Johns Hopkins Engineering

Applied Machine Learning for Mechanical Engineers

Unsupervised Machine Learning Techniques, Part 1, B

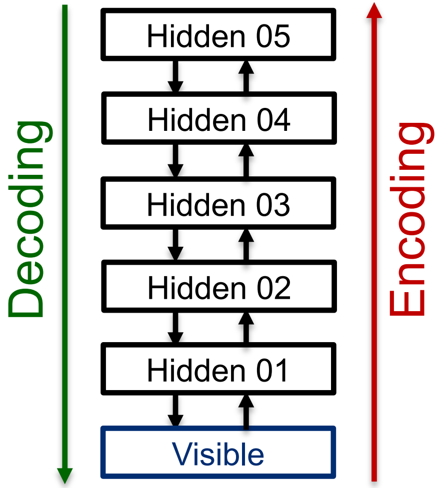
JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Deep Belief Networks and Auto-Encoders

- By the end of this lecture you will be able to:
  - Describe Deep Belief Networks and Auto-Encoders
  - Describe Restricted Boltzmann Machine (RBM)
  - Describe Deep Boltzmann Machine (DBM)

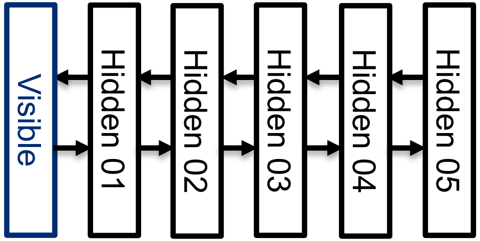
Deep Belief Networks and Auto-Encoders

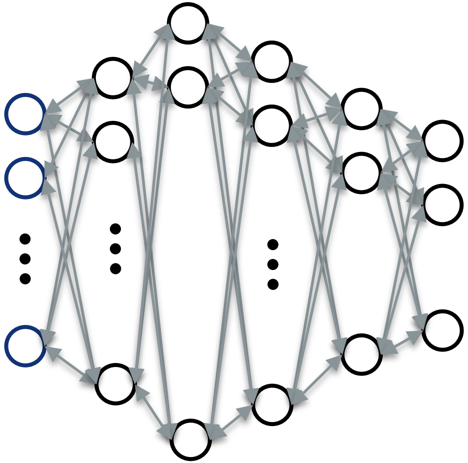
- Deep Belief Networks and Auto-Encoders
  - Multi-layer networks, usually for dimensionality reduction
  - Includes a combination of hidden layers made of CNN layers, layers of neurons, LSTM layers, etc.
  - Obviously, there are no outputs; we have a visible layer (i.e., input layer)
  - Identifying a representation of the input datapoints
  - Encode to get the representation in the last hidden neuron
  - Decode to reconstruct the inputs (e.g., use transpose Conv ?)
  - Loss function is to minimize the reconstruction error



Deep Belief Networks and Auto-Encoders

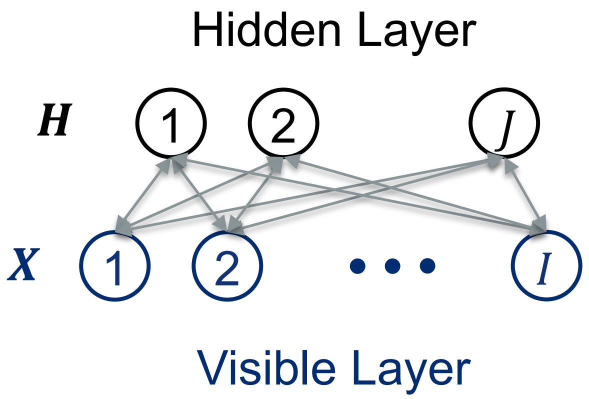
- Deep Belief Networks and Auto-Encoders
  - Simple version: multi-layer neural networks, usually for dimensionality reduction
  - Includes hidden layers of neurons only
  - Double-arrow links
    - Decoding weights could be different from encoding ones or they could be the transpose of the encoding ones
    - Decoding biases could be different from the encoding ones
  - Use any Learning algorithm (usually time-consuming)





