Deep Learning DS-GA 1008.

Lab Session Agenda - 01/31

Agenda

- *Grader introduction*
- *PyTorch*

Attendees

DSGA 1008 enrolled students.

Assignment logistic

- Teams: 2-3. Restricted #members.
    - Send Hao Liu (hl2514@nyu.edu) an email, following the format highlighted from the course website. "[DS-GA-1008 YOUR_TEAM_NAME]"
    - Who has sent me email, but not following the format, please resent to Hao Liu.
    - **DEADLINE: Feb 5th.**
    - Once again, the unregistered students won't be able to compete on Kaggle.
- Other preparation: LaTeX is a must for the submission.

PyTorch

- Installation from the documentation of the page
    - Anaconda is recommended
    - Pip or from source is also valid
- Go through the first part of : https://github.com/pytorch/tutorials/blob/master/Deep%20Learning%20with%20PyTorch.ipynb

- Reference doc: http://pytorch.org/docs/tensors.html
- Exercise 1:
    - Initialize random tensors a, b, c of size [2,3], [2,3], [3,2,3].
    - Fill tensor a with all 10
    - Fill tensor b with elements being sampled from the normal distribution
    - Point-wise multiply a with b, and put the result into tensor b
    - Print the mean and std of the elements of b
    - Fill tensor c with elements samples from the uniform distribution U(-1,1)
    - Transpose the second and third dimension of tensor c, and put the result into tensor c itself (in-place).
    - Show the contiguity property of the tensors
    - Print the second column of the third dimension of tensor c (note zero-indexed)
    - Perform operation a+b+c (note the broadcasting)
    - In-place storing the result into tensor c?
- Exercise 2 (image):
    - Image is naturally a 3D tensor: RGB channels + spatial dimension
    - A batch of images amounts to be 4D tensor: [batchSize x nchannels x width x height]
    - Download MNIST training dataset using torchvision
    - Show the size of train set data (just the size, don't dump them all out..)
    - Show the size of train set labels
    - Access the first image of MNIST train set (you may use any method to display it, such as matplotlib, where you might need to cast it into numpy)
    - Make a grid of a batch of train set data
        - Declare a DataLoader first, with size 4
        - Use the torchvision.make_grid
    - Verify that the batch of samples are consistent with the batch of labels (by printing is enough)
    - What about a different size of batch, say 32?
- Exercise 3 (Classification Neural network):
    - Declare a similar network defined here: https://github.com/pytorch/tutorials/blob/master/Deep%20Learning%20with%20PyTorch.ipynb
        - This one is used for CIFAR32, which is RGB and of size 32x32
        - You need to find a way to get it to work on MNIST, which is grayscale and of size 28x28
    - Declare a loss functional using CrossEntropy used for classification
    - Declare an optimizer using ADAM
    - Prepare a batch of samples from MNIST
    - Zeroize the gradient of the network
    - Run forward pass
    - Print the loss
    - Run backward pass

- Print the mean, max and min value of gradient module by module
- Let the optimizer act a step