딥러닝의 통계적이해

5강. 딥러닝의 제 문제와 발전

1. 딥러닝 학습의 제 문제
2. 경사소실과 활성화 함수의 선택
3. 초깃값의 설정
4. 딥러닝 학습의 최적화
5. 편의와 분산
6. 과대적합의 해소방법
7. 배치정규화
8. 하이퍼파라미터의 최적화

한국방송통신대 이긍희 교수

오늘의 학습목표

1. 활성화 함수의 선택에 대해 이해한다.
2. 신경망 학습에서 초깃값 설정을 이해한다.
3. 다양한 최적화 방법을 이해한다.
4. 과대적합의 개념을 이해한다.
5. 정칙화 방법과 드롭아웃의 개념을 이해한다.
6. 배치 정규화의 과정을 이해한다.

1. 딥러닝 학습의 제 문제

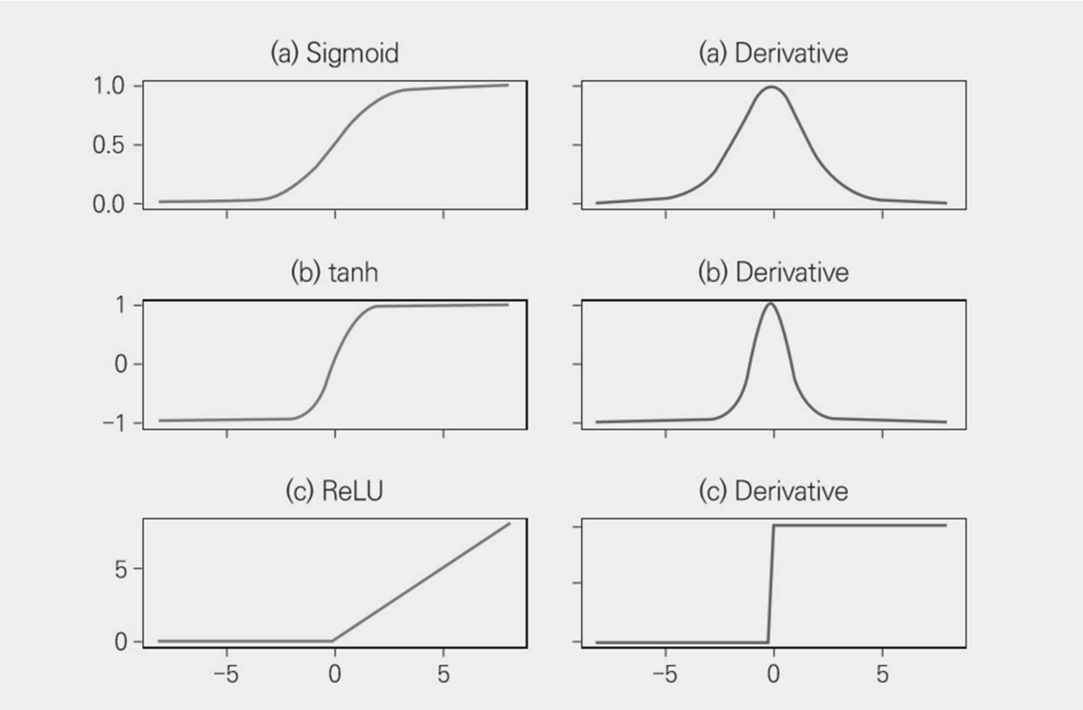
딥러닝 모형 학습시 문제

- 경사소실 : 활성화 함수의 문제로 경사하강법이 제대로 작동되지 않아 가중치가 갱신되지 않는 현상
- 초깃값 설정 : 초깃값을 잘못 설정하는 경우 손실함수가 국지적 최솟값에 머묾
- 과대적합 : 훈련데이터에서는 딥러닝 모형의 적합도가 높지만 새로운 데이터에 대한 예측이 제대로 되지 않음
- 경사하강법에서 학습시간 과다 및 국지적 최적화

