Unsupervised Neural Nets
(and ICA)

Lyle Ungar
(with contributions from Quoc Le, Socher & Manning)
Semi-Supervised Learning

- Hypothesis: $P(c|x)$ can be more accurately computed using shared structure with $P(x)$

from Socher and Manning
• Hypothesis: $P(c|x)$ can be more accurately computed using shared structure with $P(x)$

from Socher and Manning
Unsupervised Neural Nets

◆ Autoencoders
  - Take same image as input and output
    - often adding noise to the input
  - Learn weights to minimize the reconstruction error
  - Avoid perfect fitting
    - Pass through a “bottleneck”
    - Impose sparsity
      - Dropout
    - Add noise to the input

◆ Generalize PCA or ICA

http://ufldl.stanford.edu/wiki/index.php/Autoencoders_and_Sparsity
Given observations $X$, find $W$ such that components $s_j$ of $S = XW$ are “as independent of each other as possible”

- E.g. have maximum KL-divergence or low mutual information
- Alternatively, find directions in $X$ that are most skewed
  - farthest from Gaussian
- Usually mean center and “whiten” the data (make unit covariance) first
  - whiten: $(X^TX)^{-1/2} X$

Very similar to PCA

- But the loss function is not quadratic
- So optimization cannot be done by SVD
Independent Components Analysis (ICA)

- Given observations \(X\), find \(W\) and \(S\) such that components \(s_j\) of \(S = XW\) are “as independent of each other as possible”
  - \(S_k = “sources”\) should be independent

- Reconstruct \(X \sim (XW)W^+ = SW^+\)
  - \(S\) like principle components
  - \(W^+\) like loadings
  - \(x \sim \sum_j s_j w_j^+\)

- **Auto-encoder** – nonlinear generalization that “encodes” \(X\) as \(S\) and then “decodes” it
Reconstruction ICA (RICA)

- **Reconstruction ICA**: find W to minimize
  - Reconstruction error
    - $||X - SW^+||_2 = ||X - (XW)W^+||_2$
  And minimize
  - Mutual information between sources $S = XW$

$$I(s_1, s_2 \ldots s_k) = \sum_{i=1}^{k} H(s_i) - H(s)$$

$$H(y) = -\int p(y) \log p(y) \, dy$$

Difference between the entropy of each “source” $s_i$ and the entropy of all of them together

Note: this is a bit more complex than it looks, as we have real numbers, not distributions
Mutual information

Using the concept of differential entropy, we define the mutual information $I$ between $m$ (scalar) random variables, $y_i, i = 1...m$ as follows

$$I(y_1, y_2, ..., y_m) = \sum_{i=1}^{m} H(y_i) - H(y). \quad (27)$$

Mutual information is a natural measure of the dependence between random variables. In fact, it is equivalent to the well-known Kullback-Leibler divergence between the joint density $f(y)$ and the product of its marginal densities; a very natural measure for independence. It is always non-negative, and zero if and only if the variables are statistically independent. Thus, mutual information takes into account the whole dependence structure of the variables, and not only the covariance, like PCA and related methods.

Mutual information can be interpreted by using the interpretation of entropy as code length. The terms $H(y_i)$ give the lengths of codes for the $y_i$ when these are coded separately, and $H(y)$ gives the code length when $y$ is coded as a random vector, i.e. all the components are coded in the same code. Mutual information thus shows what code length reduction is obtained by coding the whole vector instead of the separate components. In general, better codes can be obtained by coding the whole vector. However, if the $y_i$ are independent, they give no information on each other, and one could just as well code the variables separately without increasing code length.

$$I(y_1, y_2, \ldots, y_m) = KL(p(y_1, y_2, \ldots, y_m) || p(y_1)p(y_2) \ldots p(y_m))$$
Unsupervised Neural Nets

- **Auto-encoders**
  - Take same image as input and output
    - often adding noise to the input
  - Learn weights to minimize the reconstruction error
  - This can be done repeatedly (reconstructing features)
  - Use for semi-supervised learning

http://theanalyticsstore.com/deep-learning/
PCA = Linear Manifold = Linear Auto-Encoder

input $x$, 0-mean
features = code = $h(x) = Wx$
reconstruction ($x$) = $W^T h(x) = W^T W x$
$W$ = principal eigen-basis of Cov($X$)

