Deep neural networks

June 1st, 2017

Yong Jae Lee
UC Davis

Many slides from Rob Fergus, Svetlana Lazebnik, Jia-Bin Huang, Derek Hoiem

Announcements

• Post questions on Piazza for review-session (6/8 lecture)

Outline

• Deep Neural Networks
• Convolutional Neural Networks (CNNs)
Traditional Image Categorization:
Training phase

- Training Images
- Training Labels
- Image Features
- Classifier Training
- Trained Classifier

Traditional Image Categorization:
Testing phase

- Training Images
- Training Labels
- Image Features
- Classifier Training
- Trained Classifier

- Test Image
- Image Features
- Trained Classifier
- Prediction

Features have been key...

Hand-crafted
What about learning the features?

- Learn a feature hierarchy all the way from pixels to classifier
- Each layer extracts features from the output of previous layer
- Layers have (nearly) the same structure
- Train all layers jointly

Learning Feature Hierarchy

Goal: Learn useful higher-level features from images

- Better performance
- Other domains (unclear how to hand engineer):
  - Kinect
  - Video
  - Multi spectral
- Feature computation time
  - Dozens of features now regularly used
  - Getting prohibitive for large datasets (10’s sec /image)
“Shallow” vs. “deep” architectures

Traditional recognition: “Shallow” architecture

Deep learning: “Deep” architecture

Biological neuron and Perceptrons

Simple, Complex, and Hyper-complex cells

Suggested a hierarchy of feature detectors in the visual cortex, with higher level features responding to patterns of activation in lower level cells, and propagating activation upwards to still higher level cells.

David H. Hubel and Torsten Wiesel
Hubel/Wiesel Architecture and Multi-layer Neural Network

Hubel and Weisel’s architecture

Multi-layer Neural Network
- A non-linear classifier

Neuron: Linear Perceptron
- Inputs are feature values
- Each feature has a weight
- Sum is the activation

activation_{\text{lin}}(x) = \sum w_i \cdot f_i(x) = w \cdot f(x)

- If the activation is:
  - Positive, output +1
  - Negative, output -1

Two-layer perceptron network
Two-layer perceptron network

Learning \( w \)

- Training examples
  \[
  (x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})
  \]

- Objective: a misclassification loss
  \[
  \min_w \sum_{i=1}^{m} \left( y^{(i)} - h_w(f(x^{(i)})) \right)^2
  \]

- Procedure:
  - Gradient descent / hill climbing
Hill climbing

• Simple, general idea:
  – Start wherever
  – Repeat: move to the best neighboring state
  – If no neighbors better than current, quit
  – Neighbors = small perturbations of w

• What’s bad?
  – Optimal?
Two-layer neural network

- Theorem (Universal function approximators): A two-layer network with a sufficient number of neurons can approximate any continuous function to any desired accuracy.

- Practical considerations:
  - Can be seen as learning the features
  - Large number of neurons
    - Danger for overfitting
  - Hill-climbing procedure can get stuck in bad local optima

Multi-layer Neural Network

- A non-linear classifier
- **Training**: find network weights \( \mathbf{w} \) to minimize the error between true training labels and estimated labels:

\[
E(\mathbf{w}) = \sum_{i=1}^{N} (y_i - f_\mathbf{w}(x_i))^2
\]

- Minimization can be done by gradient descent provided \( f \) is differentiable
- This training method is called **back-propagation**
Outline

- Deep Neural Networks
- Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNN, ConvNet, DCN)

- CNN = a multi-layer neural network with
  - **Local** connectivity:
    - Neurons in a layer are only connected to a small region of the layer before it
  - **Share** weight parameters across spatial positions:
    - Learning shift-invariant filter kernels

Image credit: A. Karpathy

Neocognitron

[Fukushima, Biological Cybernetics 1980]

Deformation-Resistant Recognition
- S-cells: (simple)
  - extract local features
- C-cells: (complex)
  - allow for positional errors
LeNet [LeCun et al. 1998]

- Stack multiple stages of feature extractors
- Higher stages compute more global, more invariant features
- Classification layer at the end

Gradient-based learning applied to document recognition [LeCun, Bottou, Bengio, Haffner 1998]

LeNet-1 from 1993

Convolutional Neural Networks

What is a Convolution?

