Introduction to Neural Networks
Linear Models, MLPs, Backpropagation

EE367/CS448I: Computational Imaging
stanford.edu/class/ee367
Lecture 8

Axel Levy
Stanford University
courtesy of David Lindell, adapted from Stanford CS231N
Neural Networks in Computational Imaging

- Now: learned pipelines for computational imaging

Learning CFAs

Learning ISPs
Neural Networks in Computational Imaging

- Now: learned pipelines for computational imaging

- Learned denoising
- Learned deblurring
- HDR Imaging
Today

- What is a neural network?
- How do we train neural networks?
Today

- What is a neural network?
- How do we train neural networks?

Wed.

- Convolutional neural networks
- Making networks deep
- Applications in denoising and deblurring
What is a neural network?

- Image classification example
Image Classification

- Image classification example

Images

MNIST Dataset
Image Classification

- Image classification example

<table>
<thead>
<tr>
<th>Images</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0</td>
<td>“zero”</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>“one”</td>
</tr>
<tr>
<td>2 2 2 2</td>
<td>“two”</td>
</tr>
<tr>
<td>3 3 3 3</td>
<td>“three”</td>
</tr>
<tr>
<td>4 4 4 4</td>
<td>“four”</td>
</tr>
<tr>
<td>5 5 5 5</td>
<td>“five”</td>
</tr>
<tr>
<td>6 6 6 6</td>
<td>“six”</td>
</tr>
<tr>
<td>7 7 7 7</td>
<td>“seven”</td>
</tr>
<tr>
<td>8 8 8 8</td>
<td>“eight”</td>
</tr>
<tr>
<td>9 9 9 9</td>
<td>“nine”</td>
</tr>
</tbody>
</table>
Image Classification

- Image classification example

What the computer “sees”
Image Classification

- Image classification example

<table>
<thead>
<tr>
<th>Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2</td>
</tr>
<tr>
<td>3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3</td>
</tr>
<tr>
<td>4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4</td>
</tr>
<tr>
<td>5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5</td>
</tr>
<tr>
<td>6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6</td>
</tr>
<tr>
<td>7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7</td>
</tr>
<tr>
<td>8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8</td>
</tr>
<tr>
<td>9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9</td>
</tr>
</tbody>
</table>

Challenges
- Intra-class variation
- stroke widths
- alignment
- writing styles
Image Classification

- Image classification example

<table>
<thead>
<tr>
<th>Images</th>
<th>Challenges</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Intra-class variation</td>
</tr>
<tr>
<td></td>
<td>- stroke widths</td>
</tr>
<tr>
<td></td>
<td>- alignment</td>
</tr>
<tr>
<td></td>
<td>- writing styles</td>
</tr>
<tr>
<td></td>
<td>Inter-class similarities</td>
</tr>
<tr>
<td></td>
<td>- “four” or “nine”?</td>
</tr>
</tbody>
</table>

Challenges:
- Intra-class variation
  - stroke widths
  - alignment
  - writing styles
- Inter-class similarities
  - “four” or “nine”?
Image Classification

- Image classification example

Images

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<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>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
</tbody>
</table>

Implementation?

```python
def classify_digit(image):
    # ???
    return image_class
```

Can’t hardcode solution!
Data-Driven Approach

1. Collect training images and labels
   \[ \{x_i^{tr}\}, \{y_i^{tr}\} \]

2. Define a **classifier** = parametric function with discretized outputs
   \[ f(x, \theta) = \ldots \]

3. Define a **loss** = score function
   \[ \mathcal{L}(\{\hat{y}_i\}, \{y_i\}) \]

4. Train the classifier using machine learning
   \[ \min_{\theta} \mathcal{L}(\{f(x_i^{tr}, \theta)\}, \{y_i^{tr}\}) \]

5. Evaluate the classifier on unseen images
   \[ \mathcal{L}(\{f(x_i^{test}, \theta^*)\}, \{y_i^{test}\}) \]
Step 2: Defining a classifier

