BE530 – Medical Deep Learning

– CNN Optimization –

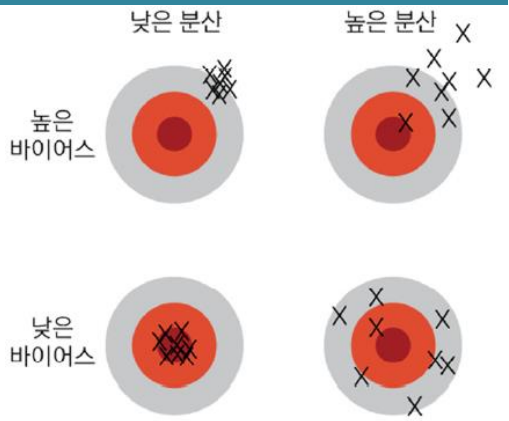
Byoung-Dai Lee

Division of AI Computer Science and Engineering

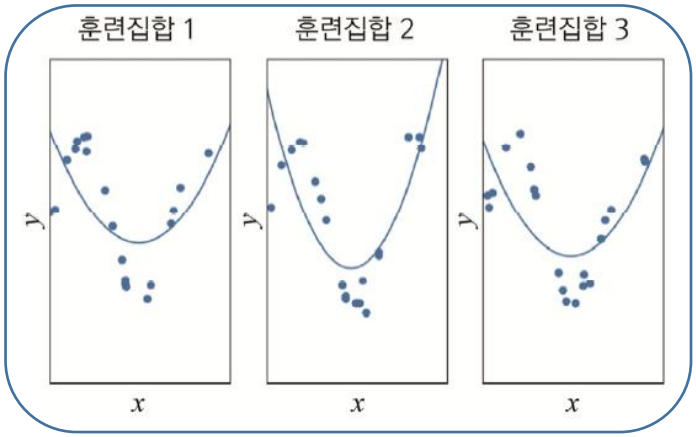
Kyonggi University

KYONGGI UNIVERSITY

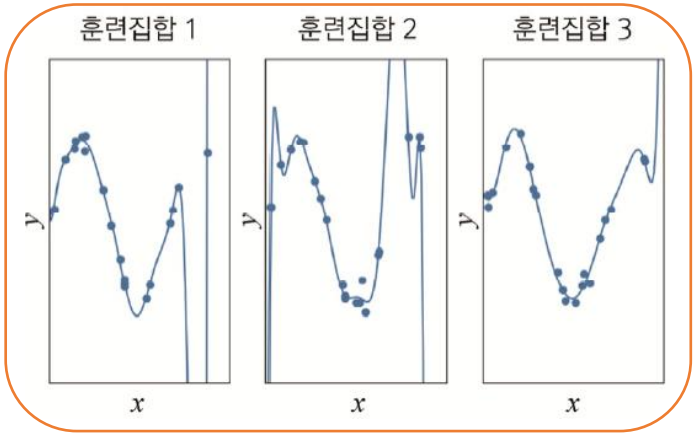
분산과 바이어스



- **분산**
  - 학습 결과로 얻은 모델들이 모두 유사 → 분산이 작음
- **바이어스**
  - 훈련집합에 높은 성능을 보이는 모델 → 바이어스가 큼



2차 모델은 바이어스는 크고 분산은 작음

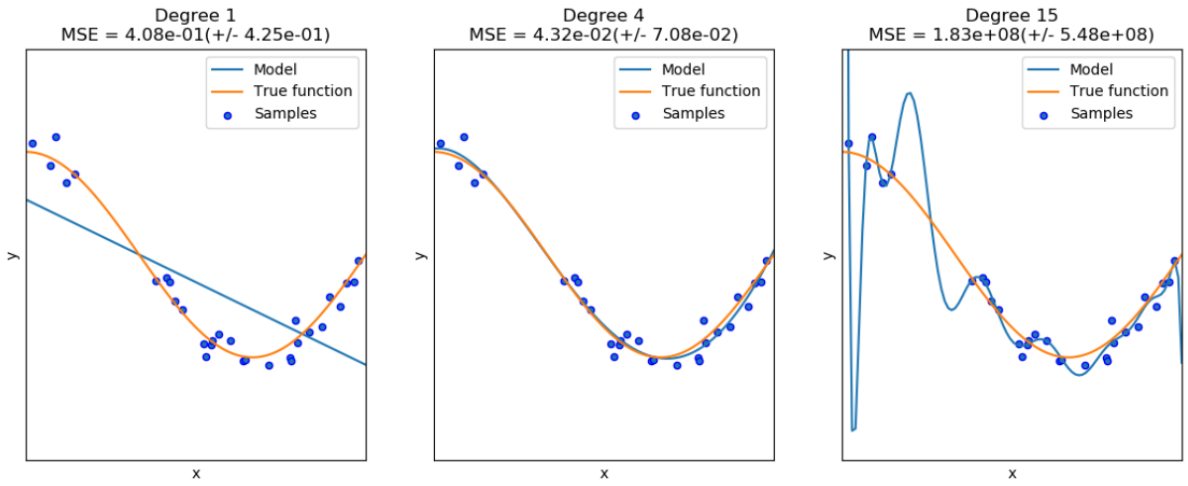


12차 모델은 바이어스는 작고 분산은 큼

- **분산과 바이어스는 Trade-off 관계**
  - 바이어스의 희생을 최소로 유지하면서 분산을 최대로 낮추는 전략

과소적합 vs. 과잉적합

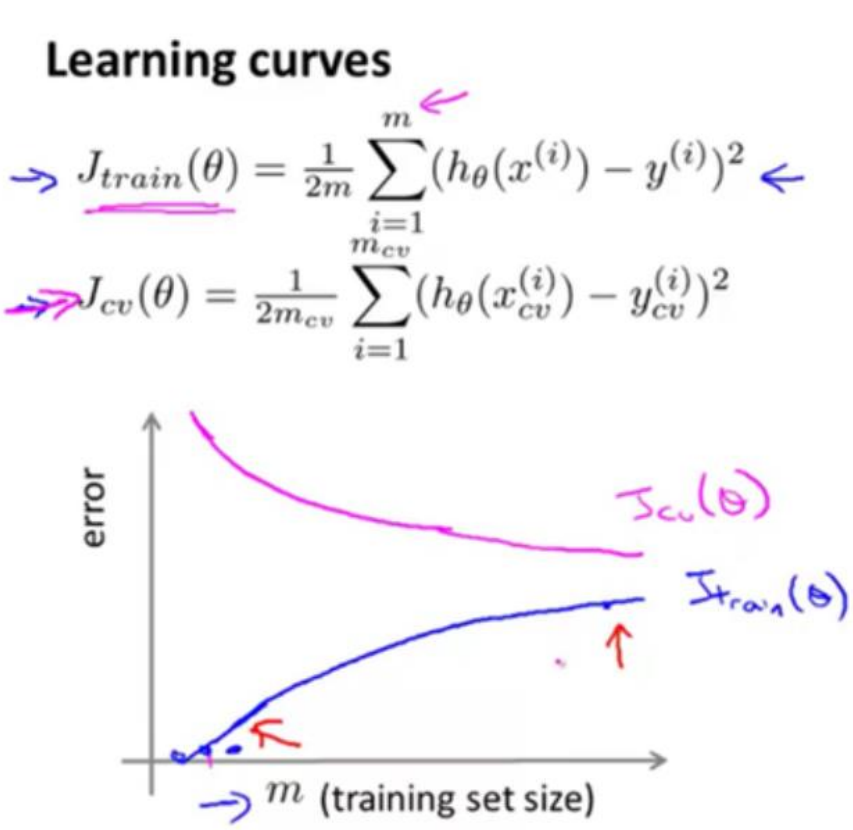
- 과소적합 (underfitting)
  - 모델의 용량이 작아 훈련집합의 샘플조차 제대로 모델링하지 못함
  - High Bias
- 과잉적합 (overfitting)
  - 훈련집합에 지나치게 적합되어 그 외의 데이터는 표현하지 못함
    - 모델의 용량이 지나치게 크기 때문에 잡음까지 수용
  - High Variance

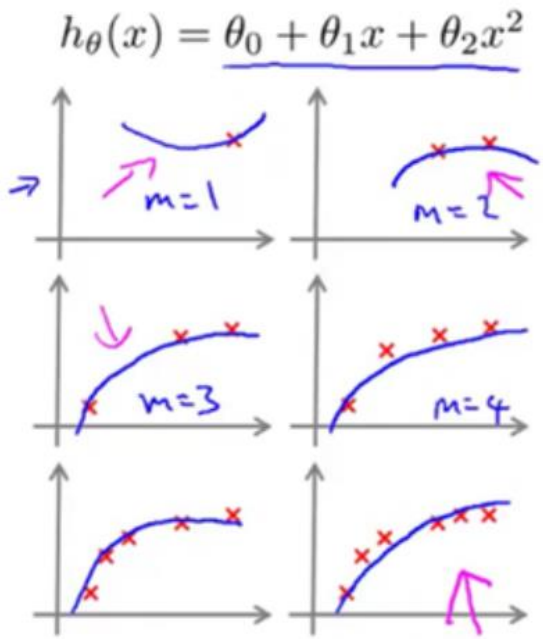


underfitting fit overfitting

학습 곡선의 이해 (1)

- 일반적인 경우

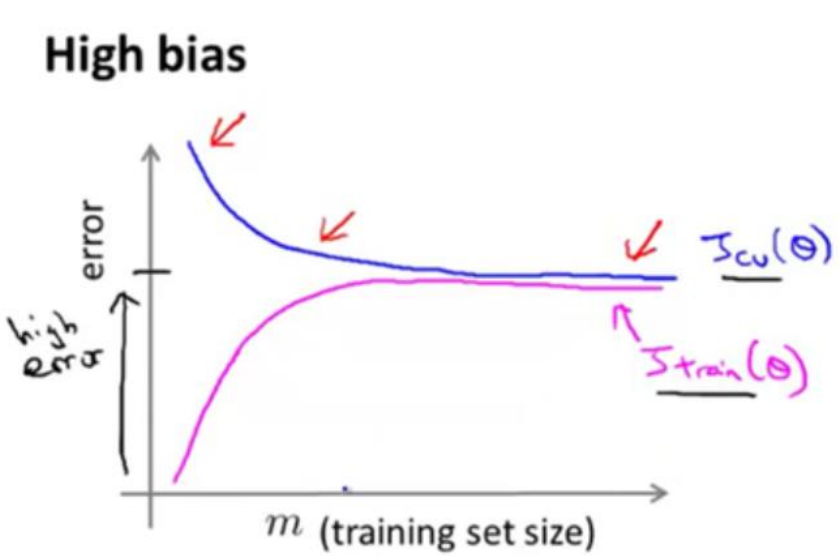




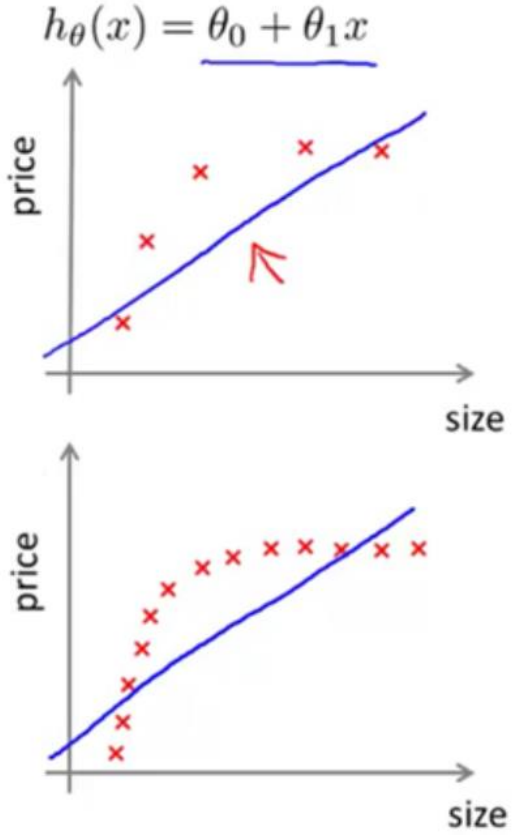
* Source: Stanford Univ. Machine Learning – Andrew Ng

학습 곡선의 이해 (2)

- 과소적합
  - When training and testing errors converge and are high



If a learning algorithm is suffering from high bias, getting more training data will not (by itself) help much.

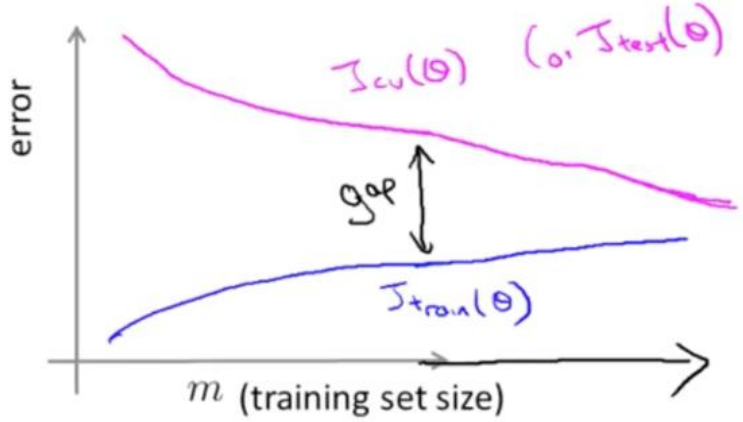


* Source: Stanford Univ. Machine Learning – Andrew Ng

학습 곡선의 이해 (3)

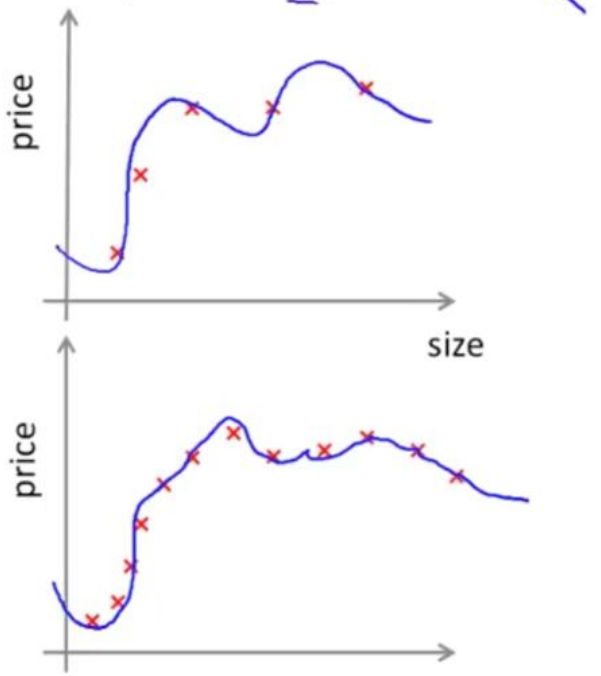
- 과잉적합
  - When there is a large gap between the errors

High variance



If a learning algorithm is suffering from high variance, getting more training data is likely to help.

$h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{100} x^{100}$
(and small λ)



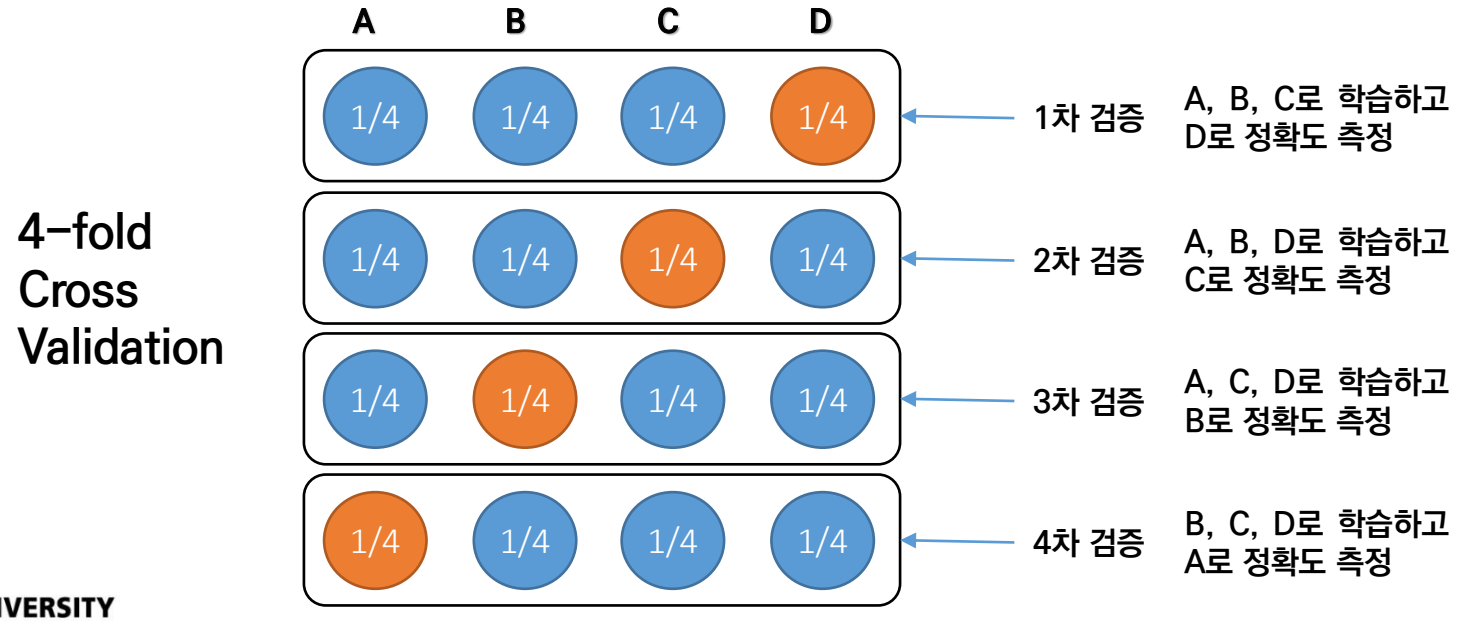
* Source: Stanford Univ. Machine Learning – Andrew Ng

Some Tips ...

- Loss가 일정 이상 줄어들지 않는다
  - CNN 구조를 더 복잡하게, filter를 더 많이 써본다
- Loss가 줄어드는데 성능은 좋아지지 않는다.
  - Overfitting의 가능성이 높으므로 CNN구조를 간단하게, filter 개수를 줄여 본다
- 초반에 loss가 줄어드는데 오래 걸린다
  - Initialization에 문제가 있다

K-fold Cross Validation

- 적은 학습데이터로 인해 검증 성능 신뢰도 저하 문제 해결
- 알고리즘
  1. 전체 데이터를 K개의 부분 집합으로 분할
  2. K-1개를 학습데이터로 정하고 나머지 1개를 테스트 데이터로 사용
  3. 학습데이터와 테스트 데이터를 바꿔가며 교차검증을 K번 반복
  4. K개의 정확도의 평균을 계산하여 최종 정확도로 설정

