Applied Machine Learning

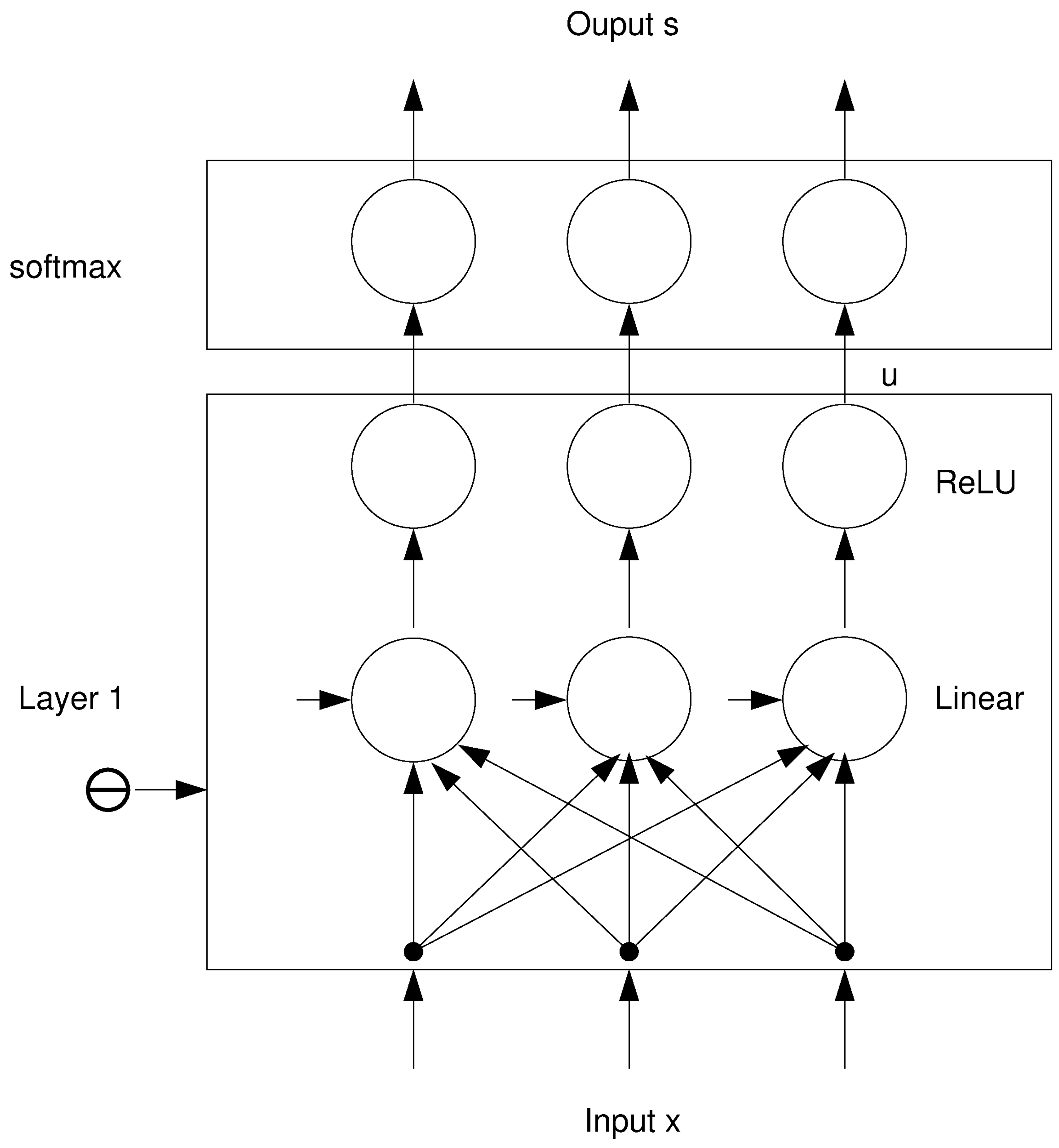
Deep Neural Networks - Loss and Gradient

