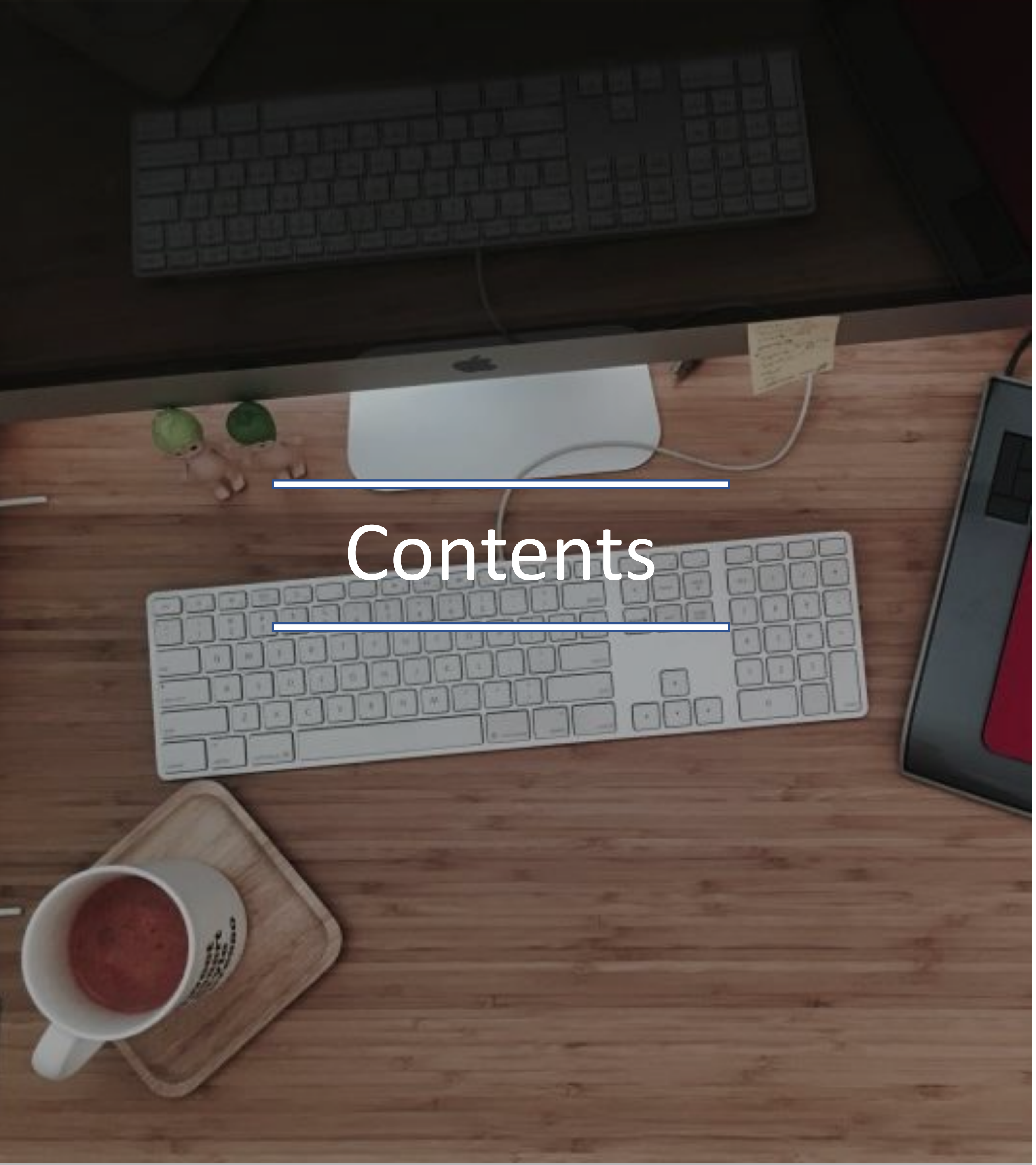
Apply to Machine Learning

Machine Learning

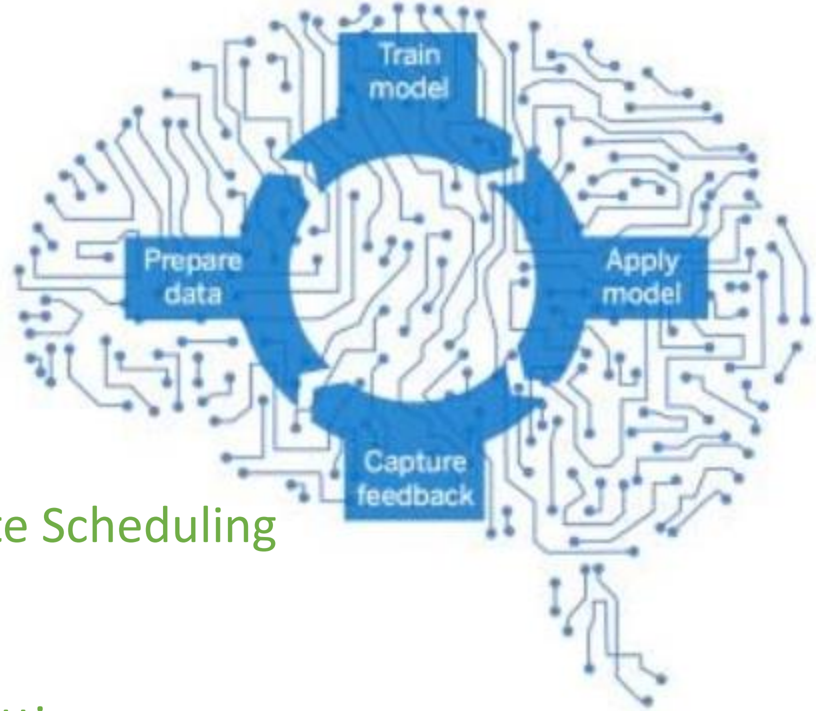
이선우(Seon Woo Lee)

인하대학교

HCI Lab
Inha University
Human-Computer Interaction Lab



Contents



1 Review

2 Learning Rate Scheduling

3 Avoid Overfitting

4 Transfer Learning

5 Practice

1 Review

신경망구현: 학습과정

1. 임의의 w와 b를 설정

2. 주어진 훈련 데이터를 이용하여, 결과값(y)를 도출

3. 결과값(y)와 실제 결과값($\hat{y}$) 사이의 오차 계산

4. 오차(Loss)를 줄이는 방향으로 w와 b를 재설정(학습)