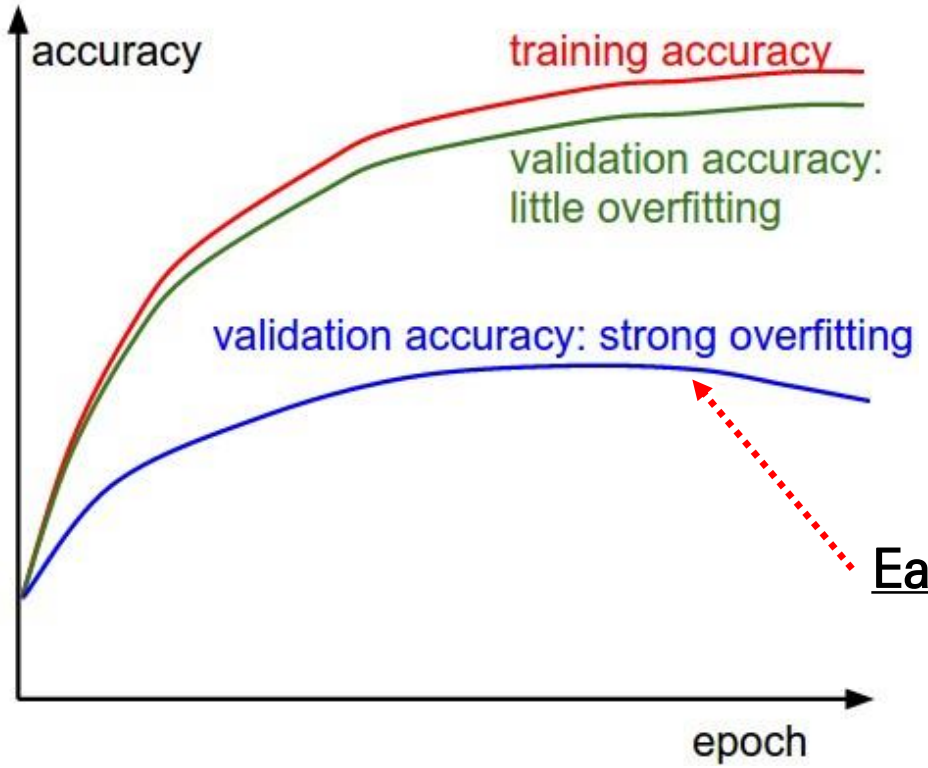


규제 (Regularization)

- 규제의 정의
  - …any modification we make to a learning algorithm that is intended to reduce its generalization error …
    - 일반화 오류를 줄이려는 의도를 가지고 학습 알고리즘을 수정하는 방법 모두
  - 모델 용량에 비해 데이터가 부족한 경우의 불량 문제(ill-posed problem)을 푸는데 사용해오 전통적인 수학 및 통계학 연구 주제
- 현대 기계학습의 전략
  - 충분히 큰 용량의 모델을 설계한 후 학습 과정에서 여러 규제 기법 적용
- 대표적 규제 기법
  - 조기 멈춤 (early stopping)
  - 가중치 벌칙 (weight decay)
  - Drop-out
  - 데이터 확대 (Data augmentation)
  - 앙상블 기법
  - …

조기 멈춤

- 과잉적합이 되는 시점에서 학습을 중단



Early Stopping 시점

가중치 벌칙 (1)

- 손실 함수에 규제항을 포함하여 모델의 용량을 제한

$$L(w) = \frac{1}{N}\sum_{i}^{N} l(w, x_i, t_i) + \lambda R(w)$$

In common use:

L2 regularization	$R(W) = \sum_k \sum_l W_{k,l}^2$
L1 regularization	$R(W) = \sum_k \sum_l \lvert W_{k,l} \rvert$
Elastic net (L1 + L2)	$R(W) = \sum_k \sum_l \beta W_{k,l}^2 + \lvert W_{k,l} \rvert$

- λ는 일반적으로 0.01 ~ 0.00001정도 범위에서 선택

가중치 벌칙 (2)

- L2 Regularization
  - 학습의 방향이 단순히 loss function(L)의 값이 작아지는 방향으로만 진행되는 것이 아니라, **가중치(w)의 값도 최소가 되는 방향으로 진행**

$$L = L_0 + \frac{\lambda}{2n}\sum_w w^2$$

$$w \leftarrow w - r\left(\frac{\partial L}{\partial w}\right) = w - r\left(\frac{\partial L_0}{\partial w} + \frac{\lambda}{n}w\right) = \left(1 - \frac{r\lambda}{n}\right)w - r\frac{\partial L_0}{\partial w}$$

원래의 w값에 $(1 - \frac{r\lambda}{n})$항목을 곱하기 때문에 값이 작아지는 방향으로 진행

  - w값이 작아지도록 학습한다는 것의 의미는?
    - local noise가 학습에 큰 영향을 끼치지 않는다는 것을 의미 ➔ outlier의 영향을 최소화

가중치 벌칙 (3)

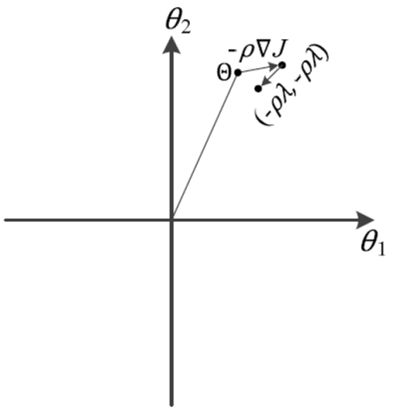
- L1 Regularization

가중치 값 자체를 줄이는 것이 아니라 w의 부호(sgn(w))에 따라 상수값을 빼주는 방식

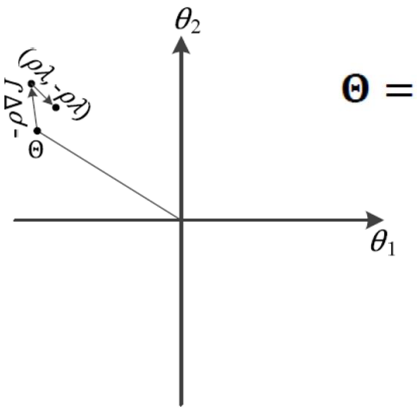
$$L = L_0 + \frac{\lambda}{n}\sum_w |w|$$

$$w \leftarrow w - r\left(\frac{\partial L}{\partial w}\right) = w - r\left(\frac{\partial L_0}{\partial w} + \frac{\lambda}{n} sgn(w)\right)$$

- 상수값을 빼주기 때문에 작은 가중치들은 거의 0으로 수렴이 되어 몇 개의 중요한 가중치만 남게 됨



(a) $\mathbf{sign}(\Theta) = (1,1)^T$인 경우

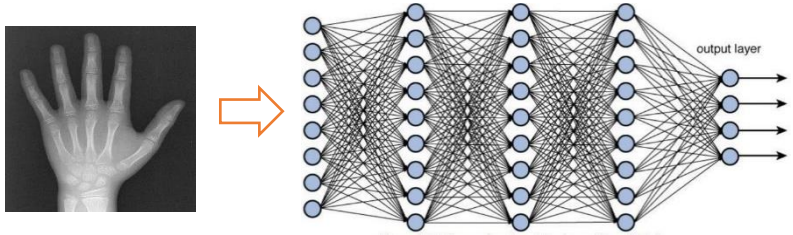


(b) $\mathbf{sign}(\Theta) = (-1,1)^T$인 경우

$$\mathbf{\Theta} = \mathbf{\Theta} - \rho\nabla J - \rho\lambda \mathbf{sign}(\mathbf{\Theta})$$

가중치 벌칙 (4)

- 골연령 자동 판독
  - formulated as a classification problem



[0, 0, 1, 0, 0, ..., 0]
/* Bone Age: 2 yrs */

  - Loss function with L2 regularization
    - 환아의 실제 골연령과 예측된 골연령간의 차이가 클수록 penalty를 부과함으로써 오차 크기가 최소가 되도록 학습

