Deep Learning for Vision: Tricks of the Trade

Marc'Aurelio Ranzato

Facebook, AI Group

www.cs.toronto.edu/~ranzato
Ideal Features

- window, right
- chair, left
- monitor, top of shelf
- carpet, bottom
- drums, corner
- ... 
- pillows on couch

Q.: What objects are in the image? Where is the lamp? What is on the couch? ...
The Manifold of Natural Images
Ideal Feature Extraction

Pixel 1 → Ideal Feature Extractor → Expression

Pixel 2 → Pose
Learning Non-Linear Features

Given a dictionary of simple non-linear functions: $g_1, \ldots, g_n$

Proposal #1: linear combination

$$f(x) \approx \sum_j g_j$$

Proposal #2: composition

$$f(x) \approx g_1(g_2(\ldots g_n(x)\ldots))$$
Learning Non-Linear Features

Given a dictionary of simple non-linear functions: \( g_1, \ldots, g_n \)

Proposal #1: linear combination
\[
 f(x) \approx \sum_j g_j
\]
- Kernel learning
- Boosting
- ...

Proposal #2: composition
\[
 f(x) \approx g_1(g_2(\ldots g_n(x)\ldots))
\]
- Deep learning
- Scattering networks (wavelet cascade)
- S.C. Zhou & D. Mumford “grammar”
Linear Combination

Prediction of class

Templete matchers

BAD: it may require an exponential nr. of templates!!!

Input image
Composition

Input image

high-level parts

mid-level parts

low level parts

prediction of class

- reuse intermediate parts
- distributed representations

GOOD: (exponentially) more efficient

Lee et al. “Convolutional DBN's ...” ICML 2009, Zeiler & Fergus
A Potential Problem with Deep Learning

Optimization is difficult: non-convex, non-linear system

\[ \theta_1 \rightarrow \theta_2 \rightarrow \theta_3 \rightarrow \theta_4 \]
A Potential Problem with Deep Learning

Optimization is difficult: non-convex, non-linear system

Solution #1: freeze first N-1 layer (engineer the features)
It makes it shallow!
- How to design features of features?
- How to design features for new imagery?
A Potential Problem with Deep Learning

Optimization is difficult: non-convex, non-linear system

Solution #2: live with it!
It will converge to a local minimum.
It is much more powerful!!

Given lots of data, engineer less and learn more!!
Just need to know a few tricks of the trade...
Deep Learning in Practice

Optimization is easy, need to know a few tricks of the trade.

Q: What's the feature extractor? And what's the classifier?
A: No distinction, end-to-end learning!
Deep Learning in Practice

It works very well in practice:
KEY IDEAS: WHY DEEP LEARNING

- We need non-linear system
- We need to learn it from data
- Build feature hierarchies
  - Distributed representations
  - Compositionality
- End-to-end learning
What Is Deep Learning?
Buzz Words

It's a Convolutional Net

It's a Contrastive Divergence

It's a Feature Learning

It's a Unsupervised Learning

It's just old Neural Nets

It's a Deep Belief Net
A Deep Learning method is: a method which makes predictions by using a sequence of non-linear processing stages. The resulting intermediate representations can be interpreted as feature hierarchies and the whole system is jointly learned from data.

Some deep learning methods are probabilistic, others are loss-based, some are supervised, other unsupervised...

It's a large family!
1957
Rosenblatt

Perceptron

THE SPACE OF
MACHINE LEARNING METHODS
80s
back-propagation & compute power
Perceptron
Neural Net
Autoencoder
Neural Net
LeCun’s CNNs
90s
Convolutional
Neural Net
Recurrent
Neural Net
Sparse Coding
Autoencoder
Neural Net
GMM
Perceptron
BayesNP

Deep Belief Net

Convolutional Neural Net

Neural Net

Deep (sparse/denoising) Autoencoder

2012 CNNs (data + GPU)

Autoencoder Neural Net

Sparse Coding

Restricted BM

SVM

Perceptron

Boosting

GMM

Restricted BM

Sparse Coding

Autoencoder

Neural Net

Convolutional Neural Net

Neural Net

Recurrent Neural Net

GMM

BayesNP
Q.: Did we make any progress since then?

