Recall …

[ Zeiler and Fergus, 2013 ]
Recall ...
Motivation …

CNNs are big black boxes, let's get some intuition for how and why they work.
**First Layer Filters ...**

Directly **visualize filters** (only works for the first layer)

- **AlexNet:**
  - $64 \times 3 \times 11 \times 11$

- **ResNet-18:**
  - $64 \times 3 \times 7 \times 7$

- **ResNet-101:**
  - $64 \times 3 \times 7 \times 7$

- **DenseNet-121:**
  - $64 \times 3 \times 7 \times 7$

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, *cs231n Stanford*
Directly **visualize filters** (only works for the first layer)

AlexNet: 64 x 3 x 11 x 11

ResNet-18: 64 x 3 x 7 x 7

ResNet-101: 64 x 3 x 7 x 7

DenseNet-121: 64 x 3 x 7 x 7

... surprisingly similar across variety of networks

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
First Layer Filters …

Directly **visualize filters** (only works for the first layer)

- AlexNet: $64 \times 3 \times 11 \times 11$
- ResNet-18: $64 \times 3 \times 7 \times 7$
- ResNet-101: $64 \times 3 \times 7 \times 7$
- DenseNet-121: $64 \times 3 \times 7 \times 7$

... surprisingly similar across variety of networks

... and nearly any dataset

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Last Layer

Test image  L2 Nearest neighbors in feature space

Recall: Nearest neighbors in pixel space

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
... you are doing this for **Assignment 2**
Visualizing **Activations**

conv5 feature map of AlexNet is 128x13x13; visualize as 128 13x13 grayscale images

[ Yosinski et al., 2014 ]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Maximally **Activating Patches**

- Pick a layer and a channel; e.g., cons5 of AlexNet is 128x13x13
- Run many images through the network
- Visualize image patches that correspond to maximal activation of the neuron

[ Springenberg et al., 2015 ]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**
Intermediate Features through \textbf{(Guided) BackProp}

- Pick a single intermediate neuron somewhere in the network, e.g., neuron in 128x13x13 conv5 feature map
- Compute \textit{gradient of neuron value with respect to image pixels}

\text{[Springenberg et al., 2015]}

\text{[Zeiler and Fergus, 2014]}

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, \textit{cs231n Stanford}
Intermediate Features through (Guided) BackProp

[ Springenberg et al., 2015 ]

[ Zeiler and Fergus, 2014 ]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Gradient **Ascent**

(Guided) **BackProp**: find the part of an image that a neuron responds to

**Gradient ascent**: generate a synthetic image that maximally activates a neuron

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Gradient **Ascent**

(Guided) **BackProp**: find the part of an image that a neuron responds to

**Gradient ascent**: generate a synthetic image that maximally activates a neuron

\[
I^* = \arg \max_I f(I) + R(I)
\]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Gradient Ascent

(Guided) **BackProp**: find the part of an image that a neuron responds to

**Gradient ascent**: generate a synthetic image that maximally activates a neuron

\[
I^* = \arg \max_I f(I) + R(I)
\]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
(Guided) **BackProp**: find the part of an image that a neuron responds to

**Gradient ascent**: generate a synthetic image that maximally activates a neuron

\[ I^* = \arg \max_I f(I) + R(I) \]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Gradient Ascent

1. Initialize image with all zeros (can also start with an existing image)
2. Forward image to compute the current scores
3. BackProp to get gradient of the neuron with respect to image pixels
4. Make a small update to an image

\[ I^* = \arg \max_I f(I) + R(I) \]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Gradient Ascent

1. Initialize image with all zeros (can also start with an existing image)
2. Forward image to compute the current scores
3. BackProp to get gradient of the neuron with respect to image pixels
4. Make a small update to an image

\[ \mathbf{I}^* = \arg \max_{\mathbf{I}} f(\mathbf{I}) + R(\mathbf{I}) \]

Natural Image Regularizer \( R(\mathbf{I}) = -\lambda ||\mathbf{I}||^2 \)

Score for class C before softmax

[ Simonyan et al., 2014 ]
Gradient Ascent

\[ I^* = \arg \max_I f(I) + R(I) \]

Natural Image Regularizer \( R(I) = -\lambda ||I||^2 \)

Score for class C before softmax

[ Simonyan et al., 2014 ]

*slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Gradient **Ascent**

... with a few additional tweaks

<table>
<thead>
<tr>
<th>bell pepper</th>
<th>cardoon</th>
<th>strawberry</th>
<th>orange</th>
<th>pineapple</th>
<th>hay</th>
<th>alp</th>
<th>bubble</th>
<th>cliff</th>
</tr>
</thead>
<tbody>
<tr>
<td>beer bottle</td>
<td>birdhouse</td>
<td>breakwater</td>
<td>breastplate</td>
<td>broom</td>
<td>caldron</td>
<td>candle</td>
<td>cinema</td>
<td>cowboy boot</td>
</tr>
<tr>
<td>entertainment</td>
<td>gasmask</td>
<td>golf ball</td>
<td>golfcart</td>
<td>gown</td>
<td>grand piano</td>
<td>hourglass</td>
<td>jack-o’-lantern</td>
<td>knot</td>
</tr>
<tr>
<td>lampshade</td>
<td>monitor</td>
<td>mosque</td>
<td>motor scooter</td>
<td>pirate</td>
<td>planetarium</td>
<td>radio</td>
<td>sarong</td>
<td>schooner</td>
</tr>
</tbody>
</table>

[Nguyen et al., 2015]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Deep Dream

https://www.youtube.com/watch?v=DgPaCWJL7XI&t=11s
Deep Dream

[Mordvinsev, Olah, Tyka]

https://www.youtube.com/watch?v=DgPaCWJL7XI&t=11s
Fooling Images / **Adversarial** Examples

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford