Deep Neural Networks Basics

For ECS 289G
Presented by Fanyi Xiao

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Computer Vision in the Pre-DNN Era

Face Detection, Viola & Jones, 2001

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Computer Vision in the Pre-DNN Era

“SIFT” & Object Recognition, David Lowe, 1999

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Computer Vision in the Pre-DNN Era

Spatial Pyramid Matching, Lazebnik, Schmid & Ponce, 2006

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Computer Vision in the Pre-DNN Era

Histogram of Gradients (HoG)
Dalal & Triggs, 2005

Deformable Part Model
Felzenswalb, McAllester, Ramanan, 2009

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Emergence of DNNs in Vision

**Year 2010**

NEC-UIUC

- Dense grid descriptor: HOG, LBP
- Coding: local coordinate, super-vector
- Pooling, SPM
- Linear SVM

[Lin CVPR 2011]

**Year 2012**

SuperVision

**Year 2014**

GoogLeNet

- Convolution
- Pooling
- Softmax
- Other

VGG

- maxpool
- conv-512
- conv-128
- conv-512
- conv-128
- conv-512
- conv-256
- conv-256
- conv-64
- conv-64
- maxpool

MSRA

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Neural Networks

Image Classification

Learn visual features "end-to-end"

Data-driven approach

- airplane
- automobile
- bird
- cat
- deer
- dog
- frog
- horse
- ship
- truck

assume given set of discrete labels
{dog, cat, truck, plane, ...}

→ cat
Compositional Models
Learned End-to-End

Hierarchy of Representations
- vision: pixel, motif, part, object
- text: character, word, clause, sentence
- speech: audio, band, phone, word

Neural Networks

figure credit Yann LeCun, ICML ’13 tutorial

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Neural Networks

Three key ingredients for training an NN:

1. Score function
2. Loss function
3. Optimization
Neural Networks

Three key ingredients for training an NN:

1. Score function: $y = f(x, W)$

$x$ -- 224*224*3 image patch

$y$ -- 1000d vector

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Neural Networks

Three key ingredients for training an NN:

2. Loss function: for example max-margin loss and cross-entropy loss

\[ L_i = \sum_{j \neq y_i} \max(0, f(x_i, W)_j - f(x_i, W)_{y_i} + \Delta) \]

\[ L_i = -\log \left( \frac{e^{f_{y_i}}}{\sum_{j} e^{f_j}} \right) \]
Neural Networks

Three key ingredients for training an NN:

3. Optimization: simple gradient descent!
Neural Networks

Three key ingredients for training an NN:

3. Optimization: in practice, *stochastic (mini-batch) gradient descent*!

```python
# Vanilla Minibatch Gradient Descent

while True:
    data_batch = sample_training_data(data, 256)  # sample 256 examples
    weights_grad = evaluate_gradient(loss_fun, data_batch, weights)
    weights += -step_size * weights_grad  # perform parameter update
```
Neural Networks

Three key ingredients for training an NN:

3. Optimization: in practice, stochastic (mini-batch) gradient descent + momentum! (Many other optimization methods like adagrad/rmsprop)

```
weights_grad = evaluate_gradient(loss_fun, data, weights)
vel = vel * 0.9 - step_size * weights_grad
weights += vel
```
Convolution Neural Networks

Let's take a closer look at AlexNet

Linear transformation: \( y' = Wx + b \)
Convolution Neural Networks

Let's take a closer look at AlexNet

Linear transformation: $y' = Wx + b$
Let's take a closer look at AlexNet

conv(h,w,stride)
Convolution Neural Networks

conv(h,w,stride)
Convolution Neural Networks

Example: conv(h=3, w=3, stride=1)

(7-3)/1+1=5
End up as a 5*5 feature map
Let's take a closer look at AlexNet

maxpool(h, w, stride)
Convolution Neural Networks

Example: maxpool(h=2,w=2,stride=2)

Single depth slice

```
<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
```

max pool with 2x2 filters and stride 2

```
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
```
Convolution Neural Networks

Let's take a closer look at AlexNet

ReLU: $y = \max(y', 0)$
Convolution Neural Networks

Problems with tanh:
Saturated response

\[ \tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \]

\[ f(x) = \max(0, x) \]

Relu: \( y = \max(y', 0) \)
- Does not saturate
- Very computationally efficient
- Converges much faster than sigmoid/tanh in practice!