2. 경사소실과 활성화 함수의 선택

2. 경사소실과 활성화 함수의 선택

활성화 함수의 미분



경사소실의 원인

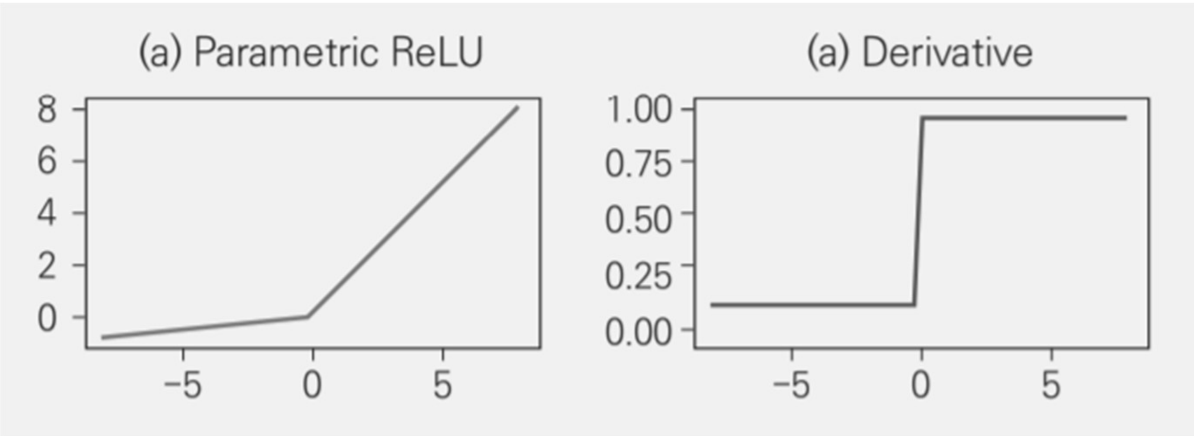
◆ 활성화 함수의 미분과 경사하강법

- 시그모이드 함수 미분 : 최댓값 0.25, 0에서 멀어질수록 작은 값
- tanh 함수 미분 : 최댓값 1, 0에서 멀어질수록 0에 가까움
- ReLU 함수 : 0보다 큰 값에서 미분값이 1 → 경사소실 미발생하여 주로 이용됨

여러 가지 활성화 함수

◆ PReLU(Parametric ReLU) 함수와 ELU 함수

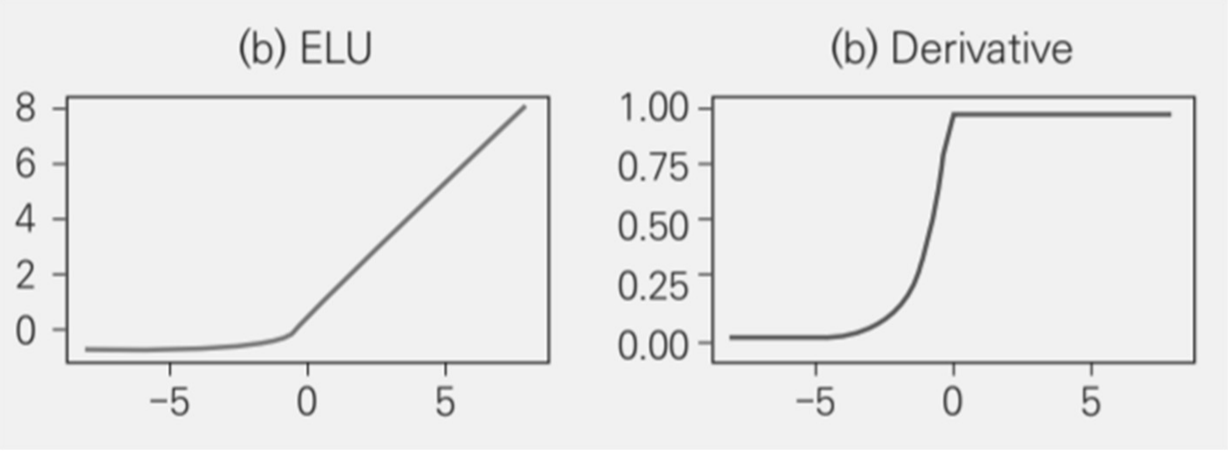
- PReLU 함수 : $a(x) = \begin{cases} x, & x > 0 \\ \alpha_1 x, & x \leq 0 \end{cases}$



여러 가지 활성화 함수

◆ PReLU(Parametric ReLU) 함수와 ELU 함수

- ELU 함수 : $a(x) = \begin{cases} x, & x > 0 \\ \alpha_2(e^x - 1), & x \leq 0 \end{cases}$



붓꽃 데이터의 분류

◆ 은닉층 2개인 신경망

- 시그모이드 함수와 ReLU 함수, 500회 학습

	훈련 데이터		시험 데이터	
	손실	정확도	손실	정확도
시그모이드 함수	0.8528	0.7083	0.8874	0.6000
ReLU 함수	0.2703	0.8917	0.2630	0.9000

3. 초깃값의 설정

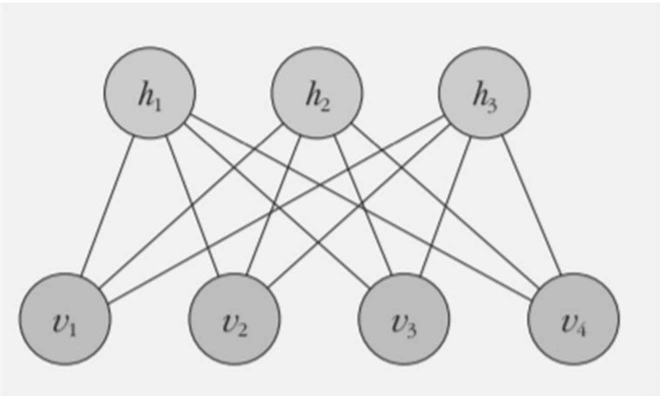
심층신뢰망

- 2006년 힌튼 교수 연구팀이 제시한 모형
- 제한된 볼츠만 머신(RBM)을 적층해서 신경망을 깊게 쌓는 방법
  - RBM은 출력층 없이 입력층(가시층)과 은닉층만 있는 신경망

- 다층 신경망을 딥러닝이라 부름

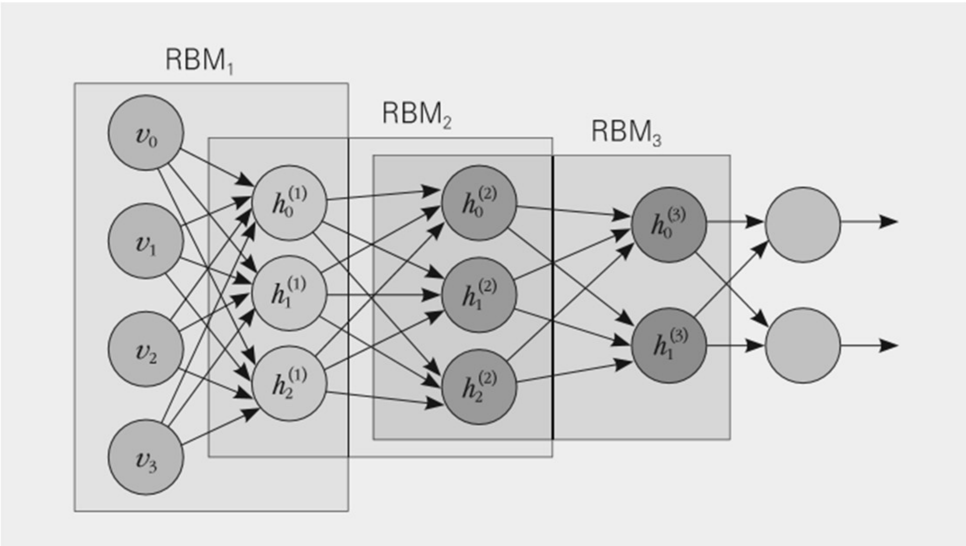
RBM의 학습

- RBM : 레이블 없는 입력층으로부터 패턴을 자동으로 찾는 비지도 학습, 뉴런은 확률 뉴런
  - RBM의 학습과정 : 입력층에서 은닉층으로 학습, 네트워크 구조를 그대로 이용하여 은닉층에서 입력층을 학습
    → 입력층과 유사한 은닉층을 찾음



심층신뢰망

◆ 심층신뢰망은 다층 신경망과 동일한 구조처럼 보이나 실제로는 RBM들을 쌓아가면서 연결한 모형



심층신뢰망

◆ 딥러닝 모형에서 RBM으로 사전학습, 가중치의 초깃값을 정해 주고, 그 다음에 학습 → 딥러닝 모형의 학습이 쉬워짐

- 컴퓨팅 성능 향상, 데이터 증가와 초깃값을 정하는 간단한 새로운 방법의 발견 등으로 심층신뢰망은 이용하지 않음

초깃값의 선택

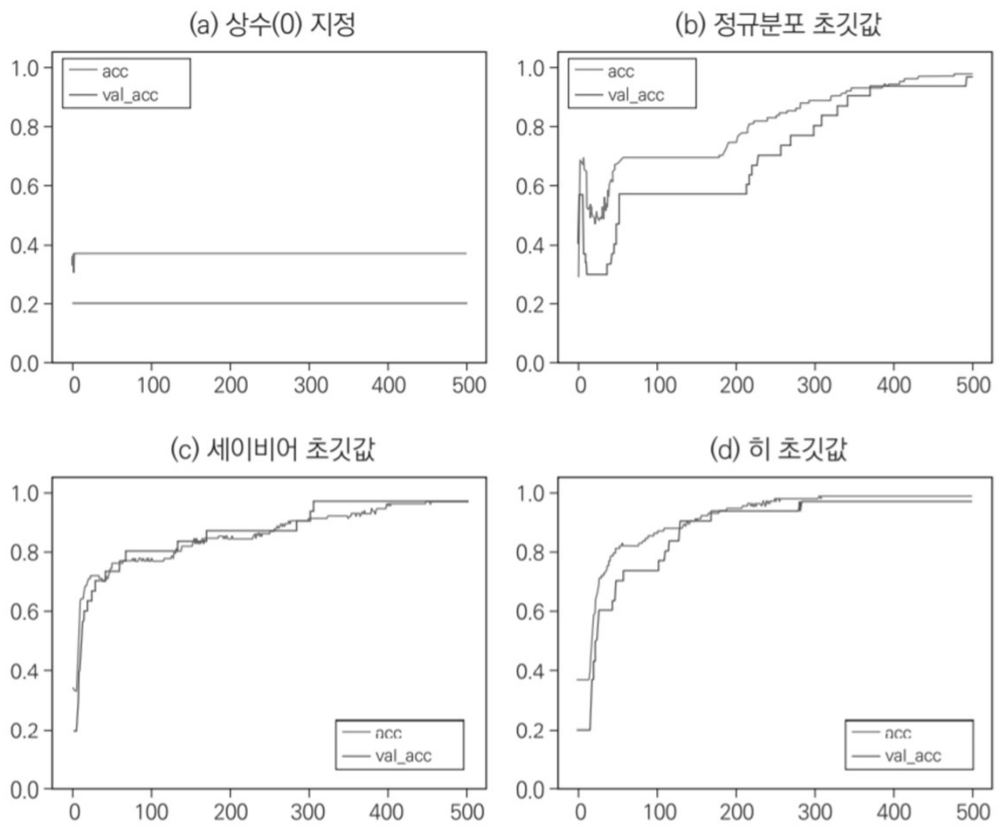
◆ 초깃값 일반적 방법 : 평균 0, 작은 분산 정규분포로부터 난수
가장 쉬운 방법 : 모두 같은 값 0으로 지정
→ 오차역전파법 적용시 가중치가 같은 값으로 갱신

◆ 세이비어(Xavier) 초깃값과 히(He) 초깃값
- 시그모이드 함수 : 세이비어 초깃값 $N\left(0, \frac{2}{n_{input}+n_{output}}\right)$

- ReLU 함수 : 히 초깃값 $N\left(0, \frac{4}{n_{input}+n_{output}}\right)$

3. 초깃값의 설정

초깃값의 비교



4. 딥러닝 학습의 최적화

확률적 경사하강법

- 손실함수의 최솟값으로 갈 때 데이터를 임의로 뽑아서 진행 → 손실함수의 최솟값으로 진동하면서 내려감
  - 확률적 경사하강법 적용시 이전의 미분값(경사)을 기억하지 않고 진행

$$w_{ij}^{(l)} := w_{ij}^{(l)} - \eta \frac{\partial J(w)}{\partial w_{ij}^{(l)}}$$

모멘텀(Momentum) 방법

- 확률적 경사하강법의 손실함수 감소 경로에서 관성을 이용하여 평활하게 움직이도록 하는 방법
  - 이전의 미분값(경사)에 β를 곱해줘서 누적해서 가중치 갱신
  - β값이 0.9라면 $\frac{\eta}{1-0.9} = 10\eta$의 속도로 최적점에 접근

$$m_{t+1} = \beta m_t + \eta \frac{\partial J(w)}{\partial w_{ij}^{(l)}}$$

$$w_{ij}^{(l)} := w_{ij}^{(l)} - m_{t+1}$$

AdaGrad 방법

- 최저점에 가까워질수록 학습률이 감소하는 방법

  - 가중치 갱신이 천천히 이루어져서 최저점에 도달하기도 전에 학습이 끝나는 문제

$$s_0 = 0$$

$$s_{t+1} = s_t + \left(\frac{\partial J(w)}{\partial w_{ij}^{(l)}}\right)^2$$

$$w_{ij}^{(l)} := w_{ij}^{(l)} - \eta \frac{\partial J(w)}{\partial w_{ij}^{(l)}} \Big/ \sqrt{s_{t+1} + \varepsilon}$$

  - ε : 매우 작은 값 (ex. $\varepsilon = 10^{-10}$)

RMSProp 방법

- AdaGrad 방법에 지수평활법을 적용하여 성능 개선, $\alpha = 0.9$

$$s_0 = 0$$

$$s_{t+1} = \alpha s_t + (1-\alpha)\left(\frac{\partial J(w)}{\partial w_{ij}^{(l)}}\right)^2$$

$$w_{ij}^{(l)} := w_{ij}^{(l)} - \eta \frac{\partial J(w)}{\partial w_{ij}^{(l)}} \Big/ \sqrt{s_{t+1} + \varepsilon}$$

Adam 방법

- 모멘텀 방법에서 미분값(경사)의 지수평활값과 RMSProp방법에서의 미분값(경사) 제곱의 지수평활값을 이용 →가중치 갱신
  - $\beta_1 = 0.9,\ \beta_2 = 0.999,\ \ \varepsilon = 10^{-8}$
  - Adam방법이 딥러닝 모형의 학습에서 자주 이용됨

Adam 방법

$$\mathrm{m}_0 = 0,\ s_0 = 0$$

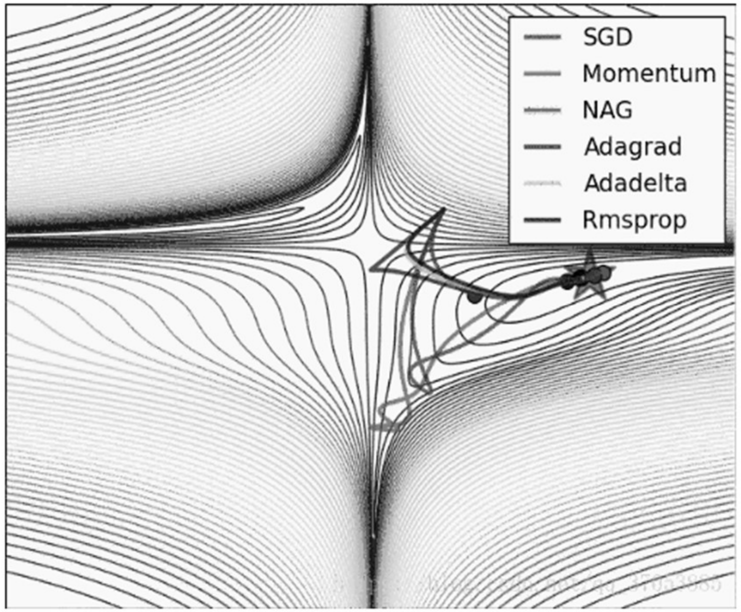
$$\mathrm{m}_{\mathrm{t+1}} = \beta_1 m_t + (1-\beta_1)\frac{\partial J(w)}{\partial w_{ij}^{(l)}}$$