Deep Neural Networks - Loss and Gradient

- Loss and gradient for a Neural Network with 2 layers
- Loss and gradient for a Deep Neural Network

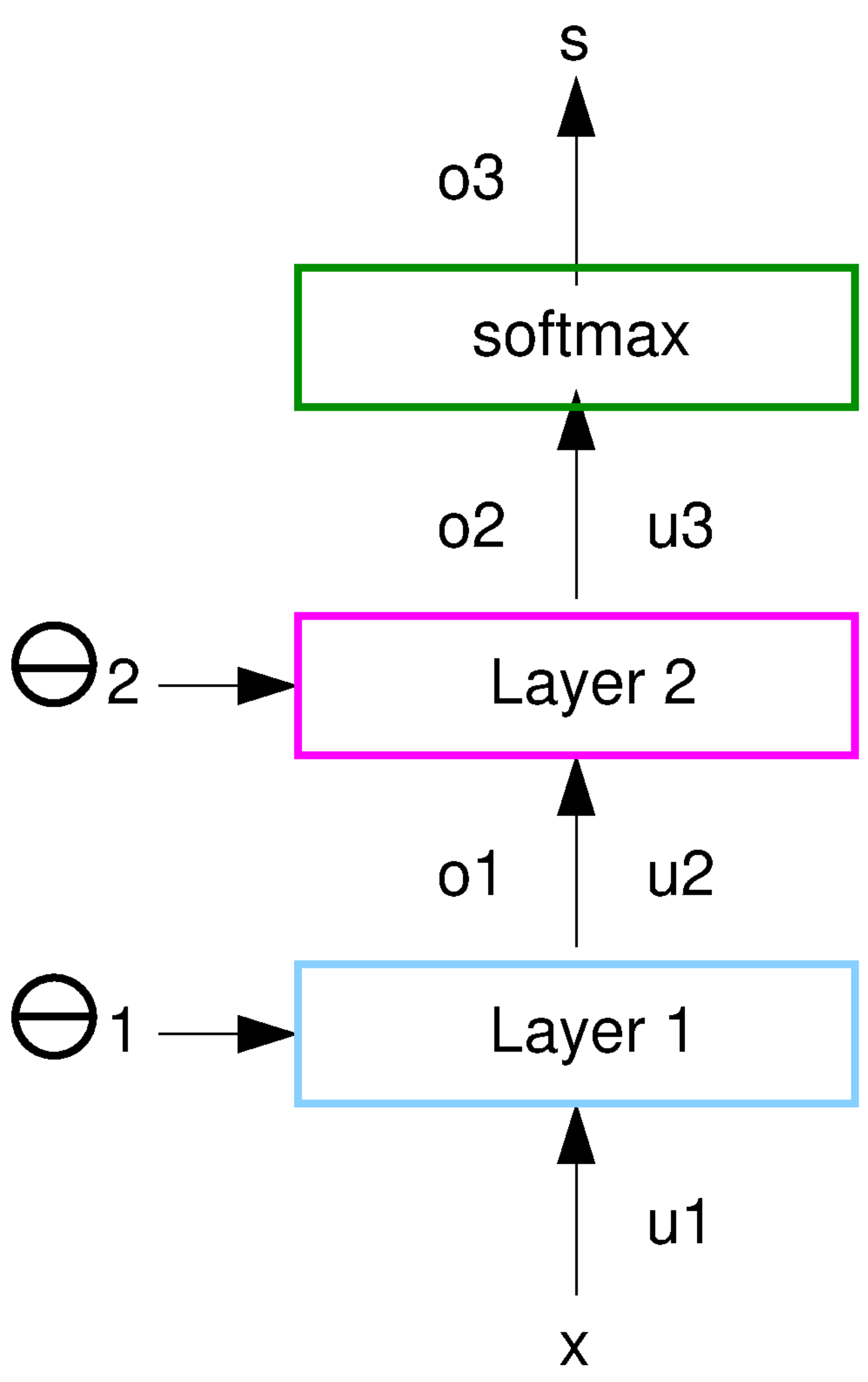
Neural Networks with Multiple Layers

- Structure
  - Layers
    - Input layer, Hidden layers, Output layer
  - Inter-layer Connections
    - Input of layer i: $\mathbf{u}^{(i)}$
    - Output of layer i: $\mathbf{o}^{(i)}$
      - input of layer $i+1$: $\mathbf{u}^{(i+1)} = \mathbf{o}^{(i)}$
    - Weights: $\theta^{(i)}$
    - Functions: $\mathbf{o}^{(i)} = f(\mathbf{u}^{(i)}, \theta^{(i)})$
      - ReLU
      - Softmax
      - others



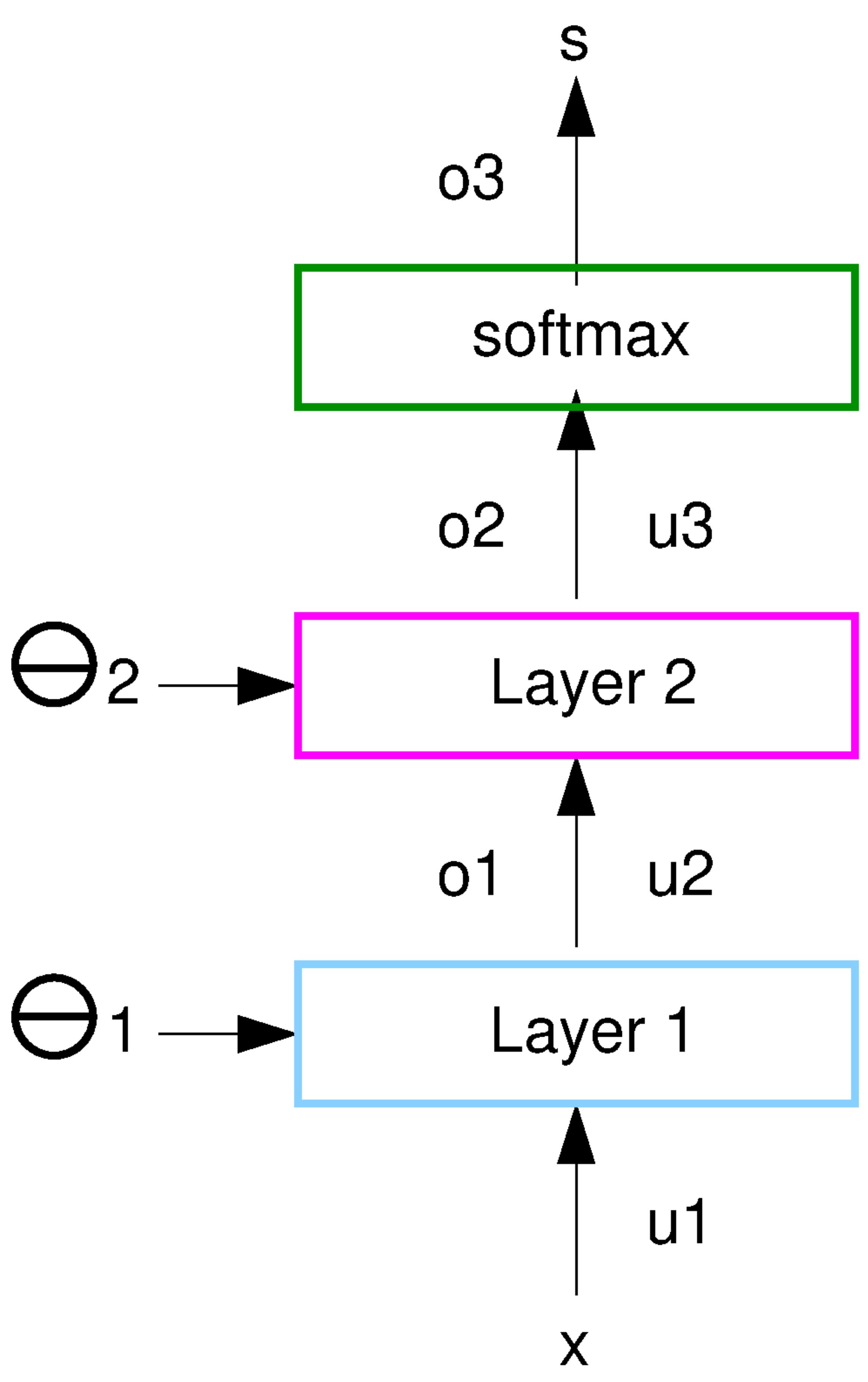
2-Layer Neural Network

- Output: $\mathbf{y} = \mathbf{s}(\mathbf{o}^{(2)}(\mathbf{o}^{(1)}(x, \theta^{(1)}), \theta^{(2)}))$
- Softmax:
  - input: $\mathbf{u}^{(3)} = \mathbf{o}^{(2)}$
  - output: $\mathbf{y} = \mathbf{s}(\mathbf{u}^{(3)})$
- Layer 2:
  - input: $\mathbf{u}^{(2)} = \mathbf{o}^{(1)}$
  - parameters: $\theta^{(2)}$
  - output: $\mathbf{o}^{(2)}(\mathbf{u}^{(2)}, \theta^{(2)})$
- Input: Layer 1:
  - input: $\mathbf{u}^{(1)} = \mathbf{x}$
  - parameters: $\theta^{(1)}$
  - output: $\mathbf{o}^{(1)}(\mathbf{u}^{(1)}, \theta^{(1)})$



