10807, Assignment 2

Out 10/12/2016; Due 23:59pm 10/26/2016.

Write up

Hand in answers to all the questions in the parts above. For the last question, the goal of your write-up is to document the experiments you have done and your main findings. So be sure to explain the results. Be concise and up to the point – do not write long paragraphs, vaguely explaining results.

- The answers to all the questions should be in pdf form. Please use latex. You can find style files on the course webpage. Do not forger to include you name, your andrew ID, and your e-mail.
- Please include a README file with instructions on how to execute your code. Your code should implement a backpropagation algorithm with cross-entropy error function for one-layer and two-layer neural networks.
- Please submit your write-up in `pdf` format, and package your code in `A2-*yourandrewid*.[zip|tar.gz]`.
- Submit your write-up and code to gradescope.

Problem 1 (10 pts)

Consider a regression problem involving multiple target variables in which it is assumed that the distribution of the targets, conditioned on the input vector $\mathbf{x}$, is a Gaussian of the form:

$$p(\mathbf{t}|\mathbf{x}, W) = \mathcal{N}(\mathbf{t}|y(\mathbf{x}, W), \mathbf{\Sigma}),$$

where $y(\mathbf{x}, W)$ is the output of a neural network with input vector $\mathbf{x}$ and weight vector W, and $\mathbf{\Sigma}$ is the covariance of the assumed Gaussian noise on the targets. Given a set of independent observations of $\mathbf{x}$ and $\mathbf{t}$, write down the error function that must be minimized in order to find the maximum likelihood solution for W, if we assume that $\mathbf{\Sigma}$ is fixed and known. Now assume that $\mathbf{\Sigma}$ is also to be determined from the data, and write down an expression for the maximum likelihood solution for $\mathbf{\Sigma}$ given the network weights W. Note that the optimizations of W and $\mathbf{\Sigma}$ are now coupled. How would you perform optimization of W and $\mathbf{\Sigma}$.

Problem 2 (10 pts)

We will now explore conditional independence properties of directed models and the concept of "Explaining away". Consider the example of the car fuel system shown in Figure 1, left panel. The random variables are as follows:

- B represents the state of the battery that is either charged $B = 1$ or flat $B = 0$.
- F represents the state of the fuel tank that is either full $F = 1$ or empty $F = 0$.
- G represents the state of the electric fuel gauge that indicates either full $G = 1$ or empty $G = 0$.

The battery is either charged or flat, and independently the fuel tank is either full or empty with prior probabilities:

$$p(B = 1) = 0.9, \quad p(F = 1) = 0.9. \tag{1}$$

Given the state of the fuel tank and the battery, the fuel gauge reads full with probabilities given by:

$$p(G = 1|B = 1, F = 1) = 0.8, \quad p(G = 1|B = 1, F = 0) = 0.2,$$
$$p(G = 1|B = 0, F = 1) = 0.2, \quad p(G = 1|B = 0, F = 0) = 0.1.$$

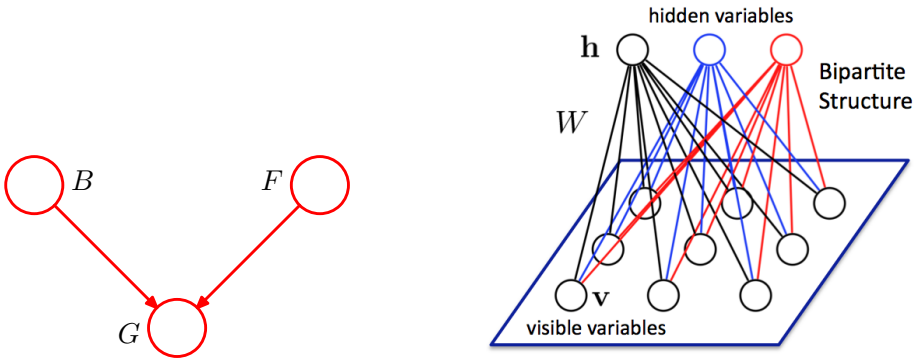

Figure 1: **Left:** Example of the car fuel system. **Right:** An example of undirected graphical model called a Restricted Boltzmann Machine.