5. 오차를 최소한으로 줄이도록 2~4의 과정을 계속반복진행

가중치 매개변수의 최적값 탐색

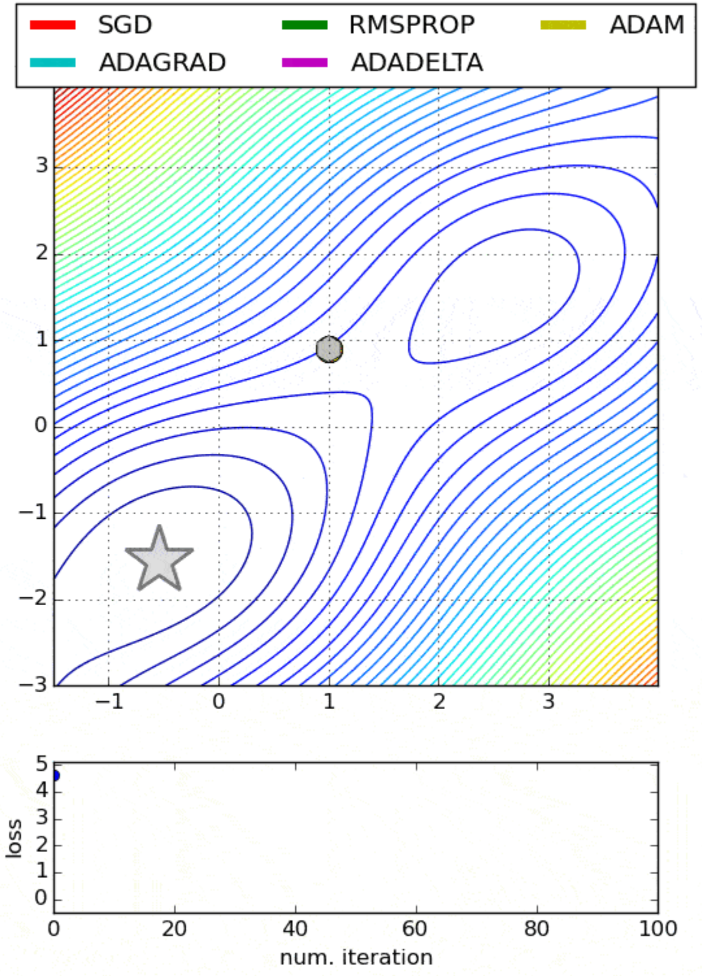
- 매개변수 갱신

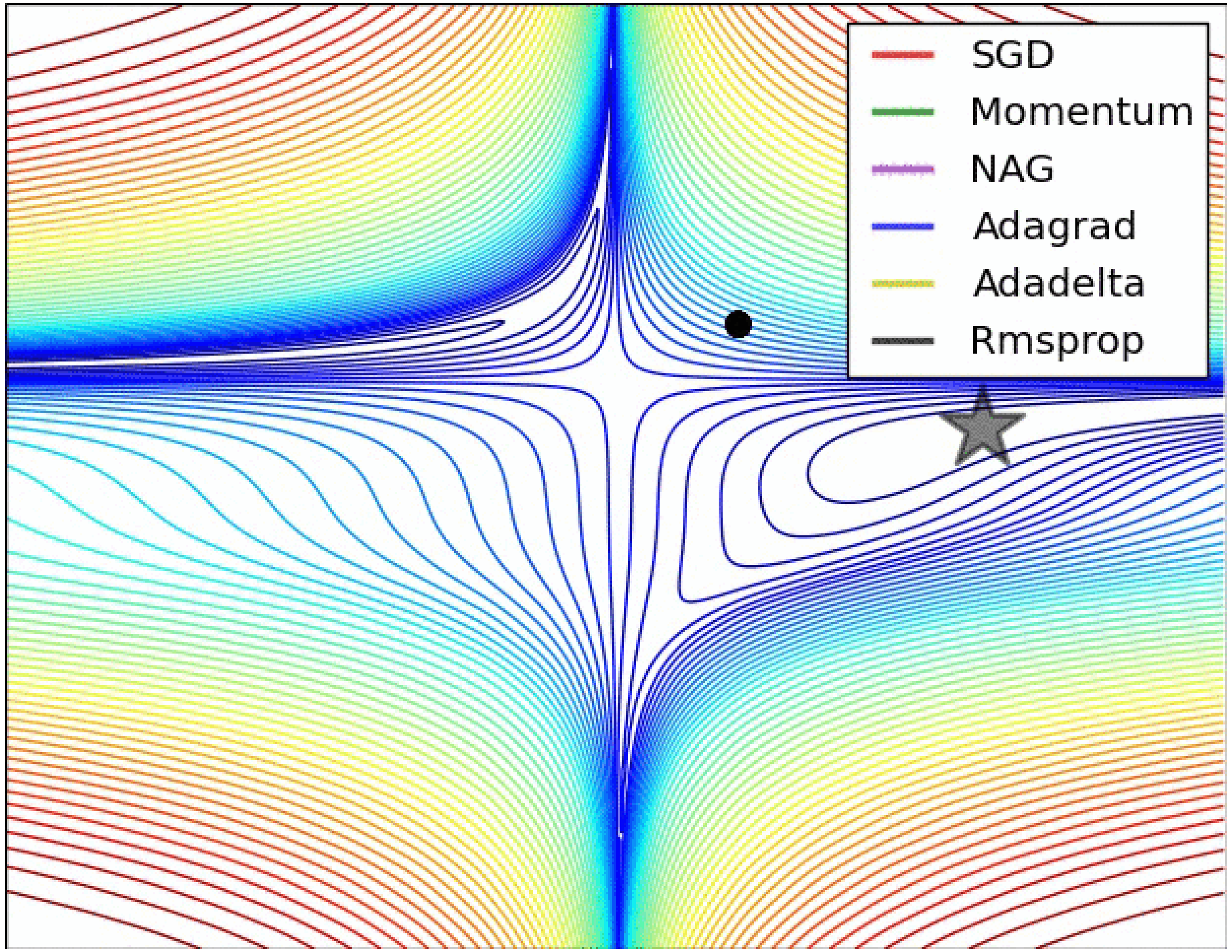
  \- 신경망 학습의 목적

  ->매개변수 최적값 찾기 = **최적화(Optimization)**

가중치 매개변수의 최적값 탐색

- 비교





가중치 매개변수의 최적값 탐색

가중치 표준편차 = **Xavier 초깃값**

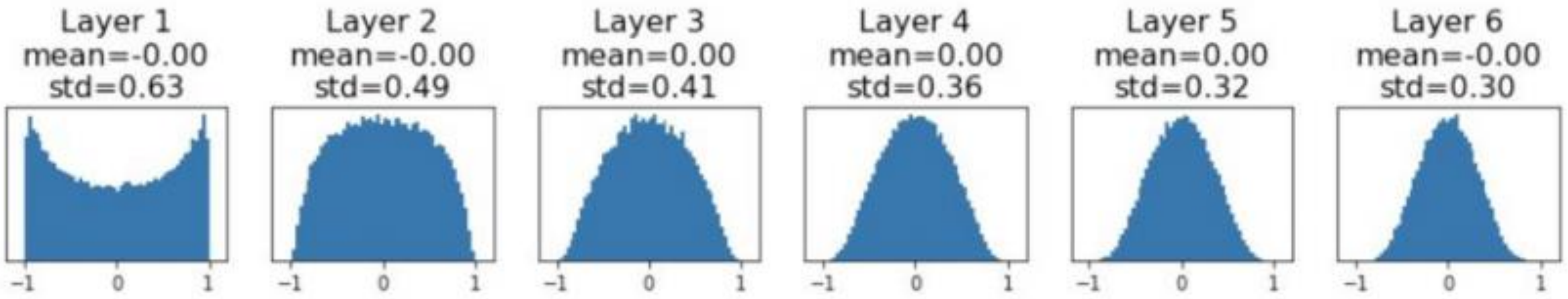
->n개의 노드(뉴런)라면 '1/$\sqrt{n}$'의 분포 사용

(합성곱 층이라면, 필터사이즈의 제곱*채널수)

```
dims = [4096] * 7
hs = []
x = np.random.randn(16, dims[0])
for Din, Dout in zip(dims[:-1], dims[1:]):
    W = np.random.randn(Din, Dout) / np.sqrt(Din)
    x = np.tanh(x.dot(W))
    hs.append(x)
```

"Xavier" initialization:
std = 1/sqrt(Din)

"Just right": Activations are nicely scaled for all layers!



Glorot and Bengio, "Understanding the difficulty of training deep feedforward neural networks", AISTAT 2010

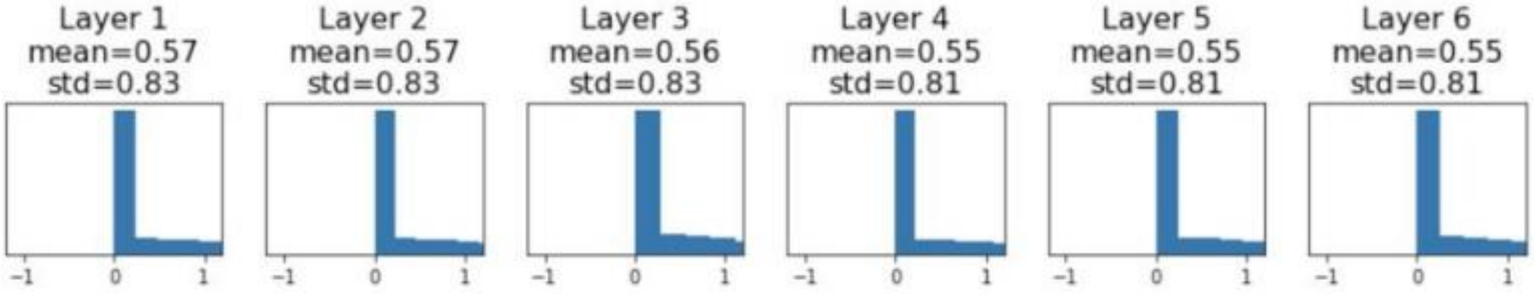
가중치 매개변수의 최적값 탐색

- 활성화 함수가 ReLU를 라면?

```
dims = [4096] * 7
hs = []
x = np.random.randn(16, dims[0])
for Din, Dout in zip(dims[:-1], dims[1:]):
    W = np.random.randn(Din, Dout) * np.sqrt(2/Din)
    x = np.maximum(0, x.dot(W))
    hs.append(x)
```

ReLU correction: std = sqrt(2 / Din)

"Just right": Activations are nicely scaled for all layers!