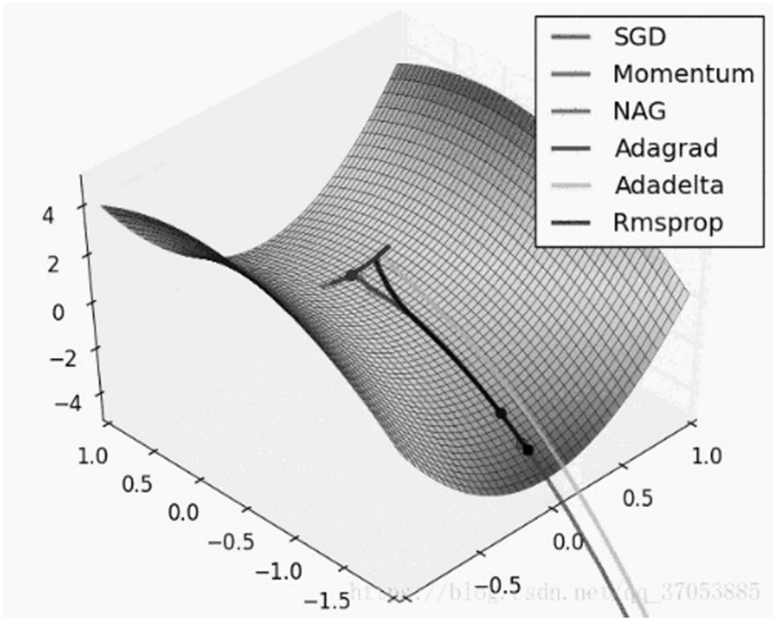
$$s_{t+1} = \beta_2 s_t + (1-\beta_2)\left(\frac{\partial J(w)}{\partial w_{ij}^{(l)}}\right)^2$$

$$w_{ij}^{(l)} \coloneqq w_{ij}^{(l)} - \eta m_{t+1} / \sqrt{s_{t+1} + \varepsilon}$$

4. 딥러닝 학습의 최적화

최적화 방법의 비교





https://ruder.io/optimizing-gradient-descent/

학습률

◆ 최적화 방법과 별도로 학습률을 학습에 따라 바꾸어 진행

- 학습률 $\eta = \eta_0 10^{-\frac{t}{n}}$로 지정

→ 학습이 진행됨에 따라 학습률을 조금씩 줄여주는 것

5. 편의와 분산

모형과 데이터

◆ 모형으로부터 얻은 데이터 $(x_i, y_i), i = 1,2, \dots, n$

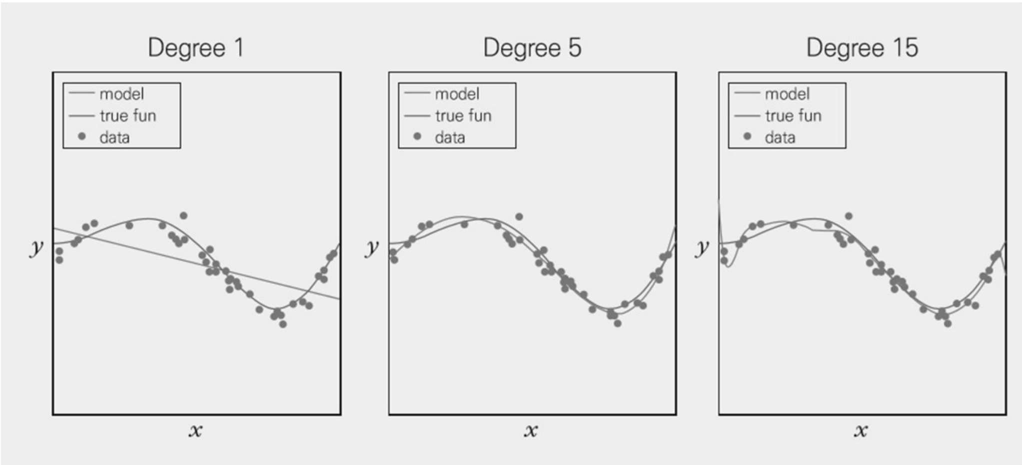
$$y_i = f(x_i) + \varepsilon_i, \qquad f(x) = x\, sin(2\pi x)$$

· 참 모형 $f(x)$, ε_i는 오차항

- 다항모형으로 추정 : $y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_i^j + \varepsilon_i$

과소적합과 과대적합

◆ 산점도의 데이터를 p를 1, 5, 15로 달리하여 다항 회귀모형으로 추정



과소적합과 과대적합

- 훈련데이터와 검증데이터의 분류 오류율을 비교하여 모형의 과대적합 여부 파악

	훈련 데이터	검증 데이터	판단
분류 오류율	낮음	보통	과대적합(고분산)
	보통	보통	과소적합(고편의)
	보통	높음	고편의, 고분산
	낮음	낮음	저편의, 저분산