A.: The main reason for the breakthrough is: data and GPU, but we have also made networks deeper and more non-linear.
ConvNets: History

- **Fukushima 1980**: designed network with same basic structure but did not train by backpropagation.

- **LeCun from late 80s**: figured out backpropagation for CNN, popularized and deployed CNN for OCR applications and others.

- **Poggio from 1999**: same basic structure but learning is restricted to top layer (k-means at second stage)

- **LeCun from 2006**: unsupervised feature learning

- **DiCarlo from 2008**: large scale experiments, normalization layer

- **LeCun from 2009**: harsher non-linearities, normalization layer, learning unsupervised and supervised.

- **Mallat from 2011**: provides a theory behind the architecture

- **Hinton 2012**: use bigger nets, GPUs, more data

LeCun et al. “Gradient-based learning applied to document recognition” IEEE 1998
ConvNets: till 2012

Common wisdom: training does not work because we “get stuck in local minima”
ConvNets: today

Local minima are all similar, there are long plateaus, it can take long time to break symmetries.

Input/output invariant to permutations

Breaking ties between parameters

Saturating units
Like walking on a ridge between valleys
ConvNets: today

Local minima are all similar, there are long plateaus, it can take long to break symmetries.

Optimization is not the real problem when:
– dataset is large
– unit do not saturate too much
– normalization layer
Today's belief is that the challenge is about:

– generalization
  
  How many training samples to fit 1B parameters?
  How many parameters/samples to model spaces with 1M dim.?

– scalability
Deep Learning is a very rich family!
I am going to focus on a few methods...
CNN

Reccurrent Neural Net

Neural Net

Perceptron

Boosting

SVM

Deep (sparse/denoising) Autoencoder

DBN

Restricted BM

GMM

Sparse Coding

Neural Net

BayesNP

UNSUPERVISED

SUPERVISED
Deep Gated MRF

Layer 1:

\[ E(x, h^c, h^m) = \frac{1}{2} x' \Sigma^{-1} x \]

\[ p(x, h^c, h^m) \propto e^{-E(x, h^c, h^m)} \]

Ranzato et al. “Modeling natural images with gated MRFs” PAMI 2013
Deep Gated MRF

Layer 1:

\[ E(x, h^c, h^m) = \frac{1}{2} x^t C C^t x \]

Ranzato et al. “Modeling natural images with gated MRFs” PAMI 2013
Deep Gated MRF

Layer 1:

\[ E(x, h^c, h^m) = \frac{1}{2} x' C \left[ \text{diag}(h^c) \right] C' x \]

Ranzato et al. “Modeling natural images with gated MRFs” PAMI 2013
Deep Gated MRF

Layer 1:

\[
E(x, h^c, h^m) = \frac{1}{2} x' C \left[ \text{diag}(h^c) \right] C' x + \frac{1}{2} x' x - x' W h^m
\]

Ranzato et al. “Modeling natural images with gated MRFs” PAMI 2013
Deep Gated MRF

Layer 1:

\[ E(x, h^c, h^m) = \frac{1}{2} x' C \left[ \text{diag} \left( h^c \right) \right] C' x + \frac{1}{2} x' x - x' W h^m \]

**Inference of latent variables:**
just a forward pass

**Training:**
requires approximations
(here we used MCMC methods)

\[ p(x) \propto \int_{h^c} \int_{h^m} e^{-E(x, h^c, h^m)} \]

Ranzato et al. “Modeling natural images with gated MRFs” PAMI 2013
Deep Gated MRF

Ranzato et al. “Modeling natural images with gated MRFs” PAMI 2013
Deep Gated MRF

Layer 1

Layer 2

Layer 3

Ranzato et al. “Modeling natural images with gated MRFs” PAMI 2013
Sampling High-Resolution Images

Gaussian model
marginal wavelet
from Simoncelli 2005

Pair-wise MRF
FoE
from Schmidt, Gao, Roth CVPR 2010
Sampling High-Resolution Images

gMRF: 1 layer

Ranzato et al. PAMI 2013

Gaussian model

Pair-wise MRF

FoE

marginal wavelet

from Simoncelli 2005

from Schmidt, Gao, Roth CVPR 2010
### Sampling High-Resolution Images

<table>
<thead>
<tr>
<th>gMRF: 1 layer</th>
<th>Gaussian model</th>
<th>marginal wavelet</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
</tbody>
</table>