He et al, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification", ICCV 2015

- He 초깃값
  ->n개의 노드(뉴런)라면 '$\sqrt{2}/\sqrt{n}$'의 표준편차 사용
  ->ReLU의 불필요한 영역(음의 영역) 처리

가중치 매개변수의 최적값 탐색

- 배치 정규화

  - Batch Nomalization
  - 각 층이 활성화를 적당히 퍼트리도록 '강제'
  - 초깃값에 큰 신경 쓸 필요가 없고, 오버피팅 억제

$$\widehat{x}^{(k)} = \frac{x^{(k)} - \mathrm{E}[x^{(k)}]}{\sqrt{\mathrm{Var}[x^{(k)}]}}$$

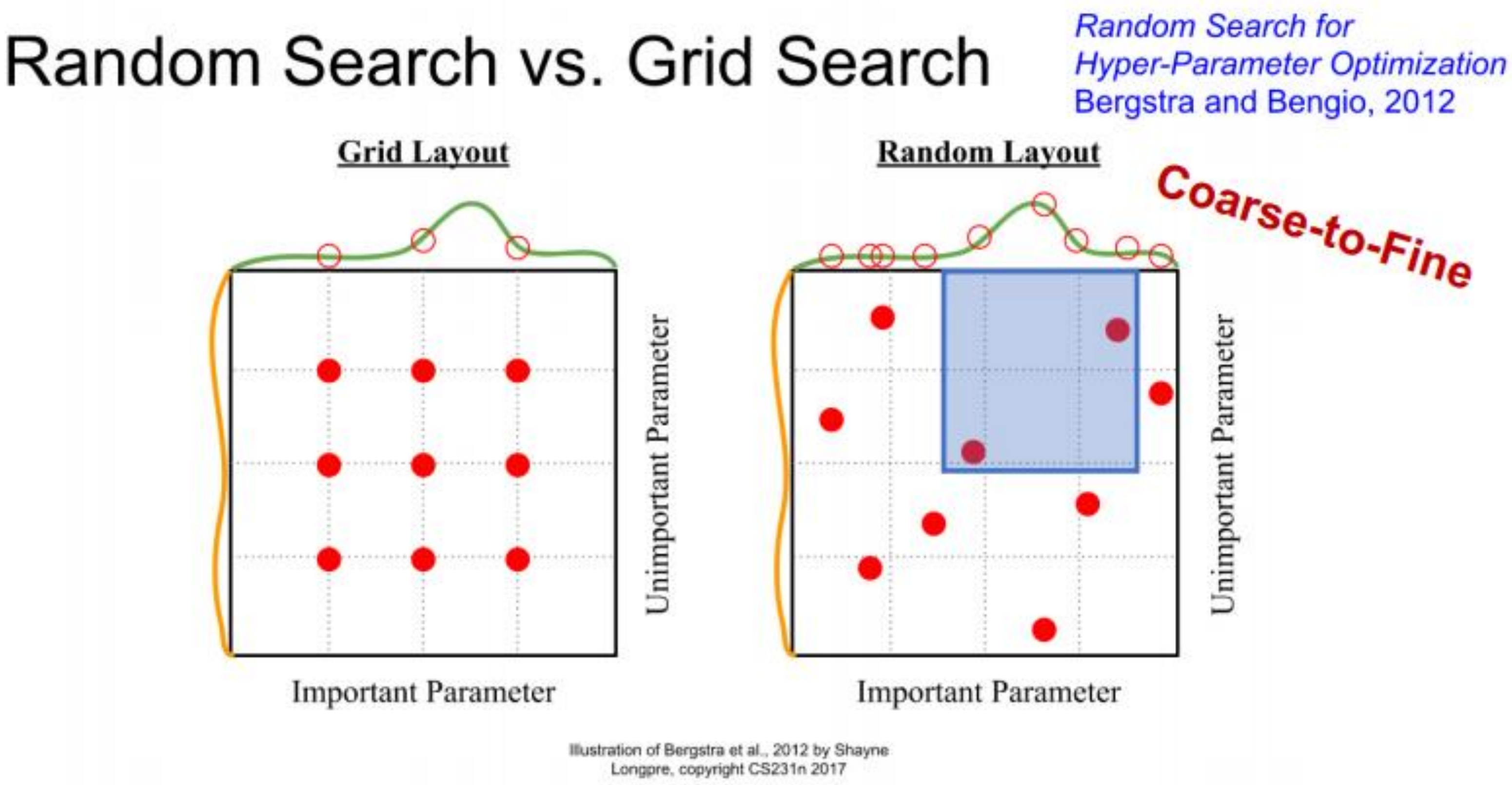
"각 층에서 활성화값이 적당히 분포되도록 조정"

하이퍼파라미터 설정

- 하이퍼파라미터 설정 절차

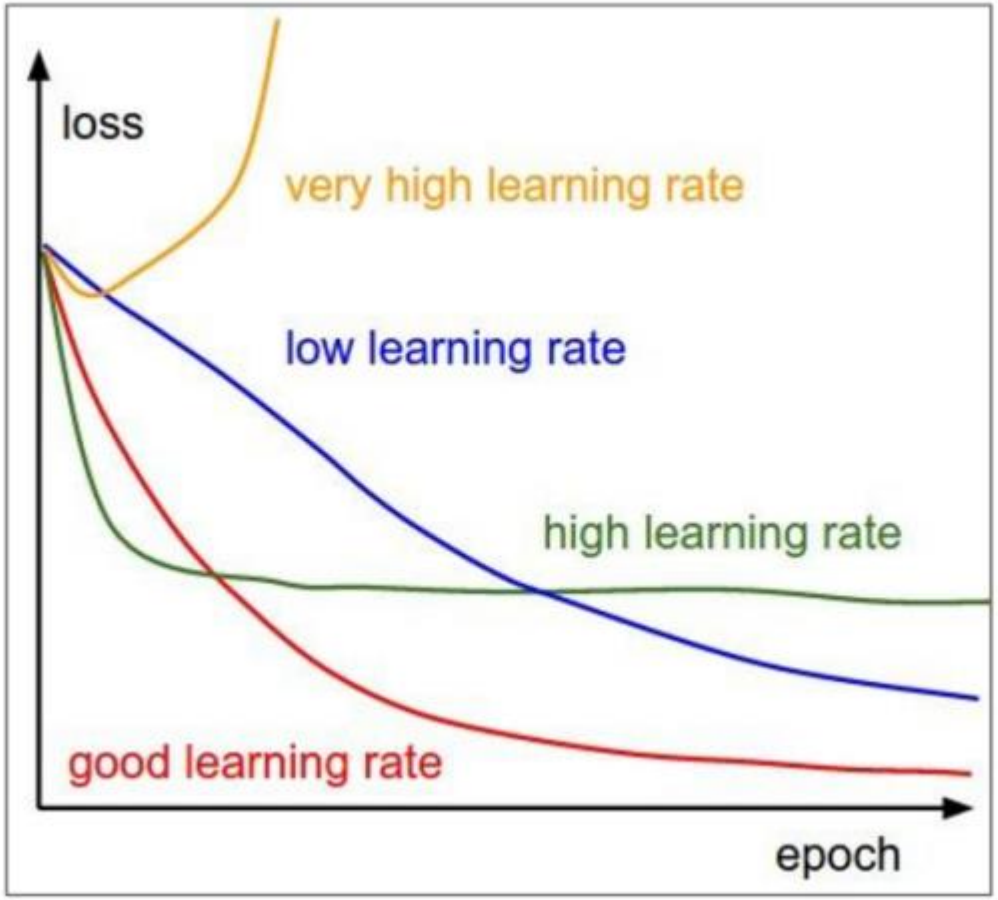
- Step 1: 초기 오차(Loss)값 확인
- Step 2: 작은 데이터에 대하여 학습
- Step 3: 오차가 줄어들 수 있는 학습속도(Learning Rate)확인
- Step 4: 1~5정도의 에폭 실험
- Step 5: 10~20 에폭실험
- Step 6: 끝까지 학습시켜보기
- Step 7: Step 5로 돌아가서 조정(Fine Tune)작업 실시

하이퍼파라미터 설정


Random Search vs. Grid Search
Random Search for
Hyper-Parameter Optimization
Bergstra and Bengio, 2012
Grid Layout
Random Layout
Coarse-to-Fine
Unimportant Parameter
Important Parameter
Unimportant Parameter
Important Parameter
Illustration of Bergstra et al., 2012 by Shayne
Longpre, copyright CS231n 2017

2 Learning Rate Scheduling

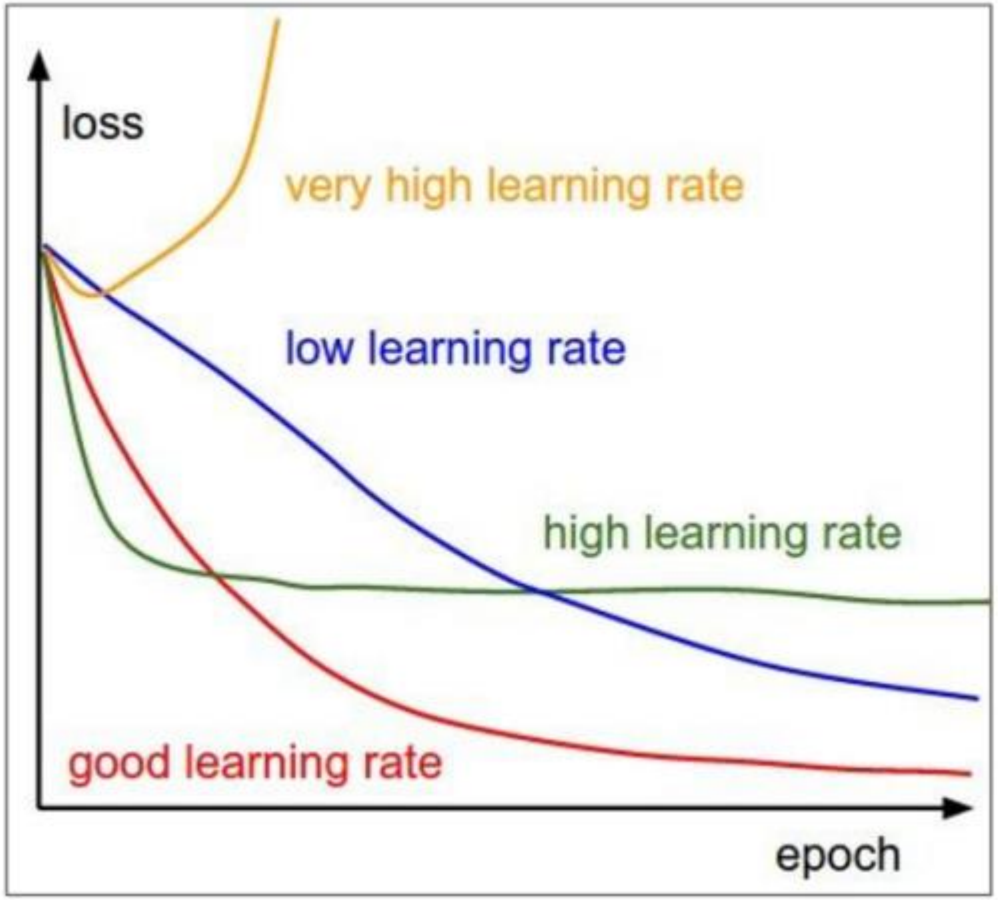
학습속도 스케줄링



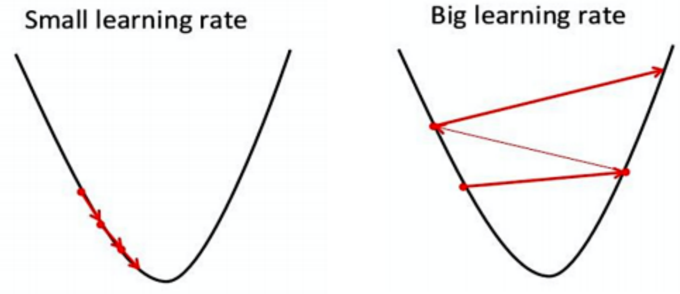
문제: 최적의 학습 속도(Learning Rate)는 어느것일까요?

정답: 상황에 따라서 다르게 정해줘야 한다!
(처음에는 강하게!, 가면 갈수록 학습 속도를 줄여야 한다.)

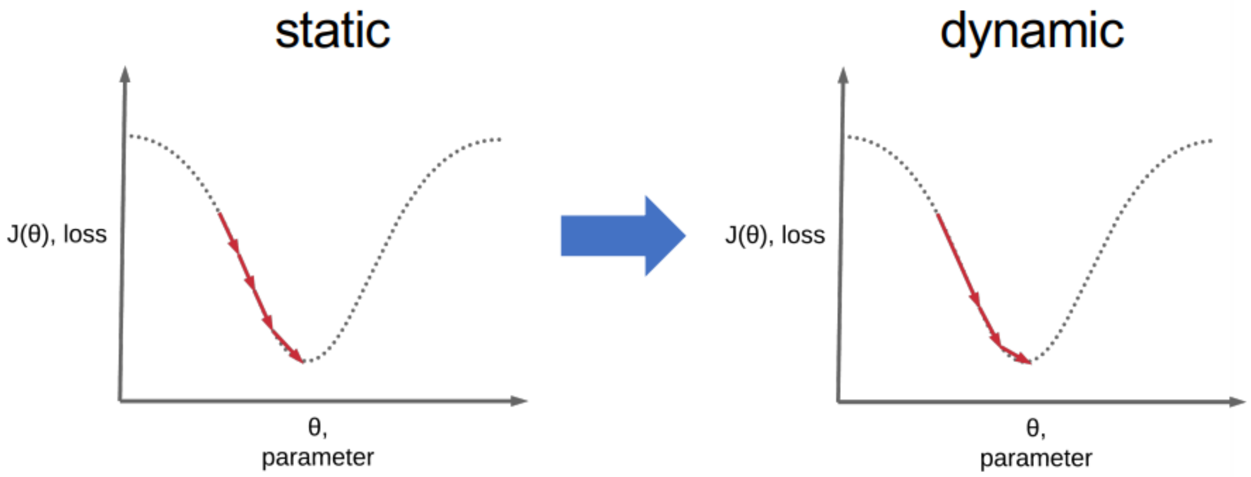
학습속도 스케줄링



loss
very high learning rate
low learning rate
high learning rate
good learning rate
epoch