Deep Belief Networks and Auto-Encoders

- Restricted Boltzmann Machine (RBM)
  - A two-layer unsupervised neural network with layers of neurons only
    - A visible layer and a hidden layer
    - The results are hidden values and the weights and biases of the links
    - Weights that connects hidden layers to the visible layer are the transpose of the weights that connect visible layer to the hidden layer
  - Iterative learning
    - Start with random weights and hidden and visible biases
    - Use a kernel (i.e., activation) function to transfer values between layers



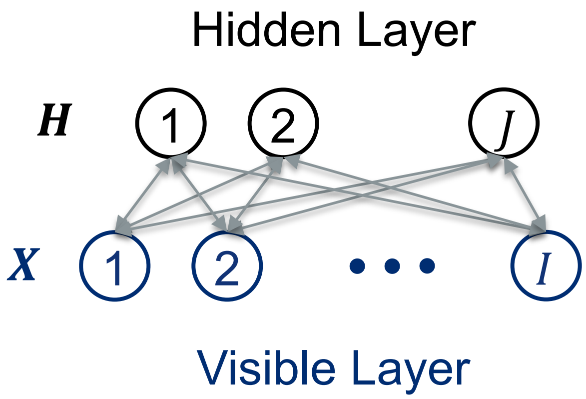
Deep Belief Networks and Auto-Encoders

- Restricted Boltzmann Machine (RBM)

  - Step 1: compute hidden layer

$$\boldsymbol{H} = sigmoid\,(\boldsymbol{XW} + \boldsymbol{OB})$$

where, $\boldsymbol{X}$ is a batch of N 1 by I input datapoints, referred to as visible inputs (so it is a N by I matrix), $\boldsymbol{W}$ is the I by J matrix of weights, $\boldsymbol{B}$ is a 1 by J hidden bias array, $\boldsymbol{O}$ is a N by 1 all-ones array, and $\boldsymbol{H}$ is the N by J matrix of hidden batch.



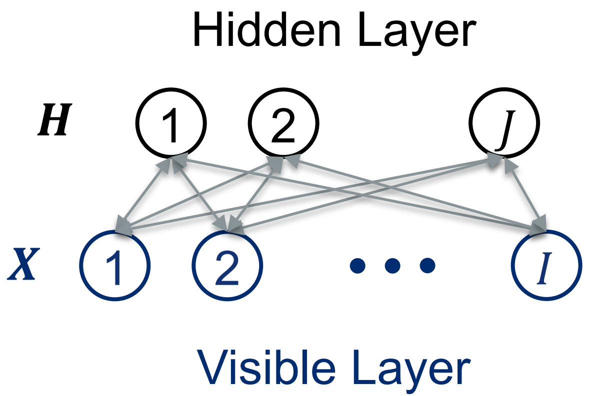
Deep Belief Networks and Auto-Encoders

- Restricted Boltzmann Machine (RBM)

  - Step 2: compute hidden state

$$\widetilde{\boldsymbol{H}} = \boldsymbol{H} \leq \boldsymbol{R}$$

where $\boldsymbol{R}$ is N by J matrix of random values between 0 and 1, and $\widetilde{\boldsymbol{H}}$ is the N by J matrix of Boolean (i.e., 0 for False and 1 for True) hidden state.



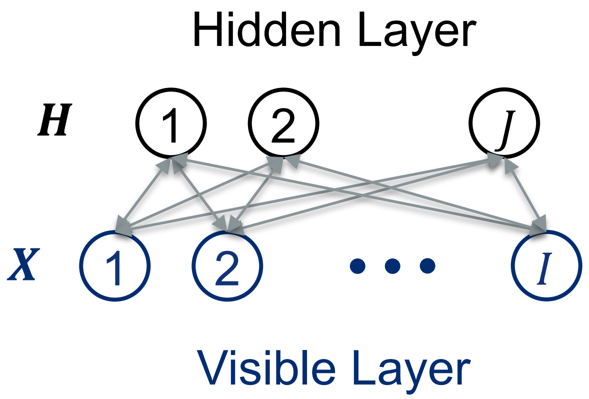
Deep Belief Networks and Auto-Encoders

- Restricted Boltzmann Machine (RBM)

