Deep Convolutional Neural Nets

COMPSCI 527 — Computer Vision
Outline

1 Why Neural Networks?

2 Circuits

3 Neurons, Layers, and Networks

4 AlexNet
Why Neural Networks?

• Neural networks are very expressive (large $\mathcal{H}$)
• Can approximate any well-behaved function from a hypercube $X$ in $\mathbb{R}^d$ to an interval $Y$ in $\mathbb{R}$ within any $\epsilon > 0$
• Universal approximators
• However
  • Complexity grows exponentially with $d = \dim(X)$
  • $L_T$ is not convex (not even close)
  • Large $\mathcal{H} \Rightarrow$ overfitting $\Rightarrow$ lots of data!
• Amazon’s Mechanical Turk made neural networks possible
• Even so, we cannot keep up with the curse of dimensionality!
Why Neural Networks?

- Neural networks are data hungry
- Availability of lots of data is not a sufficient explanation
- There must be deeper reasons
- Special structure of image space (or audio space)?
- Specialized network architectures?
- Regularization tricks and techniques?
- We don’t really know. Stay tuned...
- Be prepared for some hand-waving and empirical statements
Circuits

• Describe implementation of $h : X \to Y$ on a computer
• Algorithm: A sequence of finite steps
• Circuit: Many gates of few types, wired together

- These are NAND gates. We'll use neurons
• Algorithms and circuits are equivalent
• Algorithm can simulate a circuit
• Computer is a circuit that runs algorithms!
• Computer really only computes Boolean functions...
Deep Neural Networks as Circuits

• Neural networks are typically described as circuits
• Nearly always implemented as algorithms
• One gate, the neuron
• Many neurons that receive the same input form a layer
• A cascade of layers is a network
• A deep network has many layers
• Layers with a special constraint are called convolutional
The Neuron

- $y = \rho(a(x))$ where $a = v^T x + b$
- $\mathbf{x} \in \mathbb{R}^d, \ y \in \mathbb{R}$
- $v$ are the gains, $b$ is the bias
- Together, $\mathbf{w} = [v, b]^T$ are the weights
- $\rho(a) = \max(0, a)$ (ReLU, Rectified Linear Unit)
The Neuron as a Pattern Matcher (Almost)

- Left pattern is a drumbeat $\mathbf{g}$ (a pattern template):
- Which of the other two patterns $\mathbf{x}$ is a drumbeat?
- Normalize both $\mathbf{g}$ and $\mathbf{x}$ so that $\|\mathbf{g}\| = \|\mathbf{x}\| = 1$
- Then $\mathbf{g}^T \mathbf{x}$ is the cosine of the angle between the patterns
- If $\mathbf{g}^T \mathbf{x} \geq -b$ for some threshold $-b$, output $a = \mathbf{g}^T \mathbf{x} + b$ (amount by which the cosine exceeds the threshold)
  otherwise, output 0
- $y = \rho(\mathbf{g}^T \mathbf{x} + b)$
The Neuron as a Pattern Matcher (Almost)

- $y = \rho(v^T x + b)$
- A neuron is a pattern matcher, except for normalization...
- ...and if followed by a (trivial) classifier
- In neural networks, normalization may happen in later or earlier layers, and classification happens at the end
- This interpretation is not necessary to understand neural networks
- Nice to have a mental model, though
- Many neurons wired together can approximate any function we want
- A neural network is a *function approximator*
Layers and Networks

- A *layer* is a set of neurons that share the same input:

\[ y_i \]

- A *neural network* is a cascade of layers:

\[ y = \rho(Vx + b) \]

- A neural network is *deep* if it has many layers.
- *Two* layers can make a universal approximator.
- If neurons did not have nonlinearities, any cascade of layers would collapse to a single layer.
Neurons, Layers, and Networks

Convolutional Layers

- A layer with input $x \in \mathbb{R}^d$ and output $y \in \mathbb{R}^e$ has $e$ neurons, each with $d$ gains and one bias.
- Total of $(d + 1)e$ weights to be trained in a single layer.
- For images, $d, e$ are in the order of hundreds of thousands or even millions.
- Too many parameters.
- Convolutional layers are layers restricted in a special way.
- Many fewer parameters to train.
- Also good justification in terms of heuristic principles.

$d, e \sim 10^{5-6}$  
$ed \sim 10^{10-12}$

YANN LECUN
Hierarchical Image Model

- To find a person, look for a face, a torso, limbs,...
- To find a face, look for eyes, nose, ears, mouth, hair,...
- To find an eye look for a circle, some corners, some curved edges,...
- A hierarchical image model is less sensitive to viewpoint, body configuration, ...
- Hierarchy leads to a cascade of layers
- Low-level features are local: A neuron doesn’t need to see the entire image
- Circles are circles, regardless of where they show up: A single neuron can be reused to look for circles anywhere in the image
Correlation, Locality, and Reuse

• Does the drumbeat on the left show up in the clip on the right?

• Drumbeat $g$ has 25 samples, clip $x$ has 100

• Make $100 - 25 + 1 = 76$ neurons that look for $g$ in every possible position

• $y_i = \rho(v_i^T x + b_i)$ where $v_i^T = [0, \ldots, 0, g_0, \ldots, g_{24}, 0, \ldots 0]$

• Gain matrix $V = V_0 \cdots V_{24}$
Compact Computation

- Gain matrix $V = \begin{bmatrix} g_0 & \cdots & g_{24} \\ \\ \vdots & \ddots & \vdots \\ 0 & \cdots & g_0 \end{bmatrix}$
- $a_i = v_i^T x = \sum_{\ell=0}^{24} g_{\ell} x_{i+\ell}$ for $i = 0, \ldots, 75$
- In general,
  $$a_i = \sum_{\ell=0}^{k-1} g_{\ell} x_{i+\ell} \quad \text{for} \quad i = 0, \ldots, e-1 = 0, \ldots, d-k$$

- (One-dimensional) correlation (or convolution with $g[:: -1]$)
- $g$ is the kernel