Hierarchical Models Of Perception and Reasoning

Yann LeCun
Center for Data Science
Courant Institute of Mathematical Sciences
Center for Neural Science
New York University
Learning Representations: a challenge for Neuroscience, Cognitive Science and AI

- How do we learn representations of the perceptual world?
  - How can a perceptual system build itself by looking at the world?

- Neuroscience: how does the cortex learn perception?
  - Does the cortex “run” a single, general learning algorithm?

- AI/ML: how do we learn features or feature hierarchies?
  - What is the learning algorithm and what is the architecture?

- Deep Learning addresses the problem of learning hierarchical representations with a single algorithm (or a few)
Traditional way: handcrafted features + supervised classifier

Mainstream Approaches to Image and Speech Recognition

Low-Level Features (fixed)
- MFCC
- SIFT
- HoG

Mid-Level Features (unsupervised)
- Kernel
- Mix of Gaussians
- K-means
- Sparse Coding

Classifier (supervised)

supervised
This Basic Model has not evolved much since the 50's

- The first learning machine for perception: the Perceptron
  - Built at Cornell in 1960
- The Perceptron was a linear classifier on top of a simple feature extractor
- The vast majority of practical applications of ML today use glorified linear classifiers
- Designing a feature extractor requires considerable efforts by experts.

\[ y = \text{sign} \left( \sum_{i=1}^{N} W_i F_i(X) + b \right) \]
Models of Categorization in Psychology are equally simple

Most models of perception in psychology/psychophysics are:

- Template/examplar/prototype based
  \[ y = \text{sign} \left( \sum_{i=1}^{N} W_i K(X_i, X) + b \right) \]

- Linear (or generalized linear)
  \[ y = \text{sign} \left( \sum_{i=1}^{N} W_i F_i(X) + b \right) \]

These are simple models that are quite distant from reality

- They do not explain invariant recognition, crowding, or the decrease of “efficiency” with stimulus complexity [Pelli 2006]
The Mammalian Visual Cortex is Hierarchical

- The ventral (recognition) pathway in the visual cortex has multiple stages
- Retina - LGN - V1 - V2 - V4 - PIT - AIT ....
- Lots of intermediate representations

[picture from Simon Thorpe]

[picture from Gallant & Van Essen]
Let's be inspired by nature, but not too much

- It's nice to imitate Nature,
- But we also need to understand
  - How do we know which details are important?
  - Which details are merely the result of evolution, and the constraints of biochemistry?
- For airplanes, we developed aerodynamics and compressible fluid dynamics.
  - We figured that feathers and wing flapping weren't crucial
- What is the equivalent of aerodynamics for understanding intelligence?

L'Avion III de Clément Ader, 1897
(Musée du CNAM, Paris)
His Eole took off from the ground in 1890, 13 years before the Wright Brothers, but you probably never heard of it.
Trainable Feature Hierarchies: End-to-end learning

- A hierarchy of trainable feature transforms

- Each module transforms its input representation into a higher-level one.

How can we make all the modules trainable and get them to learn appropriate representations?
In Many Fields, Feature Learning Has Caused a Revolution (methods used in commercially deployed systems)

- **Speech Recognition I (late 1980s)**
  - Trainable mid-level features with GMM (2-layer non-linear classifier)

- **Handwriting Recognition and OCR (late 1980s to mid 1990s)**
  - Supervised convolutional nets operating on pixels

- **Face & People Detection (early 1990s to mid 2000s)**
  - Supervised convolutional nets operating on pixels (early 1990s)
  - Learned Haar features (through random generation and selection, 2001)

- **Object Recognition I (mid-to-late 2000s: Ponce, Schmid, Yu, YLC....)**
  - Trainable mid-level features (K-means or sparse coding)

- **Speech Recognition II (circa 2011)**
  - Deep neural nets for acoustic modeling

- **Object Recognition II, Scene Parsing (2012, Hinton, YLC,...)**
  - Supervised convolutional nets operating on pixels
Early Hierarchical Feature Models for Vision

[Hubel & Wiesel 1962]:

- **simple cells** detect local features
- **complex cells** "pool" the outputs of simple cells within a retinotopic neighborhood.

