Ch6. 학습 관련 기술들

- Outline :
  - 최적화 방법 (가중치 매개변수 탐색)
  - W 초깃값, 하이퍼파라미터 설정방법
  - 오버피팅 대응책 - 가중치 감소, 드롭아웃 등 정규화 방법
  - 배치 정규화

⇒ 신경망 (딥러닝) 학습의 효율과 정확도 ↑

1. 매개변수 갱신

- "최적화 (optimization)" : 손실함수 값을 최대한 낮추는 매개변수를 찾는 문제.

1) 확률적 경사 하강법 (SGD)

학습률 (learning rate)

$$W \leftarrow W - \eta \frac{\partial L}{\partial W}$$

W에 대한 손실함수(L)의 기울기

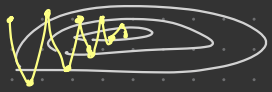
: 기울어진 방향으로 일정거리만 가는 방법.

코드 구현)

```
network = TwoLayerNet(...)
optimizer = SGD()
for i in range(10000):
    ...
    grads = network.gradient(x_batch, t_batch)
    params = network.params
    optimizer.update(params, grads)
```

* 매개변수 갱신 (최적화)을 담당하는 클래스를 분리 ⇒ 모듈화

- 장점 : 단순, 구현 easy.
- 단점



At 비등방성 함수 (방향에 따라 성질, 기울기가 달라지는 함수),

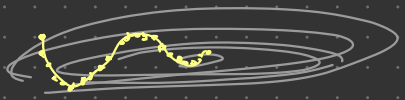
탐색경로 비효율적!

2) 모멘텀 (Momentum)

like 지면마찰, 공기저항

$$\begin{cases} v \ (속도) \leftarrow \alpha v - \eta \frac{\partial L}{\partial W} \\ W \leftarrow W + v \end{cases}$$

⇒ 기울기 방향으로 물체가 힘을 받아 가속된다는 "물리법칙" 적용.



SGD보다 x축 방향으로 더 빠르게 다가감.

3) AdaGrad

- 학습률(η)을 정하는 효과적 기술 ⇒ "학습률 감소"
- AdaGrad : 개별 매개변수에 적응적으로 학습률(η)을 조정하면서 학습

$$\begin{cases} h \leftarrow h + \frac{\partial L}{\partial W} \odot \frac{\partial L}{\partial W} \\ W \leftarrow W - \eta \frac{1}{\sqrt{h}} \frac{\partial L}{\partial W} \end{cases}$$

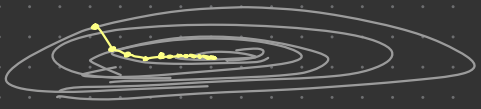
⊙ → 행렬의 원소별 곱심

⇒ 학습률 감소가 매개변수 원소마다 다르게 적용됨.

$\frac{1}{\sqrt{h}}$ 학습률 조정.

- 학습 진행할수록 갱신 강도가 약해지는 문제점

⇒ 개선책 : "RMSProp" (먼 과거 기울기 세세히 잊고, 새로운 기울기 정보 크게 반영하는 지수이동평균(EMA) 방법 사용)



y축 방향으로 갱신속도가 빠르게 약해짐.
(지그재그↓)

4) Adam <= 모멘텀 + AdaGrad

구르는 공과 비슷함 / 적응적 갱신 정도 조정

결론: SGD, 모멘텀, AdaGrad, Adam 중에서 무엇을 사용할 지는

풀어야 할 문제와 하이퍼 파라미터 설정에 따라 정확히 선택한다!

2. 가중치의 초깃값

- 가중치 감소 (Weight decay) 기법 – 가중치 매개변수 값이 작아지도록 학습.
  – 오버피팅을 억제해 범용성능을 높이는 테크닉.

↳ 가중치 초깃값을 아예 0으로 설정한다면?

⇒ bad! 가중치를 균일한 값으로 설정하면

오차역전파법에서 모든 가중치의 값이 똑같이 갱신되기 때문에.

결론) 가중치 초깃값을 무작위로 설정해야 한다.

- 은닉층의 활성화값 분포 (콘드-히스토그램 참고) → 적당히 고루 분포되어야 신경망 학습이 효율적으로 이루어진다.

↳ 활성화함수 (시그모이드, ReLU 등)의 출력 데이터

1) 출력이 0에 가까워지자 미분이 0에 다가감. ⇒ 기울기 소실 (gradient vanishing) 문제.

(데이터가 0과 1에 치우쳐서 분포하는 경우)

2) 활성화 값이 0.5 부근에 치우쳐져 있음. ⇒ 표현력 제한

(다수의 뉴런을 두는 의미가 없음)

• Xavier 초깃값

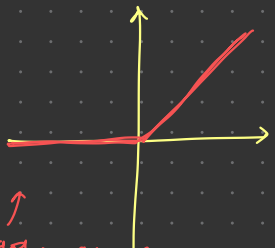
- 사비에르 글로로트와 요슈아 벤지오 의 논문에서 현장하는 가중치 초깃값.
  Xavier Glorot Yoshua Bengio
- 현재 일반적인 딥러닝 프레임워크들이 표준적으로 사용하고 있다.
- 초깃값의 표준편차가 $\frac{1}{\sqrt{n}}$ 이 되도록 설정. (n은 앞층의 노드 수)

  ⇒ 앞 층의 노드가 많을수록 대상 노드의 초깃값으로 설정하는 가중치가 좁게 퍼진다.
- 중앙부분이 선형인 함수 (Sigmoid, tanh 함수)에 적합.