Cross Entropy

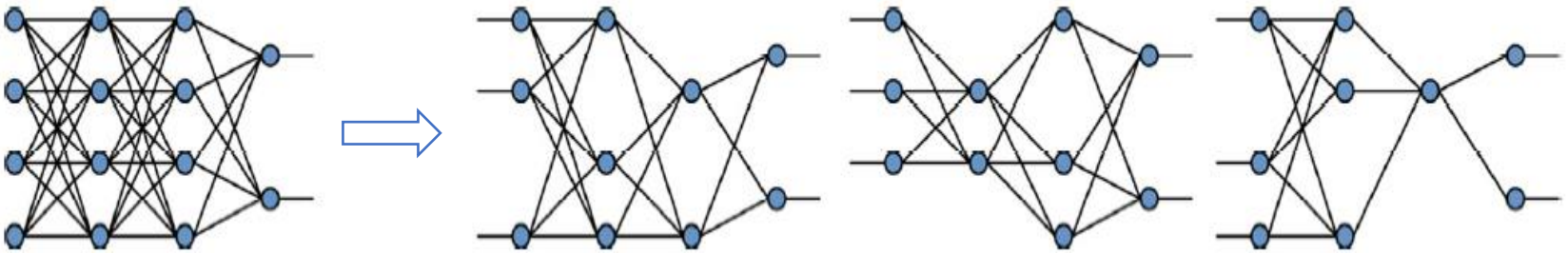
$$p_{ij} = \frac{e^{x_{ij}}}{\sum_k e^{x_{ik}}}$$

$$loss = \frac{1}{M}\sum_{i=1}^{M} -log(p_{y_i})$$

$$loss = \frac{1}{M}(\sum_{i=1}^{M} -log(p_{y_i}) + \sum_{j=1}^{N} p_{i,j} * (j - y_i)^2)$$

Drop-out (1)

- 학습 단계
  - 입력층과 은닉층의 노드 중 미리 정해 둔 비율 p만큼을 선택하고 선택되지 않은 노드는 무효화

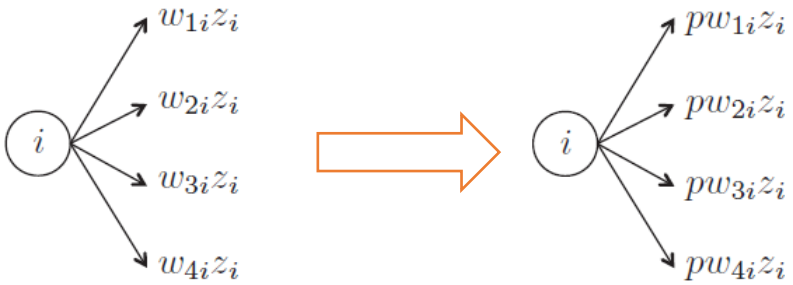


원래 신경망 Drop-out된 신경망 예시

  - 가상의 신경망을 구성하는 노드들은 가중치를 업데이트할 때마다 다시 무작위로 선택
    - 미니배치를 적용하고 있다면 미니배치 단위로 노드들을 다시 선택

Drop-out (2)

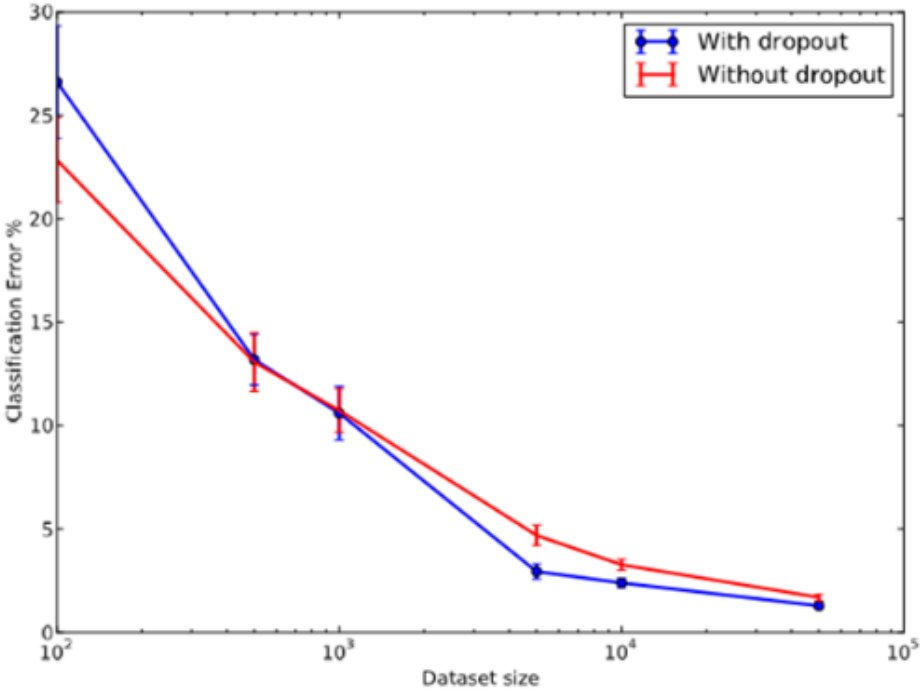
- 추론 단계
  - 학습이 완료된 후 추론 시에는 모든 노드들을 사용
  - Drop-out의 대상이 되었던 층의 노드들 모든 출력의 가중치를 *p*배로 한다
    - 해당 층에서 추론 시 노드의 수가 학습 시에 비해 1/p배 된 것과 같기 때문에 이를 보상하기 위함



- Drop-out을 거친 신경망을 여러 개 훈련한 후 이들 신경망으로부터 얻어진 결과값의 평균을 내는 것과 같은 효과
  - 최근에는 모든 Layer에서 dropout를 사용함
    - 이 때, p=0.5가 가장 많이 사용하며 괜찮은 기본값이며, cross validation을 통해 조정 가능

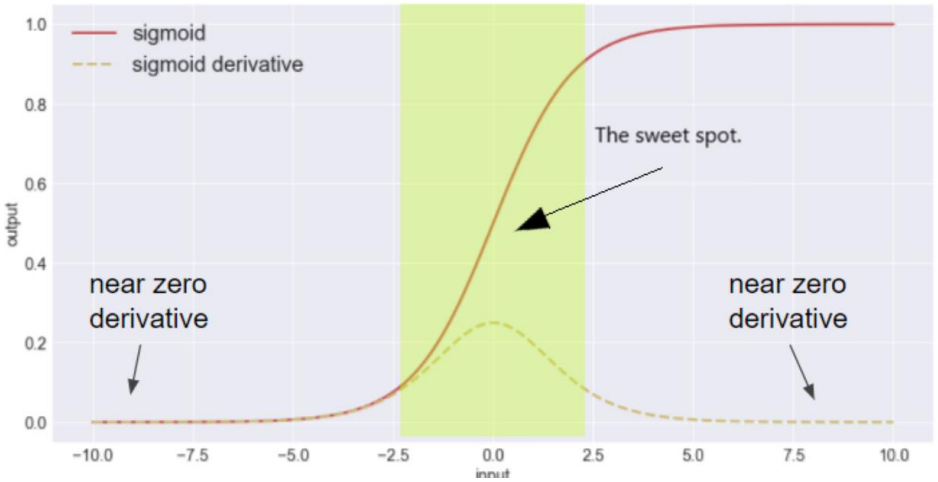
Drop-out (3)