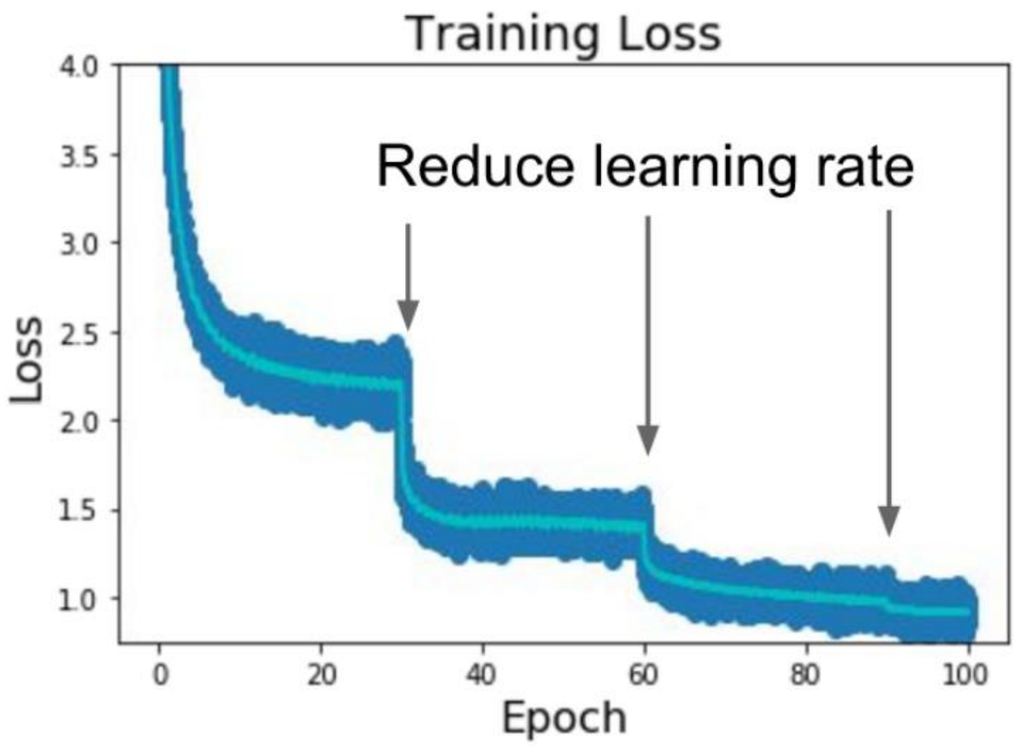


Small learning rate
Big learning rate



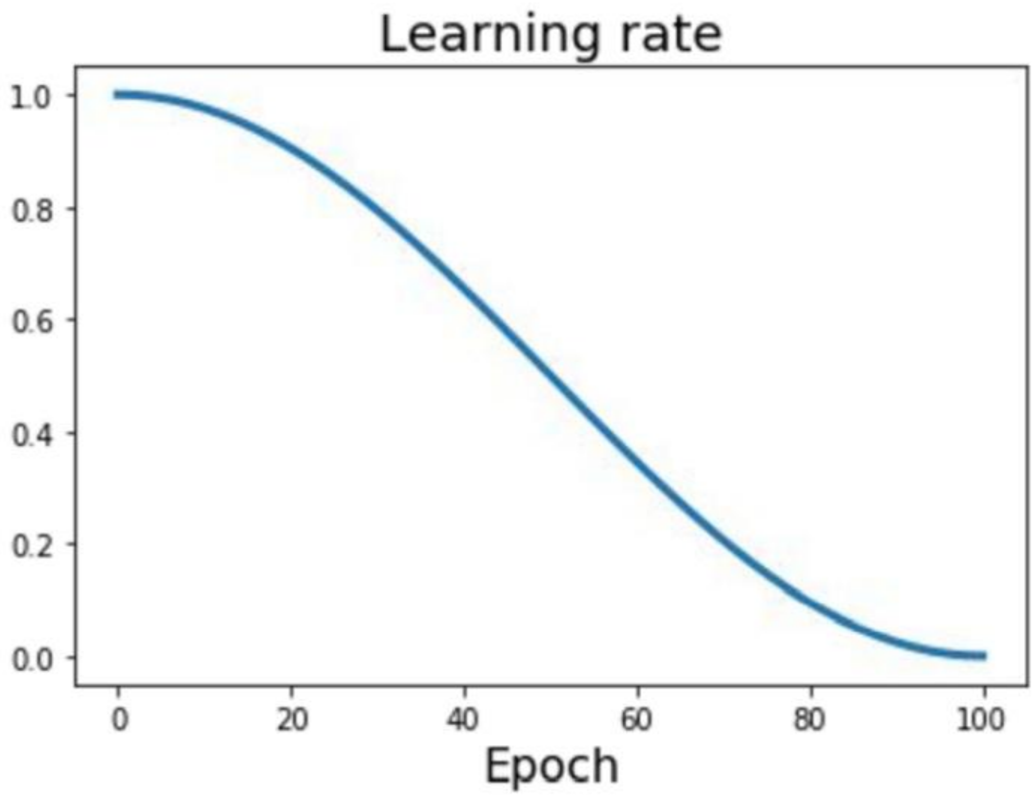
static
dynamic
J(θ), loss
θ,
parameter
J(θ), loss
θ,
parameter

학습속도 감쇠



일정 Epoch마다 학습 속도를 줄이는 방법이다.
예) 30 에폭마다 10%씩 학습속도 감소

학습속도 감쇠



학습 속도를 특정 함수의 형태를 띄게끔 구성한다.

Cosine: $\alpha_t = \frac{1}{2}\alpha_0\left(1 + \cos(t\pi/T)\right)$

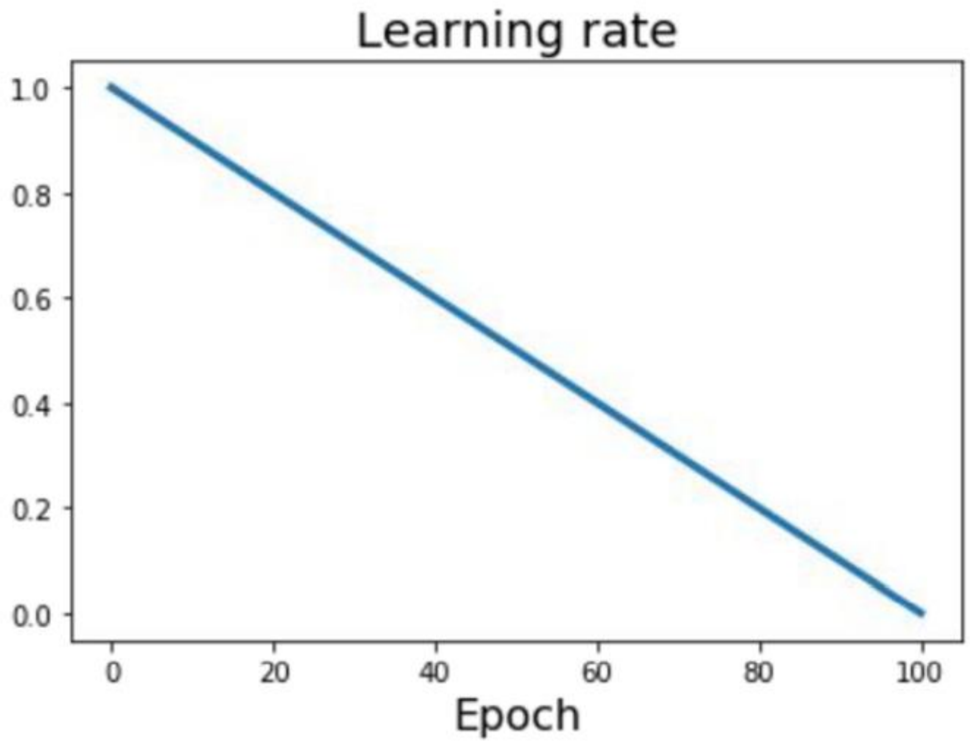
α_0 : Initial learning rate

α_t : Learning rate at epoch t

T : Total number of epochs

Loshchilov and Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts”, ICLR 2017
Radford et al, “Improving Language Understanding by Generative Pre-Training”, 2018
Feichtenhofer et al, “SlowFast Networks for Video Recognition”, arXiv 2018
Child at al, “Generating Long Sequences with Sparse Transformers”, arXiv 2019

학습속도 감쇠



Devlin et al, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, 2018

학습 속도를 특정 함수의 형태를 띄게끔 구성한다.