• He 초깃값

- 카이밍 히 의 이름을 딴 초깃값
  Kaming He
- ReLU에 특화된 초깃값.
- 초깃값의 표준편차가 $\sqrt{\frac{2}{n}}$ 인 정규분포 사용

* ReLU



음의 영역이 0이래서
더 넓게 분포시키기 위해
2배의 계수가 필요한 것.

√ 가중치 초깃값은 신경망 학습에 아주 중요한 포인트!

3. 배치 정규화

Batch Nomalization

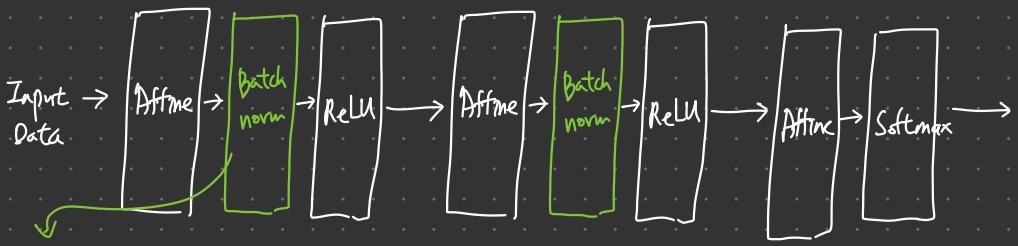
- 가중치 초깃값 말고, 각 층이 활성화를 적당히 퍼뜨리도록 '강제'한다.

  ✓ 학습 속도 개선

  ✓ 초깃값에 의존하지 않음. (초깃값 선택 장애 안녕!)

  ✓ 오버피팅 억제 (드롭아웃 등 필요성 감소)

- 배치 정규화 계층 삽입. (데이터 분포를 정규화)



< 배치 정규화 알고리즘 >

1) 미니배치 $B = \{x_1, x_2, \cdots x_m\}$ 에 대해 평균 μ_B 와 분산 σ_B^2 구한다.
m개의 입력데이터 집합

$$\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

↳ 엡실론 (epsilon) : 아주 작은 값

2) 그리고 평균 =0, 분산 =1 이 되도록 정규화한다.

3) ⇒ 정규화된 데이터 : $\{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_m\}$

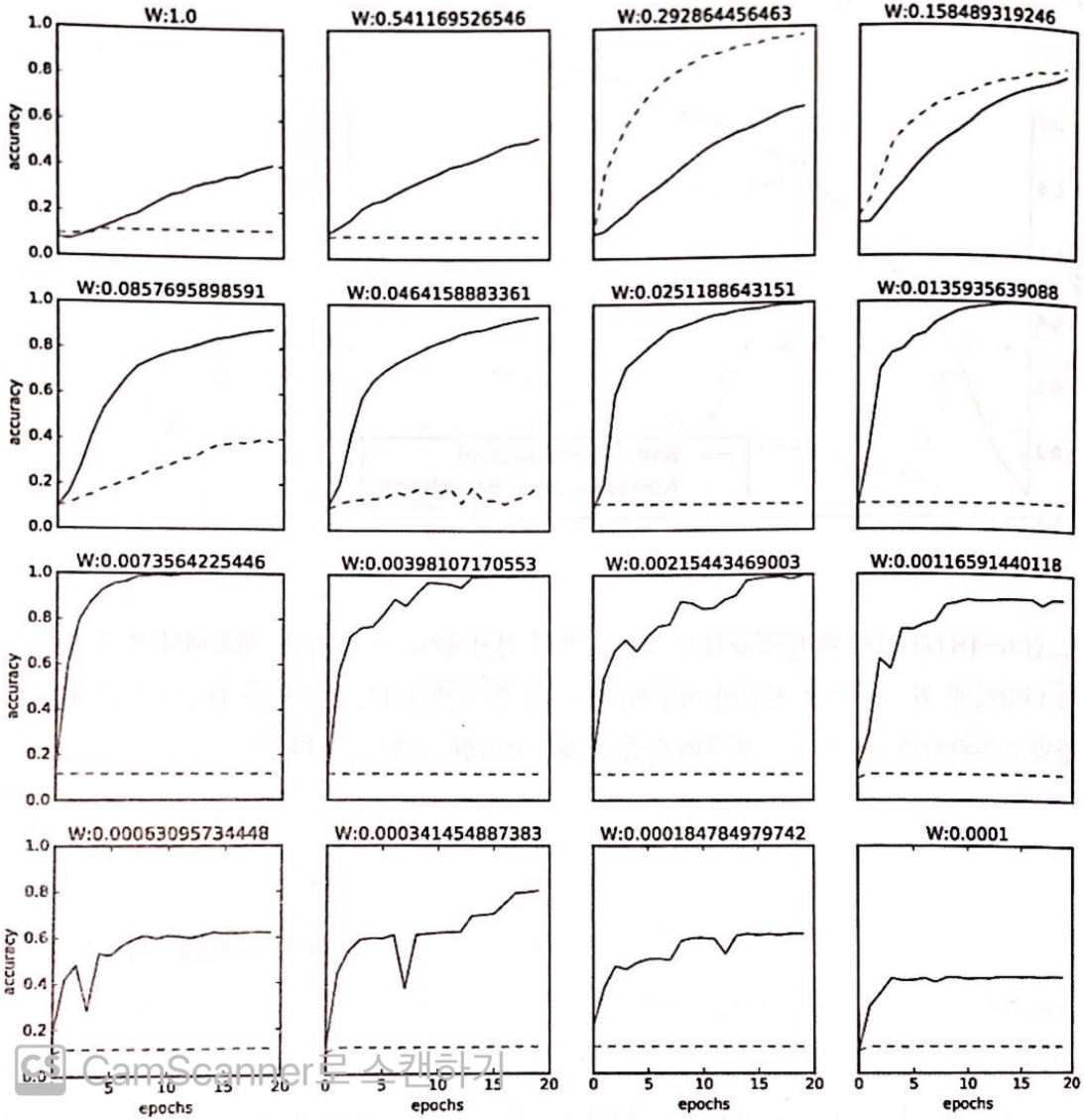
4) 정규화된 데이터에 고유한 확대(Scale)와 이동(Shift) 변환