- Linear Model

\[ f(x, W) = Wx \]
Linear Model

\[ f(x, W) = Wx \]
Linear Model

- Linear Model

\[ f(x, W) = Wx \]
Linear Model

- Linear Model
  \[ f(x, W) = Wx \]

\[ x \in \mathbb{R}^N \]

Length (dimension) of this vector = number of pixels
Linear Model

- Linear Model

\[ f(x, W) = Wx \]

\[ x \in \mathbb{R}^N \quad W \in \mathbb{R}^{10 \times N} \]

In general: \( Wx + b \)
Linear Model

- Linear Model

\[ f(x, W) = Wx \]

**Input**: \( x \in \mathbb{R}^N \)

**Weights**: \( W \in \mathbb{R}^{10 \times N} \)

**Output**: entry with the highest score
Linear Model

- Linear model: geometric interpretation
Linear Model

• Linear model: geometric interpretation

Can be seen as 10 inner products.

\[ WX = \begin{bmatrix} w_0 \cdot x \\ \vdots \\ w_9 \cdot x \end{bmatrix} \]
Linear Model

- Linear model (visual interpretation)

Learned filters (rows of W)
Linear Model

- Limits of linear classifiers

Linear classifiers learn linear decision planes

What if dataset is not linearly separable?
Multilayer Perceptrons (MLPs)

• Linear Model: \( f = Wx \)

• 2-layer MLP: \( f = W_2 \max(0, W_1 x) \)
Multilayer Perceptrons (MLPs)

- Linear Model: \( f = Wx \)
- 2-layer MLP: \( f = W_2 \max(0, W_1 x) \)
- 3-layer MLP: \( f = W_3 \max(0, W_2 \max(0, W_1 x)) \)
Multilayer Perceptrons (MLPs)

- Linear Model: \( f = Wx \)
- 2-layer MLP: \( f = W_2 \max(0, W_1 x) \)
- 3-layer MLP: \( f = W_3 \max(0, W_2 \max(0, W_1 x)) \)

Non-linearity/activation function between linear layers
Multilayer Perceptrons (MLPs)

- Linear Model  \( f = W x \)
- 2-layer MLP  \( f = W_2 \max(0, W_1 x) \)
- 3-layer MLP  \( f = W_3 \max(0, W_2 \max(0, W_1 x)) \)

Otherwise we have:

\( f = W_3 W_2 W_1 x \)
Activation Functions

...many to choose from

- softplus
- leaky ReLU
- ELU
- ReLU
- tanh
- sigmoid

... ReLU is a good general-purpose choice: ReLU(x) = max(0, x)
Multilayer Perceptrons (MLPs)

- Linear Model \( f = Wx \)
- 2-layer MLP \( f = W_2 \max(0, W_1 x) \)

Back to our classification example...

\( x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H} \)
Multilayer Perceptrons (MLPs)

- Linear Model \[ f = Wx \]
- 2-layer MLP \[ f = W_2 \max(0, W_1x) \]

Back to our classification example…

\[ x \in \mathbb{R}^D, \quad W_1 \in \mathbb{R}^{H \times D}, \quad W_2 \in \mathbb{R}^{C \times H} \]
Multilayer Perceptrons (MLPs)

- Linear Model: \( f = Wx \)
- 2-layer MLP: \( f = W_2 \max(0, W_1 x) \)

Back to our classification example...

Now we have 100 shape templates, shared between classes
Multilayer Perceptrons (MLPs)

- Overcomes limits of linear classifiers

- Can learn non-linear decision boundaries

- Complexity scales with the number of neurons/hidden layers
• More parameters is not always better!
  • Can lead to overfitting the training data
  • Performance on test data is worse
More on classification…