However, non-bounded response and dead when less than 0
(improved version leaky ReLU)

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
There are two key differences to Vanilla Neural Nets: neurons arranged in 3D volumes have local connectivity, share parameters.
Convolution Neural Networks

ILSVRC14 Winners: ~6.6% Top-5 error
- GoogLeNet: composition of multi-scale dimension-reduced modules (pictured)
- VGG: 16 layers of 3x3 convolution interleaved with max pooling + 3 fully-connected layers

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Convolution Neural Networks

Object Detection

R-CNN: Region-based Convolutional Networks
http://nbviewer.ipython.org/github/BVLC/caffe/blob/master/examples/detection.ipynb
Full R-CNN scripts available at
https://github.com/rbgirshick/rcnn

Ross Girshick et al.

Fast R-CNN
arXiv and code

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Segmentation

Fully convolutional networks for pixel prediction applied to semantic segmentation end-to-end learning efficiency in inference and learning 175 ms per-image prediction multi-modal, multi-task

Further applications
- depth estimation
- denoising

arXiv and pre-release

Jon Long* & Evan Shelhamer*,
Problem with Feed-forward Nets

What if we want to be able to have a model telling us what's the probability of the following two sentences, respectively:

1. The cat sat on the mat
2. The mat is having dinner with the cat
Problem with Feed-forward Nets

What if we want to be able to have a model telling us what's the probability of the following two sentences, respectively:

1. The cat sat on the mat
2. The mat is having dinner with the cat

Cannot handle variable length input

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Recurrent Neural Net

RNNs tie the weights at each time step

\[ h_t = \sigma \left( W^{(hh)} h_{t-1} + W^{(hx)} x_t \right) \]

\[ \hat{y}_t = \text{softmax} \left( W^{(S)} h_t \right) \]
Recurrent Neural Net

Training of RNNs is hard...

\[ h_t = W f(h_{t-1}) + W^{(hx)} x_t \]
\[ \frac{\partial h_t}{\partial h_k} = \prod_{j=k+1}^{t} \frac{\partial h_j}{\partial h_{j-1}} = \prod_{j=k+1}^{t} W^T \text{diag}[f'(h_{j-1})] \]
Recurrent Neural Net

Training of RNNs is hard...

Solution 1: clip the gradient!

Algorithm 1 Pseudo-code for norm clipping the gradients whenever they explode

\[
\hat{g} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}
\]

if \( \| \hat{g} \| \geq \text{threshold} \) then

\[
\hat{g} \leftarrow \frac{\text{threshold}}{\| \hat{g} \|} \hat{g}
\]

end if

Some theory: On the difficulty of training recurrent neural networks, Pascanu et al. ICML2013

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Recurrent Neural Net

Training of RNNs is hard...

Solution 2: NNs with gating units (LSTM/GRU)
Recurrent Neural Net

Training of RNNs is hard...

Solution 2: nets with gating units (LSTM/GRU)
Recurrent Neural Net

Training of RNNs is hard...

Solution 2: nets with gating units (LSTM/GRU)
RNN in vision

Image captioning


Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
RNN in vision

Visual attention model

1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, Kelvin Xu et al.

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
RNN in vision

RNNs for Human Dynamics

Recurrent Network Models for Human Dynamics, Katerina Fragkiadaki et al.

Most materials taken from Andrej Karpathy/Richard Socher/Nando de Freitas/caffe CVPR tutorial
Tricks

1. Numerical gradient check

```python
fx = f(x)  # evaluate function value at original point
grad = np.zeros_like(x)
# iterate over all indexes in x
it = np.nditer(x, flags=['multi_index'], op_flags=['readwrite'])
while not it.finished:
    # evaluate function at x+h
    ix = it.multi_index
    oldval = x[ix]
    x[ix] = oldval + h  # increment by h
    fxph = f(x)  # evaluate f(x + h)
    x[ix] = oldval - h
    fxmh = f(x)  # evaluate f(x - h)
    x[ix] = oldval  # restore

    # compute the partial derivative with centered formula
    grad[ix] = (fxph - fxmh) / (2 * h)  # the slope
    if verbose:
        print(ix, grad[ix]
it.iternext()  # step to next dimension
```
Tricks

1. Numerical gradient check
2. Modulize layers: only three functions needed
   (1) output=forward(input,model)
   (2) dJ_dW=computeParamGrad(input,outputGrad,model)
   (3) dJ_dInput=computeInputGrad(input,outputGrad,model)
   Everything else is just putting together lego pieces
Questions?

Thanks!