Math 6365 spring 2018

Midterm2 : Take Home Exam Due Date Sunday April 22nd

This take home exam can be prepared and presented by groups of up to 3 graduate students.

Part 1 and Part 2 should be typed. Copy of the codes to be provided at the end of the report.

Part 1 Report on GBMs and RBMs

Write a report on Generic Boltzmann Machines and Restricted Boltzmann Machines.

This Report should cover the main contents of the lectures given in class on GBMs and RBMs , plus the main aspects of the Hinton paper on RBMs sent to the class.

Indication : the report should roughly dedicate 3 to 4 pages per class lecture as well as for the Hinton paper.

Part 2 Implementation and analysis of RBM auto-encoders.

2.1 Select a data set where the descriptors can be binarized in a fairly natural way without generating a huge increase in the number of binary uits necessary to encode the input vectors. Two examples were given in class : black and white images , and daily or intraday prices for 500 to 1000 stocks, with binary encoding focused on price increases and decreases. Some of the data sets used in HW3 or HW4 are also possible for the midterm.

2.2. Present briefly the data set and describe precisely how input vectors are systematically encoded into binary vectors Call N the number of cases in the training set. Call "r" the dimension of the binarized input vectors X(1)...,X(N), which is also the dimension of the binarized output vectors Y(1) ...Y(N) of the autoencoder

2.3. Implement an RBM to generate several RBM auto-encoders of your data. Note that you are NOT requested at all to use the RBM auto encoder to implement a classification task

(a) Typically RBM training implements a gradient ascent based on comparing co-activities between pairs of connected nodes. Indicate which RBM training software you are using and what are the main options offered by the software ; indicate what type of RBM training algorithm is implemented by this software, and in particular if training with sparsity is an option.

(b) Run training experiments with various sizes h for the hidden layer, and various weights initialization, and various learning rates. Cases should be used in small successive batches during batch learning (for instance batches of size 50).

(c) For each experiment, and after each epoch EP(k), compute the current gradient norm || G(n) ||, as well as the current mean and standard deviations of the vector of weights. Plot the evolution of these 3 quantities versus "n". Compute and plot the histogram of weights after 10 epochs and at the end of the learning experiment. Interpret these 4 plots for each experiment.

(d) Explain precisely which stopping rule is implemented by your learning software.

(e) Performance analysis :

After each training experiment is completed (typically because weights have approximately stabilized), FIX THE vector W of weights obtained at the end of the training experiment. The dimension of W should be (2 r h + h + r) once the (h+r) thresholds are included in W.

Then for each binarized input vector X(n) , proceed as follows:

encode X(n) on the input layer , and initialize all other nodes at 0

implement (successively) the Boltzmann stochastic updates of all the "h" hidden nodes,

implement (successively) the Boltzmann stochastic updates of all the "r" output nodes.

Keeping X(n) fixed, repeat 20 times the preceding "(h+r)" stochastic updates.

This generates 20 random versions of the binary output vector Y(n). Compute the vector average y(n) of these 20 random output vectors. Consider y(n) as the average reconstruction of the input vector X(n).

Estimate the reconstruction error for X(n) by $MSE(n) = (1/r) \; || y(n) - X(n) ||^2$

Repeat this procedure N times, for the inputs X(1) X(N), and then compute the global mean squared error of reconstruction for your auto-encoder by $MSE = (1/N) [MSE(1) + ... + MSE(N)]$

(f) Use the values of MSE to compare the reconstruction performance of your various trained auto-encoders