- https://cs231n.github.io/linear-classify/
# Data-Driven Approach

1. Collect training images and labels

   $\{x_{i}^{tr}\}, \{y_{i}^{tr}\}$

2. Define a **classifier** = parametric function with discretized outputs

   $f(x, \theta) = \ldots$

3. Define a **loss** = score function

   $\mathcal{L}(\{\hat{y}_{i}\}, \{y_{i}\})$

4. Train the classifier using machine learning

   $\min_{\theta} \mathcal{L}(\{f(x_{i}^{tr}, \theta)\}, \{y_{i}^{tr}\})$

5. Evaluate the classifier on unseen images

   $\mathcal{L}(\{f(x_{i}^{test}, \theta^{*})\}, \{y_{i}^{test}\})$
Image Inpainting

masked input → vectorize → $W_1$ → $W_2$ → reshape → predicted output
Step 1: Collect training inputs and outputs

masked images

ground truth
Step 2: Define a classifier

masked input

vectorize

\( W_1 \)

\( W_2 \)

100

reshape

predicted output
Step 3: Defining a loss

\[ \mathcal{L}_\theta = \frac{1}{2} \| y - \hat{y} \|_2^2 \]

network parameters
\( \theta = \{W_1, W_2\} \)
Step 3: Defining a loss

\[ \mathcal{L}_\theta = \frac{1}{2} \| y - \hat{y} \|_2^2 \]

network parameters
\[ \theta = \{W_1, W_2\} \]

ground truth image

network prediction

input
Step 4: Training the model

Gradient-based optimization

\[
\nabla_\theta \mathcal{L}
\]

\[
\theta^{(k+1)} = \theta^{(k)} - \alpha \nabla_\theta \mathcal{L}(\theta^{(k)})
\]

[Li et al. '18]
Step 4: Training the model

Need to calculate the partial derivative with respect to each parameter

\[
\frac{\partial}{\partial W_1} \mathcal{L}_\theta = \frac{\partial}{\partial W_1} \frac{1}{2} \| y - \hat{y} \|_2^2
\]

\[
\frac{\partial}{\partial W_2} \mathcal{L}_\theta = \frac{\partial}{\partial W_2} \frac{1}{2} \| y - \hat{y} \|_2^2
\]
Step 4: Training the model

Generally there are 3 options

1. Numerical differentiation
2. Symbolic differentiation
3. “Automatic” differentiation
Numerical Differentiation

\[
\frac{\partial f(x)}{\partial x} \approx \lim_{h \to 0} \frac{f(x + h) - f(x)}{h}
\]

Compute this with a "small" \( h \) (e.g. \( 10^{-6} \))

Easy to implement!

Not very accurate....
Symbolic Differentiation

\[
\frac{\partial L_\theta}{\partial W_1} = \frac{\partial}{\partial W_1} \frac{1}{2} \|y - \hat{y}\|_2^2 \\
= \frac{\partial}{\partial W_1} \frac{1}{2} (y - W_2 \sigma(W_1 x))^T \cdot (y - W_2 \sigma(W_1 x))
\]

chain rule, product rule…

Accurate

Tedious (must be manually calculated for each term)
Automatic Differentiation

Think about the problem as a “computational graph”

Divide and conquer using the chain rule

Enables “backpropagation” – an efficient way to take derivatives of the loss wrt all the parameters in the graph
Automatic Differentiation

Think about the problem as a “computational graph”

Divide and conquer using the chain rule
Automatic Differentiation

Think about the problem as a “computational graph”

Divide and conquer using the chain rule

\[
\frac{\partial L}{\partial W_2} = \frac{\partial \hat{y}}{\partial W_2} \frac{\partial L}{\partial \hat{y}}
\]
Automatic Differentiation

Think about the problem as a “computational graph”

Divide and conquer using the chain rule

\[
\frac{\partial L}{\partial W_1} = \frac{\partial f}{\partial W_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial L}{\partial \hat{y}}
\]
Automatic Differentiation

Think about the problem as a “computational graph”

Divide and conquer using the chain rule

\[
\frac{\partial L}{\partial W_1} = \frac{\partial f}{\partial W_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial L}{\partial \hat{y}}
\]

We can calculate analytical expressions for each of these terms and then plug in our values
Autodiff Example

(assume scalar values for now)

\[
\frac{\partial L}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial L}{\partial \hat{y}}
\]
Autodiff Example

(assume scalar values for now)

\[ \frac{\partial L}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \hat{y}} \frac{\partial L}{\partial \hat{y}} \]

\[ \frac{\partial L}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}} \frac{1}{2} (\hat{y} - y)^2 = \hat{y} - y \]
Autodiff Example

(assume scalar values for now)

\[ \frac{\partial L}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial L}{\partial \hat{y}} \]
Autodiff Example

(assume scalar values for now)

\[
\frac{\partial L}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \hat{y}} \frac{\partial L}{\partial \hat{y}}
\]

\[
\frac{\partial \hat{y}}{\partial g} = \frac{\partial}{\partial g} w_2 \cdot g = w_2
\]
Autodiff Example

(assume scalar values for now)

\[
\frac{\partial L}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial L}{\partial \hat{y}}
\]
Autodiff Example

(assume scalar values for now)

\[
\frac{\partial L}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial L}{\partial \hat{y}}
\]

\[
\frac{\partial g}{\partial f} = \frac{\partial}{\partial f} \sigma(f) = \frac{\partial}{\partial f} \max(0, f) = \begin{cases} 
0, & f < 0 \\
1, & \text{else}
\end{cases}
\]
Autodiff Example

(assume scalar values for now)

\[
\frac{\partial L}{\partial w_1} = \begin{bmatrix} \frac{\partial f}{\partial w_1} \end{bmatrix} \begin{bmatrix} \frac{\partial g}{\partial f} & \frac{\partial \hat{y}}{\partial g} & \frac{\partial L}{\partial \hat{y}} \end{bmatrix}
\]
(assume scalar values for now)

\[
\frac{\partial \mathcal{L}}{\partial w_1} = \begin{bmatrix} \frac{\partial f}{\partial w_1} & \frac{\partial g}{\partial f} & \frac{\partial \hat{y}}{\partial g} & \frac{\partial \mathcal{L}}{\partial \hat{y}} \end{bmatrix} \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_1} \end{bmatrix}
\]

\[
\frac{\partial f}{\partial w_1} = \frac{\partial}{\partial w_1} w_1 \cdot x = x
\]
Autodiff Example

(assume scalar values for now)
Autodiff Example

Let’s plug in the values now…

\[
\begin{align*}
x & = 5 \\
\omega_1 & = 3 \\
\omega_2 & = 2 \\
y & = 22 \\
f & = x \ast \omega_1 \\
g & = f \sigma \\
\hat{y} & = \omega_2 \ast \hat{y} \\
L & = \hat{y}^2 \\
\frac{\partial f}{\partial \omega_1} & = \frac{\partial f}{\partial f} \\
\frac{\partial g}{\partial f} & = \frac{\partial g}{\partial g} \\
\frac{\partial \hat{y}}{\partial \hat{y}} & = \frac{\partial \hat{y}}{\partial \hat{y}} \\
\frac{\partial L}{\partial \hat{y}} & = \frac{\partial L}{\partial \hat{y}}
\end{align*}
\]
Autodiff Example

Let’s plug in the values now…

\[
\begin{align*}
\frac{\partial f}{\partial w_1} & \quad \frac{\partial g}{\partial f} & \quad \frac{\partial \hat{y}}{\partial g} & \quad \frac{\partial L}{\partial \hat{y}} \\
\end{align*}
\]
Autodiff Example

Let’s plug in the values now…

\[
\begin{align*}
\frac{\partial f}{\partial w_1} & \quad \frac{\partial g}{\partial f} & \quad \frac{\partial \hat{y}}{\partial g} & \quad \frac{\partial L}{\partial \hat{y}}
\end{align*}
\]
Autodiff Example

Let’s plug in the values now…
Let's plug in the values now…
Autodiff Example

Let’s plug in the values now…

\[
\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y
\]
Autodiff Example

Let’s plug in the values now…

\[
\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial L}{\partial \hat{y}} = \hat{y} - y
\]
Let’s plug in the values now…

\[
\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial L}{\partial \hat{y}} = \hat{y} - y
\]
Autodiff Example

Let’s plug in the values now…

\[
\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial L}{\partial \hat{y}} = \hat{y} - y
\]
Autodiff Example