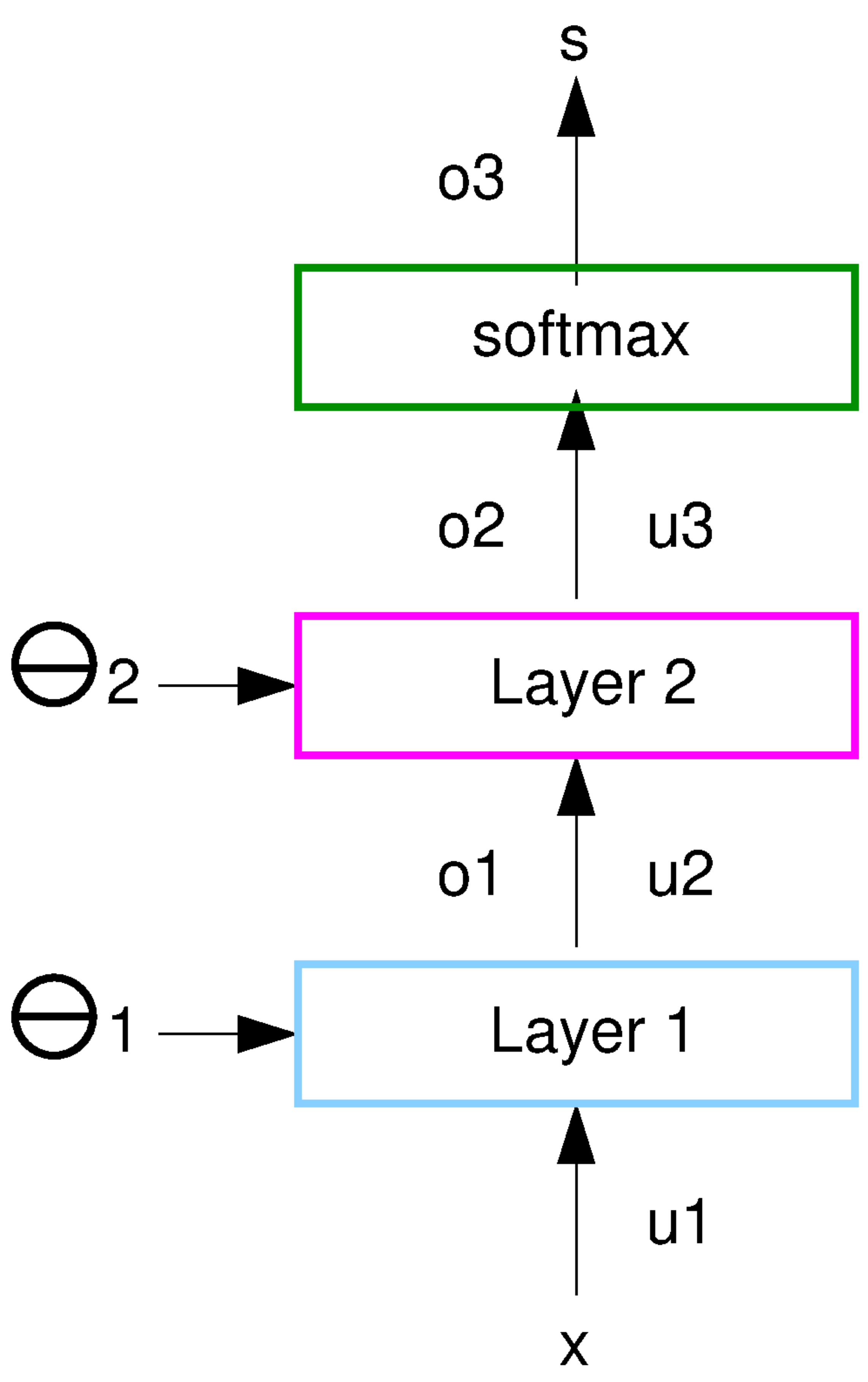
Gradient for 2-Layer NN

- Analyze one change at a time from the output and backwards
- Softmax Layer:
  - Loss: $L(\mathbf{s})$
  - changes in $\mathbf{s}$ => Loss L
- Layer 2:
  - Loss: $L(\mathbf{s}(\mathbf{o}^{(2)}(\mathbf{u}^{(2)}, \theta^{(2)})))$
    - changes in $\theta^{(2)}$ => $\mathbf{o}^{(2)}$ => $\mathbf{s}$ => Loss L
- Layer 1:
  - Loss: $L(\mathbf{s}(\mathbf{o}^{(2)}(\mathbf{o}^{(1)}(\mathbf{u}^{(1)}, \theta^{(1)}), \theta^{(2)})))$
    - changes in $\theta^{(1)}$ => $\mathbf{o}^{(1)}$ => $\mathbf{o}^{(2)}$ => $\mathbf{s}$ => Loss L



Gradient for 2-Layer NN

- Vector function $\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{x}) \end{bmatrix}$ vector variable $\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$
- Jacobian matrix: $\mathbf{J}_{\mathbf{f};\mathbf{x}} = \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_m} \end{bmatrix} = \begin{bmatrix} \nabla f_1(\mathbf{x})^\top \\ \vdots \\ \nabla f_n(\mathbf{x})^\top \end{bmatrix}$