$$y_i \leftarrow \gamma \hat{x}_i + \beta$$

↳ 확대 ↳ 이동

$\gamma=1$, $\beta=0$ 부터 시작,
학습하면서 적합한 값으로 조정.

그림 6-19 실선이 배치 정규화를 사용한 경우, 점선이 사용하지 않은 경우 : 가중치 초깃값의 표준편차는 각 그래프 위에 표기



학습 속도가 훨씬 빨라지는군!

4. 바른 학습을 위해

4-1) 오버피팅

: 신경망이 훈련 데이터에만 지나치게 적응되어 그 외의 데이터에는 제대로 대응하지 못하는 상태

주로 ∨ 매개변수가 많고 표현력이 높은 모델

∨ 훈련데이터가 적음.

4-2) 가중치 감소

오버피팅은 가중치 매개변수의 값이 커서 발생하는 경우가 많다.

: 학습 과정에서 큰 가중치에 대해서는 그에 상응하는 큰 패널티를 부과하여 오버피팅을 억제하는 방법.

- L2 norm = $\sum_{i=1}^{n} \sqrt{w_i^2}$
- L1 norm = $\sum_{i=1}^{n} |w_i|$
- L∞ norm = $\text{Max}(|w_1|, |w_2|, \cdots, |w_n|)$

가중치 W → 가중치 감소 $\frac{1}{2}\lambda W^2$ 를 손실함수에 더한다

λ : 크게 설정할수록 큰 가중치에 대한 패널티가 커진다.

정규화 세기조절하는 하이퍼 파라미터

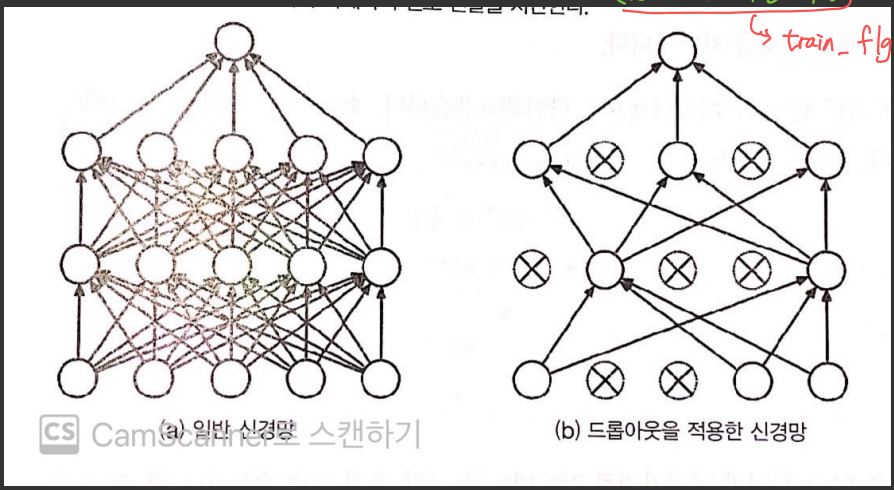
4-3) 드롭 아웃 (Drop out)

: 뉴런을 임의로 삭제하면서 학습하는 기법

훈련 때는 삭제할 뉴런 무작위 선택

시험 때는 모든 뉴런에 신호 전달 (훈련 때 삭제한 비율 곱하여 출력)

↳ train_flg = True 일 때



(a) 일반 신경망 (b) 드롭아웃을 적용한 신경망