Weight Decay (init_ wieght)

○ 가중치 초기값은 균일한 값으로 설정되면 안된다. → $\frac{dE}{dw_i}$ 에서 $w_{t+1} = w_t - \eta \cdot \frac{dE}{dw_t}$ 로

○ abs(w)가 큰 값을 가지면 sigmoid에선 0 or 1 모든 가중치의 값이 똑같이 갱신되기 때문에
relu에선 폭주,난폭값을 가짐 $\begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$ → 가중치를 여러개 갖는 의미가 사라짐

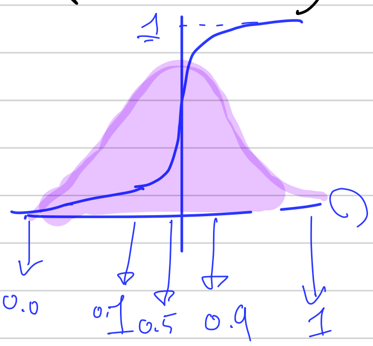
그럼 weight 들의 초기값이 학습에 영향을 줄까?
⇒ activation function에 따라 원하는 분포가 다름.

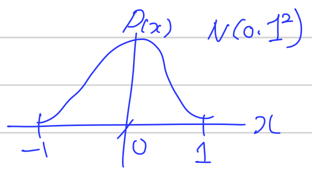
if) activation = Sigmoid(x)

$w = N(0, 1^2)$

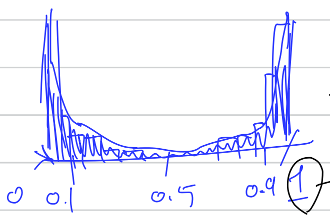
$$\begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix} = \begin{pmatrix} N(0,1) & N(0,1) \\ N(0,1) & N(0,1) \end{pmatrix}$$

μ σ^2





〈출력값 분포〉

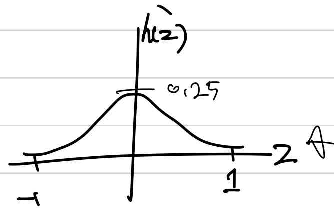


histogram

○ 모든 $h(\sum x_i w_i)$이 0 or 1을 갖는다.
○ 출력값 이자 입력값인 $y_i = 0$이면 $w_i \cdot y_i$ term이 0이되어 가중치가 갱신되지 않고
$y_i = 1$이면 back Propagation

$$\frac{dE}{dw} = \frac{dE}{do}\frac{do}{dz}\frac{dz}{dw}$$

$$= (t - o)(h(z))(1 - h(z))w$$



→ Vanishing gradient Problem

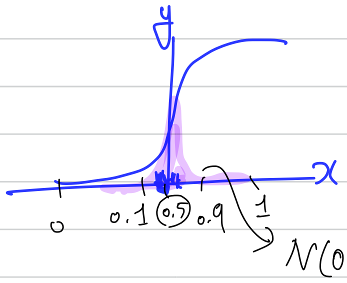
0이 되는 현상

→ 갈수록 ($\frac{dE}{dw}$ 값, 미분값이 적어져 학습이 안되는 현상.

$w_{t+1} = w_t + \eta \cdot \frac{dE}{dw}$

$w = N(0, 0.01)$

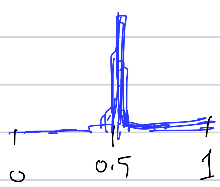
$$\begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix} = \begin{pmatrix} N(0,0.01) & N(0,0.01) \\ N(0,0.01) & N(0,0.01) \end{pmatrix}$$



$h\begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix} = \begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}$

○ 0과 가깝게 $0 \fallingdotseq N(0, 0.01)$

출력값 분포



histogram

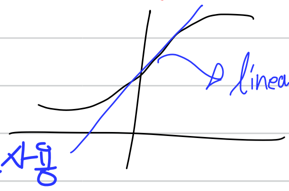
○ 모든 $h(x_i w_i)$ 값들이 0.5로 고정된다.
⇒ 가중치, layer를 쌓는 이유가 없다.
⇒ 학습이 잘된다.
⇒ 각각의 가중치가 asymmetry, 비대칭.
다양하게 가져야 좋은 학습이 된다.

기본 keras kernel init default 값

Xavier with Sigmoid

○ Xavier Glorot
○ Yoshua Bengio
○ 2010

○ activation function 이 linear 할때! → Sigmoid는 좌우대칭, 가운데 부분이 선형.
○ tanh 일때 가장 바람직! (S 자형)



선형적.
ex) $wx+wx^2+wx^3+\dots$ ⇒ 선형적

○ 앞 계층의 node가 n개 일때 $\sigma=\sqrt{\frac{1}{fan_{in}}}$ 의 분포를 사용

○ LeCun Normal initialization

○ 뒤 층까지 고려한 분포 $\sigma=\sqrt{\frac{2}{fan_{in}+fan_{out}}}$ $\sigma=\sqrt{\frac{6}{fan_{in}+fan_{out}}}$

