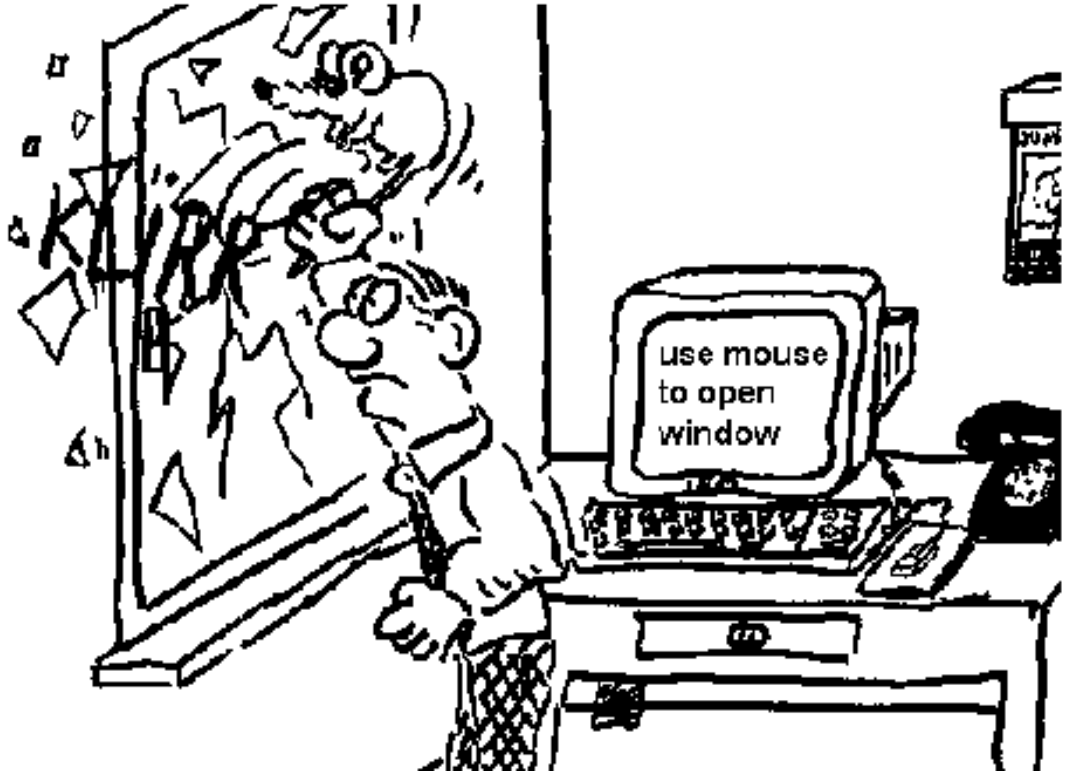
Deep Learning Basics

Lecture 3: Optimization

최성준 (고려대학교 인공지능학과)

boostcamp AI Tech

Introduction

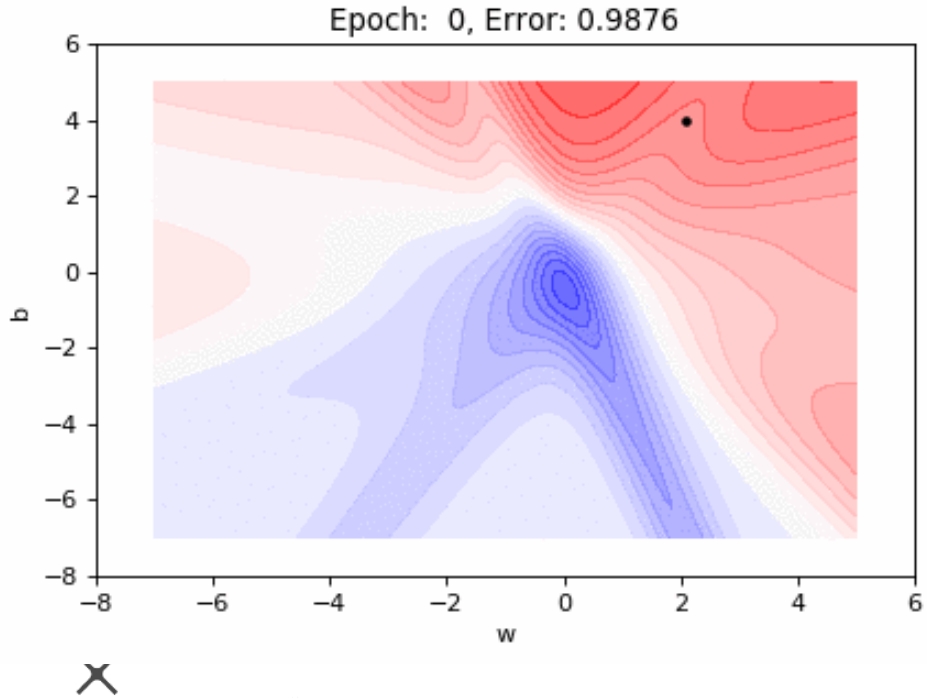


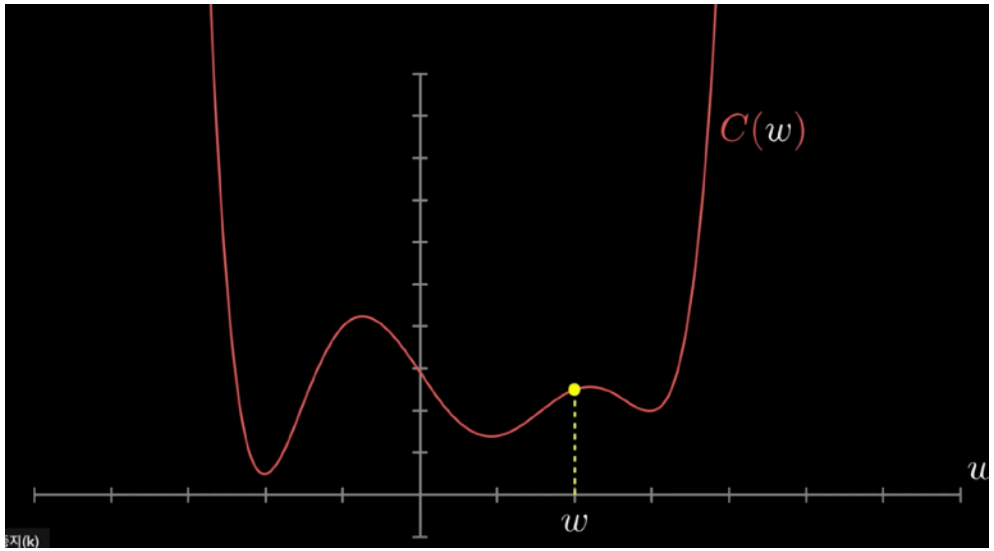
"Language is the source of misunderstandings"

Antoine de Saint-Exupéry (1900-1944)

Introduction

- Gradient Descent
  - First-order iterative optimization algorithm for finding a local minimum of a differentiable function.





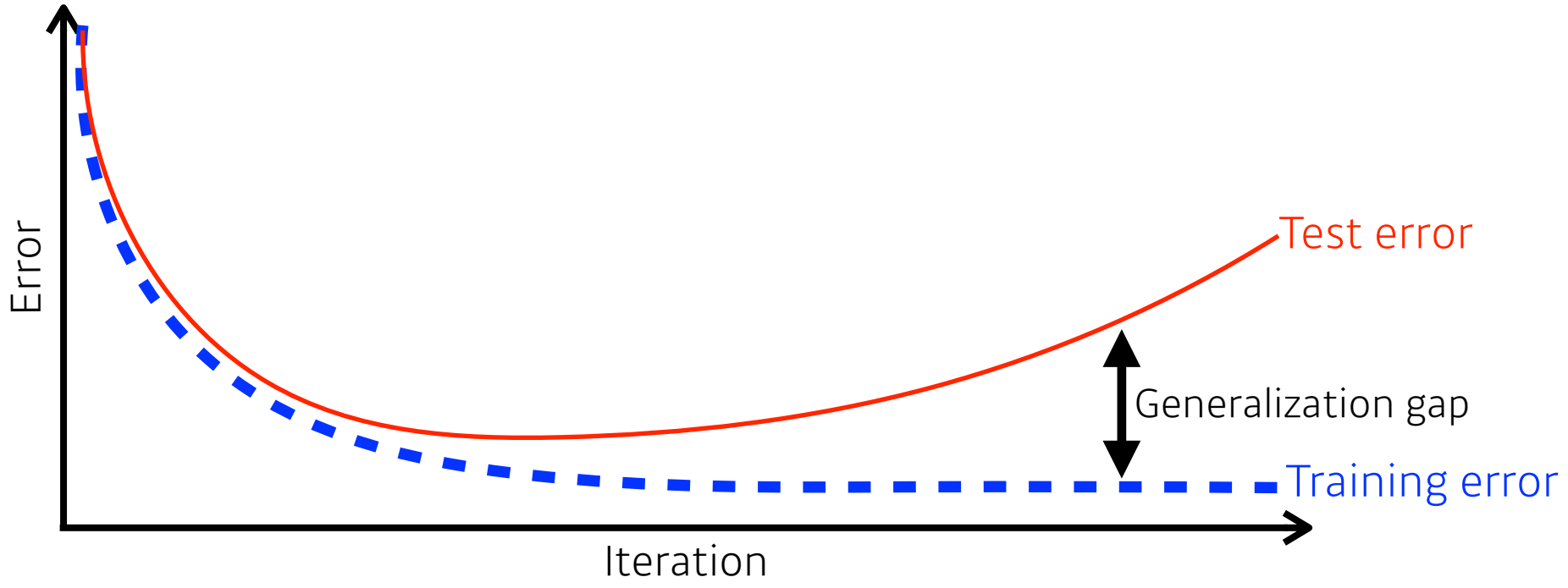
Important Concepts in **Optimization**

Important Concepts in **Optimization**

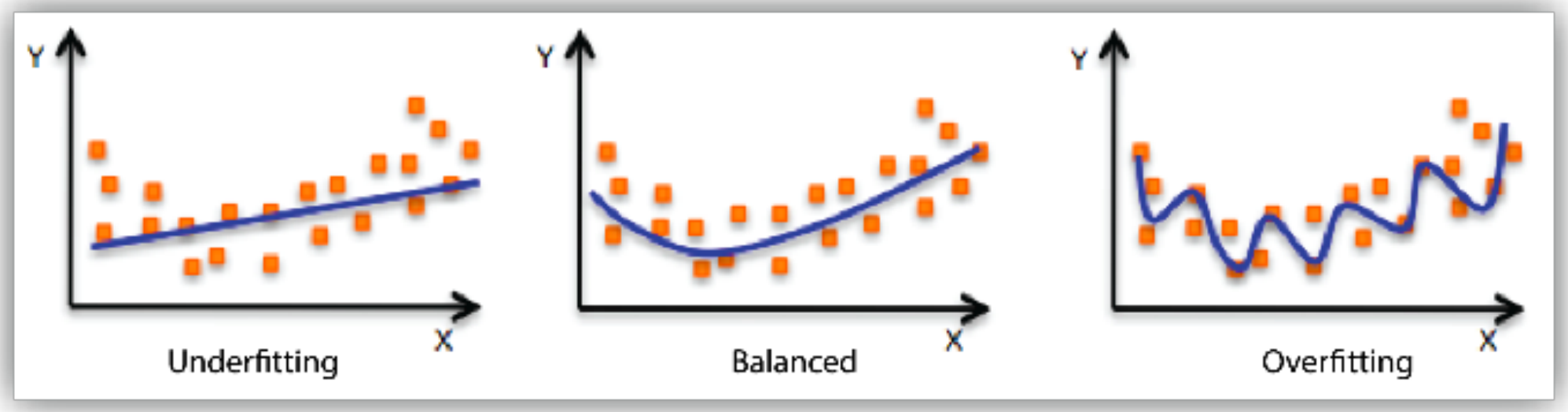
- Generalization
- Under-fitting vs. over-fitting
- Cross validation
- Bias-variance tradeoff
- Bootstrapping
- Bagging and boosting

Generalization

- How well the learned model will behave on unseen data.



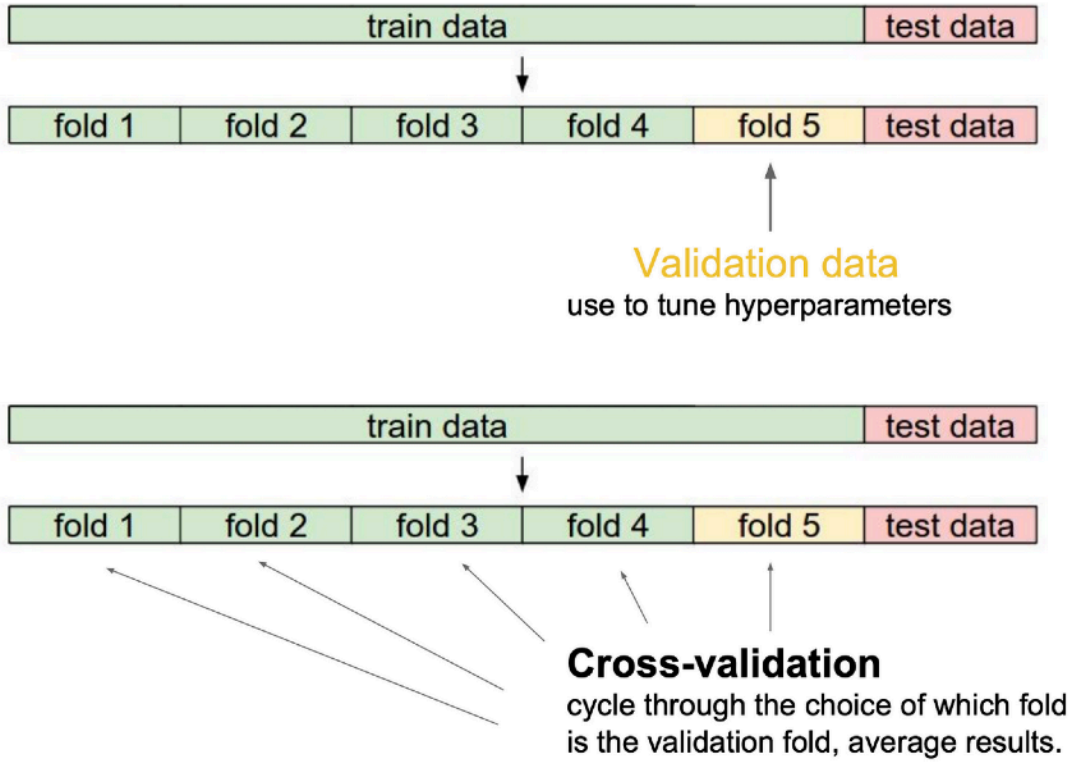
Underfitting vs. Overfitting



https://docs.aws.amazon.com/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html

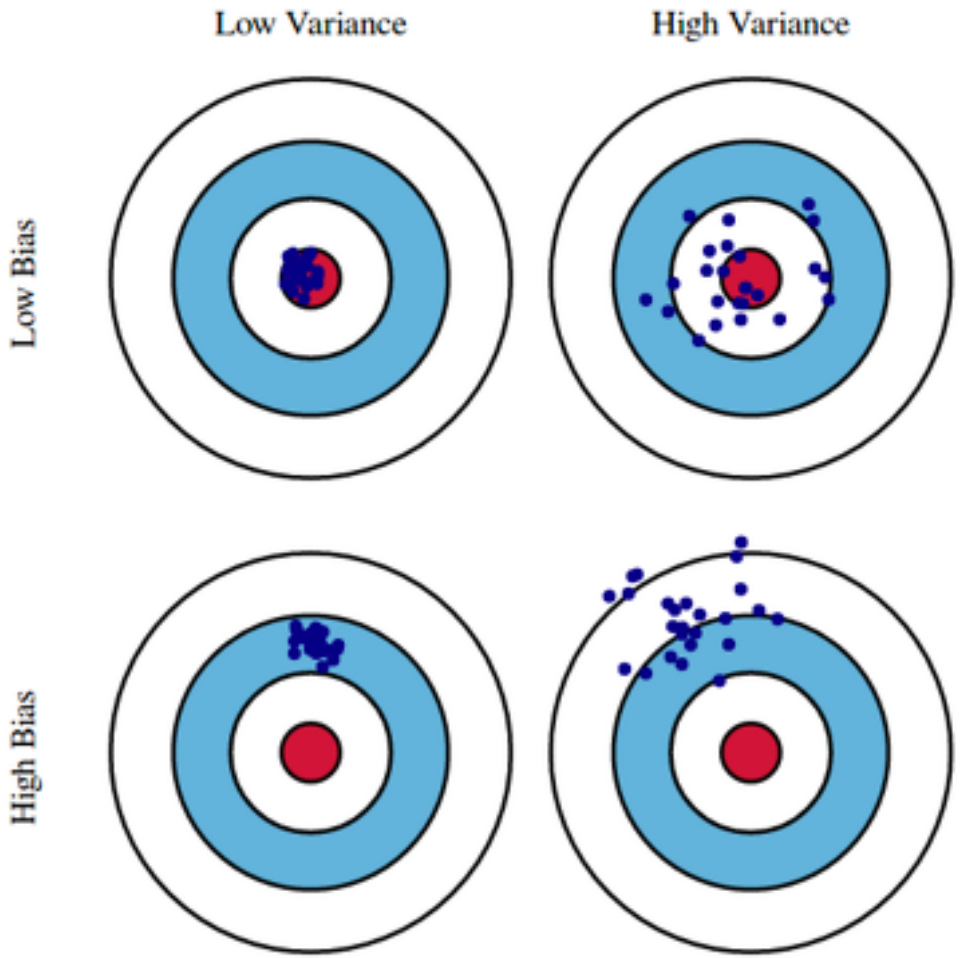
Cross-validation

- Cross-validation is a model validation technique for assessing how the model will generalize to an independent (test) data set.



https://blog.quantinsti.com/cross-validation-machine-learning-trading-models/

Bias and Variance



https://work.caltech.edu/telecourse

Bias and Variance Tradeoff

$$\text{Given } \mathcal{D} = \{(x_i, t_i)\}_{i=1}^{N}, \text{ where } t = f(x) + \epsilon \text{ and } \epsilon \sim \mathcal{N}(0, \sigma^2)$$

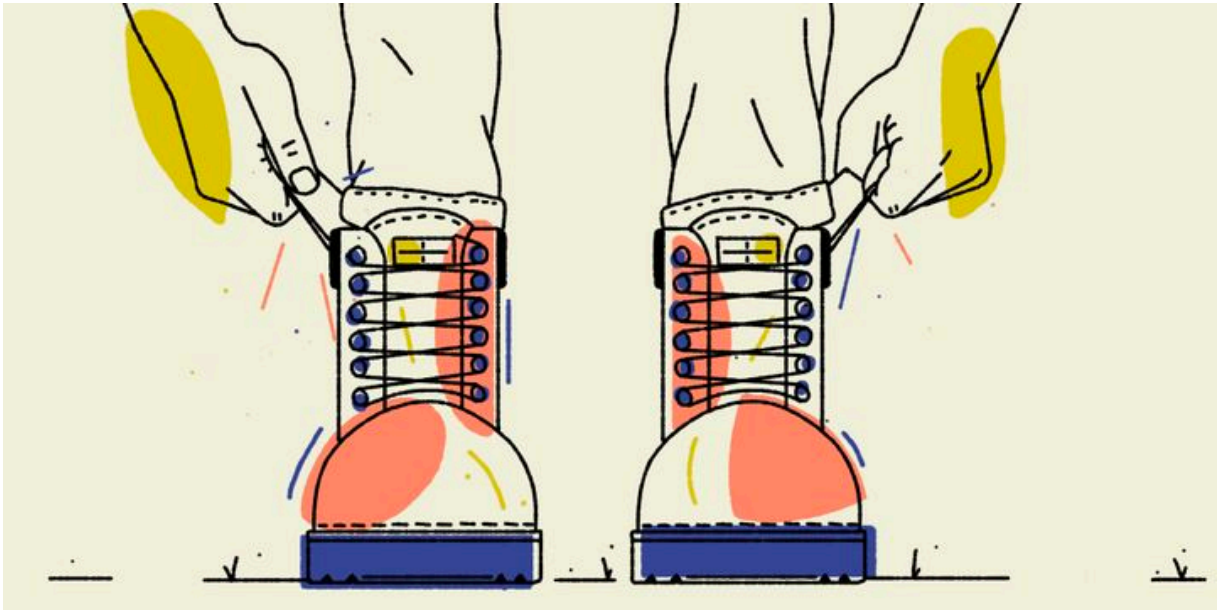
We can derive that what we are minimizing (**cost**) can be decomposed into three different parts: **bias²**, **variance**, and **noise**.

$$\underbrace{\mathbb{E}\left[(t - \hat{f})^2\right]}_{\textbf{cost}} = \mathbb{E}\left[(t - f + f - \hat{f})^2\right]$$

$$= \dots$$

$$= \underbrace{\mathbb{E}\left[(f - \mathbb{E}[\hat{f}]^2)^2\right]}_{\textbf{bias}^2} + \underbrace{\mathbb{E}\left[(\mathbb{E}[\hat{f}] - \hat{f})^2\right]}_{\textbf{variance}} + \underbrace{\mathbb{E}\left[\epsilon\right]}_{\textbf{noise}}$$

Bootstrapping



Pull yourself up by the **bootstraps**.

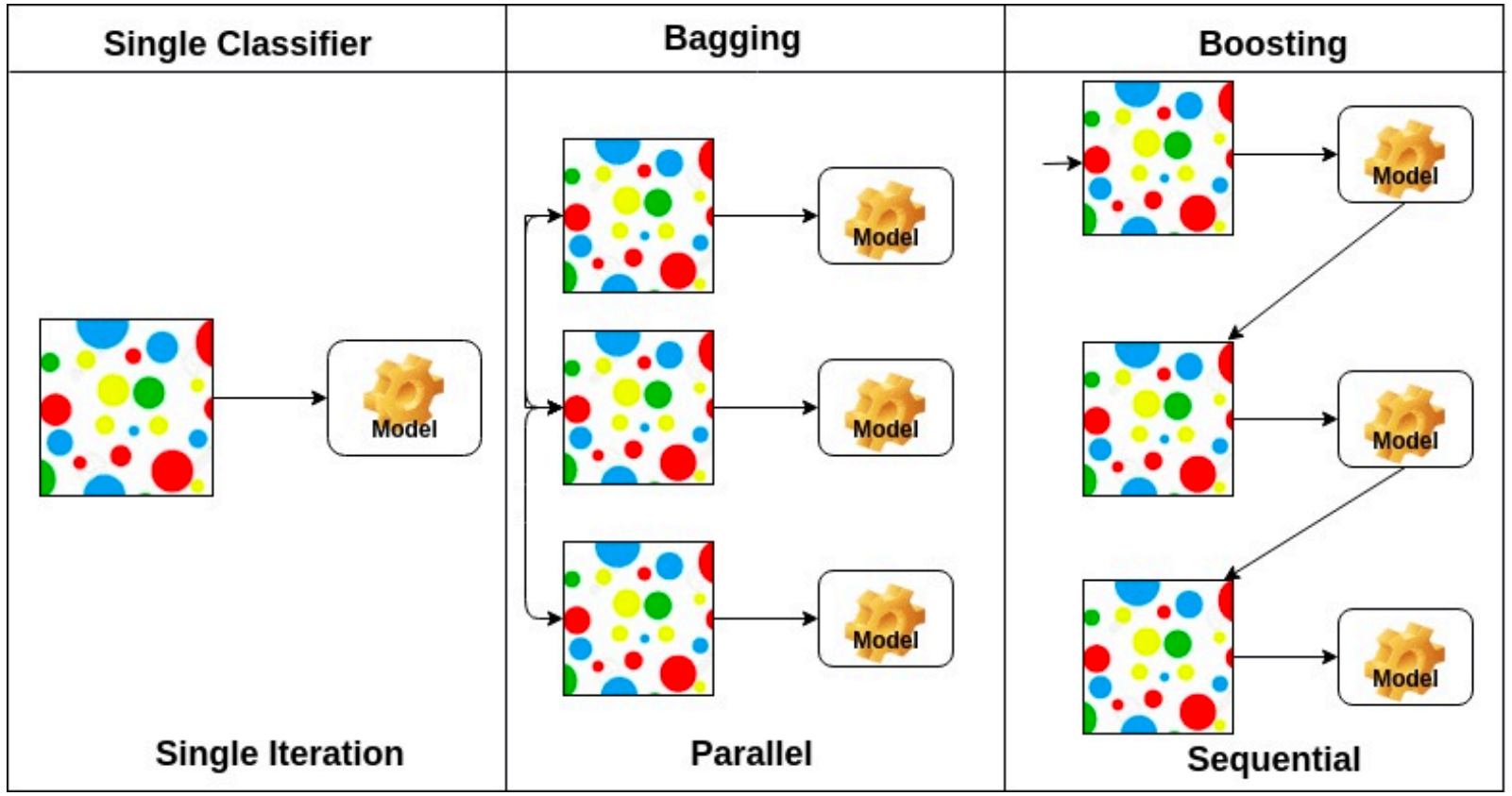
- Bootstrapping is any test or metric that uses random sampling with replacement.

Bagging vs. Boosting

- Bagging (Bootstrapping aggregating)
  - Multiple models are being trained with bootstrapping.
  - ex) Base classifiers are fitted on random subset where individual predictions are aggregated (voting or averaging).

- Boosting
  - It focuses on those specific training samples that are hard to classify.
  - A strong model is built by combining weak learners in sequence where each learner learns from the mistakes of the previous weak learner.

Bagging vs. Boosting



https://www.datacamp.com/community/tutorials/adaboost-classifier-python

Practical Gradient Descent Methods

Gradient Descent Methods

- Stochastic gradient descent
  - Update with the gradient computed from a single sample.
- Mini-batch gradient descent
  - Update with the gradient computed from a subset of data.
- Batch gradient descent
  - Update with the gradient computed from the whole data.

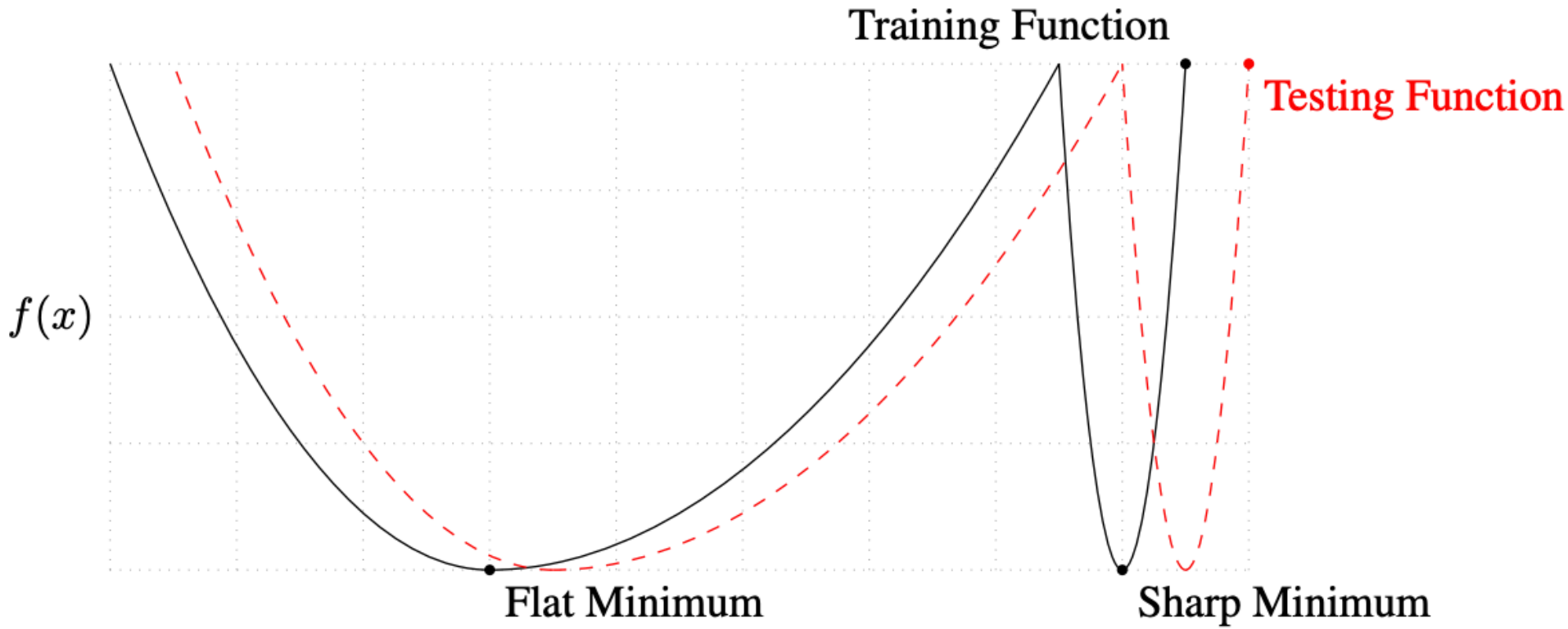
Batch-size Matters

- "It has been observed in practice that when using a larger batch there is a degradation in the quality of the model, as measured by its ability to generalize."

- "We ... present numerical evidence that supports the view that large batch methods tend to converge to **sharp minimizers** of the training and testing functions. In contrast, small-batch methods consistently converge to **flat minimizers**... this is due to the inherent noise in the gradient estimation."

