Machine Learning

Prof. Sergei Gleyzer

Lecture

PH451, PH551

Feb. 26, 2024

Announcements

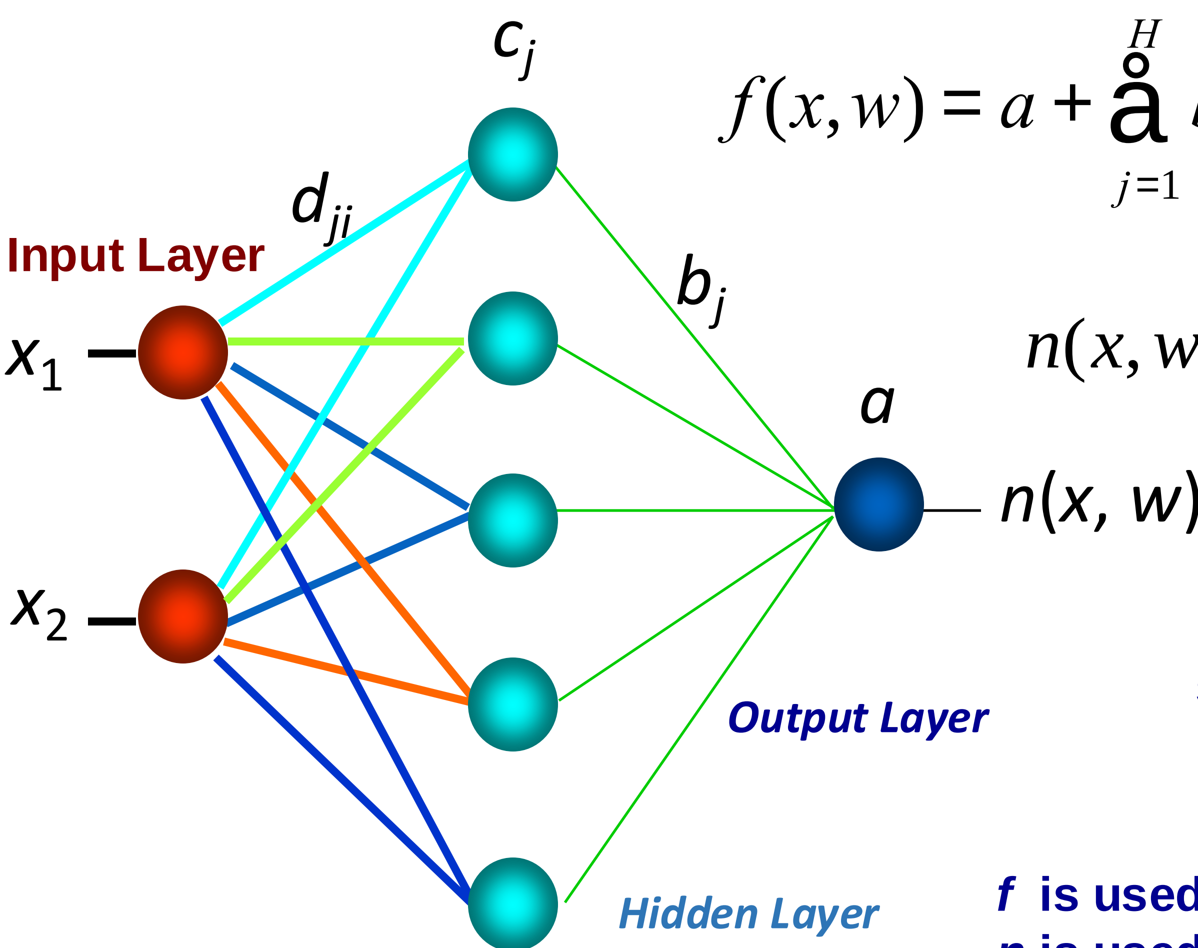
- **Hands-on #5 – due next Thursday**

- **Extra Credit Opportunity:**
  - **5 minute videos**

Outline

- **Training Neural Networks**

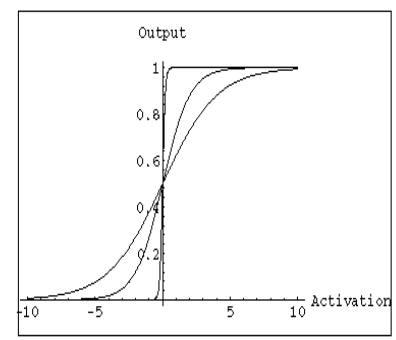
Graphical Representation



$$f(x,w) = a + \sum_{j=1}^{H} b_j \tanh\left[c_j + \sum_{i=1}^{I} d_{ji} x_i\right]$$

$$n(x,w) = \frac{1}{1+\exp[-f(x,w)]}$$

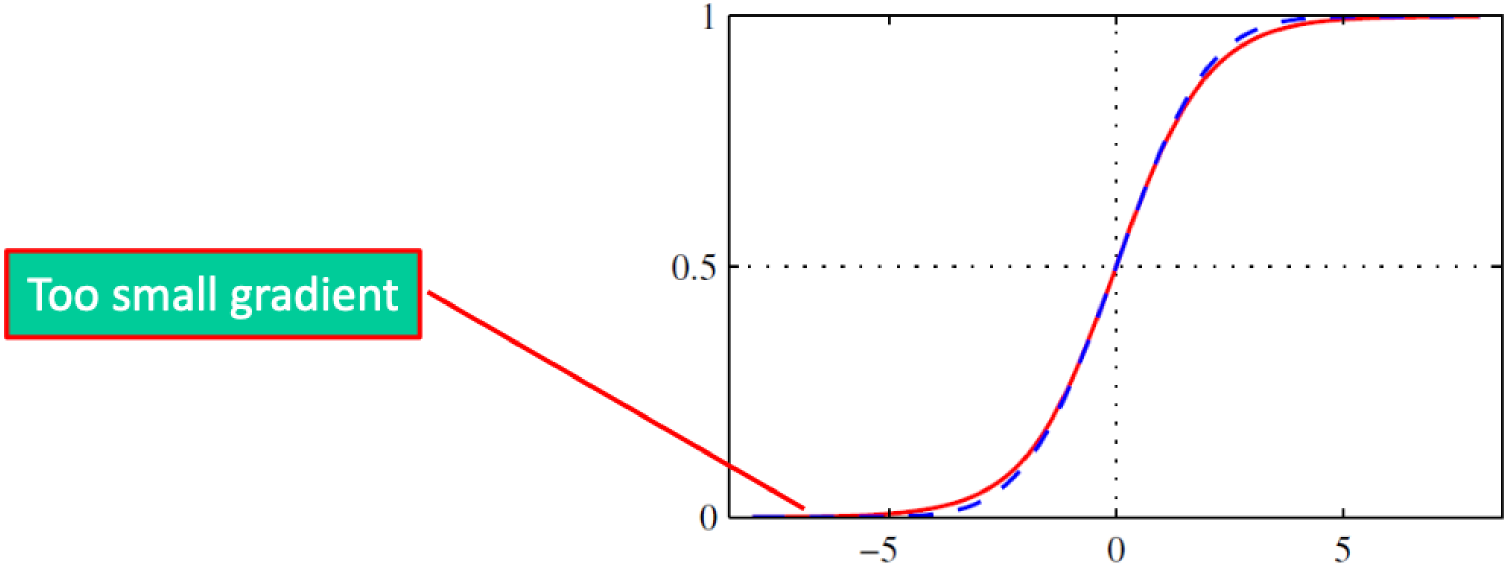
sigmoid



f **is used for regression**
n **is used for classification**
w* = *a*, *b*, *c*, *d

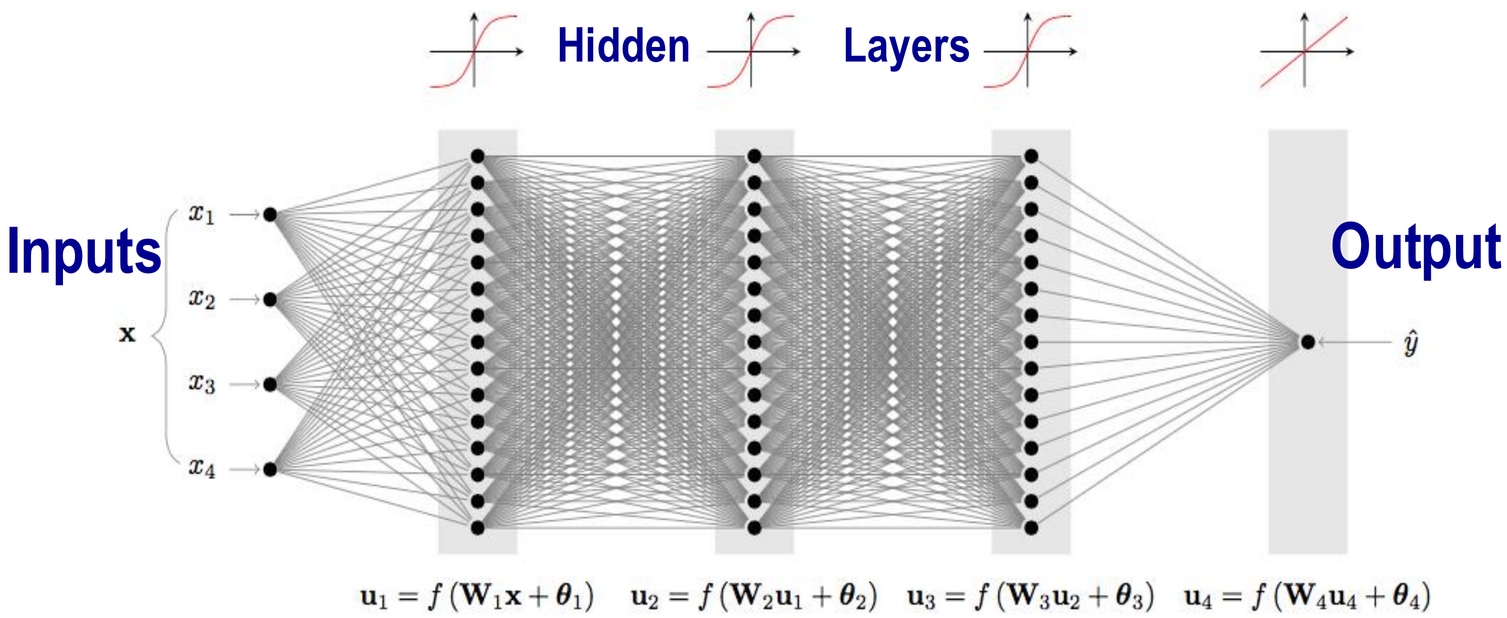
Recap: Vanishing Gradients

- Problem with sigmoid: saturation



Deep Learning

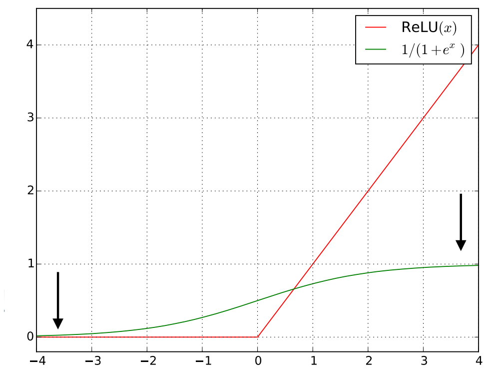
Deep Neural Networks (DNN) achieve significant performance improvements