Cognitron & Neocognitron [Fukushima 1974-1982]
The Convolutional Net Model
(Multistage Hubel-Wiesel system)

Training is supervised
With stochastic gradient descent

[LeCun et al. 89]
[LeCun et al. 98]
Feature Transform: Normalization → Filter Bank → Non-Linearity → Pooling

- Stacking multiple stages of [Normalization → Filter Bank → Non-Linearity → Pooling].
- **Normalization**: variations on whitening
  - Subtractive: average removal, high pass filtering
  - Divisive: local contrast normalization, variance normalization
- **Filter Bank**: dimension expansion, projection on overcomplete basis
- **Non-Linearity**: sparsification, saturation, lateral inhibition...
  - Rectification, Component-wise shrinkage, tanh, winner-takes-all
- **Pooling**: aggregation over space or feature type, subsampling
  - \( X_i \); \( L_p : \sqrt[p]{X_i^p} \); \( PROB : \frac{1}{b} \log \left( \sum_i e^{bX_i} \right) \)
Feature Transform:
Normalization → Filter Bank → Non-Linearity → Pooling

Filter Bank → Non-Linearity = Non-linear embedding in high dimension
Feature Pooling = contraction, dimensionality reduction, smoothing
Learning the filter banks at every stage
Creating a hierarchy of features
Basic elements are inspired by models of the visual (and auditory) cortex
  Simple Cell + Complex Cell model of [Hubel and Wiesel 1962]
  Many “traditional” feature extraction methods are based on this
  SIFT, GIST, HoG, SURF...

[Fukushima 1974-1982], [LeCun 1988-now],
since the mid 2000: Hinton, Seung, Poggio, Ng,....
Convolutional Network (ConvNet)

- **Non-Linearity:** half-wave rectification, shrinkage function, sigmoid
- **Pooling:** average, L1, L2, max
- **Training:** Supervised (1988-2006), Unsupervised+Supervised (2006-now)
Convolutional Network Architecture

Stage 1:
- Input: high-pass filtered contrast-normalized 83x83 (raw: 91x91)
- Filter Bank + Tanh + Gain
- 64 features 75x75
- 64 filters
- 9x9 kernels
- Abs + Contrast Norm + Pooling + Downsampling
- 64 features 14x14
- 5x5 subsampling
- 10x10 pooling

Stage 2:
- 256 features 6x6
- 4096 filters
- 9x9 kernels
- Abs + Contrast Norm + Pooling + Downsampling
- 256 features 1x1
- 4x4 subsampling
- 6x6 pooling

Parzen Windows Classifier

Classifier
Convolutional Network (vintage 1990)

filters → tanh → average-tanh → filters → tanh → average-tanh → filters → tanh

Curved manifold

Flatter manifold
"Mainstream" object recognition pipeline 2006-2012: somewhat similar to ConvNets

Fixed Features + unsupervised mid-level features + simple classifier

- SIFT + Vector Quantization + Pyramid pooling + SVM
  - [Lazebnik et al. CVPR 2006]
- SIFT + Local Sparse Coding Macrofeatures + Pyramid pooling + SVM
  - [Boureau et al. ICCV 2011]
- SIFT + Fisher Vectors + Deformable Parts Pooling + SVM
  - [Perronin et al. 2012]
Tasks for Which Deep Convolutional Nets are the Best

- Handwriting recognition MNIST (1998), Arabic HWX...
- OCR in the Wild (2011): StreetView House Numbers
- Traffic sign recognition [2011] GTSRB competition
- Pedestrian Detection [2013]: INRIA datasets and others
- Volumetric brain image segmentation [2009] (connectomics)
- Scene Parsing [2012] Stanford backgrounds, SiftFlow, Barcelona datasets
- Object Recognition [2012] ImageNet competition
- Scene parsing from depth images [2013] NYU RGB-D dataset

