D&A

Deep Session 3차시

학습 기법들.

2022 / 03 / 24

D&A 운영진 이수빈

CONTENTS.

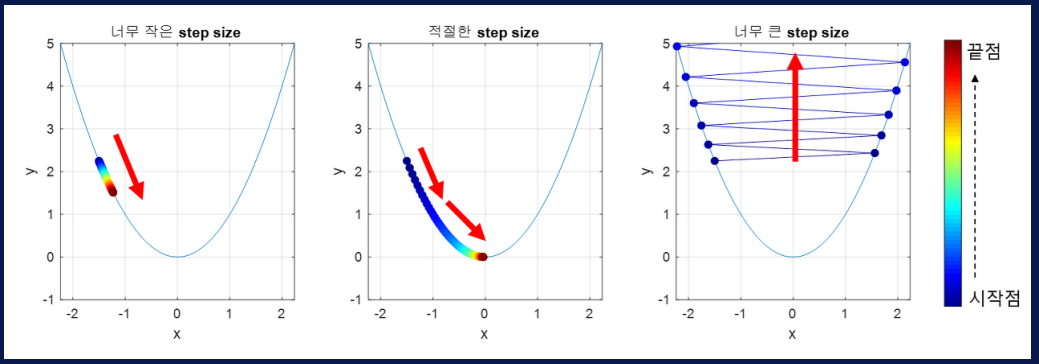
01. 최적화

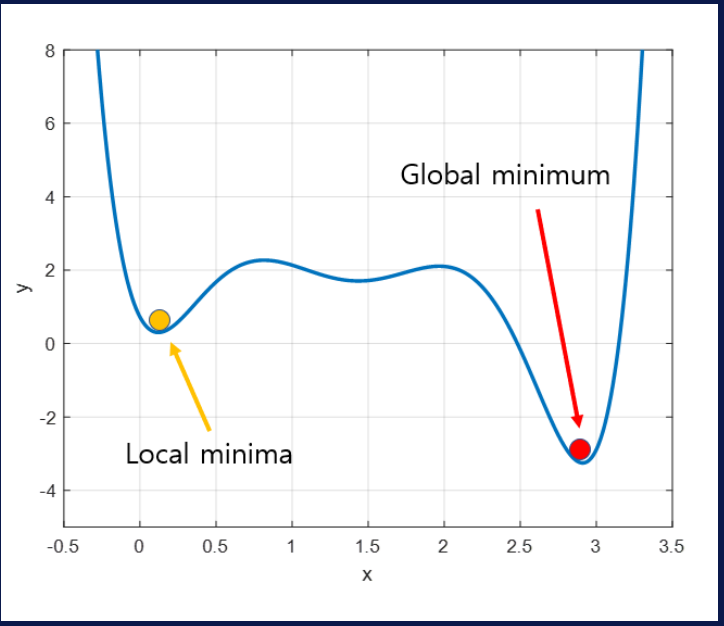
최적화란?

가중치 매개변수의 최적값 탐색하는 방법, 경사 하강법의 단점을 보완할 수 있는 다양한 최적화 방법이 고안되었다.

경사 하강법 (Gradient Descent)의 단점

- 학습할 때마다 모든 데이터셋 사용하기 때문에 학습이 오래 걸리며 메모리 문제가 발생할 가능성이 높다.
- 적절한 학습률을 찾기 어렵다.
  - 너무 작을 경우, 시간이 오래 걸림
  - 너무 클 경우, 최솟값 계산이 어렵고 loss 값이 커지는 방향으로 진행될 가능성이 높다.
- 목적인 Global minimum을 찾지 못하고, Local minimum에 갇힐 수 있다.



