CS 4824/ECE 4424: Deep Neural Networks I

Acknowledgement:
Many of these slides are derived from Tom Mitchell, Pascal Poupart, Pieter Abbeel, Eric Eaton, Carlos Guestrin, William Cohen, and Andrew Moore.

Deep Neural Networks

- **DNN**: neural network with many hidden layers
- **Advantage**: highly expressive
- **Challenges**:
  - How to effectively train a deep neural network?
  - How to avoid overfitting?

Expressiveness

- Neural networks with one hidden layer of sigmoid/tanh units can approximate arbitrarily closely neural networks with several layers of sigmoid/hyperbolic units

- However, as we increase the number of layers, the number of units needed may decrease exponentially (with the number of layers)

Example – Parity Function

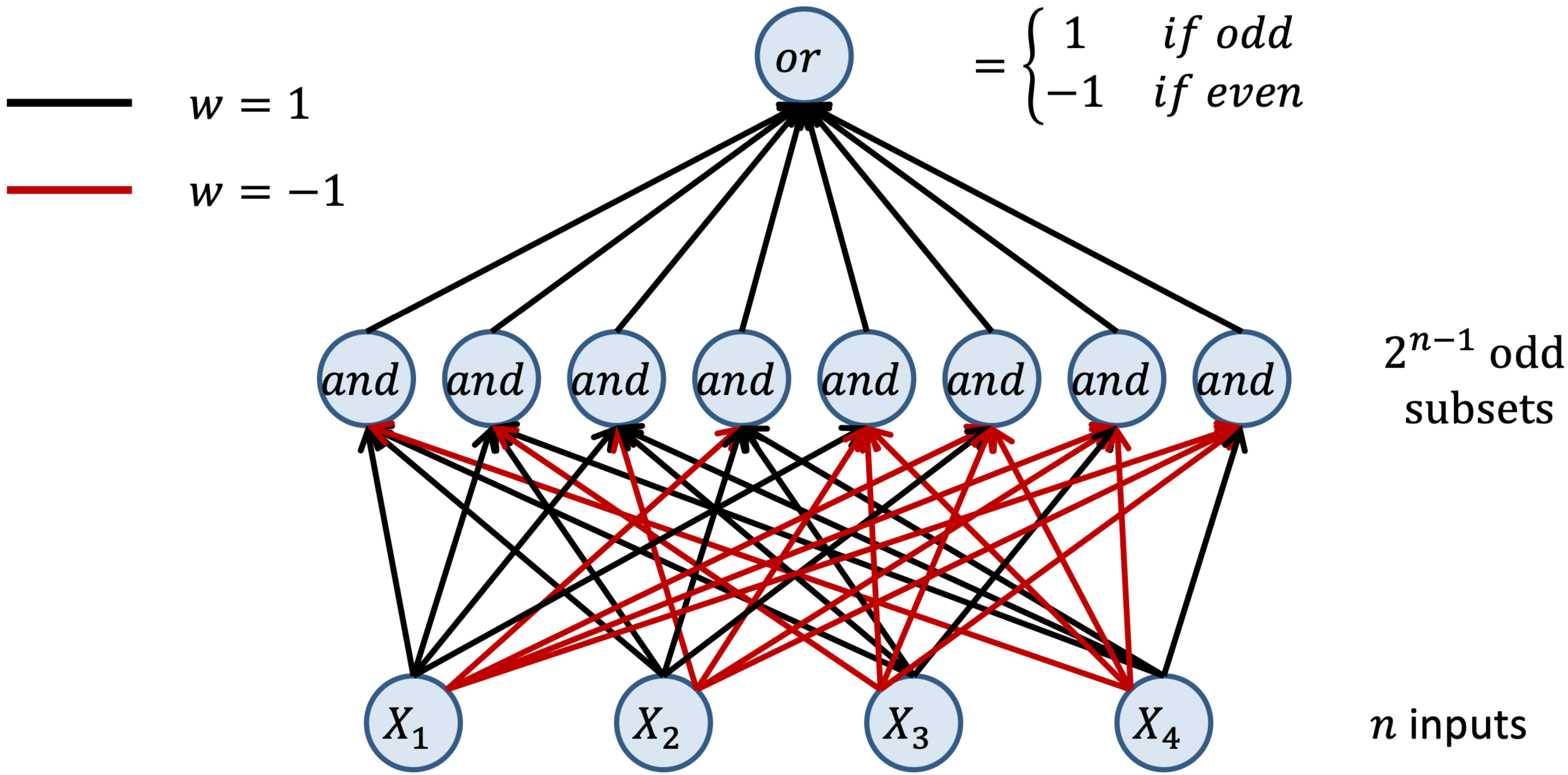
- Odd or even $\begin{cases} 1 & if\ odd \\ -1 & if\ even \end{cases}$