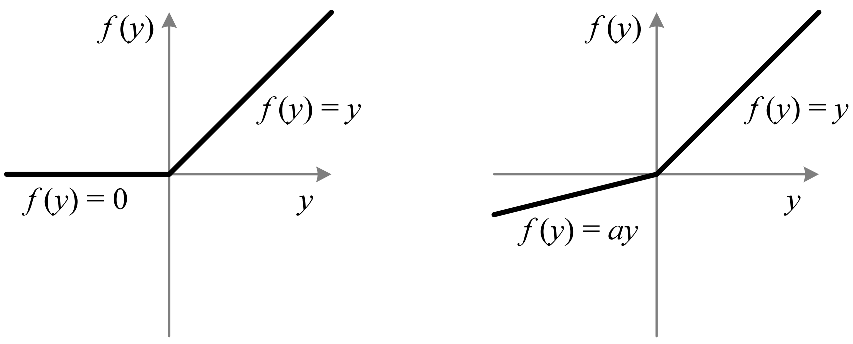
ReLU

Rectified Linear Unit (ReLU)

- Rectified neuron
- Faster training convergence
  - Better solutions than sigmoids
    - Vanishing gradients



  - Trained by back-propagation



ReLU and Parametric PReLU

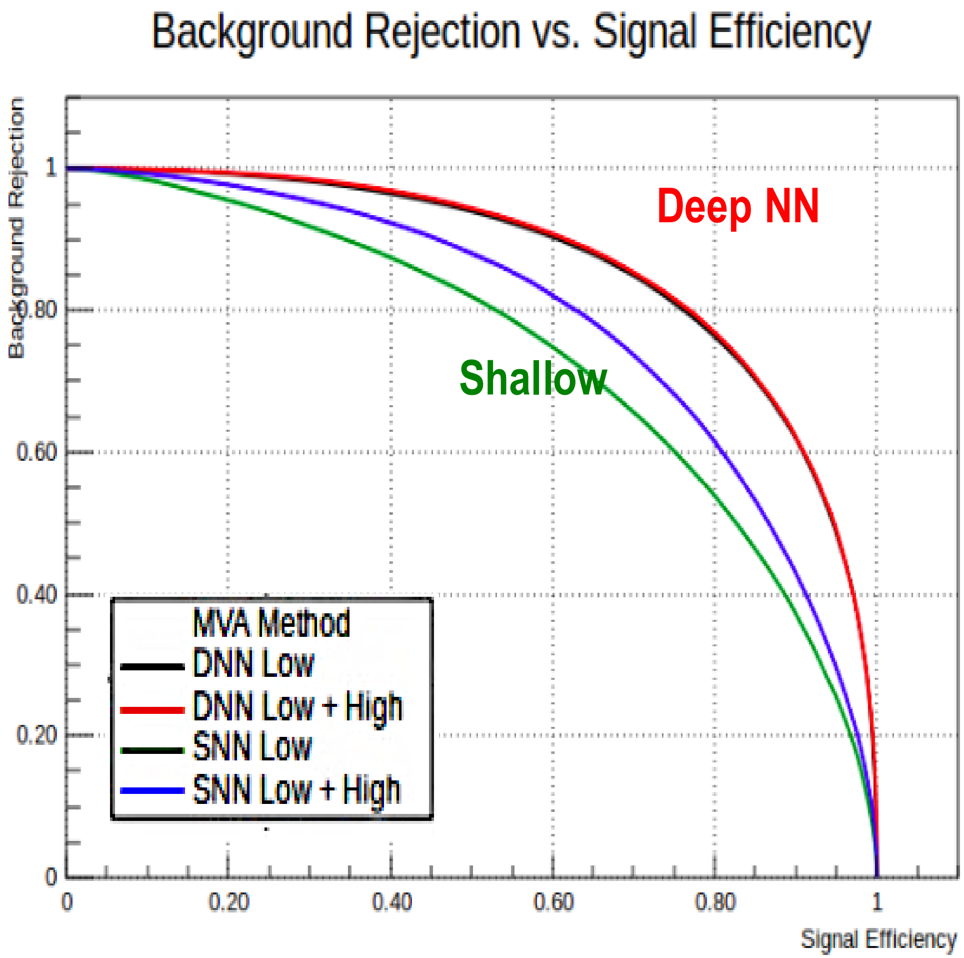
Batch Normalization

- **Another way of dealing with vanishing gradient problem without dropping sigmoid-like activations**
  - Ioffe and Szegedy, 2015
  - Learn the optimal scale (and mean) of each layer over mini-batches
  - Standardize inputs, rescale and offset

Exploding Gradients

- **Gradients can sometimes explode**
  - get larger and larger leading to divergence
  - can happen with Recurrent Networks
- **Solution: clip gradients during back-propagation**
  - i.e. impose a maximum gradient threshold
  - How do you know what your gradients are doing?
    - **TensorBoard**
    - **Weights and Biases (W&B)**

Deep Feature Extraction



Deep neural networks capable of feature extraction (implicit and explicit)

Goal is to find relevant and remove (or "forget") irrelevant information

Back to Human Learning

- **One of the key elements of learning and successful brain function is forgetting. Why?**
- Like "garbage collection", that often occurs during sleep, our brains process the data and **forget or ignore the irrelevant**
  - If we don't do this, we will be overwhelmed with information
  - This is a key idea that also **applies to deep neural networks**

Transfer Learning

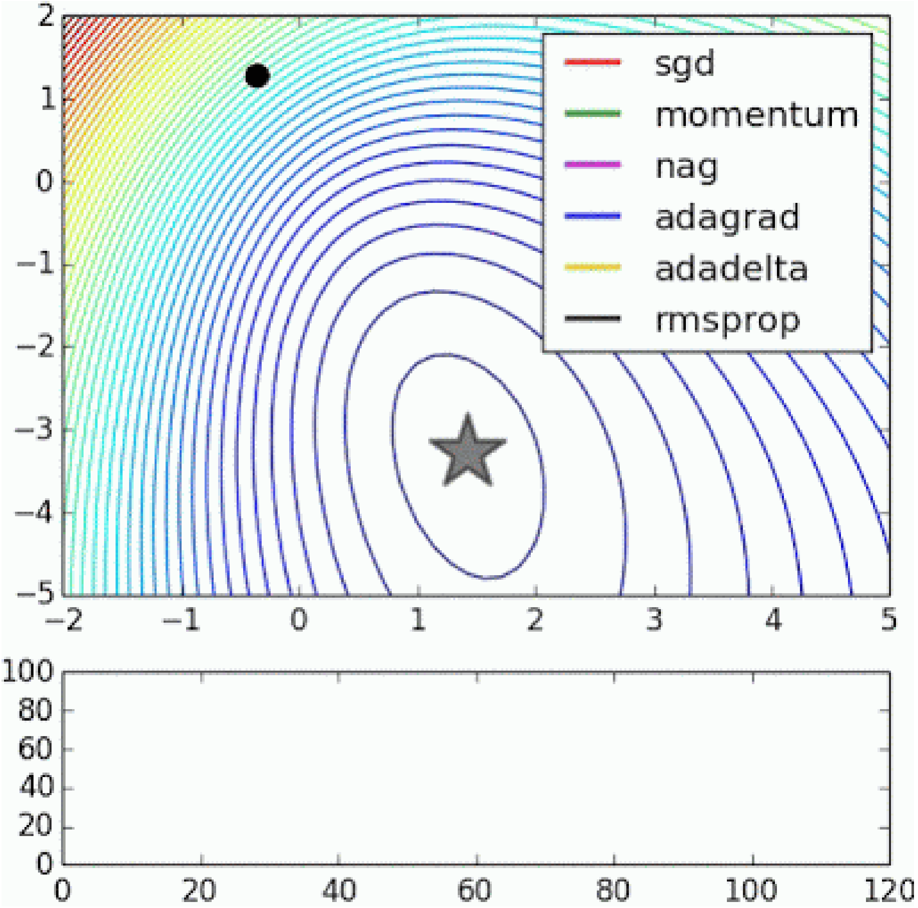
- **In some situations, it is better to not start from scratch – transfer learning**
- **If there is a successful model for a related task, you can:**
  - Start with this model
  - Freeze the early layers
  - Modify Output
  - Train the later layers

Optimizers

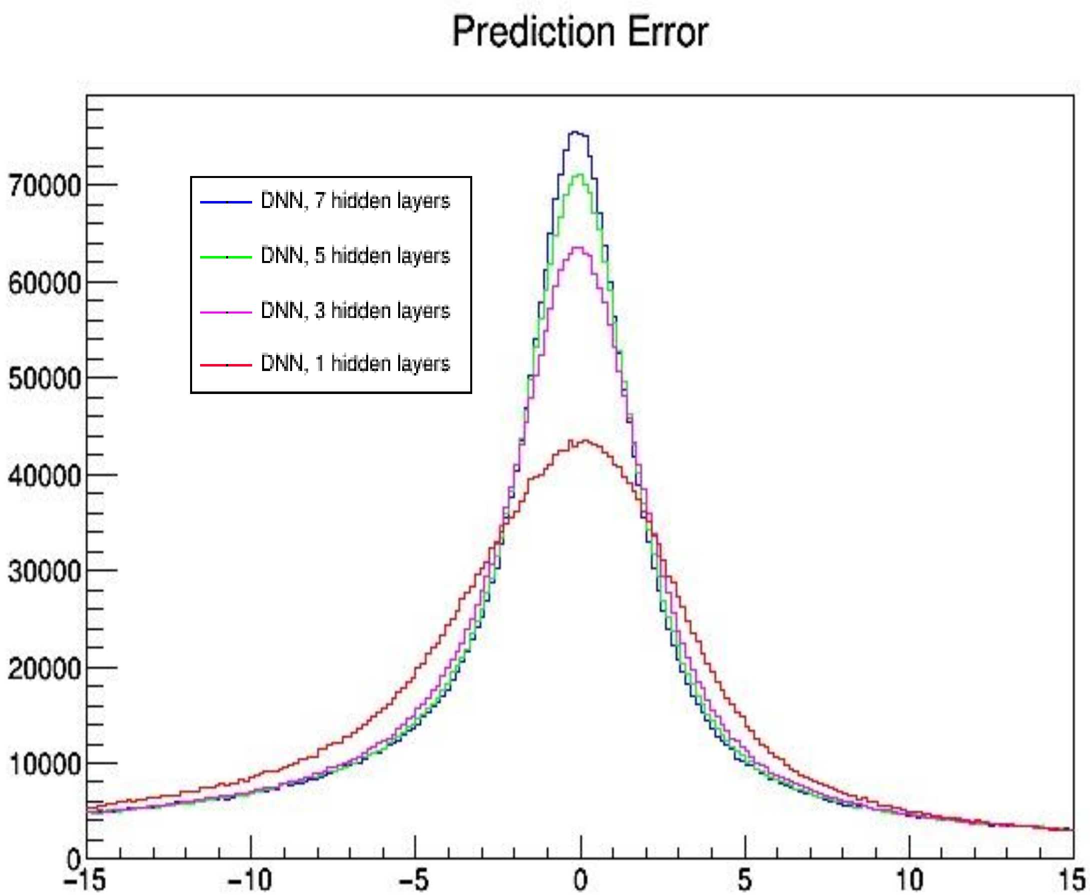
Goal: improve gradient descent

- **Momentum optimization**
  - Instead of regular but slow updates in GD
  - Add a momentum term to dampen oscillations
    - "Ball rolling down the hill"
- **Adagrad:** Scale down gradients (decay the learning rate) faster for steeper dimensions
- **RMSProp/AdaDelta:** Hinton et al., use only gradients from recent iterations
- **Adam:** Also keep exponentially decaying average of past gradients (like momentum)
  - "Ball rolling down the hill with friction"