- Softmax Layer: Loss: $L(\mathbf{s})$
  - changes in $\mathbf{s}$ => Loss L: $\nabla_{\mathbf{s}} L$
- Layer 2: Loss: $L(\mathbf{s}(\mathbf{o}^{(2)}(\mathbf{u}^{(2)}, \theta^{(2)})))$
  - changes in $\theta^{(2)}$ => $\mathbf{o}^{(2)}$ => $\mathbf{s}$ => Loss L
    $\nabla_{\theta^{(2)}} L = \nabla_{\mathbf{s}} L \times \mathbf{J}_{\mathbf{s};\mathbf{o}^{(2)}} \times \mathbf{J}_{\mathbf{o}^{(2)};\theta^{(2)}}$
- Layer 1: Loss: $L(\mathbf{s}(\mathbf{o}^{(2)}(\mathbf{o}^{(1)}(\mathbf{u}^{(1)}, \theta^{(1)}), \theta^{(2)})))$
  - changes in $\theta^{(1)}$ => $\mathbf{o}^{(1)}$ => $\mathbf{o}^{(2)}$ => $\mathbf{s}$ => Loss L
    $\nabla_{\theta^{(1)}} L = \nabla_{\mathbf{s}} L \times \mathbf{J}_{\mathbf{s};\mathbf{o}^{(2)}} \times \mathbf{J}_{\mathbf{o}^{(2)};\mathbf{o}^{(1)}} \times \mathbf{J}_{\mathbf{o}^{(1)};\theta^{(1)}}$

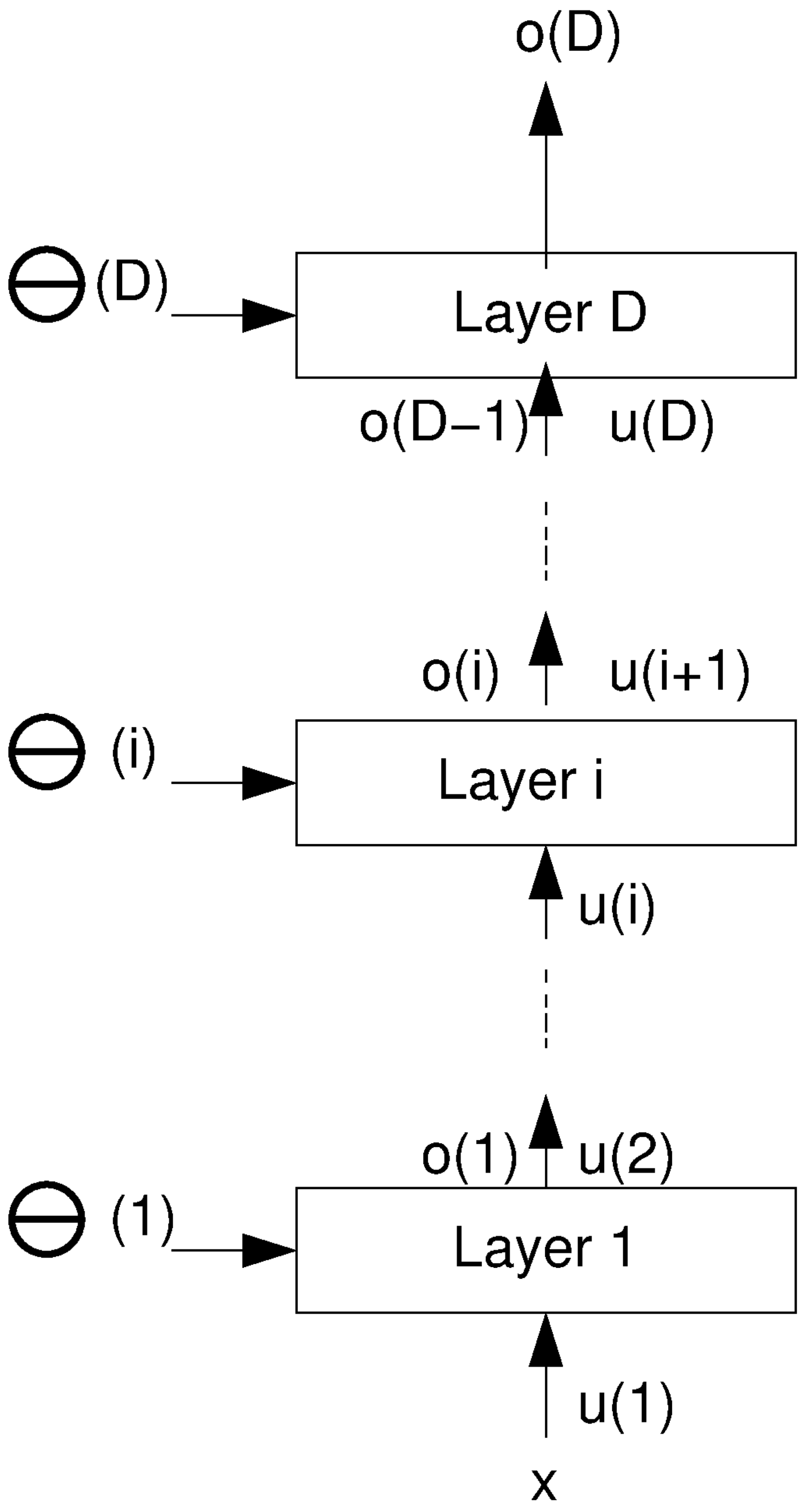


Deep Neural Networks: Multiple Layers

- Stack of D layers

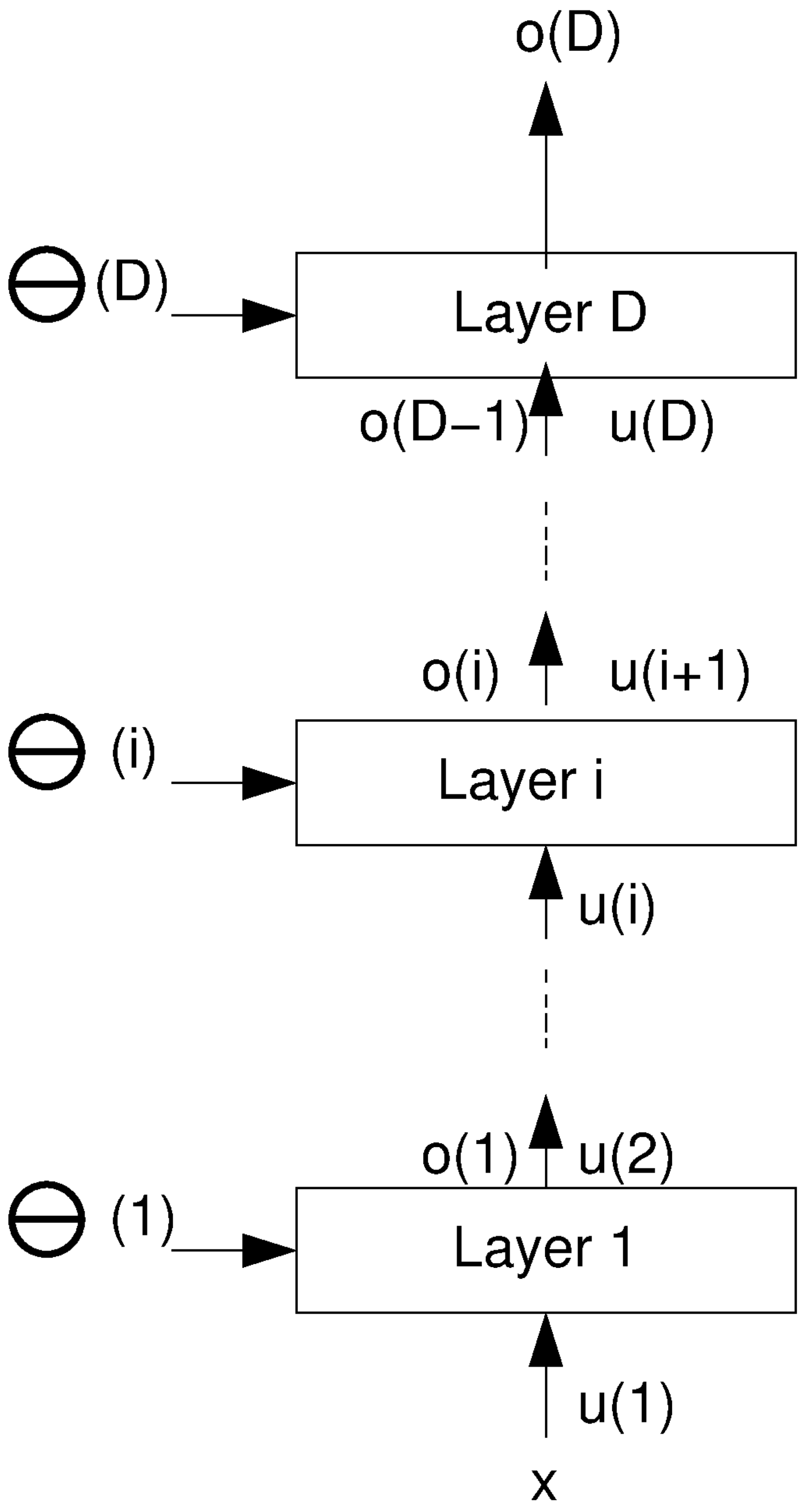
$$\begin{aligned}\mathbf{o}^{(D)} &= \mathbf{o}^{(D)}(\mathbf{u}^{(D)}, \theta^{(D)})\\ \mathbf{u}^{(D)} &= \mathbf{o}^{(D-1)}(\mathbf{u}^{(D-1)}, \theta^{(D-1)})\\ &\vdots\\ \mathbf{u}^{(2)} &= \mathbf{o}^{(1)}(\mathbf{u}^{(1)}, \theta^{(1)})\\ \mathbf{u}^{(1)} &= \mathbf{x}\end{aligned}$$

- Loss: $L(\mathbf{y}, \mathbf{o}^{(D)})$
- Cost function: $\frac{1}{N}\sum_i L(\mathbf{y}_i, \mathbf{o}^{(D)}(\mathbf{x}_i, \theta))$ + regularization term
- Loss for item $\mathbf{x}$,$\mathbf{y}$: $L(\mathbf{y}, \mathbf{o}^{(D)})$: changes in Loss: $\nabla_{\mathbf{o}^{(D)}} L$
- Layer D : changes in Loss at output with respect to $\theta^{(D)}$
  - $\nabla_{\theta^{(D)}} L = \nabla_{\mathbf{o}^{(D)}} L \times \mathbf{J}_{\mathbf{o}^{(D)};\theta^{(D)}}$
- Layer $D-1$: changes in Loss at output with respect to $\theta^{(D-1)}$
  - $\nabla_{\theta^{(D-1)}} L = \nabla_{\mathbf{o}^{(D)}} L \times \mathbf{J}_{\mathbf{o}^{(D)};\mathbf{u}^{(D)}} \times \mathbf{J}_{\mathbf{o}^{(D-1)};\theta^{(D-1)}}$
- Layer i: changes in Loss at output with respect to $\theta^{(i)}$
  - $\nabla_{\theta^{(i)}} L = \nabla_{\mathbf{o}^{(D)}} L \times \mathbf{J}_{\mathbf{o}^{(D)};\mathbf{u}^{(D)}} \times \ldots \mathbf{J}_{\mathbf{o}^{(i+1)};\mathbf{u}^{(i+1)}} \times \mathbf{J}_{\mathbf{o}^{(i)};\theta^{(i)}}$



Deep Neural Networks: Multiple Layers

- Layer i: changes in Loss at output with respect to $\theta^{(i)}$
  - $\nabla_{\theta^{(i)}} L = \nabla_{\mathbf{o}^{(D)}} L \times \mathbf{J}_{\mathbf{o}^{(D)};\mathbf{u}^{(D)}} \times \ldots \mathbf{J}_{\mathbf{o}^{(i+1)};\mathbf{u}^{(i+1)}} \times \mathbf{J}_{\mathbf{o}^{(i)};\theta^{(i)}}$
  - $$\begin{array}{rcl} \mathbf{v}^{(D)} & = & \nabla_{\mathbf{o}^{(D)}} L \\ \nabla_{\theta^{(D)}} L & = & \mathbf{v}^{(D)} \times \mathbf{J}_{\mathbf{o}^{(D)};\theta^{(D)}} \\ & \vdots & \\ \mathbf{v}^{(i)} & = & \mathbf{v}^{(i+1)} \times \mathbf{J}_{\mathbf{o}^{(i+1)};\mathbf{u}^{(i+1)}} \\ \nabla_{\theta^{(i)}} L & = & \mathbf{v}^{(i)} \times \mathbf{J}_{\mathbf{o}^{(i)};\theta^{(i)}} \\ & \vdots & \end{array}$$



Deep Neural Networks - Loss and Gradient

- Loss and gradient for a Neural Network with 2 layers
- Loss and gradient for a Deep Neural Network

Applied Machine Learning

Deep Neural Networks - Loss and Gradient