Back Propagation
Outline

• Logistic regression and perceptron as neural networks
• Likelihood gradient for 2-layered neural network
• General recipe for back propagation
Back Propagation

• Back propagation:
  • Compute hidden unit activations: forward propagation
  • Compute gradient at output layer: error
  • Propagate error back one layer at a time
  • Chain rule via dynamic programming
Logistic Squashing Function

\[ \sigma(x) = \frac{1}{1 + \exp(-x)} \]
Logistic Squashing Function

\[ \sigma(x) = \frac{1}{1 + \exp(-x)} \]

\[ \frac{d \sigma(x)}{d x} = \sigma(x)(1 - \sigma(x)) \]
Multi-Layered Perceptron

\[ h_1 = \sigma(w_{11}^T x) \]

\[ h = [h_1, h_2]^T \]

\[ h_2 = \sigma(w_{12}^T x) \]

\[ p(y|x) = \sigma(w_{21}^T h) \]

\[ p(y|x) = \sigma \left( w_{21}^T \left[ \sigma(w_{11}^T x), \sigma(w_{12}^T x) \right]^T \right) \]
Gradients

\[ p(y|x) = \sigma \left( w_{21}^T \left[ \sigma(w_{11}^T x), \sigma(w_{12}^T x) \right]^T \right) \]

\[ p(y|x) = \sigma(w_{21}^T h) \]

\[ \ell(W) = \sum_{i=1}^{n} \log p(y_i|x_i) \]

\[ \nabla_{w_{21}} \ell = \sum_{i=1}^{n} \frac{1}{p(y_i|x_i)} \times \nabla_{w_{21}} p(y_i|x_i) \]

\[ \nabla_{w_{21}} \ell = \sum_{i=1}^{n} \frac{1}{p(y_i|x_i)} \times \nabla_{w_{21}} \sigma(w_{21}^T h) \]

\[ \nabla_{w_{21}} \ell = \sum_{i=1}^{n} (I(y_i = 1) - \sigma(w_{21}^T h)) \nabla_{w_{21}} w_{21}^T h \]

\[ \nabla_{w_{21}} \ell = \sum_{i=1}^{n} (I(y_i = 1) - \sigma(w_{21}^T h)) h \]
Gradients

\[
p(y|x) = \sigma \left( w_{21}^T \begin{bmatrix} \sigma(w_{11}^T x), \sigma(w_{12}^T x) \end{bmatrix}^T \right)
\]

\[
\nabla_{w_{11}} \mathbb{L} = \sum_{i=1}^{n} \frac{1}{p(y_i|x_i)} \times \nabla_{w_{11}} p(y_i|x_i)
\]

\[
p(y|x) = \sigma(w_{21}^T h)
\]

\[
\nabla_{w_{11}} \mathbb{L} = \sum_{i=1}^{n} (I(y_i = 1) - \sigma(w_{21}^T h)) \nabla_{w_{11}} w_{21}^T h
\]

\[
\nabla_{w_{11}} \mathbb{L} = \sum_{i=1}^{n} (I(y_i = 1) - \sigma(w_{21}^T h)) w_{21}^T (\nabla_{w_{11}} h)
\]

\[
\nabla_{w_{11}} \mathbb{L} = \sum_{i=1}^{n} (I(y_i = 1) - \sigma(w_{21}^T h)) w_{21}[1] \nabla_{w_{11}} \sigma(w_{11}^T x_i)
\]

\[
\mathbb{L}(W) = \sum_{i=1}^{n} \log p(y_i|x_i)
\]
Gradients

\[ p(y|x) = \sigma \left( w_{21}^T \left[ \sigma(w_{11}^T x), \sigma(w_{12}^T x) \right]^T \right) \]

\[ \nabla_{w_{11}} \ell = \sum_{i=1}^{n} \frac{1}{p(y_i|x_i)} \times \nabla_{w_{11}} p(y_i|x_i) \]

\[ p(y|x) = \sigma(w_{21}^T h) \]

\[ \ell(W) = \sum_{i=1}^{n} \log p(y_i|x_i) \]

\[ \nabla_{w_{11}} \ell = \sum_{i=1}^{n} (l(y_i = 1) - \sigma(w_{21}^T h)) \nabla_{w_{11}} w_{21}^T h \]

\[ \nabla_{w_{11}} \ell = \sum_{i=1}^{n} (l(y_i = 1) - \sigma(w_{21}^T h)) w_{21}^T (\nabla_{w_{11}} h) \]

\[ \nabla_{w_{11}} \ell = \sum_{i=1}^{n} (l(y_i = 1) - \sigma(w_{21}^T h)) w_{21}[1] \nabla_{w_{11}} \sigma(w_{11}^T x_i) \]

\[ \nabla_{w_{11}} \ell = \sum_{i=1}^{n} (l(y_i = 1) - \sigma(w_{21}^T h)) w_{21}[1] \sigma(w_{11}^T x_i)(1 - \sigma(w_{11}^T x_i)) x_i \]
\[ \ell(W) = \sum_{i=1}^{n} \log p(y_i|x_i) = \sum_{i=1}^{n} \log \sigma \left( w_{21}^\top \left[ \sigma(w_{11}^\top x_i), \sigma(w_{12}^\top x_i) \right]^\top \right) \]

\[ \nabla_{w_{11}} \ell = \sum_{i=1}^{n} \left( I(y_i = 1) - \sigma(w_{21}^\top h) \right) w_{21}[1] \sigma(w_{11}^\top x_i)(1 - \sigma(w_{11}^\top x_i))x_i \]
\[ \nabla_{w_{11}} \text{ll} = \sum_{i=1}^{n} (I(y_i = 1) - \sigma(w_{21}^T h))w_{21}[1] \sigma(w_{11}^T x_i)(1 - \sigma(w_{11}^T x_i))x_i \]

- **raw error**
  - \( \log \sigma(w_{21}^T h) \)

- **blame for error**
  - \( w_{21}^T h \)

- **gradient of blamed error**
  - \( h_1 = \sigma(w_{11}^T x) \)
Matrix Form

\[ h_1 = s(W_1x) \]
Matrix Form

\[ h_1 = s(W_1 x) \]

\[ s(v) = [s(v_1), s(v_2), s(v_3), \ldots]^T \]

\[
W_1 =
\begin{bmatrix}
W_{11} & W_{12} & W_{13} & W_{14} & W_{15}
\end{bmatrix}
\]
Matrix Form

\[ x \]

\[ h_1 = s(W_1 x) \]

\[ s(v) = [s(v_1), s(v_2), s(v_3), ...]^T \]

\[
\begin{array}{cccccc}
W_{11} & W_{12} & W_{13} & W_{14} & W_{15} \\
W_{21} & W_{22} & W_{23} & W_{24} & W_{25} \\
W_{31} & W_{32} & W_{33} & W_{34} & W_{35}
\end{array}
\]

\# of input units

\# of output units
Matrix Form

\[
J(W) = \ell(f(x, W))
\]

\[
x
\]

\[
h_1 = s(W_1 x)
\]

\[
h_2 = s(W_2 h_1)
\]

\[
\ldots
\]

\[
h_{m-1} = s(W_{m-1} h_{m-2})
\]

\[
f(x, W) = s(W_m h_{m-1})
\]
Matrix Gradient Recipe

\[ h_1 = s(W_1 x) \]
\[ h_2 = s(W_2 h_1) \]
\[ \ldots \]
\[ h_{m-1} = s(W_{m-1} h_{m-2}) \]
\[ f(x, W) = s(W_m h_{m-1}) \]
\[ J(W) = \ell(f(x, W)) \]

\[ \nabla_{W_1} J = \delta_1 x^\top \]
\[ \nabla_{W_i} J = \delta_i h_{i-1}^\top \]
\[ \delta_i = (W_{i+1} \delta_{i+1}) \odot s'(W_i h_{i-1}) \]
\[ \nabla_{W_{m-1}} J = \delta_{m-1} h_{m-2}^\top \]
\[ \delta_{m-1} = (W_m \delta_m) \odot s'(W_{m-1} h_{m-2}) \]
\[ \nabla_{W_m} J = \delta_m h_{m-1}^\top \]
\[ \delta_m = \ell'(f(x, W)) \]
Matrix Gradient Recipe

\[ h_1 = s(W_1 x) \]
\[ h_i = s(W_i h_{i-1}) \]
\[ f(x, W) = s(W_m h_{m-1}) \]
\[ J(W) = \ell(f(x, W)) \]

\[ \delta_i = (W_{i+1}^\top \delta_{i+1}) \odot s'(W_i h_{i-1}) \]
\[ \nabla_{W_1} J = \delta_1 x^\top \]
\[ \delta_m = \ell'(f(x, W)) \]
\[ \nabla_{W_i} J = \delta_i h_{i-1}^\top \]

Feed Forward Propagation

Back Propagation
Challenges

• Local minima (non-convex)

• Overfitting

Remedies

• Regularization

• Parameter sharing: convolution

• Pre-training: initializing weights smartly

• Training data manipulation, e.g., dropout, noise, transformations

• Huge data sets