Generative Models

João Paulo Papa and Marcos Cleison Silva Santana

December 17, 2019

UNESP - São Paulo State University
School of Sciences, Departament of Computing
Bauru, SP - Brazil

RECOGNA
LABORATORY

Outline

1. Generative versus Discriminative Models
2. Restricted Boltzmann Machines
3. Deep Belief Networks
4. Deep Boltzmann Machines
5. Conclusions

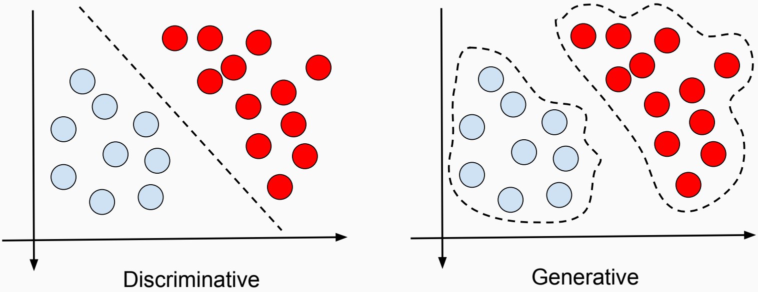
Generative versus Discriminative Models

Introduction

General Concepts:

- Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$ be a dataset where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \mathbb{N}$ stand for a given sample and its label, respectively.
- A **generative** model learns the **conditional probabilities** $p(\mathbf{x}|y)$ and the **class priors** $p(y)$, meanwhile **discriminative** techniques model the conditional probabilities $p(y|\mathbf{x})$.
- Suppose we have a binary classification problem, i.e., $y \in \{1, 2\}$. Generative approaches learn the model of each class, and the decision is taken as the **most likely** one. On the other hand, discriminative techniques put all effort in modeling the **boundary** between classes.

Pictorial example:



Quick-and-dirty example:

- Let $\mathcal{D} = \{(1,1),(1,1),(2,1),(2,2)\}$ be our dataset. Generative approaches compute:
  - $p(y=1) = 0.75$ and $p(y=2) = 0.25$ (**class priors**).
  - $p(x=1|y=1) = 0.50$, $p(x=1|y=2) = 0$, $p(x=2|y=1) = 0.25$ and $p(x=2|y=2) = 0.25$ (conditional probabilities).
- We can then use the **Bayes rule** to compute the posterior probability for classification purposes:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}. \tag{1}$$

Quick-and-dirty example:

- Using Equation 1 to compute the posterior probabilities:

$$
\begin{aligned}
p(y=1|x=1) &= \frac{p(x=1|y=1)p(y=1)}{p(x=1)} \\
&= \frac{p(x=1|y=1)p(y=1)}{p(x=1|y=1)p(y=1)+p(x=1|y=2)p(y=2)} \\
&= \frac{0.50 \times 0.75}{0.50 \times 0.75 + 0 \times 0.25} = \frac{0.50 \times 0.75}{0.50 \times 0.75} = 1.
\end{aligned}
$$

- By keeping doing that, we have $p(y=2|x=1)=0$, $p(y=1|x=2)=0.5$ and $p(y=2|x=2)=0.5$.
- Classification takes the **highest posterior probability**: given a test sample $(1, ?)$, its label is 1 since $p(y=1|x=1)=1$.

Introduction

Summarizing:

- Generative models:
  - Compute $p(\mathbf{x}|y)$ and $p(y)$.
  - Can use both labeled and/or unlabeled data.
  - E.g.: Bayesian classifier, Mixture Models and Restricted Boltzmann Machines.
- Discriminative models:
  - Compute $p(y|\mathbf{x})$.
  - Use labeled data only.
  - E.g.: Support Vector Machines, Logistic Regression and Artificial Neural Networks.

Restricted Boltzmann Machines

Boltzmann Machines

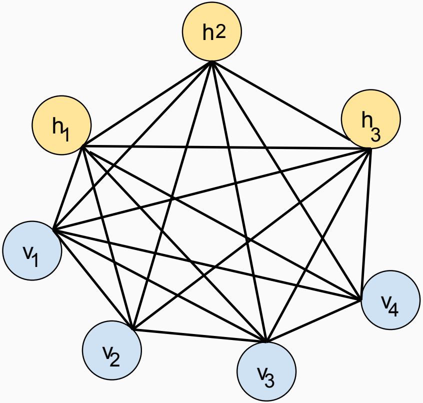
General concepts:

- Simmetrically-connected and neuron-like network.
- Stochastic decisions are taken into account to turn on or off the neurons.
- Proposed initially to learn features from binary-valued inputs.
- Slow for training with many layers of **feature detectors**.
- **Energy**-based model.

Boltzmann Machines

General concepts:

- Let $\mathbf{v} \in \{0,1\}^m$ and $\mathbf{h} \in \{0,1\}^n$ be the set of **visible** and **hidden** layers, respectively. A standard representation of a Boltzmann Machine is given below:



Boltzmann Machines

General concepts:

- Connections are encoded by $\mathbf{W}$, where w_{ij} stands for the connection weight between units i and j.
- Learning algorithm: given a training set (input data), the idea is to find $\mathbf{W}$ in such a way the optimization problem is addressed.
- Let $\mathcal{S} = \{s_1, s_2, \ldots, s_{mn}\}$ be an ordered set composed of the visible and hidden units.
- Each unit s_i updates its state according to the following:

$$z_i = \sum_{j \neq i} w_{ij} s_j + b_i, \tag{2}$$

  where b_i corresponds to the bias of unit s_i.

General concepts:

- Further, unit s_i is turned "on" with a probability given as follows:

$$p(s_i = 1) = \frac{1}{1 + e^{-z_i}}. \tag{3}$$

- If the units are updated sequentially in any order that does not depend on their total inputs, the model will eventually reach a **Boltzmann distribution** where the probability of a given **state** vector $\mathbf{x}$ is determined by the **energy** of that entity with respect to **all possible** binary state vectors $\mathbf{x}'$:

$$p(\mathbf{x}) = \frac{e^{-E(\mathbf{x})}}{\sum_{\mathbf{x}'} e^{-E(\mathbf{x}')}}. \tag{4}$$

General Concepts:

- Boltzmann Machines make small updates in the weights in order to minimize the energy so that the probability of each unit is maximized (the energy of a unit is inversely proportional to its probability).
- Learning phase aims at computing the following partial derivatives:

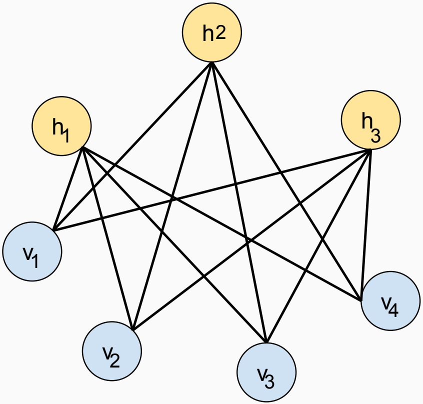
$$\sum_{\mathbf{v} \in \text{data}} \frac{\partial \log p(\mathbf{x})}{\partial w_{ij}}. \tag{5}$$

- Main drawback: it is **impractical** to compute the denominator of Equation 6 for large networks.
- Alternative: Restricted Boltzmann Machines (RBMs).

Restricted Boltzmann Machines

General Concepts:

- Bipartite graphs, i.e., there are no connections between the visible and hidden layers.



General Concepts:

- The learning process is a "bit easier" (computationally speaking).
- The energy is now computed as follows:

$$E(\mathbf{v},\mathbf{h}) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} v_i h_j w_{ij}, \tag{6}$$

  where $\mathbf{a} \in \mathbb{R}^m$ and $\mathbf{b} \in \mathbb{R}^n$ stand for the biases of the visible and hidden layers, respectively.

- The probability of a given configuration $p(\mathbf{v},\mathbf{h})$ can be observed is now computed as follows:

$$p(\mathbf{v},\mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}, \tag{7}$$

  where the denominator stands for the so-called **partition function**.

General Concepts:

- The learning step aims at solving the following problem:

$$\arg\max_{\mathbf{W}} \prod_{\mathbf{v} \in \text{data}} p(\mathbf{v}), \tag{8}$$

which can be addressed by taking the partial derivates in the negative log-likelihood:

$$-\frac{\partial \log p(\mathbf{v})}{\partial w_{ij}} = p(h_j|\mathbf{v})v_i - p(\tilde{h}_j|\tilde{\mathbf{v}}).\tilde{v}_i, \tag{9}$$

where

$$p(h_j|\mathbf{v}) = \sigma\left(\sum_i w_{ij}v_i + b_j\right), \tag{10}$$

and

General Concepts:

$$p(v_i|\mathbf{h}) = \sigma\left(\sum_j w_{ij} h_j + a_i\right), \tag{11}$$

where σ is the sigmoid function. The weights can be updated as follows (considering the whole training set):

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} + \eta(p(\mathbf{h}|\mathbf{v})\mathbf{v} - p(\tilde{\mathbf{h}}|\tilde{\mathbf{v}})\tilde{\mathbf{v}})), \tag{12}$$

where η stands for the learning rate. The conditional probabilities can be computed as follows:

$$p(\mathbf{h}|\mathbf{v}) = \prod_j p(h_j|\mathbf{v}), \tag{13}$$

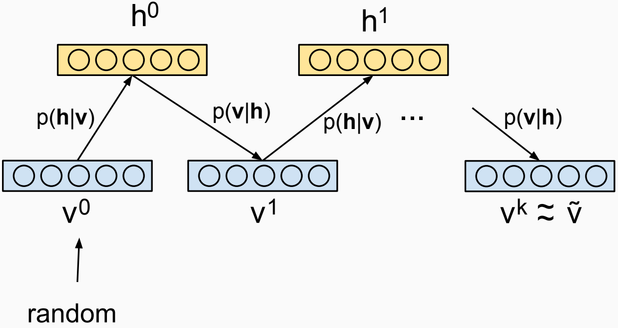
and

$$p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}). \tag{14}$$

Restricted Boltzmann Machines

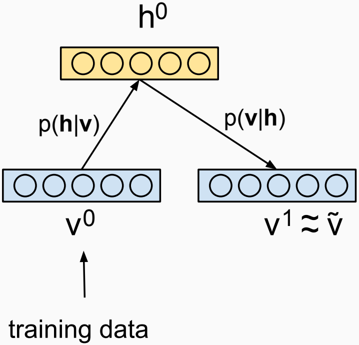
Drawback:

- To compute the "red" part of Equation 9, which is an approximation of the "true" model (training data).
- Standard approach: **Gibbs sampling** (takes time).



Alternative:

- To use the **Contrastive Divergence** (CD).
- CD-k means k sampling steps. It has been shown that CD-1 is enough to obtain a good approaximation.

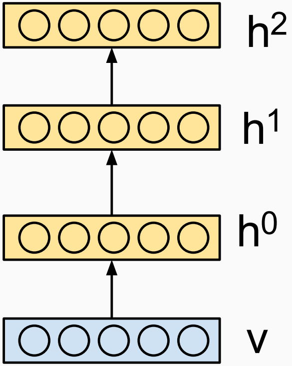


Deep Belief Networks

Deep Belief Networks

General concepts:

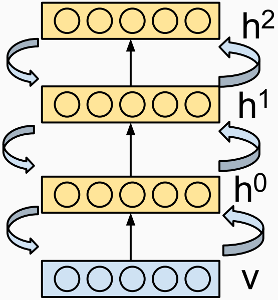
- Composed of stacked RBMs on top of each other.

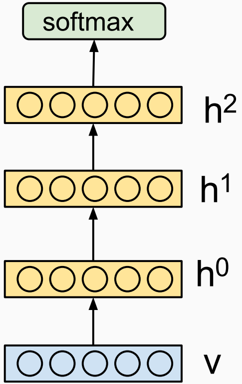


Deep Belief Networks

General concepts:

- Learning can be accomplished in two steps:
  1. A greedy training, where each RBM is trained independently, and the output of one layer serves as the input to the other.
  2. A fine-tuning step (generative or discriminative).



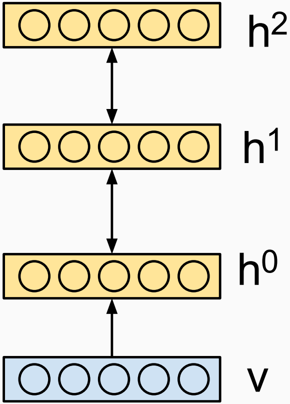


Deep Boltzmann Machines

Deep Boltzmann Machines

General concepts:

- Composed of stacked RBMs on top of each other, but layers from below and above are also considered for inference.



Conclusions

Conclusions

Main remarks:

- RBM-based models can be used for unsupervised feature learning and pre-training networks.
- Simple mathematical formulation and learning algorithms.
- Learning step can be easily made parallel.

RECOGNA
LABORATORY

Thank you!

recogna.tech

marcoscleison.unit@gmail.com
joao.papa@unesp.br