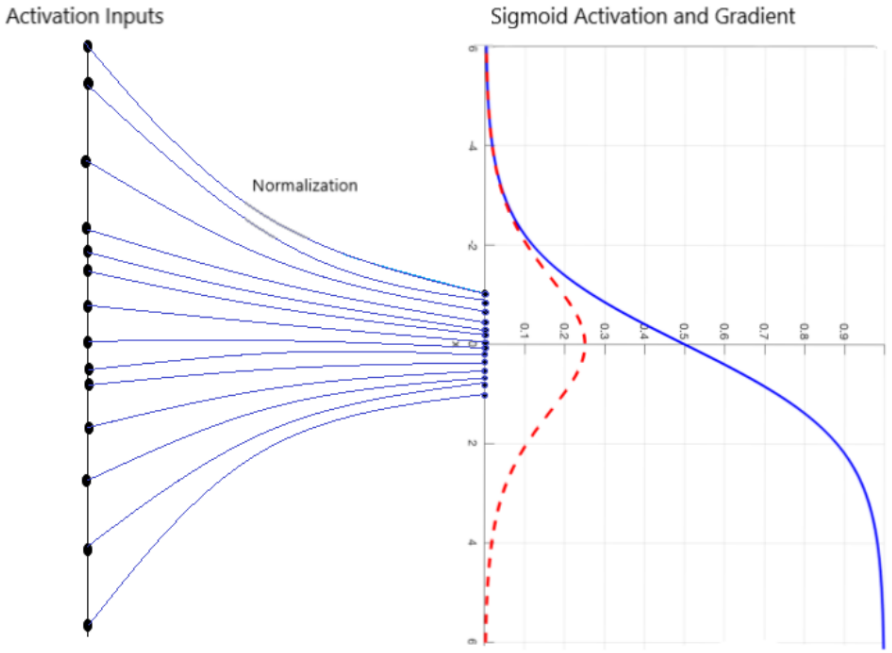
- 학습 데이터 양에 따른 drop-out 효과
  - 데이터의 양을 계속 늘리면 어느 순간부터 dropout의 효과 감소
  - 데이터 양이 많아지면서 overfitting 가능성이 줄어들기 때문



배치 정규화 (1)

- 공변량 시프트 (Covariate Shift) 현상
  - 학습이 진행됨에 따라 각각의 계층에서 입력으로 사용되는 데이터의 분포가 수시로 변경됨
  - 층이 깊어질수록 문제가 더욱 심각
  - 학습을 방해하는 요인으로 작용





배치 정규화 (2)

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

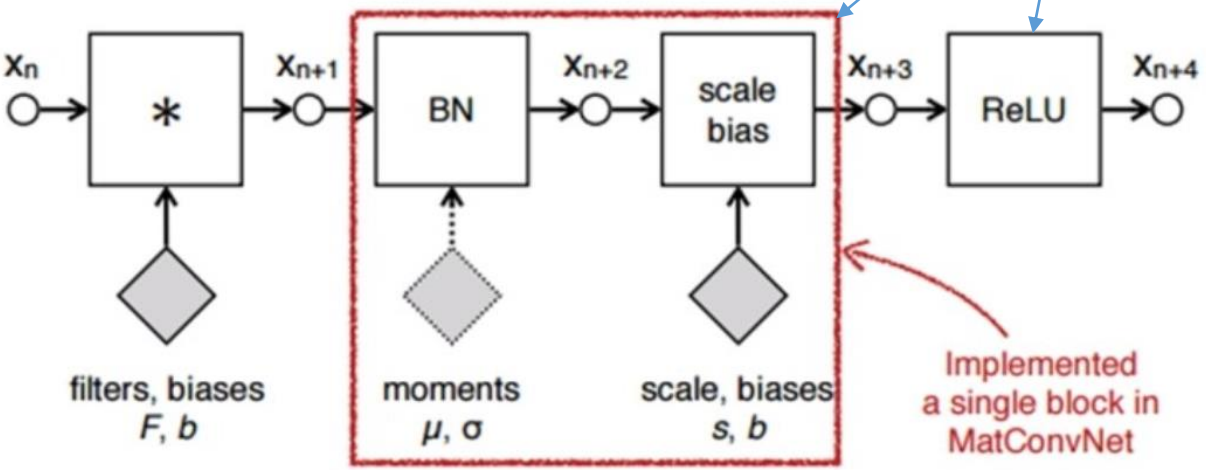
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

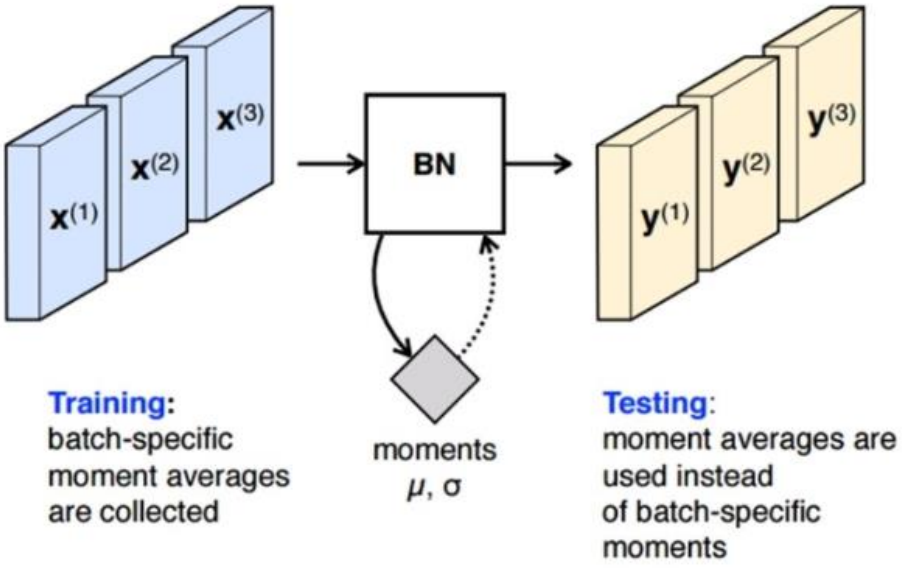
$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

