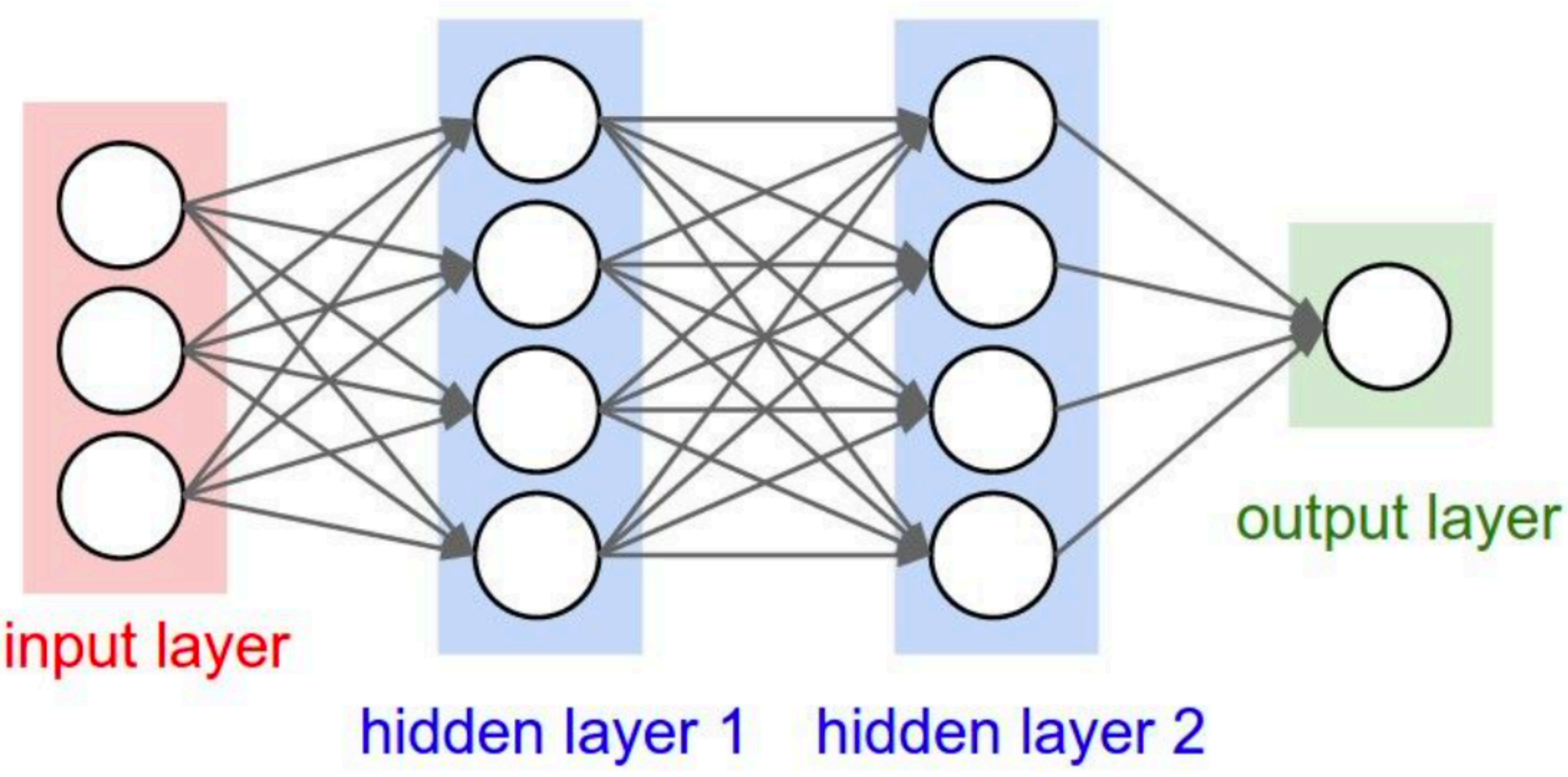
CSE 152: Computer Vision

Hao Su

Lecture 8: Statistical and Optimization Perspectives of Deep Learning

Multi-Layer Perceptron



Why Deep?

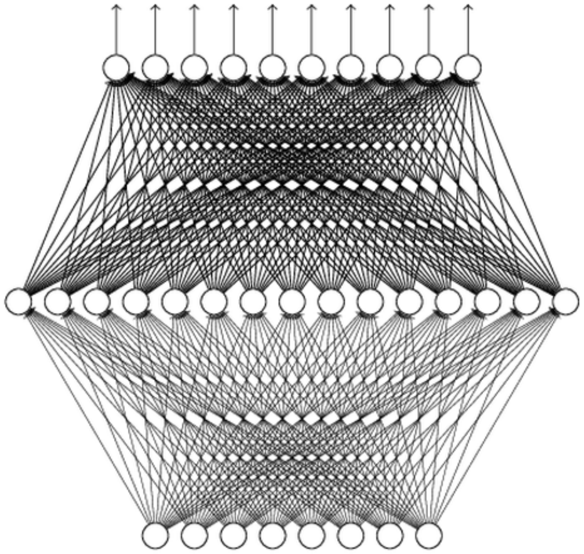
Universality Theorem

Any continuous function f

$$f : R^N \rightarrow R^M$$

Can be realized by a network with one hidden layer

(given **enough** hidden neurons)