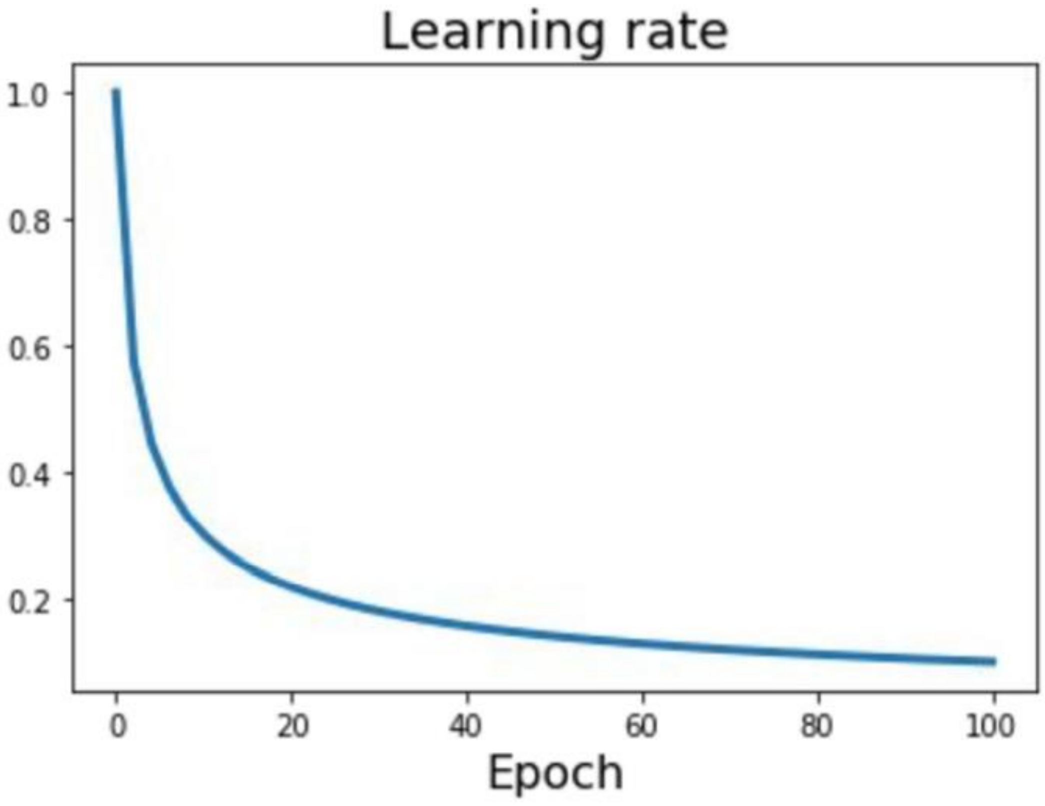
Linear: $\alpha_t = \alpha_0(1 - t/T)$

α_0 : Initial learning rate

α_t : Learning rate at epoch t

T : Total number of epochs

학습속도 감쇠



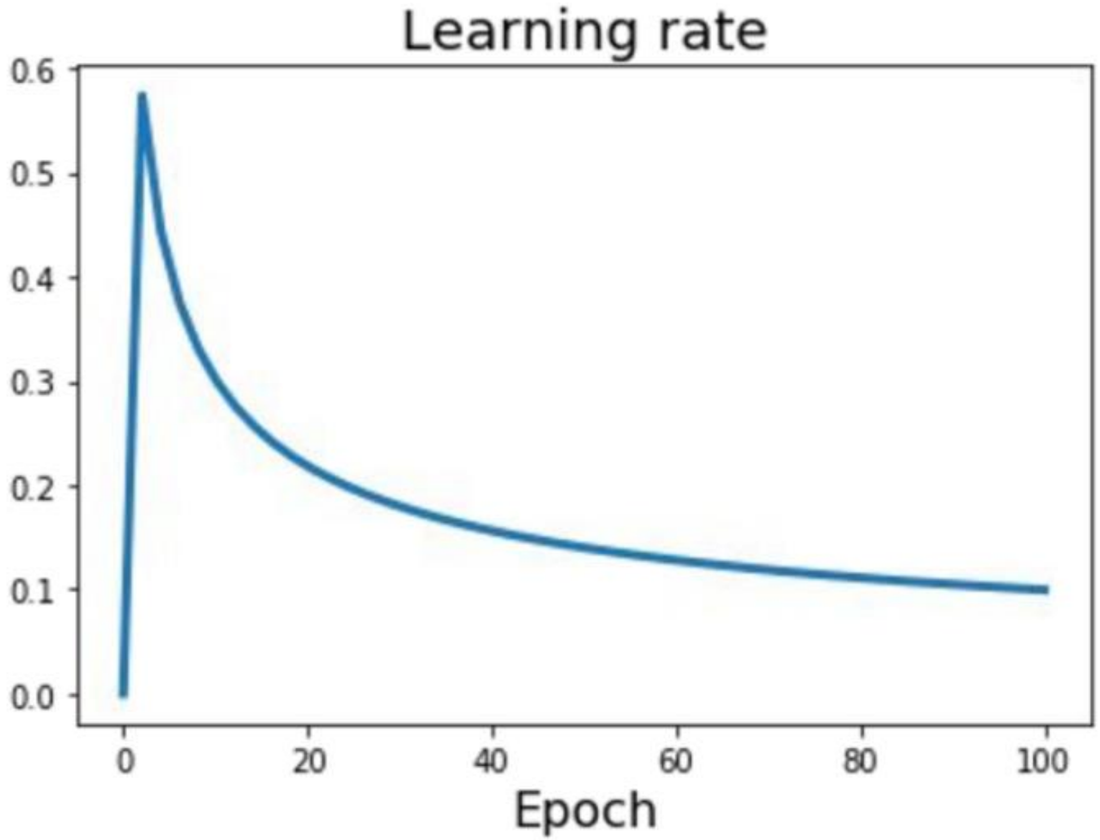
학습 속도를 특정 함수의 형태를 띄게끔 구성한다.

Inverse sqrt: $\alpha_t = \alpha_0 / \sqrt{t}$

α_0 : Initial learning rate
α_t : Learning rate at epoch t
T : Total number of epochs

Vaswani et al, "Attention is all you need", NIPS 2017

학습속도 감쇠



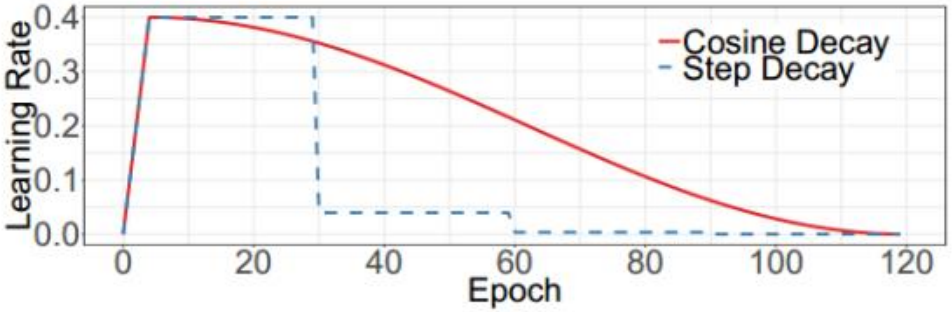
처음에는 학습속도를 크게잡고 점차 줄여 나가는 방법

경험에 의한 방법(Empirical rule of thumb)

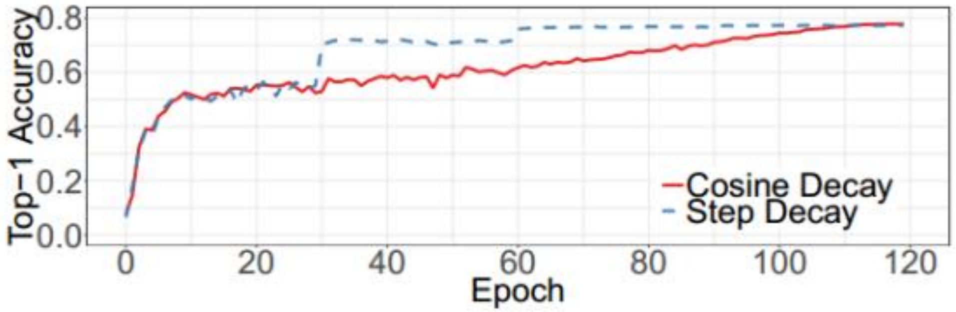
학습속도 감쇠

- 실제결과

Large Batch	Small Batch
• Accurate estimate of the gradient (low variance)	• Noisy estimate of the gradient (high variance)
• High computation cost per iteration	• Low computation cost per iteration
• High availability of parallelism (fast training)	• Low availability of parallelism (slow training)



(a) Learning Rate Schedule



(b) Validation Accuracy

He, Tong, et al. "Bag of tricks for image classification with convolutional neural networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

3 Avoid Overfitting

오버피팅 대응책 : 정규화

- Overfitting

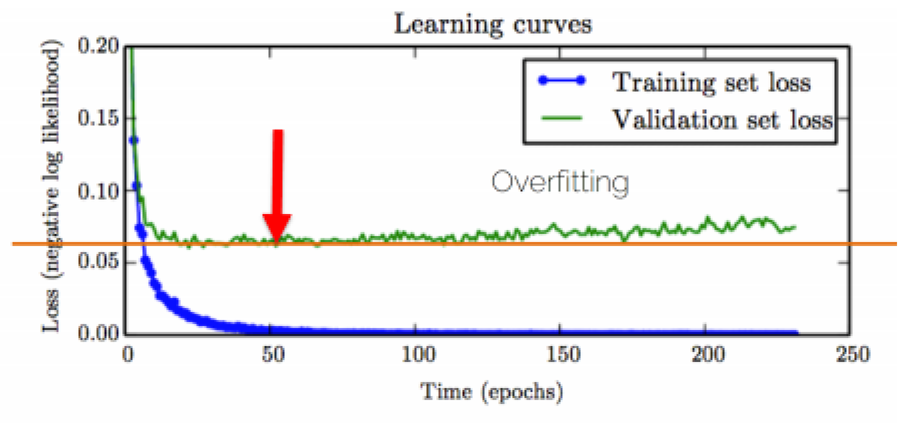
  - 훈련데이터에 너무 최적화 됨
  - 시험데이터 정확도 저하
  - 2가지 해결책

