Topics:
- Convolutional Neural Networks
  - What is a convolution?
  - FC vs Conv Layers
Administrativia

• HW1 Reminder
  – Due: 09/26, 11:55pm

• Project Teams Google Doc
  – https://docs.google.com/spreadsheets/d/1ouD6ctaemV_3nb2MQHs7rUOAaW9DFLu8l5Zd3yOFs7E/edit?usp=sharing
  – Project Title
  – 1-3 sentence project summary TL;DR
  – Team member names
Plan for Today

• Convolutional Neural Networks
  – What is a convolution?
  – FC vs Conv Layers
Recall: Linear Classifier

$\mathbf{f}(\mathbf{x}, \mathbf{W}) = \mathbf{W} \mathbf{x} + \mathbf{b}$

Array of $32 \times 32 \times 3$ numbers (3072 numbers total)

$\mathbf{W}$ parameters or weights

$10$ numbers giving class scores

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Example with an image with 4 pixels, and 3 classes (cat/dog/ship)

Input image

Stretch pixels into column

\[
\begin{pmatrix}
0.2 & -0.5 & 0.1 & 2.0 \\
1.5 & 1.3 & 2.1 & 0.0 \\
0 & 0.25 & 0.2 & -0.3 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
56 \\
231 \\
24 \\
2 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
1.1 \\
3.2 \\
-1.2 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
-96.8 \\
437.9 \\
61.95 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
W \\
b \\
\end{pmatrix}
\]

Cat score

Dog score

Ship score

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Recall: (Fully-Connected) Neural networks

(Before) Linear score function:

$$f = Wx$$

(Now) 2-layer Neural Network

$$f = W_2 \max(0, W_1 x)$$
Convolutional Neural Networks
(without the brain stuff)
- Spatial correlation is local
- Waste of resources + we have not enough training samples anyway.

Example: 200x200 image
40K hidden units

~2B parameters!!!
Locally Connected Layer

Example: 200x200 image
40K hidden units
“Filter” size: 10x10
4M parameters

Note:
This parameterization is good when input image is registered (e.g., face recognition).

Assumption #1

Slide Credit: Marc'Aurelio Ranzato
Locally Connected Layer

**STATIONARITY?**
Statistics similar at all locations

\[ W_i \neq W_j \]
Share the same parameters across different locations (assuming input is stationary):
Convolutions with learned kernels
Convolutions!

math $\rightarrow$ CS $\rightarrow$ programming
Convolutions for mathematicians

\[ x(t) \ast w(t) = y(t) \]

\[ w(t) = e^{-\frac{(t-t_0)^2}{2\sigma^2}} \]

\[ y(t) = (x \ast w)(t) = \int_{-\infty}^{\infty} x(t-a)w(a) \, da \]

\[ = (w \ast x)(t) = \int_{-\infty}^{\infty} x(a)w(t-a) \, da \]
Convolutions for mathematicians

\[ y(t) = \int_{-\infty}^{\infty} x(a) w(t-a) \, da \]

- Flip
- About y-axis
- Translate by \( t \)
- \( w(-a) \)
- \( w(t-a) \neq w(- (a-t)) \)

(C) Dhruv Batra
"Convolution of box signal with itself2" by Convolution_of_box_signal_with_itself.gif: Brian Ambergderivative work: Tinos (talk)
Convolutions for mathematicians

- One dimension

\[ y(t) = \int_{-\infty}^{\infty} x(t-a)w(a) \, da \]

- Two dimensions

\[ y(t_1, t_2) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x(t_1-a, t_2-b) \, da \, db \]
Convolutions for computer scientists

\[ y[t_1, t_2] = \sum_{a=-\infty}^{\infty} \sum_{b=-\infty}^{\infty} x[t_1-a, t_2-b] w[a, b] \]
Convolutions for programmers

\[ y[a, c] = \sum_{a=0}^{k_1-1} \sum_{b=0}^{k_2-1} x[a+a, c+b] w[a, b] \]
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer
Convolutional Layer

Slide Credit: Marc'Aurelio Ranzato
Convolutional Layer
Convolutional Layer
Convolution Explained

• http://setosa.io/ev/image-kernels/

• https://github.com/bruckner/deepViz
Convolutional Layer
Learn multiple filters.

E.g.: 200x200 image
100 Filters
Filter size: 10x10
10K parameters
FC vs Conv Layer

$h_i^l = \sum_{j=1}^{c_{l-1}} h_j^l \cdot W_{ij} + \text{bias}(i)$

$H_i^l = \sum_{d=1}^{c_{l-1}} H_i^d \star W_{ij} + \text{bias}(i)$

$H_i^l[k, c] = \sum_{a=0}^{k_i-1} \sum_{b=0}^{k_2-1} H_i^d[k+a, c+b] \cdot W_{ij}[a, b]$

$(k_1 \times k_2 \times c_{l-1}) \times c_l = (k_1 \times k_2 \times c_{l-1}) \times c_l$
Convolution Layer

32x32x3 image -> preserve spatial structure

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Convolution Layer

32x32x3 image

5x5x3 filter

Convolve the filter with the image i.e. “slide over the image spatially, computing dot products”

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Convolution Layer

A Convolution Layer takes a 32x32x3 image and a 5x5x3 filter as inputs. The filter is convolved with the image, i.e., "slide over the image spatially, computing dot products". Filters always extend the full depth of the input volume.
Convolution Layer

32x32x3 image
5x5x3 filter $w$

1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image (i.e. $5\times5\times3 = 75$-dimensional dot product + bias)

$$w^T x + b$$

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Convolution Layer

32x32x3 image
5x5x3 filter
convolve (slide) over all spatial locations
activation map

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Convolution Layer

Consider a second, green filter

32x32x3 image
5x5x3 filter

Convolve (slide) over all spatial locations

Activation maps

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
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!
Im2Col

Figure Credit: https://petewarden.com/2015/04/20/why-gemm-is-at-the-heart-of-deep-learning/
GEMM

Input Matrix

Kernel Matrix

(C) Dhruv Batra   Figure Credit: https://petewarden.com/2015/04/20/why-gemm-is-at-the-heart-of-deep-learning/
Time Distribution of AlexNet

GPU Forward Time Distribution

CPU Forward Time Distribution

Figure Credit: Yangqing Jia, PhD Thesis
**Preview:** ConvNet is a sequence of Convolution Layers, interspersed with activation functions.

![Diagram of ConvNet layers with dimensions and filters](image-url)
**Preview:** ConvNet is a sequence of Convolutional Layers, interspersed with activation functions.

- **First ConvLayer:**
  - Input: 3x32x32
  - Output: 6x28x28
  - Convolution (CONV) and ReLU activation function
  - Filters: 6, 5x5x3

- **Second ConvLayer:**
  - Input: 28x28x6
  - Output: 10x24x24
  - Convolution (CONV) and ReLU activation function
  - Filters: 10, 5x5x6

- **Planned ConvLayers:**
  - More convolutional layers follow the same pattern with decreasing dimensions and increasing filters.

*Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n*