Deep Learning Issues

Challenges in Deep Learning

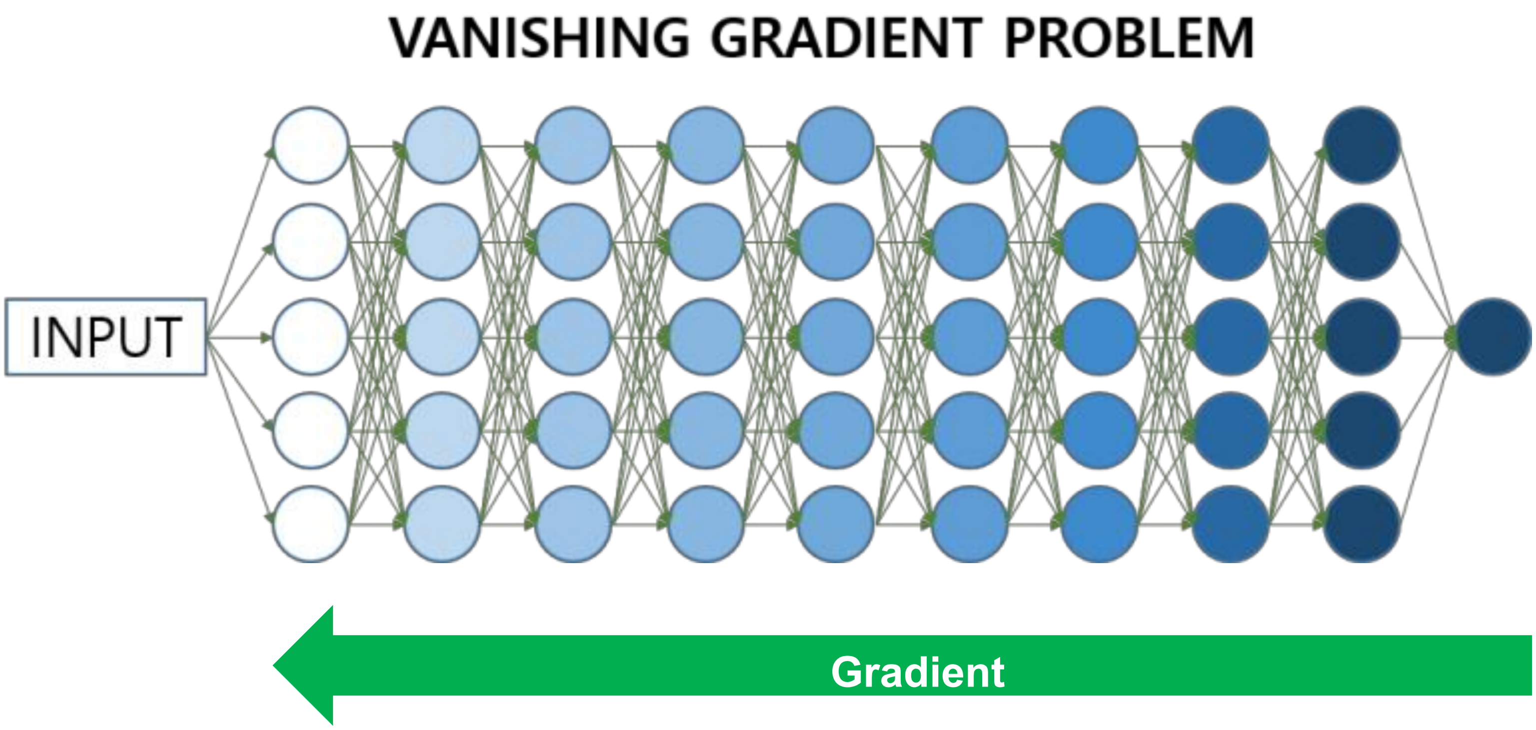
- Difficulties in deep learning
  - Backpropagation algorithm does not work or slow
  - Not better than shallow networks

- Why?
  - The vanishing/exploding gradient problem
  - Local minima, saddle points, plateaus
  - Overfitting
  - Internal covariate shift [Ioffe15]
  - Scattered gradient problem [Balduzzi17]
  - Many unknown reasons

Vanishing Gradient Problem

- Conventional back-propagation algorithm does not work well for deep networks.

⇒ Gradient가 0이되면
아래층에 학습이 잘안된다.



