HOMEWORK 2

CMU 10-707: DEEP LEARNING (FALL 2017)
https://piazza.com/cmu/fall2017/10707
OUT: Oct 9, 2017
DUE: Oct 23, 2017 11:59 pm
TAs: Otilia Stretcu, Shunyuan Zhang

Problem 1 (10 pts)

Problem 2 (10 pts)

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|\text{pa}_k) \tag{1}$$

Problem 3 (10pts)

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i=1}^{D}\sum_{j=1}^{P} W_{ij} v_i h_j - \sum_{i=1}^{D} v_i b_i - \sum_{j=1}^{P} h_j a_j. \tag{2}$$

$$P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}} \exp(-E(\mathbf{v}, \mathbf{h}; \theta)), \tag{3}$$

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

Proof Here

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

where

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

Problem 4 (10 pts)

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

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

- (5 pts) Expression here
- (5 pts) Proof Here

Problem 5 (60 pts)

For this question you will write your own implementation of the Contrastive Divergence algorithm for training RBMs, and compare it to an 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/10707/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 the CD algorithm for training an RBM model. Implement both an autoencoder and a denoising autoencoder model.

a) Basic generalization [20 points]

Train an RBM model with 100 hidden units, starting with CD with $k = 1$ step. For initialization use samples from a normal distribution with mean 0 and standard deviation 0.1. Choose a reasonable learning rate (e.g. 0.1 or 0.01). Use your own judgement to decide on the stoping criteria (i.e number of epochs or convergence criteria).

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.

c) Sampling from the RBM model [5 points]

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

d) Unsupervised Learning as pretraining [5 points]

Use the weights you learned in question **a)** to initialize a 1-layer neural network with 100 hidden units. Train the resulting neural network to classify the 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. For initializing the autoencoder, use samples from a normal distribution with mean 0 and standard deviation 0.1. Does the model learn useful filters? Visualize the learned weights. Similar to RBMs, use the learned autoencoder weights to initialize a 1-layer neural network with 100 hidden units. Train the resulting neural network to classify the 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. We recommend a drop-out rate of 10%, but feel free to explore other rates. Visualize the learned weights. Use the learned weights to initialize a 1-layer neural network with 100 hidden units. Train the resulting neural network to classify the 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.