0이고 양수임

- 식의 각 항이 의미하는 바
  - $E_z[\log p_\theta(x^{(i)}|z)]$: 얼마나 z로부터 실제와 비슷한 이미지로 잘 만들어 내는지 여부 = Decoder Network가 내놓는 $P_\theta(x|z)$가 실제 $P_\theta(x|z)$와 유사한 정도를 나타냄 → 크기가 클 수록 θ가 시스템을 잘 표현하고 있음을 나타냄 → Decoder Network가 잘 동작하는지 여부
  - $D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z))$: Encoder가 x로 부터 예상한 z의 분포가 가우시안 분포에 얼마나 근사한지 여부 ($p_\theta(z)$가 가우시안 분포라 가정하였으므로)를 나타내는 Regularized Term
    - 음의 부호가 붙어있으므로 두 분포가 유사하여 마이너스 값이 적어질 수록 Maximization을 이룰 수 있음
  - $D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z|x^{(i)}))$: Encoder가 x로 부터 예상한 z의 분포가 실제 x로부터 z로 가는 PDF와 얼마나 유사한지 정도를 나타냄
    - 단, $p_\theta(z|x^{(i)})$는 앞서 설명했듯이 계산이 불가능하므로 이 항의 값은 알수없고 그저 D_{KL}의 특성에 따라 0또는 양수의 값이라는 것만 추측할 수 있음



- 따라서, 앞의 두 항의 합이 Lower Bound라는 것을 알 수있음. 뒤의 항은 값은 모르고 0또는 양수의 값이라는 추측만 알고 있기 때문에
- 따라서 앞의 목적에 따라($\log p_\theta(x^{(i)})$를 Maximize) 앞의 두 항이 Lower Bound이므로 이를 Maximize시켜주면서 학습하면 $\log p_\theta(x^{(i)})$가 Maximize 한다는 사실을 알 수 있음



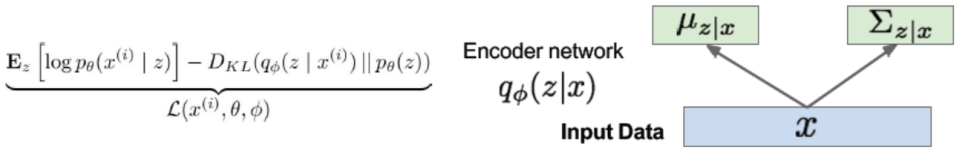
- (강사도 이 식을 이해하는데 6개월 걸림;;;;)

- 모델 구조

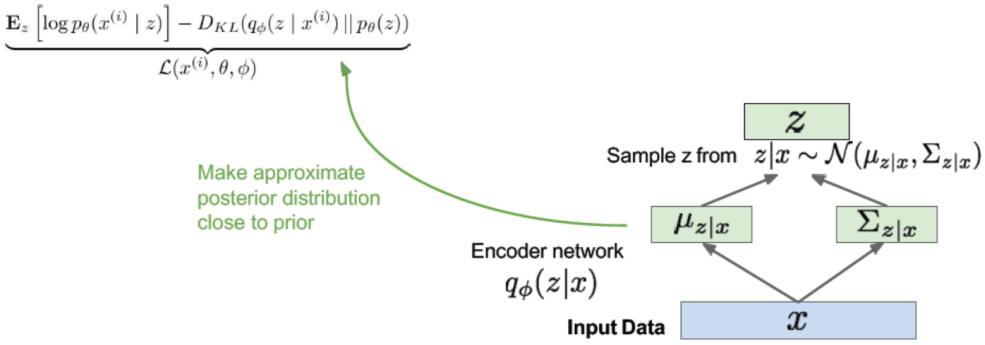
1. **Training 과정**

① Encoder Network는 x를 입력받아 가우시안을 결정하기 위해서 Mean과 Covariance를 뽑아냄

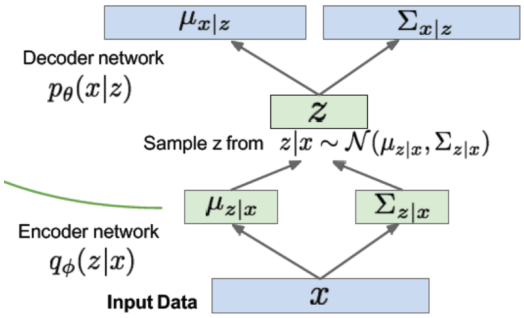
(중요 : 여기서 우리는 계산식을 알 수 있는 것이 아니라 각각의 값을 통해서 계산한 결과를 활용할 수만 있음을 기억)



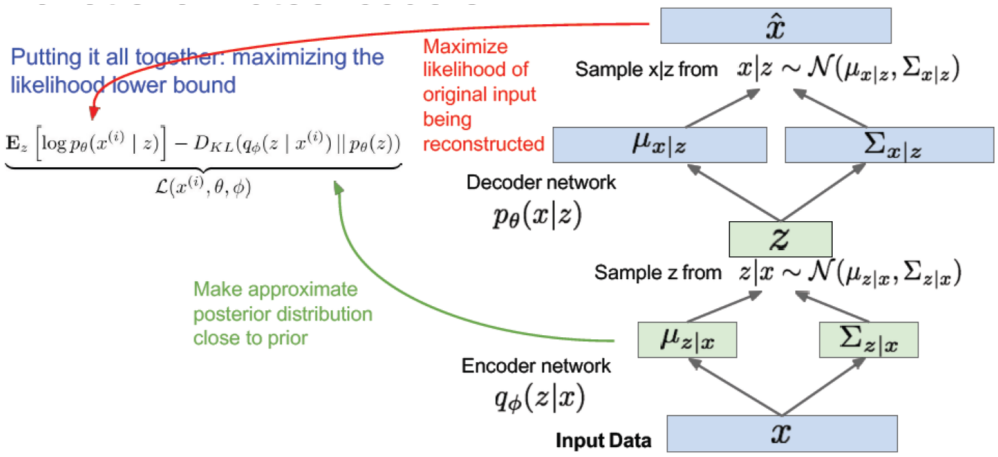
② 앞서 계산한 (z | x)의 Mean과 Covariance를 가지는 가우시안 분포에서 z값을 샘플링



③ 샘플링한 z값을 decoder network에 입력



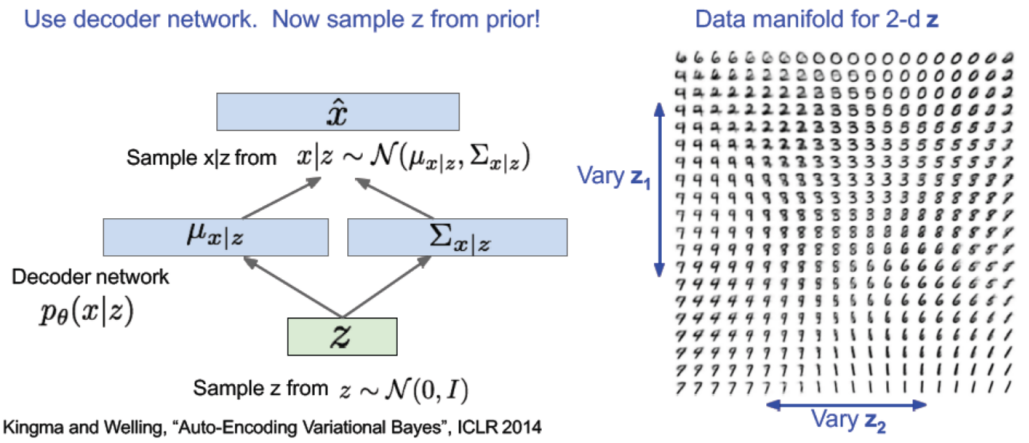
④ Decoder network는 latent vector를 입력받아 각각의 x를 대표하는 Mean과 Covariance를 추출하고, 이를 추종하는 가우시안 분포에서 샘플링을 하면 $\hat{x}$가 도출이 되고 이는 새로운 이미지를 의미함



⑤ 샘플링을 통해 얻어지는 $\hat{x}$의 갯수가 많아 질수록, $E_z\left[\log p_\theta\left(x^{(i)}|z\right)\right]$값을 우리는 도출할 수 있게됨

- 즉, 이 과정을 통해 우리는 Likelihood를 어떤 값으로 계산할 수 있고, 이를 미분하여 Backpropagation하여 gradient ascent를 하면(=Log likelihood를 maximize) 목적에 맞게 θ와ϕ가 step을 밟아 나갈 것임

2. 새로운 이미지를 만들어가는 과정



- 스탠다드 가우시안으로 가정한 z에서 임의로 샘플링하여 기존의 z대신 입력함
  - 기존의 z는 x가 Decoder Network에서 샘플링한 Encoding된 z였었음
- 그럼 새로운 이미지가 생성되어 도출됨
  - 단, 우리는 z값이 어떤 특정 Feature의 Degree를 학습을 할 수는 없다 (예 : 얼굴 이미지를 생성하는데, 웃는 얼굴의 정도를 나타내는 이미지만 도출되도록 할 수는 없음) → Data에 Degree에 대한 Label이 없으므로
  - 그저 Data의 모든 값을 변화시켜보면서 무엇을 나타내고 있는지를 알 수는 있음 → Unsupervised Learning이기 때문

- VAE의 장점과 단점
  - 장점
    - 수학적으로 접근한 이론적인 generative model임
    - Encoder Network를 학습하기 때문에 Autoencoder로서의 장점인 추가 학습에 높은 효율을 나타낸다 (학습 후 Decoder Network를 떼서 다른 Data를 학습할 경우 Label이 적어도 높은 학습 효율을 나타내는 장점)
  - 단점
    - Maximizes lower bound of likelihood 방식은 좋지만 실제 계산식으로 도출하는 방식이 아니기 때문에 Pixel RNN이나 Pixel CNN 방식보다 좋은 성능은 보여주지 못한다 (단, Pixel RNN과 Pixel CNN은 엄청 느림)
    - z를 가우시안 분포로 가정했지만 실제로는 가우시안이 아닌 경우도 많기 때문에 이미지에 Blur 효과가 나올 수 있음