CS 4803 / 7643: Deep Learning

Topics:
  – (Finish) Visualization
  – Training NNs

Zsolt Kira
Georgia Tech
Administrative

• PS2/HW2 Out, Due Feb 26th! (note 1-day extension)

• Added project ideas
  – Fill out spreadsheet! https://gtvault-my.sharepoint.com/:x:/g/personal/sdharur3_gatech-edu/EVXbNc4oxelMmj1T5WsEIRQBE4Hn532GeLQVcmOnWdG2Jg?e=dIGNfX
  – **Due March 12th** but you don’t have to wait to get feedback!
Recap

• CNN Architectures
• Visualization
example 5x5 filters
(32 total)
Visualizing Learned Filters

Layer 1

Layer 2

Figure Credit: [Zeiler & Fergus ECCV14]
Visualizing Learned Filters

Layer 4

Layer 5

Figure Credit: [Zeiler & Fergus ECCV14]
Case Study: GoogLeNet

Inception module

ILSVRC 2014 winner (6.7% top 5 error)
Recently: Neural Architecture Search

- Can we better search the space of architectures?
- Several approaches
  - Evolutionary algorithms (typically of repeated blocks)
  - Training overparameterized networks & pruning

![Chart showing accuracy vs. number of parameters for various architectures.](chart.png)

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Top1 Acc.</th>
<th>#Params</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-152 (He et al., 2016)</td>
<td>77.8%</td>
<td>60M</td>
</tr>
<tr>
<td>EfficientNet-B1</td>
<td>78.8%</td>
<td>7.8M</td>
</tr>
<tr>
<td>ResNeXt-101 (Xie et al., 2017)</td>
<td>80.9%</td>
<td>84M</td>
</tr>
<tr>
<td>EfficientNet-B3</td>
<td>81.1%</td>
<td>12M</td>
</tr>
<tr>
<td>SENet (Hu et al., 2018)</td>
<td>82.7%</td>
<td>146M</td>
</tr>
<tr>
<td>NASNet-A (Zoph et al., 2018)</td>
<td>82.7%</td>
<td>89M</td>
</tr>
<tr>
<td>EfficientNet-B4</td>
<td>82.6%</td>
<td>19M</td>
</tr>
<tr>
<td>GPipe (Huang et al., 2018)†</td>
<td>84.3%</td>
<td>556M</td>
</tr>
<tr>
<td>EfficientNet-B7</td>
<td>84.4%</td>
<td>66M</td>
</tr>
</tbody>
</table>

† Not plotted

(C) Dhruv Batra & Zsolt Kira
Visualization

• What do individual neurons look for in images?
  – Visualizing filters
  – Last layer embeddings
  – Visualizing activations
  – Maximally activating patches
• How pixels affect model decisions?
  – Occlusion maps
  – Salient or “important” pixels
    • Gradient-based visualizations
Visualization

• **What do individual neurons look for in images?**
  – Visualizing filters
  – Last layer embeddings
  – Visualizing activations
  – Maximally activating patches

• **How pixels affect decisions?**
  – Occlusion maps
  – Salient or “important” pixels
    • Gradient-based visualizations
Visualizing filters in 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

Huang et al, “Densely Connected Convolutional Networks”, CVPR 2017
Visualizing filters in intermediate layers

Visualize the filters/kernels (raw weights)

We can visualize filters at higher layers, but not that interesting

layer 1 weights
16 x 3 x 7 x 7

layer 2 weights
20 x 16 x 7 x 7

layer 3 weights
20 x 20 x 7 x 7

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Visualizing activations in intermediate layers
Maximally Activating Patches

Pick a layer and a channel; e.g. conv5 is 128 x 13 x 13, pick channel 17/128

Run many images through the network, record values of chosen channel

Visualize image patches that correspond to maximal activations

Figure copyright Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller, 2015; reproduced with permission.

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Visualize the “space” of FC7 feature vectors by reducing dimensionality of vectors from 4096 to 2 dimensions

Simple algorithm: Principal Component Analysis (PCA)

More complex: **t-SNE**

Van der Maaten and Hinton, “Visualizing Data using t-SNE”, JMLR 2008
Figure copyright Laurens van der Maaten and Geoff Hinton, 2008. Reproduced with permission.
Last Layer: Dimensionality Reduction

Van der Maaten and Hinton, "Visualizing Data using t-SNE", JMLR 2008
Krizhevsky et al, "ImageNet Classification with Deep Convolutional Neural Networks", NIPS 2012. Figure reproduced with permission.

See high-resolution versions at http://cs.stanford.edu/people/karpathy/cnnembed/

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

- What do individual neurons look for in images?
  - Visualizing filters
  - Last layer embeddings
  - Visualizing activations
  - Maximally activating patches

- How pixels affect decisions from CNNs?
  - Occlusion maps
  - Salient or “important” pixels
    - Gradient-based visualizations
How pixels affect decisions?
Visual Explanations

Where does an intelligent system “look” to make its predictions?
Which pixels matter: Occlusion Maps

Idea: Mask part of the image before feeding to CNN, check how much predicted probabilities change

P(elephant) = 0.95

P(elephant) = 0.75

Zeiler and Fergus, “Visualizing and Understanding Convolutional Networks”, ECCV 2014

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Which pixels matter: Occlusion Maps

Mask part of the image before feeding to CNN, check how much predicted probabilities change.

Zeiler and Fergus, “Visualizing and Understanding Convolutional Networks”, ECCV 2014

Boat image is CC0 public domain
Elephant image is CC0 public domain
Go-Karts image is CC0 public domain

Faithful 😊
Very expensive 😞

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Which pixels matter: Saliency via Backprop

Forward pass: Compute probabilities

Compute gradient of (unnormalized) class score with respect to image pixels, take absolute value and max over RGB channels


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


Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Saliency Maps: Segmentation without supervision

Use GrabCut on saliency map

Figures copyright Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, 2014; reproduced with permission.
Rother et al., “Grabcut: Interactive foreground extraction using iterated graph cuts”, ACM TOG 2004

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Remember ReLUs?

\[ h^{l+1} = \text{ReLU}(h^l) = \max\{0, h^l\} \]

\[
\frac{\partial h^{l+1}}{\partial h^l} = \begin{cases} 
0 & \text{if } h^l < 0 \\
1 & \text{if } h^l > 0 
\end{cases} = [[h^l > 0]]
\]
\[ h^{l+1} = \max\{0, h^l\} \]

**Forward pass**

\[
\begin{align*}
  h^l & \quad \rightarrow \\
  \begin{pmatrix}
    1 & -1 & 5 \\
    2 & -5 & -7 \\
    -3 & 2 & 4 \\
  \end{pmatrix}
  & \quad \rightarrow \\
  \begin{pmatrix}
    1 & 0 & 5 \\
    2 & 0 & 0 \\
    0 & 2 & 4 \\
  \end{pmatrix}
\end{align*}
\]

**Backward pass:**

**Backpropagation**

\[
\begin{align*}
  \frac{\partial L}{\partial h^l} & = [h^l > 0] \frac{\partial L}{\partial h^{l+1}} \\
  \begin{pmatrix}
    -2 & 0 & -1 \\
    6 & 0 & 0 \\
    0 & -1 & 3 \\
  \end{pmatrix}
  & \quad \leftarrow \\
  \begin{pmatrix}
    -2 & 3 & -1 \\
    6 & -3 & 1 \\
    2 & -1 & 3 \\
  \end{pmatrix}
\end{align*}
\]

**Backward pass:**

**Deconvnet**

\[
\begin{align*}
  \frac{\partial L}{\partial h^l} & = [\frac{\partial L}{\partial h^{l+1}} > 0] \frac{\partial L}{\partial h^{l+1}} \\
  \begin{pmatrix}
    0 & 3 & 0 \\
    6 & 0 & 1 \\
    2 & 0 & 3 \\
  \end{pmatrix}
  & \quad \leftarrow \\
  \begin{pmatrix}
    -2 & 3 & -1 \\
    6 & -3 & 1 \\
    2 & -1 & 3 \\
  \end{pmatrix}
\end{align*}
\]

**Backward pass:**

**Guided backpropagation**

\[
\begin{align*}
  \frac{\partial L}{\partial h^l} & = [h^l > 0 \&\& \frac{\partial L}{\partial h^{l+1}} > 0] \frac{\partial L}{\partial h^{l+1}} \\
  \begin{pmatrix}
    0 & 0 & 0 \\
    6 & 0 & 0 \\
    0 & 0 & 3 \\
  \end{pmatrix}
  & \quad \leftarrow \\
  \begin{pmatrix}
    -2 & 3 & -1 \\
    6 & -3 & 1 \\
    2 & -1 & 3 \\
  \end{pmatrix}
\end{align*}
\]
Gradient-based visualizations

Backprop for `dog'

Guided Backprop for `cat'

Guided Backprop for `dog'

Backprop for `cat'

Backprop for `dog'

Noisy

Not Class-Discriminative
Grad-CAM
Visual Explanations from Deep Networks via Gradient-based Localization
[ICCV '17]

Ramprasaath Selvaraju       Michael Cogswell       Abhishek Das       Ramakrishna Vedantam

Devi Parikh       Dhruv Batra
Grad-CAM Motivation

- Perturb semantic neurons in the image and see how it affects the decision

- Last convolutional layer forms a best compromise between high-level semantics and detailed spatial resolution
Guided Grad-CAM

Grad-CAM

Guided Grad-CAM

Rectified Conv Feature Maps

CNN

Any Task-specific Network

Backprop till conv

Grad-CAM

Guided Backpropagation

Gradients ➔ Activations

Neuron Importance

$$\alpha_k^j = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

grads via backprop

Image Classification

Image Captioning

Visual Question Answering

Is there a cat? Question

RNN/LSTM

FC Layer

Yes

...
Guided Grad-CAM
What animal is in this picture? Dog

- Even simple non-attention based CNN + LSTM models learn to look at appropriate regions
Grad-CAM Visual Explanations for VQA

What animal is in this picture? Cat

- Even simple non-attention based CNN + LSTM models learn to look at appropriate regions
Generating prototypical images for a class
Visualizing CNN features: Gradient Ascent on Pixels

**(Guided) backprop:**
Find the part of an image that a neuron responds to?

**Gradient ascent on pixels:**
Generate a synthetic image that maximally activates a neuron

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

Neuron value  Natural image regularizer

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Visualizing CNN features: Gradient Ascent on Pixels

1. Initialize image to zeros

Repeat:
2. Forward image to compute current scores
3. Backprop to get gradient of neuron value with respect to image pixels
4. Make a small update to the image

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Visualizing CNN features: Gradient Ascent on Pixels

\[
\text{arg max}_I S_c(I) - \lambda \|I\|_2^2
\]

Simple regularizer: Penalize L2 norm of generated image
Visualizing CNN features: Gradient Ascent on Pixels

\[ \arg \max_I S_c(I) - \lambda \| I \|_2^2 \]

Simple regularizer: Penalize L2 norm of generated image

Figures copyright Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, 2014; reproduced with permission.

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Fooling Images / Adversarial Examples

(1) Start from an arbitrary image
(2) Pick an arbitrary class
(3) Modify the image to maximize the class
(4) Repeat until network is fooled

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Fooling Images / Adversarial Examples

[African elephant image]

[koala image]

Difference

[10x Difference]

[schooner image]

[iPod image]

Difference

[10x Difference]

Boat image is CC0 public domain
Elephant image is CC0 public domain

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Given the feature vector can you reconstruct the image?
Feature Inversion

Given a CNN feature vector for an image, find a new image that:
- Matches the given feature vector
- "looks natural" (image prior regularization)

\[ x^* = \arg\min_{x \in \mathbb{R}^{H \times W \times C}} \ell(\Phi(x), \Phi_0) + \lambda \mathcal{R}(x) \]

\[ \ell(\Phi(x), \Phi_0) = \|\Phi(x) - \Phi_0\|^2 \]

\[ \mathcal{R}_{V^\beta}(x) = \sum_{i,j} \left( (x_{i,j+1} - x_{ij})^2 + (x_{i+1,j} - x_{ij})^2 \right)^{\frac{\beta}{2}} \]

Feature Inversion

Reconstructing from different layers of VGG-16

\( y \quad \text{relu2}_2 \quad \text{relu3}_3 \quad \text{relu4}_3 \quad \text{relu5}_1 \quad \text{relu5}_3 \)
Side-effect - style transfer

- **Content representation:** feature map at each layer
- **Style representation:** Covariance matrix at each layer
  - Spatially invariant
  - Average second-order statistics

- Idea: Optimize x to match content of one image and style of another

\[
G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l.
\]

Neural Style

Step 1: Extract **content targets** (ConvNet activations of all layers for the given content image)

*content activations*

e.g.
at CONV5_1 layer we would have a [14x14x512] array of target activations
Neural Style

Step 2: Extract **style targets** (Gram matrices of ConvNet activations of all layers for the given style image)

Example of style gram matrices:

\[ G = V^T V \]

E.g., at CONV1 layer (with [224x224x64] activations) would give a [64x64] Gram matrix of all pairwise activation covariances (summed across spatial locations)
Neural Style

Step 3: Optimize over image to have:
- The **content** of the content image (activations match content)
- The **style** of the style image (Gram matrices of activations match style)

\[
L_{total}(p, d, x) = \alpha L_{content}(p, x) + \beta L_{style}(d, x)
\]

Adapted from Andrej Karpathy
Style transfer
Summary of Methods

• We will use **gradients** to understand and visualize what the network has captured.
  – We will also sometimes use **activations** as a representation of the image (just as the last hidden feature vector can be a low-dimensional representation of the image content, we can do this across all layers)

• Each method consists of:
  – **What the target category & layer/filter is:** sometimes a target **category**, sometimes a specific **layer** or slice of the output map (corresponding to a **filter/kernel**) is also chosen
  – **An input:** sometimes an image, sometimes a zero image
  – **Optimization criteria:** What is being optimized, sometimes the class score is maximized (gradient **ascent**), sometime a loss is **minimized**