Now suppose that instead of observing the state of the fuel gauge G directly, the gauge is seen by the driver D who reports the reading of the gauge. The report is either that the gauge shows full $D = 1$ or that it shows empty $D = 0$. Our driver is unreliable as expressed through the following probabilities:

$$p(D = 1|G = 1) = 0.9, \quad p(D = 0|G = 0) = 0.9. \tag{2}$$

Suppose that the driver tells us that the fuel gauge shows empty, i.e. we observe $D = 0$.

- Draw the new graphical model with D
- Compute the probability that the fuel tank is empty: $p(F = 0|D = 0)$.
- Similarly, compute the corresponding probability given also the observation that the battery is flat: $p(F = 0|D = 0, B = 0)$.
- Discuss the intuition behind this result and how it is related to Figure 1, left panel.

Problem 3 (10pts)

In class, we considered a bipartite undirected graphical model, called Restricted Boltzmann Machine (RBM), shown in Figure 1. An RBM contains a set of observed and hidden random variables, where binary visible variables $v_i \in \{0, 1\}$, $i = 1, ..., D$, are connected to binary hidden variables $h_j \in \{0, 1\}$, $j = 1, ..., P$. The energy of the joint configuration is given by:

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i=1}^{D}\sum_{j=1}^{P} W_{ij} v_i h_j - \sum_{i=1}^{D} v_i b_i - \sum_{j=1}^{P} h_j a_j. \tag{3}$$

where $\theta = \{W, \mathbf{a}, \mathbf{b}\}$ denote model parameters, with $\mathbf{a}$ and $\mathbf{b}$ representing the bias terms, and $\mathbf{v} = \{v_1, ..., v_D\}$, $\mathbf{h} = \{h_1, ..., h_P\}$. The probability of the joint configuration is given by the Boltzmann distribution:

$$P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}} \exp(-E(\mathbf{v}, \mathbf{h}; \theta)), \tag{4}$$

where $\mathcal{Z} = \sum_{\mathbf{v},\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}; \theta))$ denotes the normalizing constant. Derive that conditioned on observed variables $\mathbf{v}$, the distribution over the hidden variables factorizes (so the hidden variables become independent conditioned on $\mathbf{v}$):

$$P_\theta(h_1, ..., h_P|\mathbf{v}) = \prod_{j=1}^{P} P_\theta(h_j|\mathbf{v}), \tag{5}$$

where

$$P_\theta(h_j = 1|\mathbf{v}) = \frac{1}{1 + \exp(-\sum_{i=1}^{D} W_{ij} v_i - a_j)}. \tag{6}$$

Note: inferring the states of the hidden variables conditioned on the observed data is easy. This represents a major advantage of this model, which is successfully used in many domains, ranging from collaborative filtering to speech recognition.

Problem 4 (10 pts)

Consider the use of iterated conditional modes (ICM) to minimize the energy function (that we considered in class for image denoising example) given by:

$$E(\mathbf{x}, \mathbf{y}) = h \sum_i x_i - \beta \sum_{i,j} x_i x_j - \eta \sum_i x_i y_i, \tag{7}$$

where $x_i \in \{-1, 1\}$ is a binary variable denoting the state of pixel i in the unknown noise-free image, i and j are indices of neighboring pixels, and $y_i \in \{-1, 1\}$ denotes the corresponding value of pixel i in the observed noisy image. The joint distribution is defined as:

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{\mathcal{Z}} \exp\left(-E(\mathbf{x}, \mathbf{y})\right) \tag{8}$$

- (5 pts) Write down an expression for the difference in the values of the energy associated with the two states of a particular variable x_j, with all other variables held fixed, and show that it depends only on quantities that are local to x_j in the graph.
- (5 pts) Consider a particular case of the energy function above in which the coefficients $\beta = h = 0$. Show that the most probable configuration of the latent variables is given by $x_i = y_i$ for all i.

Problem 5 (60 pts)

For this question you will write your own implementation of Contrastive Divergence algorithm for training RBMs and compare it to Autoencoder model. Please do not use any toolboxes. We recommend that you use MATLAB or Python, but you are welcome to use any other programming language if you wish.

The goal is to build a generative model of images of 10 handwritten digits of "zero", "one",..., "nine". The images are 28 by 28 in size (MNIST dataset), which we will be represented as a vector $\mathbf{x}$ of dimension 784 by listing all the pixel values in raster scan order. The labels t are 0,1,2,...,9 corresponding to 10 classes as written in the image. There are 3000 training cases, containing 300 examples of each of 10 classes, 1000 validation (100 examples of each of 10 classes), and 3000 test cases (300 examples of each of 10 classes). they can be found in the file digitstrain.txt, digitsvalid.txt and digitstest.txt:
http://www.cs.cmu.edu/∼rsalakhu/20807_2015/assignments.html

Format of the data: digitstrain.txt contains 3000 lines. Each line contains 785 numbers (comma delimited): the first 784 real-valued numbers correspond to the 784 pixel values, and the last number denotes the class label: 0 corresponds to digit 0, 1 corresponds to digit 1, etc. digitsvalid.txt and digitstest.txt contain 1000 and 3000 lines and use the same format as above.

Contrastive Divergence (CD), Autoencoders

Implement CD algorithm for training an RBM model. Implement both autoencoder and denoising autoencoder models.

a) Basic generalization [20 points]

Train an RBM model with 100 hidden units, starting with CD with $k = 1$ step. You should use initialization scheme discussed in class as well as choose a reasonable learning rate (e.g. 0.1 or 0.01). Train the model repeatedly (more than 5 times) using different random seeds, so that each time, you start with a slightly different initialization of the weights.

As a proxy for monitoring the progress, plot the average training cross-entropy reconstruction error on the y-axis vs. the epoch number (x-axis). On the same figure, plot the average validation cross-entropy error function.

Examine the plots of training error and validation error. How does the network's performance differ on the training set versus the validation set during learning? Visualize the learned W as 100 28x28 images (plot all filters as one image, as we have seen in class). Do the learned features exhibit any structure?

b) Number of CD steps [5 points]

Try learning an RBM model with CD with k=5, k=10, and k=20 steps. Describe the effect of this modification on the convergence properties, and the generalization of the network. Does more CD steps always result in better reconstruction error? why or why not?

c) Sampling from the RBM model [5 points]

To qualitatively test model performance, initialize 100 Gibbs chains with random configuration of the visible variables, and run Gibbs sampler for 1000 steps. Display 100 sampled images. Do they look like handwritten digits?

d) Unsupervised Learning as pretraining [5 points]

Use learned weights to initialize a 1-layer neural network with 100 hidden units. Train the resulting neural network to classify 10 digits, as done in the first assignment. Does this pretraining help compared to training the same neural network initialized with random weights in terms of classification accuracy? Describe your findings.

e) Autoencoder [5 points]

Instead of training an RBM model, train an autoencoder model with 100 sigmoid hidden units. Does the model learn useful filters? Visualize learned weights. Similar to RBMs, use learned autoencoder weights to initialize a 1-layer neural network with 100 hidden units. Train the resulting neural network to classify 10 digits. How does it compare to pretraining with RBMs?

f) Denoising Autoencoder [10 points]

Train a denosing autoencoder by using drop-out on the inputs. Visualize learned weights. Use learned weights to initialize a 1-layer neural network with 100 hidden units. Train the resulting neural network to classify 10 digits. How does it compare to pretrainig with standard autoencoders and RBMs in terms of classification accuracy?

g) Number of hidden units [10 points]

Try training RBMs, autoencoders and denoising autoencoders with 50, 100, 200, and 500 hidden units. Describe the effect of this modification on the convergence properties, and the generalization of the network.