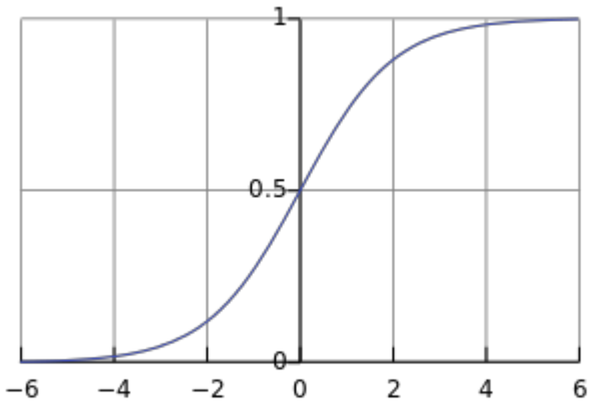
Vanishing Gradient Problem

- Conventional back-propagation algorithm does not work well for deep networks.
  - Backpropagation formula

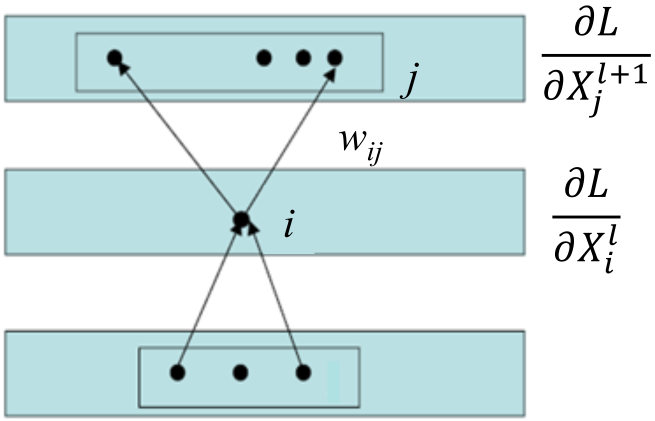
$$\frac{\partial L}{\partial X_i^l} = \underbrace{\frac{\partial L}{\partial X_i^{l+1}} \frac{\partial X_i^{l+1}}{\partial X_i^l}}_{\text{Chaine rule}} = \underline{f'(net_j^{l+1}) \sum_j w_{ij}^{l+1} \frac{\partial L}{\partial X_j^{l+1}}}$$

CNN은
vanishing
gradient가 존재

Saturated regime of
Activation functions



Blended gradient



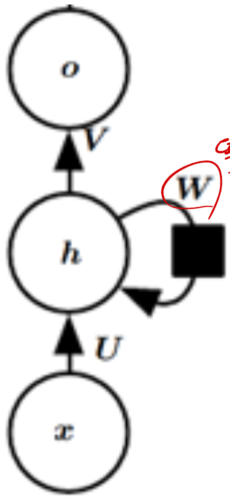
Vanishing/Exploding Gradient on RNN

- One step propagation on RNN

$$h^{(t)} = W^\top h^{(t-1)}$$

$$h^{(t)} = \left(W^t\right)^\top h^{(0)}$$

↳ W가 1이상이면 ∞, 1미만이면 0으로수렴



똑같은 W를 계속 곱해주기 때문이다

- Eigen decomposition of W

$$W = Q\Lambda Q^\top$$

$$h^{(t)} = Q^\top \Lambda^t Q h^{(0)}$$

- Gradients propagated over many stages tend to either vanish ($|\lambda_i|<1$) or explode ($|\lambda_i|>1$)

Solutions of Vanishing Gradient Problem

- Layer-wise unsupervised pre-training
  - DBN, stacked auto-encoders

- Architectures to avoid vanishing gradient problem
  - Convolutional neural networks (CNN)
    - Sparse connection, shared weights
  - Gated units (LSTM, GRU, GLU)

- Improved structures and learning algorithms
  - Piece-wise linear activation functions
    - max-out, ReLU, LReLU, PReLU, ELU, etc.…
  - Skip connection (ResNet, DenseNet, DPN)

    → MS에서 만듦. ↳ layer를 50개 100개 이상 사용하게 됨.

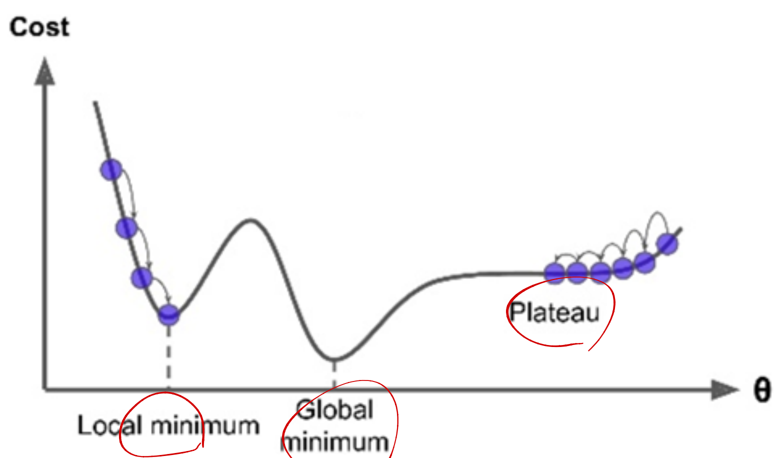
  - Batch normalization
  - Xavier initialization, He initialization, LL-initialization

    → 입력분산과 출력분산이 유지되어야 한다?
    ① 더 커지면 불안정해짐
    ② 더 작아지면 의미가 없어짐

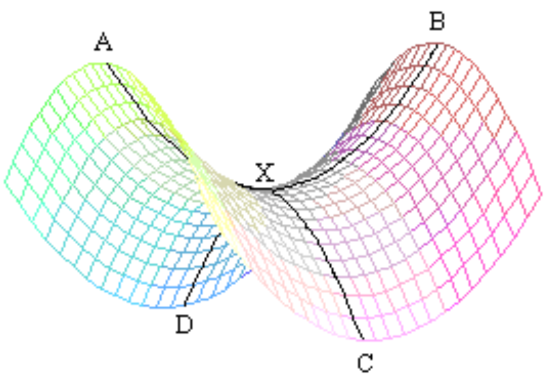
  - Transfer learning, multi-task learning
  - Auxiliary networks, deeply supervised network

Local Minima, Saddle Point, Plateau

- Learning sometimes stops at local minima, saddle points or plateaus
  - Small networks: local minima is major issues
  - Large networks: plateau or saddle points are major issues



Local minima / Plateau

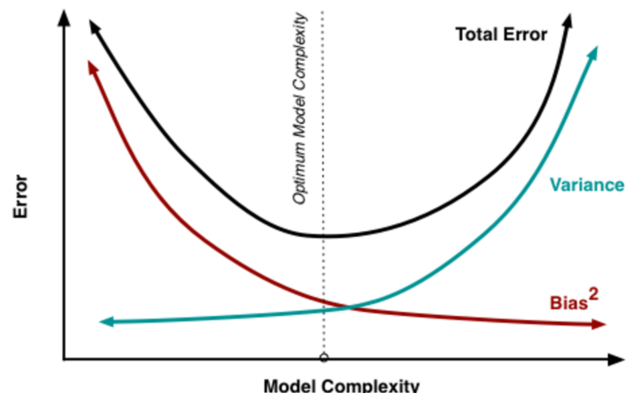


Saddle point

안장점

Overfitting

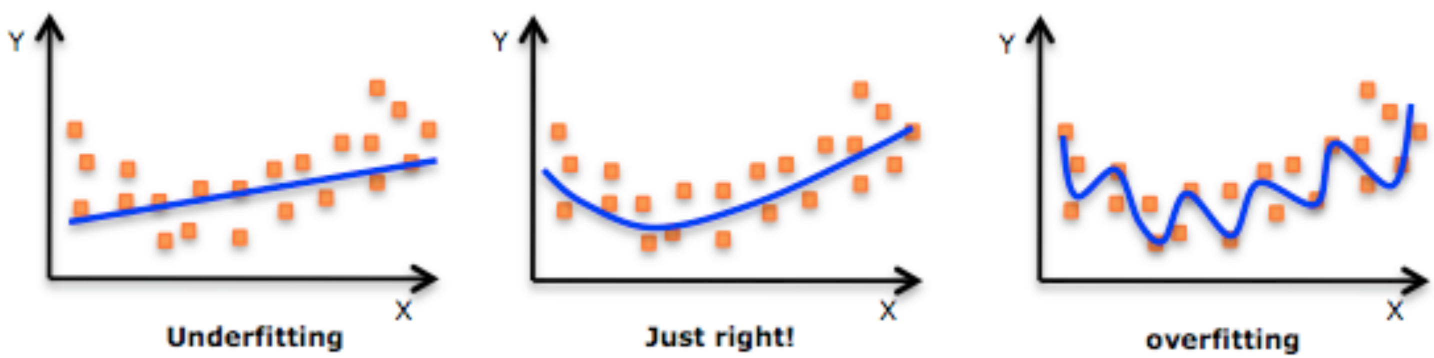
- Overfitting occurs when a model is excessively complex relative to the number of observations.
  - Large capacity model + insufficient data