딥러닝 모형의 개선

- 훈련데이터에서 낮은 성능 : 편의(과소적합)

  → 보다 깊거나 넓은 신경망으로 모형을 더 복잡하게 작성
- 훈련데이터 좋은 성과, 시험데이터에서 낮은 성과 : 편의가 낮지만 분산이 큼(과대적합)

  → 새로운 데이터 추가, 모형을 단순화하여 다시 학습

6. 딥러닝 모형의 정칙화

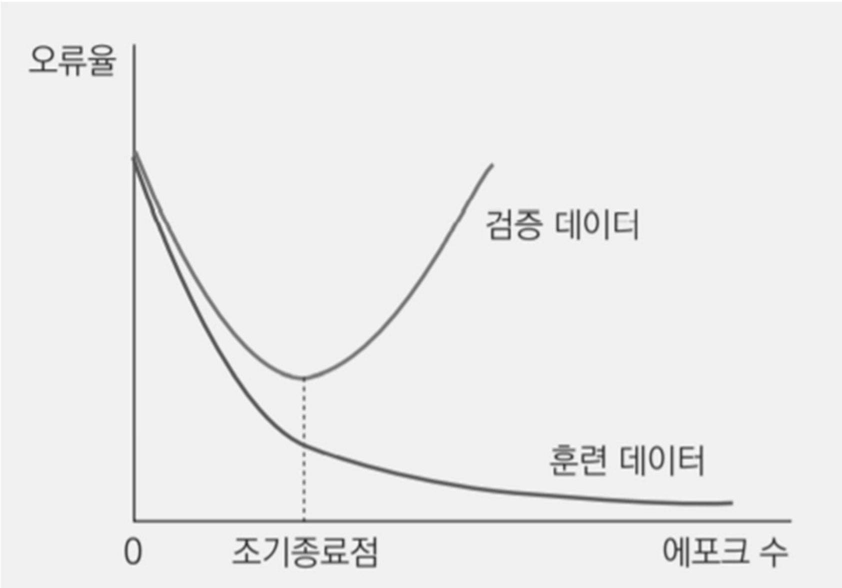
모형의 선택

- 머신러닝, 딥러닝 모형의 성과가 비슷 → 간단한 모형 선택
  - 간단한 모형이 계산량도 적고, 분산도 작기 때문
- 딥러닝 모형에서 성과가 나쁜 경우 과대적합(고분산) 상황
  - 훈련데이터를 추가하거나 모형을 단순화

6. 딥러닝 모형의 정칙화

조기 학습종료

◆ 조기학습 종료 : 검증데이터의 손실함수가 감소하다가 증가하는 부분에서 학습을 중단하고 그 가중치를 이용

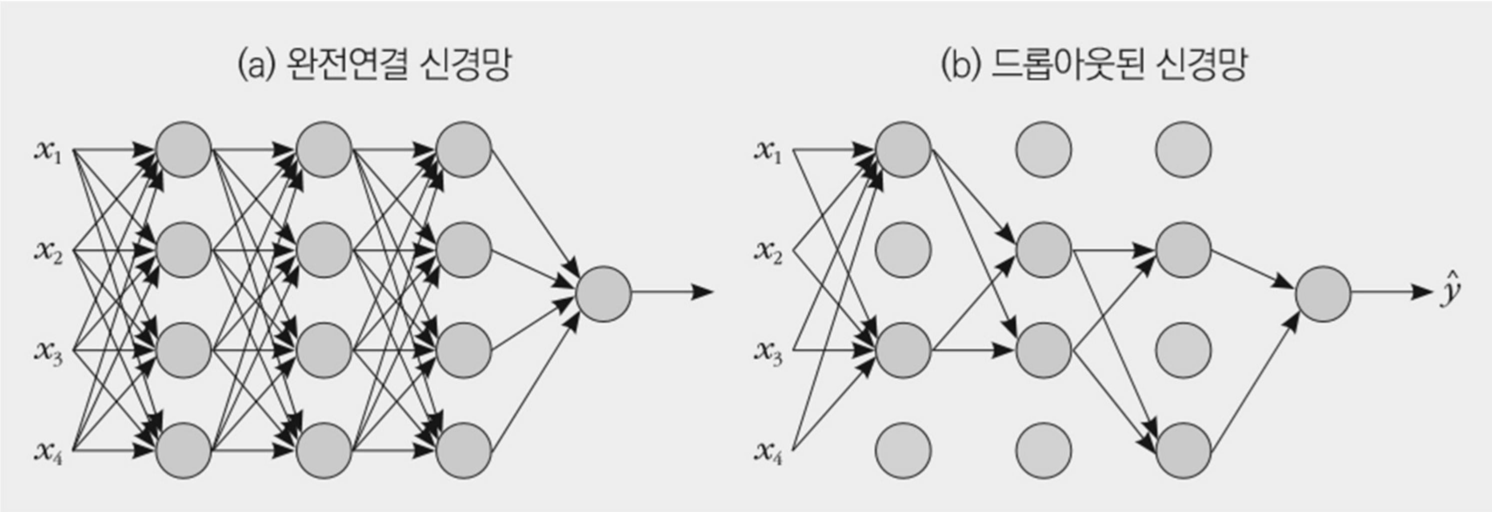


정칙화

- 정칙화(regularization) : 적합부분으로 이루어진 기존 손실함수에 가중치 벌칙항을 더해 손실함수 정의, 가중치 구함
  - L_2 정칙화 : 적절한 수준에서 적합되는 가중치값 선택
  - L_1 정칙화 : 많은 가중치들이 0이 됨