The list of perceptual tasks for which ConvNets hold the record is growing.
Most of these tasks (but not all) use purely supervised convnets.
Ideas from Neuroscience and Psychophysics

- The whole architecture: simple cells and complex cells
- Local receptive fields
- Self-similar receptive fields over the visual field (convolutions)
- Pooling (complex cells)
- Non-Linearity: Rectified Linear Units (ReLU)
- LGN-like band-pass filtering and contrast normalization in the input
- Divisive contrast normalization (from Heeger, Simoncelli....)
  - Lateral inhibition
- Sparse/Overcomplete representations (Olshausen-Field....)
- Inference of sparse representations with lateral inhibition
- Sub-sampling ratios in the visual cortex
  - between 2 and 3 between V1-V2-V4
- Crowding and visual metamers give cues on the size of the pooling areas
Simple ConvNet Applications with State-of-the-Art Performance

**Traffic Sign Recognition (GTSRB)**
- German Traffic Sign Reco Bench
- 99.2% accuracy

**House Number Recognition (Google)**
- Street View House Numbers
- 94.3% accuracy
Object Recognition [Krizhevski, Sutskever, Hinton 2012]

- ImageNet Large Scale Visual Recognition Challenge
- 1000 categories, 1.5 Million labeled training samples
- Method: large convolutional net
  - 650K neurons, 630M synapses, 60M parameters
  - Trained with backprop on GPU
  - “with all tricks Yann came up with in the last 20 years, plus dropout”
- Error rate: 15% (whenever correct class isn't in top 5)
- Previous state of the art: 25% error
- A REVOLUTION IN COMPUTER VISION
### Object Recognition [Krizhevski, Sutskever, Hinton 2012]

<table>
<thead>
<tr>
<th>Mite</th>
<th>Container Ship</th>
<th>Motor Scooter</th>
<th>Leopard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mite</td>
<td>Container Ship</td>
<td>Motor Scooter</td>
<td>Leopard</td>
</tr>
<tr>
<td>Black Widow</td>
<td>Lifeboat</td>
<td>Go-Kart</td>
<td>Jaguar</td>
</tr>
<tr>
<td>Cockroach</td>
<td>Amphibian</td>
<td>Moped</td>
<td>Cheetah</td>
</tr>
<tr>
<td>Tick</td>
<td>Fireboat</td>
<td>Bumper Car</td>
<td>Snow Leopard</td>
</tr>
<tr>
<td>Starfish</td>
<td>Drilling Platform</td>
<td>Golfcart</td>
<td>Egyptian Cat</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Grille</th>
<th>Mushroom</th>
<th>Cherry</th>
<th>Madagascar Cat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convertible</td>
<td>Agaric</td>
<td>Dalmatian</td>
<td>Squirrel Monkey</td>
</tr>
<tr>
<td>Grille</td>
<td>Mushroom</td>
<td>Grape</td>
<td>Spider Monkey</td>
</tr>
<tr>
<td>Pickup</td>
<td>Jelly Fungus</td>
<td>Elderberry</td>
<td>Titi</td>
</tr>
<tr>
<td>Beach Wagon</td>
<td>Gill Fungus</td>
<td>Ffordshire Bullterrier</td>
<td>Indri</td>
</tr>
<tr>
<td>Fire Engine</td>
<td>Dead-Man's-Fingers</td>
<td>Currant</td>
<td>Howler Monkey</td>
</tr>
</tbody>
</table>
http://horatio.cs.nyu.edu

Object Recognition on-line demo [Zeiler & Fergus]

Image Classifier Demo
Upload your images to have them classified by a machine! Upload multiple images using the button below or dropping them on this page. The predicted objects will be refreshed automatically. Images are resized such that the smallest dimension becomes 256, then the center 256x256 crop is used. More about the demo can be found here.

