Backpropagation

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

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

“local gradient”

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

gradients

Slides from Fei-Fei Li, Justin Johnson, Serena Yeung
http://vision.stanford.edu/teaching/cs231n/
Readings

• Stochastic Gradient Descent & Backpropagation
  – http://cs231n.github.io/optimization-1/
  – http://cs231n.github.io/optimization-2/
Announcements

• Project 4 (Stereo) due tomorrow, April 26, 2018, by 11:59pm

• Quiz 3 in class, Monday, 4/30, first 10 minutes of class

• Final exam in class, May 9
  – Will provide some study materials
Today

• How to train CNNs
  – Backpropagation algorithm
  – Best practices for training deep CNNs
  – Data augmentation
Last time: neural networks

• Computation graph for a 2-layer neural network

[Diagram of a 2-layer neural network with input layer, hidden layer, and output layer]
Last time: convolutional neural networks
Last time: convolutional layers

For example, if we had 6 5x5 filters, we’ll get 6 separate activation maps:

We stack these up to get a “new image” of size 28x28x6!
Last time: convolutional neural networks

preview:
A closer look at spatial dimensions:

7x7 input (spatially) assume 3x3 filter applied with stride 3?

doesn’t fit! cannot apply 3x3 filter on 7x7 input with stride 3.
Output size:
\[(N - F) / \text{stride} + 1\]

e.g. \(N = 7, F = 3\):
- stride 1 => \((7 - 3)/1 + 1 = 5\)
- stride 2 => \((7 - 3)/2 + 1 = 3\)
- stride 3 => \((7 - 3)/3 + 1 = 2.33\)
In practice: Common to zero pad the border

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

e.g. input 7x7
3x3 filter, applied with **stride 1**
**pad with 1 pixel** border => what is the output?

(recall:)

\[(N - F) / \text{stride} + 1\]
In practice: Common to zero pad the border

E.g. input 7x7
3x3 filter, applied with **stride 1**
Pad with 1 pixel border => what is the output?

7x7 output!
In practice: Common to zero pad the border

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

e.g. input 7x7
3x3 filter, applied with **stride 1**
**pad with 1 pixel** border => what is the output?

**7x7 output!**
in general, common to see CONV layers with
stride 1, filters of size FxF, and zero-padding with
(F-1)/2. (will preserve size spatially)
e.g. F = 3 => zero pad with 1
     F = 5 => zero pad with 2
     F = 7 => zero pad with 3
Remember back to…
E.g. 32x32 input convolved repeatedly with 5x5 filters shrinks volumes spatially! (32 -> 28 -> 24 ...). Shrinking too fast is not good, doesn’t work well.
Examples time:

Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Output volume size: ?
Examples time:

Input volume: 32x32x3
10 5x5 filters with stride 1, pad 2

Output volume size:
\[(32+2\times2-5)/1+1 = 32\] spatially, so 32x32x10
Examples time:

Input volume: $32 \times 32 \times 3$
10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?
Examples time:

Input volume: \textbf{32x32x3}
10 5x5 filters with stride 1, pad 2

Number of parameters in this layer? each filter has \(5 \times 5 \times 3 + 1 = 76\) params (+1 for bias) => \(76 \times 10 = 760\)
(btw, 1x1 convolution layers make perfect sense)

1x1 CONV with 32 filters
(each filter has size 1x1x64, and performs a 64-dimensional dot product)
Pooling layer
- makes the representations smaller and more manageable
- operates over each activation map independently:
MAX POOLING

Single depth slice

\[
\begin{array}{cccc}
1 & 1 & 2 & 4 \\
5 & 6 & 7 & 8 \\
3 & 2 & 1 & 0 \\
1 & 2 & 3 & 4 \\
\end{array}
\]

max pool with 2x2 filters and stride 2

\[
\begin{array}{cc}
6 & 8 \\
3 & 4 \\
\end{array}
\]
Fully Connected Layer (FC layer)
- Contains neurons that connect to the entire input volume, as in ordinary Neural Networks
ConvNetJS CIFAR-10 demo

This demo trains a Convolutional Neural Network on the CIFAR-10 dataset in your browser, with nothing but Javascript. The state of the art on this dataset is about 90% accuracy and human performance is at about 94% (not perfect as the dataset can be a bit ambiguous). I used this python script to parse the original files (python version) into batches of images that can be easily loaded into page DOM with img tags.

This dataset is more difficult and it takes longer to train a network. Data augmentation includes random flipping and random image shifts by up to 2px horizontally and vertically.

By default, in this demo we're using Adadelta which is one of per-parameter adaptive step size methods, so we don't have to worry about changing learning rates or momentum over time. However, I still included the text fields for changing these if you'd like to play around with SGD+Momentum trainer.

Report questions/bugs/suggestions to @karpathy.

https://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html
Summary of CNNs

- ConvNets stack CONV,POOL,FC layers
- Trend towards smaller filters and deeper architectures
- Trend towards getting rid of POOL/FC layers (just CONV)
- Typical architectures look like
  \[
  [(\text{CONV-RELU})^N\text{-POOL~}]^M\text{-}(\text{FC-RELU})^K,\text{SOFTMAX}
  \]
  where $N$ is usually up to $\sim 5$, $M$ is large, $0 \leq K \leq 2$.
  - but recent advances such as ResNet/GoogLeNet challenge this paradigm
Questions?
Bigger picture

• A convolutional neural network can be thought of as a function from images to class scores
  – With millions of adjustable weights...
  – ... leading to a very non-linear mapping from images to features / class scores.
  – We will set these weights based on classification accuracy on training data...
  – ... and hopefully our network will generalize to new images at test time
Back to optimization

• Now we know what the structure of our function from images -> class scores is

• How do we learn the weights?

• Answer: Stochastic gradient descent
  – Requires that we compute the derivative of the training loss with respect to all weights
Where we are

• Function $f$ maps images to class scores

\[ s = f(x; W) = Wx \]

$f$ is a deep CNN

• Loss function maps class scores to “badness”

\[ L_i = -\log \left( \frac{e^{f_{y_i}}}{\sum_j e^{f_j}} \right) \]

Cross-entropy loss

\[ L = \frac{1}{N} \sum_{i=1}^{N} L_i + \sum_k W_k^2 \]

Data loss + regularization

want \[ \nabla_W L \]

(gradient of $L$ w.r.t. $W$, computed analytically)
Computation graphs

Forwards pass: compute loss using current weights

Backwards pass: compute gradients of loss w.r.t. weights, then update the weights (backpropagation algorithm)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

e.g. \( x = -2, y = 5, z = -4 \)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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

\[
\begin{align*}
q &= x + y \quad \frac{\partial q}{\partial x} = 1, \quad \frac{\partial q}{\partial y} = 1 \\
\end{align*}
\]

\[
\begin{align*}
f &= qz \quad \frac{\partial f}{\partial q} = z, \quad \frac{\partial f}{\partial z} = q \\
\end{align*}
\]

Want: \( \frac{\partial f}{\partial x}, \ \frac{\partial f}{\partial y}, \ \frac{\partial f}{\partial z} \)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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: \( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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: \( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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: \( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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: \( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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: \( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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: \( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \)
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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:

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

\ [%] Chain rule:

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

Upstream gradient

Local gradient
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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

Chain rule:

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

Diagram:
- Upstream gradient
- Local gradient
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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

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

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

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

Chain rule:

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

Upstream gradient

Local gradient
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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

Chain rule:

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

Upstream gradient

Local gradient
Backpropagation: a simple example

\[ f(x, y, z) = (x + y)z \]

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

Chain rule:

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

[Diagram showing the calculation process]

Upstream gradient
Local gradient
The diagram illustrates a function $f$ with local gradients.

- $x$ and $y$ are input variables.
- $z$ is the output variable.
- The function $f$ has local gradients with respect to $x$ and $y$.

The statement "local gradient" refers to the partial derivatives of $f$ with respect to $x$ and $y$:
The diagram illustrates a function $f$ with inputs $x$ and $y$, and output $z$. The "local gradient" of $f$ is denoted by $\frac{\partial z}{\partial x}$ and $\frac{\partial z}{\partial y}$. The gradient with respect to $z$ is denoted by $\frac{\partial L}{\partial z}$. The diagram also shows the concept of gradients.
\[
\frac{\partial L}{\partial x} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial x} + \frac{\partial L}{\partial y} \frac{\partial y}{\partial x} + \frac{\partial L}{\partial z}
\]
\[
\frac{\partial L}{\partial x} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial x}
\]

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

"local gradient"
Another example:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

\begin{align*}
f(x) &= e^x & \rightarrow & \frac{df}{dx} &= e^x \\
f_a(x) &= a x & \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:

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

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

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

\[
\begin{align*}
    (\text{Upstream gradient}) & = -0.53(e^{-1}) = -0.20 \\
    (\text{Local gradient}) & = 0.37 - 0.53
\end{align*}
\]
Another example:

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

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

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

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

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

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

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

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

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

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

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

[upstream gradient] x [local gradient]

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

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

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

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

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

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

[upstream gradient] x [local gradient]

\( x_0: [0.2] \times [2] = 0.4 \)
\( w_0: [0.2] \times [-1] = -0.2 \)

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

**add** gate: gradient distributor
Patterns in backward flow

**add gate**: gradient distributor

Q: What is a **max** gate?
Patterns in backward flow

**add** gate: gradient distributor

**max** gate: gradient router
Patterns in backward flow

**add** gate: gradient distributor
**max** gate: gradient router

Q: What is a **mul** gate?
Patterns in backward flow

**add** gate: gradient distributor

**max** gate: gradient router

**mul** gate: gradient switcher
Gradients add at branches
Vectorized operations

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

4096-d input vector  \rightarrow  \rightarrow  \rightarrow  \rightarrow  \rightarrow  \rightarrow

4096-d output vector
Vectorized operations

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

Jacobian matrix

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

4096-d input vector

4096-d output vector

Q: what is the size of the Jacobian matrix?
Vectorized operations

$f(x) = \max(0, x)$ (elementwise)

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

Jacobian matrix

4096-d input vector

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

4096-d output vector
Vectorized operations

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

Jacobian matrix

4096-d input vector

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

4096-d output vector

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

Q2: what does it look like?
Summary so far...

- neural nets will be very large: impractical to write 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
Questions?