On Large-batch Training for Deep Learning: Generalization Gap and Sharp Minima, 2017

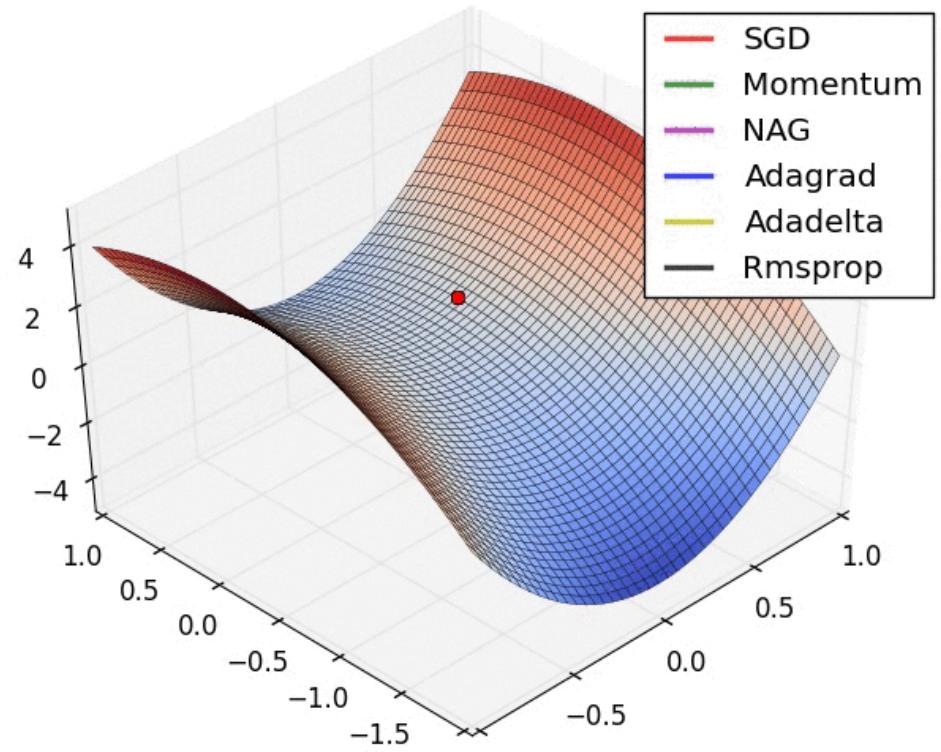
Batch-size Matters



On Large-batch Training for Deep Learning: Generalization Gap and Sharp Minima, 2017

Gradient Descent Methods

- Stochastic gradient descent
- Momentum
- Nesterov accelerated gradient
- Adagrad
- Adadelta
- RMSprop
- Adam



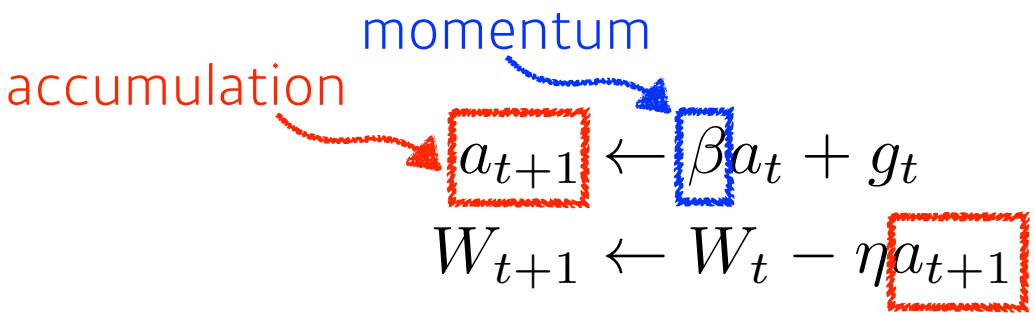
Gradient Descent

$$W_{t+1} \leftarrow W_t - \eta g_t$$

Learning rate

Gradient

Momentum

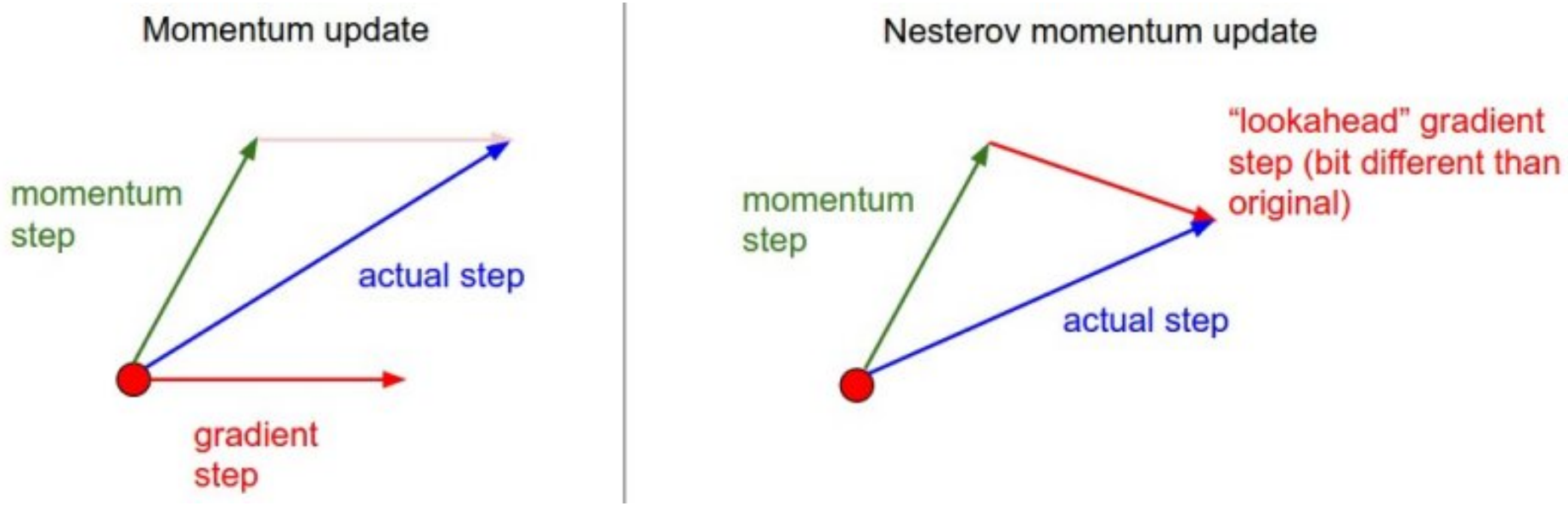


Nesterov Accelerated Gradient

Lookahead gradient

$$a_{t+1} \leftarrow \beta a_t + \nabla \mathcal{L}(W_t - \eta \beta a_t)$$
$$W_{t+1} \leftarrow W_t - \eta a_{t+1}$$

Nesterov Accelerated Gradient



https://golden.com/wiki/Nesterov_momentum

Adagrad

- **Adagrad** adapts the learning rate, performing larger updates for infrequent and smaller updates for frequent parameters.

$$W_{t+1} = W_t - \frac{\eta}{\sqrt{G_t + \epsilon}} g_t$$

for numerical stability

Sum of gradient squares

What will happen if the training occurs for a long period?

Adadelta

- **Adadelta** extends **Adagrad** to reduce its monotonically decreasing the learning rate by restricting the accumulation window.

EMA of gradient squares

$$G_t = \gamma G_{t-1} + (1-\gamma) g_t^2$$

$$W_{t+1} = W_t - \frac{\sqrt{H_{t-1} + \epsilon}}{\sqrt{G_t + \epsilon}} g_t$$

EMA of difference squares

$$H_t = \gamma H_{t-1} + (1-\gamma)(\Delta W_t)^2$$

There is **no learning rate** in Adadelta.