○ Xavier Normal initialization ○ Xavier uniform distribution

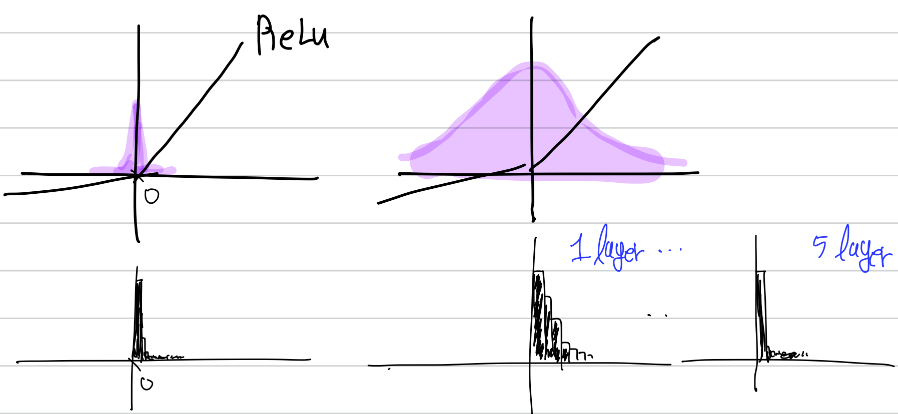
○ input 의 크기와 output의 크기가 커질수록 분산은 작게

He with ReLu

○ 2015

○ activation function 이 non-linear 할때
○ ReLu 계열에 어울림

○ normal distribution 과 uniform distribution 이 있는데 명확한 선택기준 없지만 normal을 쓰는 트랜드라 he 논문에서 말함.



○ 0에 몰려 weight 이 갱신 불가 ○ layer에 얼마 못가 $N(0,0.01^2)$ 때와 마찬가지로 0에 몰려 weight update가 거의 되지 않음

○ He initialization
앞 계층 node가 n_{in} 개 일때 $\sigma=\sqrt{\frac{2}{n_{in}}}$ → 직관적으로는 음수대 값들이 다 0 이 됐기에 그 만큼의 다양성을 위해 2배를 해준다. $\sigma=\sqrt{\frac{6}{fan_{in}}}$

○ He normal distribution ○ He uniform distribute

2 conv

이미지 R,G,B feature map

kernel shape = kernel size + (input channel, filters)

(5×5) $(3, 6)$ $\Rightarrow$ conv2D((5,5),6)

$= (5,5,3,6)$

input channel output channel = filters.

1 conv

kernel shape = $4 + (1, 20)$

$= (4, 1, 20)$

receptive field $= \prod_{i=0}^{k-2}$ kernel shape[i] (k= kernel shape의 차원수)

$= 4$ (conv1D) $k=4$

$= 25$ (conv2D) $k=3$

fan in = receptive field × input channel

(input tensor의 차원크기)

$= 25 \times 3$ ($\underbrace{\text{kernel shape}[k-2]}_{\text{input channel}}$) 2D

$= 4 \times 1$ 1D

fan out = receptive field × output channel

(output tensor의 차원크기) $= 25 \times 6$ ($\underbrace{\text{kernel shape}[k-1]}_{\text{output channels}}$) 2D

$= 4 \times 20$ (kernel shape[k-1]) 1D

Batch Normalization

batch_Norm 각 층의 활성화(y)를 적당히 퍼뜨리기 가중치의 초깃값을 덜 신경쓰게.

○ 2015 Sergey Ioffe, christian Szegedy

○ 단순 whitening이 아닌 신경망안에 포함되어 μ, σ 를 조정

○ scale과 shift 연산을 위해 γ, β 가 들어가게되어

- 정규화한 부분을 원래대로 되돌리는 identity mapping 가능
- 학습을 통해 γ, β 를 정할수 있음.

○ standard = 표준화 → $N(0, 1^2)$ $\quad \frac{x-\mu}{\sigma}$

○ min-max = 정규화 → (0,1) 범위 $\quad \frac{X - \min X}{\max X - \min X}$
(normalization) 가장 작은값, 가장 큰값

input $\vec{X} = \{x_1, x_2, x_3, \cdots x_n\}$

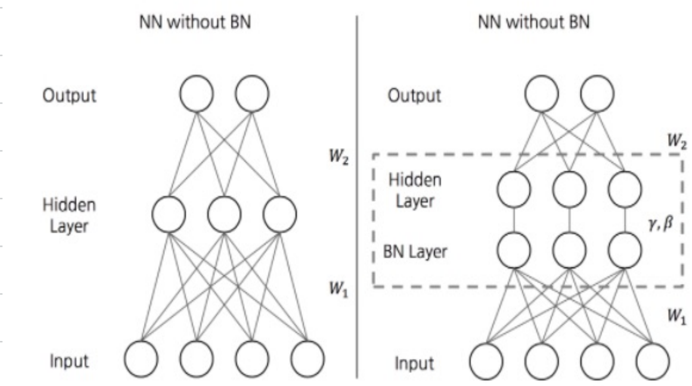
output $BN_{\gamma,\beta}(\vec{X})$

mean $\mu_{\vec{X}} = \frac{1}{n}\sum_{i=1}^{n} x_i$

variance $\sigma_{\vec{X}}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_{\vec{X}})^2$

normalize $\hat{X}_i = \dfrac{x_i - \mu_{\vec{X}}}{\sqrt{\sigma_{\vec{X}}^2 + \epsilon}}$ $\quad \epsilon = 10^{-8}$

shift, scale $y_i = \underset{\text{scale}}{\gamma} \cdot \hat{x}_i + \underset{\text{shift}}{\beta} = \underset{\text{output}}{BN_{\gamma,\beta}(\vec{X})}$



Chain rule

i) $$\frac{dl}{d\hat{x}_i} = \frac{dl}{dy_i}\cdot\frac{dy_i}{d\hat{x}_i} = \frac{dl}{dy_i}\cdot\gamma$$

ii) $$\frac{dl}{d\sigma_{\vec{X}}^2} = \frac{dl}{d\hat{x}_i}\,\frac{d\hat{x}_i}{d\sigma_{\vec{X}}^2} = \frac{dl}{d\hat{x}_i}(x_i - \mu_{\vec{X}})\,\frac{d\,\frac{1}{\sqrt{\sigma_{\vec{X}}^2+\epsilon}}}{d\,\sigma_{\vec{X}}^2}$$

$$= \frac{dl}{d\hat{x}_i}(x_i - \mu_{\vec{X}})\,\frac{-\frac{1}{2}\cdot(\sigma_{\vec{X}}^2+\epsilon)^{-\frac{1}{2}}}{(\sigma_{\vec{X}}^2+\epsilon)}$$

$$= \sum_{i=1}^{m}\frac{dl}{d\hat{x}_i}(x_i - \mu_{\vec{X}})\left(-\frac{1}{2}\right)(\sigma_{\vec{X}}^2+\epsilon)^{-\frac{3}{2}}$$

iii) $$\frac{dl}{d\mu_{\vec{X}}} = \frac{dl}{d\sigma_{\vec{X}}^2}\cdot\frac{d\sigma_{\vec{X}}^2}{d\mu_x} + \frac{dl}{d\hat{x}_i}\cdot\frac{d\hat{x}_i}{d\mu_x}$$

$$= \sum_{i=1}^{m}\frac{dl}{d\sigma_{\vec{X}}^2}\cdot\left(2\cdot\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_{\vec{X}})\right) + \frac{dl}{d\hat{x}_i}\cdot\left(\frac{-1}{\sqrt{\sigma_{\vec{X}}^2+\epsilon}}\right)\sum_{i=1}^{m}$$

iv) $$\frac{dl}{dx_i} = \frac{dl}{d\hat{x}_i}\,\frac{d\hat{x}_i}{dx_i} + \frac{dl}{d\sigma_{\vec{X}}^2}\,\frac{d\sigma_{\vec{X}}^2}{dx_i} + \frac{dl}{d\mu_x}\,\frac{d\mu_x}{dx_i}$$

$$= \frac{dl}{d\hat{x}_i}\left(\frac{1}{\sqrt{\sigma_{\vec{X}}^2+\epsilon}}\right) + \frac{dl}{d\sigma_{\vec{X}}^2}\,\frac{2}{n}(x_i - \mu_{\vec{X}}) + \frac{dl}{d\mu_x}\,\frac{1}{n}$$

v) $$\frac{dl}{d\gamma} = \frac{dl}{dy_i}\,\frac{dy_i}{d\gamma} = \sum_{i=1}^{m}\frac{dl}{dy_i}\cdot\hat{x}_i$$

vi) $$\frac{dl}{d\beta} = \frac{dl}{dy_i}\,\frac{dy_i}{d\beta} = \sum_{i=1}^{m}\frac{dl}{dy_i}$$

○ 더 큰 learning rate 사용 가능.

internal covariate shift 감소, Parameter scale 영향(x)
더 큰 weight가 더 작은 gradient를 유도하기에
Parameter growth가 안정화 되는 효과 (발산(x), missing(x))

○ mini-batch를 어떻게 설정하냐에 따라 data sample 에대한 다른 결과가 나온다

⇒ drop out, 앙상블과 같은 효과, 의존도 떨어짐
⇒ 더 general한 model을 learning 하는 효과가 생김.

Dropout.

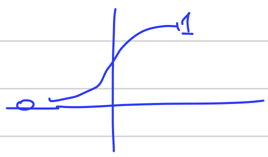
np.random.randn(10, 100) × 0.01

표준정규분포 $N(0, 0.01^2)$ = 표준편차가 0.01인 정규분포

σ * np.random.randn(...) + μ

표정. 정규분포 $N(\mu, \sigma^2)$

$N(0, 1)$을 init으로 시작하는 weight로
5layer sigmoid를 넘길때 활성화 함수의 출력값의 분포가 안좋아짐



$y = \frac{1}{1+e^{-x}}$

$x=0$, $y=\frac{1}{2}$

$x=$

0과 1의 쪽에 치우치게 됨.

⇒ 역전파의 기울기가 작아진다.