Upload Images  Remove All  Show help tips

I agree to the Terms of Use

Demo Notes
- If your images have objects that are not in the 1,000 categories of ImageNet, the model will not know about them.
- Other objects can be added from all 20,000+ ImageNet categories (it may be slow to load the autocomplete results...just wait a little).
- The maximum file size for uploads in this demo is 10 MB.
- Only image files (JPEG, JPG, GIF, PNG) are allowed in this demo.
- You can drag & drop files from your desktop on this webpage with Google Chrome, Mozilla Firefox and Apple Safari.
- Some mobile browsers are known to work, others will not. Try updating your browser or contact us with the problem.
- All images for your current IP and browsing session are shown above and not shown to others.
- This demo is powered by research out of New York University. Click here to find out more
- If you encounter problems, please contact zeiler@cs.nyu.edu

Demo created by: Matthew Zeiler
Labeling every pixel with the object it belongs to

- Would help identify obstacles, targets, landing sites, dangerous areas
- Would help line up depth map with edge maps

[Farabet et al. ICML 2012]
Scene Parsing/Labeling: ConvNet Architecture

- Each output sees a large input context:
  - 46x46 window at full rez; 92x92 at ½ rez; 184x184 at ¼ rez
  - [7x7conv]->[2x2pool]->[7x7conv]->[2x2pool]->[7x7conv]->
  - Trained supervised on fully-labeled images

- Laplacian Pyramid
- Level 1 Features
- Level 2 Features
- Upsampled Level 2 Features
- Categories
Scene Parsing/Labeling: Performance

### Stanford Background Dataset [Gould 1009]: 8 categories

<table>
<thead>
<tr>
<th>Method</th>
<th>Pixel Acc.</th>
<th>Class Acc.</th>
<th>CT (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gould et al. 2009 [14]</td>
<td>76.4%</td>
<td>-</td>
<td>10 to 600s</td>
</tr>
<tr>
<td>Munoz et al. 2010 [32]</td>
<td>76.9%</td>
<td>66.2%</td>
<td>12s</td>
</tr>
<tr>
<td>Tighe et al. 2010 [46]</td>
<td>77.5%</td>
<td>-</td>
<td>10 to 300s</td>
</tr>
<tr>
<td>Socher et al. 2011 [45]</td>
<td>78.1%</td>
<td>-</td>
<td>?</td>
</tr>
<tr>
<td>Kumar et al. 2010 [22]</td>
<td>79.4%</td>
<td>-</td>
<td>&lt; 600s</td>
</tr>
<tr>
<td>Lempitzky et al. 2011 [28]</td>
<td>81.9%</td>
<td>72.4%</td>
<td>&gt; 60s</td>
</tr>
<tr>
<td>singlescale convnet</td>
<td>66.0 %</td>
<td>56.5 %</td>
<td>0.35s</td>
</tr>
<tr>
<td>multiscale convnet</td>
<td>78.8 %</td>
<td>72.4%</td>
<td>0.6s</td>
</tr>
<tr>
<td>multiscale net + superpixels</td>
<td>80.4%</td>
<td>74.56%</td>
<td>0.7s</td>
</tr>
<tr>
<td>multiscale net + gPb + cover</td>
<td>80.4%</td>
<td>75.24%</td>
<td>61s</td>
</tr>
<tr>
<td>multiscale net + CRF on gPb</td>
<td>81.4%</td>
<td>76.0%</td>
<td>60.5s</td>
</tr>
</tbody>
</table>