- Possible odd combinations

X1	X2	X3	X4

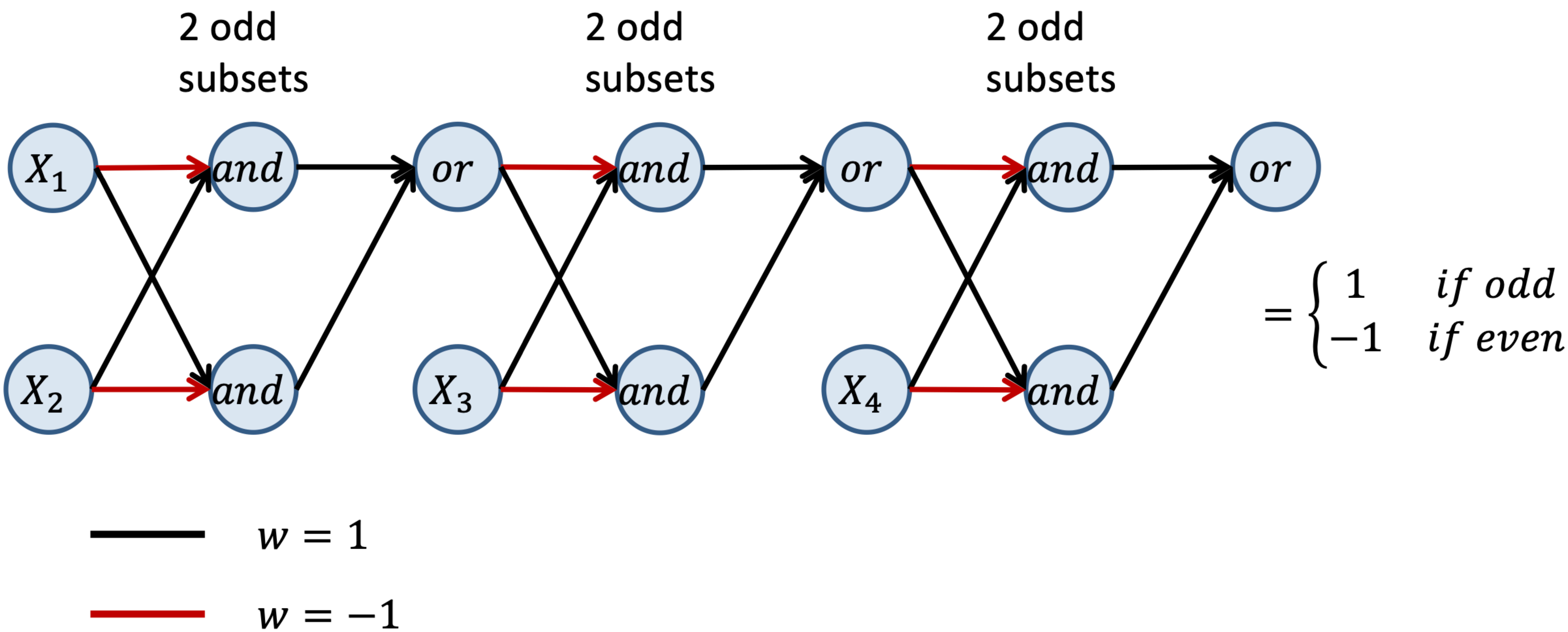
Example – Parity Function

- Single layer of hidden nodes

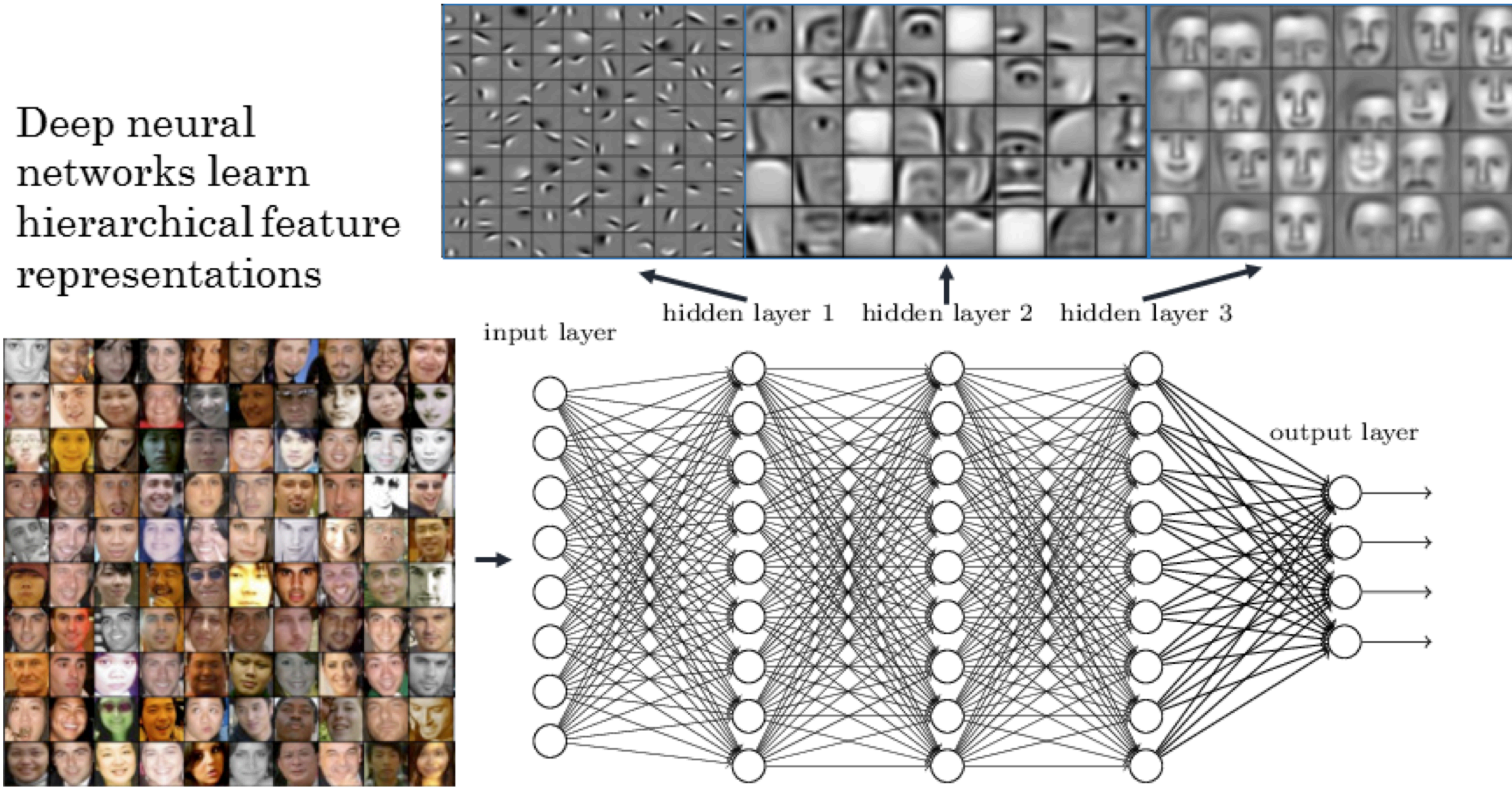


Example – Parity Function

- 2n – 2 layers of hidden nodes



The power of depth (practice)



- Challenge: how to train deepNNs?

Gradient-based training

- Efficient gradient computation: linear in number of weights

- Convergence:
  - Slow convergence (linear rate)
  - May get trapped in local optima

Slow Convergence

- **Issue**: gradient is not always ideal
- Illustration:

Adaptive Gradients

- **Idea**: adjust the learning rate of each dimension separately

- AdaGrad:
  - $r_t \leftarrow r_{t-1} + \left(\frac{\partial E_n}{\partial w_{ji}}\right)^2$ (sum of squares of partial derivative)

  - $w_{ji} \leftarrow w_{ji} - \frac{\eta}{\sqrt{r}_t}\frac{\partial E_n}{\partial w_{ji}}$ (update rule)

- **Problem**: learning rate $\frac{\eta}{\sqrt{r}_t}$ decays too quickly

RMSprop

- **Idea**: divide by root mean square (RMS) (instead of square root of the sum) of partial derivatives

- **RMSprop**

$$r_t \leftarrow \alpha r_{t-1} + (1-\alpha)\left(\frac{\partial E_n}{\partial w_{ji}}\right)^2 \quad (0 \leq \alpha \leq 1)$$

- $$w_{ji} \leftarrow w_{ji} - \frac{\eta}{\sqrt{r}_t}\frac{\partial E_n}{\partial w_{ji}} \quad \text{(update rule)}$$

- **Problem**: gradient lacks momentum

Adaptive Moment Estimation

- **Idea**: replace gradient by its moving average to induce momentum

- **Adam**:

$$r_t \leftarrow \alpha r_{t-1} + (1-\alpha)\left(\frac{\partial E_n}{\partial w_{ji}}\right)^2 \quad (0 \leq \alpha \leq 1)$$

$$s_t \leftarrow \beta s_{t-1} + (1-\beta)\left(\frac{\partial E_n}{\partial w_{ji}}\right) \quad (0 \leq \beta \leq 1)$$

- $$w_{ji} \leftarrow w_{ji} - \frac{\eta}{\sqrt{r_t}} s_t \qquad \text{(update rule)}$$

Challenges in Deep Neural Networks

- Deep neural networks often suffer from vanishing gradients
- High expressivity of deep neural networks increases the risk of overfitting