RMSprop

- **RMSprop** is an unpublished, adaptive learning rate method proposed by Geoff Hinton in his lecture.

EMA of gradient squares

$$G_t = \gamma G_{t-1} + (1-\gamma) g_t^2$$

$$W_{t+1} = W_t - \frac{\eta}{\sqrt{G_t + \epsilon}} g_t$$

stepsize

Adam

- Adaptive Moment Estimation (Adam) leverages both past gradients and squared gradients.

Momentum

EMA of gradient squares

Stepsize

$$m_t = \beta_1 m_{t=1} + (1-\beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

$$W_{t+1} = W_t - \frac{\eta}{\sqrt{v_t + \epsilon}} \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t} m_t$$

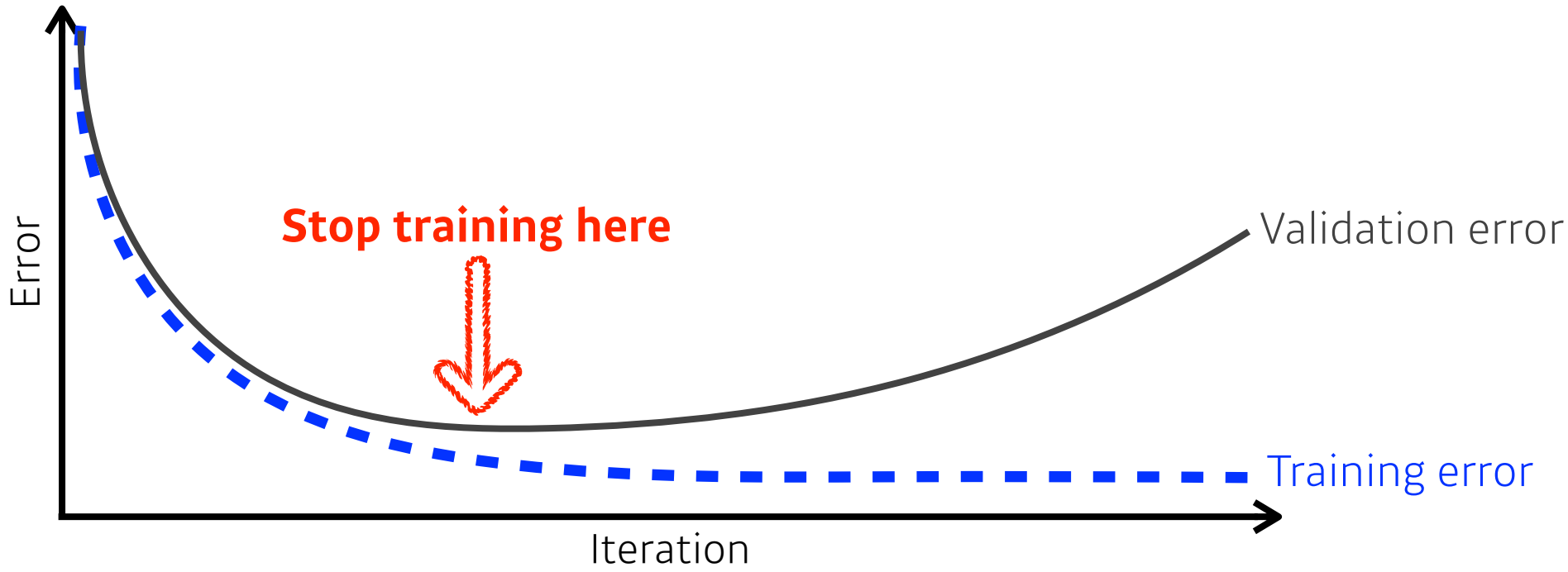
Adam effectively combines momentum with adaptive learning rate approach.

Regularization

Regularization

- Early stopping
- Parameter norm penalty
- Data augmentation
- Noise robustness
- Label smoothing
- Dropout
- Batch normalization

Early Stopping



Note that we need **additional validation data** to do early stopping.

Parameter Norm Penalty

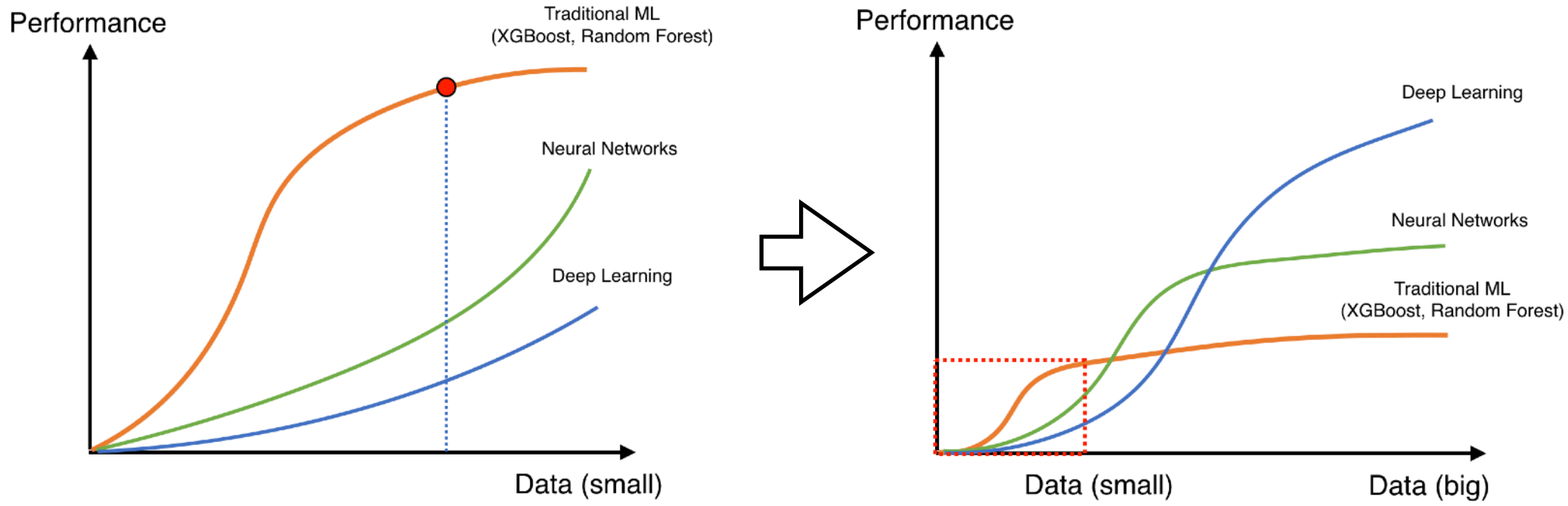
- It adds smoothness to the function space.