[Farabet et al. IEEE T. PAMI 2012]
## Scene Parsing/Labeling: Performance

<table>
<thead>
<tr>
<th>Method</th>
<th>Pixel Acc.</th>
<th>Class Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liu et al. 2009 [31]</td>
<td>74.75%</td>
<td>-</td>
</tr>
<tr>
<td>Tighe et al. 2010 [44]</td>
<td>76.9%</td>
<td>29.4%</td>
</tr>
<tr>
<td>raw multiscale net¹</td>
<td>67.9%</td>
<td>45.9%</td>
</tr>
<tr>
<td>multiscale net + superpixels¹</td>
<td>71.9%</td>
<td>50.8%</td>
</tr>
<tr>
<td>multiscale net + cover¹</td>
<td>72.3%</td>
<td>50.8%</td>
</tr>
<tr>
<td>multiscale net + cover²</td>
<td>78.5%</td>
<td>29.6%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Pixel Acc.</th>
<th>Class Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tighe et al. 2010 [44]</td>
<td>66.9%</td>
<td>7.6%</td>
</tr>
<tr>
<td>raw multiscale net¹</td>
<td>37.8%</td>
<td>12.1%</td>
</tr>
<tr>
<td>multiscale net + superpixels¹</td>
<td>44.1%</td>
<td>12.4%</td>
</tr>
<tr>
<td>multiscale net + cover¹</td>
<td>46.4%</td>
<td>12.5%</td>
</tr>
<tr>
<td>multiscale net + cover²</td>
<td>67.8%</td>
<td>9.5%</td>
</tr>
</tbody>
</table>

- **SIFT Flow Dataset**
- [Liu 2009]:
- 33 categories
- **Barcelona dataset**
- [Tighe 2010]:
- 170 categories.

[Farabet et al. IEEE T. PAMI 2012]
Scene Parsing/Labeling: SIFT Flow dataset (33 categories)

Samples from the SIFT-Flow dataset (Liu)

[Farabet et al. ICML 2012]
Scene Parsing/Labeling: SIFT Flow dataset (33 categories)

[Farabet et al. ICML 2012]
Scene Parsing/Labeling

[Farabet et al. ICML 2012]
Scene Parsing/Labeling

[Farabet et al. ICML 2012]
Scene Parsing/Labeling

[Farabet et al. 2012]
Scene Parsing/Labeling

[Farabet et al. 2012]
Scene Parsing/Labeling

- No post-processing
- Frame-by-frame
- ConvNet runs at 50ms/frame on Virtex-6 FPGA hardware
  - But communicating the features over ethernet limits system perf.
Scene Parsing/Labeling: Temporal Consistency

- Majority Vote on Spatio-Temporal Super-Pixels
- Reset every second
Scene Parsing/Labeling on RGB+Depth Images

Ground truths

Our results

- wall
- books
- chair
- furniture
- sofa
- object
- TV
- bed
- ceiling
- floor
- pict./deco
- table
- window
- uknw
Scene Parsing/Labeling on RGB+Depth Images

Ground truths

Our results
Unsupervised Learning: Disentangling the independent, explanatory factors of variation
Learning Representations of Data:
- Discovering the independent explanatory factors of the data

The Manifold Hypothesis:
- Natural data lives in a low-dimensional (non-linear) manifold
- Because variables in natural data are mutually dependent
Example: all face images of a person
- 1000x1000 pixels = 1,000,000 dimensions
- But the face has 3 coordinates and 3 Euler angles
- And humans have less than about 50 muscles in the face
- Hence the manifold of face images for a person has <56 dimensions

The perfect representations of a face image:
- Its coordinates on the face manifold
- Its coordinates away from the manifold

We do not have good and general methods to learn from data a function that turns an image into this kind of representation
Regularized Encoder-Decoder Model (auto-Encoder) for Unsupervised Feature Learning

- **Encoder**: computes feature vector Z from input X
- **Decoder**: reconstructs input X from feature vector Z
- **Feature vector**: high dimensional and regularized (e.g. sparse)
- **Factor graph with energy function E(X,Z) with 3 terms:**
  - Linear decoding function and reconstruction error
  - Non-Linear encoding function and prediction error term
  - Pooling function and regularization term (e.g. sparsity)

\[
E(Y,Z) = \| Y - W_d Z \|^2 + \| Z - g_e(W_e, Y) \|^2 + \sum_j \sqrt{\sum_{k \in P_j} Z_k^2}
\]
Predictive Sparse Decomposition (PSD): Training