  - Step 3: reconstruct the inputs (called fantasy inputs, $\bar{\boldsymbol{X}}$)

$$\bar{\boldsymbol{X}} = sigmoid\,(\tilde{\boldsymbol{H}}\boldsymbol{W}^T + \boldsymbol{O}\boldsymbol{C})$$

where, T is the transpose function and $\boldsymbol{C}$ is a 1 by I visible bias array and $\bar{\boldsymbol{X}}$ is the N by I matrix of reconstructed inputs (i.e., fantasy inputs).



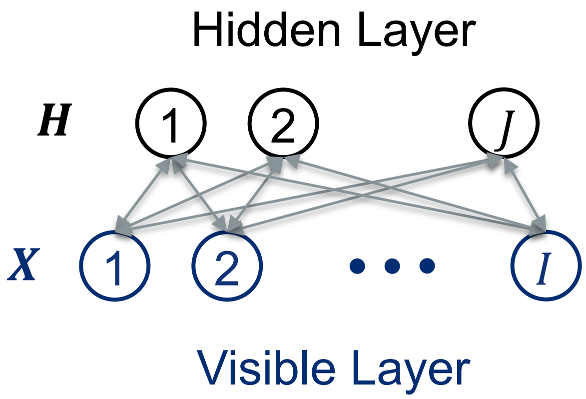
Deep Belief Networks and Auto-Encoders

- Restricted Boltzmann Machine (RBM)

  - Step 4: reconstruct the hidden layer (called fantasy hidden, $\overline{\boldsymbol{H}}$)

$$\overline{\boldsymbol{H}} = sigmoid\,(\overline{\boldsymbol{X}}\boldsymbol{W} + \boldsymbol{OB})$$

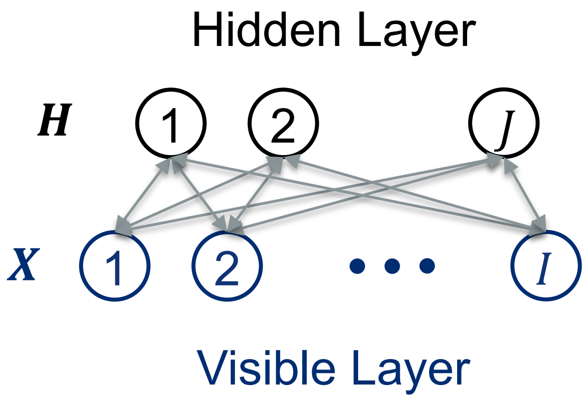
where $\overline{\boldsymbol{H}}$ is the N by J matrix of reconstructed hidden (i.e., fantasy hidden).



Deep Belief Networks and Auto-Encoders

- Restricted Boltzmann Machine (RBM)
  - Step 5: Update the weights and biases by Constructive Divergence (CD):