Gradient Descent Optimizers

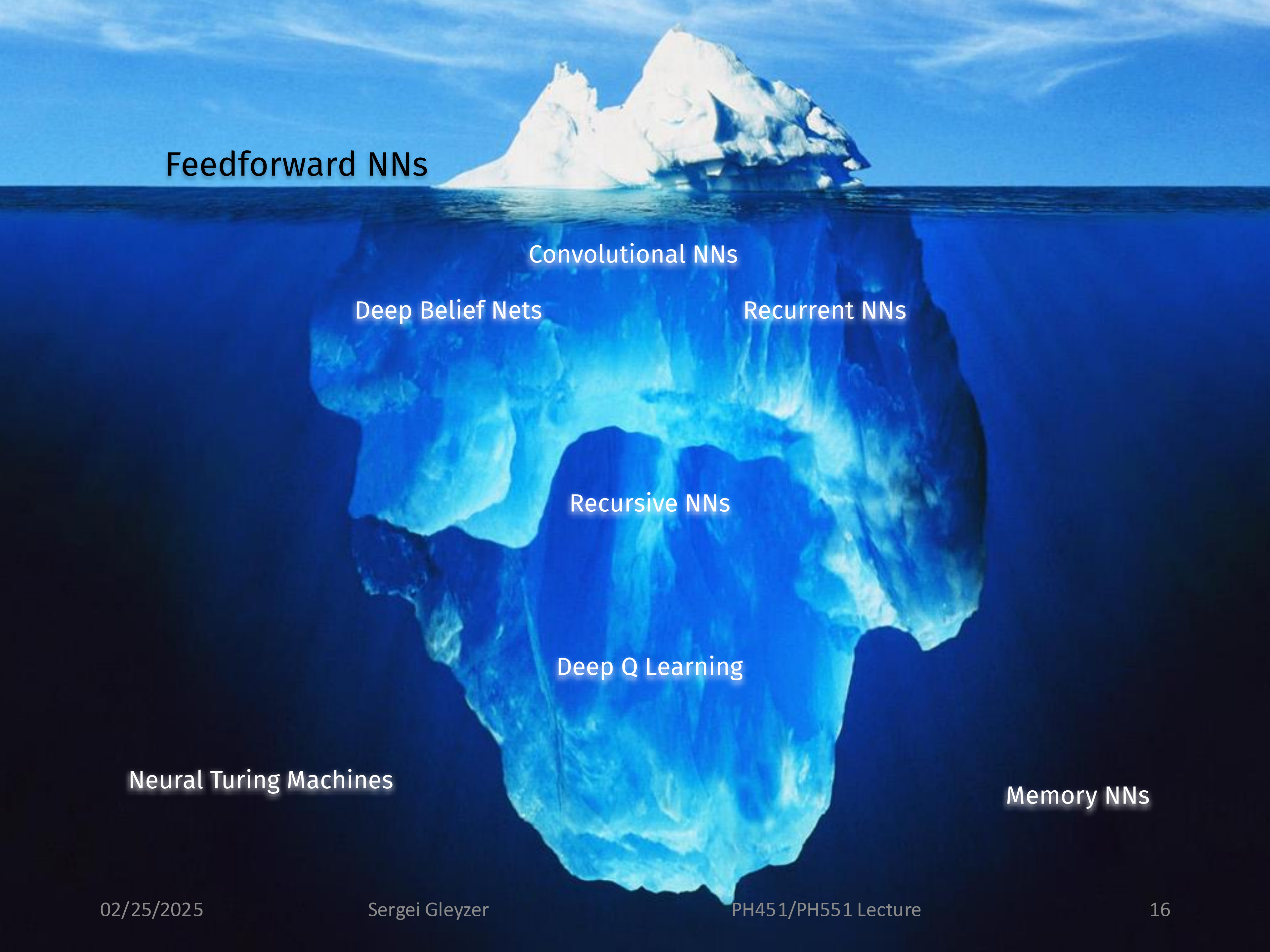


Deep Regression



Going deep also improves regression

Choice of Loss Function again important to match the data


Feedforward NNs
Convolutional NNs
Deep Belief Nets
Recurrent NNs
Recursive NNs
Deep Q Learning
Neural Turing Machines
Memory NNs