    1) 가중치감소　　2) 드롭아웃

오버피팅 대응책(+)

- 그 밖의 과적합(Overfitting) 피하는 방법

Early Stopping: Always do this




```python
# Additional information
EPOCH = 5
PATH = "model.pt"
LOSS = 0.4

torch.save({
            'epoch': EPOCH,
            'model_state_dict': net.state_dict(),
            'optimizer_state_dict': optimizer.state_dict(),
            'loss': LOSS,
            }, PATH)
```

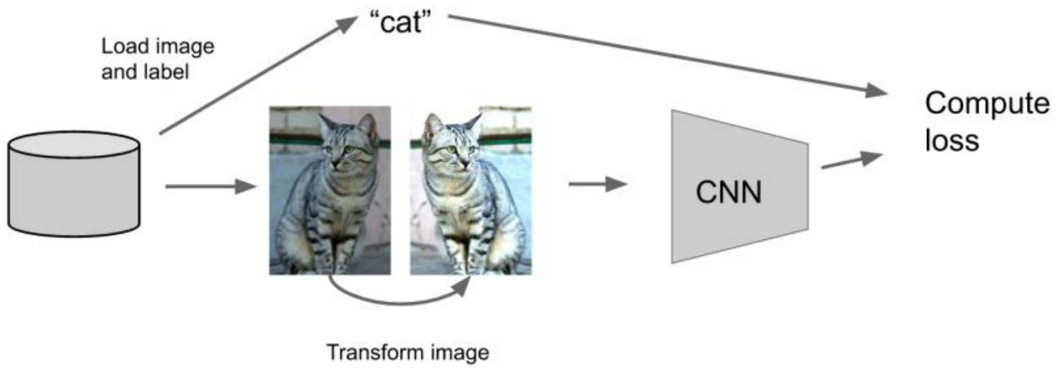
checkpont

Stop training the model when accuracy on the validation set decreases
Or train for a long time, but always keep track of the model snapshot
that worked best on val

오버피팅 대응책(+)

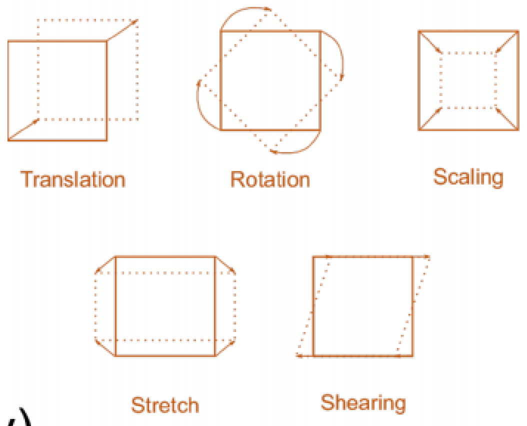
- 그 밖의 과적합(Overfitting) 피하는 방법(이미지)

Regularization: Data Augmentation



Data Augmentation
Get creative for your problem!

Random mix/combinations of :
- translation
- rotation
- stretching
- shearing,
- lens distortions, … (go crazy)



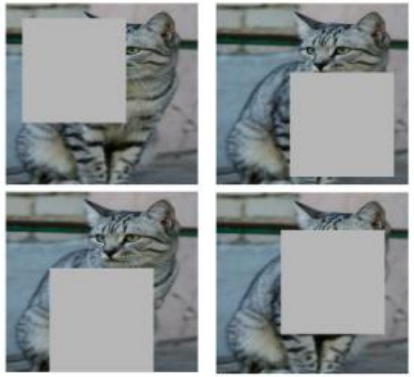
오버피팅 대응책(+)

- 그 밖의 과적합(Overfitting) 피하는 방법(이미지)

Regularization: Cutout

Training: Set random image regions to zero
Testing: Use full image

Examples:
Dropout
Data Augmentation
DropConnect
Cutout / Random Crop



Works very well for small datasets like CIFAR, less common for large datasets like ImageNet

DeVries and Taylor, "Improved Regularization of Convolutional Neural Networks with Cutout", arXiv 2017

Regularization: Mixup

Training: Train on random blends of images
Testing: Use original images

Examples:
Dropout
Data Augmentation
DropConnect
Cutout / Random Crop
Mixup

Target label:
cat: 0.4
dog: 0.6

Randomly blend the pixels of pairs of training images, e.g. 40% cat, 60% dog

Zhang et al, "*mixup*: Beyond Empirical Risk Minimization", ICLR 2018

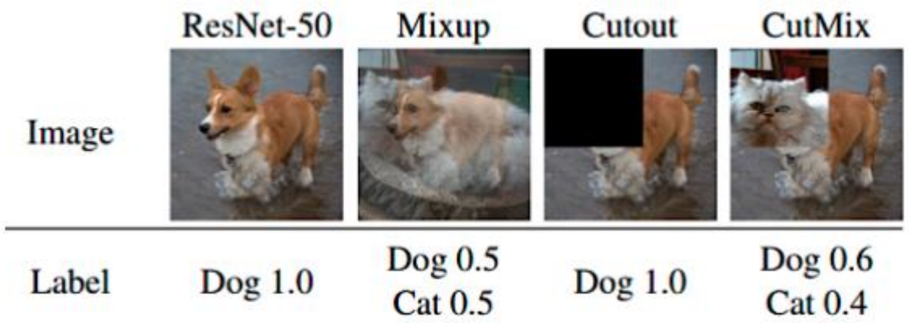
오버피팅 대응책(+)

- 그 밖의 과적합(Overfitting) 피하는 방법(이미지)

Regularization: Cutmix

Training: Train on random blends of images
Testing: Use original images

Examples:
Dropout
Data Augmentation
DropConnect
Cutout / Random Crop
Mixup
Cutmix



Yun, Sangdoo, et al. "Cutmix: Regularization strategy to train strong classifiers with localizable features." *ICCV* 2019

Regularization - In practice

Training: Add random noise
Testing: Marginalize over the noise

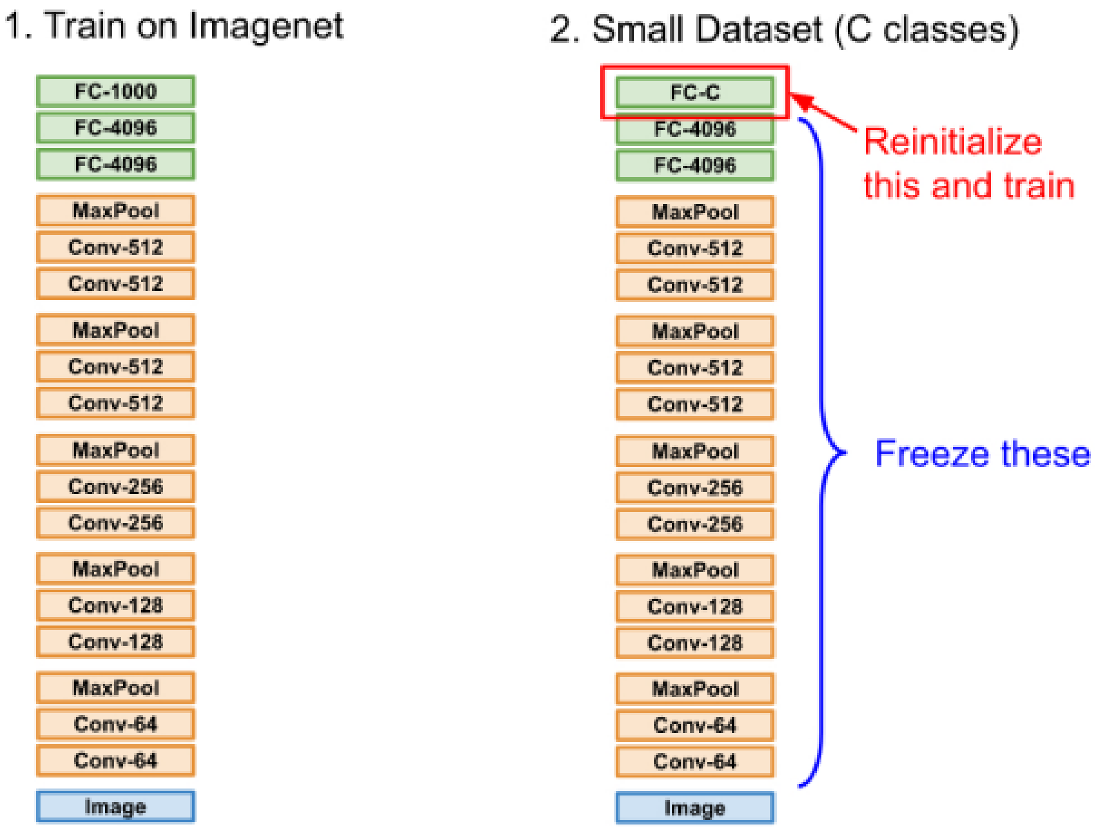
Examples:
Dropout
Data Augmentation
DropConnect
Cutout / Random Crop
Mixup
Cutmix