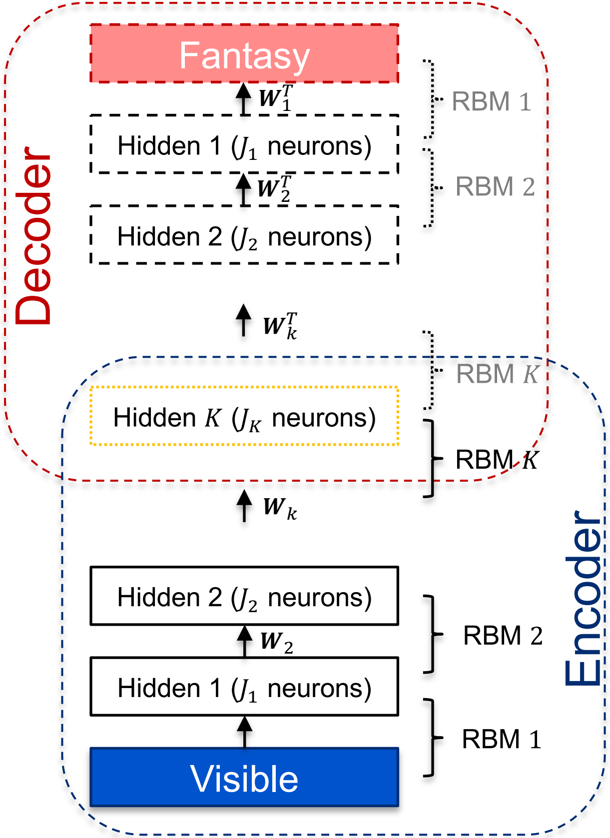
$$\boldsymbol{W}^{s+1} = \boldsymbol{W}^{s} + \Delta\boldsymbol{W}^{s}$$
$$\boldsymbol{B}^{s+1} = \boldsymbol{B}^{s} + \Delta\boldsymbol{B}^{s}$$
$$\boldsymbol{C}^{s+1} = \boldsymbol{C}^{s} + \Delta\boldsymbol{C}^{s}$$
$$\Delta\boldsymbol{W}^{s} = \mu\,\Delta\boldsymbol{W}^{s-1} + \eta\,\frac{\left(\boldsymbol{X}^{T}\boldsymbol{H} - \overline{\boldsymbol{X}}^{T}\overline{\boldsymbol{H}}\right)}{N} - W_{cost}\times\boldsymbol{W}^{s}$$
$$\Delta\boldsymbol{B}^{s} = \mu\,\Delta\boldsymbol{B}^{s-1} + \eta\left(\sum_{n=1}^{N}\boldsymbol{H}_n - \sum_{n=1}^{N}\overline{\boldsymbol{H}}_n\right)$$
$$\Delta\boldsymbol{\Psi}^{s} = \mu\,\Delta\boldsymbol{\Psi}^{s-1} + \eta\left(\sum_{n=1}^{N}\boldsymbol{X}_n - \sum_{n=1}^{N}\overline{\boldsymbol{X}}_n\right)$$



where $s \in \{1,2,\dots,S\}$ is the iteration number, S is the total number of iteration, $\eta \in [0,1]$ is the learning rate, $\mu \in [0,1]$ is the momentum coefficient, $W_{cost} \in [0,1]$ is the weight cost, a parameter that helps to prevent overfitting in each iteration (i.e., regularization). The terms $\boldsymbol{X}_n$, $\overline{\boldsymbol{X}}_n$, $\boldsymbol{H}_n$, and $\overline{\boldsymbol{H}}_n$ are the n^{th} row vector (i.e., datapoint) in $\boldsymbol{X}$, $\overline{\boldsymbol{X}}$, $\boldsymbol{H}$, and $\overline{\boldsymbol{H}}$, respectively.

Deep Belief Networks and Auto-Encoders

- Deep Boltzmann Machine (DBM)
  - Made of multiple RBMs where the hidden layer of one RBM is the visible layer of the next one
  - Composed of an encoder process and a decoder process
  - Use any learning algorithm



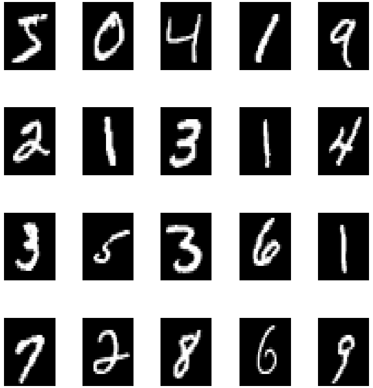
Deep Belief Networks and Auto-Encoders

- Feature Extraction in RBM and DBM
  - The hidden features, are new representation of the input vectors (i.e., visible vector)
  - Connection weights are like CNN kernels, they are like filters, extract appropriate properties from the original features, and decode them into hidden features stochastically
  - In image recognition problems, the connection weights corresponding to a hidden node can be visualized. They usually project a form of edge, local stroke, or pen stroke in the original images.
  - In voice recognition problems, such weights can be heard as a form of phonemes. Feature extraction in RBM are explained through two examples in this section.

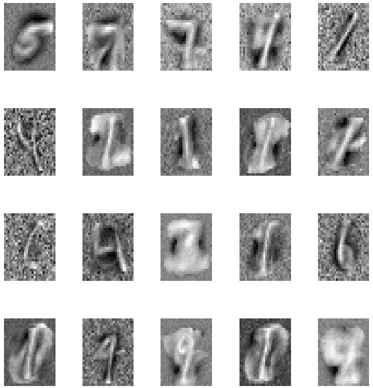
Deep Belief Networks and Auto-Encoders

- Feature Extraction in RBM and DBM

  - Example from MNIST



A sample of MNIST data repository



High-level filters learned by RBM
on a portion of the MNIST dataset

“High-level” refers to filters of the very first layers

Deep Belief Networks and Auto-Encoders

- Feature Extraction in RBM and DBM
  - Example of face image database from Yale University

data repository

High-level filters learned by RBM

http://cvc.cs.yale.edu/cvc/projects/yalefaces/yalefaces.html

Deep Belief Networks and Auto-Encoders

- Feature Extraction in RBM and DBM
  - Example of face image database from Yale University

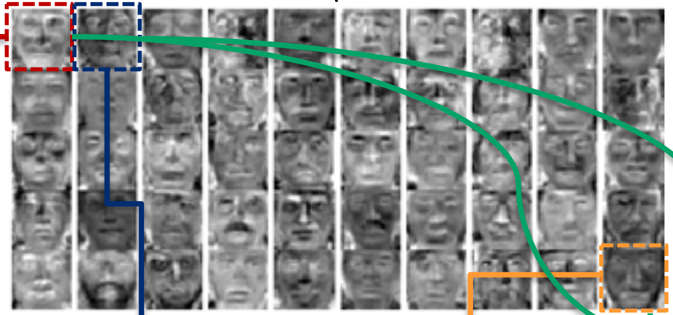
$$\bar{\boldsymbol{X}} = sigmoid\,(\widetilde{\boldsymbol{H}}\boldsymbol{W}^T + \boldsymbol{OC})$$

$$\bar{\boldsymbol{X}} = sigmoid\left(\sum_{j=1}^{J} \widetilde{\boldsymbol{H}}_j \boldsymbol{W}_j^T + \boldsymbol{C}\right)$$

A batch of inputs with 11 images

$$\boldsymbol{H} = sigmoid\,(\boldsymbol{XW} + \boldsymbol{OB}) \longrightarrow \widetilde{\boldsymbol{H}}$$

Features extracted



$$\bar{\boldsymbol{X}} = sigmoid\,(\widetilde{\boldsymbol{H}}_1 \boxed{\boldsymbol{W}_1^T} + \widetilde{\boldsymbol{H}}_2 \boxed{\boldsymbol{W}_2^T} + \ \ldots + \widetilde{\boldsymbol{H}}_J \boxed{\boldsymbol{W}_J^T} + \boldsymbol{C})$$

I-dimensional row vector

Reshaped weights

Reconstructed batch

Deep Belief Networks and Auto-Encoders

- Feature Extraction in RBM and DBM

  - The encoder is the overarching guider of the system, the big picture guy, intended to perform the high functions of the brain, and mathematically the dimensionality reducer.

  - The decoder is the details guy, remembering the higher dimensional aspects.

  - Each hidden layer in the encoder is a feature detector.

  - The decoder is applied to retrieve the so-called fantasy vector using the transpose of the corresponding weight matrices in a way similar to the encoder.

  - The more layer of DBM makes it harder to learn and converge (i.e., optimize the weights and biases)

Deep Belief Networks and Auto-Encoders

- In this lecture, you learned about:
  - Deep Belief Networks and Auto-Encoders
  - Restricted Boltzmann Machine (RBM)
  - Deep Boltzmann Machine (DBM)
- In the next lecture, we will talk about Encoders and Generative Networks

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING