CS 466/566
Introduction to Deep Learning

Lecture 7 – Mid-class Summary

Compiled from various resources

Course Summary

- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - **Supervised Machine Learning**:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - **Unsupervised Machine learning:**
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - **Reinforcement Learning**

- **Types of Models**
  - **Generative vs Discriminative Models**

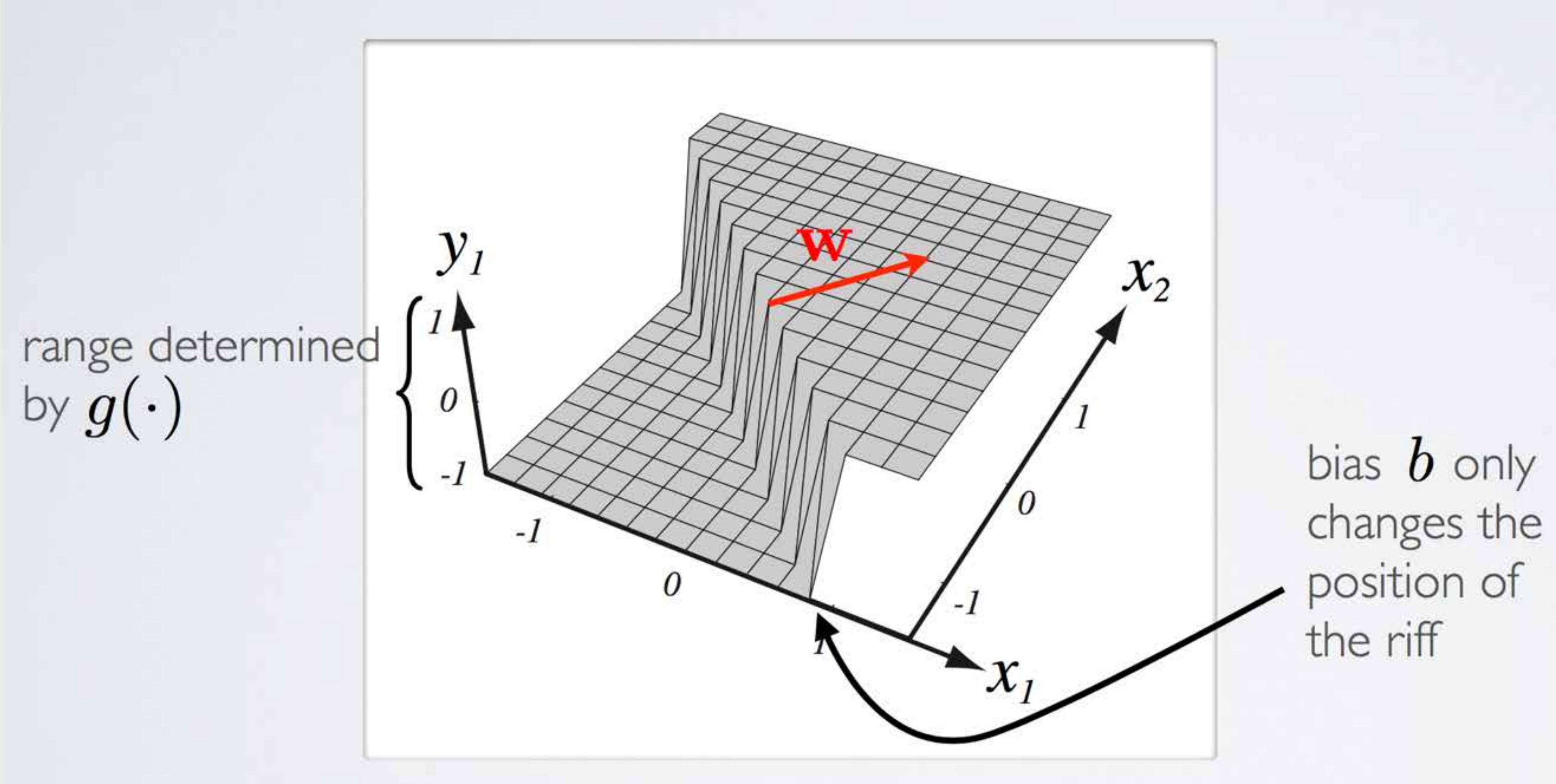
Course Summary

- Neural Networks
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

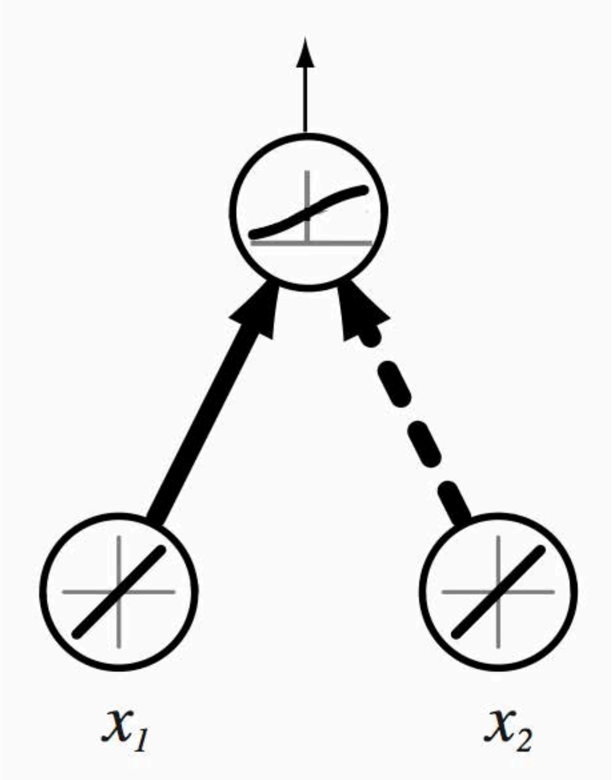
- Types of Machine Learning
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- Types of Models
  - Generative vs Discriminative Models

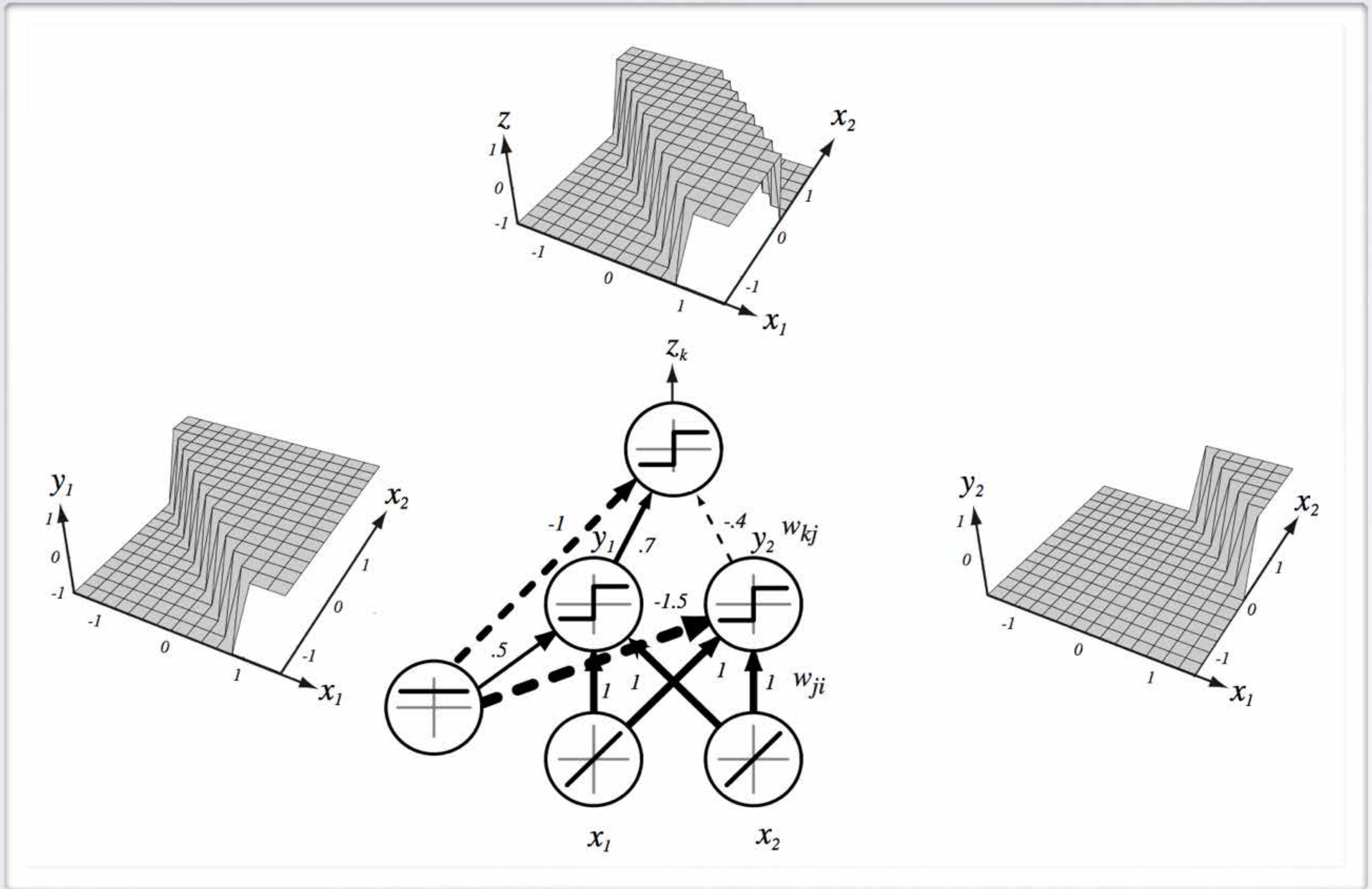
Capacity of a Neuron



(from Pascal Vincent's slides)

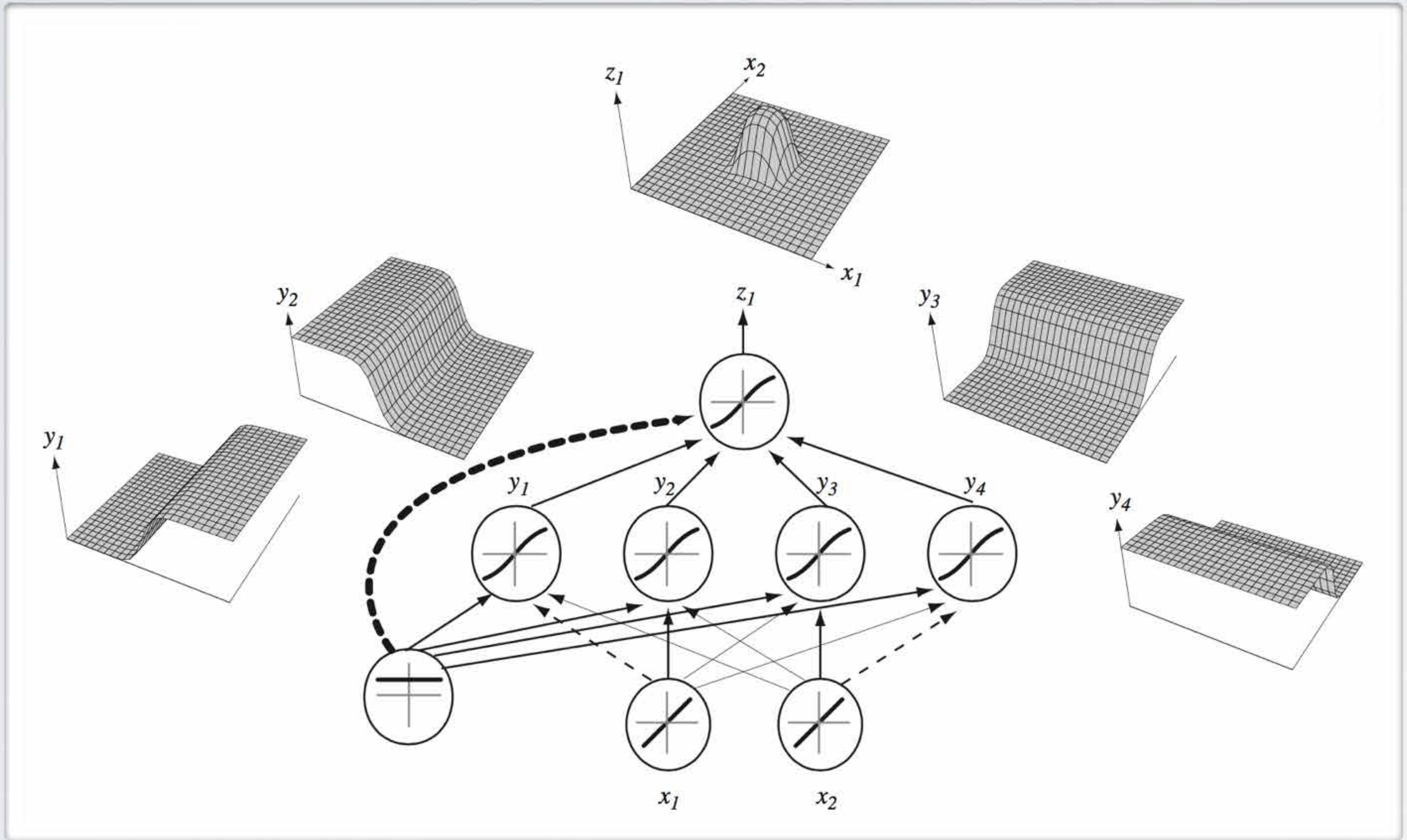


Capacity of a Neural Network



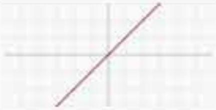
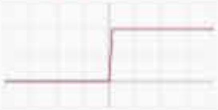
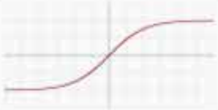
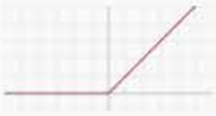
(from Pascal Vincent's slides)

Capacity of a Neural Network



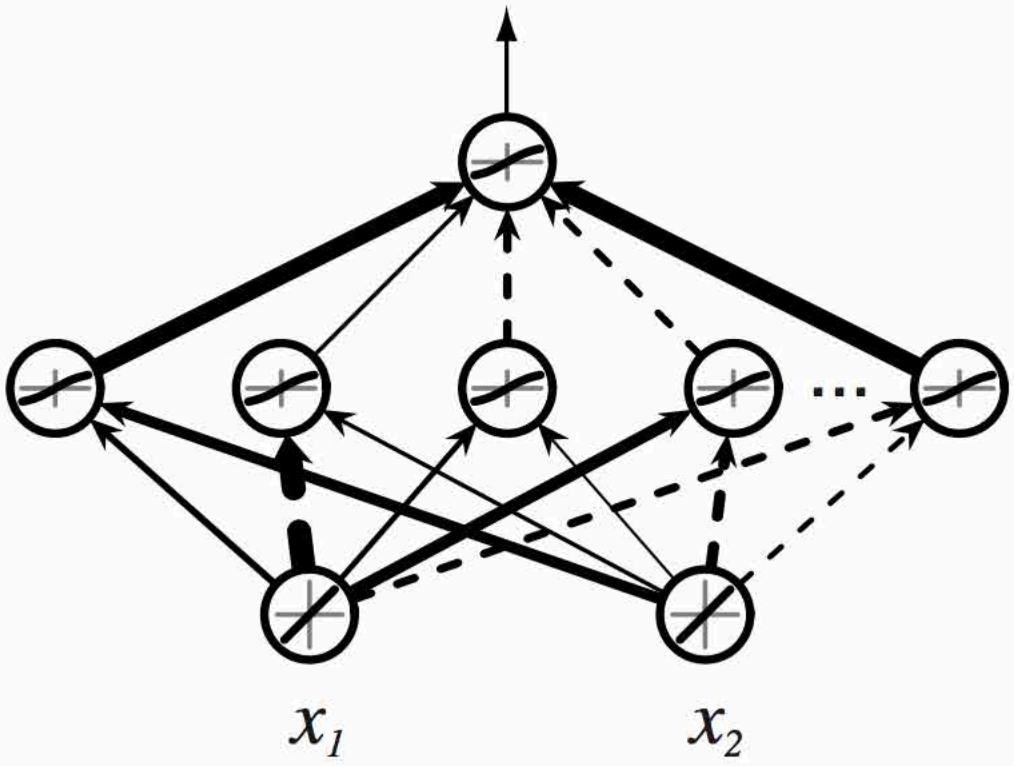
(from Pascal Vincent's slides)

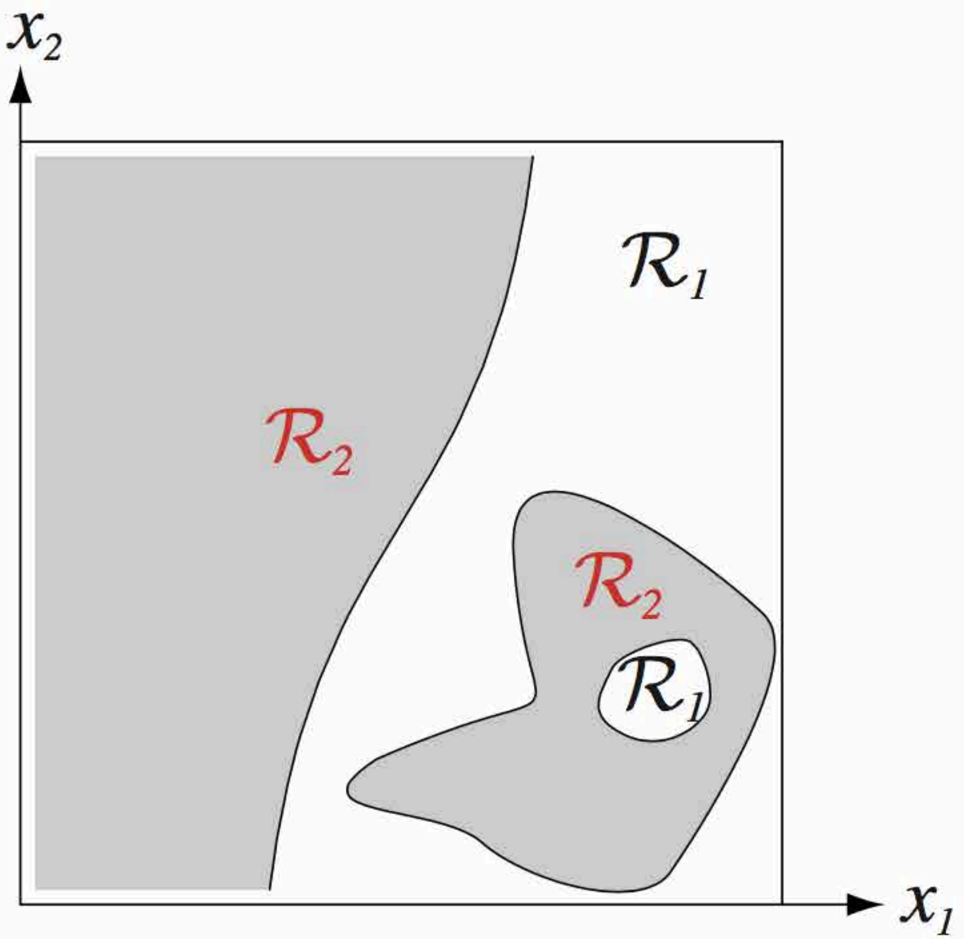
Activation functions

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for} \quad x \neq 0 \\ ? & \text{for} \quad x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1+e^{-x}}$

- Activation functions are the things that give non-linear separation power to Neural Networks.
- Do not confuse activations of hidden layer neurons and logistic regression.
- For a binomial (two class) classification problem, we may choose sigmoid function both for hidden layer activation and output layer at the same time.
- This doesn’t mean that they are put there for the same purpose.

Capacity of a Neural Network (Classification)





Course Summary

- Neural Networks
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- Types of Machine Learning
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- Types of Models
  - Generative vs Discriminative Models

Course Summary

- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

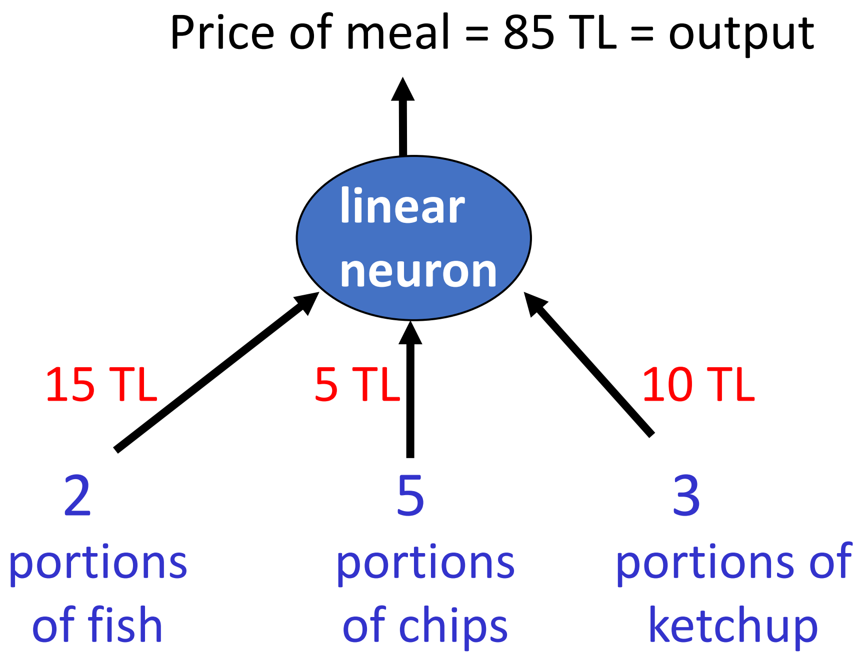
- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- **Types of Models**
  - Generative vs Discriminative Models

A toy example to illustrate regression

- Each day you get lunch at the cafeteria.
  - Your diet consists of fish, chips, and ketchup.
  - You get several portions of each.
- The cashier only tells you the total price of the meal
  - After several days, you should be able to figure out the price of each portion.
- The iterative approach: Start with random guesses for the prices and then adjust them to get a better fit to the observed prices of whole meals.

The true weights used by the cashier



Cost Functions

- Cost function for regression
  - Quadratic cost: Also known as mean squared error or sum squared error

$$J(\theta) = \sum_i \left(y^{(i)} - h_\theta\left(x^{(i)}\right)\right)^2$$

- Cost function for classification (cross-entropy)
  - Two-class Classification = Logistic Regression:

$$J(\theta) = -\sum_i \left(y^{(i)} \log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_\theta\left(x^{(i)}\right)\right)\right)$$

  - Multi-class Classification = Multinomial Logistic Regression:
    - Apply Softmax to any vector **z** such that: $z \rightarrow \left\{\sum_i \frac{exp(z_i)}{\sum_k exp(z_k)}\right\}$

$$J(\theta) = \left\{-\sum_i \left(y^{(i)} \log\left(h_\theta\left(x^{(i)}\right)\right)\right)\right\}$$

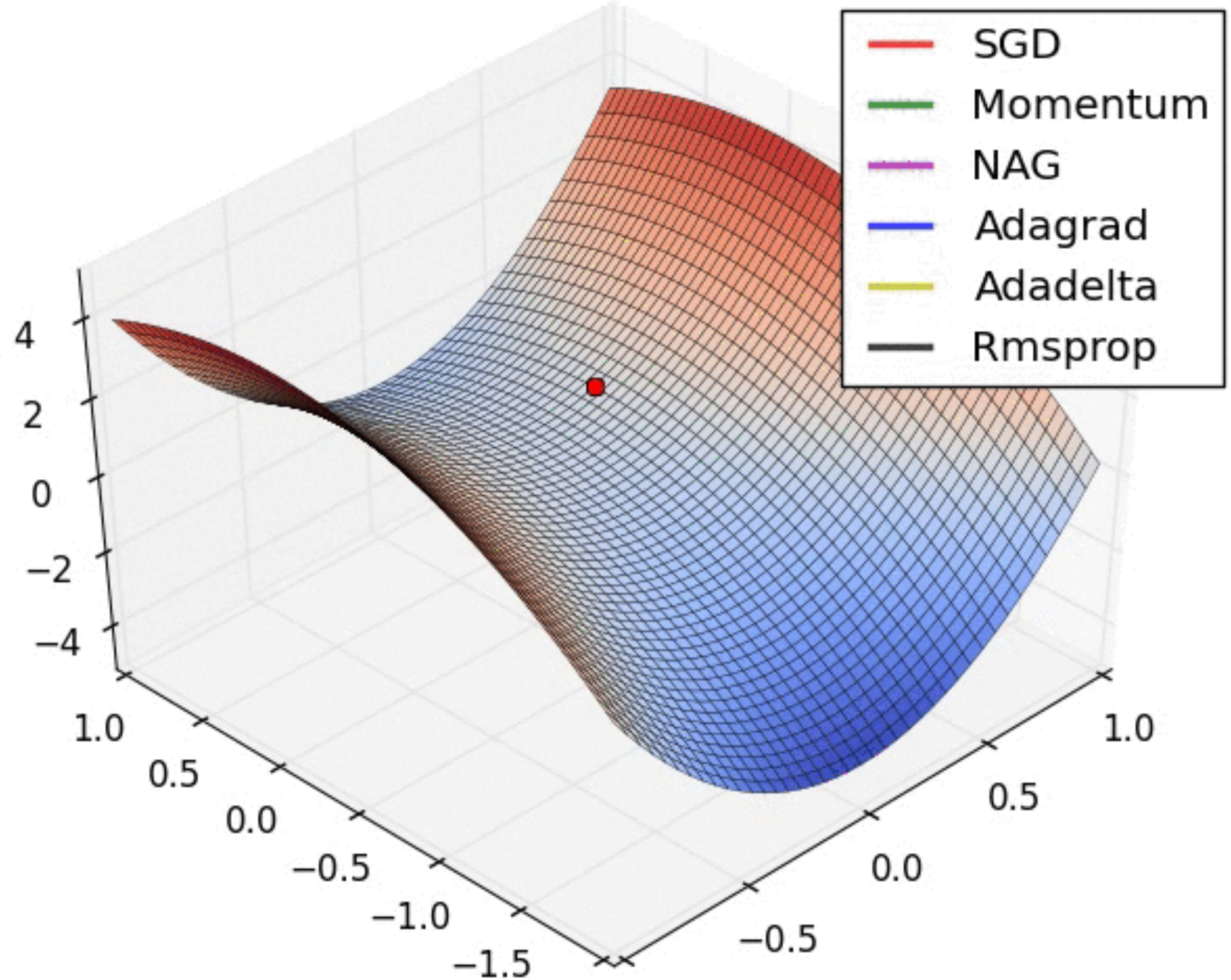
Course Summary

- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- **Types of Models**
  - Generative vs Discriminative Models

Learning: Try to find minima of cost function



- Many training algorithms exist
    - Gradient Descent (Batch)
    - Stochastic Gradient Descent (online)
    - Minibatch Gradient Descent
- Learning Rate

Course Summary

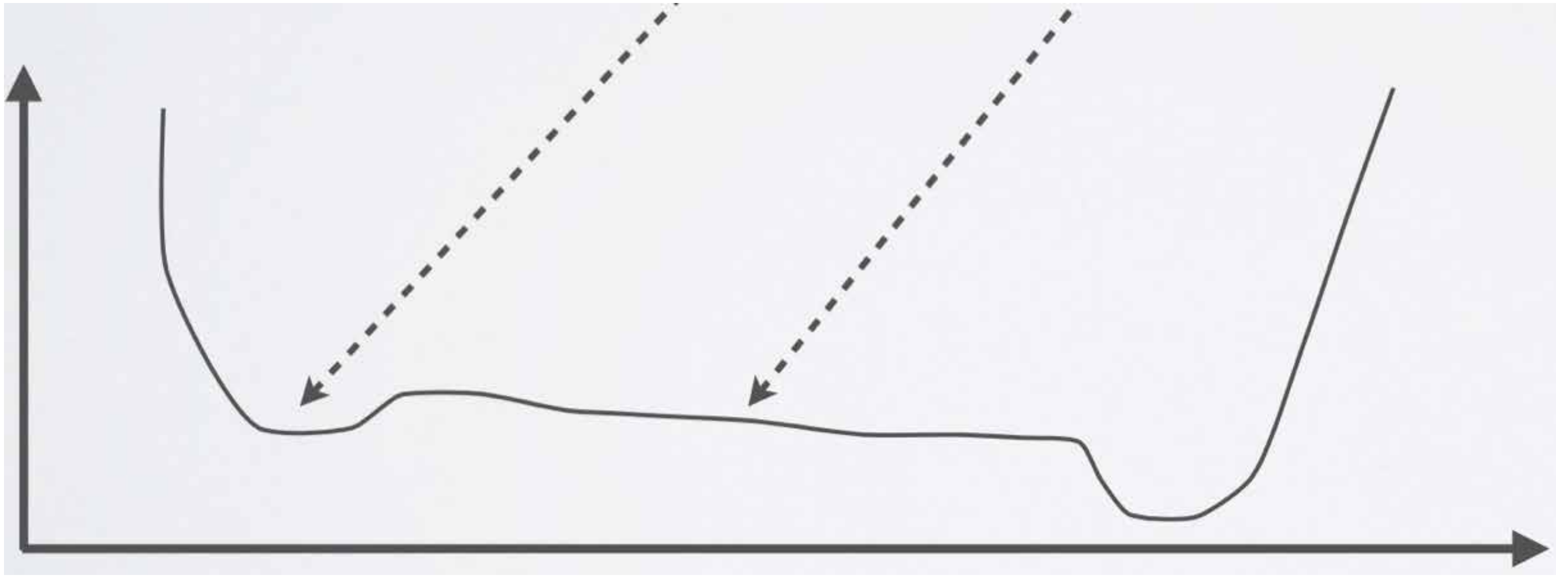
- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