$\boxed{\text{deep network의 depth가 길어질수록}}$

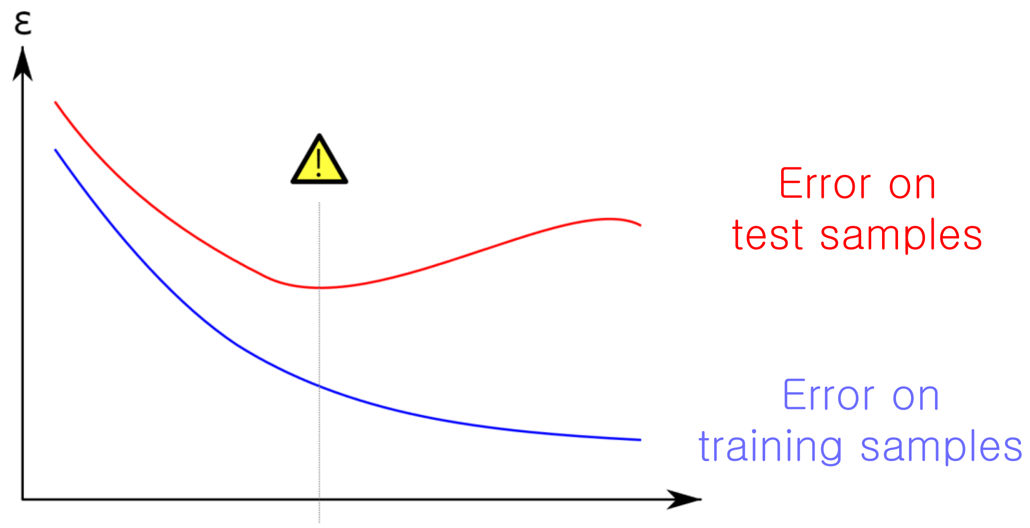
⇒ $\boxed{\text{학습이 어렵다}}$

⇒ deeplearning을 쓸때에, overfitting 문제가 더 나빠짐



Overfitting

- Large gab between training and test accuracy



- Remedies
  - More data or simpler model
  - Regularization, transfer learning, batch norm, dropout, etc.

Generalization of Deep Networks

- Traditional knowledge
  - Model with too large capacity does not generalize well

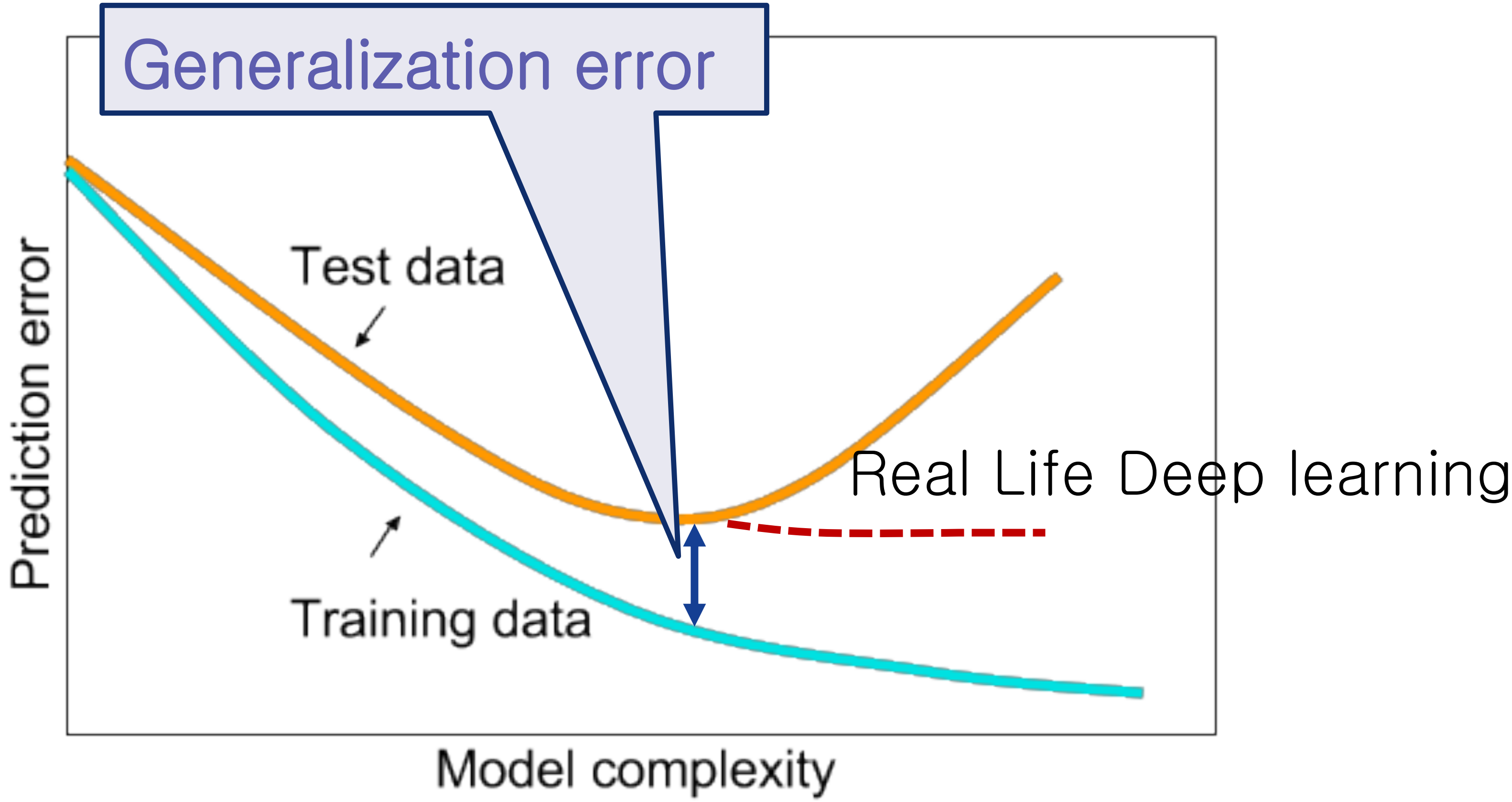
- New observations in deep learning
  - Network depth helps improve generalization
  - Many huge networks generalize well.
    - Train VGG19 (20M parameters) on CIFAR10 (50K samples)

  ➔ Generalization of deep networks is not explainable with conventional knowledge

- Current trend: powerful model + additional techniques
  - Regularization techniques
  - Data augmentation 변형 ~> 추가 Data
  - Unsupervised pretraining / semi-supervised learning

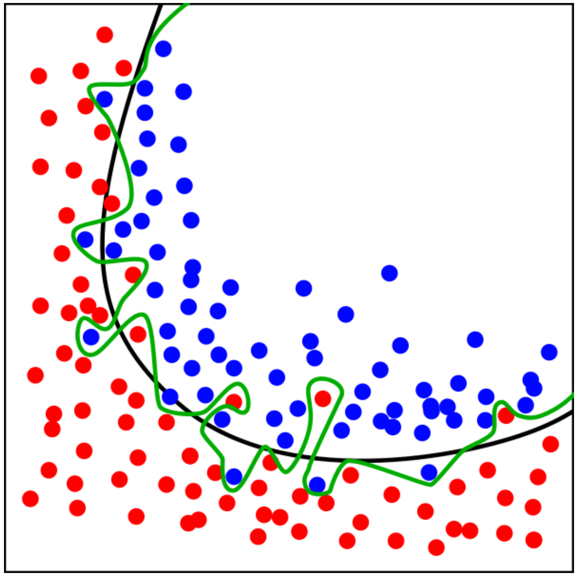
Overfitting in Deep Learning

- In deep learning, over-parameterization is often successful



Regularization

[Wikipedia]



- Introduce additional information to solve ill-posed problems or reduce overfitting

- Add regularization term to loss function

$$E(W) + \lambda \|W\|$$

  - $E(W)$: main loss function
  - λ: regularization factor
  - $\|W\|$: norm of W
    - L2-norm is more popular
    - L1-norm is used for some models (e.g. sparse autoencoder)

- Related topics
  - Support vector machines
  - Prior probability

Q&A

Thank you
for your attention!