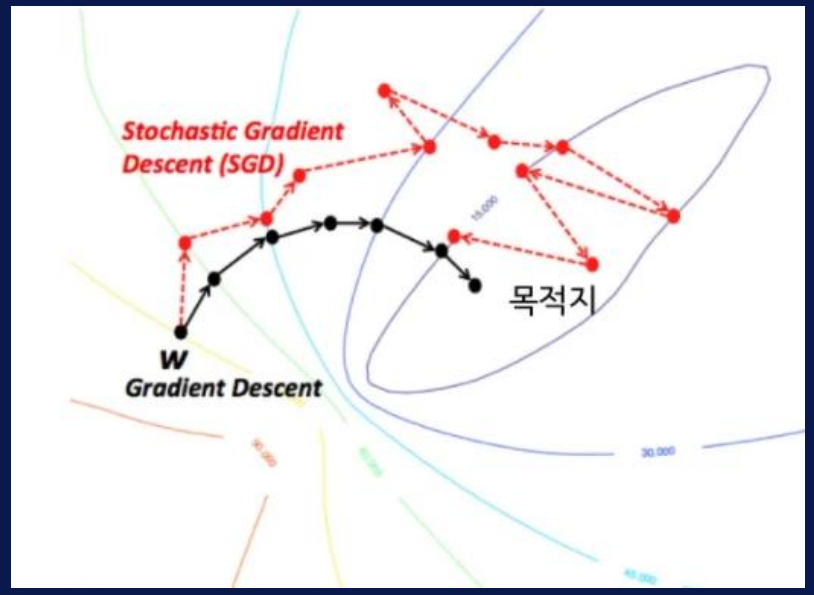


01. 최적화

SGD란?

확률적 경사 하강법 (Stochastic Gradient Descent)

- 경사 하강법의 단점을 보완
  - 모든 데이터셋 사용 → 랜덤하게 추출한 일부 데이터 사용 → 자주 업데이트하여 보다 빠른 속도로 최적해를 찾는다.
  - 속도를 제외한 다른 부분은 개선하지 못하였다. (학습률 설정, local minimum)



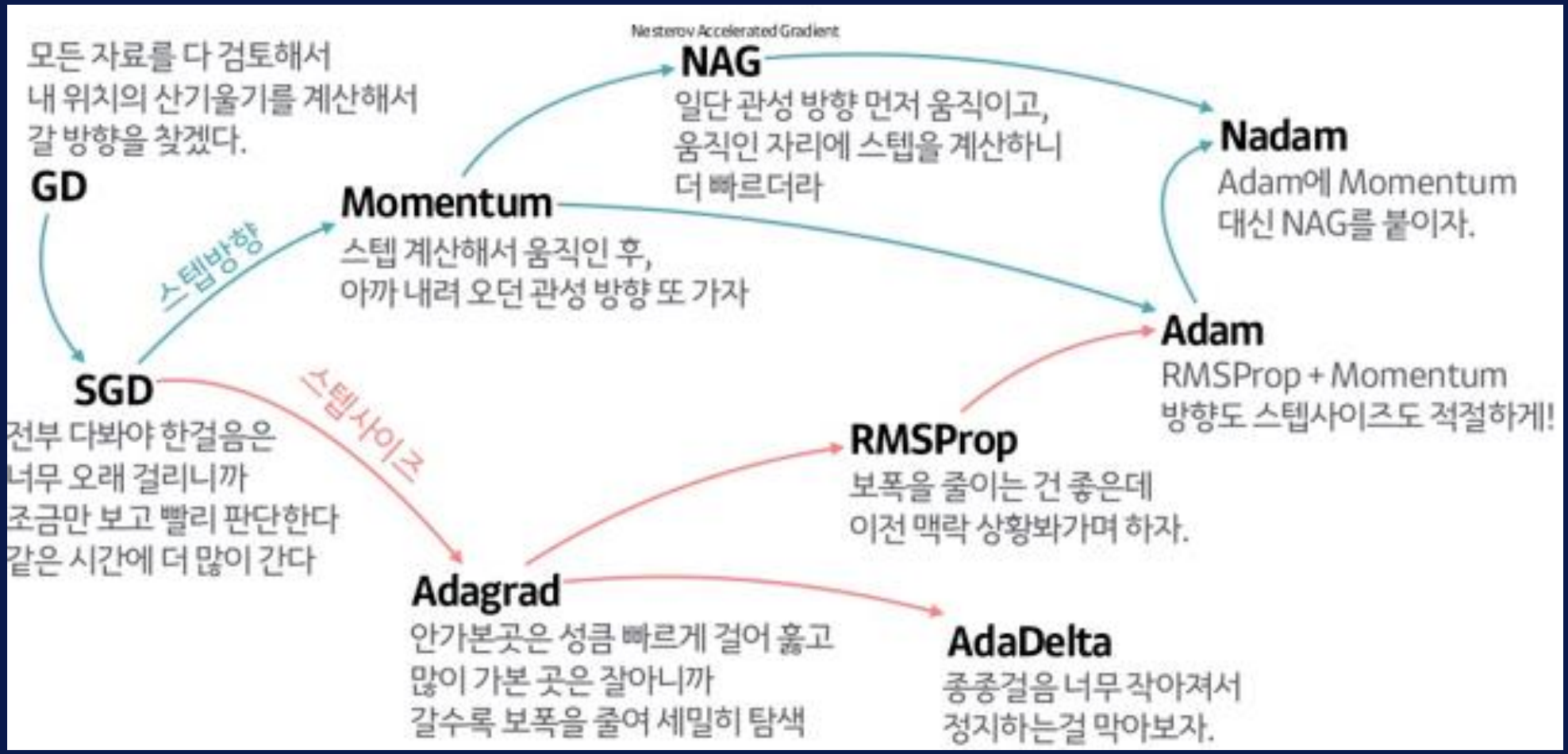
$$W(t+1) = W(t) - \alpha \frac{\delta}{\delta w} Cost(w)$$

- 비등방성 함수 : 비효율적인 탐색 경로 (지그재그)

  ⇒ 헤매지 않고 가는 방법? 어느 방향으로 움직일지, 얼만큼 움직일지 결정

01. 최적화

Optimizer 발달 계보 : 어느 방향으로 움직일지 (스텝 방향), 얼만큼 움직일지 (스텝 사이즈, 학습률)



01. 최적화

Momentum 이란?

SGD의 경로 변동을 줄이기 위해 제안된 최적화 방법

이동 시 관성 방향을 고려하여 진동과 폭 줄이기

local minimum 지나칠 수 있음

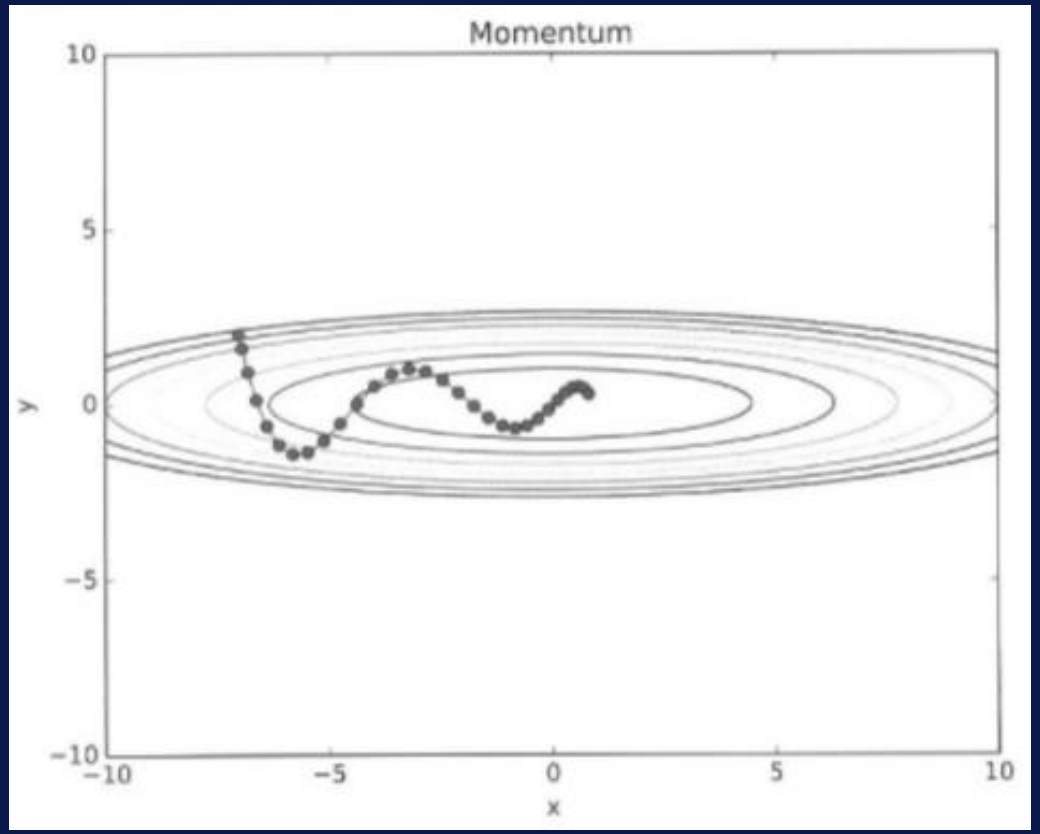
$$V(t) = m * V(t-1) - \alpha \frac{\partial}{\partial w} Cost(w)$$

$$W(t+1) = W(t) - V(t)$$

단점 : 관성으로 인해 최적값에 수렴되지 않을 경우 있음

⇒ NAG (Nesterove Accelerated Gradient)

관성 방향으로 움직인 후 기울기 계산하여 최적값을 찾는다.