Let’s plug in the values now…

\[
\begin{align*}
\frac{\partial f}{\partial w_1} &= 5 \\
\frac{\partial g}{\partial f} &= 1 \\
\frac{\partial \hat{y}}{\partial g} &= 2 \\
\frac{\partial L}{\partial \hat{y}} &= 8 \\
\end{align*}
\]

\[= \hat{y} - y\]
Let’s plug in the values now…

\[
\begin{align*}
\frac{\partial f}{\partial w_1} &= 5 \\
\frac{\partial g}{\partial f} &= 1 \\
\frac{\partial \hat{y}}{\partial g} &= 2 \\
\frac{\partial L}{\partial \hat{y}} &= 8
\end{align*}
\]

\[
\frac{\partial L}{\partial w_1} = 80
\]
Save these intermediate values during forward computation
Then we perform a “backward pass”
What about $\frac{\partial L}{\partial w_2}$?
Autodiff Example

What about $\frac{\partial L}{\partial w_2}$?

We can re-use computation!

$$\frac{\partial f}{\partial w_1} = 5$$
$$\frac{\partial f}{\partial w_2} = 1$$
$$\frac{\partial g}{\partial w_2} = 2$$

$$\frac{\partial L}{\partial \hat{y}} = \hat{y} - y$$

$$\frac{\partial L}{\partial w_2} = \frac{\partial \hat{y}}{\partial w_2} \frac{\partial L}{\partial \hat{y}}$$
Autodiff Example

PyTorch Code:

```python
class Multiply(torch.autograd.Function):
    @staticmethod
    def forward(ctx, x, y):
        ctx.save_for_backward(x, y)
        z = x * y
        return z
    @staticmethod
    def backward(ctx, grad_z):
        x, y = ctx.saved_tensors
        grad_x = y * grad_z  # dz/dx * dL/dz
        grad_y = x * grad_z  # dz/dy * dL/dz
        return grad_x, grad_y
```

- Need to stash some values for use in backward
- Upstream gradient
- Multiply upstream and local gradients
Image Inpainting Training Loop

1. Sample batch of images from dataset

2. Run forward pass to calculate network output for each image

3. Run backward pass to calculate gradients with backpropagation

4. Update parameters with stochastic gradient descent
4. Update parameters with stochastic gradient descent

\[ \nabla \theta L = \left( \frac{\partial L}{\partial W_1}, \frac{\partial L}{\partial W_2} \right) \]

\[ W_2^{(k+1)} = W_2^{(k)} - \alpha \frac{\partial L}{\partial W_2} \]

\[ W_1^{(k+1)} = W_1^{(k)} - \alpha \frac{\partial L}{\partial W_1} \]

“stochastic” refers to the fact that inputs are processed in batches.
Vector Differentiation

But wait, aren’t these vectors?
Recap: vector differentiation

Scalar wrt Scalar

\[ x \in \mathbb{R} \quad y \in \mathbb{R} \]

\[ \frac{\partial y}{\partial x} \in \mathbb{R} \]
Vector Differentiation

Recap: vector differentiation

Scalar wrt Scalar
\[ x \in \mathbb{R}, y \in \mathbb{R} \]
\[ \frac{\partial y}{\partial x} \in \mathbb{R} \]

Vector wrt Vector
\[ x \in \mathbb{R}^N, y \in \mathbb{R}^M \]
\[ \frac{\partial y}{\partial x} \in \mathbb{R}^{N \times M} \]

“input x output”
Vector Differentiation

Recap: vector differentiation

\[
\frac{\partial y}{\partial x} = \begin{bmatrix}
\frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\
\vdots & \ddots & \vdots \\
\frac{\partial y_1}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n}
\end{bmatrix}
\]

Vector wrt Vector

\[
x \in \mathbb{R}^N \quad y \in \mathbb{R}^M
\]

\[
\frac{\partial y}{\partial x} \in \mathbb{R}^{N \times M}
\]

“input x output”
Recap: vector differentiation

Example 1: matrix multiply

\[ \frac{\partial \hat{y}}{\partial g} = \frac{\partial}{\partial g} W_2 g \]