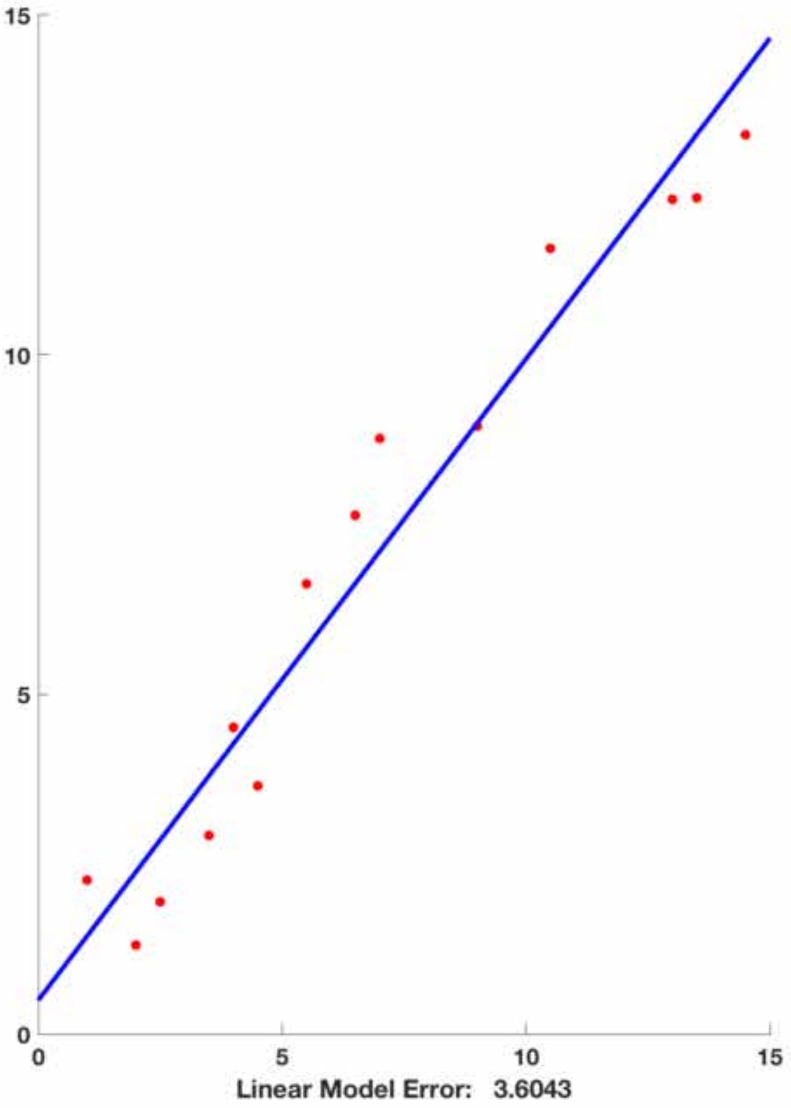
- **Types of Models**
  - Generative vs Discriminative Models

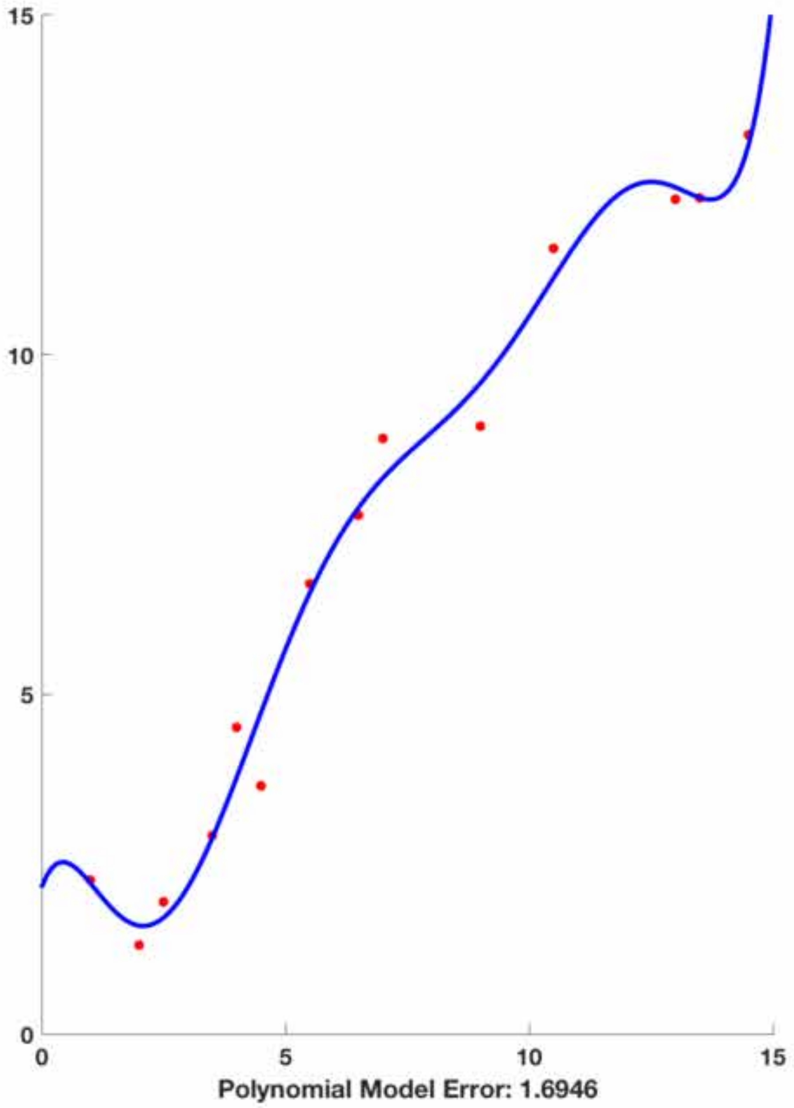
Walking on the landscape of cost function

- there isn't a single global optimum (non-convex optimization)
  - we can permute the hidden units (with their connections) and get the same function
  - we say that the hidden unit parameters are not identifiable
  - Optimization can get stuck in local minimum or plateaus

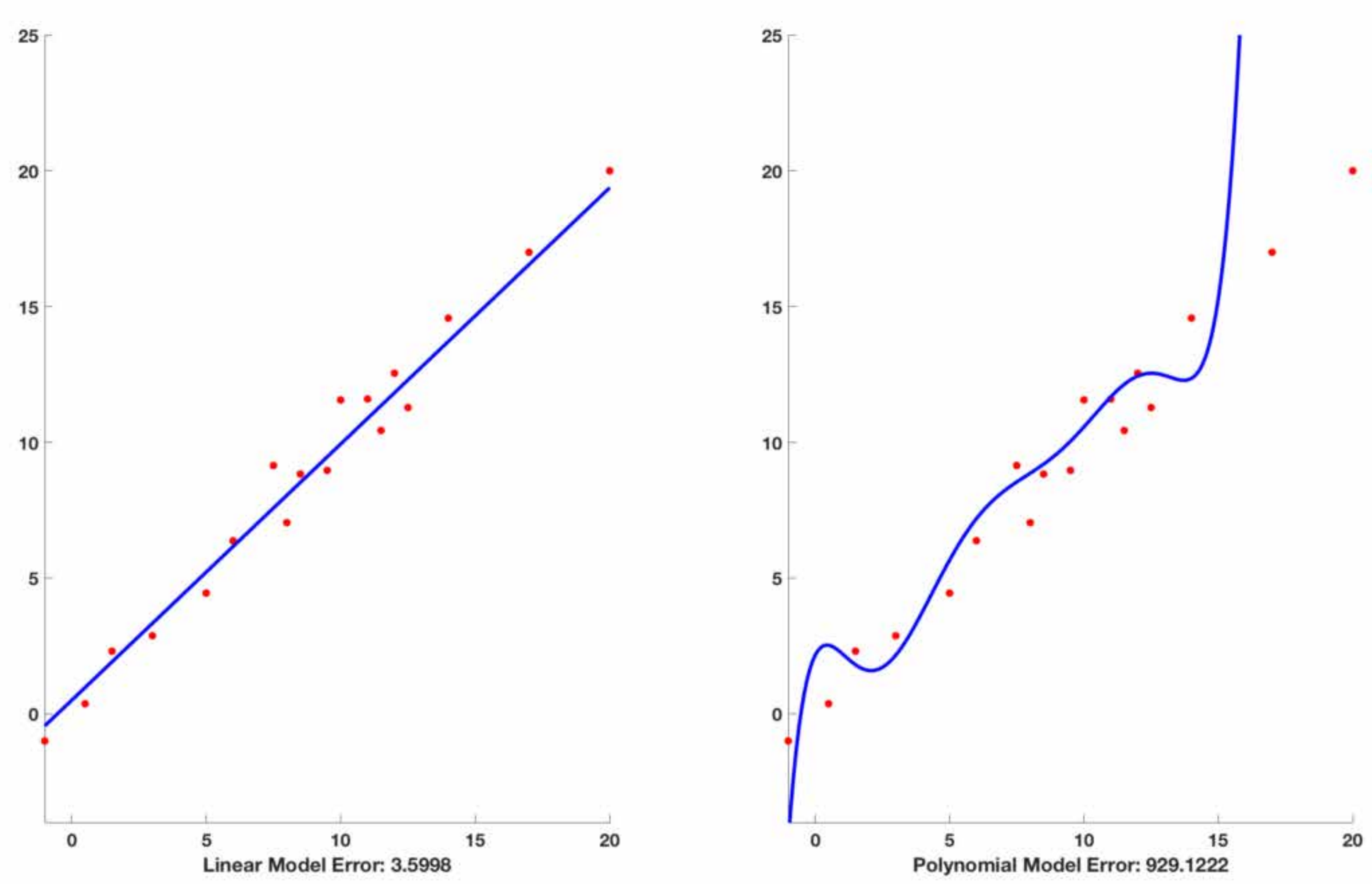


Training Set Performance





Test Set Performance



Bias / Variance

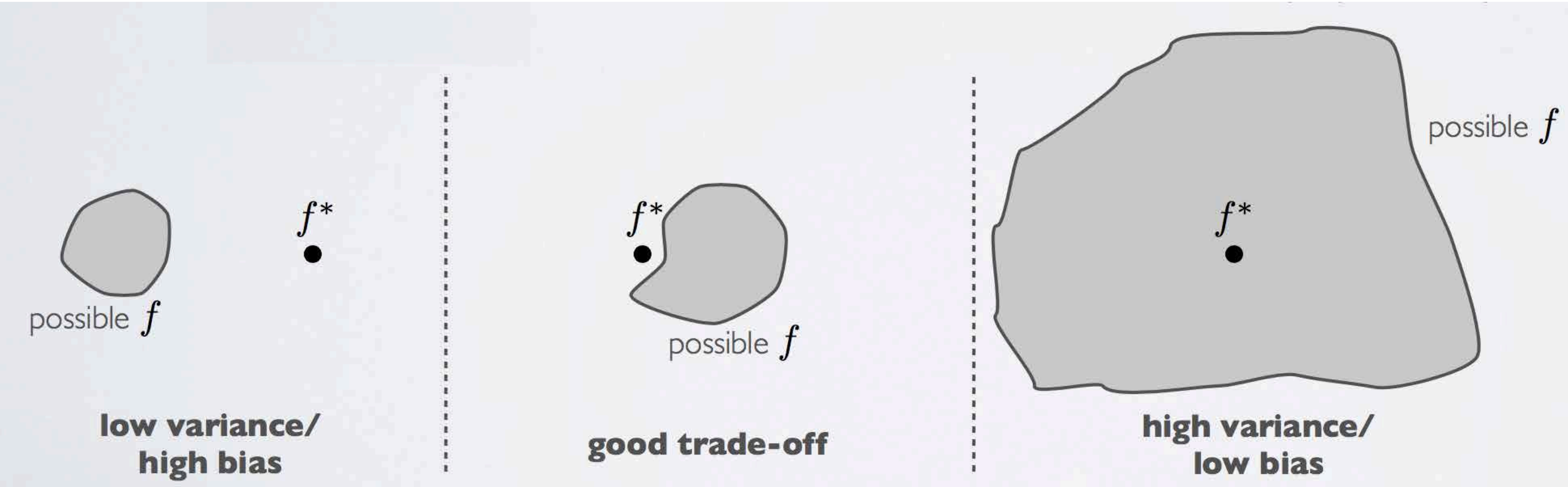
- Linear model had a higher **bias.**
- Polynomial model suffers from a different problem. The model depends a lot on the choice of training data.
- If you change the data slightly, the error will swing widely.
- Therefore, the model is said to have high **variance**.
- To keep the **bias** low, we need a complex model (e.g. a higher degree polynomial)
- But a complex model has a tendency to overfit and increase the **variance**.

Underfitting, Overfitting: Knowing when to stop

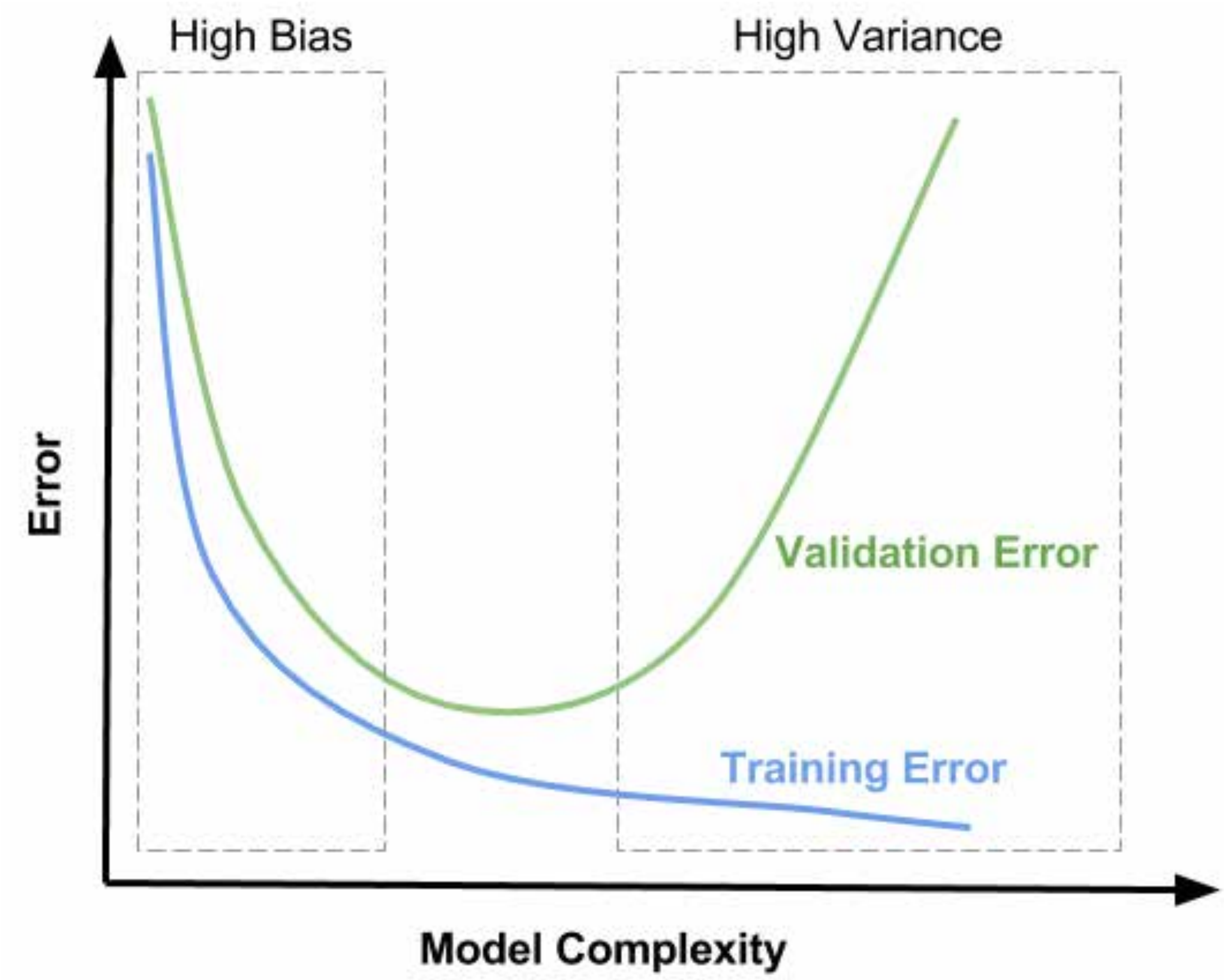
- To select the number of epochs, stop training when validation set error increases (with some look ahead)

Bias – Variance Trade-off

- **Variance of trained model:** does it vary a lot if the training set changes?
- **Bias of trained model:** the average model close to the true solution
- **Generalization error:** (can be seen as) the sum of the (squared) bias and variance



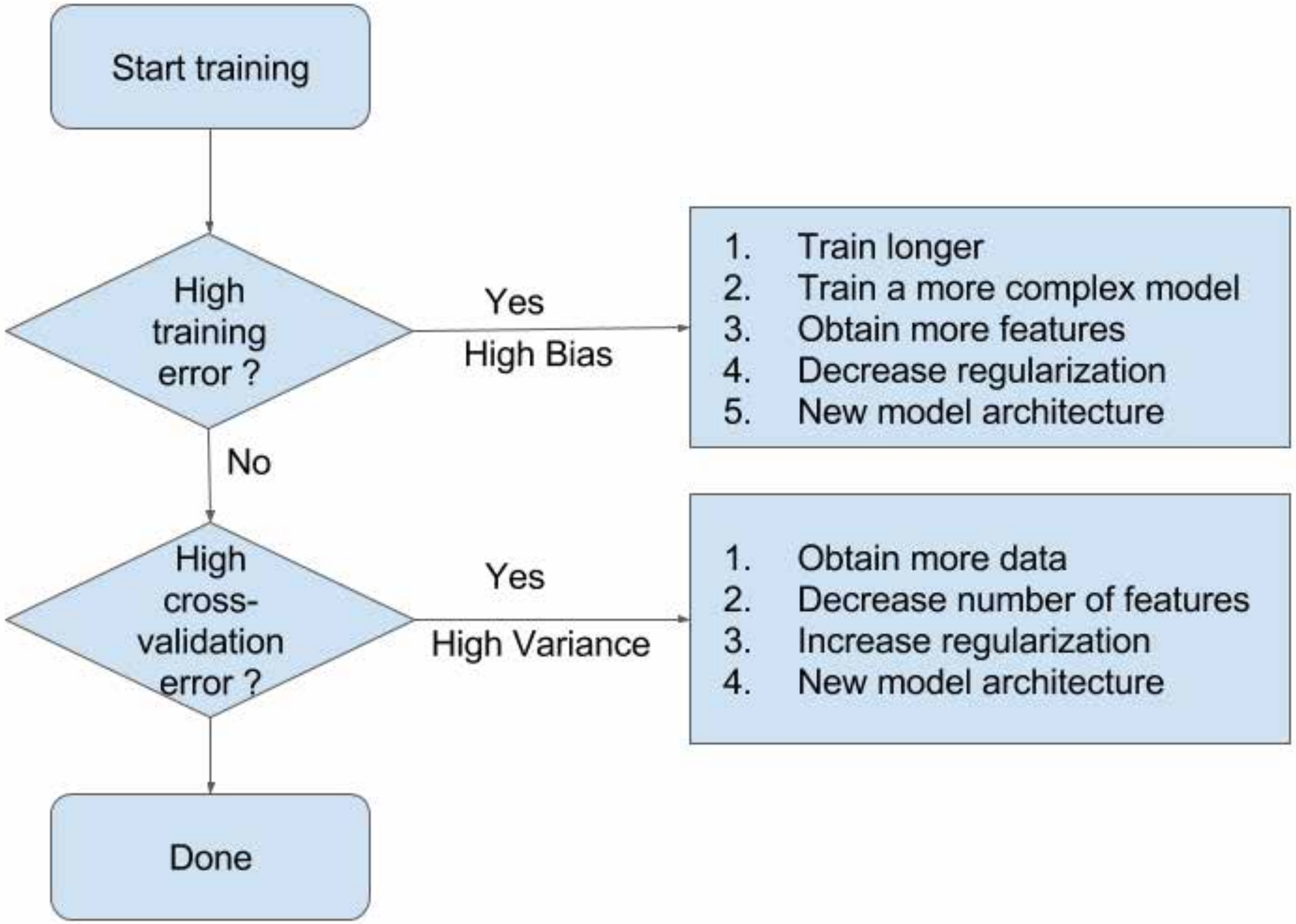
Bias – Variance Trade-off



How to detect?

- **How to detect a high bias problem?**
  - High training error.
  - Validation error is similar in magnitude to the training error.

- **How to detect a high variance problem?**
  - Low training error
  - Very high Validation error



Course Summary

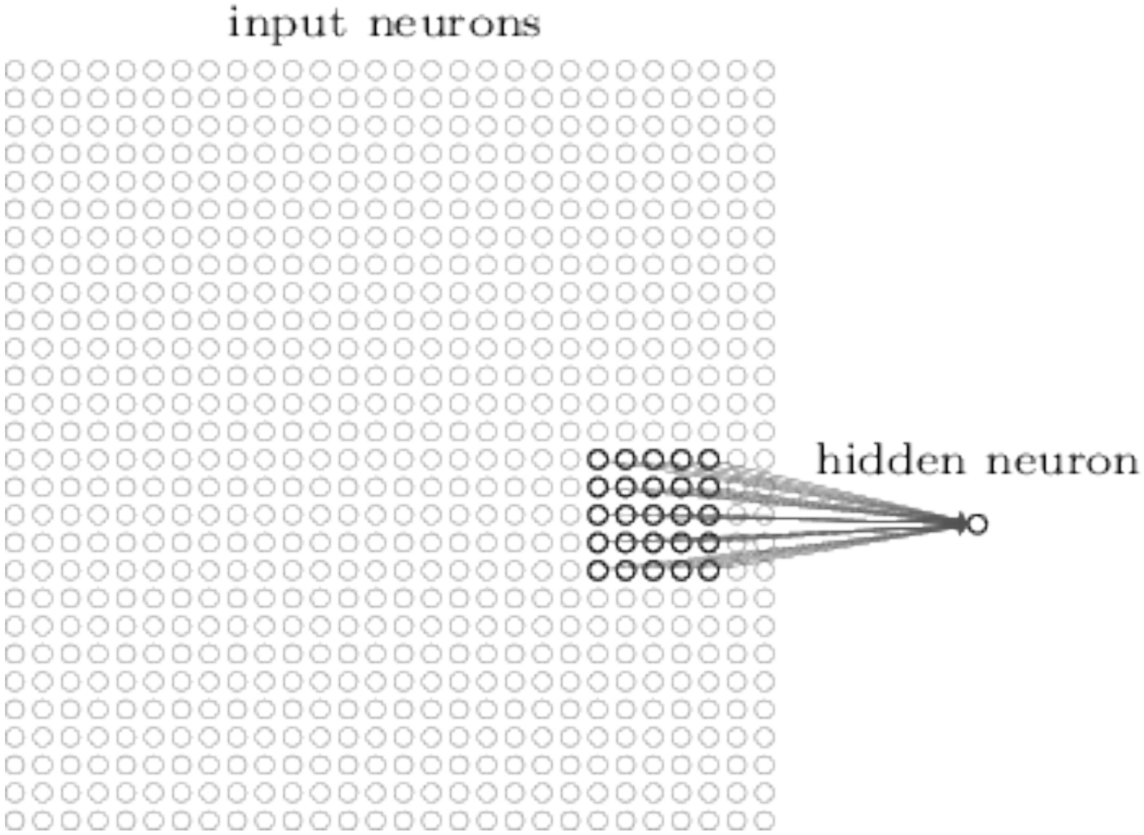
- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- **Types of Models**
  - Generative vs Discriminative Models

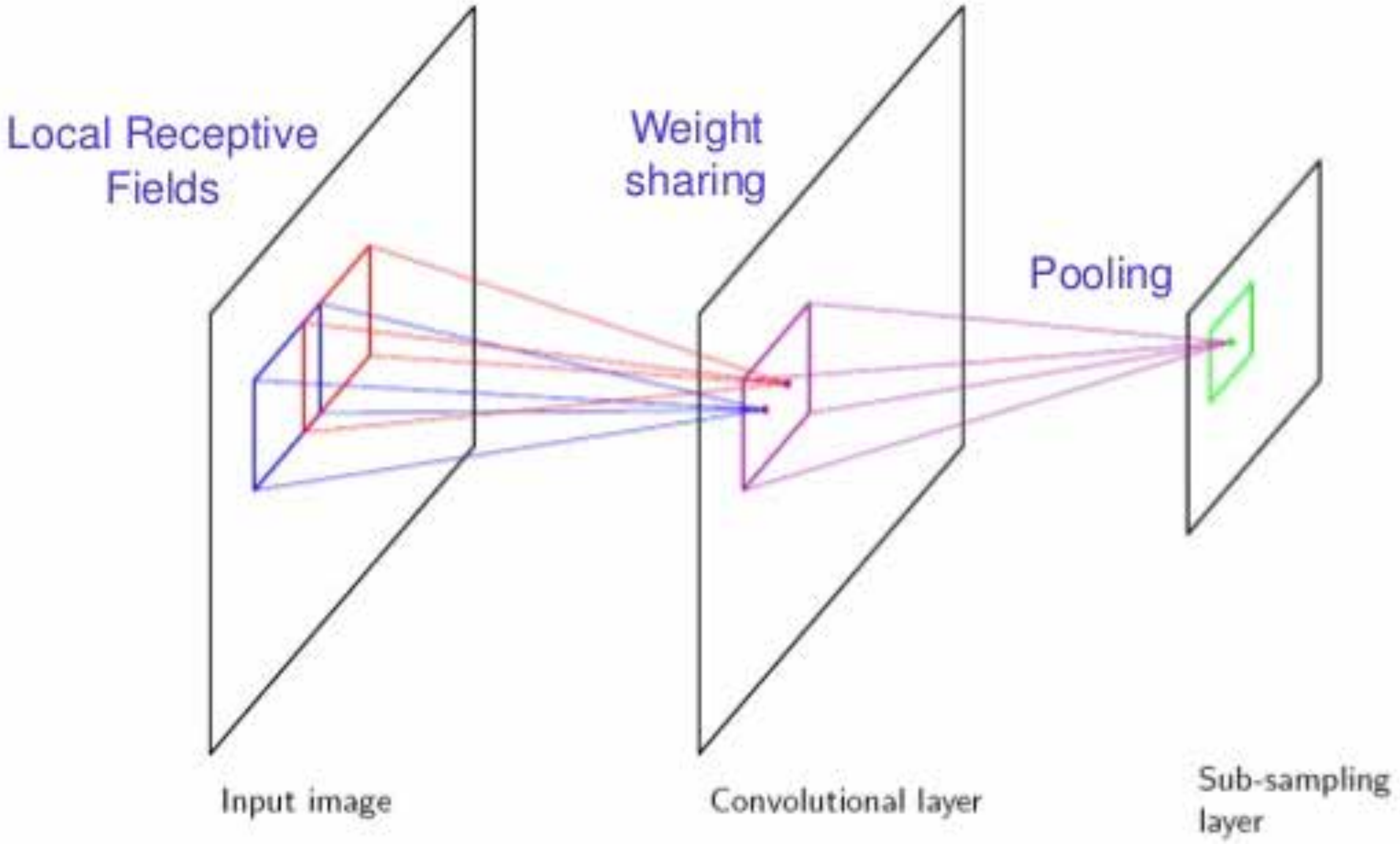
1st simplification: Local Receptive Fields

- Consider a 2D representation of input neurons (e.g. M-NIST data)
- Consider a specific neuron in the hidden layer.
- Now, it will be connected to a local neighborhood only
  - Instead of connected it to each and every input



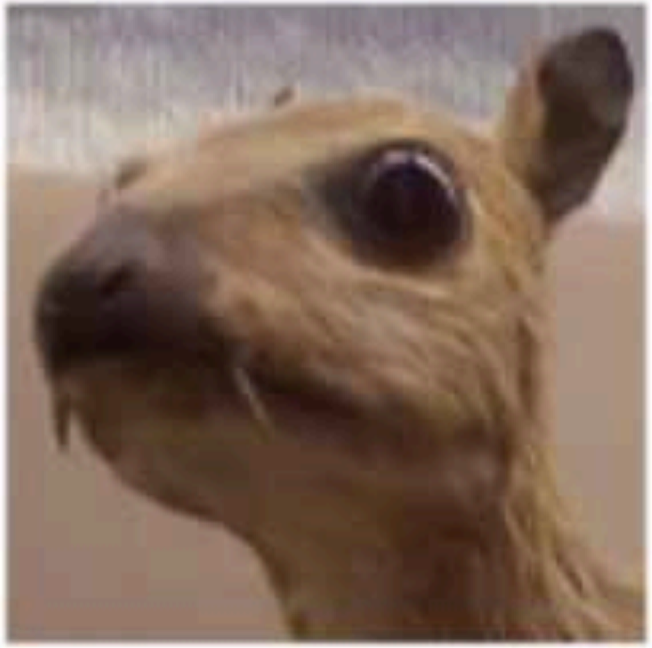
2^{nd} Simplification: Shared Weights

(LeCun et al., 1989)



Recall: Convolution

Input image

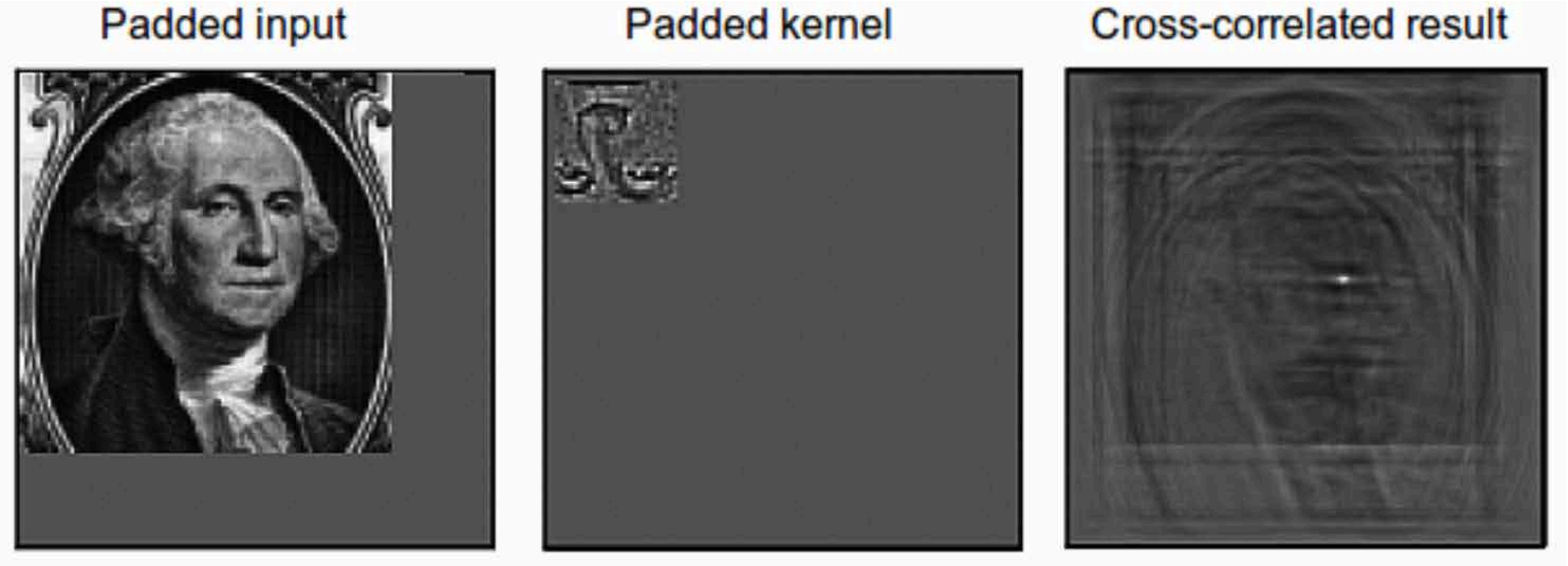


Convolution Kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

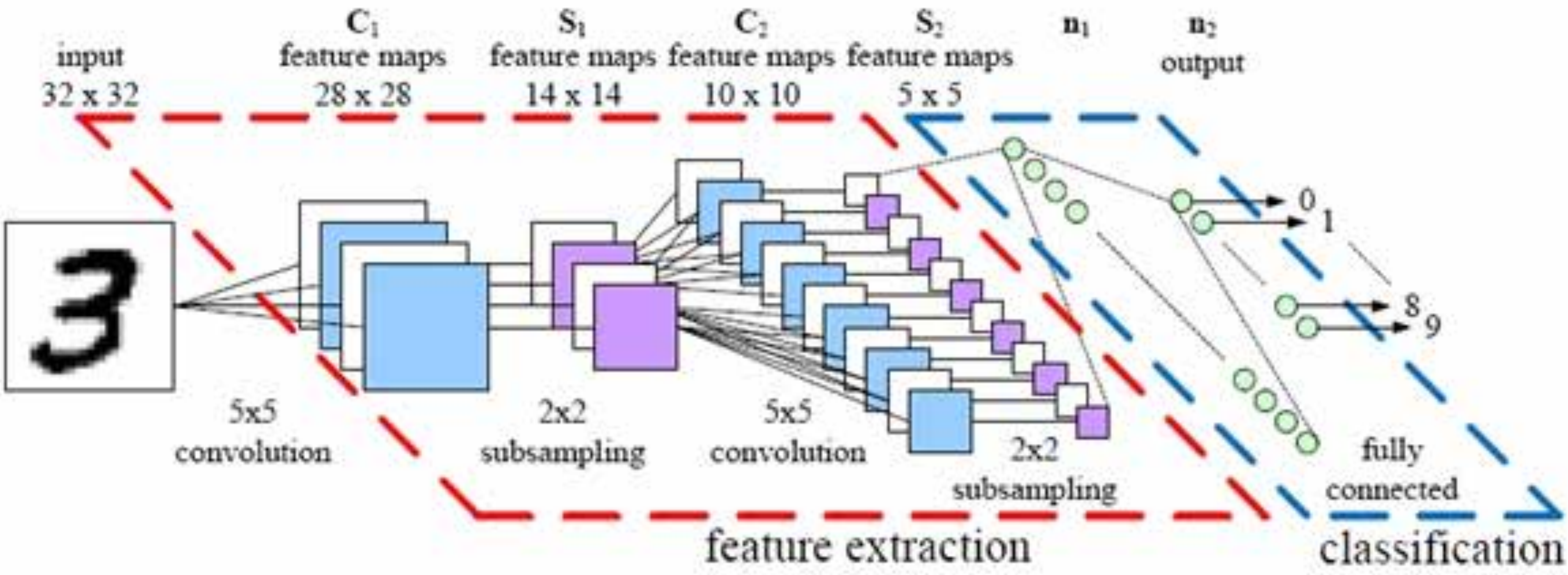
Feature map

Convolutional Filters



Convolutional filters can be interpreted as feature detectors, that is, the input (feature map) is filtered for a certain feature (the kernel) and the output is large if the feature is detected in the image.

An example CNN pipeline



Course Summary

- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

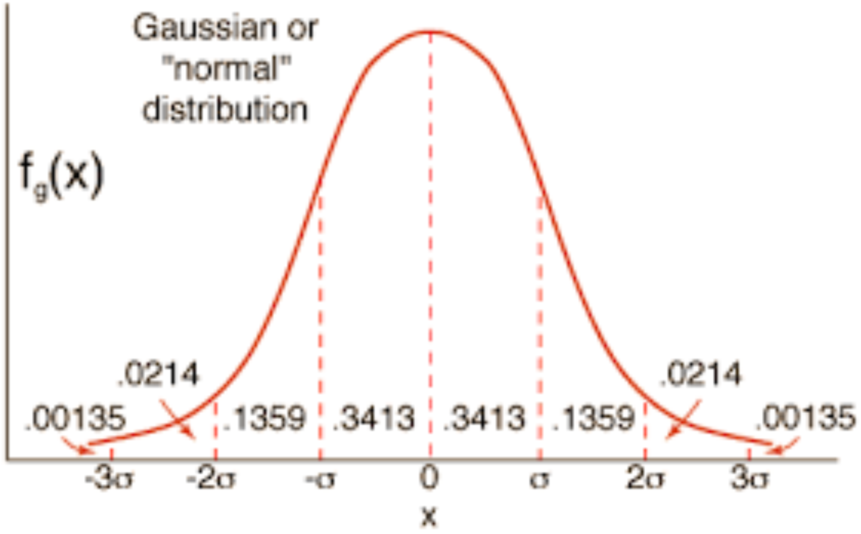
- **Types of Models**
  - Generative vs Discriminative Models

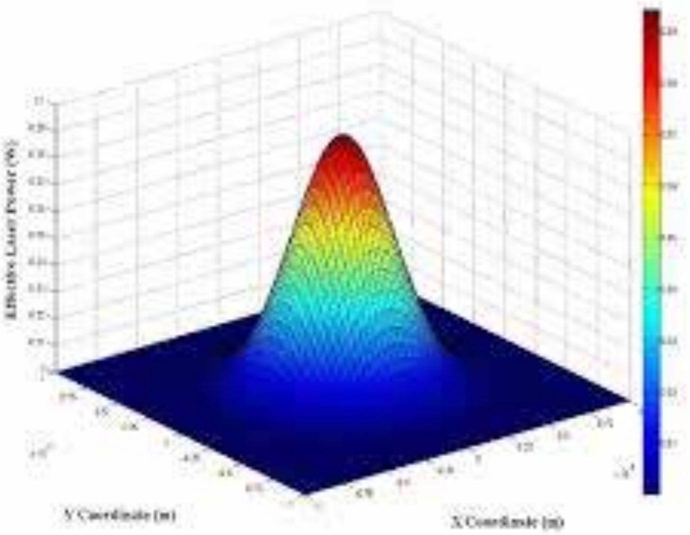
Regularization: L2 Regularization

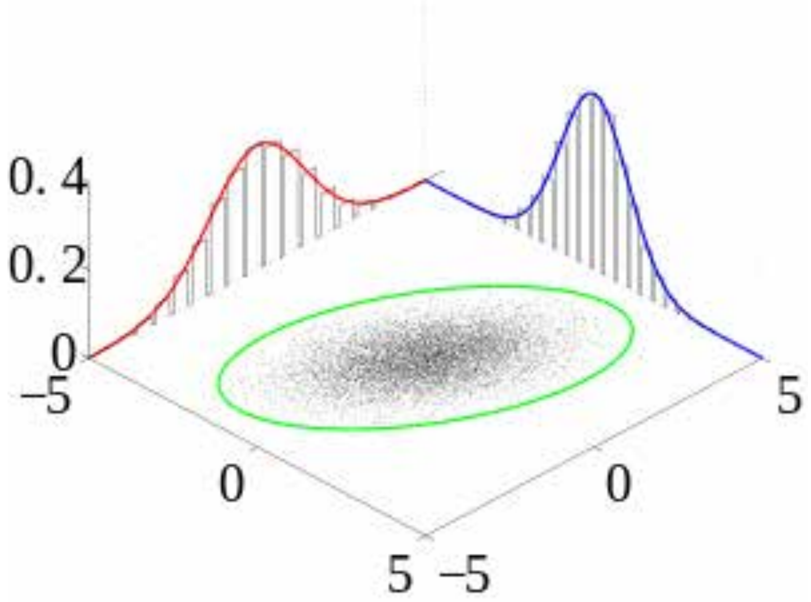
- Penalizes the gradient term in back-prop algorithm:

$$\Omega(\boldsymbol{\theta}) = \sum_k \sum_i \sum_j \left(W_{i,j}^{(k)}\right)^2 = \sum_k ||\mathbf{W}^{(k)}||_F^2$$

- Gradient: $\nabla_{\mathbf{W}^{(k)}} \Omega(\boldsymbol{\theta}) = 2\mathbf{W}^{(k)}$
  - Only applied on weights, not biases
  - Can be interpreted as having a Gaussian Prior over weight values.





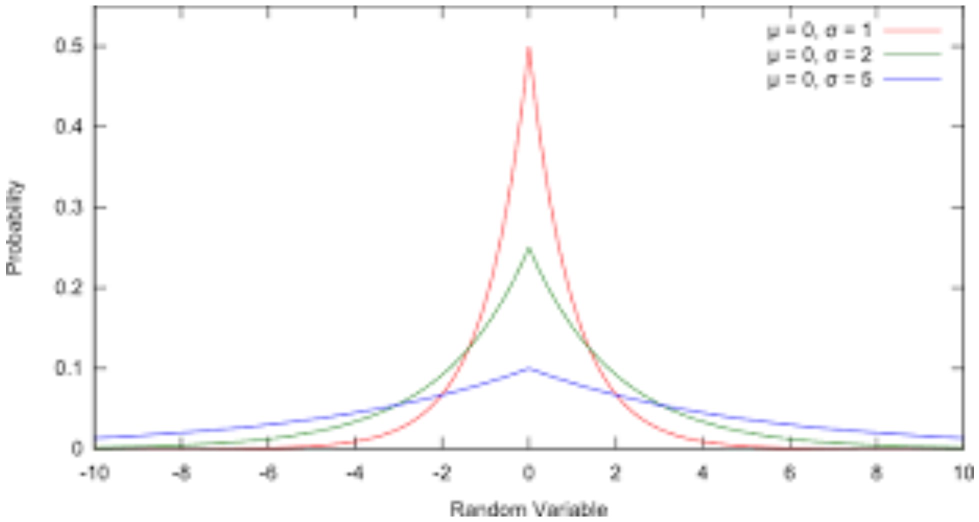


Regularization: L1 Regularization

- Penalizes the gradient term in back-prop algorithm:

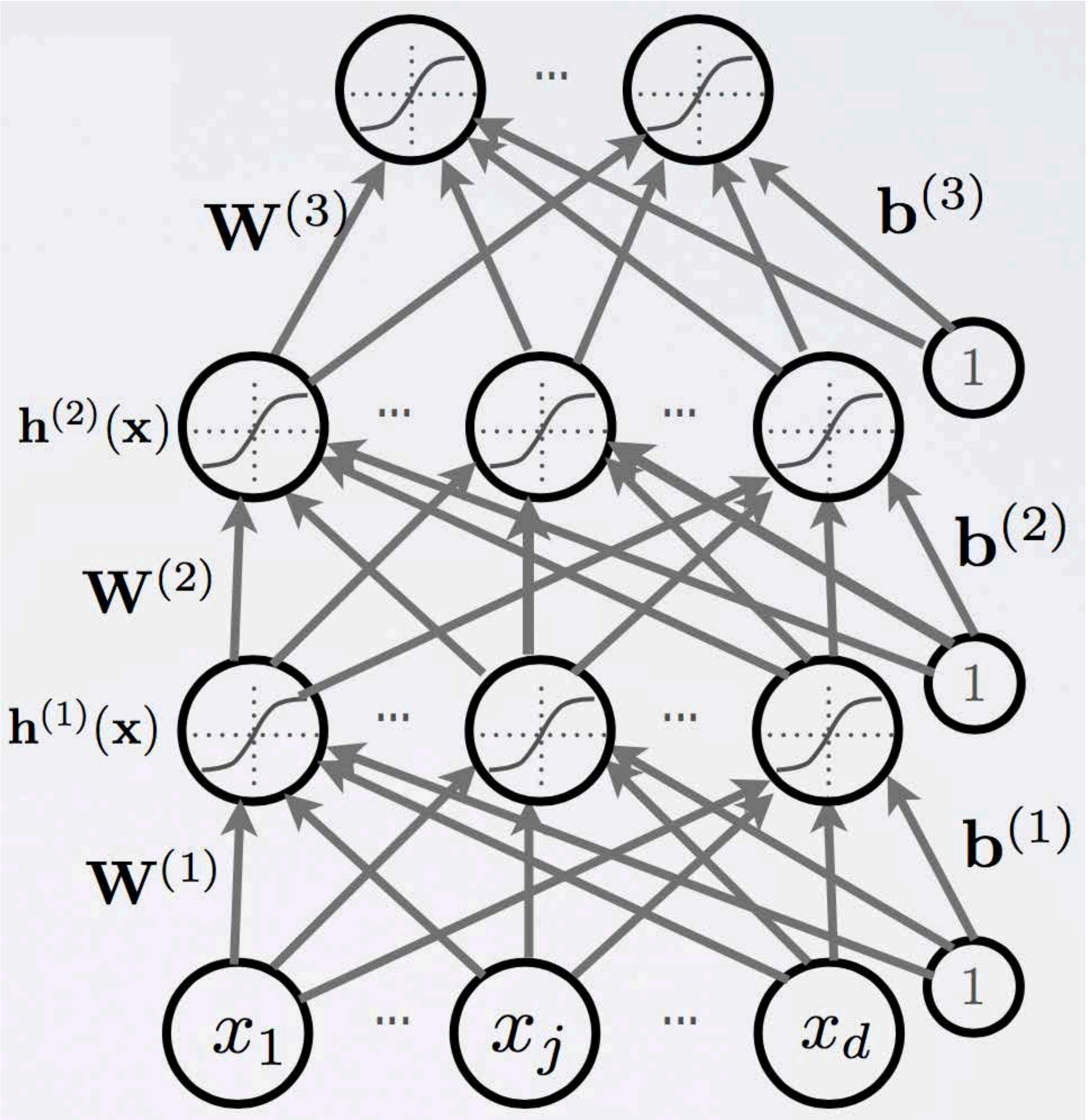
$$\Omega(\boldsymbol{\theta}) = \sum_k \sum_i \sum_j |W_{i,j}^{(k)}|$$

- Gradient: $\nabla_{\mathbf{W}^{(k)}} \Omega(\boldsymbol{\theta}) = \mathrm{sign}(\mathbf{W}^{(k)})$
  - Only applied on weights, not biases (again)
  - Unlike L2, L1 regularization will make some of the weights exactly 0.
  - Can be interpreted as having a Laplacian Prior over weight values.



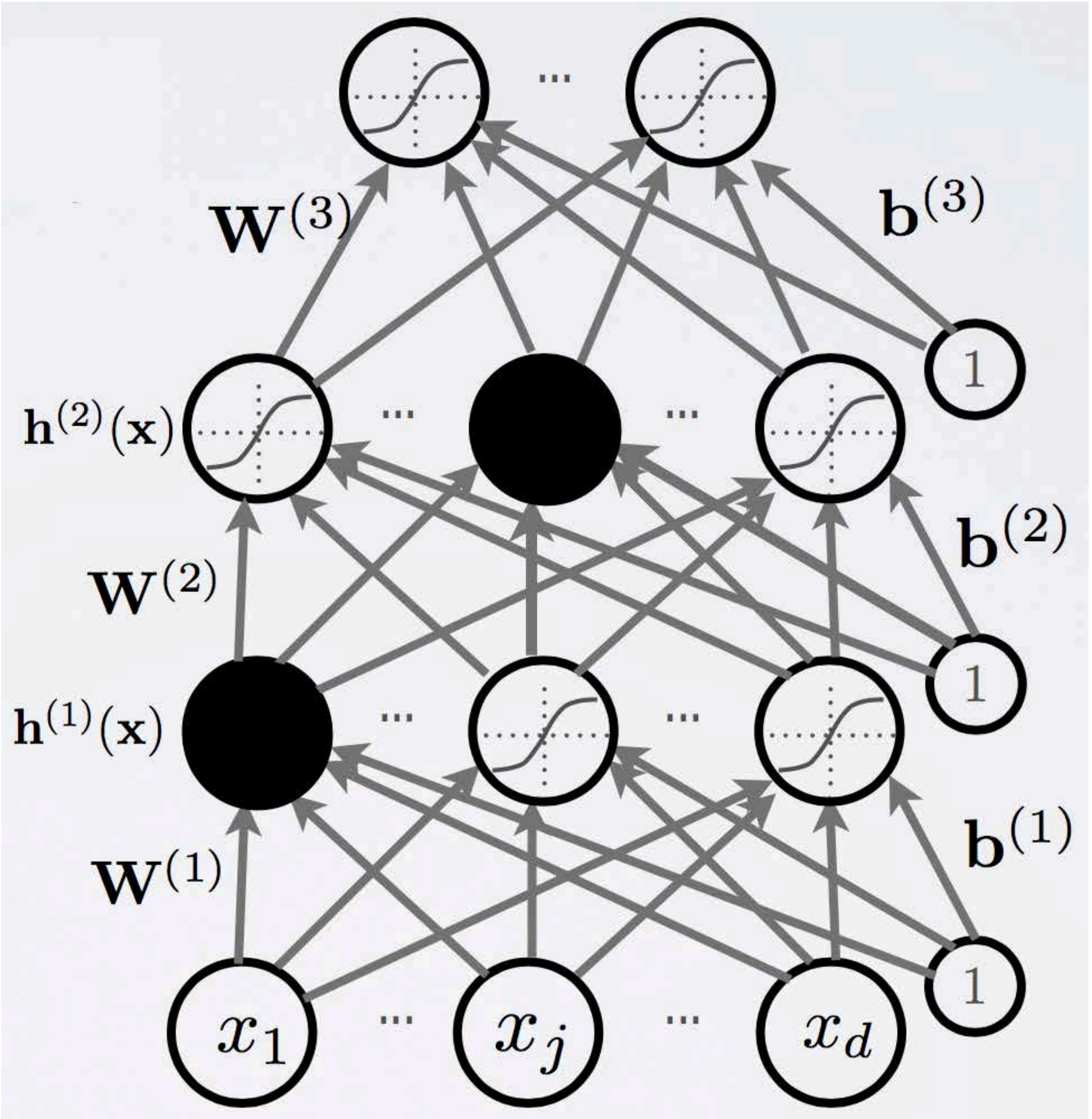
Regularization: Dropout

- Idea: Cripple neural network by removing hidden units randomly.
  - Each hidden unit's activation is set to 0 with a probability, independently.
  - Usually 0.5 works well.



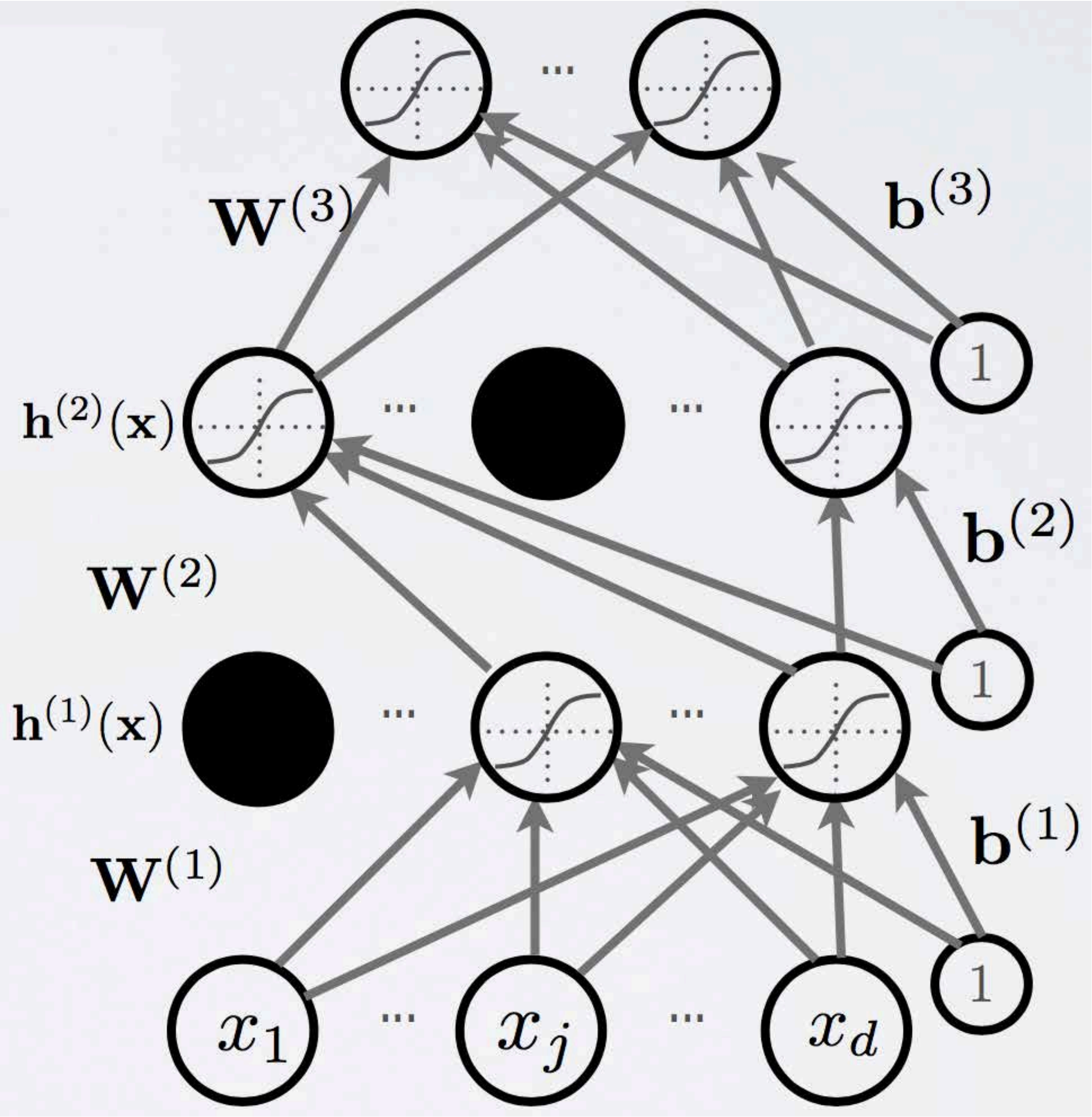
Regularization: Dropout

- Idea: Cripple neural network by removing hidden units randomly.
  - Each hidden unit's activation is set to 0 with a probability, independently.
  - Usually 0.5 works well.

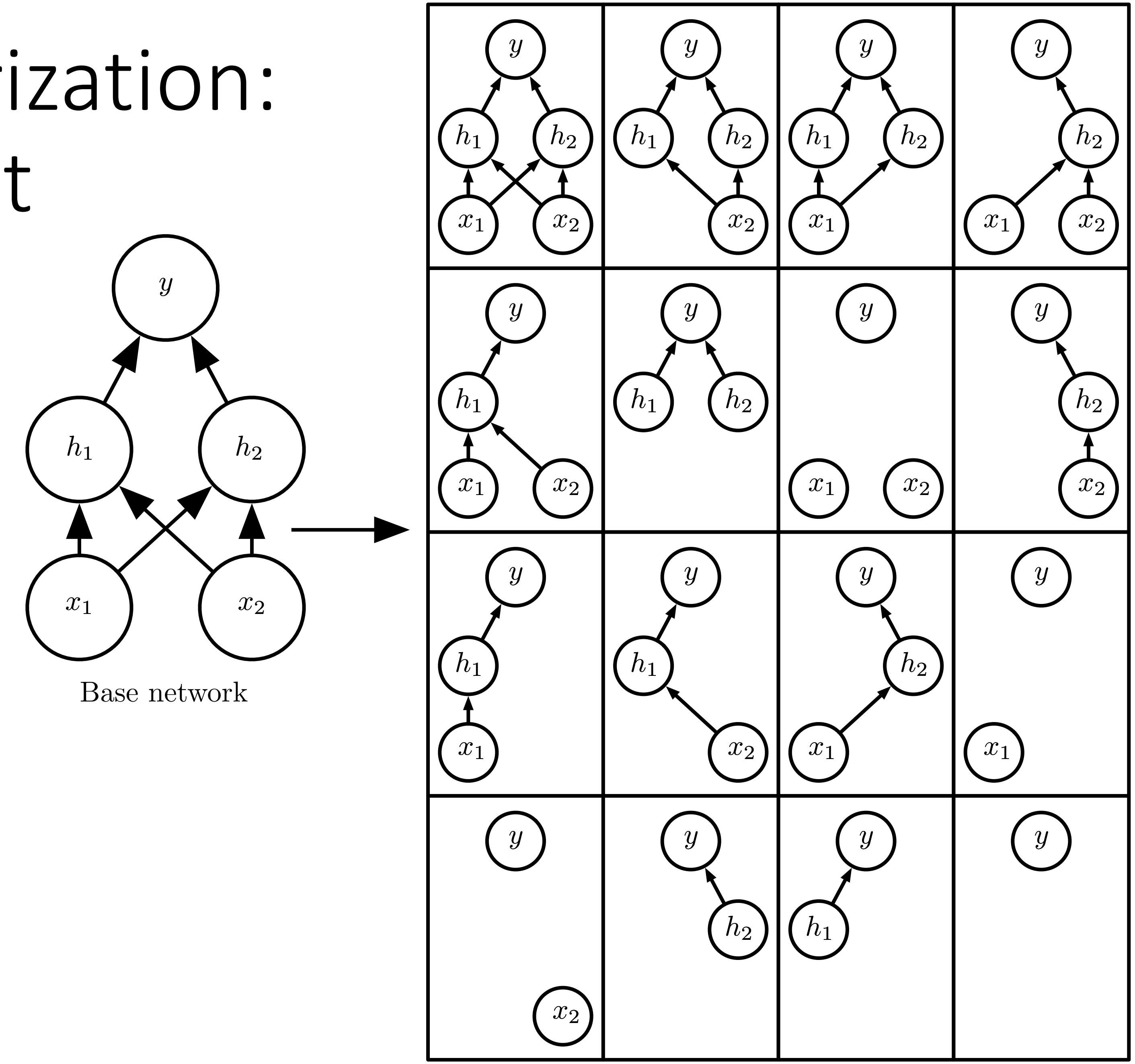


Regularization: Dropout

- Idea: Cripple neural network by removing hidden units randomly.
  - Hidden units cannot cooperate with each other in a specific layer.
  - Therefore, hidden units must be more generally useful.



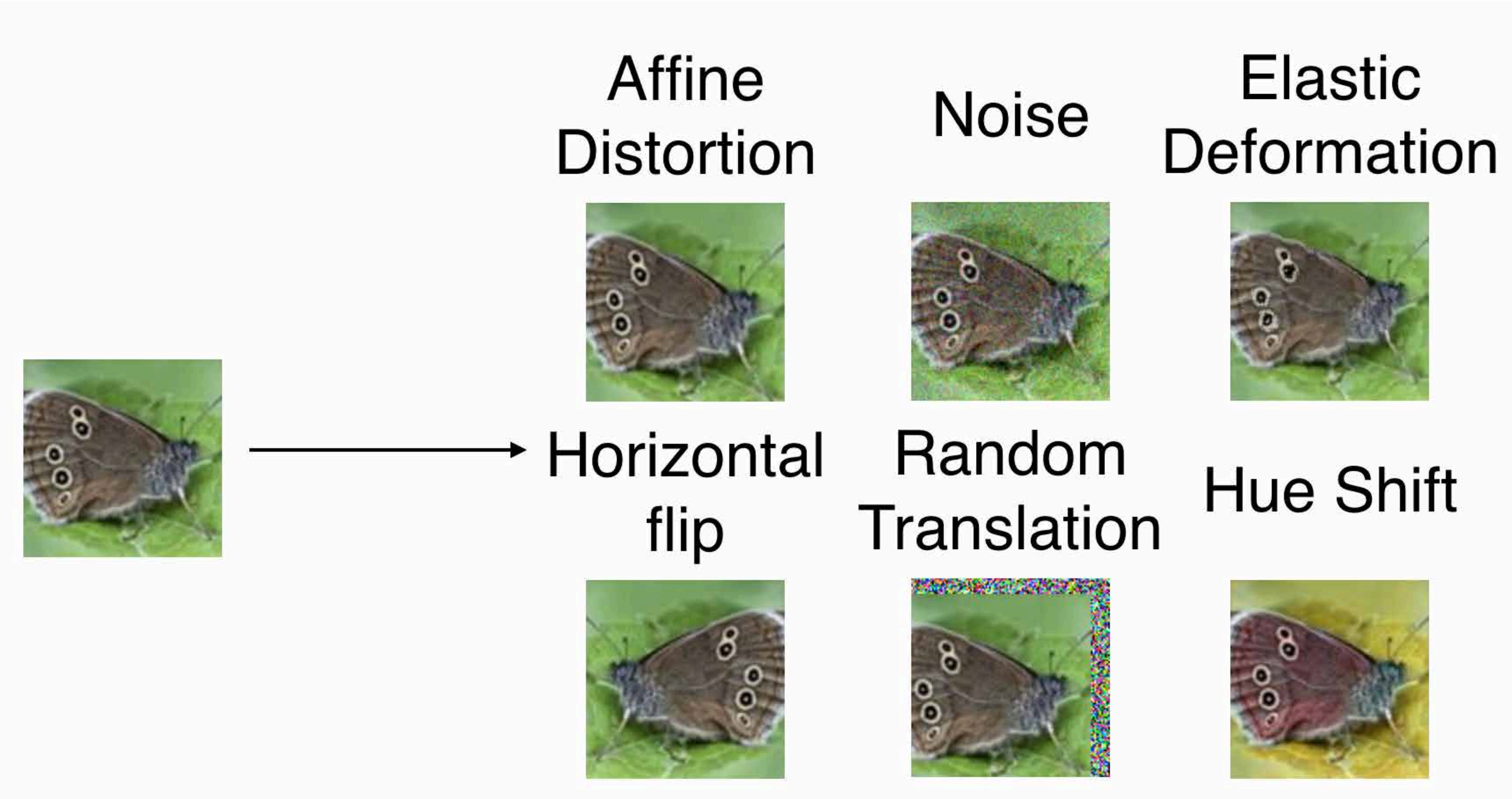
Regularization: Dropout



Base network

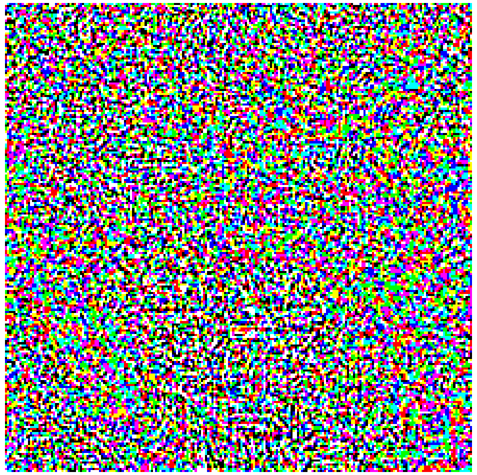
Ensemble of subnetworks

Regularization: Dataset Augmentation



Still Have a Problem: Adversarial Examples

$+ .007 \times$



$=$

$\boldsymbol{x}$	$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$	$\boldsymbol{x} + \epsilon\,\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
$y =$“panda” w/ 57.7% confidence	“nematode” w/ 8.2% confidence	“gibbon” w/ 99.3 % confidence