BN은 일반적으로 non-linear 함수 앞쪽에 배치



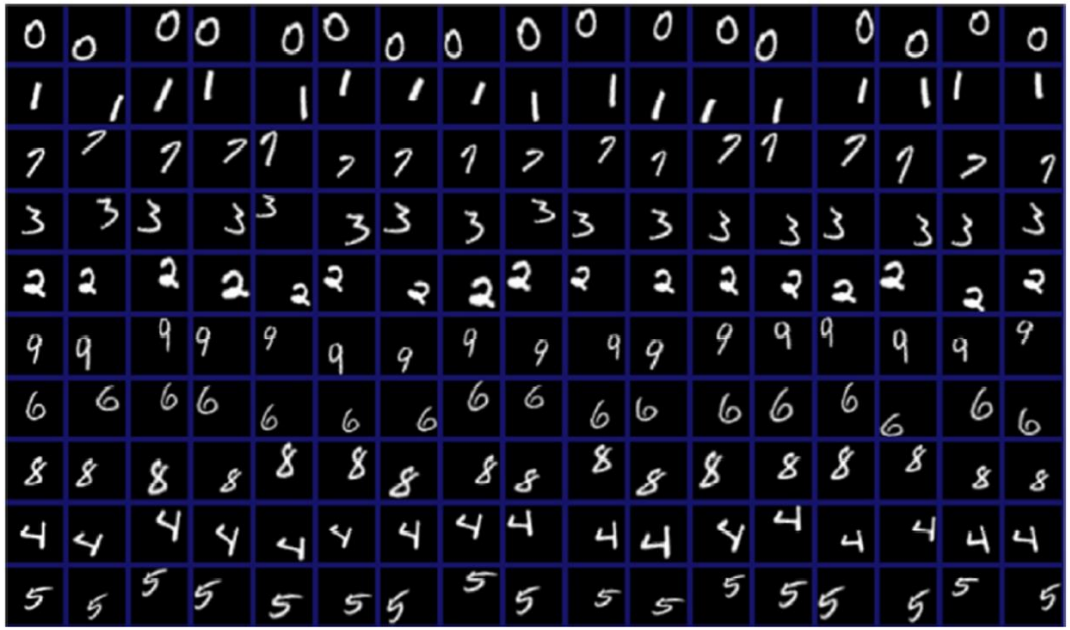
배치 정규화 (3)

- BN 적용 방법
  - Training ➔ mini-batch마다 γ과 β를 구하고 그 값을 저장함
  - Test ➔ 학습 시 mini-batch마다 구했던 γ과 β의 평균을 사용
- BN 적용 효과
  - BN을 적용하면 regularization 효과를 얻을 수 있기 때문에 drop-out layer 제거 가능
  - Vanishing/exploding gradient 문제 해결
  - 학습 속도 개선



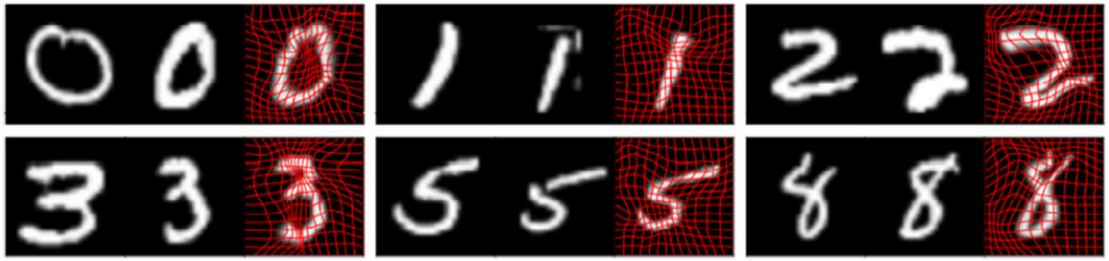
데이터 확대 (1)

- 데이터 확대 (Data Augmentation)
  - 학습데이터를 인위적으로 변형하여 확대
- Affine 변형
  - 이동, 회전, 크기 변형
  - 수작업 변형, 모든 부류가 같은 변형 사용

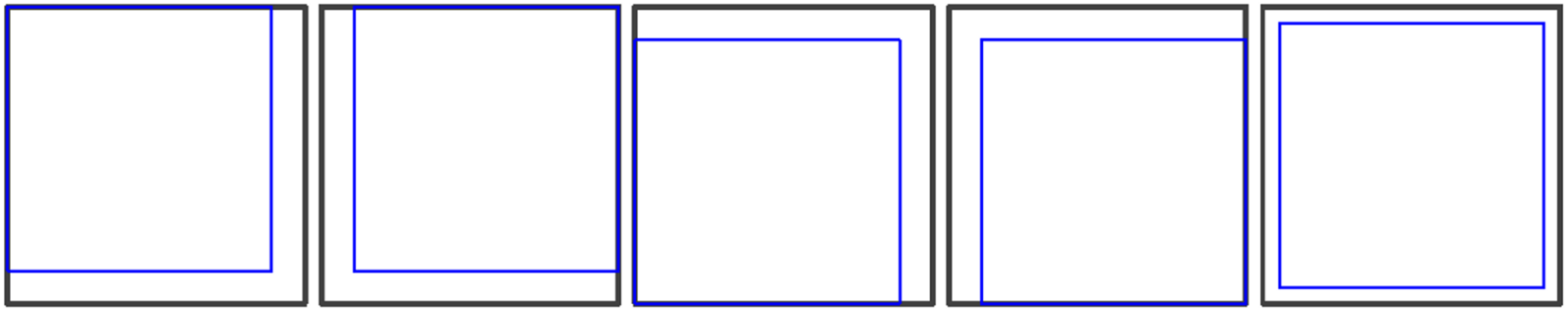


데이터 확대 (2)

- 모핑을 이용한 변형
  - 비선형 변환으로 Affine 변환에 비해 훨씬 다양한 형태의 확대



- 자연 영상 확대
  - 256×256 영상에서 224×224 영상으로 Cropping하여 이동, 좌우 반전 등을 적용하여 데이터 확대

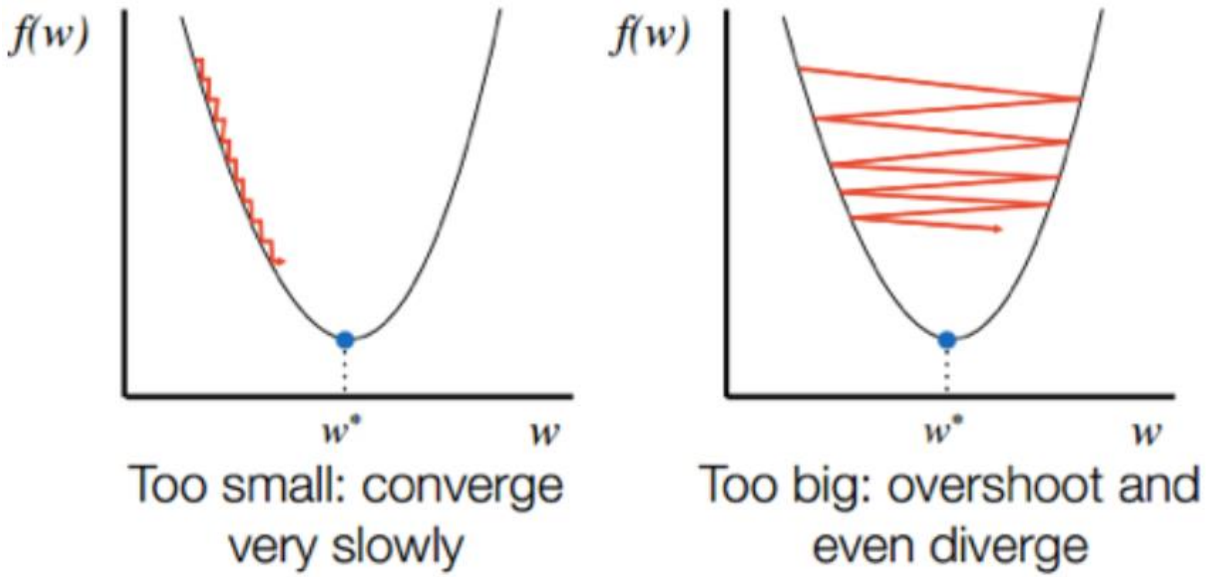


Hyperparameters

- Learning rate
- Loss function
- Regularization parameters
- Mini-batch 크기
- Training 반복 횟수
- Hidden Unit 수
- 가중치 초기화
- ...

Learning Rate

- 학습 초기에 값을 크게 설정했다가 학습의 진행과 함께 학습율을 점차 줄여가는 방법
- 신경망의 모든 층에서 같은 값을 사용하는 것이 아니라 서로 다른 값을 사용
  - AdaGrad, Adam → 학습률 자동 감소



Loss Function

- MSE Loss → Regression

$$E = \frac{1}{2}\sum_{i=1}^{n}(y_i - t_i)^2$$

- Cross Entropy Loss → Classification

$$E = \sum_{i=1}^{K} y_i log(\hat{y}_i)$$

- Focal Loss → Class Imbalance

$$\mathrm{FL}(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t).$$

- Dice Loss → Segmentation

$$\text{for each class } c, \quad DSC_c = \frac{\sum_{i=1}^{N} p_{ic} g_{ic} + \epsilon}{\sum_{i=1}^{N} p_{ic} + g_{ic} + \epsilon}$$

$$DL_c = \sum_{c} 1 - DSC_c$$

- ...

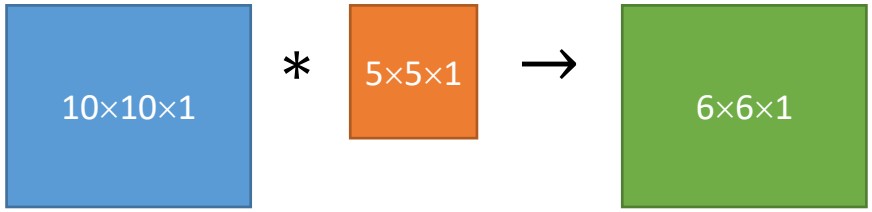
Hidden Unit의 수

- Hidden Layer의 수가 증가할 경우
  - DNN이 더욱 많은 문제를 해결할 수 있도록 학습 능력이 개선됨
  - 그러나 Network의 크기가 커질수록 overfitting에 빠질 가능성이 높고 신경망에 대한 학습 시간도 길어짐
  - 적절한 결과를 도출하기 위해서는 학습 데이터 양 또한 늘려야 함
- Hidden Unit의 개수
  - 모든 Hidden Layer의 뉴런의 개수를 동일하게 유지하는 것이 같은 Hidden Layer의 개수에 뉴런의 개수를 가변적으로 하는 것보다 효과적
  - 첫번째 Hidden Layer에 있는 뉴런의 개수가 Input Layer에 있는 뉴런의 개수보다 큰 것이 효과적인 경우가 많음

Filter의 형태

- 일반적으로 32×32나 28×28과 같은 작은 입력 영상에 대해서는 5×5 필터를 주로 사용하지만 큰 크기의 자연 영상을 처리할 때나 혹은 1단계 필터에 11×11이나 15×15와 같은 큰 크기의 필터를 사용하기도 함
- 여러 개의 작은 크기의 필터를 중첩하여 사용하는 것이 좋음
  - 이는 작은 필터를 여러 개 중첩하면 중간 단계에 있는 non-linearity를 활용하여 원하는 특징을 좀 더 돋보이도록 할 수 있음
  - 층을 깊게 한 신경망은 깊지 않은 경우보다 적은 매개변수로 같은 (혹은 그 이상) 수준의 표현력을 달성할 수 있음

Layer를 1개 사용했을 때



5×5 필터는 2개의 3×3 필터로 대체할 수 있으며, 따라서 더 적은 수의 매개변수를 사용 (5×5=25 〉 2×3×3=18)

Layer를 2개 사용했을 때

Others ...

- Pooling의 윈도우 크기
  - Stride과 같은 값으로 설정하는 것이 일반적
  - 예를 들어, 윈도우가 3×3이면 Stride는 3으로, 4×4이면 Stride를 4로 설정
- Layer별 Feature 학습 시간
  - 앞쪽 Layer들은 몇 번의 학습 epoch만에 feature들이 수렴하는 것을 볼 수가 있으나 뒤쪽 Layer로 갈수록 feature 습득에 오랜 시간이 필요함
    - 40~50 epoch 이상 반복 학습을 통해야 feature들이 보이기 시작함 (image classification 경우)

이미지 명암 정규화

- 이미지 집합에 대한 통계치 사용
  - 학습 이미지의 픽셀 단위 평균을 구해 대상 이미지로부터 평균을 뺀 이미지를 CNN의 입력으로 사용

$$\widetilde{x_{ijk}} = \sum_{n=1}^{N} x_{ijk}^{(n)} \qquad x_{ijk} \leftarrow x_{ijk} - \widetilde{x_{ijk}}$$

$x_{ijk}^{(n)}$: n번째 샘플의 픽셀 (i, j)의 채널 k의 값

- Local Contrast Normalization (LCN)
  - 이미지 한 장 한 장에 대해 개별적으로 처리
  - 감산 정규화(subtractive normalization), 제산 정규화(divisive normalization)

References

- Machine Learning, https://www.coursera.org/learn/machine-learning
- Intuit and Implement: Batch Normalization, https://mc.ai/intuit-and-implement-batch-normalization/
- Automated Bone Age Classification with Deep Neural Networks, http://cs231n.stanford.edu/reports/2016/pdfs/310_Report.pdf
- 라온피플 머신러닝 아카데미, https://laonple.blog.me/
- 기계학습, 한빛아카데미


ANY
QUESTIONS?