Project 2 - How to Better Train a Neural Network (Due 09/28)

Objectives:

The objective of this project is to obtain in-depth understanding on practical issues in training a neural network, including CNN. MNIST will be used as the benchmark.

Requirements:

- Task 1: Based on Nielsen's original BPNN code, "network.py", experiment the following
  - Task 1.1: Effect of cost function using the default network structure of [784,10] with no hidden layer
    - Implement quadratic cost function with sigmoid as activation function and plot the convergence curve
    - Implement cross entropy as cost function with sigmoid as activation function and plot the convergence curve
    - Implement log-likelihood as cost function with softmax as output layer activation function and plot the convergence curve
    - In the report, need to comment on the behavior of different cost function in resolving the learning slowdown problem. Also explain what is learning slowdown problem.
  - Task 1.2: Effect of regularization using the default network structure of [784,10] with no hidden layer, and cost entropy as cost function
    - Add L2-normalization in the cost function and plot the convergence curve.
    - Add L1-normalization in the cost function and plot the convergence curve.
    - Base on the L1-normalization, expand training dataset using affine transformations, i.e., scaling, rotation, and translation. Plot the convergence curve.
    - In the report, need to comment on the problem of overfitting, and how regularization terms can potentially solve the overfitting problem. Also comment on the difference between L1 and L2 normalization. Finally, comment on the two aspects of improving accuracy, i.e., improve the algorithm or improve the training set.
  - Task 1.3: Effect of hidden layers based on cross-entropy with L1-normalization and expanded training set. (No dropout needed.)
    - Add one hidden layer with 30 nodes. Plot the convergence curve.
    - Add two hidden layers with 30 nodes each. Plot the convergence curve and the change rate of each node in the hidden layers. Use the partial derivative of the cost function with respect to the bias as an indicator of the change rate of each node.
    - (692 only, bonus for 599) Based on your L1-normalization implementation and the expanded training set, add dropout. Experiment convergence curve with different percentages of dropout.
    - In the report, need to comment on the unstable gradient problem.
    - (692 only, bonus for 599) In the report, comment on the difference between normalization-based regularization and dropout-based regularization.
- Task 2: CNN (LeNet) with TensorFlow
  - Study the sample code from Liu Liu on how to train a LeNet-5 to recognize MNIST digits.
  - Modify hyperparameters to get to the best performance you can achieve.
  - In the report, provide a plot of accuracy improvement using the previously mentioned techniques. Also plot the convergence curve for LeNet-5. Comment on why you think LeNet-5 further improves the accuracy is any at all. And if it doesn't, why not?

  Note that this is an open-ended problem. We'll have a leaderboard updated frequently to report the best in class.

Report

Please submit the report (in .pdf) and source code (in .zip) through Canvas before midnight on the due date.