드롭아웃

- 드롭아웃(Dropout) : 입력층부터 은닉층 사이의 뉴런 간 네트워크를 임의로 없애고 학습하는 것



드롭아웃

- 층마다 50%, 80%의 뉴런을 삭제하고 미니배치 단위로 학습
  - 연결이 많을 땐 드롭아웃 비율을 늘리고, 층에 뉴런이 몇 개 없다면 드롭아웃을 시키지 않을 수도 있음
  - 드롭아웃의 앙상블 효과 → 딥러닝 모형의 성능 향상
  - 시험데이터 적용 : 드롭아웃을 쓰지 않고 모든 가중치를 이용

데이터 증식

- 데이터 증식(data augmentation) : 데이터의 수 늘리기
  - 딥러닝 모형은 추정해야 할 네트워크 가중치가 많아서 매우 많은 데이터가 필요, 실제로는 부족
  - 이미지 한 장을 바탕으로 수백, 수천개를 생성 :
    - 수평이동, 대칭이동, 정규분포, 이항분포를 따르는 노이즈를 추가, 이미지의 일부 확대 등

6. 딥러닝 모형의 정칙화

데이터 증식

7. 배치 정규화

표준화

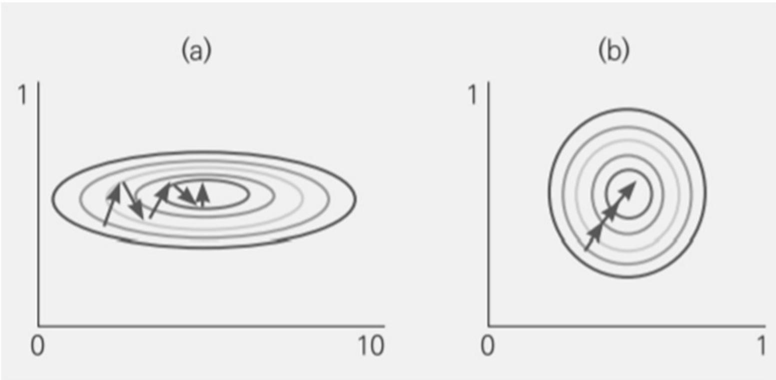
- 통계학의 표준화 : 데이터에서 단위 또는 범위 영향력을 제거

$$\frac{x_i - \min x_i}{\max x_i - \min x_i}, \qquad \frac{x_i - \bar{x}}{\hat{\sigma}}$$

7. 배치 정규화

표준화

- 손실함수의 모양이 타원형과 같이 한 축으로 길게 나타나 최적점을 찾기 어려움
  - 표준화를 하면 손실함수가 원형에 가깝게 되어 최적점을 빠르게 찾을 수 있음



경사하강법과 활성화 함수

- 경사하강법에서 활성화 함수에 의존하지 않는 방법이 필요
  - 입력층, 은닉층의 데이터 값을 표준화 → 활성화 함수 적용 전 값들이 0 근처의 값으로 분포
    → 딥러닝 학습에서 활성화 함수의 영향력을 줄임

배치 정규화

◆ 아래 과정을 미니 배치 단위로 반복

① 표준화 : 분산은 1, 평균은 0인 $z_{norm}^{(l)}$을 구함

- γ와 β를 더해서 새로운 입력값 $\tilde{z}^{(l)}$을 만듬

$$\hat{\mu}^{(l)} = \frac{1}{m}\sum_i z_i^{(l)}, \quad \hat{\sigma}^{2(l)} = \frac{1}{m}\sum_i \left(z_i^{(l)} - \hat{\mu}^{(l)}\right)^2$$

$$z_{norm}^{(l)} = \frac{z^{(l)} - \mu^{(l)}}{\sqrt{\hat{\sigma}^{2(l)} + \varepsilon}}, \quad \tilde{z}^{(l)} = \gamma z_{norm}^{(l)} + \beta$$

배치 정규화

② $\tilde{z}^{(l)}$에 활성화 함수를 적용

- γ와 β 등을 초깃값 1과 0으로 두고 미니배치에 대해 경사하강법을 적용

$$x \xrightarrow{\mathrm{w}^{(1)}} z^{(1)} \xrightarrow{\beta,\gamma} \tilde{z}^{(1)} \rightarrow h^{(1)} = a\left(\tilde{z}^{(1)}\right) \xrightarrow{\mathrm{w}^{(2)}} z^{(2)} \cdots$$

배치 정규화

- 배치 정규화는 양쪽 극단 값이 덜 발생 → 학습이 잘 이루어지도록 하고 경사소실 문제 해소
  - 은닉층 입력값들이 제대로 분포 → 학습에서 초깃값의 의존성이 줄어들고 과대적합을 억제 (드롭아웃과 정칙화 불필요)
  - γ와 β같이 추가적 추정 모수 증가 → 모형이 더 복잡, 추가적 학습시간 소요

8. 하이퍼파라미터의 최적화

하이퍼파라미터

- 신경망 학습 전에 미리 값을 정하는 모수
  - 경사하강법의 학습률 η와 최적화방법, 미니배치의 크기, 딥러닝 모형의 설계 등

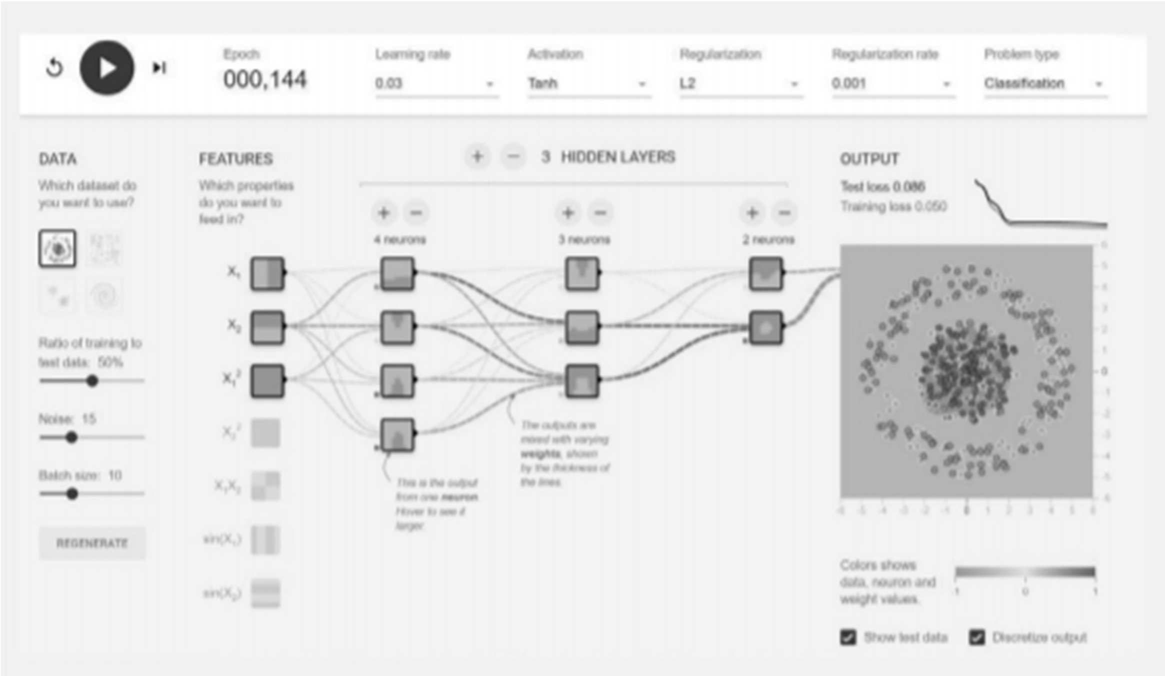
하이퍼파라미터의 탐색

- 하이퍼파라미터를 임의로 탐색한 후 좋은 결과를 보이는 범위를 정하고 그 범위 내에서 더 세밀하게 탐색
  - 탐색이 환경상 불가능하다면 좋은 성과를 보였다고 알려진 논문의 하이퍼파라미터를 이용

8. 하이퍼파라미터의 최적화

구글 텐서플로우 플레이그라운드

◆ 하이퍼파라미터의 효과를 파악하기 위한 도구



AutoML 프로그램

- 하이퍼파라미터를 자동으로 정하는 AUTOML 프로그램
  - 좋은 하이퍼파라미터와 신경망 구조를 검색·추천해서 딥러닝 전문가가 아니더라도 딥러닝 모형을 만들 수 있도록 함

학습정리

- ✓ 오차역전파법에서 경사소실 문제를 해결하기 위해 신경망에서 시그모이드 함수 대신 ReLU 활성화 함수를 이용하고 있다.
- ✓ 예비 학습을 위해 RBM을 쌓아 올린 심층신뢰망을 기반으로 딥러닝 모형이 발전하였다.

학습정리

- ✓ 가중치 초깃값으로 시그모이드 활성화 함수를 이용하는 경우 세이비어(Xavier) 초깃값, ReLU 함수를 이용한 경우엔 히(He) 초깃값이 이용된다.
- ✓ 확률적 경사하강법을 개선한 학습방법으로는 모멘텀, AdaGrad, RMSProp, Adam 등이 있다.

학습정리

- ✓ 딥러닝 모형의 과대적합 해소 방법으로는 조기학습종료, 정칙화, 드롭아웃과 데이터 증식 등이 있다.
- ✓ 배치 정규화를 진행하면 학습속도를 높이고, 초깃값 의존성을 줄일 수 있다.

딥러닝의 통계적이해

다음시간안내

6강.
합성곱신경망의 기초(1)