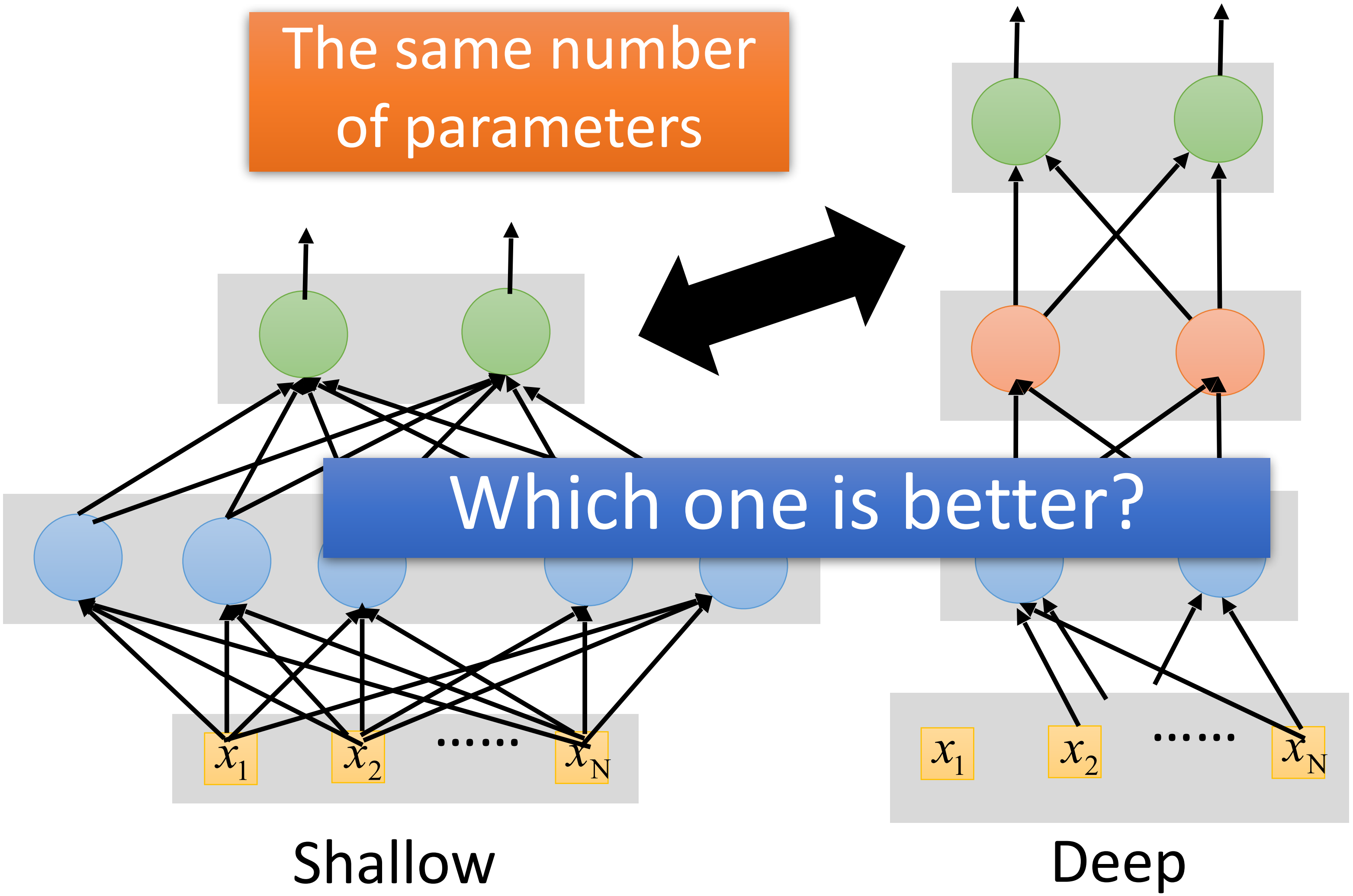
Reference for the reason: http://neuralnetworksanddeeplearning.com/chap4.html

The Unreasonable Effectiveness of Gradient Descent

- While the loss function for neural networks is highly non-convex, empirically (and theoretically), we can show that, with many hidden neurons, the value of local minima are almost as **small** as the global minimum

Then why “Deep” neural network not “Fat” neural network?

Fat + Short v.s. Thin + Tall



Fat + Short v.s. Thin + Tall

"Why deep" is a very "deep" question!

No simple answer yet, even no fully convincing answer yet!

Statistical View of Machine Learning

We start from understanding some simple classifiers,
to draw inspiration for understanding neural networks!

Review: Nearest Neighbor

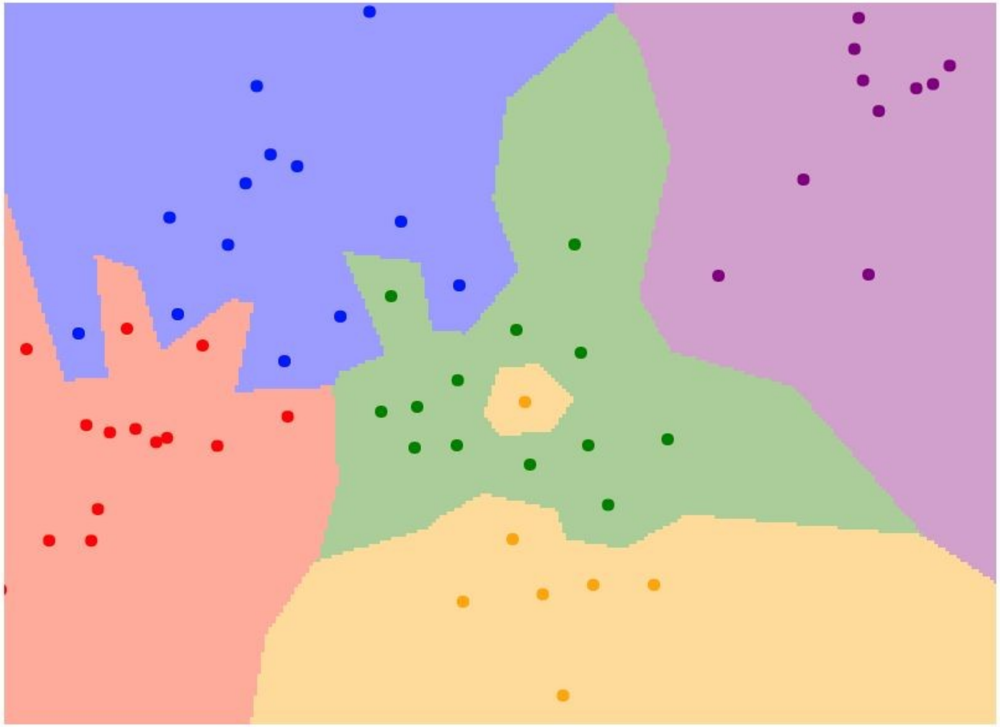
```python
def train(images, labels):
  # Machine learning!
  return model
```