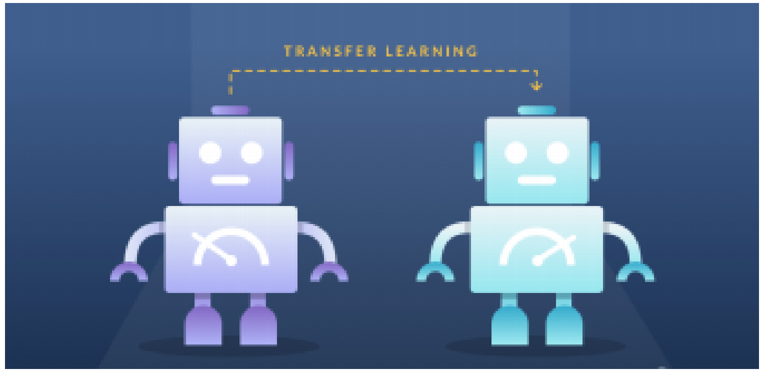
- Consider dropout for large fully-connected layers
- Batch normalization and data augmentation almost always a good idea
- Try cutout and mixup especially for small classification datasets
- Try Cutmix if possible

4 전이학습(Transfer Learning)

전이학습(Transfer Learning)

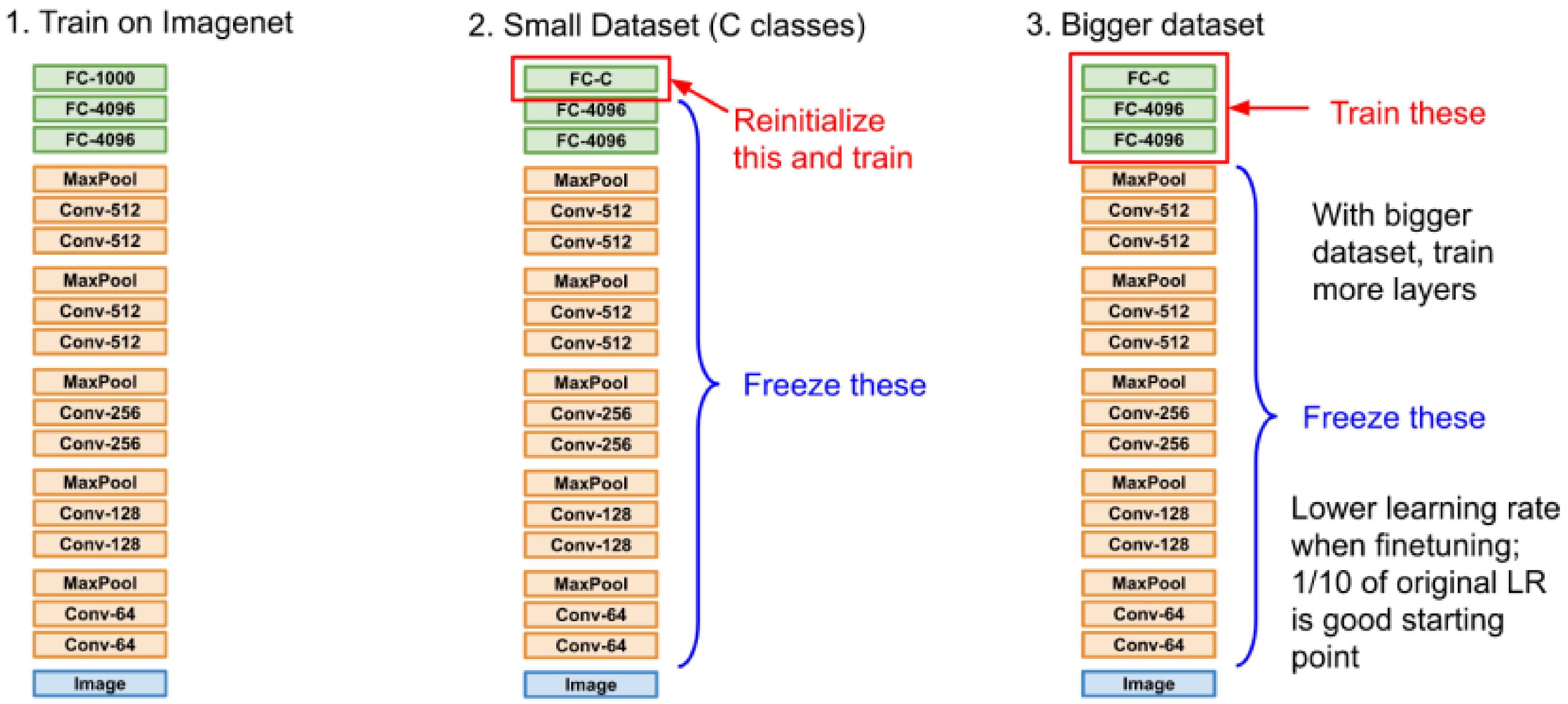
- 사전에 학습된 데이터를 이용하여 문제를 해결하는 방법





전이학습(Transfer Learning)

- 사전에 학습된 데이터를 이용하여 문제를 해결하는 방법



5 Practice

Pytorch & Numpy Tutorials

https://github.com/LEE-SEON-WOO/DL-Tutorials-Py_or_Tf

DL-Tutorials-Py_or_Tf

Star 0 LinkedIn

Among the top 10 ML repos on GitHub

- Learn Python DL-Tutorials-Py_or_Tf with notebooks.
- Use data science libraries like NumPy and Pandas.
- Implement basic ML models in TensorFlow 2.0 + Keras or PyTorch.
- Learn best practices with clean code, simple math and visualizations.

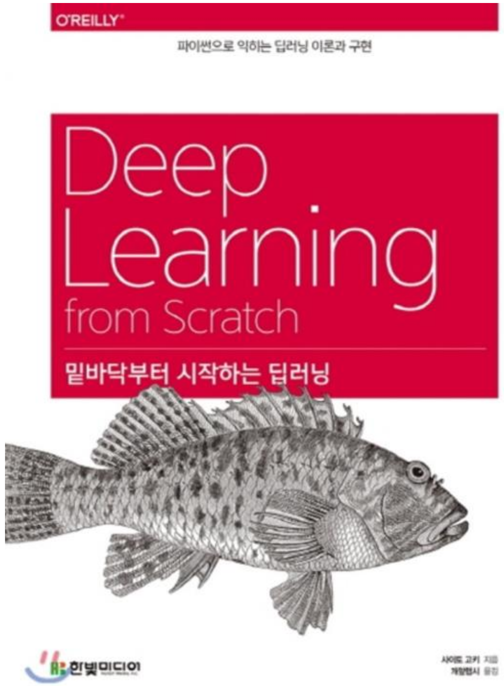
Notebooks	Python	NumPy
Pandas	TensorFlow	PyTorch
Linear Regression	Logistic Regression	Multilayer Perceptrons
Data & Models	Utilities	Preprocessing
Convolutional Neural Networks	Embeddings	Recurrent Neural Networks

Book

주교재

밑바닥부터 시작하는 딥러닝 1, 사이토 고키 지음, 개앞맵시 옮김 .

(Deep Learning from Scratch의 번역서입니다 .)



부교재

Pytorch로 시작하는 딥러닝, 비슈누 수브라마니안 지음 김태완 옮김 .

(Deep Learning with PYTORCH 의 번역서입니다 .)