Linear Optimization Assignment #1

Due: Sunday, April 1, 24:00:00

Instruction: Write a report and complete code. Download the code from ftp(10.13.71.168). Upload them to ftp(10.13.72.84).

- Upload:
  - Address: 10.13.72.84
  - Username: opt; Passwd: opt18; Port: 21
- Download:
  - Address: 10.13.71.168
  - Username: opt; Passwd: opt18; Port: 21

Problem 1

A McCulloch-Pitts (M-P) neuron accepts bipolar input $x_i \in \{-1, 1\}$ and gives $y = \text{sgn}(\sum_{i=1}^{m} w_i x_i + b)$. Give weights and bias for a M-P neuron with inputs $x, y, z \in \{-1, 1\}$ and whose output is z if $x = -1$ and $y = 1$, and is -1 otherwise.

Problem 2: Gradient Descent

SGD has trouble navigating ravines, i.e. areas where the surface curves much more steeply in one dimension than in another, which are common around local optima. Momentum is a method that helps accelerate SGD in the relevant direction and dampens oscillations as can be seen in fig 1.

$$\begin{aligned} v_t &= \mu v_{t-1} + \varepsilon \nabla_u f(u) \\ u_{t+1} &= u_t - v_t \end{aligned}$$

a). Execute four iterations of gradient descent with momentum to find the minimum of the function $f(u) = \frac{u^3}{3} + 50u^2 - 100u - 30$. Start with $u_0 = 20$ and $v_0 = 0$, use a learning rate that is set to $\varepsilon = 0.01$, and parameter μ set to 0.1.

b). Evaluate the benefit of using a momentum for the task in a). by comparing your findings to the vanilla gradient descent method.

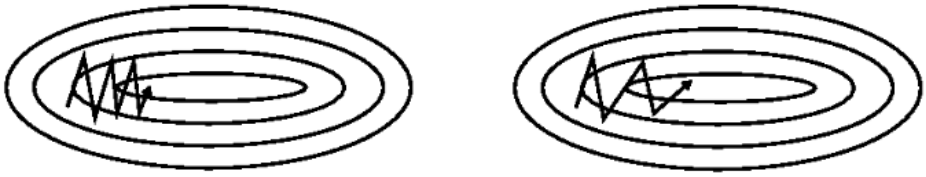

Figure 1: Left: SGD without momentum; Right: SGD with momentum

Problem 3: Forward-Backward-Pass

Examine the multi-layer perceptron given in fig 2.

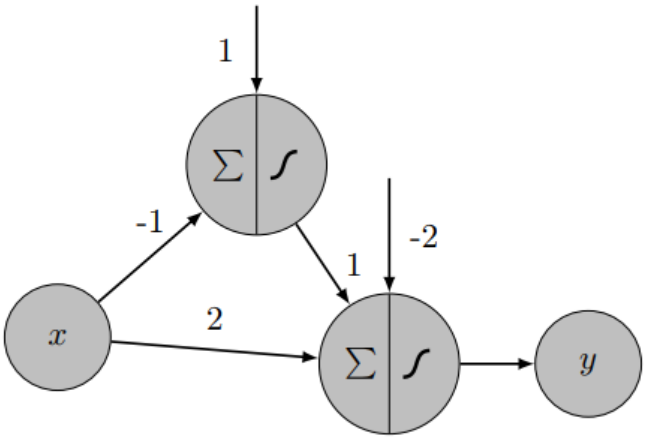

Figure 2: MLP with two logistic units

a). Both neurons use the logistic activation function $f(u) = \frac{1}{1+e^{-u}}$ The network has a single input variable x and one output variable y. Calculate the output of both neurons and the error made by the MLP when applying a pattern with $x = 0$ and target value 0.5.

b). Calculate the partial derivatives of the error with respect to the weights for the pattern used in task a).

Problem 4: Multi-Layer Perceptron Architecture

Now, consider the network structure of a multi-layer perceptron with 10 neurons given in fig 3. Each circle denotes a neuron, the arrows denote connections between neurons.

a). Which of the neurons are input neurons, which ones are output neurons?

b). How many layers does this MLP have? Which neurons belong to which layer?

c). Assume we are applying a pattern to the MLP. Give an order in which the neuron activations a_i can be calculated.

