DBN's: Introduction

Boltzmann Machines:

\[ P(v, h) = \frac{1}{Z} e^{-E(v, h)} \]

motivated by modeling the brain, but just an MRF.

\[ E(v, h) = -\sum_{i} v_i w_{ij} h_j - \sum_{i} c_i v_i - \sum_{j} h_i w_{ij} h_j \]

Example: \( \{ v^M: \mu=1, N \} \)

Task: Learn the weights of the model \( \{ w_{ij}, c_i, w_{ij}^h \} \).

Can be learnt, in principle, by the EM algorithm or stochastic variant (e.g. data augmentation)

Difficult because it is hard to do inference on this type of model in general.

Why learn this model? It learns a generative model for data. It learns “receptive” fields
- e.g. the \( w_{ij} \), how a hidden unit “\( h_i \) is activated
  by the inputs.

To build a DBN, you first consider a simplification “Restricted Boltzmann Machine” RBM.

The connection terms \( w_{ij} \) between the hidden units are removed. \( \Rightarrow \) replace \( -\sum_{i} h_i w_{ij} h_j \) by \(-\sum_{i} hi \).

This has the property that \( P(h|v) \) and \( P(v|h) \) have simple forms:

\[ P(h|v) = \prod_{i} P(h_i|v) = e^{-h_i(\sum_{j} w_{ij} v_j + b_i)} \]

similarly \( P(v|h) = \prod_{i} P(v_i|h) = e^{-v_i(\sum_{j} w_{ij} h_j + b_i)} \)

Note Title
Key point RBM's:
- Easy to sample from \( P(v|h) \) i.e. sample from \( P(v|h) \) unbiased
- Easy to sample from \( P(h|v) \)

This enables us to do inference (stochastically) and makes it practical to learn the parameter \( \langle w \rangle \) of an RBM.

But RBM's are too simple \( \Rightarrow \) so add another RBM

\[ \begin{align*}
1^{\text{st}} \text{RBM} & \quad \text{1} \quad \text{X} \quad \text{w} \quad \text{2} \quad \text{2nd} \quad \text{RBM} & \quad \text{1} \quad \text{X} \quad \text{w} \quad \text{2}
\end{align*} \]

use the hidden states \( (h_1 \ldots h_m) \) of the 1\text{st} RBM as input to the 2\text{nd} RBM.

then add another RBM and so on

\[ \begin{align*}
& \quad \text{1} \quad \text{X} \quad \text{w} \quad \text{2} \quad \text{3rd} \quad \text{RBM} & \quad \text{1} \quad \text{X} \quad \text{w} \\
& \quad \text{1} \quad \text{X} \quad \text{w} \quad \text{2} \quad \text{n}^{\text{th}} \text{RBM}
\end{align*} \]

Then the model layer by layer.

Intuition: the receptive fields

\[ \begin{align*}
(\hat{h}_1) \quad \text{capture image features like a dictionary in the previous two lectures.}
\end{align*} \]

\rightarrow \text{then the receptive fields } (w_2, w_3, \ldots) \quad \text{of the}

\[ \begin{align*}
\text{higher levels learn better dictionaries.}
\end{align*} \]

The hidden states at the top levels can be used as inputs to a classifier level.

This can be trained in supervised mode – i.e. the \( \langle h_i \rangle \) are \( \langle u_i \rangle \) are provided by the training data.

Effort for learning how to recognize digits – Hinton et al.

For other variants see Le Cun, Ng, Bengio.

Hinton (Talk: 4 March (Thur) 4:15 p.m. CS)
Active Appearance Models (AAMs)

Object variations in geometry, variations in appearance.

First, suppose objects are aligned (i.e., remove spatial variation).

Perform PCA (principal component analysis).

\[ \mathbf{J} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{I}(x) \]

\[ \mathbf{K}(x,y) = \frac{1}{n} \sum_{i=1}^{n} \left( \mathbf{I}(x) - \mathbf{J} \right) \cdot \left( \mathbf{I}(y) - \mathbf{J} \right)^{T} \]

Solve for eigenvalues & eigenvectors.

\[ \sum_{i=1}^{m} \lambda_{i} \mathbf{v}_{i} \]

Represent object appearance.

\[ \mathbf{I}(x) = \mathbf{J} + \sum_{i=1}^{m} \alpha_{i} \mathbf{v}_{i} \]

Efficient representation if we can approximate \( \mathbf{I}(x) \) using a small no. basis functions, i.e., small.

Square Error

\[ \sum_{i=1}^{n} \sum_{i=1}^{m} \left( \mathbf{I}(x) - \mathbf{J} - \sum_{i=1}^{m} \alpha_{i} \mathbf{v}_{i} \right)^{2} \]

For faces

Intuition: PCA assumes that the data lies in a linear space. This may be true (approximately) for faces, but certainly is not true for text.

Example: data \( \mathbf{I}(x) = \mathbf{B} \mathbf{S}(x \mathbf{v}_{i}) + \mathbf{z}_{i} \mathbf{v}_{i} \mathbf{v}_{i}^{T} \)

Suppose the data is \( \mathbf{I} \) or \( \mathbf{I}^{T} \mathbf{B} \) or \( \mathbf{B}^{T} \mathbf{B} \).

Then PCA gives \( (N-1) \) principal components, all with the same small eigenvalue.
Second: fix appearance, model geometry

\[ I(x) \rightarrow I(s(x)) \]
where \( s(x) \) is a spatial warp

Also represent the spatial warp by PCA also.

i.e. take training data \( \{ s^\mu(x) : \mu \leq N \} \)
perform PCA on \( \{ s^\mu \} \)
express:
\[ s(x) = \sum_{b=1}^{q} \beta_b e_b(x) + \epsilon(x) \]

where \( q \) is no. of coefficients.
\( e_b(x) \) are the eigenvectors.

How to get the training data?
- Typically hand labeled.
- Can label a set of sparse points and interpolate to get dense estimate of \( s(x) \) hand labeled correspondence.
- Interpolate by spatial smoothers.

In principle, can learn the spatial and basis function by EM.

Generator Model

\[ P(I \mid \{ e, \beta \}, (\alpha, \beta)) = \frac{1}{Z} \exp \left( -E[I; (e, \beta)] \right) \]

\[ E[I; (e, \beta)] = \sum_x (I(x) - \sum \alpha \beta_a (\sum \beta_b e_b(x)))^2 \]

set of images \( \{ I^\mu : \mu \leq \Lambda \} \)
parameters \( \{ \alpha^\mu, \beta^\mu \} \) - different coefficients for each image.

\[ P(<I^\mu \mid (e, \beta), \lambda^\mu, \beta^\mu>) \]
\[ = \prod_{\mu} P(I^\mu \mid (e, \beta), \lambda^\mu, \beta^\mu) \]

EM

\[ M \text{-step estimate } <e, \beta> \]
\[ E \text{-step estimate posterior } q(\lambda^\mu, \beta^\mu) \]

Problem \( \Rightarrow \) many local minima \( \rightarrow \)
Recent work: e.g. I. Kokkinos.

Start with objects only approximately located.
- e.g. box round object
  - Background clutter

To simplify the problem and make it tractable:
- filter the image to detect edges and other features like ridges - this removes the appearance variation so need to estimate the B's and s's

Simply need to estimate the spatial warps and their coefficients:
\[ \hat{I}(x) = T \left( \sum_{b=1}^{m} \beta_b \mathbf{s}_b(x) \right) + \varepsilon(x) \]

Task: estimate T(·) the edge/ridge image of the object.

Still require EM, E-step to estimate T(s_b)
M-step to estimate \( \beta_b \)

EM does converge (Kokkinos & Zisserman) with reasonable initial conditions.

Complication: what is m? How many basis functions?

Strategy (KZ) -> greedy search.
Assume m=1, learn \( s_1(x) \)
Then set m=2, learn \( s_2(x) \)...

Note: there were limitations using PCA to represent the appearance (i.e. approx. linear assumption).

What are the limitations of PCA to represent spatial warps?

Roughly true for faces, torso/body of a cow.
But bad for representing the spatial variability of the legs of a cow.

Better to treat the legs as separate path connected to the torso/body.