Assignment 4 for #70240413
"Statistical Machine Learning"

Instructor: Prof. Jun Zhu

June 10, 2018

1 Probabilistic Graphical Models

1.1 Conditional Queries in a Bayesian Network

Consider the Bayesian network in Fig. 1 which represents the following fictitious biological model. Each G_i represents the genotype of a person: $G_i = 1$ if they have a healthy gene and $G_i = 2$ if they have an unhealthy gene. G_2 and G_3 represent the descendants of G_1 and therefore may inherit this specific gene from G_1. $X_i \in \mathbb{R}$ is a continuous measure of blood pressure, which is low if the person is healthy or high if unhealthy.

We define the CPDs as follows

$$P(G_1) = (0.5, 0.5) \tag{1}$$

$$P(G_i|G_1) = \begin{pmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{pmatrix} \quad (i = 2, 3) \tag{2}$$

$$p(X_i|G_i = 1) = \mathcal{N}(X_i|\mu = 55, \sigma^2 = 10) \quad (i = 1, 2, 3) \tag{3}$$

$$p(X_i|G_i = 2) = \mathcal{N}(X_i|\mu = 65, \sigma^2 = 10) \quad (i = 1, 2, 3) \tag{4}$$

1. What is the **Markov blanket** of G_3?
2. Suppose you only observe $X_2 = 50$. What is the posterior belief on G_1, i.e., $P(G_1|X_2 = 50)$?
3. Now suppose you observe both $X_2 = 50$ and $X_3 = 50$. What is $P(G_1|X_2, X_3)$? Explain your answer intuitively.

1.2 Conditional Random Fields

Conditional Random Fields have been successfully applied in many structured prediction problems. For example, in text analysis, we may be interested in finding the noun phrases (NPs) in a sentence. To illustrate, the NPs are underlined in the following sentence:

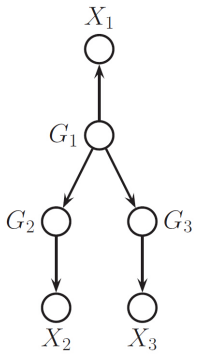

Figure 1: A simple DAG representing inherited diseases

I am Grandalf the White, and I come back to you now at the turn of the tide. - Grandalf, *The Lord of the Rings*.

Let $x_i \in \Sigma$ (the vocabulary) be the tokens in the sentence, and $y_i \in \mathcal{L} = \{B, I, O\}$ be the labels where B is the beginning of an NP, I is an intermediate token in NP, and O stands for others. We would like to have the following labels:

I [B] am [O] Gandalf [B] the [I] White [I], and [O] I [B] come [O] back [O] to [O] you [B] now [O] at [O] the [B] turn [I] of [I] the [I] tide [I].

We consider the following parametric form of CRF:

$$p(\mathbf{y}|\mathbf{x};\mathbf{w}) = \frac{1}{Z(\mathbf{x};\mathbf{w})} \exp\left\{\mathbf{w}^\top \sum_{i=2}^{n} \mathbf{f}(\mathbf{x}, y_i, y_{i-1})\right\}, \tag{5}$$

where

$$Z(\mathbf{x};\mathbf{w}) = \sum_{\mathbf{y}' \in \mathcal{L}^n} \exp\left\{\mathbf{w}^\top \sum_{i=2}^{n} \mathbf{f}(\mathbf{x}, y'_i, y'_{i-1})\right\}. \tag{6}$$

Here i indexes the tokens in sentence $\mathbf{x}$, and $\mathbf{w} \in \mathbb{R}^d$ is the parameter vector, $\mathbf{f} : \Sigma \times \mathcal{L} \times \mathcal{L} \to \mathbb{R}^d$ is the d-dimensional feature vector function, and n is the number of words in the sentences.

1. Draw the **undirected graph** and the **factor graph** of the CRF introduced above.

2. Let $n = 5$, draw the **junction tree** of the above CRF. Answer $p(y_3|\mathbf{x};\mathbf{w})$ by running the **sum-product algorithm** on the junction tree.

3. To learn the parameters of the CRF, we maximize the conditional log-likelihood of training data $\mathcal{D} = \left\{(\mathbf{x}^i, \mathbf{y}^i)\right\}_{i=1}^N$ over parameters $\mathbf{w}$ using gradient descent:

$$\begin{aligned}\mathcal{L}(\mathbf{w}) &= \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}} \log p(\mathbf{y}|\mathbf{x};\mathbf{w}) \\ &= \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}} \left\{ \mathbf{w}^\top \sum_{i=2}^{n} \mathbf{f}(\mathbf{x}, y_i, y_{i-1}) - \log Z(\mathbf{x};\mathbf{w}) \right\}. \end{aligned} \tag{7}$$

Show that

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}} \sum_{i=2}^{n} \left\{ \mathbf{f}(\mathbf{x}, y_i, y_{i-1}) - \mathbb{E}_{p(\mathbf{y}'|\mathbf{x};\mathbf{w})} \left[\mathbf{f}(\mathbf{x}, y'_i, y'_{i-1}) \right] \right\}. \tag{8}$$

4. Give a $O(n)$-time **belief propagation** algorithm on the junction tree (or the factor graph) that computes

$$\sum_{i=2}^{n} \mathbb{E}_{p(\mathbf{y}'|\mathbf{x};\mathbf{w})} \left[\mathbf{f}(\mathbf{x}, y'_i, y'_{i-1}) \right]$$

for a training instance $(\mathbf{x}, \mathbf{y})$. You should provide sufficient details for the algorithm and messages.

2 Deep Generative Models: Class-conditioned VAE

Consider a class-conditioned VAE for generating MNIST digits (`http://yann.lecun.com/exdb/mnist/` , should be **binarized** before training):

$$y \sim \text{Discrete}(\boldsymbol{\pi}) \tag{9}$$

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d) \tag{10}$$

$$\mathbf{x}|\mathbf{z} \sim \text{Bernoulli}(\text{NN}_\theta(y, \mathbf{z})) \tag{11}$$

where $\mathbf{z}$, y and $\mathbf{x}$ are random variables. y denotes the class (label) of the digit. Discrete$(\cdot)$ is a discrete distribution on $\{1, 2, \cdots, K\}$, where K is the number of classes, and $K = 10$ in this case. $\mathbb{P}(y = j) = \pi_j$. $\mathbf{z} \in \mathbb{R}^d$ is the latent representation as in the original VAE. $\mathbf{x} \in \{0, 1\}^{784}$ denotes the image. $\text{NN}_\theta(\cdot)$ is a mapping (neural network) from the concatenation of y (use the one-hot representation) and $\mathbf{z}$ to $\mathbf{x}$.

In the problem we fix $d = 40$, $\pi_j = \frac{1}{K}, \forall j = 1, \ldots, K$. Given the training set of MNIST images $\mathcal{D} = \{(\mathbf{x}^i, y^i)\}_{i=1}^N$, You need to do maximum likelihood learning of the network parameters:

$$\max_\theta \log p(\mathcal{D}) \tag{12}$$

Preparation

- Case if you are not familiar with python/numpy, please work through the tutorial to get started: `http://cs231n.github.io/python-numpy-tutorial/`
- Case if you are not familiar with Tensorflow, learn it by these tutorials:
  - `https://www.tensorflow.org/programmers_guide/low_level_intro`
  - `https://www.tensorflow.org/programmers_guide/tensors`
  - `https://www.tensorflow.org/programmers_guide/variables`
  - `https://www.tensorflow.org/programmers_guide/graphs`
  - `https://www.tensorflow.org/tutorials/layers`
- To get started with ZhuSuan, follow this tutorial on variational autoencoders: `http://zhusuan.readthedocs.io/en/latest/tutorials/vae.html`. Then learn the basic concepts through `http://zhusuan.readthedocs.io/en/latest/tutorials/concepts.html`

Requirements

1. Following the variational Bayes algorithm of the original VAE, derive the algorithm for this class-conditioned variant.

   Hint: You need to design the variational distribution and write down the variational lower bound.

2. Implement the algorithm with ZhuSuan, and train the model on the whole training set of MNIST.

3. Visualize generations of your learned model. Set y observed as $\{1, 2, \cdots, K\}$, and generate multiple $\mathbf{x}$s for each y using your learned model.

*4. (Optional) Can you get similar results with only 10 labeled training data for a single class? Think about how to use unlabeled data to help the training and implement your algorithm.