Linear manifold

code = $h(x)$
reconstruction ($x$)

code = 0
reconstruction error vector

LSA example:
$x$ = (normalized) distribution of co-occurrence frequencies

from Socher and Manning
The Manifold Learning Hypothesis

- Examples concentrate near a lower dimensional “manifold” (region of high density where small changes are only allowed in certain directions).
Auto-Encoders are like nonlinear PCA

Minimizing reconstruction error forces latent representation of “similar inputs” to stay on manifold

from Socher and Manning
Stacking for deep learning

reconstruction of input

features

input

from Socher and Manning
Stacking for deep learning

Now learn to reconstruct the features (using more abstract ones)

from Socher and Manning
Stacking for deep learning

- Recurse – many layers deep
- Can be used to initialize supervised learning

Even more abstract features

More abstract features

features

input

Output \( f(X) \) six \( \Rightarrow \) Target \( Y \)

two!

from Socher and Manning
Tera-scale deep learning

Quoc V. Le
Stanford University and Google

Now at Google
Joint work with

Kai Chen  Greg Corrado  Jeff Dean  Matthieu Devin

Rajat Monga  Andrew Ng  Marc’ Aurelio Ranzato  Paul Tucker  Ke Yang

Additional Thanks: Samy Bengio, Zhenghao Chen, Tom Dean, Pangwei Koh, Mark Mao, Jiquan Ngiam, Patrick Nguyen, Andrew Saxe, Mark Segal, Jon Shlens, Vincent Vanhouke, Xiaoyun Wu, Peng Xe, Serena Yeung, Will Zou
TICA:

$$\min_w \sum_j \sum_i h_j(W; x^{(i)})$$

$$s.t. \quad WW^T = I$$

Reconstruction ICA:

$$\min_w \frac{\lambda}{m} \sum_{i=1}^m \left\| W^T W x^{(i)} - x^{(i)} \right\|_2^2 + \sum_j \sum_i h_j(W; x^{(i)})$$

**Lemma 3.1** When the input data $\{x^{(i)}\}_{i=1}^m$ is whitened, the reconstruction cost $\frac{\lambda}{m} \sum_{i=1}^m \left\| W^T W x^{(i)} - x^{(i)} \right\|_2^2$ is equivalent to the orthonormality cost $\lambda \left\| W^T W - I \right\|_F^2$.

**Lemma 3.2** The column orthonormality cost $\lambda \left\| W^T W - I_n \right\|_F^2$ is equivalent to the row orthonormality cost $\lambda \left\| W W^T - I_k \right\|_F^2$ up to an additive constant.

---

**Equivalence between Sparse Coding, Autoencoders, RBMs and ICA**

**Build deep architecture by treating the output of one layer as input to another layer**

Visualization of features learned

<table>
<thead>
<tr>
<th>Most are local features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
Challenges with 1000s of machines
Asynchronous Parallel SGDs

Le, et al., *Building high-level features using large-scale unsupervised learning*. ICML 2012
Local receptive field networks

Le, et al., *Tiled Convolutional Neural Networks*. NIPS 2010
10 million 200x200 images
1 billion parameters
Dataset: 10 million 200x200 unlabeled images from YouTube/Web

Train on 2000 machines (16000 cores) for 1 week

1.15 billion parameters
- 100x larger than previously reported
- Small compared to visual cortex

Le, et al., *Building high-level features using large-scale unsupervised learning*. ICML 2012
The face neuron

Top stimuli from the test set

Optimal stimulus by numerical optimization

Le, et al., *Building high-level features using large-scale unsupervised learning*. ICML 2012
The cat neuron

Top stimuli from the test set

Optimal stimulus by numerical optimization

Le, et al., *Building high-level features using large-scale unsupervised learning*. ICML 2012
What you should know

- **Supervised neural nets**
  - Generalize *logistic regression*
  - Often have built-in structure
    - local receptive fields, max pooling
  - Usually solved by minibatch stochastic gradient descent
    - With chain rule ("backpropagation"),

- **Unsupervised neural nets**
  - Generalize PCA or ICA
  - Generally learn an “overcomplete basis”
  - Often trained recursively as nonlinear autoencoders
  - Used in semi-supervised learning
    - Or to initialize supervised deep nets