Course Summary

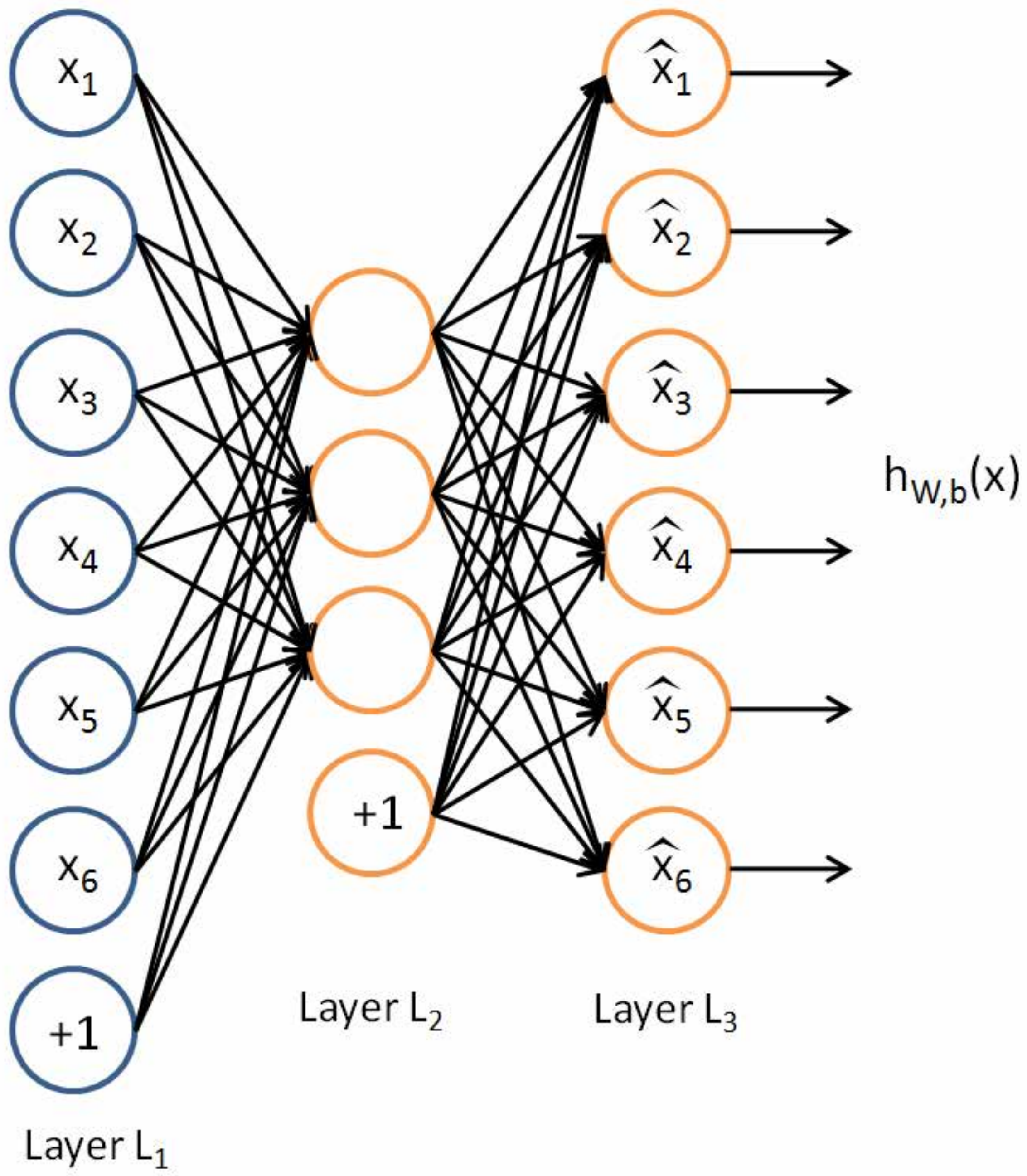
- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

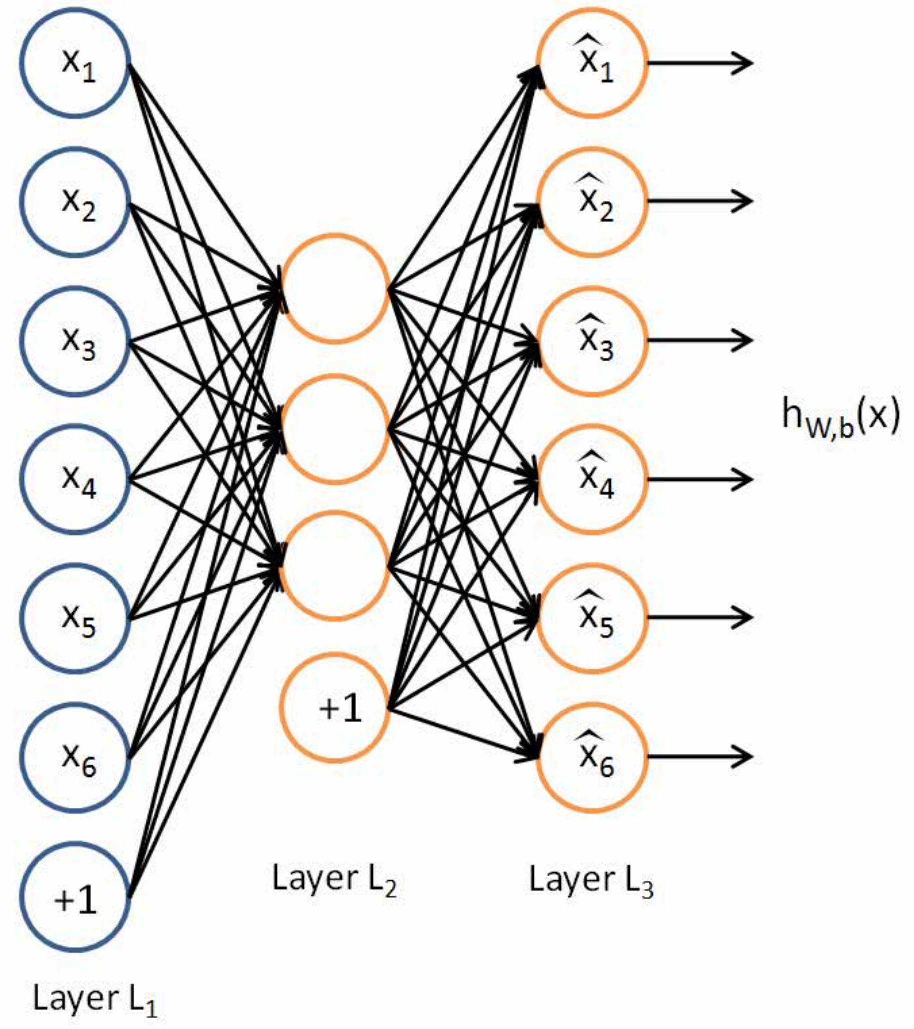
- **Types of Models**
  - Generative vs Discriminative Models

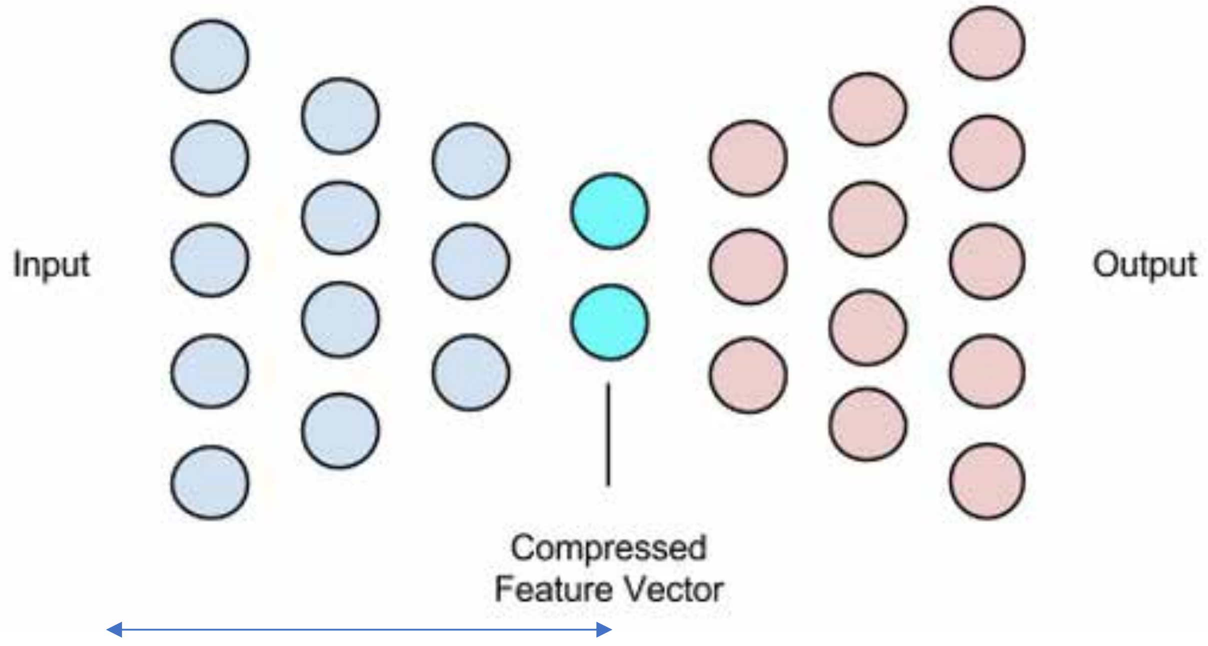
Auto-encoders

- What was hard about training deep MLPs?
- An auto-encoder tries to match the input to the output!
- Basically, does nothing, right?
- Not right.
- It depends on the number of neurons in hidden layer.



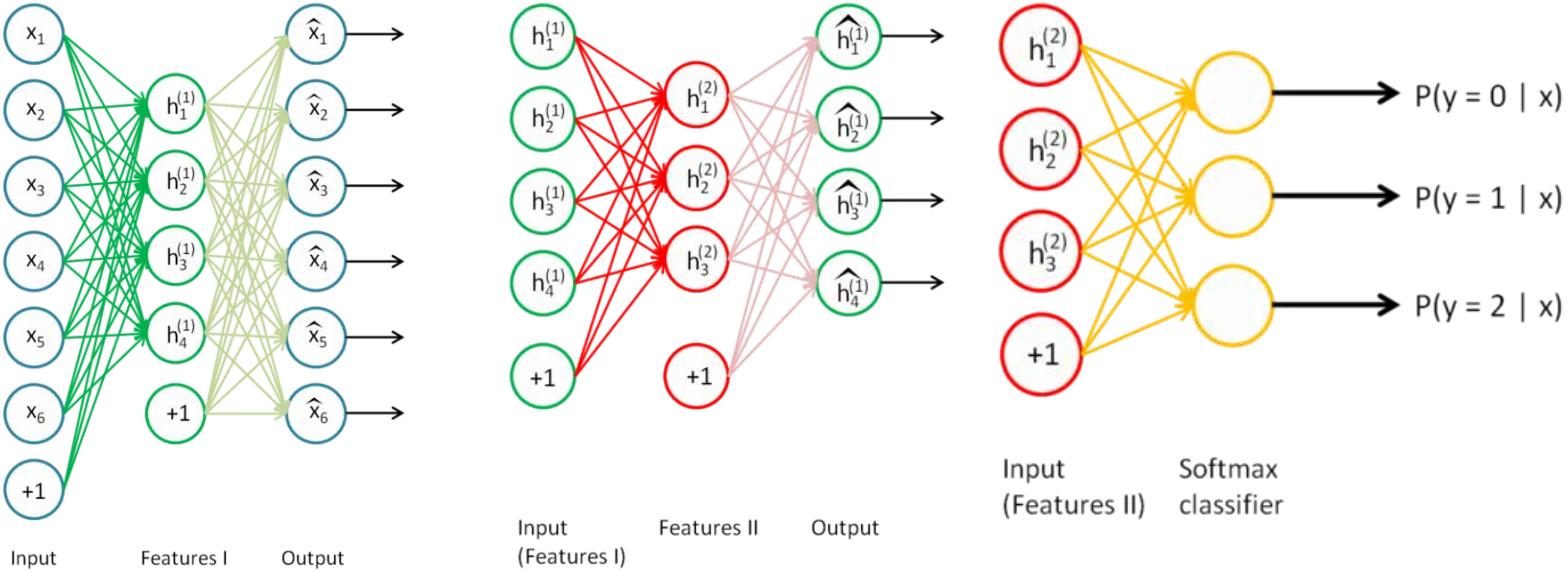
Auto-encoders / Deep Auto-encoders





Representation learning

Stacked Auto-Encoders



Use learned weights as initial weights of real training!

Course Summary

- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- **Types of Models**
  - Generative vs Discriminative Models

Course Summary

- Neural Networks
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- Types of Machine Learning
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- Types of Models
  - Generative vs Discriminative Models

Denoising Auto-encoders

- Denoising auto-encoders attempt to address identity-function risk by randomly corrupting input (i.e. introducing noise) that the auto-encoder must then reconstruct, or denoise.

- Parameters and Corruption level
  - The amount of noise to apply to the input takes the form of a percentage. Typically, 30% is fine
  - if you have very little data, you may want to consider adding more corruption.

Color Constancy:
This picture has no red pixels.

Why do we see red in this picture?

“The Dress”

- "**The dress**" is a photo that became a viral Internet picture on 26 February 2015, when viewers disagreed over whether the item of clothing depicted was “*black and blue*” or “*white and gold*”.
- The phenomenon revealed differences in human color perception which have been the subject of ongoing scientific investigation in neuroscience and vision science, with a number of papers published in peer-reviewed science journals.

Course Summary

- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- **Types of Models**
  - Generative vs Discriminative Models

Discriminative vs Generative Models

- Discriminative Models
  - **Discriminative models** learn the (hard or soft) **boundary** between classes

- Generative Models
  - providing **a model of how the data is actually generated.**
  - model the **distribution** of individual classes
  - often outperforms discriminative models on smaller datasets because their **generative assumptions place some structure on your model that prevent overfitting.**

Generative Adversarial Networks (GANs)

- They have recently been used
  - to model very basic patterns of motion in video.
  - to reconstruct 3D models of objects from images.
  - to improve astronomical images.
- Downsides:
  - the images are generated off some arbitrary noise. If you wanted to generate a picture with specific features, there's no way of determining which initial noise values would produce that picture, other than searching over the entire distribution.
  - It only discriminates between "real" and "fake" images. There's no constraints that an image of a cat has to look like a cat. This leads to results where there's no actual object in a generated image, but the style just looks like picture.
- How to solve these problems?

Course Summary

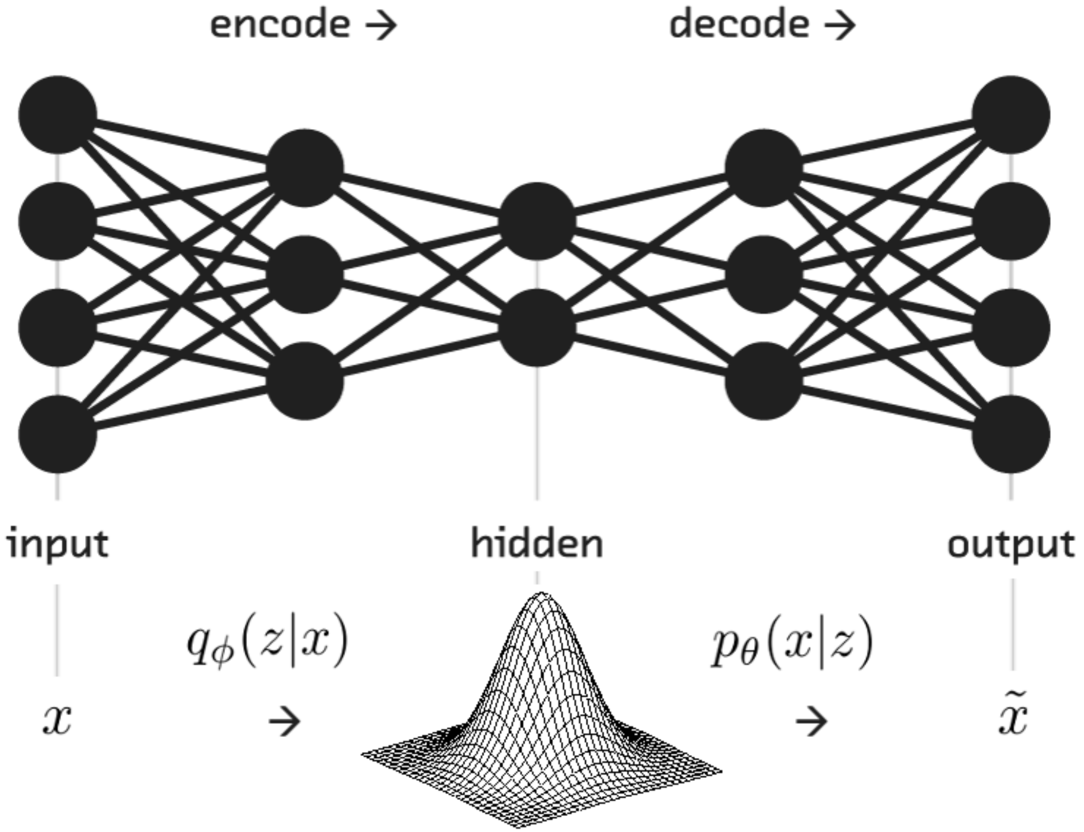
- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

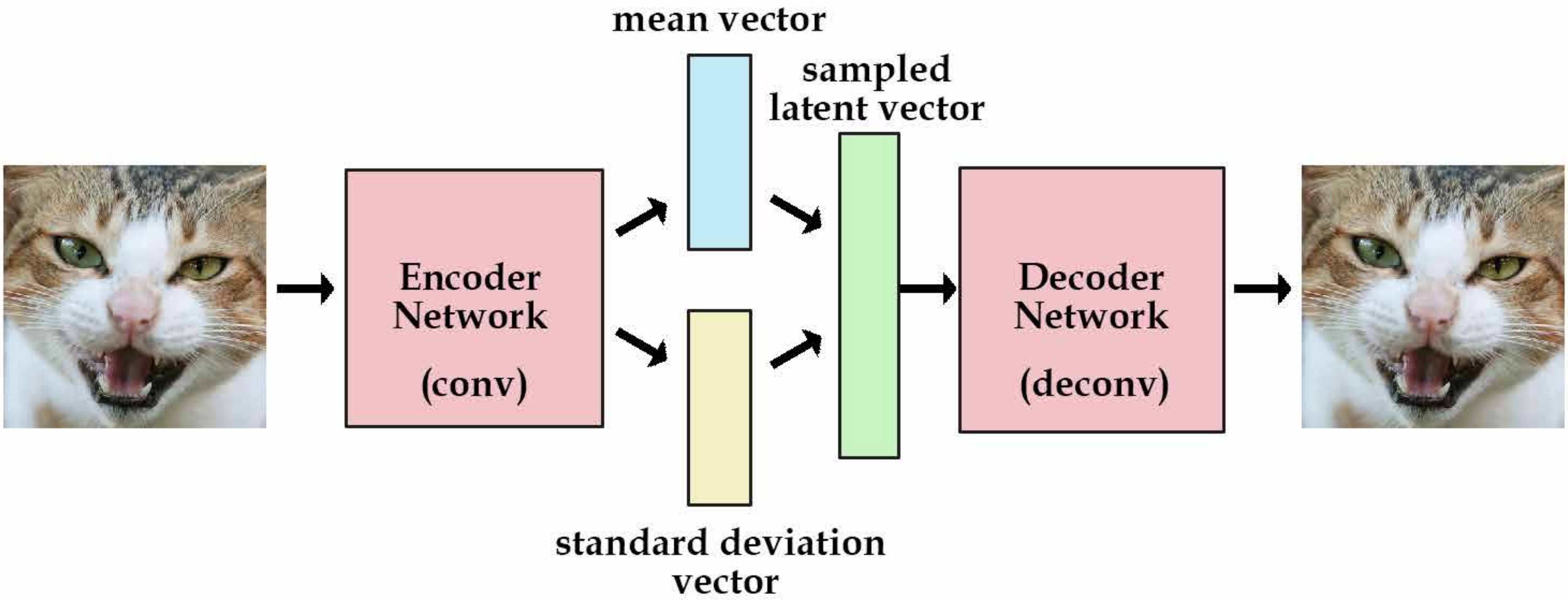
- **Types of Models**
  - Generative vs Discriminative Models

Variational Auto-encoders (VAEs)

the lovechild of Bayesian inference and unsupervised deep learning

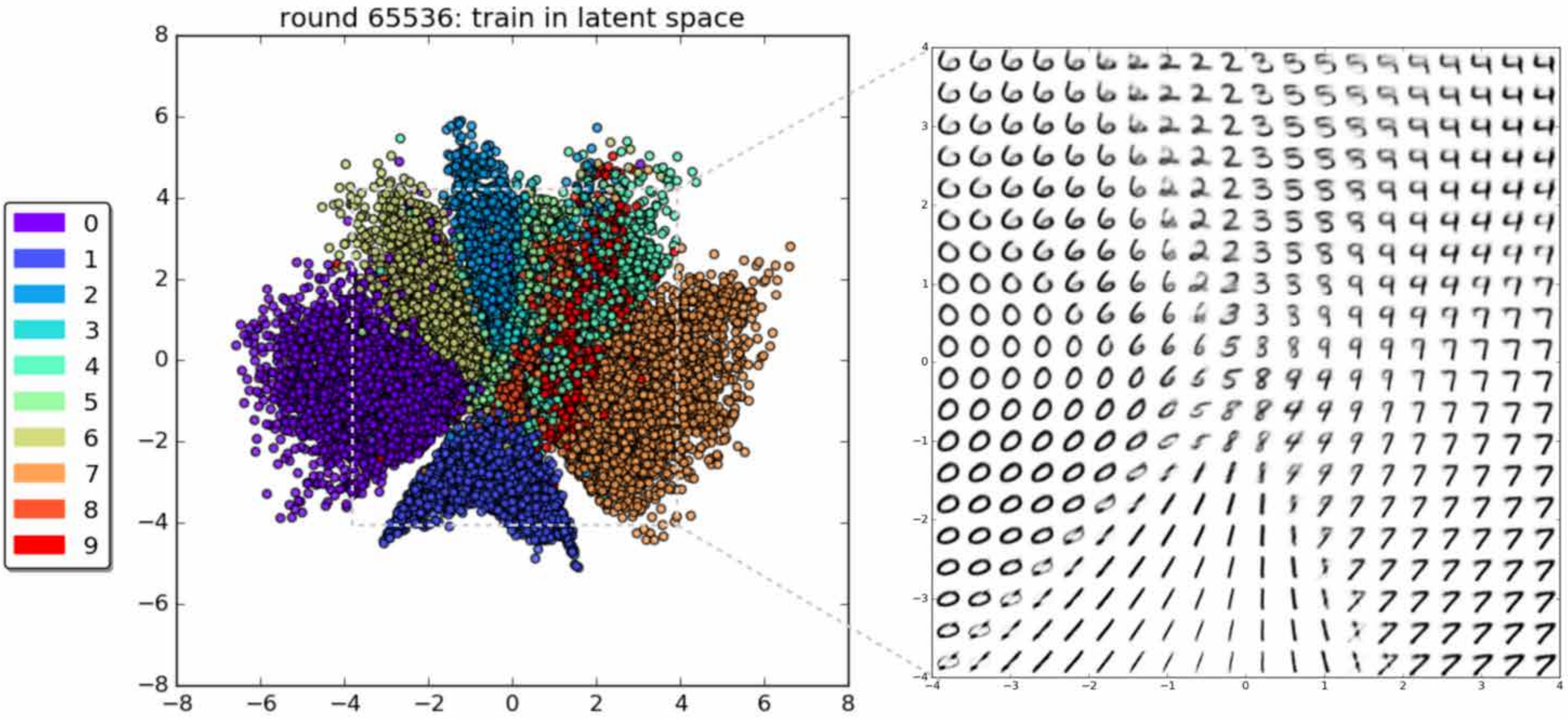


Variational Auto-encoders (VAEs)
re-parameterization trick



Variational Auto-encoders (VAEs) - MNIST 2D embedding

how optimizing the encoder and decoder in tandem enables efficient pairing of inference and generation



Course Summary

- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- **Types of Models**
  - Generative vs Discriminative Models

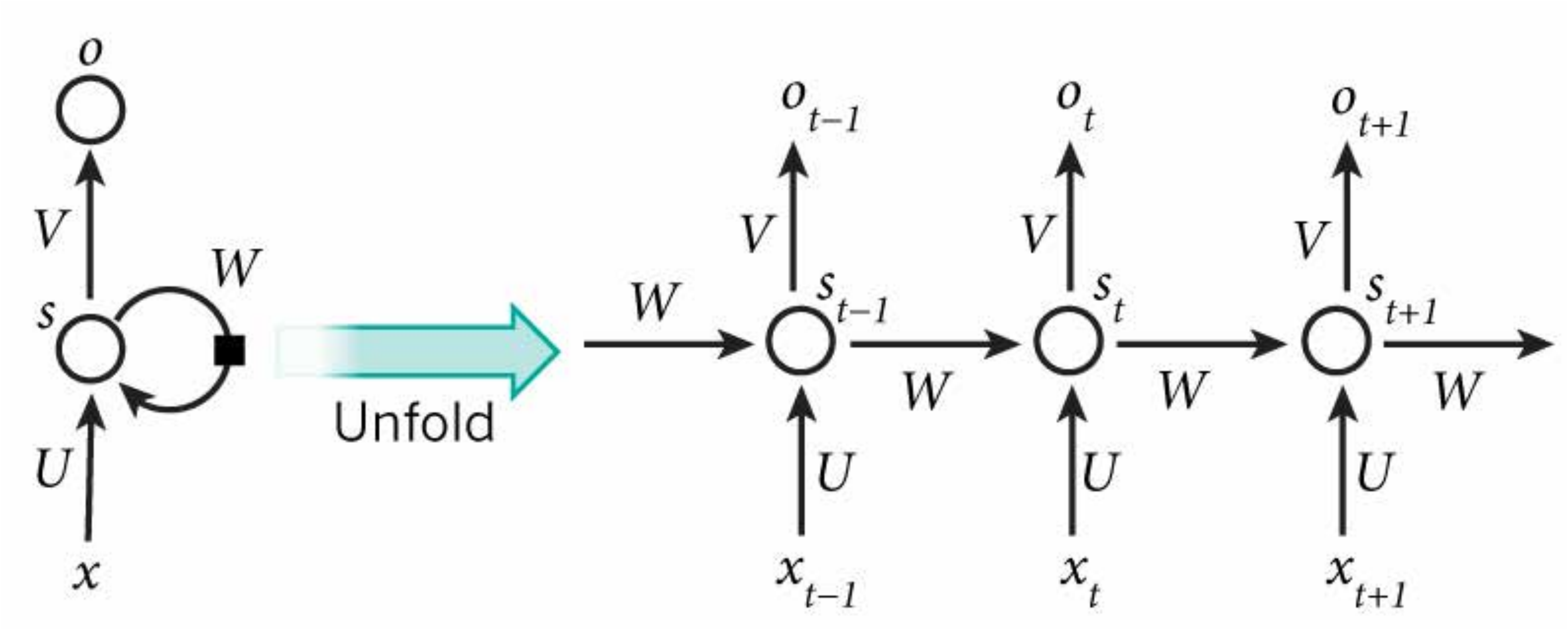
Course Summary

- **Neural Networks**
  - Capacity of a neuron and neural network
    - Perceptron: A linear neuron
    - Activation functions: required non-linearity
  - Learning: Training a Neural Network
    - Cost function for regression
    - Cost function for classification
    - Finding weights: The Learning Algorithm
  - Training Problems, Solutions
    - Underfitting / Overfitting / Model Complexity
    - Bias / Variance of a trained network
    - Simplifying the Model (ConvNet)
      - Local Receptive Fields
      - Weight Sharing
    - Regularization
  - Recurrent Neural Networks
    - Word Embeddings

- **Types of Machine Learning**
  - Supervised Machine Learning:
    - Regression
    - Classification
      - Two-class classification – Logistic Regression
      - Multi-class classification – Multinomial Logistic Regression
  - Unsupervised Machine learning:
    - Principal Component Analysis (PCA)
    - Standard Auto-encoders
    - Deep (stacked) Auto-encoders
    - Convolutional Auto-encoders
    - Denoising Auto-encoders
    - Variational Auto-encoders
  - Reinforcement Learning

- **Types of Models**
  - Generative vs Discriminative Models

Recurrent NN and Unfolding



By unrolling we simply mean that we write out the network for the complete sequence. For example, if the sequence we care about is a sentence of 5 words, the network would be unrolled into a 5-layer neural network, one layer for each word.

Word Embeddings: How to represent words?

1 of k encoding:

- What we do instead is generate one boolean column for each category. Only one of these columns could take on the value 1 for each sample.

Sample	Category	Numerical
1	Human	1
2	Human	1
3	Penguin	2
4	Octopus	3
5	Alien	4
6	Octopus	3
7	Alien	4

Sample	Human	Penguin	Octopus	Alien
1	1	0	0	0
2	1	0	0	0
3	0	1	0	0
4	0	0	1	0
5	0	0	0	1
6	0	0	1	0
7	0	0	0	1