Parameter Norm Penalty

$$\text{total cost} = \text{loss}(\mathcal{D}; W) + \frac{\alpha}{2}\|W\|_2^2$$

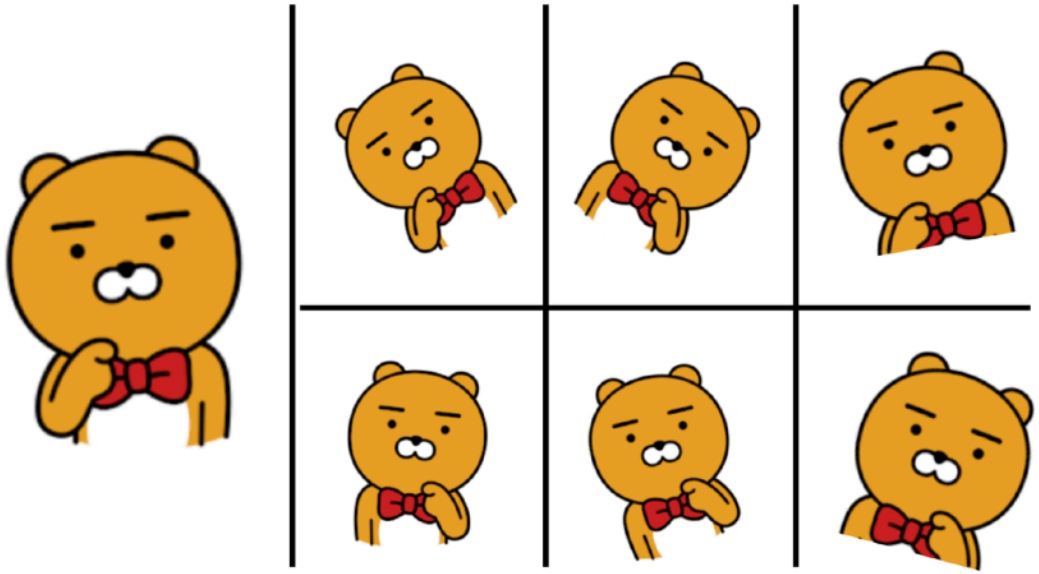
Data Augmentation

- More data are always welcomed.



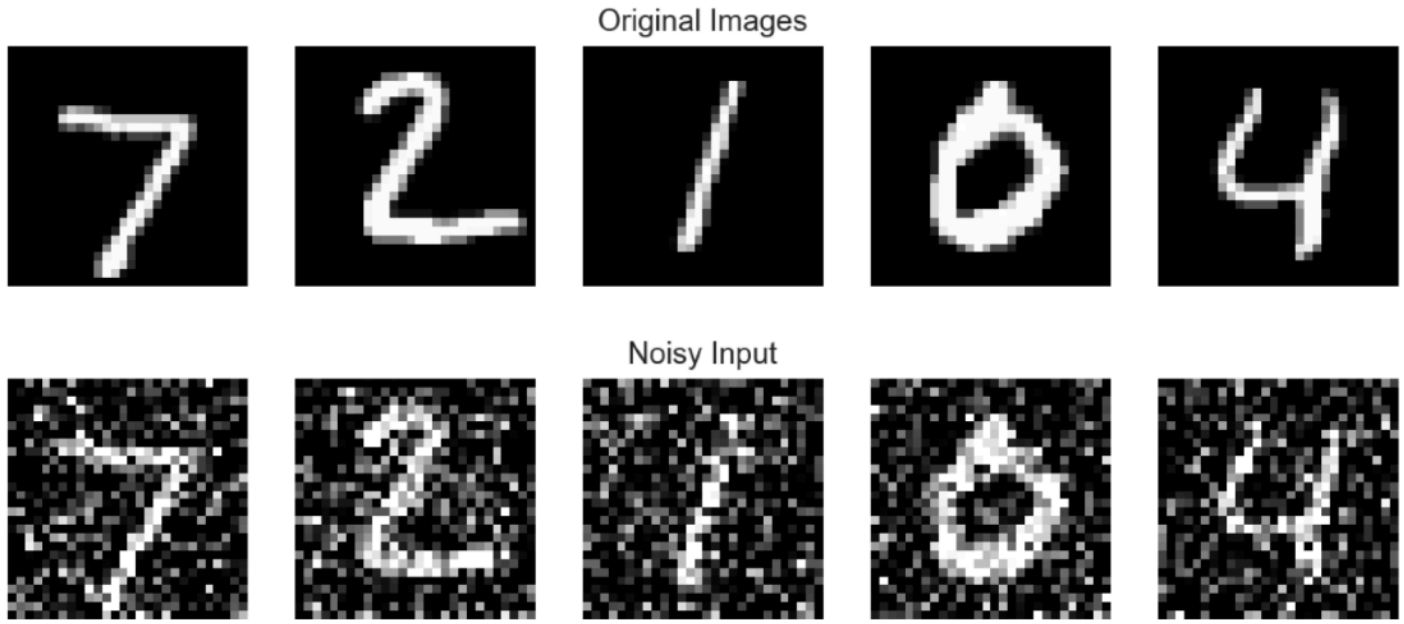
Data Augmentation

- More data are always welcomed.
- However, in most cases, training data are given in advance.
- In such cases, we need data augmentation.



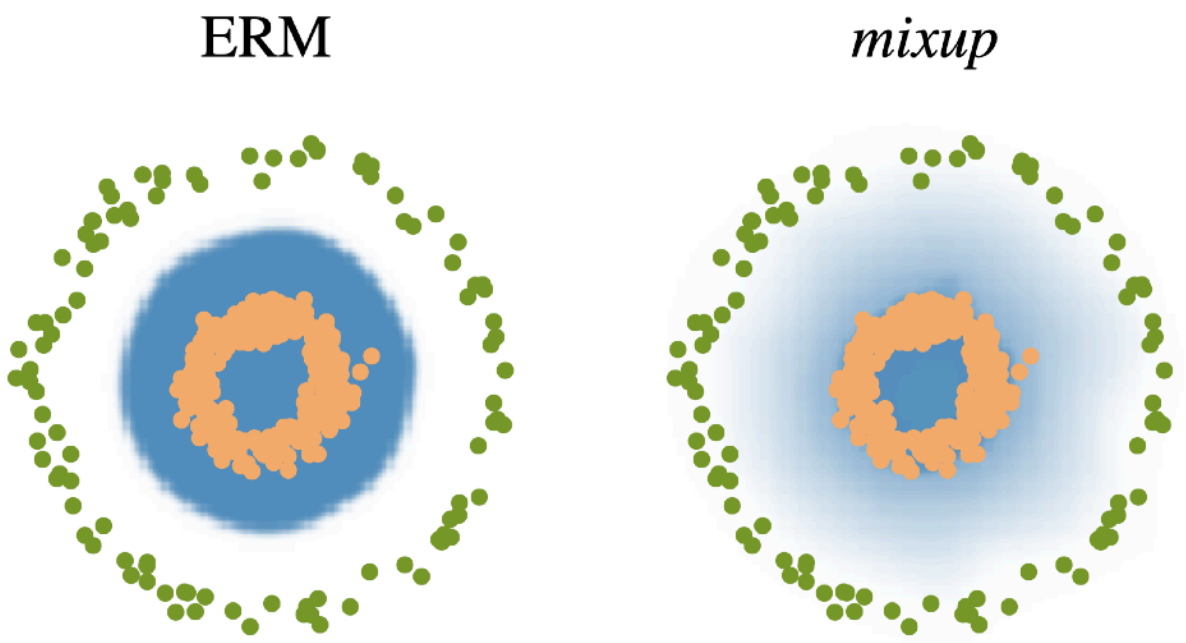
Noise Robustness

- Add random noises inputs or weights.



Label Smoothing

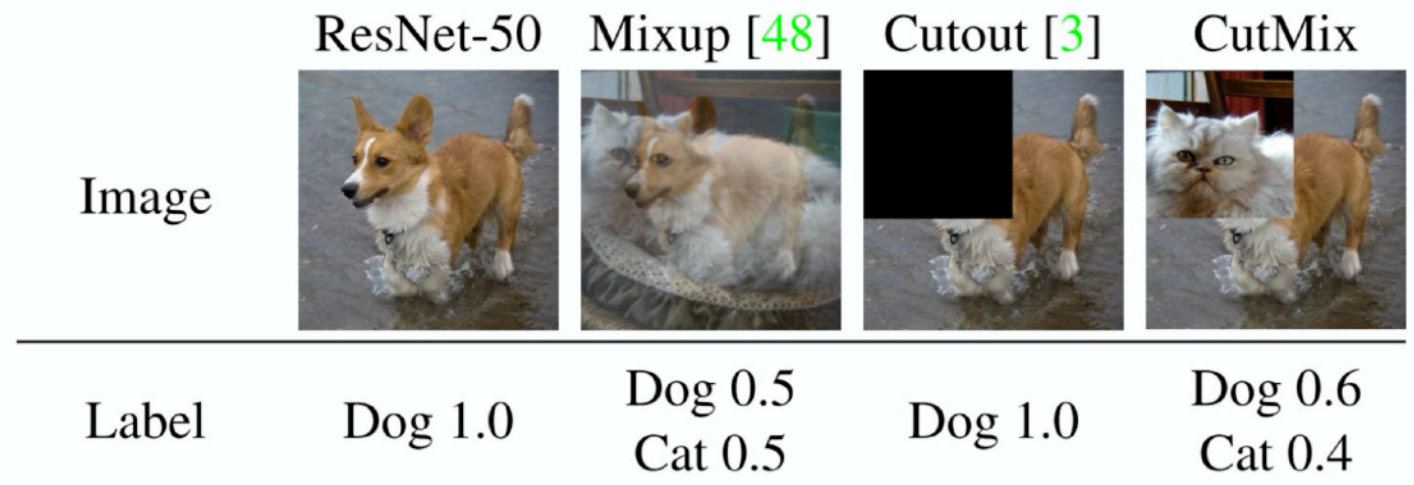
- **Mix-up** constructs augmented training examples by mixing both input and output of two randomly selected training data.



mixup: Beyond Empirical Risk Minimization, 2018

Label Smoothing

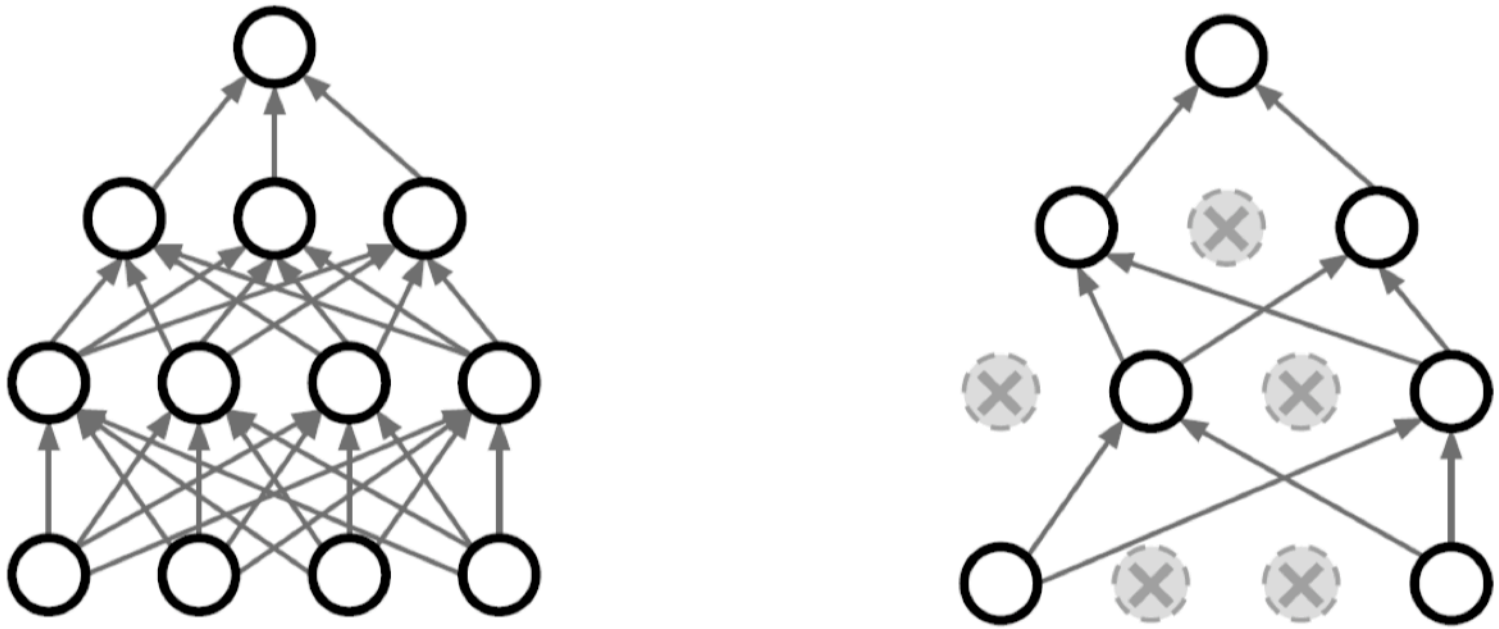
- **CutMix** constructs augmented training examples by mixing inputs with cut and paste and outputs with soft labels of two randomly selected training data.



CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features, 2019

Dropout

- In each forward pass, randomly set some neurons to zero.



Batch Normalization

- Batch normalization compute the empirical mean and variance independently for each dimension (layers) and normalize.

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

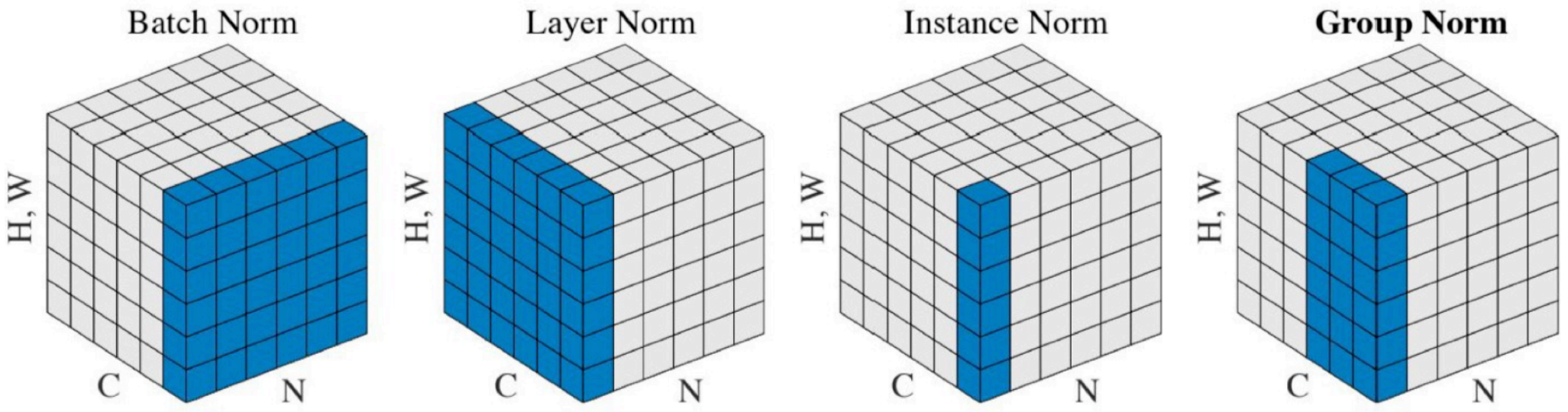
$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

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

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, 2015

Batch Normalization

- There are different variances of normalizations.



Group Normalization, 2018

Thank you for listening