\( g \in \mathbb{R}^N \)

\( \hat{y} \in \mathbb{R}^M \)

\( W_2 \in \mathbb{R}^{M \times N} \)
Recap: vector differentiation

Example 1: matrix multiply

\[
\frac{\partial \hat{y}}{\partial g} = \frac{\partial}{\partial g} W_2 g
\]

\(g \in \mathbb{R}^N\)

\(\hat{y} \in \mathbb{R}^M\)

\(W_2 \in \mathbb{R}^{M \times N}\)
Recap: vector differentiation

Example 2: elementwise functions

\[ h = f \odot g \]

\[ f \in \mathbb{R}^N \]
\[ h \in \mathbb{R}^N \]
\[ \frac{\partial h}{\partial f} \in \mathbb{R}^{N \times N} \]
Recap: vector differentiation

Example 2: elementwise functions

\[ h = f \odot g \]

\[ f \in \mathbb{R}^N \]

\[ h \in \mathbb{R}^N \]

\[ \frac{\partial h}{\partial f} \in \mathbb{R}^{N \times N} \]

\[
\begin{bmatrix}
\frac{\partial h_1}{\partial f_1} & \cdots & \frac{\partial h_n}{\partial f_1} \\
\vdots & \ddots & \vdots \\
\frac{\partial h_1}{\partial f_n} & \cdots & \frac{\partial h_n}{\partial f_n}
\end{bmatrix}
\]

\[
\frac{\partial h}{\partial f} = \begin{bmatrix} g_1 & 0 \\ \vdots & \ddots \\ 0 & \cdots & g_n \end{bmatrix} = \text{diag}(g)
Recap: vector differentiation

Final hint: dimensions should always match up!

\[
\frac{\partial \mathcal{L}}{\partial W_1} = \frac{\partial f}{\partial W_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial \mathcal{L}}{\partial \hat{y}}
\]
Summary

Linear models and MLPs

Gradient descent

Automatic differentiation, backpropagation

Computational graphs
Next Time

Convolutional neural networks

Building blocks of deep networks

Image processing with deep networks
References and Further Reading

slides adapted from Stanford CS231N: http://cs231n.stanford.edu/slides/

CS229/CS231n notes on linear classifiers
https://cs231n.github.io/linear-classify/

CS231n Notes on backprop
http://cs231n.stanford.edu/handouts/linear-backprop.pdf
https://cs231n.github.io/optimization-2/

Intro to pytorch autograd
https://pytorch.org/tutorials/beginner/blitz/autograd_tutorial.html

Extending pytorch autograd functions
Extra backpropagation example

\[ f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}} \]
Extra backpropagation example

\[ f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}} \]
Extra backpropagation example

\[ f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}} \]
Extra backpropagation example

\[
f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}
\]
Extra backpropagation example

\[ f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}} \]

\[ f(x) = \frac{1}{x} \quad \Rightarrow \quad \frac{\partial f}{\partial x} = -\frac{1}{x^2} \]

\[ \frac{\partial J}{\partial x} = \frac{\partial J}{\partial f} \cdot \frac{\partial f}{\partial x} = -\frac{1}{x^2} \]
Extra backpropagation example

\[ f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}} \]

\[ f(x) = x + 1 \quad \Rightarrow \quad \frac{\partial f}{\partial x} = 1 \]
Extra backpropagation example

\[ f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}} \]

\[ f(x) = e^x \Rightarrow \frac{\partial f}{\partial x} = e^x \]

\[ \frac{\partial J}{\partial x} = \frac{\partial J}{\partial f} \frac{\partial f}{\partial x} = \frac{\partial J}{\partial f} \cdot e^x \]
Extra backpropagation example

\[ f = \frac{1}{1 + e^{-\left(w_0 + w_1 x_1 + w_2 x_2\right)}} \]

\[ f(x, w) = xw \quad \Rightarrow \quad \frac{\partial f}{\partial x} = w, \quad \frac{\partial f}{\partial w} = x \]
Extra backpropagation example

\[ f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}} \]