→ Memorize all data and labels

```python
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```

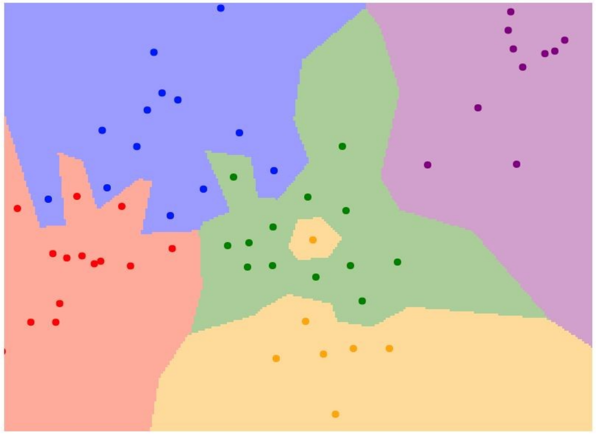
→ Predict the label of the most similar training image

What does Nearest Neighbor look like?

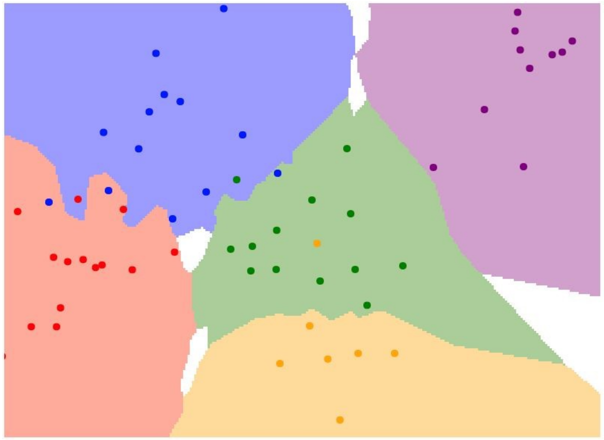


K-Nearest Neighbors

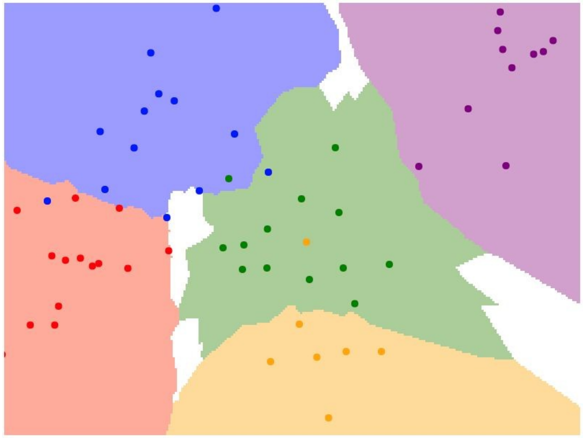
Instead of copying label from nearest neighbor,
take **majority vote** from K closest points



K = 1



K = 3



K = 5

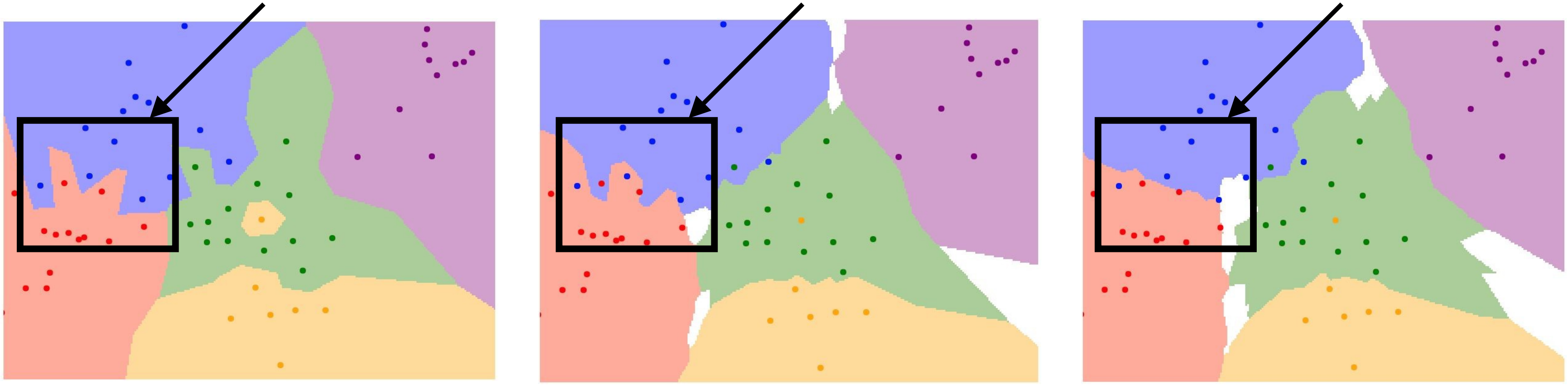
Inductive bias

- What is the best value of k to use? What is the best distance to use?

- These are hyperparameters: choices about the algorithm that we ***set*** rather than ***learn***

- The deep v.s. fat choice for neural networks is similarly a choice of "algorithms"

K-Nearest Neighbors



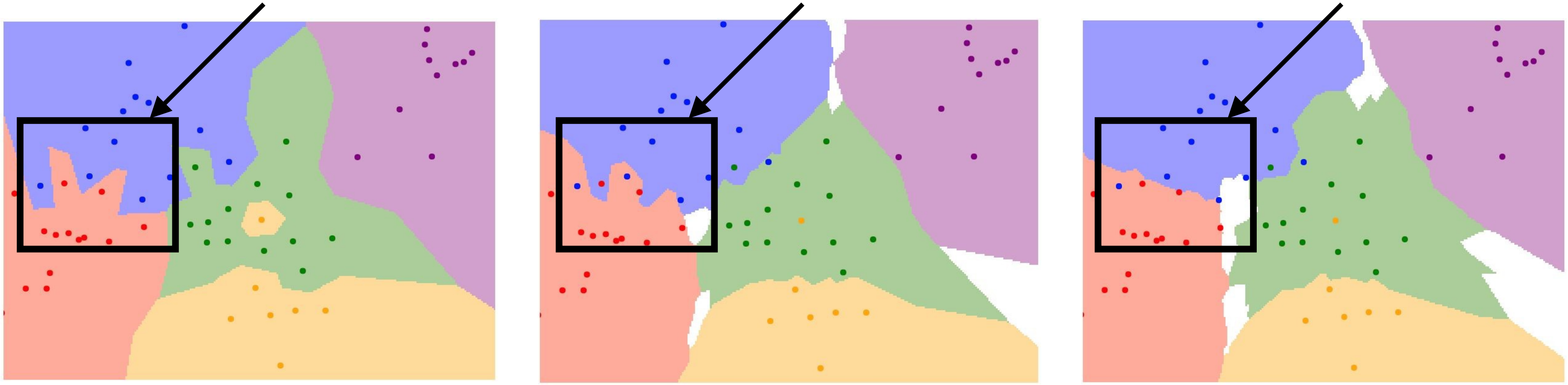
K = 1 K = 3 K = 5

Observations:

Small K (e.g., K=1): every sample matters, sophisticated boundary

Large K (e.g., K=5): voting finds the consensus in the neighborhood, simpler boundary

K-Nearest Neighbors



K = 1 K = 3 K = 5

Observations:

Small K (e.g., K=1): every sample matters, sophisticated boundary, **high model complexity**

Large K (e.g., K=5): voting finds the consensus in the neighborhood, simpler boundary, **low model complexity**

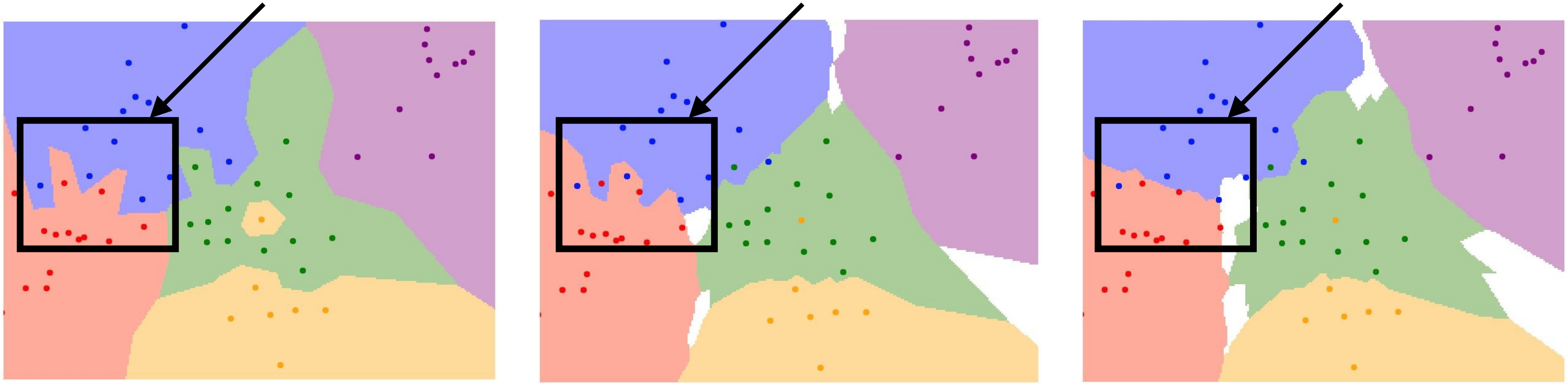
Bias and Variance

- Bias – error caused because the model lacks the ability to represent the (complex) concept

- Variance – error caused because the learning algorithm overreacts to small changes (noise) in the training data

TotalLoss = Bias + Variance (+ noise)

K-Nearest Neighbors



K = 1　　K = 3　　K = 5

Which one has higher bias? higher variance?

- Bias – error caused because the model lacks the ability to represent the (complex) concept
- Variance – error caused because the learning algorithm overreacts to small changes (noise) in the training data

The Power of a Model Building Process

Weaker Modeling Process
(higher bias)

- Simple Model (e.g. linear, large K in KNN)
- Small Feature Set (e.g. few neurons)
- Constrained Search (e.g. few iterations of gradient descent)

More Powerful Modeling Process
(higher variance)

- Complex Model (e.g. networks, small K in KNN)
- Large Feature Set (e.g. many neurons)
- Unconstrained Search (e.g. exhaustive search)

Overfitting v.s. Underfitting

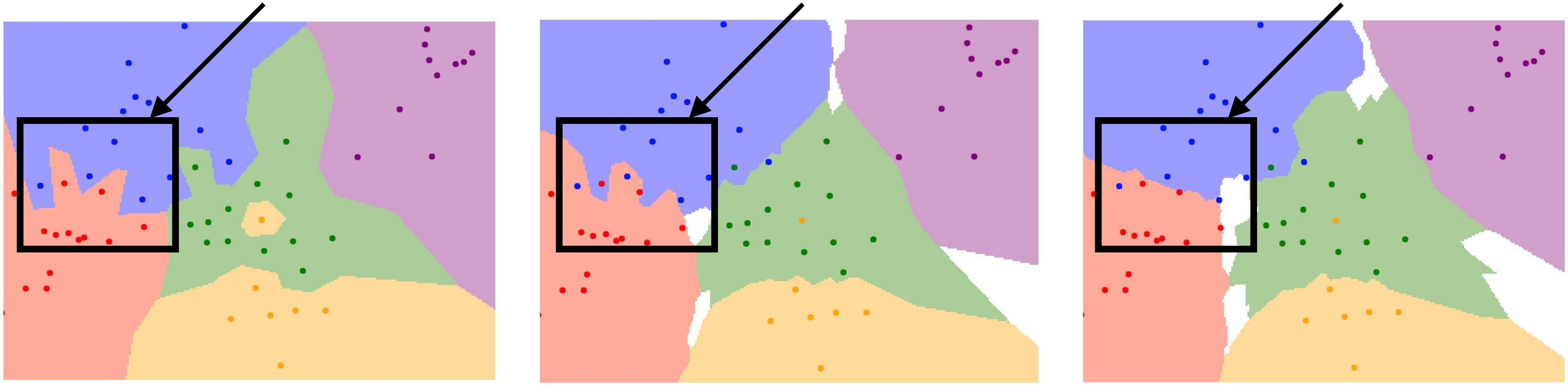
Overfitting

- Fitting the data too well
  - Features are noisy / uncorrelated to concept
  - Modeling process very sensitive (powerful)
  - Too much search

Underfitting

- Learning too little of the true concept
  - Features don't capture concept
  - Too much bias in model
  - Too little search to fit model

K-Nearest Neighbors



K = 1 K = 3 K = 5

Which one tends to overfit? to underfit?

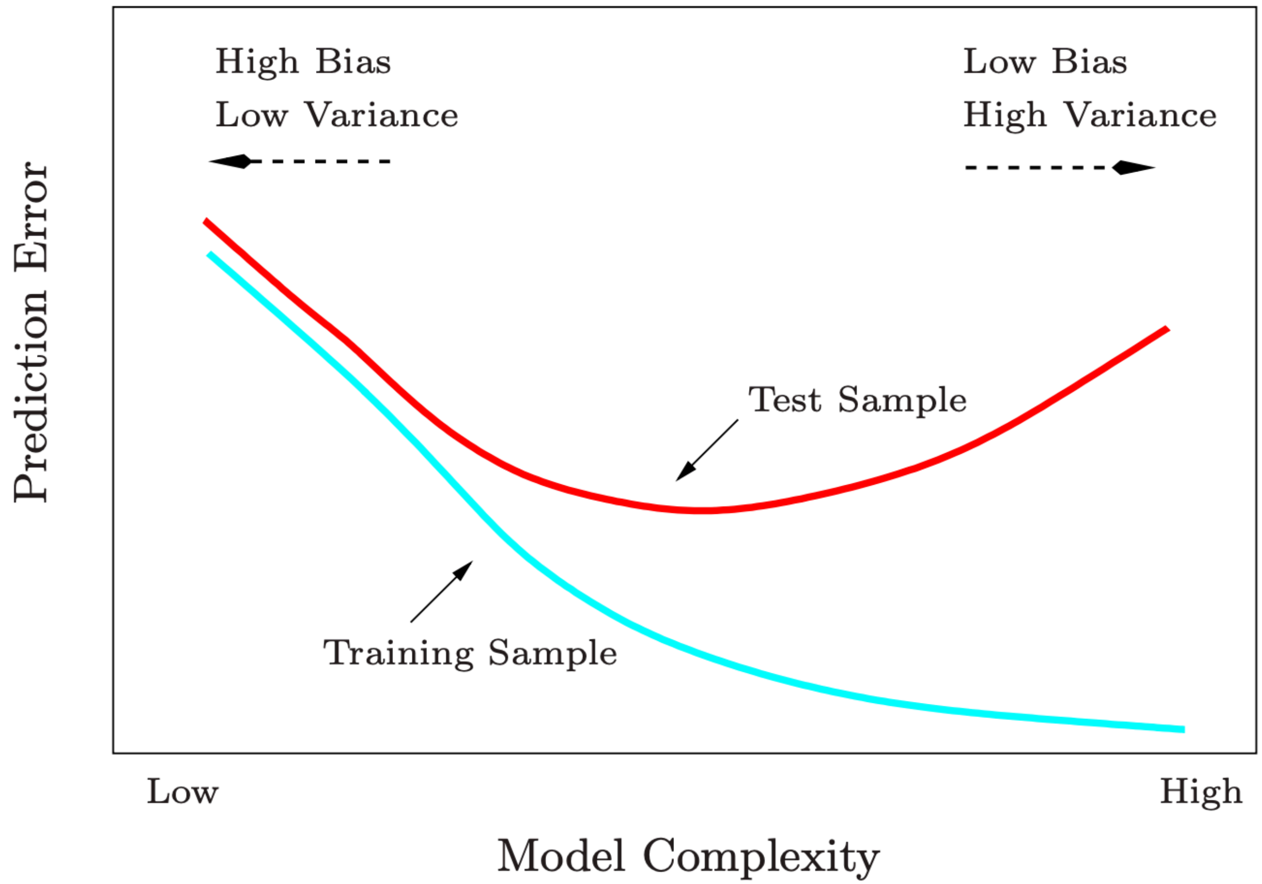


FIGURE 2.11. *Test and training error as a function of model complexity.*

Credit: Elements of Statistical Learning, Second edition

Summary of Overfitting and Underfitting

- Bias / Variance tradeoff a primary challenge in machine learning
- Internalize: More powerful modeling is not always better
- Learn to identify overfitting and underfitting
