Backpropagation

Gokberk Cinbis
Where we are...

\[ s = f(x; W) = Wx \]  
\[ L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \]  
\[ L = \frac{1}{N} \sum_{i=1}^{N} L_i + \sum_k W_k^2 \]

want \[ \nabla_W L \]
Optimization

# Vanilla Gradient Descent

```python
while True:
    weights_grad = evaluate_gradient(loss_fun, data, weights)
    weights += - step_size * weights_grad # perform parameter update
```

(image credits to Alec Radford)

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Gradient Descent

\[ \frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x + h) - f(x)}{h} \]

**Numerical gradient:** slow :(, approximate :(, easy to write :)

**Analytic gradient:** fast :), exact :), error-prone :( 

In practice: Derive analytic gradient, check your implementation with numerical gradient
Computational Graph

\[ f = Wx \]

\[ L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \]
Convolutional Network (AlexNet)

input image
weights
loss

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Neural Turing Machine

input tape

loss

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
\[ f(x, y, z) = (x + y)z \]

e.g. \( x = -2, y = 5, z = -4 \)
\[ f(x, y, z) = (x + y)z \]

e.g. \( x = -2, \ y = 5, \ z = -4 \)

\[
q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1
\]

\[
f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q
\]

Want: \( \frac{\partial f}{\partial x}, \ \frac{\partial f}{\partial y}, \ \frac{\partial f}{\partial z} \)

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
\[ f(x, y, z) = (x + y)z \]

*Example:* \( x = -2, y = 5, z = -4 \)

\[ q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1 \]

\[ f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q \]

Want:

\[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \]
\[ f(x, y, z) = (x + y)z \]

e.g. \( x = -2, y = 5, z = -4 \)

\[
q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1
\]

\[
f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q
\]

Want: \[ \frac{\partial f}{\partial x}, \quad \frac{\partial f}{\partial y}, \quad \frac{\partial f}{\partial z} \]
$$f(x, y, z) = (x + y)z$$

e.g. $x = -2$, $y = 5$, $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$
\[ f(x, y, z) = (x + y)z \]
e.g. \( x = -2, y = 5, z = -4 \)

\[
q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1
\]

\[
f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q
\]

Want: \( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \)

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
\( f(x, y, z) = (x + y)z \)

e.g. \( x = -2, y = 5, z = -4 \)

\[
q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1
\]

\[
f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q
\]

Want: \( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \)

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
\[ f(x, y, z) = (x + y)z \]

E.g. \( x = -2, y = 5, z = -4 \)

\[ q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1 \]

\[ f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q \]

Want: \[ \frac{\partial f}{\partial x}, \quad \frac{\partial f}{\partial y}, \quad \frac{\partial f}{\partial z} \]

Chain rule:

\[
\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}
\]

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
\[ f(x, y, z) = (x + y)z \]

\text{e.g. } x = -2, \ y = 5, \ z = -4

\[
q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1
\]

\[
f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q
\]

Want: \[ \frac{\partial f}{\partial x}, \quad \frac{\partial f}{\partial y}, \quad \frac{\partial f}{\partial z} \]
\[ f(x, y, z) = (x + y)z \]

E.g. \( x = -2, \quad y = 5, \quad z = -4 \)

\[ q = x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1 \]

\[ f = qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q \]

Want: \( \frac{\partial f}{\partial x}, \quad \frac{\partial f}{\partial y}, \quad \frac{\partial f}{\partial z} \)

Chain rule:

\[ \frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \cdot \frac{\partial q}{\partial x} \]

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
activations

\[ f \]

\[ x \]

\[ y \]

\[ z \]

---

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
activations

\[
\frac{\partial z}{\partial x}
\]

\[
\frac{\partial z}{\partial y}
\]

\[
f
\]

"local gradient"

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
activations

\[ \frac{\partial L}{\partial x} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial x} \]

“local gradient”

\[ \frac{\partial z}{\partial y} \]

\[ \frac{\partial L}{\partial z} \]

gradients

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
activations

\[
\frac{\partial L}{\partial x} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial x}
\]

\[
\frac{\partial L}{\partial y} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial y}
\]

"local gradient"

gradients

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
activations

\[ \frac{\partial L}{\partial x} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial x} \]

\[ \frac{\partial L}{\partial y} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial y} \]

“local gradient”

\[ \frac{\partial L}{\partial z} \]

gradients
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]
Another example:

\[
f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}
\]

\[
f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x
\]

\[
f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a
\]

\[
f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -\frac{1}{x^2}
\]

\[
f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1
\]
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[ f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x \]

\[ f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a \]

\[ f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -\frac{1}{x^2} \]

\[ f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1 \]
Another example: \[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[ (-1)^{1.37^2} (1.00) = -0.53 \]

\[ f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x \]

\[ f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a \]

\[ f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1 \]

\[ f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -\frac{1}{x^2} \]
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[ f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x \]

\[ f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a \]

\[ f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -\frac{1}{x^2} \]

\[ f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1 \]
Another example: 
\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[ (1)(-0.53) = -0.53 \]

\[
\begin{align*}
  f(x) &= e^x \\
  f_a(x) &= ax \\
  f_c(x) &= c + x
\end{align*}
\]

\[
\begin{align*}
  \frac{df}{dx} &= e^x \\
  \frac{df}{dx} &= a \\
  \frac{df}{dx} &= 1
\end{align*}
\]

\[
\begin{align*}
  \frac{df}{dx} &= -1/x^2
\end{align*}
\]

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[
\begin{align*}
\text{Introduction to CV} & \\
\text{Lecture 4 - 13 Jan 2016} & \\
\text{Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson} & \\
\end{align*}
\]

\[
\begin{align*}
f(x) &= e^x & \Rightarrow & & \frac{df}{dx} &= e^x \\
f_a(x) &= ax & \Rightarrow & & \frac{df}{dx} &= a \\
f_c(x) &= c + x & \Rightarrow & & \frac{df}{dx} &= 1 \\
f(x) &= \frac{1}{x} & \Rightarrow & & \frac{df}{dx} &= -\frac{1}{x^2} \\
\end{align*}
\]
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[ (e^{-1})(-0.53) = -0.20 \]

\[
\begin{align*}
  f(x) &= e^x \\
  \frac{df}{dx} &= e^x \\
  f_a(x) &= ax \\
  \frac{df}{dx} &= a \\
  f_c(x) &= c + x \\
  \frac{df}{dx} &= 1 \\
  f(x) &= \frac{1}{x} \\
  \frac{df}{dx} &= -1/x^2
\end{align*}
\]
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2)}} \]

\[ f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x \]

\[ f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a \]

\[ f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1 \]

\[ f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -\frac{1}{x^2} \]

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2)}} \]

\((-1) \times (-0.20) = 0.20\)

\[
\begin{align*}
 f(x) &= e^x \\
 f_a(x) &= ax \\
 f_c(x) &= c + x \\
 f(x) &= \frac{1}{x} \\
 f(x) &= \frac{1}{x^2} \\
 f(x) &= 1
\end{align*}
\]

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[
\begin{align*}
  f(x) = e^x & \quad \rightarrow \quad \frac{df}{dx} = e^x \\
  f_a(x) = ax & \quad \rightarrow \quad \frac{df}{dx} = a \\
  f_c(x) = c + x & \quad \rightarrow \quad \frac{df}{dx} = 1
\end{align*}
\]

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[
\begin{align*}
\text{[local gradient]} \times \text{[its gradient]} & \\
\[1\] \times [0.2] = 0.2 & \\
[1] \times [0.2] = 0.2 \quad \text{(both inputs!)}
\end{align*}
\]

\[
\begin{align*}
\text{f}(x) &= e^x & \Rightarrow & \quad \frac{df}{dx} &= e^x \\
\text{f}_a(x) &= ax & \Rightarrow & \quad \frac{df}{dx} &= a \\
\text{f}_c(x) &= c + x & \Rightarrow & \quad \frac{df}{dx} &= 1
\end{align*}
\]
Another example: \[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

\[
\begin{align*}
f(x) &= e^x \quad \rightarrow \quad & \frac{df}{dx} &= e^x \\
 f_a(x) &= ax \quad \rightarrow \quad & \frac{df}{dx} &= a \\
f_c(x) &= c + x \quad \rightarrow \quad & \frac{df}{dx} &= 1
\end{align*}
\]
Another example:

\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

[local gradient] \( x \) [its gradient]
\[ x_0: [2] \times [0.2] = 0.4 \]
\[ w_0: [-1] \times [0.2] = -0.2 \]

\[
\begin{align*}
f(x) &= e^x \\
f_a(x) &= ax
\end{align*}
\]
\[
\begin{align*}
\frac{df}{dx} &= e^x \\
\frac{df}{dx} &= a
\end{align*}
\]
\[
\begin{align*}
f(x) &= \frac{1}{x} \\
f_c(x) &= c + x
\end{align*}
\]
\[
\begin{align*}
\frac{df}{dx} &= -\frac{1}{x^2} \\
\frac{df}{dx} &= 1
\end{align*}
\]
\[ f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}} \]

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

\[ \frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2} = \left( \frac{1 + e^{-x} - 1}{1 + e^{-x}} \right) \left( \frac{1}{1 + e^{-x}} \right) = (1 - \sigma(x)) \sigma(x) \]
The function $f(w, x) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2)}}$ is illustrated in the diagram. The derivative of the sigmoid function is:

$$\frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2} = \left(\frac{1 + e^{-x} - 1}{1 + e^{-x}}\right) \left(\frac{1}{1 + e^{-x}}\right) = (1 - \sigma(x))\sigma(x)$$

A specific example in the diagram is:

$$(0.73) \times (1 - 0.73) = 0.2$$
Patterns in backward flow

**add** gate: gradient distributor

**max** gate: gradient router

**mul** gate: gradient… “switcher”?

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Gradients add at branches

See multivariable chain rule.
**Implementation:** forward/backward API

Graph (or Net) object. *(Rough psuedo code)*

```python
class ComputationalGraph(object):
    #...
    def forward(inputs):
        # 1. [pass inputs to input gates...]
        # 2. forward the computational graph:
        for gate in self.graph.nodes_topologically_sorted():
            gate.forward()
        return loss # the final gate in the graph outputs the loss
    def backward():
        for gate in reversed(self.graph.nodes_topologically_sorted()):
            gate.backward() # little piece of backprop (chain rule applied)
        return inputs_gradients
```
Implementation: forward/backward API

\[ x \rightarrow * \rightarrow z \]
\[ y \rightarrow \]

\[(x, y, z \text{ are scalars})\]

```python
class MultiplyGate(object):
    def forward(self, x, y):
        z = x * y
        return z

    def backward(self, dz):
        # dx = ... #todo
        # dy = ... #todo
        return [dz, dz]
```

\[ \frac{\partial L}{\partial z} \]
\[ \frac{\partial L}{\partial x} \]
Implementation: forward/backward API

```python
class MultiplyGate(object):
    def forward(x, y):
        z = x*y
        self.x = x # must keep these around!
        self.y = y
        return z
    def backward(dz):
        dx = self.y * dz # [dz/dx * dL/dz]
        dy = self.x * dz # [dz/dy * dL/dz]
        return [dx, dy]
```

(x, y, z are scalars)
Example: Torch Layers

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Example: Torch Layers
Example: Torch MulConstant

\[ f(X) = aX \]

- **Initialization**
- **Forward()**
- **Backward()**

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Example: Caffe Layers

<table>
<thead>
<tr>
<th>Layer Name</th>
<th>Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convolution Layer</td>
<td>Filtered image</td>
<td>Convolution of input image with filter weights</td>
</tr>
<tr>
<td>Pooling Layer</td>
<td>Pooling</td>
<td>Reduces spatial dimensions of input</td>
</tr>
<tr>
<td>Activation Layer</td>
<td>Activation</td>
<td>Non-linear transformation of input</td>
</tr>
<tr>
<td>Dropout Layer</td>
<td>Dropout</td>
<td>Randomly drops units with probability p</td>
</tr>
</tbody>
</table>

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Caffe Sigmoid Layer (CPU)

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

\[ (1 - \sigma(x)) \sigma(x) \]

*top_diff  (chain rule)
Gradients for vectorized code

This is now the **Jacobian matrix** (derivative of each element of \( z \) w.r.t. each element of \( x \))

\[
\frac{\partial L}{\partial x} = \frac{\partial z}{\partial x} \frac{\partial L}{\partial z}
\]

\[
\frac{\partial L}{\partial y} = \frac{\partial z}{\partial y}
\]

\[
\frac{\partial L}{\partial z}
\]

(x,y,z are now vectors)

"local gradient"

Slide by Fei-Fei Li, Andrej Karpathy & Justin Johnson
Vectorized operations

\[ f(x) = \max(0, x) \] (elementwise)

4096-d input vector \[ \rightarrow \] 4096-d output vector
Vectorized operations

$f(x) = \max(0, x) \quad (\text{elementwise})$

Q: what is the size of the Jacobian matrix?

4096-d input vector

Jacobian matrix

4096-d output vector
Vectorized operations

\[
\frac{\partial L}{\partial x} = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial L}{\partial f} \end{bmatrix}
\]

Jacobian matrix

Q: what is the size of the Jacobian matrix? [4096 x 4096!]

Q2: what does it look like?

\[ f(x) = \max(0,x) \] (elementwise)

4096-d input vector

4096-d output vector
Vectorized operations

Q: what is the size of the Jacobian matrix? [4096 x 4096!]

Q2: what does it look like? [A diagonal matrix of 1s and 0s]
Vectorized operations

in practice we process an entire minibatch (e.g. 100) of examples at one time:

100 4096-d input vectors

\[ f(x) = \max(0, x) \] (elementwise)

100 4096-d output vectors

i.e. Jacobian would technically be a \([409600 \times 409600]\) matrix :\}
Summary so far

- neural nets will be very large: no hope of writing down gradient formula by hand for all parameters
- **backpropagation** = recursive application of the chain rule along a computational graph to compute the gradients of all inputs/parameters/intermediates
- implementations maintain a graph structure, where the nodes implement the `forward()` / `backward()` API.
- **forward**: compute result of an operation and save any intermediates needed for gradient computation in memory
- **backward**: apply the chain rule to compute the gradient of the loss function with respect to the inputs.