Deep Neural Network의 학습 방법

2021.12.19 김은희

Classic Machine Learning

- 1단계 : Collect data
- 2단계 : Define functions

Model : $f_\theta(\cdot)$ kNN? SVM? Random Forest? …
Loss : $L(f_\theta(x), y)$

- 3단계 : Training "Find $\theta^* = \underset{\theta}{\operatorname{argmin}} L(f_\theta(x), y)$"
- 4단계 : Testing $y_{new} = f_{\theta^*}(x_{new})$ 와 y 비교 → 학습 결과 확인

Deep Neural Networks

- 1단계 : Collect data
- 2단계 : Define functions

Model : $f_\theta(\cdot)$ CNN? RNN? GAN? … 여기서 θ는 네트워크의 weight, bias를 의미
Loss : $L(f_\theta(x), y)$ 일반적으로 MSE나 Cross-Entropy Error를 사용

- 3단계 : Training "Find $\theta^* = \underset{\theta}{\operatorname{argmin}} L(f_\theta(x), y)$"
- 4단계 : Testing $y_{new} = f_{\theta^*}(x_{new})$ 와 y 비교 → 학습 결과 확인

출력 $f_\theta(x)$과 정답 y가 가까워지게 하자

- 목표를 바라보는 관점 2가지
  - 파라미터를 찾아 → 함수 찾는 관점
  - 확률 분포의 파라미터를 찾는 관점 (Maximum Likelihood 관점)

함수를 찾는 관점

$y_{new} = f_{\theta^*}(x_{new})$ 인 파라미터를 찾자

$\underset{\theta}{\operatorname{argmin}} L(f_\theta(x), y)$인 θ^*를 찾는 방법

- Motivation : θ 에서 $\Delta\theta$만큼 업데이트 → $\mathrm{L}(\theta)$와 $\mathrm{L}(\theta + \Delta\theta)$비교

While $\mathrm{L}(\theta + \Delta\theta) < \mathrm{L}(\theta)$:

$\theta \mathrel{+}= \Delta\theta$

if $\mathrm{L}(\theta + \Delta\theta) == \mathrm{L}(\theta)$:

break

$\underset{\theta}{\operatorname{argmin}} L(f_\theta(x), y)$인 θ^*를 찾는 방법

특히 θ의 dimension이 커질수록 복잡

- $\mathrm{L}(\theta + \Delta\theta) < \mathrm{L}(\theta)$ 인 $\Delta\theta$ 를 어떻게 찾을까? Stopping point는?
- Taylor expansion을 사용하여 근사하기
  - $f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \cdots$
  - 1st derivative term만 사용해 근사하면 $\mathrm{L}(\theta + \Delta\theta)$는 다음과 같이 근사
  - $\mathrm{L}(\theta + \Delta\theta) \approx L(\theta) + \Delta\theta \cdot \nabla L$
  - $\mathrm{L}(\theta + \Delta\theta) - L(\theta) = \Delta L = \Delta\theta \cdot \nabla L$ 으로 식 변형을 해 보자 … (1)
  - 희망사항은 $\Delta L < 0$이 되는 것 (⇔ update 후 loss의 감소)
  - 여기서 $\Delta\theta = -\eta\nabla L \ (\eta > 0)$ (learning rate) 라 하고 (1)에 대입
  - 그럼 $\Delta L = -\eta\|\nabla L\|^2 < 0$ 이 되어 희망사항 충족!
  - 따라서 $L(\theta + \Delta\theta)$에서 $\Delta\theta = -\eta\nabla\mathrm{L}$이어야 한다.

Gradient descent

Backpropagation

- Loss function의 미분 값이 Deep Neural Network을 학습시키는데 가장 중요하다.
- 그런데 수많은 레이어와 파라미터에 대해 loss function을 일일히 미분하는 것은 연산량을 많이 요구함 (DNN의 암흑기)
- Backpropagation은 복잡한 미분을 chain rule을 이용해 계산해 나갈 수 있음
  - 원하는 gradient = upstream gradient * local gradient
  - https://youtu.be/573EZkzfnZ0 [Sung Kim 교수님 강의 추천해요!]

확률 분포의 파라미터를 찾는 관점

$y_{new} \sim f_{\theta^*}(x_{new})$인 파라미터를 찾자

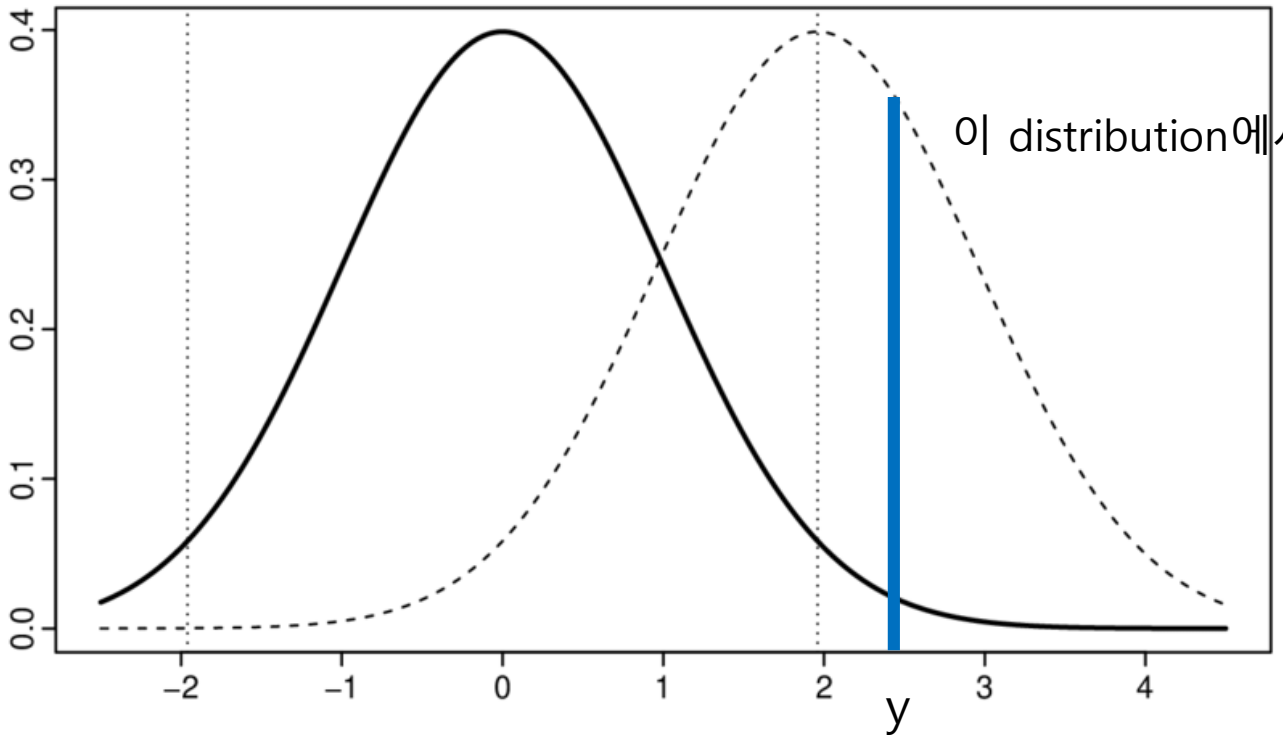
출력 $f_\theta(x)$과 정답 y가 가까워지게 하자

- 목표를 바라보는 관점 2가지
  - Loss function을 정답과 출력값의 차이로 정의하는 관점
  - Maximum Likelihood로 바라보는 관점

- 즉, 목표를 $P(y|f_\theta(x))$의 최대화로 해석할 수 있다.
  - 네트워크의 출력값들이 주어져 있을 때 = 확률분포가 정해져 있을 때
  - 정답 출력이 나올 확률의 최대화

출력 $f_\theta(x)$과 정답 y가 가까워지게 하자

- 따라서, 이 관점을 사용할 때에는 $P(y|f_\theta(x))$이 어떤 분포를 따를지도 미리 정해야 한다. 즉, 출력값 == 확률분포의 파라미터



출력 $f_\theta(x)$과 정답 y가 가까워지게 하자

- 여기서 loss를 negative loss likelihood로 독특하게 정의하는데
  - $\theta^* = \underset{\theta}{\text{argmin}}[-\log(P(y|f_\theta(x))]$
  - 이런 θ^*찾기 == Maximum log likelihood가 최대가 되게 하는 θ^* 찾기
- θ^*의 발견 == 확률분포 모델 발견 → Sampling 가능!
- $y_{new} \sim P(y|f_{\theta^*}(x_{new}))$

Source

- 오토인코더의 모든 것
  https://www.youtube.com/watch?v=o_peo6U7IRM