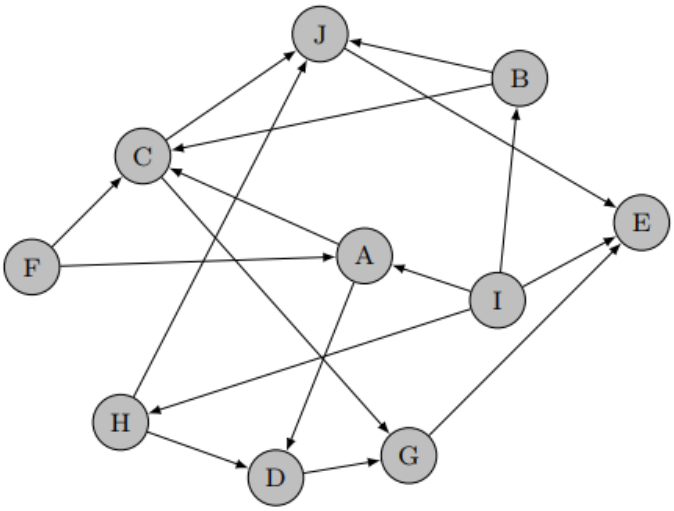

Figure 3: MLP with ten neurons

Problem 5: Multi-Layer Perceptrons

Which of the functions given by the plots in fig 4 can be implemented by multi-layer perceptrons? The MLP should only contain neurons with logistic activation functions. (Note: The weights of the networks must be finite numbers.)

Problem 6: MLP and BackPropogation

- Download "hw2.zip" from ftp(10.13.71.168), The code pass test under Python 2.7.14 and Python 3.6.3. Complete code surrounded by "TODO" in "net.py" and "opt.py". For example,

```
############################################################################
# TODO: Implement the affine forward pass. Store the result in out. You   #
# will need to reshape the input into rows.                                #
############################################################################


############################################################################
#                             END OF YOUR CODE                             #
############################################################################
```

- Run "python main_check.py" and paste the settings (*e.g.* random seed, loss function name) and the results to your report. Make sure your code pass gradient check, *i.e.*, relative error between Numerical gradient and analytic gradient is small (*e.g.* smaller than 1e-7). We use centered formula $\frac{\mathrm{d}f(x)}{\mathrm{d}x} \approx \frac{f(x+h)-f(x)}{h}$ to compute numerical gradient since it has an error on order of $O(h)$ and use relative error as metric.

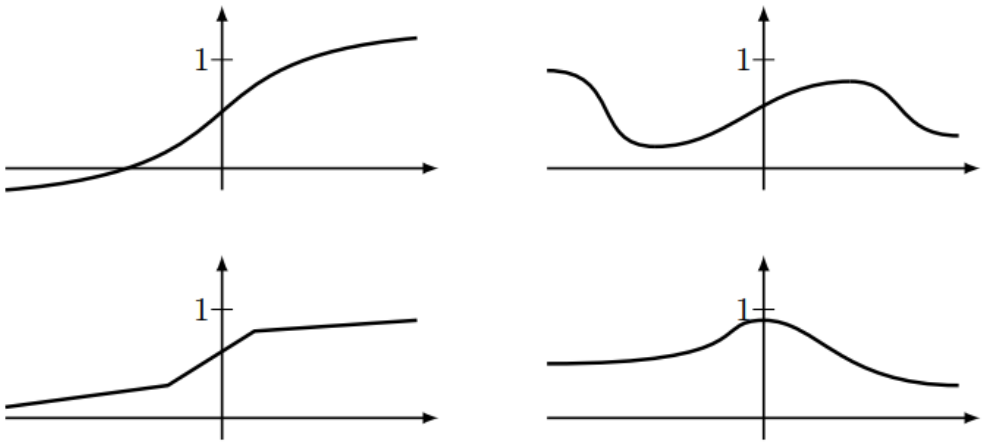

Figure 4: Functions to be realized by MLPs

- Run "python main_train.py" to train a model on fake data.
- Do something extra surrounding the topics in this assignment, using the code you developed. For example, is there some other interesting question we could have asked? Is there any insightful visualization you can plot? We did not use large cifar10 data. You can select some samples (*e.g.* 50 or 500) in cifar10 or use the dataset in Problem 7 to do real case analysis.
- You do not necessarily strictly use the framework provided by teacher assistant. You can use any language and framework and write from scratch. Even tensorflow/pytorch with autograd is permitted, if you have time. Just make sure implement MLP by yourself and then do similar experiments, *i.e.*, do not use any function like MLP or LinearRegression.
- Finally, upload to ftp(10.13.72.84) before the deadline(Sunday, April 1, 24:00:00).

Problem 7: Predict House Prices

Given 79 explanatory variables describing (almost) every aspect of residential homes, such as the size and the location of the house, please predict the final price of each home.

- Download "hw2.zip" from ftp(10.13.71.168). The dataset is cleaned to become simple train/val format. You can try to modify the process of data clean.
- Make sure implement the regression model by yourself and do not use something like Linear-Regression. Regression is similar to MLP, which have been implemented in Problem 6.
- Encourage to do model selection. For example, try lasso regression, huber loss or learning rate decay and select best model by cross validation.
- You do not necessarily strictly use the framework provided by teacher assistant. You can use any language and write from scratch. Just make sure implement regression model by yourself and then do similar experiments.

- Finally, upload to ftp(10.13.72.84) before the deadline(Sunday, April 1, 24:00:00).