- Training on natural images patches.
  - 12X12
  - 256 basis functions
Learned Features on natural patches: V1-like receptive fields
Better Idea: Give the “right” structure to the encoder

[Gregor & LeCun, ICML 2010], [Bronstein et al. ICML 2012]

ISTA/FISTA: iterative algorithm that converges to optimal sparse code

\[ Z(t + 1) = \text{Shrinkage}_{\lambda/L} \left[ Z(t) - \frac{1}{L} W_d^T (W_d Z(t) - Y) \right] \]

\[ Z(t + 1) = \text{Shrinkage}_{\lambda/L} \left[ W_e^T Y + S Z(t) \right] ; \quad W_e = \frac{1}{L} W_d ; \quad S = I - \frac{1}{L} W_d^T W_d \]
LISTA: Train We and S matrices to give a good approximation quickly

Think of the FISTA flow graph as a recurrent neural net where We and S are trainable parameters

Time-Unfold the flow graph for K iterations
Learn the We and S matrices with “backprop-through-time”
Get the best approximate solution within K iterations
Discriminative Recurrent Sparse Auto-Encoder

Architecture

- Rectified linear units
- Classification loss: cross-entropy
- Reconstruction loss: squared error
- Sparsity penalty: L1 norm of last hidden layer
- Rows of $W_d$ and columns of $W_e$ constrained in unit sphere

[Rolfe & LeCun 2013]
DrSAE Discovers manifold structure of handwritten digits

Image = prototype + sparse sum of “parts” (to move around the manifold)
Replace the dot products with dictionary element by convolutions.

- Input Y is a full image
- Each code component Z_k is a feature map (an image)
- Each dictionary element is a convolution kernel

**Regular sparse coding**

\[
E(Y, Z) = \|Y - \sum_k W_k Z_k\|^2 + \alpha \sum_k |Z_k|
\]

**Convolutional S.C.**

\[
E(Y, Z) = \|Y - \sum_k W_k * Z_k\|^2 + \alpha \sum_k |Z_k|
\]

\[
Y = \sum_k W_k * Z_k
\]

“deconvolutional networks” [Zeiler, Taylor, Fergus CVPR 2010]
Convolutional Formulation

- Extend sparse coding from **PATCH** to **IMAGE**

\[
\mathcal{L}(x, z, D) = \frac{1}{2} ||x - \sum_{k=1}^{K} D_k * z_k||^2_2 + \sum_{k=1}^{K} ||z_k - f(W^k * x)||^2_2 + |z|_1
\]

- **PATCH** based learning
- **CONVOLUTIONAL** learning
Filters and Basis Functions obtained with 1, 2, 4, 8, 16, 32, and 64 filters.
Using PSD to Train a Hierarchy of Features

Phase 1: train first layer using PSD

\[ \|Y^i - \tilde{Y}\|^2 \]

\[ W_d Z \]

\[ W_e, Y^i \]

\[ \|Z - \tilde{Z}\|^2 \]

\[ g_e(W_e, Y^i) \]

\[ \lambda \sum \]

\[ |z_j| \]
Using PSD to Train a Hierarchy of Features

- Phase 1: train first layer using PSD
- Phase 2: use encoder + absolute value as feature extractor
Using PSD to Train a Hierarchy of Features

Phase 1: train first layer using PSD
Phase 2: use encoder + absolute value as feature extractor
Phase 3: train the second layer using PSD
Using PSD to Train a Hierarchy of Features

- Phase 1: train first layer using PSD
- Phase 2: use encoder + absolute value as feature extractor
- Phase 3: train the second layer using PSD
- Phase 4: use encoder + absolute value as 2\textsuperscript{nd} feature extractor

\[
g_e(W_e, Y^i) \rightarrow |z_j| \rightarrow |z_j| \rightarrow \text{FEATURES}
\]
Using PSD to Train a Hierarchy of Features