- Tuning parameters & interpreting output correctly is key

Back to Neural Networks

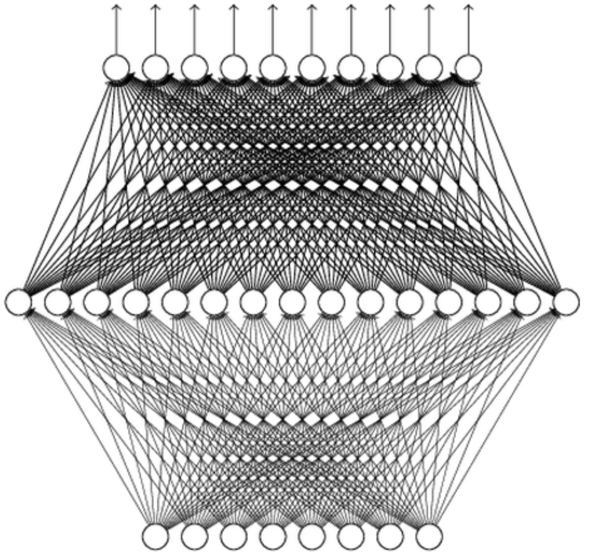
Recap: Universality Theorem

Any continuous function f

$$f : R^N \rightarrow R^M$$

Can be realized by a network with one hidden layer

(given **enough** hidden neurons)



Reference for the reason: http://neuralnetworksanddeeplearning.com/chap4.html

Universality is Not Enough

- Neural network has very high capacity (millions of parameters)
- By our basic knowledge of bias-variance tradeoff, too many parameters should imply very low bias, and very high variance. The test loss may not be small.
- Many efforts of deep learning are about mitigating overfitting!

Address Overfitting for NN

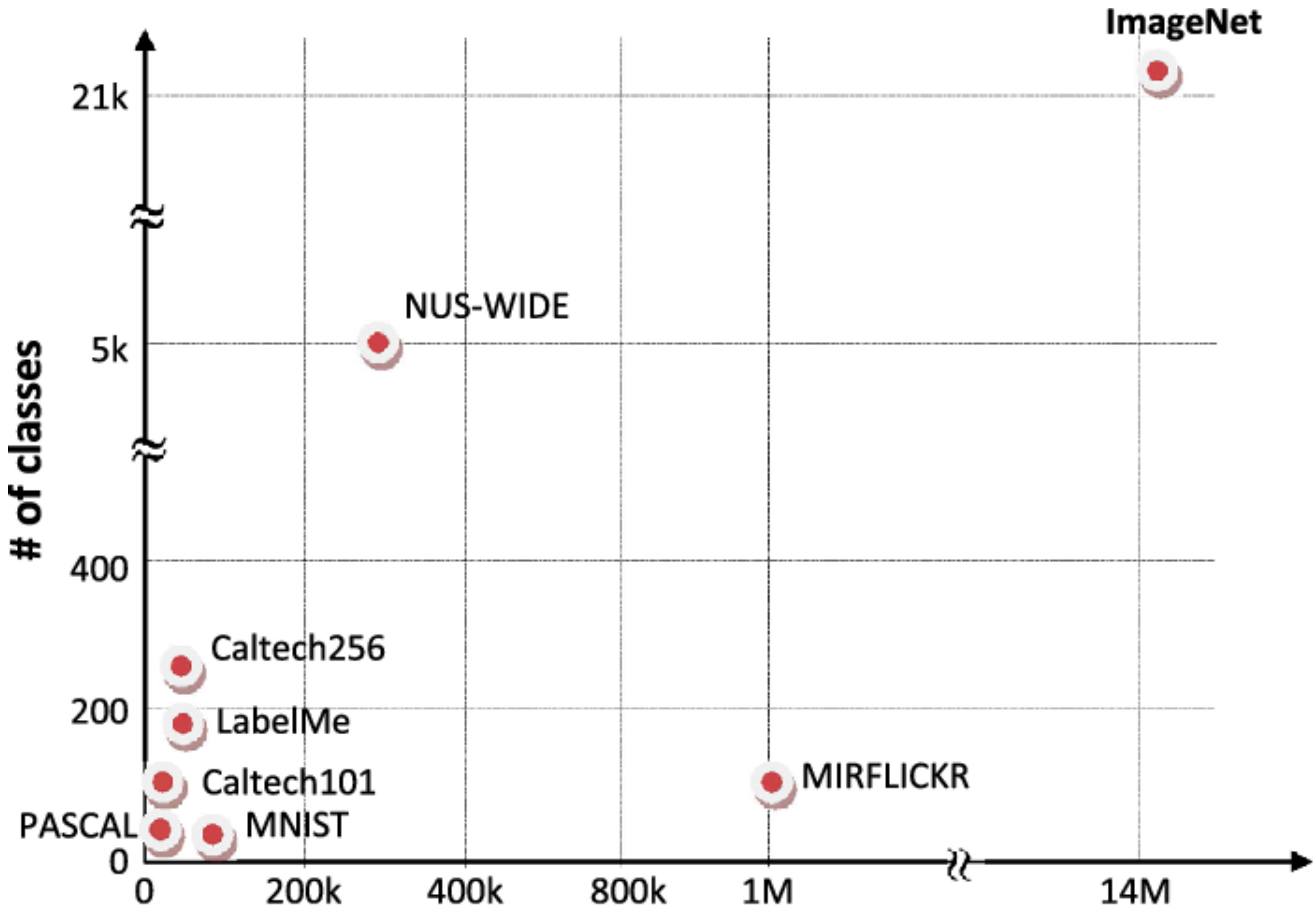
- Use larger training data set

- Design better network architecture

Address Overfitting for NN

- **Use larger training data set**

- Design better network architecture



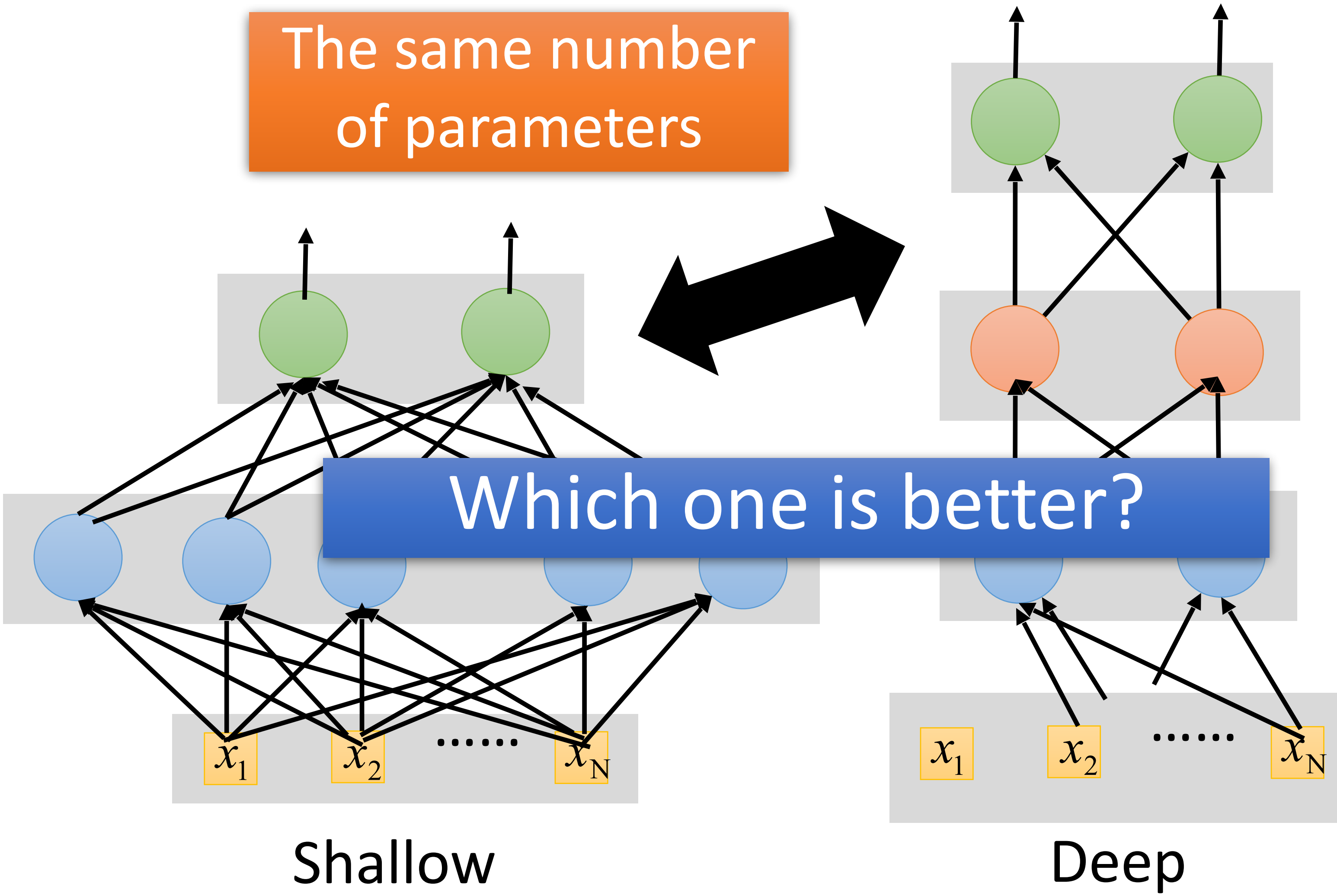
ImageNet Large Scale Visual Recognition Challenge
Russakovsky, Deng, Su, et al. IJCV 2015

Address Overfitting for NN

- **Design better network architecture**

- Use larger training data set

Fat + Short v.s. Thin + Tall

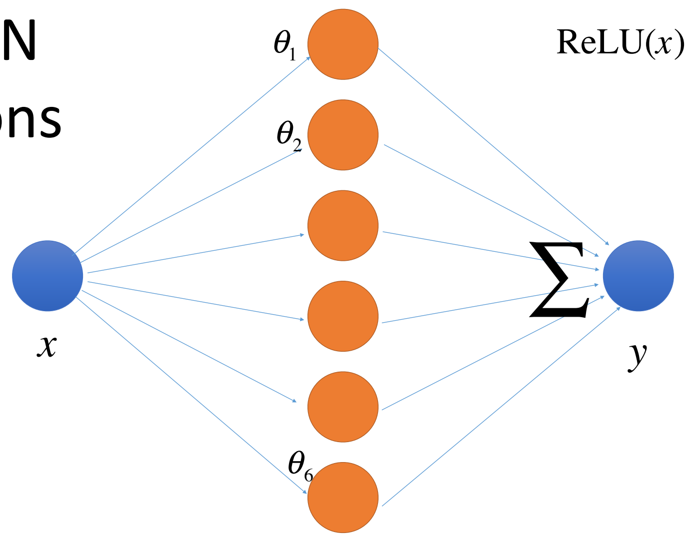


The Intuition behind Deep

- To achieve the same representation power, we can use fewer neurons with a deeper architecture

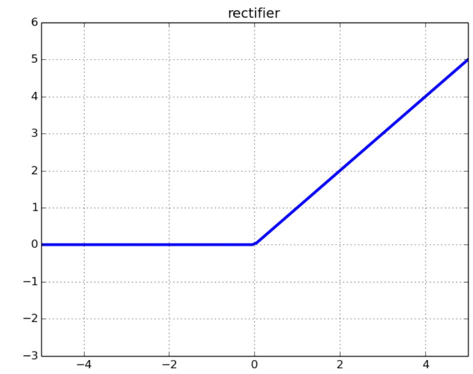
- Fewer neurons risk less for overfitting (lacking rigor for this argument)

Fat + Short NN
With 6 neurons



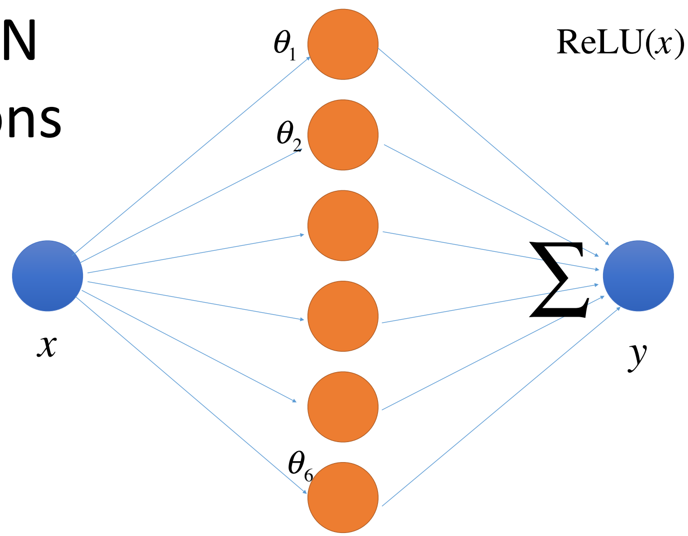
$$a_i = \text{ReLU}(x - \theta_i)$$

$$\text{ReLU}(x) = \begin{cases} 0 \; if \;\; x < 0 \\ x \; if \;\; x >= 0 \end{cases}$$

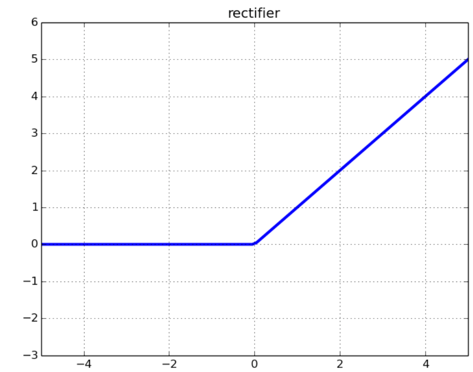


Assume $w_i = 1$ for all neurons, just learn the bias term b_i

Fat + Short NN
With 6 neurons

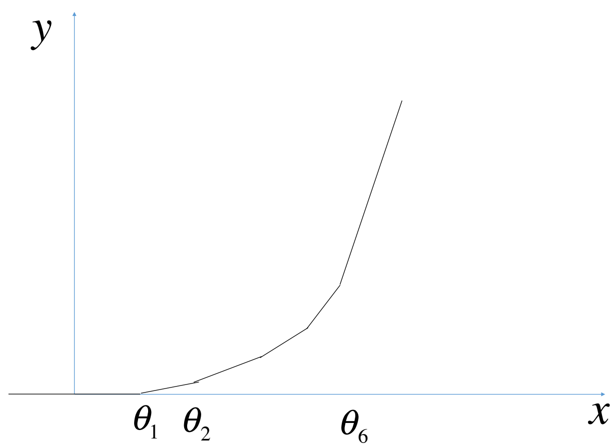


$$\mathrm{ReLU}(x) = \begin{cases} 0 \; if \;\; x < 0 \\ x \; if \;\; x >= 0 \end{cases}$$



$$a_i = \mathrm{ReLU}(x - \theta_i)$$

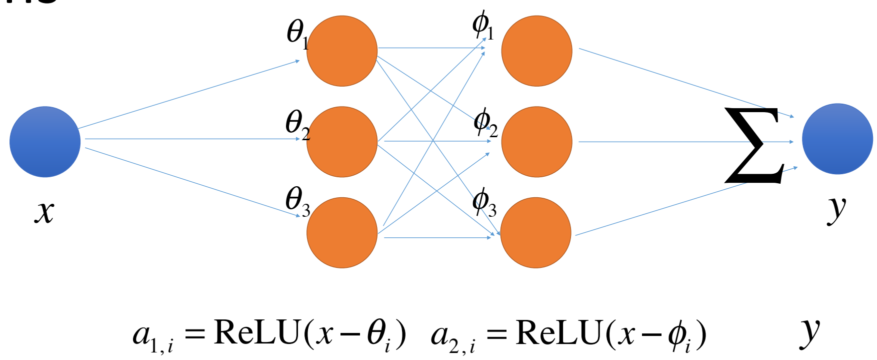
$$y = \sum_i \mathrm{ReLU}(\text{x-}\theta_i)$$



piece-wise linear, 6 knots

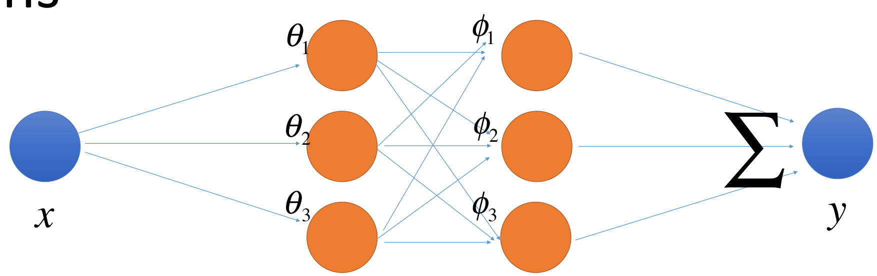
Assume $w_i = 1$ for all neurons, just learn the bias term b_i

Thin + Tall NN
With 6 neurons



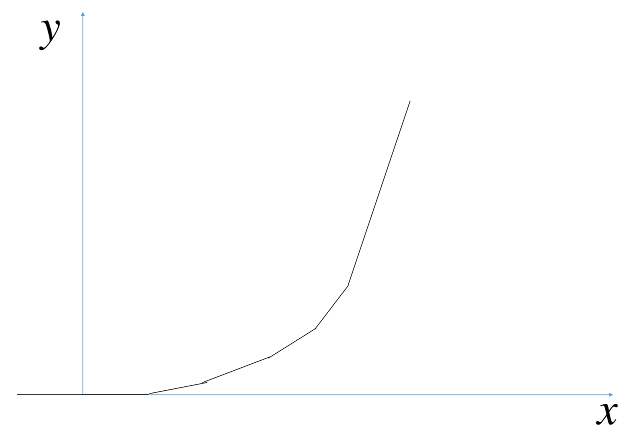
Assume $w_i = 1$ for all neurons, just learn the bias term b_i

Thin + Tall NN
With 6 neurons



$$a_{1,i} = \text{ReLU}(x - \theta_i) \quad a_{2,i} = \text{ReLU}(x - \phi_i)$$

$$y = \sum_{1 \le i \le 3} \text{ReLU}([\sum_{1 \le j \le 3} \text{ReLU}(x\text{-}\theta_j)]\text{-}\phi_i)$$

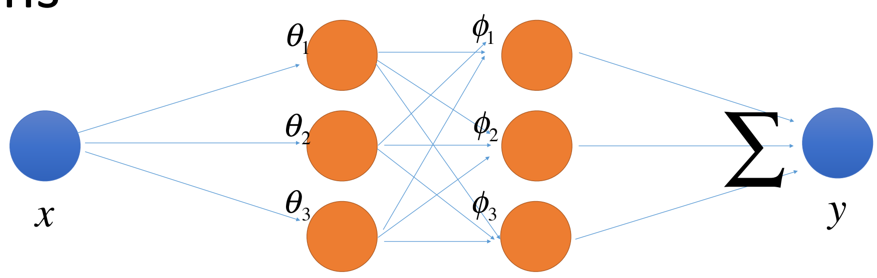


piece-wise linear, can have **9 knots**!

Assume $w_i = 1$ for all neurons, just learn the bias term b_i

Thin + Tall NN

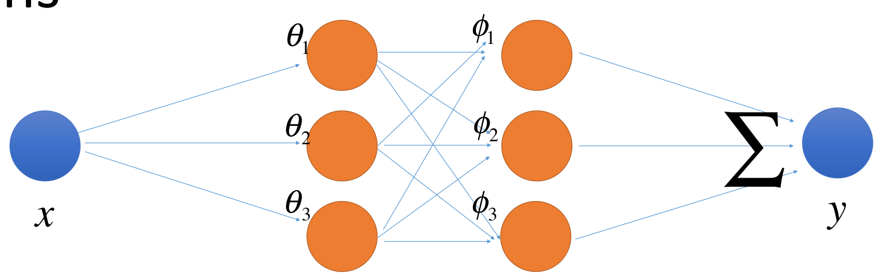
With 6 neurons



Interpretation I: With the same number of neurons, create combinatorial data flow

Thin + Tall NN

With 6 neurons



Interpretation I: With the same number of neurons, create combinatorial data flow

Interpretation II: Abstract data progressively (edge-part-object)

Next lecture:

A big step-forward to reduce parameters of networks:

Convolutional Neural Network