

Machine Learning & Data Mining CS/CNS/EE 155

Lecture 16: Deep Learning

Announcements

- Last two lectures will not be covered on Final Exam
 - Come for the love of (machine) learning
 - Will release some practice questions for topics not covered in homeworks
- Final Exam will be released on March 16th
- Student have 48 hours from time of download
 - Open book of everything on course website
 - No collaboration

Recap: Linear Models

• Linear scoring function in input features:

$$f(x \mid w, b) = w^T x - b$$

 Sometimes non-linear transform at the end – E.g., logistic Regression

$$P(y=1 \mid x, w, b) = \tau(f(x \mid w, b)) = \frac{1}{1 + \exp\{-f(x \mid w, b)\}}$$

Recap: Multiclass Logistic Regression

Binary LR:
$$P(y=1 | x, w, b) = \frac{1}{1 + e^{-(w^T x - b)}}$$
 $y \in \{0, 1\}$

"Log Linear" Property: $P(y=1 | x, w, b) \propto e^{w^T x - b}$

Extension to Multiclass:
$$P(y = k | x, w, b) \propto e^{w_k^T x - b_k}$$
 Keep a (w_k, b_k) for each class

Multiclass LR:
$$P(y = k \mid x, w, b) = \frac{e^{w_k^T x - b_k}}{\sum_m e^{w_m^T x - b_m}}$$
 $y \in \{1, ..., K\}$

Train via Gradient Descent:

$$\partial_w \sum_{(x,y)\in S} -\log P(y \mid x, w, b)$$

Example: Handwritten Digit Recognition

$$P(y=2|\mathbf{R},w,b)$$

$$P(y=9|\mathcal{D},w,b)$$

- What is feature representation x?
 - Each pixel is a feature
 - Logistic regression yields ≈80% accuracy
 - Can we do better?

Errors In Linear Logistic Regression

• Often makes mistakes on 8's:



• Shares many pixels with 5's and 3's:



- Linear model on pixels not powerful enough
 - E.g., doesn't capture interactions between pixels

http://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html Lecture 16: Deep Learning

Feature Engineering

- Linear models require good features x
- Directed edge detection:
 - (With some blurring)
 - "Oriented Gradients"

Bottom Left Edge

Top Right Edge Left Edge Top Left Edge

• Logistic regression yields ≈90% accuracy

http://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html Lecture 16: Deep Learning

Comparing 8's vs 3's

• New feature representation better distinguishes between 8's and 3's:



http://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html

Learn Features Automatically?

- Feature engineering is tedious
 - Don't know which ones are good
 - Can we just learn them automatically?
- Actually, we did!
 - From convolutional net
 - Learns features
 - Learns logistic regression
 - "Deep Learning"



Outline For Today

- Introduction to Deep Learning

 Learning Features for Predictive Modeling
- Deep Convolutional Networks
 Very popular in Computer Vision
- Tips for Training Deep Networks
- Brief Overview of other Deep Networks

Outline For Today

- Introduction to Deep Learning

 Learning Features for Predictive Modeling
- Deep Convolutional Networks
 Very popular in Computer Vision
- Tips for Training Deep Networks
- Brief Overview of other Deep Networks

Recap: 1 Layer Neural Network

- 1 Neuron
 - Takes input x
 - Outputs y



$$f(x | w,b) = w^{T}x - b \longrightarrow y = \tau(f(x))$$

= $w_{1}^{*}x_{1} + w_{2}^{*}x_{2} + w_{3}^{*}x_{3} - b$

~Logistic Regression!
 – Gradient Descent



Recap: 2 Layer Neural Network



- 2 Layers of Neurons
 - 1st Layer takes input x

Non-Linear!

- 2nd Layer takes output of 1st layer
- Can approximate arbitrary functions
 - Provided hidden layer is large enough
 - "fat" 2-Layer Network

Deep Neural Networks



- Why prefer Deep over a "Fat" 2-Layer?
 - Compact Model
 - (exponentially large "fat" model)

Image Source: http://blog.peltarion.com/2014/06/22/deep-learning-and-deep-neural-networks-in-synapse/

Expressive Power

- Deeper networks are "exponentially more expressive" than shallower networks.
- Related Example: Boolean Circuits
 - Thought Experiment: How many gates required if only depth-2 circuits allowed?



http://en.wikipedia.org/wiki/Circuit_complexity

AND & OR as Transfer Functions

 Deep networks can implement AND & OR transfer functions.

 $\operatorname{AND}(f_1, f_2) = \min\{f_1, f_2\}$

 $OR(f_1, f_2) = max\{f_1, f_2\}$ Used in practice

What Happens if No Transfer Function?

• Just linear transforms?



- Deep structure collapses to linear model!
 - Linear operators are associative & commutative
 - Applying a linear operator to a linear operator yields a linear operator



Recap: Training Neural Networks

- Gradient Descent! **
 - (Supervised Learning)



Parameters:

 $-(w_{11}, b_{11}, w_{12}, b_{12}, w_{2}, b_{2})$

 $f(x | w,b) = w^{T}x - b \quad y = \tau(f(x))$

$$\partial_{w_2} \sum_{i=1}^{N} L(y_i, \tau_2) = \sum_{i=1}^{N} \partial_{w_2} L(y_i, \tau_2) = \sum_{i=1}^{N} \partial_{\tau_2} L(y_i, \tau_2) \partial_{w_2} \tau_2 = \sum_{i=1}^{N} \partial_{\tau_2} L(y_i, \tau_2) \partial_{f_2} \tau_2 \partial_{w_2} f_2$$

$$\partial_{w_{1m}} \sum_{i=1}^{N} L(y_i, \tau_2) = \sum_{i=1}^{N} \partial_{\tau_2} L(y_i, \tau_2) \partial_{f_2} \tau_2 \partial_{w_{1m}} f_2 = \sum_{i=1}^{N} \partial_{\tau_2} L(y_i, \tau_2) \partial_{f_2} \tau_2 \partial_{\tau_{1m}} f_2 \partial_{f_{1m}} \tau_{1m} \partial_{w_{1m}} f_{1m}$$

**additional details end of lecture

Backpropagation = Gradient Descent (lots of chain rules)

Original Biological Inspiration

- David Hubel & Torsten Wiesel discovered "simple cells" and "complex cells" in the 1959
 - Some cells activate for simple patterns
 - E.g., lines at certain angles
 - Some cells activate for more complex patterns
 - Appear to take activations of simple cells as input





Image Source:

https://cms.www.countway.harvard.edu/wp/wp-content/uploads/2013/09/0002595_ref.jpg https://cognitiveconsonance.files.wordpress.com/2013/05/c_fig5.jpg Lecture 16: Deep Learning

The Brain is Hierarchical



Image Source: http://www.cs.nyu.edu/~yann/talks/lecun-ranzato-icml2013.pdf

But Let's Not Get Carried Away

- We need some kind of wings to fly
 - But no flapping



• Do we even need wings?



No Longer Biologically Inspired

- Original inspiration created the feed-forward network
- Field is now called "Deep Learning"
 - Most common name
- Really just Automated Feature Learning
 - Lots of optimization tricks

Outline For Today

- Introduction to Deep Learning

 Learning Features for Predictive Modeling
- Deep Convolutional Networks
 Very popular in Computer Vision
- Tips for Training Deep Networks
- Brief Overview of other Deep Networks

Convolutions

- Images typically have invariant patterns
 - E.g., directional gradients are translational invariant:



Apply convolution to local sliding windows

http://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html Lecture 16: Deep Learning

Convolutional Filters

- Applies to an image patch x
 - Converts local window into single value
 - Slide across image

$$x \otimes W = \sum_{ij} W_{ij} x_{ij}$$

Local Image Patch



Left-to-Right Edge Detector

-1	0	+1
-1	0	+1
-1	0	+1

W

Gabor Filters

 Most common low-level convolutions for computer vision



- Grey = 0
- Light = positive
- Dark = negative

http://en.wikipedia.org/wiki/Gabor_filter

Lecture 16: Deep Learning

-1

-1

-1

0

0

0

W

+1

+1

+1

Gaussian Blur Filters

- Weights decay according to Gaussian Distribution
 - Variance term controls radius

Example W: Apply per RGB Channel



- Black = 0
- White = Positive

http://en.wikipedia.org/wiki/Gaussian_blur



Deep Convolutional Networks

- Learn layers of convolutional filters W
 - Apply convolution to outputs of previous layer



- Note: convolutions are linear operators
 - Need non-linear transform
 - Otherwise all layers collapse to single convolution

Image Source: http://cs.nyu.edu/~fergus/presentations/nips2013_final.pdf

Convolutional Layer

• Current Convolutional Layer consists of:



Convolution

$$\max\{0,0\} =$$

Rectilinear Transform

- Simplifies Backprop
- Chain rule super easy
- Also easier to train
- Main modeling concepts!

Combine them to create convolutional layer

$$\max\{ \mathbf{8} \otimes [\mathbf{0}, \mathbf{0}\} = \mathbf{0}$$

Max Pooling



- Assume Convolution Layer is eye detector
- How to make detector more robust to the exact location of the eye?

http://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/tutorial_p2_nnets_ranzato_short.pdf

Max Pooling

Maximum response from a ulletneighborhood of convolutional layer outputs I.e., an OR gate!

http://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/tutorial_p2_nnets_ranzato_short.pdf

Alternative: L2 Pooling



- Softer version of max pooling
 - Harder to differentiate

Local Contrast Normalization

- Standardize output of convolutional layer using mean & variability estimated from neighboring outputs
- Simple Example:



$$\mu_{ij} = \operatorname{mean}\left\{f_{i'j'} | (i', j') \text{ close to } (i, j)\right\}$$

$$\sigma_{ij}^{2} = \operatorname{mean}\left\{ \left(f_{i'j'} - \mu_{i'j'} \right)^{2} \middle| (i',j') \text{ close to } (i,j) \right\}$$

Biologically Inspired!

• Other examples in references below:

http://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/tutorial_p2_nnets_ranzato_short.pdf http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf

input (24x24x1) max activation: 1, min: 0



Input

conv (23x23x8) filter size 6x6x1, stride 1 max activation: 3.73813, min: -8.09174



8 Convolutional Filters in 1st Layer

relu (23x23x8) max activation: 3.73813, min: 0 max gradient: 0.00316, min: -0.00215

Activations:



Rectilinear Transform

pool (11x11x8) pooling size 2x2, stride 2 max activation: 3.29955, min: 0



Max Pooling

http://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html

Deep Convolutional Networks



- Stack multiple layers together
- Multiclass logistic regression at top
- Train using gradient descent

Down Sampling

- Adjacent Sliding Window Convolution

 Yields output of same dimensions as input
- Good to compress into fewer pixels
 Skip a few pixels for each convolution
- "Stride"
 - How far away next convolution is
 - No Down Sampling: Stride = 1
 - Down Sampling 2x: Stride = 2

Also Max Pooling

Online Demo

http://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html

ImageNET

- Object recognition competition (2012)
 - 1.5 Million Labeled Training Examples
 - ≈1000 classes



Leopard



Mushroom



Mite

http://www.image-net.org/

Deep Convolutional Net for ImageNET

- 7 Hidden Layers
 - 5 Convolutional
 - 2 Regular
- Multiclass Logistic Regression at top
- Trained using stochastic gradient descent

 And a lot of tricks
- Won the 2012 ImageNET competition

http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf http://www.image-net.org/challenges/LSVRC/2012/results.html Lecture 16: Deep Learning



http://www.image-net.org/

http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf http://ftp.cs.nyu.edu/~fergus/papers/zeilerECCV2014.pdf

Visualizing CNN (Layer 1)



Visualizing CNN (Layer 2)





Top Image Patches

Visualizing CNN (Layer 3)





Top Image Patches

Part that Triggered Filter

Visualizing CNN (Layer 4)

Ð			15					A CONTRACTOR
R	Q		-		-	Maria Maria	ONTRO	
63			al and a second	S.	and and	a star	- Marine	
	Ś					Ö		9
٢		-OF		13 NO 10 NO 10 NO		0		
3		No.			0 (1)		3	
A CONTRACTOR	Ser.	W.		(a)	J.a.	3	1	5
-	X		and the second s	6.0	Carlos Carlos			
X	T	A CONTRACTOR OF				A. C.	1	

Part that Triggered Filter



Top Image Patches

Visualizing CNN (Layer 5)





Top Image Patches

Use Hidden Layers as Features



- Stack hidden layer activations as new feature representation
- Train an SVM =)
- Generalize to other datasets

Failure Cases



Predicts Correctly

Predicts Incorrectly



Predicts Correctly

Predicts Incorrectly

http://arxiv.org/pdf/1312.6199v4.pdf

Outline For Today

- Introduction to Deep Learning

 Learning Features for Predictive Modeling
- Deep Convolutional Networks
 Very popular in Computer Vision
- Tips for Training Deep Networks
- Brief Overview of other Deep Networks

Training Deep Networks

- Deep Networks are extremely non-convex
 Hard to train well
- Deep Networks have extremely high capacity

 Many parameters, easy to overfit
- Real success stories only in the last 10 years
 - A lot of (annotated) data
 - Increase in computational power
 - Better bag of tricks

Stochastic Gradient Descent + Tricks!

- Some related to choice of model/architecture
 - Rectilinear over sigmoid transfer functions
 - Local contrast normalization
 - Sparse Connections that enable parallelism
 - Ensemble of Deep Networks
- Rest are optimization techniques
 - Gradient Clamping
 - Mini-batching
 - Momentum
 - Adaptive Learning Rates
 - Random Initialization
 - Dropout

http://yyue.blogspot.com/2015/01/a-brief-overview-of-deep-learning.html

Rectilinear vs Sigmoid (The Vanishing Gradient Problem)



$$\partial_{w_{1m}} L(y, \tau_2)$$

$$= \partial_{\tau_2} L(y, \tau_2) \partial_{f_2} \tau_2 \partial_{w_{1m}} f_2$$

$$= \partial_{\tau_2} L(y, \tau_2) \partial_{f_2} \tau_2 \partial_{\tau_{1m}} f_2 \partial_{f_{1m}} \tau_{1m} \partial_{w_{1m}} f_{1m}$$

$$f_{1m} = w_{1m}^T x$$
Large f \rightarrow vanishing gradient
Compounded with more layers

Rectilinear vs Sigmoid (The Vanishing Gradient Problem)



Gradient Clamping

- Rectilinear functions can grow unbounded:
 - Gradients can get very large
 - Compounding effect in lower layers
 - Opposite of the vanishing gradient effect with sigmoids
- Solution: clamp gradients
 E.g., clamp norm to 15



http://yyue.blogspot.com/2015/01/a-brief-overview-of-deep-learning.html Lecture 16: Deep Learning

Dense Convolutional Networks



 Every Convolutional Layer uses every output from previous layer

Sparse Convolutional Networks



Learning Rate & Momentum

$$w = w - \eta \partial_w$$
 Gradient Descent

If validation performance plateaus or gets worse
 Divide learning rate by 2

$$w = w - \eta \partial_w + \gamma m_w$$
 Momentum

Momentum is a weighted combination of recent gradient updates

http://www.cs.toronto.edu/~fritz/absps/momentum.pdf

Dropout

• Randomly turn off nodes during training



- Choose randomly for each SGD minibatch
 - Decorrelates node in each layer
 - Less overfitting

http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf

Outline For Today

- Introduction to Deep Learning

 Learning Features for Predictive Modeling
- Deep Convolutional Networks
 Very popular in Computer Vision
- Tips for Training Deep Networks
- Brief Overview of other Deep Networks

Unsupervised Deep Learning

- Supervised = Learn Feature Encoding
 - Uses supervised label as signal
- Unsupervised = Also Learn Decoding
 - Uses reconstruction error as signal



http://research.google.com/archive/unsupervised_icml2012.html Lecture 16: Deep Learning

Unsupervised Deep Learning

- "Deep" dimensionality reduction
 - Can think of matrix factorization as "shallow" linear dimensionality reduction
- Encoding: convert image to features
 - Matrix Factorization: z = Ux
- Decoding: convert features to image

– Matrix Factorization: $x = U^T z$

Deep Belief Networks

Generative Model

 Encodes image as distribution over hidden state activations

Can sample images given a setting of hidden states

http://www.cs.toronto.edu/~hinton/adi/

Deep Recurrent Networks

- Sequence Prediction
 - x & y are sequences



http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf http://www1.icsi.berkeley.edu/~vinyals/Files/rnn_denoise_2012.pdf

Deep Recursive Networks

- Input: parse tree
- Output: sentiment



- Recursively instantiate model on parse tree
 - Each node takes the outputs of its children, and computes hidden layer activations as output
 - Logistic regression at the top

http://nlp.stanford.edu/sentiment/

Recap: Deep Learning

- Hierarchies (or layers) of non-linear transforms
 - Often interpreted as feature learning
 - Sometimes makes sense/visualizable
- Supervised training at the top layer
 - Unsupervised also possible (less successful)
- Train using stochastic gradient descent
 - But requires a lot of additional tricks
 - Also requires sufficient training data
- Nowhere close to general human cognition

Resources

- http://caffe.berkeleyvision.org/
- <u>http://deeplearning.net/software/theano/</u>
- <u>http://torch.ch/</u>
- <u>https://code.google.com/p/cuda-convnet/</u>
- http://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/
- http://deeplearning.net/tutorial/
- http://deeplearning.stanford.edu/tutorial/
- http://nlp.stanford.edu/sentiment/

Next Lecture

 Semi-supervised Learning & Active Learning – Probably...

- Homework 4 due next Tuesday
- Miniproject 2 due next Friday