- Phase 1: train first layer using PSD
- Phase 2: use encoder + absolute value as feature extractor
- Phase 3: train the second layer using PSD
- Phase 4: use encoder + absolute value as 2\textsuperscript{nd} feature extractor
- Phase 5: train a supervised classifier on top
- Phase 6 (optional): train the entire system with supervised back-propagation
Pedestrian Detection, Face Detection

[Osadchy, Miller LeCun JMLR 2007], [Kavukcuoglu et al. NIPS 2010]
Pedestrian Detection: INRIA Dataset. Miss rate vs false positives

ConvNet
Color+Skip
Supervised

ConvNet
Color+Skip
Unsup+Sup

ConvNet
B&W
Unsup+Sup

ConvNet
B&W
Supervised

[Kavukcuoglu et al. NIPS 2010] [Sermanet et al. ArXiv 2012]
Unsupervised pre-training with convolutional PSD

- 128 stage-1 filters on Y channel.
- Unsupervised training with convolutional predictive sparse decomposition
Unsupervised pre-training with convolutional PSD

Stage 2 filters.

Unsupervised training with convolutional predictive sparse decomposition
Unsupervised Learning: Invariant Features
Unsupervised PSD ignores the spatial pooling step.
Could we devise a similar method that learns the pooling layer as well?
Idea [Hyvarinen & Hoyer 2001]: group sparsity on pools of features
- Minimum number of pools must be non-zero
- Number of features that are on within a pool doesn't matter
- Pools tend to regroup similar features

\[
E(Y, Z) = \| Y - W_d Z \|^2 + \| Z - g_e(W_e, Y) \|^2 + \sum_j \sum_{k \in P_j} \sqrt{Z_k^2}
\]
Groups are local in a 2D Topographic Map

- The filters arrange themselves spontaneously so that similar filters enter the same pool.
- The pooling units can be seen as complex cells.
- Outputs of pooling units are invariant to local transformations of the input.
  - For some it's translations, for others rotations, or other transformations.
Image-level training, local filters but no weight sharing

Training on 115x115 images. Kernels are 15x15 (not shared across space!)
119x119 Image Input
100x100 Code
20x20 Receptive field size
sigma=5

Michael C. Crair, et. al. The Journal of Neurophysiology
Vol. 77 No. 6 June 1997, pp. 3381-3385 (Cat)

K Obermayer and GG Blasdel, Journal of Neuroscience, Vol 13, 4114-4129 (Monkey)
Image-level training, local filters but no weight sharing

- Color indicates orientation (by fitting Gabors)
Replace the L1 sparsity term by a lateral inhibition matrix

Easy way to impose some structure on the sparsity

\[
\min_{W,Z} \sum_{x \in X} ||Wz - x||^2 + |z|^T S |z|
\]
Invariant Features via Lateral Inhibition: Structured Sparsity

- Each edge in the tree indicates a zero in the $S$ matrix (no mutual inhibition)
- $S_{ij}$ is larger if two neurons are far away in the tree
Non-zero values in $S$ form a ring in a 2D topology

- Input patches are high-pass filtered
Object is *cross-product* of object type and instantiation parameters

- Mapping units [Hinton 1981], capsules [Hinton 2011]

![Diagram of object types and sizes]

**Object type**

- [Karol Gregor et al.]

**Object size**

- small
- medium
- large
What-Where Auto-Encoder Architecture

Decoder

\[ S^t \]

\[ S^{t-1} \]

\[ S^{t-2} \]

Predicted input

Inferred code

Predicted code

Encoder

\[ S^t \]

\[ S^{t-1} \]

\[ S^{t-2} \]

Input
Low-Level Filters Connected to Each Complex Cell

C1  (where)

C2  (what)
Generating from the Network
Challenges
The Graph of Deep Learning ↔ Sparse Modeling ↔ Neuroscience

- Compr. Sensing [Candès-Tao 04]
- L2-L1 optim [Nesterov, Nemirovski, Daubechies, Osher,...]
- Restricted Boltzmann Machine [Hinton 05]
- Speech Recognition [Goog, IBM, MSFT 12]
- Basis/Matching Pursuit [Mallat 93; Donoho 94]
- Sparse Modeling [Olshausen-Field 97]
- Sparse Auto-Encoder [LeCun 06; Ng 07]
- Backprop [many 85]
- Convolutional Net [LeCun 89]
- Stochastic Optimization [Nesterov, Bottou Nemirovski,...]
- Object Reco [LeCun 10]
- Object Recog [Hinton 12]
- Scene Labeling [LeCun 12]
- Connectomics [Seung 10]
- Visual Metamers [Simoncelli 12]
- Normalization [Simoncelli 94]
- Scattering Transform [Mallat 10]
- Neocognitron [Fukushima 82]
Marrying feed-forward convolutional nets with generative “deconvolutional nets”
- Deconvolutional networks
  - [Zeiler-Graham-Fergus ICCV 2011]

Feed-forward/Feedback networks allow reconstruction, multimodal prediction, restoration, etc...
- Deep Boltzmann machines can do this, but there are scalability issues with training
Deep Learning systems can be assembled into factor graphs

- Energy function is a sum of factors
- Factors can embed whole deep learning systems
- X: observed variables (inputs)
- Z: never observed (latent variables)
- Y: observed on training set (output variables)

Inference is energy minimization (MAP) or free energy minimization (marginalization) over Z and Y given an X

\[ E(X,Y,Z) \]

Energy Model (factor graph)

\[ X \] (observed)
\[ Z \] (unobserved)
\[ Y \] (observed on training set)
Deep Learning systems can be assembled into factor graphs

- Energy function is a sum of factors
- Factors can embed whole deep learning systems
- X: observed variables (inputs)
- Z: never observed (latent variables)
- Y: observed on training set (output variables)

Inference is energy minimization (MAP) or free energy minimization (marginalization) over Z and Y given an X

- $F(X,Y) = \text{MIN}_z E(X,Y,Z)$
- $F(X,Y) = -\log \sum_z \exp[-E(X,Y,Z)]$
Integrating deep learning and structured prediction is a very old idea

- In fact, it predates structured prediction

Globally-trained convolutional-net + graphical models

- trained discriminatively at the word level
- Loss identical to CRF and structured perceptron
- Compositional movable parts model

A system like this was reading 10 to 20% of all the checks in the US around 1998
Deep Learning systems can be assembled into factor graphs

- Energy function is a sum of factors
- Factors can embed whole deep learning systems
- X: observed variables (inputs)
- Z: never observed (latent variables)
- Y: observed on training set (output variables)

Inference is energy minimization (MAP) or free energy minimization (marginalization) over Z and Y given an X

\[ F(X,Y) = \text{MIN}_z E(X,Y,Z) \]

\[ F(X,Y) = -\log \text{SUM}_z \exp[-E(X,Y,Z)] \]
Future Challenges

- Integrated feed-forward and feedback
  - Deep Boltzmann machine do this, but there are issues of scalability.

- Integrating supervised and unsupervised learning in a single algorithm
  - Again, deep Boltzmann machines do this, but....

- Integrating deep learning and structured prediction ("reasoning")
  - This has been around since the 1990's but needs to be revived

- Learning representations for complex reasoning
  - "recursive" networks that operate on vector space representations of knowledge [Pollack 90's] [Bottou 2010] [Socher, Manning, Ng 2011]

- Representation learning in natural language processing
  - [Y. Bengio 01], [Collobert Weston 10], [Mnih Hinton 11] [Socher 12]

- Better theoretical understanding of deep learning and convolutional nets
  - e.g. Stephane Mallat's "scattering transform", work on the sparse representations from the applied math community....