01. 최적화

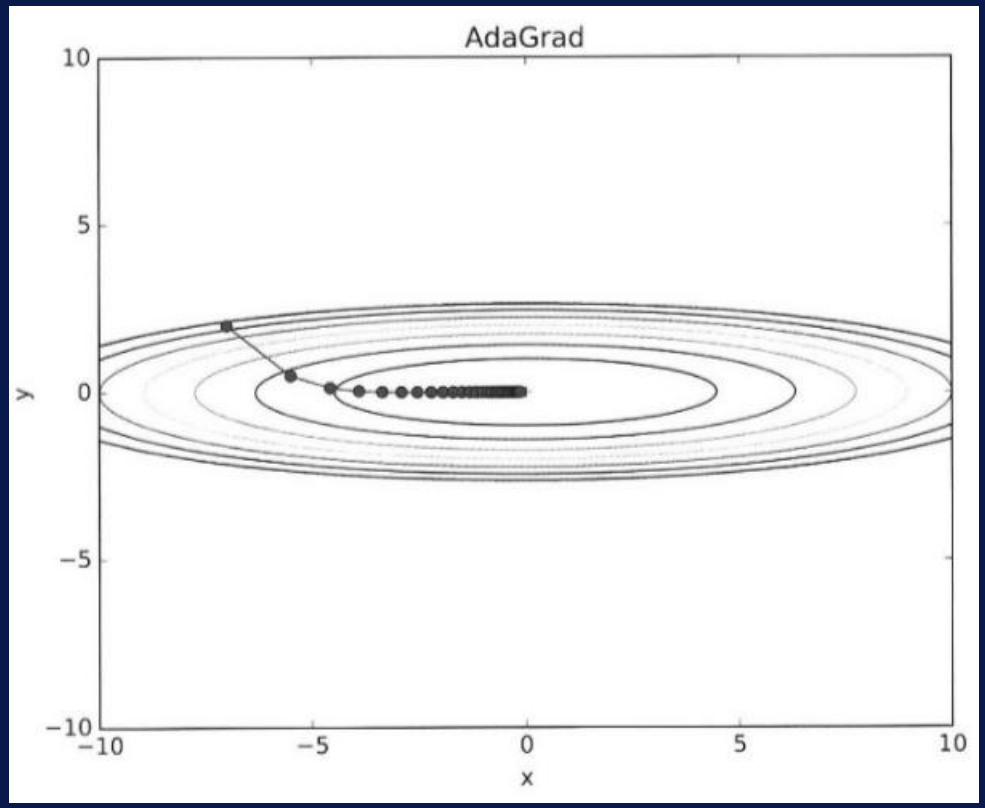
AdaGrad란?

이전 gradient를 기반으로 학습률에 변화를 주는 방법

크게 변동이 있을 경우 학습률 감소시키고, 반대의 경우에는 증가시킨다.

$$G(t) = G(t-1) + (\alpha \frac{\partial}{\partial w} Cost(w(t))^2 = \sum_{i=0}^{t} (\frac{\partial}{\partial w(i)} Cost(w(i))^2$$

$$W(t+1) = W(t) - \alpha * \frac{1}{\sqrt{G(t) + \epsilon}} * \frac{\partial}{\partial w(i)} Cost(w(i))$$



단점

- 학습률이 0에 수렴할 수 있다.
  ⇒ RMSProp
- Gradient 양이 적을 때 움직임이 멈출 수 있다.
  ⇒ AdaDelta

01. 최적화

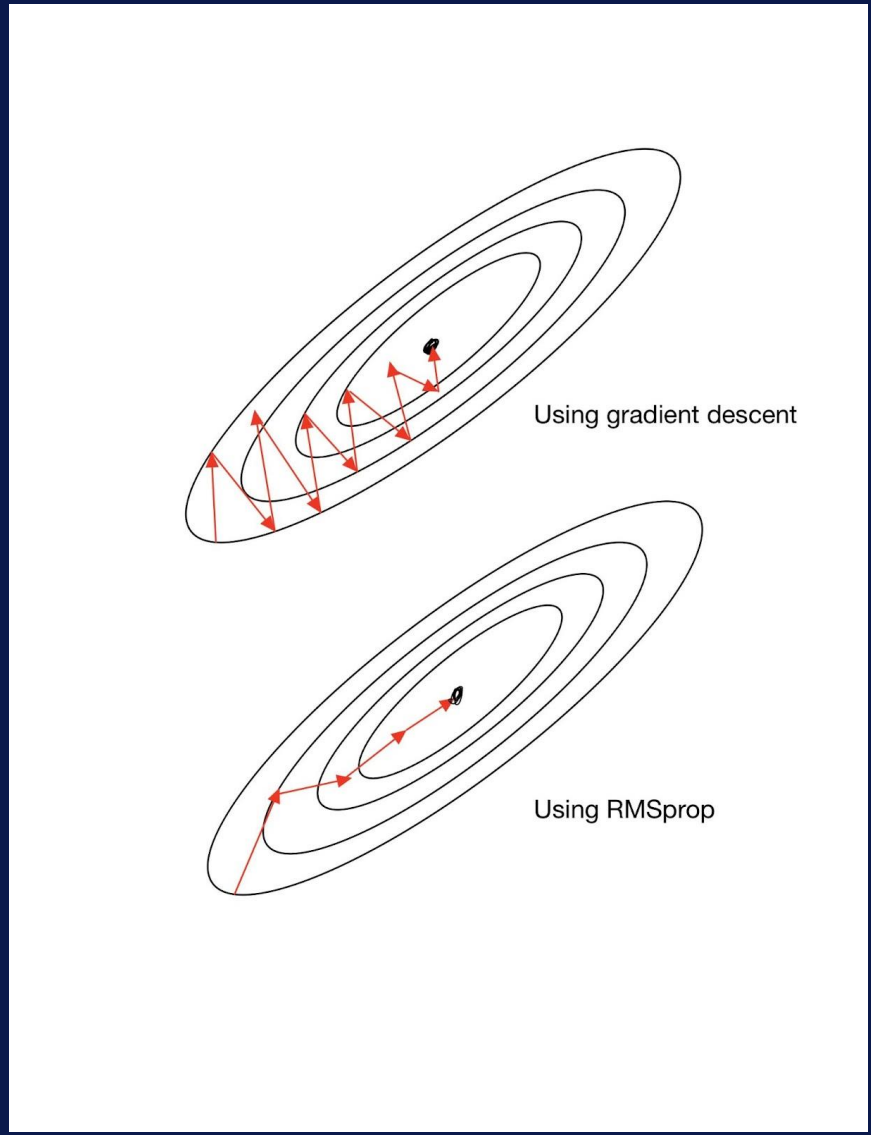
RMSProp란?

AdaGrad의 단점인 학습률이 0에 수렴하는 문제를 해결하기 위해 제안

최근 n개의 gradient만 반영한 지수이동평균 고려

$$G = \gamma G + (1-\gamma)(\nabla_\theta J(\theta_t))^2$$

$$\theta = \theta - \frac{\eta}{\sqrt{G+\epsilon}} * \nabla_\theta J(\theta_t)$$



01. 최적화

Adam이란?

RMSprop + Momentum

- 3개의 파라미터 : 학습률, 모멘텀용 계수, 2차 모멘텀용 계수

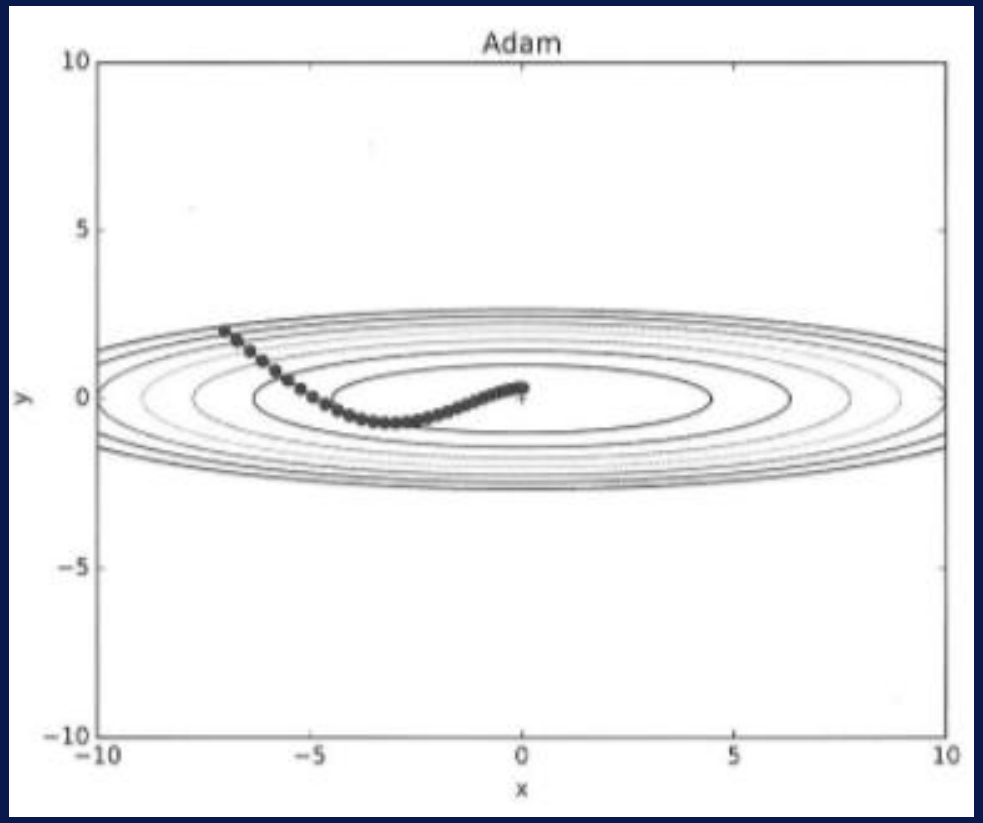
$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta J(\theta)$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla_\theta J(\theta)^2)$$

$$\hat{m} = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v_t} = \frac{v_t}{1-\beta_1^t}$$

$$\theta = \theta - \frac{\eta}{\sqrt{\hat{v_t} + \epsilon}}$$



02. 가중치 초기화

신경망 학습은 가중치를 랜덤하게 초기화 하며 loss의 최소화 값을 찾는다.

빠르게 학습시키기 위해선 가중치의 초깃값을 어떻게 설정할 것인지 중요하다.

그냥 0 (균일한 값)으로 설정한다면?

Back propagation 진행 시 모든 가중치 값이 똑같이 업데이트 된다.

⇒ 가중치 고르지 않게 랜덤하게 설정해야 한다.

02. 가중치 초기화

LeCun Initialization

- LeCun Normal Initialization

$$W \sim N(0, Var(W))$$

$$Var(W) = \sqrt{\frac{1}{n_{in}}}$$

(n_{in} : 이전 layer(input)의 노드 수)

- LeCun Uniform Initialization

$$W \sim U(-\sqrt{\frac{1}{n_{in}}}, \ +\sqrt{\frac{1}{n_{in}}})$$

(n_{in} : 이전 layer(input)의 노드 수)

- 분포에서 추출한 랜덤한 값으로 초기화

Xavier Initialization

- Xavier Normal Initialization

$$W \sim N(0, Var(W))$$

$$Var(W) = \sqrt{\frac{2}{n_{in} + n_{out}}}$$

(n_{in} : 이전 layer(input)의 노드 수, n_{out} : 다음 layer의 노드 수)

- Xavier Uniform Initialization

$$W \sim U(-\sqrt{\frac{6}{n_{in} + n_{out}}}, \ +\sqrt{\frac{6}{n_{in} + n_{out}}})$$

(n_{in} : 이전 layer(input)의 노드 수, n_{out} : 다음 layer의 노드 수)

- 이전과 다음 노드 수 고려하여 확률분포 조정
- 활성화 함수 sigmoid와 같은 S자 모양일 때

He Initialization

- He Normal Initialization

$$W \sim N(0, Var(W))$$

$$Var(W) = \sqrt{\frac{2}{n_{in}}}$$

(n_{in} : 이전 layer(input)의 노드 수)

- He Uniform Initialization

$$W \sim U(-\sqrt{\frac{6}{n_{in}}}, \ +\sqrt{\frac{6}{n_{in}}})$$

(n_{in} : 이전 layer(input)의 노드 수)

- ReLU 함수의 비효율성 보완

03. 정규화

Internal Covariate Shift란?

학습 과정에서 계층 별로 입력 데이터의 분포가 달라 나타나는 문제로 학습 속도가 느려지며 Gradient Vanishing에 영향을 준다.

- 정규화 방식
  - Batch Normalization
  - Layer Normalization
  - Insatance Normalization
  - Group normalization
- 장점
  - 학습 속도 감소
  - 오버피팅 억제
  - 가중치 초기값에 대한 의존성 감소 → 하이퍼파라미터 설정 자유로워짐

03. 정규화

Batch Normalization

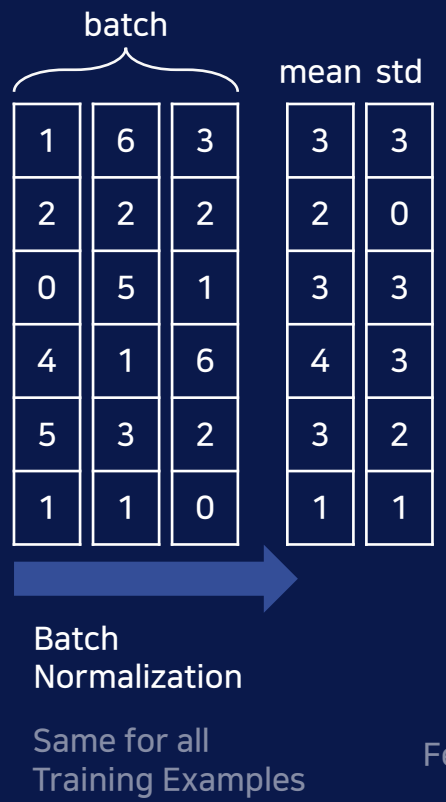
매 순간 input을 N(0, 1)로 정규화 = Whitening

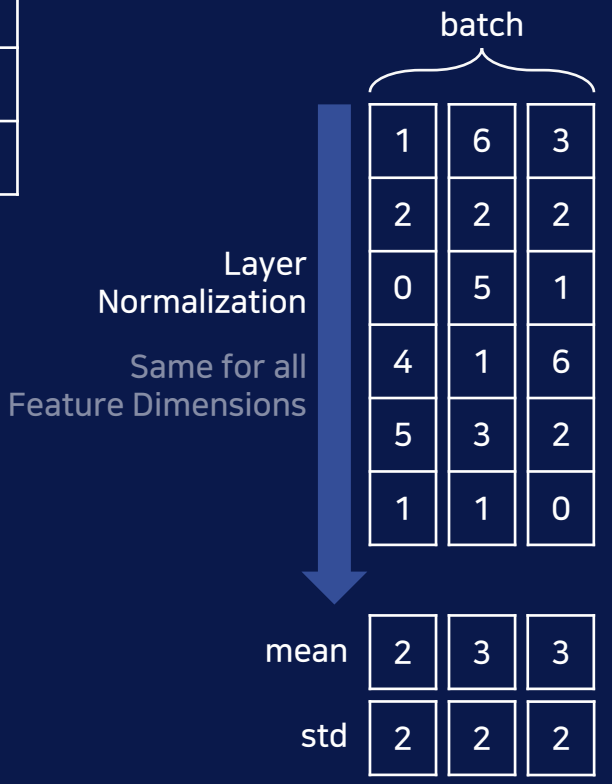
- 장점
  - Back propagation 시 parameter scale 영향을 받지 않아 학습률을 크게 잡을 수 있어 빠르게 학습 가능
  - 자체 regularization 효과
    → L1, L2, Dropout 과정 생략 가능

Layer Normalization

Batch의 모든 feature 정규화

Batch size = 1일 때, batch normalization 불가능한 경우 사용 가능





04. 오버피팅 억제

오버피팅이란?

훈련 데이터에만 지나치게 적응되어 다른 데이터에는 제대로 대응하지 못하는 경우

오버피팅이 주로 일어나는 경우

- 매개변수가 많고 표현력이 높은 모델 ⇒ 매개변수에 대한 가중치 감소, 보다 단순한 모델을 위한 Dropout
- 적은 훈련 데이터

04. 오버피팅 억제

가중치 감소란?

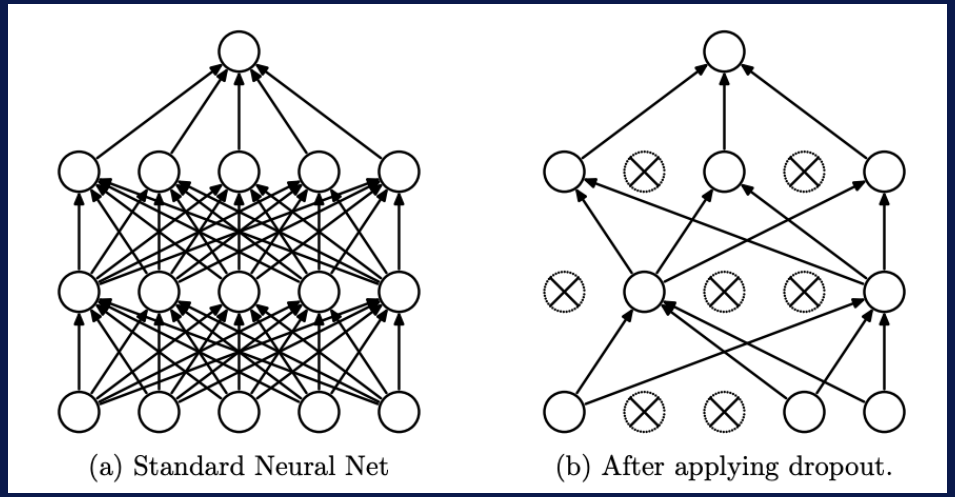
학습 과정에서 큰 가중치에 대해 그에 맞는 큰 패널티 부과한다. (ex. L2 norm)

Dropout이란?

뉴런을 임의로 삭제(w=0)하여 모델을 학습시키고, 시험 때는 모든 뉴런에 신호를 전달한다.

단, 시험 때는 각 뉴런 출력에 훈련 때 삭제한 비율 곱하여 출력한다.

무작위로 선택된 뉴런으로 학습 시킨 모델은 매번 다른 모델을 학습 시킨 것과 같기 때문에 ensemble 효과를 얻을 수 있다.



(a) Standard Neural Net (b) After applying dropout.

05. 하이퍼파라미터 최적화

- 검증 데이터 사용하기
  - 훈련 데이터 : 매개변수 학습
  - 검증 데이터 : 하이퍼파라미터 성능 평가
  - 시험 데이터 : 신경망 성능 평가
- 대략적인 범위 설정하여 최적화 값까지 범위 좁히기
  - 범위 설정 → 범위 내 값 무작위 추출 → 추출한 값으로 학습하여 검증 데이터로 평가
  - 위 과정 반복하여 범위 좁히기
  - 베이지안 최적화, 랜덤 서치, 그리드 서치 와 같은 방법 사용

과제

- MLP 모델에서 dropout 비율, 활성화 함수, Initialization 방법, Batch Normalization 부분 변경해보며 모델 성능 비교해보기

첨부자료 출처

01. 최적화

경사하강법 단점 이미지 출처
: https://angeloyeo.github.io/2020/08/16/gradient_descent.html
SGD 이미지 출처
: https://wikidocs.net/book/498
Optimizer 발달 계보 이미지 출처
: https://www.slideshare.net/yongho/ss-79607172
Momentum, AdaGrad, Adam 이미지 출처
: 밑바닥부터 시작하는 딥러닝, 사이토 고키
RMSprop 이미지 출처
: https://wiki.hasty.ai/solvers-optimizers/rmsprop

04. 오버피팅 억제

Dropout 이미지 출처
: https://jmlr.org/papers/volume15/srivastava14a.old/srivastava14a.pdf

---

폰트

네이버 글꼴 모음 _ 나눔 스퀘어 사용
출처 : https://hangeul.naver.com/font

D&A

Deep Session 3차시 학습 기법들

Thank You.

2022 / 03 / 24

D&A 운영진 이수빈