- Weighted moving sum
Why Convolution?

- Few parameters (filter weights)
- Dependencies are local
- Translation invariance

Convolutional Neural Networks

- Feature maps
- Spatial pooling
- Non-linearity
- Convolution (Learned)

Convolutional Neural Networks

- Feature maps
- Spatial pooling
- Non-linearity
- Convolution (Learned)

Rectified Linear Unit (ReLU)
Non-Linearity

- Per-element (independent)
- Options:
  - Tanh
  - Sigmoid: \(1/(1+\exp(-x))\)
  - Rectified linear unit (ReLU)
    - Makes learning faster
    - Simplifies backpropagation
    - Avoids saturation issues
      ➔ Preferred option

Convolutional Neural Networks

Spatial Pooling

- Average or max
- Non-overlapping / overlapping regions
- Role of pooling:
  - Invariance to small transformations
  - Larger receptive fields (see more of input)
Engineered vs. learned features

Convolutional filters are trained in a supervised manner by back-propagating classification error.

Label
- Convolution/pool
- Convolution/pool
- Convolution/pool
- Convolution/pool
- Convolution/pool
- Dense
- Dense
- Dense
- Dense
- Label

Compare: SIFT Descriptor

Image Pixels
- Apply oriented filters
- Spatial pool (Sum)
- Normalize to unit length
- Feature Vector

Lowe [IJCV 2004]

Compare: Spatial Pyramid Matching

SIFT features
- Filter with Visual Words
- Take max VW response
- Multi-scale spatial pool (Sum)
- Global image descriptor

Lazebnik, Schmid, Ponce [CVPR 2006]
Previous Convnet successes

- Handwritten text/digits
  - MNIST (0.17% error [Ciresan et al. 2011])
  - Arabic & Chinese [Ciresan et al. 2012]

- Simpler recognition benchmarks
  - CIFAR-10 (9.3% error [Wan et al. 2013])
  - Traffic sign recognition
    - 0.56% error vs 1.16% for humans [Ciresan et al. 2011]

ImageNet Challenge 2012

- ~14 million labeled images, 20k classes
- Images gathered from Internet
- Human labels via Amazon Turk
- ImageNet Challenge: 1.2 million training images, 1000 classes

AlexNet

Similar framework to LeCun’98 but:
- Bigger model (7 hidden layers, 650,000 units, 60,000,000 params)
- More data ($10^6$ vs. $10^7$ images)
- GPU implementation (50x speedup over CPU)
  - Trained on two GPUs for a week

A. Krizhevsky, I. Sutskever, and G. Hinton,
ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012
AlexNet for image classification

Fixed input size: 224x224x3

ImageNet Classification Challenge


Industry Deployment

• Used in Facebook, Google, Microsoft
• Startups
• Image Recognition, Speech Recognition, ....
• Fast at test time

Taigman et al. DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CVPR '14
Visualizing CNNs

• What input pattern originally caused a given activation in the feature maps?
Layer 3

Beyond classification

- Detection
- Segmentation
- Regression
- Pose estimation
- Matching patches
- Synthesis

and many more…
R-CNN: Regions with CNN features

- Trained on ImageNet classification
- Finetune CNN on PASCAL

Labeling Pixels: Semantic Labels

Labeling Pixels: Edge Detection
CNN for Regression

DeepPose [Toshev and Szegedy CVPR 2014]

CNN as a Similarity Measure for Matching

Stereo matching [Simon and LeCun CVPR 2015]
Compare patch [Kaznacheev and Komodakis 2015]

FaceNet [Rebuffi et al. 2015]

Match ground and aerial images [Lu et al. CVPR 2019]

CNN for Image Generation

Learning to Generate Chairs with Convolutional Neural Networks [Dosovitskiy et al. CVPR 2015]
Chair Morphing

Transfer Learning

• Improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned.
• Weight initialization for CNN

Deep learning libraries

• Tensorflow
• Caffe
• Torch
• MatConvNet
Fooling CNNs

What is going on?

Questions?

See you Tuesday!