*from Simoncelli 2005*

<table>
<thead>
<tr>
<th>Pair-wise MRF</th>
<th>FoE</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
</tr>
</tbody>
</table>

*from Schmidt, Gao, Roth CVPR 2010*

---

Ranzato et al. PAMI 2013
Sampling High-Resolution Images

gMRF: 1 layer

Gaussian model

marginal wavelet

from Simoncelli 2005

Pair-wise MRF

FoE

from Schmidt, Gao, Roth CVPR 2010

Ranzato et al. PAMI 2013
Sampling High-Resolution Images

- gMRF: 3 layers
  - Ranzato et al. PAMI 2013

- Gaussian model
  - from Simoncelli 2005

- marginal wavelet

- Pair-wise MRF
  - from Schmidt, Gao, Roth CVPR 2010

- FoE
Sampling High-Resolution Images

gMRF: 3 layers

Gaussian model

Pair-wise MRF

FoE

marginal wavelet

from Simoncelli 2005

from Schmidt, Gao, Roth CVPR 2010

Ranzato et al. PAMI 2013
Sampling High-Resolution Images

- gMRF: 3 layers
  - Ranzato et al. PAMI 2013

- Gaussian model
  - from Simoncelli 2005

- Pair-wise MRF
  - from Schmidt, Gao, Roth CVPR 2010

- marginal wavelet
  - from Schmidt, Gao, Roth CVPR 2010
Sampling High-Resolution Images

gMRF: 3 layers

Gaussian model
from Simoncelli 2005

Pair-wise MRF
FoE

from Schmidt, Gao, Roth CVPR 2010

Ranzato et al. PAMI 2013
Sampling After Training on Face Images

unconstrained samples

conditional (on the left part of the face) samples

Ranzato et al. PAMI 2013
Pros

- Feature extraction is fast
- Unprecedented generation quality
- Advances models of natural images
- Trains without labeled data

Cons

- Training is inefficient
- Slow
- Tricky
- Sampling scales badly with dimensionality
- What's the use case of generative models?

Conclusion

- If generation is not required, other feature learning methods are more efficient (e.g., sparse auto-encoders).
- What's the use case of generative models?
CONV NETS: TYPICAL ARCHITECTURE

One stage (zoom)

Whole system

Input Image

1st stage

2nd stage

3rd stage

Fully Conn. Layers

Class Labels
CONV NETS: TYPICAL ARCHITECTURE

One stage (zoom)

Conceptually similar to:

SIFT $\rightarrow$ K-Means $\rightarrow$ Pyramid Pooling $\rightarrow$ SVM
Lazebnik et al. “...Spatial Pyramid Matching...” CVPR 2006

SIFT $\rightarrow$ Fisher Vect. $\rightarrow$ Pooling $\rightarrow$ SVM
CONV NETS: EXAMPLES

- OCR / House number & Traffic sign classification

Ciresan et al. “MCDNN for image classification” CVPR 2012
CONV NETS: EXAMPLES

- Texture classification

Sifre et al. “Rotation, scaling and deformation invariant scattering...” CVPR 2013
CONV NETS: EXAMPLES

- Pedestrian detection

Sermanet et al. “Pedestrian detection with unsupervised multi-stage..” CVPR 2013
CONV NETS: EXAMPLES

- Scene Parsing

Farabet et al. “Learning hierarchical features for scene labeling” PAMI 2013
CONV NETS: EXAMPLES

- Segmentation 3D volumetric images

Ciresan et al. “DNN segment neuronal membranes...” NIPS 2012
Turaga et al. “Maximin learning of image segmentation” NIPS 2009
CONV NETS: EXAMPLES

- Action recognition from videos

Taylor et al. “Convolutional learning of spatio-temporal features” ECCV 2010
CONV NETS: EXAMPLES

- Robotics

Sermanet et al. “Mapping and planning ...with long range perception” IROS 2008
CONV NETS: EXAMPLES

- Denoising

Burger et al. “Can plain NNs compete with BM3D?” CVPR 2012
CONV NETS: EXAMPLES

- Dimensionality reduction / learning embeddings

Hadsell et al. “Dimensionality reduction by learning an invariant mapping” CVPR 2006
CONV NETS: EXAMPLES

- Image classification

Krizhevsky et al. “ImageNet Classification with deep CNNs” NIPS 2012
CONV NETS: EXAMPLES

- Deployed in commercial systems (Google & Baidu, spring 2013)
How To Use ConvNets...(properly)
CHOOSING THE ARCHITECTURE

- Task dependent
- Cross-validation

- [Convolution $\rightarrow$ LCN $\rightarrow$ pooling] $^*$ + fully connected layer

- The more data: the more layers and the more kernels
  - Look at the number of parameters at each layer
  - Look at the number of flops at each layer

- Computational cost

- Be creative :)

Ranzato
HOW TO OPTIMIZE

- SGD (with momentum) usually works very well

- Pick learning rate by running on a subset of the data
  - Bottou “Stochastic Gradient Tricks” Neural Networks 2012
  - Start with large learning rate and divide by 2 until loss does not diverge
  - Decay learning rate by a factor of ~100 or more by the end of training

- Use \[\text{non-linearity}\]

- Initialize parameters so that each feature across layers has similar variance. Avoid units in saturation.
HOW TO IMPROVE GENERALIZATION

- Weight sharing (greatly reduce the number of parameters)
- Data augmentation (e.g., jittering, noise injection, etc.)
- Dropout
  Hinton et al. “Improving Nns by preventing co-adaptation of feature detectors” arxiv 2012
- Weight decay (L2, L1)
- Sparsity in the hidden units
- Multi-task (unsupervised learning)
OTHER THINGS GOOD TO KNOW

- Check gradients numerically by finite differences
- Visualize features (feature maps need to be uncorrelated) and have high variance.

Good training: hidden units are sparse across samples and across features.
OTHER THINGS GOOD TO KNOW

- Check gradients numerically by finite differences
- Visualize features (feature maps need to be uncorrelated) and have high variance.

**Bad training:** many hidden units ignore the input and/or exhibit strong correlations.
OTHER THINGS GOOD TO KNOW

- Check gradients numerically by finite differences
- Visualize features (feature maps need to be uncorrelated) and have high variance.
- Visualize parameters

**GOOD**

**BAD**

too noisy

too correlated

lack structure

**Good training:** learned filters exhibit structure and are uncorrelated.
OTHER THINGS GOOD TO KNOW

- Check gradients numerically by finite differences
- Visualize features (feature maps need to be uncorrelated) and have high variance.
- Visualize parameters
- Measure error on both training and validation set.
- Test on a small subset of the data and check the error $\rightarrow 0$. 
WHAT IF IT DOES NOT WORK?

- Training diverges:
  - Learning rate may be too large → decrease learning rate
  - BPROP is buggy → numerical gradient checking

- Parameters collapse / loss is minimized but accuracy is low
  - Check loss function:
    - Is it appropriate for the task you want to solve?
    - Does it have degenerate solutions?

- Network is underperforming
  - Compute flops and nr. params. → if too small, make net larger
  - Visualize hidden units/params → fix optimization

- Network is too slow
  - Compute flops and nr. params. → GPU,distrib. framework, make net smaller
SUMMARY

- Deep Learning = Learning Hierarchical representations. Leverage compositionality to gain efficiency.
- Unsupervised learning: active research topic.
- Supervised learning: most successful set up today.

Optimization
- Don't we get stuck in local minima? No, they are all the same!
- In large scale applications, local minima are even less of an issue.

Scaling
- GPUs
- Distributed framework (Google)
- Better optimization techniques

Generalization on small datasets (curse of dimensionality):
- Input distortions
- weight decay
- dropout
THANK YOU!

SOFTWARE

Torch7: learning library that supports neural net training

http://www.torch.ch
http://code.cogbits.com/wiki/doku.php  (tutorial with demos by C. Farabet)

Python-based learning library  (U. Montreal)

- http://deeplearning.net/software/theano/  (does automatic differentiation)

C++ code for ConvNets  (Sermanet)

– http://eblearn.sourceforge.net/

Efficient CUDA kernels for ConvNets  (Krizhevsky)

– code.google.com/p/cuda-convnet

More references at  the CVPR 2013 tutorial on deep learning: