Day 1 Lectures

Lawrence Carin

Duke University

25 July 2017

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

Outline

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

Notation

- Lower case bold letters $\boldsymbol{x}$ are vectors and upper case bold letters $\mathbf{X}$ are matrices
- Non-bold symbols x are scalars
- $\boldsymbol{x} \in \mathbb{R}^N$ is an N-dimensional vector, each element of which is real
- $\mathbf{X} \in \mathbb{R}^{N \times M}$ is an $N \times M$ matrix, each element of which is real
- $\boldsymbol{x} \in \{0, 1\}^N$ and $\mathbf{X} \in \{0, 1\}^{N \times M}$ are similarly vectors and matrices, with binary elements
- $\boldsymbol{x} \in \mathbb{Z}_+^N$ and $\mathbf{X} \in \mathbb{Z}_+^{N \times M}$ are vectors and matrices with elements that are non-negative integers (i.e., counts)

Distributions and More Notation

- We will often characterize the data $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1,M}$ as being drawn from a distribution

- $p(\boldsymbol{x}; \boldsymbol{\theta})$ denotes a distribution with parameters $\boldsymbol{\theta}$

- Typically we assume each $\boldsymbol{x}_i$ drawn independently, denoted

$$\boldsymbol{x}_i \sim p(\boldsymbol{x}; \boldsymbol{\theta}), \forall\ i = 1, \ldots, M$$

- For example, the Gaussian distribution for real vector $\boldsymbol{x} \in \mathbb{R}^N$ is represented

$$p(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{\exp[(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})]}{\sqrt{(2\pi)^N |\boldsymbol{\Sigma}|}}$$

where $\boldsymbol{\theta}$ is composed of mean $\boldsymbol{\mu} \in \mathbb{R}^N$ and the positive-definitive covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{N \times N}$

Problem 1

- Represent the covariance matrix in the eigendecomposition

$$\mathbf{\Sigma} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T = \mathbf{U}\mathbf{\Lambda}^{1/2}\mathbf{\Lambda}^{1/2}\mathbf{U}^T$$

where the columns of $\mathbf{U} \in \mathbb{R}^{N\times N}$ are orthonormal, and $\mathbf{\Lambda} \in \mathbb{R}_+^{N\times N}$ is a diagonal matrix with non-negative elements. The decomposition is arranged such that the eigenvalues are in order of descending amplitude, $i.e.$, $\lambda_{i+1,j+1} < \lambda_{i,j}$. Prove that drawing $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \mathbf{\Sigma})$ is equivalent to the construction

$$\boldsymbol{x}_i = \boldsymbol{\mu} + \mathbf{U}\mathbf{\Lambda}^{1/2}\boldsymbol{h}_i$$

with $\boldsymbol{h}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$, where $\mathbf{I}_N$ is the $N \times N$ identity matrix.

Principal Component Analysis

- In the above setup, data assumed generated

$$\boldsymbol{x}_i = \boldsymbol{\mu} + \mathbf{U}\boldsymbol{\Lambda}^{1/2}\boldsymbol{h}_i, \qquad \boldsymbol{h}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$$

- Often only a small subset of the diagonal elements of $\boldsymbol{\Lambda}$ have significant amplitude

- Let $\boldsymbol{\Lambda}_P \in \mathbb{R}^{P \times P}$ be a diagonal matrix; the diagonal elements of $\boldsymbol{\Lambda}_P$ are the same as the P *largest* diagnonal elements in $\boldsymbol{\Lambda}$

- $\mathbf{U}_P \in \mathbb{R}^{N \times P}$ composed of the first P columns of $\mathbf{U}$

- PCA:

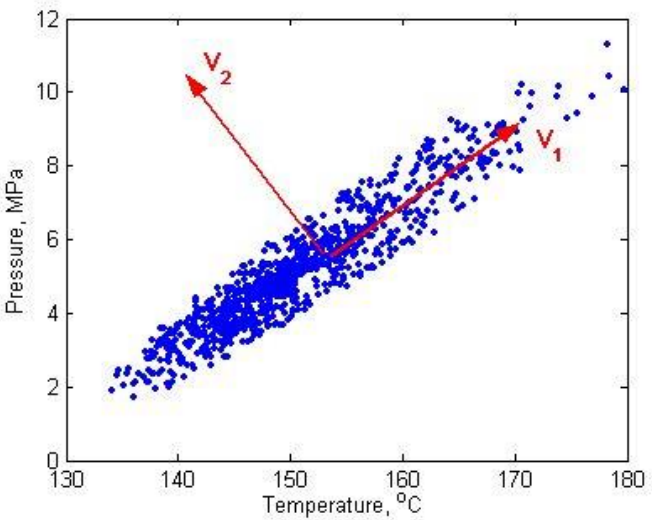
$$\begin{aligned} \boldsymbol{h}_i &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}_P) \\ \boldsymbol{z}_i &= \boldsymbol{\mu} + \mathbf{U}_P\boldsymbol{\Lambda}_P^{1/2}\boldsymbol{h}_i \\ \boldsymbol{x}_i &\sim \mathcal{N}(\boldsymbol{z}_i, \alpha^{-1}\mathbf{I}_N) \end{aligned}$$

Global and Latent Parameters of Generative Model

$$
\begin{aligned}
\boldsymbol{h}_i &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}_P) \\
\boldsymbol{z}_i &= \boldsymbol{\mu} + \mathbf{U}_P \boldsymbol{\Lambda}_P^{1/2} \boldsymbol{h}_i \\
\boldsymbol{x}_i &\sim \mathcal{N}(\boldsymbol{z}_i, \alpha^{-1}\mathbf{I}_N)
\end{aligned}
$$

- Parameters $\boldsymbol{h}_i$ are latent, and unique to each associated data $\boldsymbol{x}_i$
- Parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}, \mathbf{U}_P, \boldsymbol{\Lambda}_P\}$ are “global,” shared among all $\boldsymbol{x}_i$
- We seek to *learn* $\boldsymbol{\theta}$ from training data $\{\boldsymbol{x}_i\}_{i=1,M}$
- At test time, we often wish to *infer* $\boldsymbol{h}_i$ for an associated test $\boldsymbol{x}_i$

PCA Example, $P = 2$



Factor Analysis

- The above model may be expressed, for data sample $\boldsymbol{x}_i$, as

$$\begin{aligned} \boldsymbol{h}_i &\sim \mathcal{N}(0, \mathbf{I}_P) \\ \boldsymbol{z}_i &= \boldsymbol{\mu} + \mathbf{F}\boldsymbol{h}_i \\ \boldsymbol{x}_i &\sim \mathcal{N}(\boldsymbol{z}_i, \alpha^{-1}\mathbf{I}_N) \end{aligned}$$

  where the P columns of $\mathbf{F} \in \mathbb{R}^{N \times P}$ were orthogonal

- In *Factor Analysis* we generalize this concept:
  - The columns of $\mathbf{F}$ need only be *linearly independent*, and need not be orthogonal
  - The distribution from which the factor scores $\boldsymbol{h}_i$ are drawn can be generalized, e.g., to impose sparsity

Linearity

- Recall the factor model

$$\boldsymbol{h}_i \sim \mathcal{N}(0, \mathbf{I}_P)$$
$$\boldsymbol{z}_i = \boldsymbol{\mu} + \mathbf{F}\boldsymbol{h}_i$$
$$\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{z}_i, \alpha^{-1}\mathbf{I}_N)$$

- The relationship $\boldsymbol{h}_i \to \boldsymbol{z}_i$ constitutes an affine transformation

- $\mathbf{F}\boldsymbol{h}_i$ is a *linear* transformation, the result of which is within a subspace spanned by the columns of $\mathbf{F}$

- $\boldsymbol{\mu}$ Constitutes a shift/rotation of the location of the linear subspace

Poisson Distribution

- The Poisson distribution is widely used to model count data, $x \in \mathcal{Z}_+$

$$p(x = k; \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}, \quad \text{with } k \text{ a non-negative integer}$$

- The model parameters for the Poisson distribution are $\theta = \lambda \geq 0$
- $\mathbb{E}(x) = \mathsf{Var}(x) = \lambda$
- Let $\boldsymbol{\lambda} \in \mathbb{R}_+^K$, then if $\boldsymbol{x} \sim \text{Poisson}(\boldsymbol{\lambda})$, then $x_k \sim \text{Poisson}(\lambda_k)$

Poisson Factor Analysis

- Assume we wish to model data $\boldsymbol{x}_i \in \mathcal{Z}_+^N$, with $i = 1, \dots, M$
- For example, counts in a corpus of M documents, with an N-dimensional vocabulary
- Generalize the factor model:

$$\begin{aligned} \boldsymbol{h}_i &\sim p(\boldsymbol{h}) \\ \boldsymbol{\lambda}_i &= \boldsymbol{\lambda}_0 + \mathbf{F}\boldsymbol{h}_i \\ \boldsymbol{x}_i &\sim \text{Poisson}(\boldsymbol{\lambda}_i) \end{aligned}$$

- $\mathbf{F} \in \mathbb{R}_+^{N \times K}$ is a non-negative real *factor loading* matrix and $\boldsymbol{\lambda}_0 \in \mathbb{R}_+^N$
- Appropriate $p(\boldsymbol{h})$ developed later in the Summer School

Outline

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

Bernoulli Distribution and Sigmoid Link

- The **Bernoulli distribution** is employed for random variable $x \in \{0, 1\}$ and is characterized by the single parameter π, with $0 < \pi < 1$. If $x \sim \text{Bern}(\pi)$, then $x = 1$ with probability π, and $x = 0$ with probability $1 - \pi$.

- The **sigmoid function** $\sigma(y)$ for $y \in \mathbb{R}$ is defined as

$$\sigma(y) = \exp(y)/[1 + \exp(y)] \in (0, 1)$$

- The sigmoid function may be employed to characterize the Bernoulli distribution in terms of a *real* variable y:

$$x \sim \text{Bern}(\sigma(y))$$

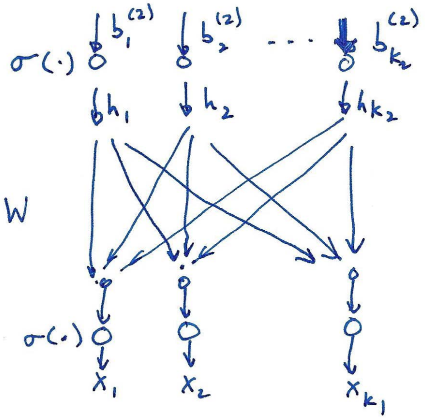
Sigmoid Belief Network (SBN)

- Assume that the data of interest are binary, $\boldsymbol{x} \in \{0,1\}^N$
- The distribution of the data is represented $p(\boldsymbol{x}; \boldsymbol{b}^{(1)}, \boldsymbol{b}^{(2)}, \mathbf{W})$ where $\boldsymbol{b}^{(1)} \in \mathbb{R}^{K_1}$, $\boldsymbol{b}^{(2)} \in \mathbb{R}^{K_2}$ and $\mathbf{W} \in \mathbb{R}^{K_2 \times K_1}$.
- h_k, $b_k^{(1)}$, and $b_k^{(2)}$ represent component k of vectors $\boldsymbol{h}$, $\boldsymbol{b}^{(1)}$ and $\boldsymbol{b}^{(2)}$, respectively
- Data drawn from SBN as follows:

$$\begin{aligned} h_k &\sim \text{Bern}(\sigma(b_k^{(2)})),\ \ k = 1, \dots, K_2 \\ \boldsymbol{z} &= \mathbf{W}\boldsymbol{h} + \boldsymbol{b}^{(1)} \\ x_j &\sim \text{Bern}(\sigma(z_j)),\ \ j = 1, \dots, K_1 \end{aligned}$$

- $\boldsymbol{h}$ is a *latent* random variable for the observed data $\boldsymbol{x}$

Schematic of SBN



Problem 2

- Let $\mathbf{W}_{k:}$ represent the kth row of $\mathbf{W}$, and hence $x_k \sim \text{Bern}(\sigma(\mathbf{W}_{k:}\boldsymbol{h} + b_k^{(1)}))$, or stated otherwise

$$\begin{aligned} p(x_k = 1) &= \exp(\mathbf{W}_{k:}\boldsymbol{h} + b_k^{(1)})/(1 + \exp(\mathbf{W}_{k:}\boldsymbol{h} + b_k^{(1)})), \\ p(x_k = 0) &= 1/(1 + \exp(\mathbf{W}_{k:}\boldsymbol{h} + b_k^{(1)})) \end{aligned}$$

- The distribution model parameters are $\boldsymbol{\theta} = (\mathbf{W}, \boldsymbol{b}^{(1)}, \boldsymbol{b}^{(2)})$. Show that the joint distribution for the data $\boldsymbol{x}$ and associated latent variables $\boldsymbol{h}$ is

$$p(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{x}^T\mathbf{W}\boldsymbol{h} + \boldsymbol{x}^T\boldsymbol{b}^{(1)} + \boldsymbol{h}^T\boldsymbol{b}^{(2)})}{\prod_{k=1}^{K_1}[1 + \exp(\mathbf{W}_{k:}\boldsymbol{h} + b_k^{(1)})]\prod_{k'=1}^{K_2}[1 + \exp(b_{k'}^{(2)})]}$$

- Give an expression for the *marginal* probability of the data $p(\boldsymbol{x}; \boldsymbol{\theta})$

Restricted Boltzmann Machine (RBM)

- The directed nature of the SBN graphical model is attractive, but the form of $p(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta})$ is relatively complicated

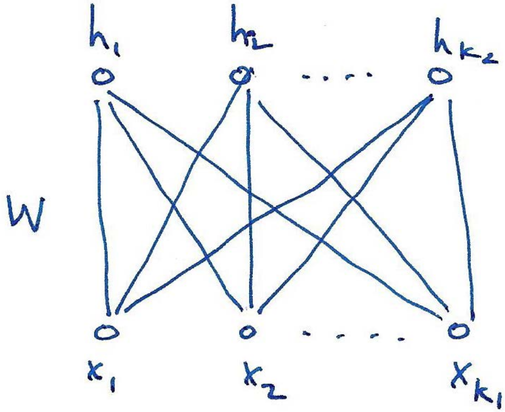
- This motivates the RBM, with distribution defined

$$p(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{x}^T \mathbf{W} \boldsymbol{h} + \boldsymbol{x}^T \boldsymbol{b}^{(1)} + \boldsymbol{h}^T \boldsymbol{b}^{(2)})}{\mathcal{Z}(\boldsymbol{\theta})}$$

where $\mathcal{Z}(\boldsymbol{\theta})$ is the *partition function*,

$$\mathcal{Z}(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \sum_{\boldsymbol{h}} \exp(\boldsymbol{x}^T \mathbf{W} \boldsymbol{h} + \boldsymbol{x}^T \boldsymbol{b}^{(1)} + \boldsymbol{h}^T \boldsymbol{b}^{(2)})$$

Schematic of RBM



Problem 3

- For the RBM, prove that:

$$p(x_k|\boldsymbol{h};\boldsymbol{\theta}) = \mathsf{Bern}(\sigma(\mathbf{W}_{k:}\boldsymbol{h} + b_k^{(1)}))$$
$$p(h_k|\boldsymbol{x};\boldsymbol{\theta}) = \mathsf{Bern}(\sigma(\boldsymbol{x}^T\mathbf{W}_{:k} + b_k^{(2)}))$$

Philosophy of Generative-Model Approach

- Given training data $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1,M}$, the goal is to learn model parameters $\boldsymbol{\theta}$ that fit the data

- Data generation modeled as

$$\boldsymbol{h}_i \sim p_h(\boldsymbol{h}; \boldsymbol{\theta})$$
$$\boldsymbol{x}_i \sim p_x(\boldsymbol{x}|\boldsymbol{h}; \boldsymbol{\theta})$$

- Given a new (test) sample $\boldsymbol{x}_*$, the objective is to infer $\boldsymbol{h}_*$

- The inferred $\boldsymbol{h}_*$ often yields information/insight about $\boldsymbol{x}_*$

- The dimension of $\boldsymbol{h}$ is often much smaller than N, and therefore it may be better to perform classification, regression or clustering based upon $\boldsymbol{h}$ rather than on $\boldsymbol{x}$

Modeling Without a Generative Model

- Assume that our goal is to learn a mapping $\boldsymbol{x} \to y$, where $y \in \mathbb{R}$ (regression) or $y \in \{0, 1\}$ (binary classification)
- The generative-modeling approach would consider a model of the form

$$\text{Inference} : \boldsymbol{x} \to \boldsymbol{h}, \qquad \text{Prediction} : \boldsymbol{h} \to y$$

- We develop a model of the data $\boldsymbol{x}$ in terms of latent $\boldsymbol{h}$; the mapping to y is performed using $\boldsymbol{h}$
- The generative approach is appropriate when one wants to *understand* the data $\boldsymbol{x}$, or if the training data is

$$\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1,M_1} \cup \{\boldsymbol{x}_i, y_i\}_{i=M_1+1,M_2}$$

with $M_1 \gg M_2$

Non-Generative Model

- Another class of models avoids generative learning altogether
- Given training data $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1,M}$, the goal is to explicitly learn functional mapping

$$\hat{y}(\boldsymbol{x}) = f(\boldsymbol{x}; \boldsymbol{\phi})$$

- No attempt is made to model the data itself; $\boldsymbol{\phi}$ are the parameters of the function to be learned
- Neural networks are an important class of such models

Outline

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

From RBM to Feedforward Model

- Recall that for the restricted Boltzmann machine (RBM), we have

$$p(x_k|\boldsymbol{h};\boldsymbol{\theta}) = \text{Bern}(\sigma(\mathbf{W}_{k:}\boldsymbol{h} + b_k^{(1)})), \quad p(h_k|\boldsymbol{x};\boldsymbol{\theta}) = \text{Bern}(\sigma(\boldsymbol{x}^T\mathbf{W}_{:k} + b_k^{(2)}))$$

- Given $\boldsymbol{x}$, we may be interested in performing a task with the latent $\boldsymbol{h}$

- For example, we may have *labeled* training data $\{\boldsymbol{x}_i, y_i\}_{i=1,M}$, with $\boldsymbol{x}_i \in \mathbb{R}^N$ and $y_i \in \{0,1\}$

- We may develop the model

$$p(y_i = 1|\boldsymbol{h}_i;\boldsymbol{w}) = \sigma(\boldsymbol{w}^T\boldsymbol{h}_i)$$

implying

$$p(y_i, \boldsymbol{h}_i|\boldsymbol{x}_i;\boldsymbol{w},\mathbf{W},\boldsymbol{b}^{(2)}) = \underbrace{\text{Bern}(y_i;\sigma(\boldsymbol{w}^T\boldsymbol{h}_i))}_{p(y_i|\boldsymbol{h}_i;\boldsymbol{\theta})} \underbrace{\prod_{k=1}^{K_2} \text{Bern}(h_{ik};\sigma(\boldsymbol{x}^T\mathbf{W}_{:k} + b_k^{(2)}))}_{p(\boldsymbol{h}_i|\boldsymbol{x}_i;\boldsymbol{\theta})}$$

from RBM

Approximation to RBM

- Marginalizing out the latent $\boldsymbol{h}$, we have

$$p(y=1|\boldsymbol{x}) = \sum_{\boldsymbol{h}} \sigma(\boldsymbol{w}^T \boldsymbol{h})) \underbrace{\prod_{k=1}^{K_2} [\sigma(\boldsymbol{x}^T \mathbf{W}_{:k} + b_k^{(2)})]^{h_k} [1 - \sigma(\boldsymbol{x}^T \mathbf{W}_{:k} + b_k^{(2)}]^{1-h_k}}_{p(\boldsymbol{h}|\boldsymbol{x})}$$

$$= \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{x})}[\sigma(\boldsymbol{w}^T \boldsymbol{h})]$$

- Rough approximation:

$$p(y=1|\boldsymbol{x}) = \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{x})}[\sigma(\boldsymbol{w}^T \boldsymbol{h})] \approx \sigma(\boldsymbol{w}^T \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{x})}(\boldsymbol{h}))$$

- **Problem 4**: Prove that $\mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{x})}(\boldsymbol{h}) = \sigma(\mathbf{W}^T \boldsymbol{x} + \boldsymbol{b}^{(2)})$, where the $\sigma(\cdot)$ is applied pointwise

Multilayered Perceptron

- Rather than modeling the data $\{\boldsymbol{x}_i\}_{i=1,M}$, the MLP considers labeled data $\{\boldsymbol{x}_i, y_i\}_{i=1,M}$, and learns $p(y_i|\boldsymbol{x}_i)$

$$p(y_i = 1|\boldsymbol{x}_i) = \sigma(\boldsymbol{w}^T \boldsymbol{h}_i^*)$$
$$\boldsymbol{h}_i^* = \sigma(\mathbf{W}^T \boldsymbol{x}_i + \boldsymbol{b})$$

- In RBM $\boldsymbol{h} \in \{0,1\}^{K_2}$; for MLP $\boldsymbol{h}^* \in (0,1)^{K_2}$

- The MLP is very fast at test: $\boldsymbol{x}_i \rightarrow \boldsymbol{h}_i^*$ an explicit functional mapping

- May "pretrain" for $\mathbf{W}$ and $\boldsymbol{b}$ using an RBM on unlabeled data $\{\boldsymbol{x}_i\}_{i=1,M_1}$, and then "refine" $\mathbf{W}$ and $\boldsymbol{b}$, plus learn $\boldsymbol{w}$, using labeled data $\{\boldsymbol{x}_i, y_i\}_{i=M_1+1,M_1+M_2}$

Generalized MLP Nonlinearity

- The form of the MLP discussed above

$$\begin{aligned} p(y_i = 1|\boldsymbol{x}_i) &= \sigma(\boldsymbol{w}^T \boldsymbol{h}_i^*) \\ \boldsymbol{h}_i^* &= \sigma(\mathbf{W}\boldsymbol{x}_i + \boldsymbol{b}) \end{aligned}$$

  was motivated by connections to the RMB

- Can generalize as

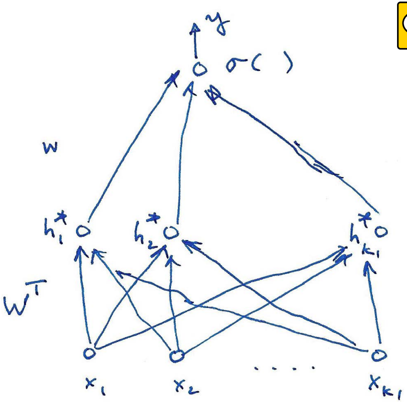
$$\begin{aligned} p(y_i = 1|\boldsymbol{x}_i) &= \sigma(\boldsymbol{w}^T \boldsymbol{h}_i^*) \\ \boldsymbol{h}_i^* &= g(\mathbf{W}\boldsymbol{x}_i + \boldsymbol{b}) \end{aligned}$$

  where $g(\cdot)$ is a general nonlinear function

- Examples for $g(\cdot)$: tanh$(\cdot)$ and the rectified linear unit (ReLu):

$$g(s) = s \;\; \text{if } s > 0, \qquad g(s) = 0 \;\; \text{if } s < 0$$

MLP Schematic



Non-Generative Model: Filter Plus Nonlinearity

- Recall the MLP:

$$
\begin{aligned}
p(y_i = 1 | \boldsymbol{x}_i) &= \sigma(\boldsymbol{w}^T \boldsymbol{h}_i^*) \\
\boldsymbol{h}_i^* &= g(\mathbf{W}\boldsymbol{x}_i + \boldsymbol{b})
\end{aligned}
$$

- Hidden variable $\boldsymbol{h}_i^*$ may be viewed as formed by filtering followed by nonlinearity

- Filter: $\mathbf{W}\boldsymbol{x}_i$

- Nonlinearity: $g(\cdot + \boldsymbol{b})$

- Hidden variables $\boldsymbol{h}_i^*$ then feed logistic classifier: $\sigma(\boldsymbol{w}^T \boldsymbol{h}_i^*)$

Outline

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

Generative Model Global & Local Parameters

- For SBN and RBM models, we have distributions of the form $p(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta})$

- $\boldsymbol{x}$ represents the observed data, and $\boldsymbol{h}$ represents the latent (or "hidden") parameters associated with $\boldsymbol{x}$

- The parameters $\boldsymbol{\theta}$ are "global," in the sense that they are associated with all observed $\boldsymbol{x}$

- Latent parameters $\boldsymbol{h}$ are "local," in that they are linked with associated observations $\boldsymbol{x}$

- The distribution of the data, given $\boldsymbol{\theta}$, is $p(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{\boldsymbol{h}} p(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta})$

Learning for Generative Models

- Assume that we are given data $p(\mathcal{D}) = \{\boldsymbol{x}_i\}_{i=1,M}$, which we model with $p(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta})$

- When *learning* our goal is to estimate $\boldsymbol{\theta}$ based on $\mathcal{D}$

- Assuming that the data are drawn i.i.d. from our model distribution, we have

$$p(\mathcal{D}; \boldsymbol{\theta}) = \prod_{i=1}^{M} p(\boldsymbol{x}_i; \boldsymbol{\theta})$$

- Two perspectives:

  - Optimization: Seek a single *point* estimate for $\boldsymbol{\theta}$, denoted $\hat{\boldsymbol{\theta}}(\mathcal{D})$
  - Bayesian: Seek a *distribution* $p(\boldsymbol{\theta}|\mathcal{D})$

- The learning is performed based on given training data $\mathcal{D}$

Inference

- Based on training data $\mathcal{D}$, assume we have learned a model $p(\boldsymbol{x}, \boldsymbol{h}; \mathcal{D})$

- If optimization-based learning is performed:

$$p(\boldsymbol{x}, \boldsymbol{h}; \mathcal{D}) = p(\boldsymbol{x}, \boldsymbol{h}; \hat{\boldsymbol{\theta}}(\mathcal{D}))$$

- If Bayesian-based learning is performed:

$$p(\boldsymbol{x}, \boldsymbol{h} | \mathcal{D}) = \int d\boldsymbol{\theta} p(\boldsymbol{x}, \boldsymbol{h}; \boldsymbol{\theta}) p(\boldsymbol{\theta} | \mathcal{D})$$

- Numerical methods are typically employed to implement the integral in the Bayesian setup

- Given a *new* data sample $\boldsymbol{x}$, the goal is to infer the associated latent $\boldsymbol{h}$:
  - Optimization: Seek a single *point* estimate for $\boldsymbol{h}$, denoted $\hat{\boldsymbol{h}}(\boldsymbol{x})$
  - Bayesian: Seek a *distribution* $p(\boldsymbol{h}|\boldsymbol{x})$

Outline

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

Summary of the Bayesian Setup

- Assume a model $\boldsymbol{x}_i \sim p(\boldsymbol{x}|\boldsymbol{\theta})$, where the model parameters $\boldsymbol{\theta}$ are treated as an unknown random variable

- For simplicity we assume that the latent $\boldsymbol{h}_i$ is marginalized out, i.e.,

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \int d\boldsymbol{h} p(\boldsymbol{x}, \boldsymbol{h}|\boldsymbol{\theta})$$

- We assume that the model parameters are drawn from a prior distribution $p(\boldsymbol{\theta})$

- Assuming training data $\boldsymbol{x}_i \sim p(\boldsymbol{x}|\boldsymbol{\theta})$, $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1,M}$, the *posterior* distribution of the model parameters is

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\boldsymbol{\theta}) \prod_{i=1}^{M} p(\boldsymbol{x}_i|\boldsymbol{\theta})}{\int d\boldsymbol{\theta} p(\boldsymbol{\theta}) \prod_{i=1}^{M} p(\boldsymbol{x}_i|\boldsymbol{\theta})} = \frac{p(\boldsymbol{\theta}) \prod_{i=1}^{M} p(\boldsymbol{x}_i|\boldsymbol{\theta})}{\mathcal{Z}(\mathcal{D})}$$

Summary of the Optimization Setup

- A maximum *a posterior* (MAP) point estimation of the model parameters may be determined by maximizing the log posterior:

$$\hat{\boldsymbol{\theta}} = \mathrm{argmax}_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}|\mathcal{D}) = \mathrm{argmax}_{\boldsymbol{\theta}} \{ \sum_{i=1}^{M} \log p(\boldsymbol{x}_i|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \}$$

- The term $\log p(\boldsymbol{\theta})$ may be reviewed as a *regularizer*, that constraints the range of $\boldsymbol{\theta}$ considered as we fit the data with the term $\displaystyle\sum_{i=1}^{M} \log p(\boldsymbol{x}_i|\boldsymbol{\theta})$

Optimization for Non-Generative Models

- Assume we desire to estimate a direct function mapping $\hat{\ell}(\boldsymbol{x}) = f(\boldsymbol{x}; \boldsymbol{\theta})$

- There is no generative model as employed in the above discussion

- We generalize the concept of loss function:

$$\hat{\boldsymbol{\phi}} = \text{argmin}_{\phi}\{\sum_{i=1}^{I} \ell[y_i, f(\boldsymbol{x}_i; \boldsymbol{\phi})] + r(\boldsymbol{\phi})\}$$

- $\ell[y_i, f(\boldsymbol{x}_i; \boldsymbol{\phi})]$ is an appropriate loss function between the true y_i and the estimate $\hat{y}_i = f(\boldsymbol{x}_i; \boldsymbol{\phi})$

- $r(\boldsymbol{\phi})$ is a regularization (constraint) imposed on the parameters $\boldsymbol{\phi}$

Outline

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

Big Data

- In the above discussions, we assume data $\{\boldsymbol{x}_i\}_{i=1,M}$ and/or $\{\boldsymbol{x}_i, y_i\}_{i=1,M}$ are available to learn a model

- The model may be a generative description of the data $p(\boldsymbol{x}_i, \boldsymbol{h}_i; \boldsymbol{\theta})$, typically with latent parameters $\boldsymbol{h}_i$

- The model may also be non-generative for $\boldsymbol{x}$, and may yield $p(y_i|\boldsymbol{x}_i; \boldsymbol{\phi})$ with parameters $\boldsymbol{\phi}$

- The model fit is performed with the M training samples

- For massive M, a naive implementation of learning (optimization or Bayesian) is computationally intractable

Approximations for Big Data

- The computational challenge is manifested by the data-fit terms

$$\sum_{i=1}^{M} \ell[y_i, f(\boldsymbol{x}_i; \boldsymbol{\phi})] \quad \text{or} \quad \sum_{i=1}^{M} \log\ p(\boldsymbol{x}_i|\boldsymbol{\theta})$$

- In the Summer School, we will consider methods whereby these are represented by random approximations

$$\frac{M}{m}\sum_{i\in\mathcal{S}} \ell[y_i, f(\boldsymbol{x}_i; \boldsymbol{\phi})] \quad \text{or} \quad \frac{M}{m}\sum_{i\in\mathcal{S}} \log\ p(\boldsymbol{x}_i|\boldsymbol{\theta})$$

  with $\mathcal{S}$ a *random* selection of m elements from $\{1,\ldots,M\}$, and $m \ll M$

- This random subsampling will be examined from multiple perspectives, for both optimization and Bayesian learning/inference

Outline

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

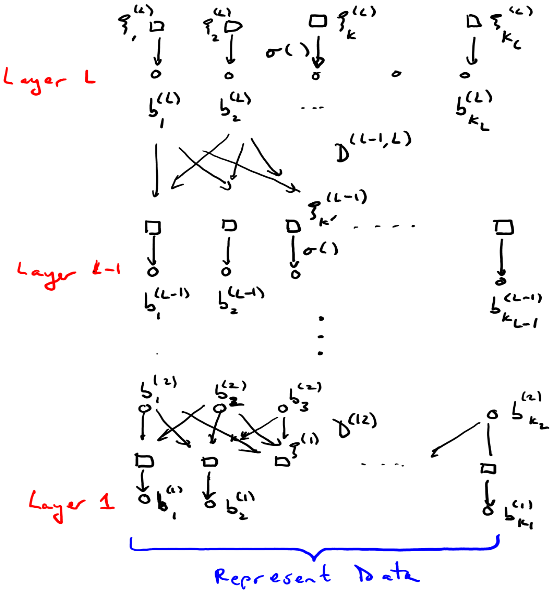
Review of SBN, RBM and MLP

- The SBN and RBM are generative models of data $\{\boldsymbol{x}_i\}_{i=1,M}$

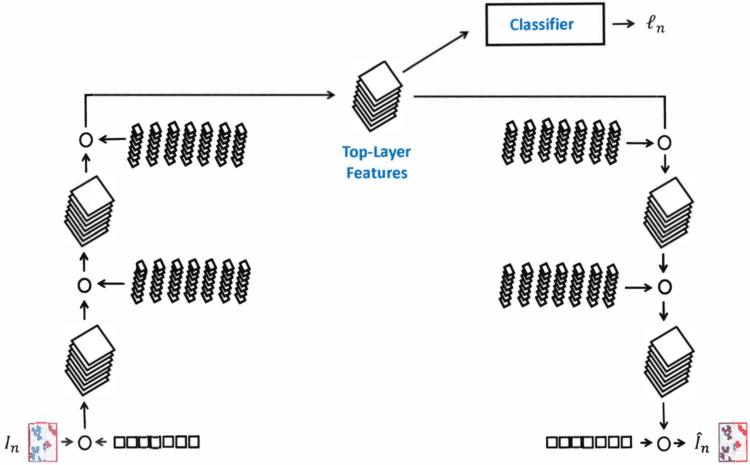
- MLP is a discriminative functional mapping from $\boldsymbol{x}_i \rightarrow y_i$, based on training data $\{\boldsymbol{x}_i, y_i\}_{i=1,M}$

- For images, the vector products in these models are simply replaced by *convolutional* filter

Deep Sigmoid Belief Network



Deep Discriminative & Generative Convolutional Models



Outline

1. Notation & Basic Generative Models
2. Nonlinear Generative Models: SBN & RBM
3. Non-Generative Models: Feedforward Neural Network
4. Model Learning and Inference
5. Bayesian- and Optimization-Based Learning/Inference
6. Challenges and Opportunities of Big Data
7. Adding Model Depth
8. Summer School Road Map

Roadmap for the Summer School

- Plan for the remainder of the MLSS