EECS 498/598 Deep Learning HW1

Fei Yi 53455316 feiyi@umich.edu

1. Logistic regression and beyond: Binary classification with a sigmoid output layer
    a. No hidden units:
        i. Performance: train acc: 0.964000; val acc: 0.952000
        ii. Model: input_dim=20, hidden_dim=None, weight_scale=1e-3, reg=0.2
        iii. Solver: update rule: sgd_momentum, learning_rate=1, lr_decay=0.95, num_epochs=20, batch_size = 50
    b. With hidden units:
        i. Performance: train acc: 0.962000; val_acc: 0.952000
        ii. Model: input_dim=20, hidden_dim=32, weight_scale=1e-3, reg=0.3
        iii. Solver: update rule: sgd_momentum, learning_rate=8e-2, lr_decay=0.9, num_epochs=20, batch_size = 50

2. SVM and beyond: binary classification with a hinge-loss output layer
    a. No hidden units:
        i. Performance: train acc: 0.964000; val_acc: 0.944000
        ii. Model: input_dim=20, hidden_dim=None, weight_scale=1e-3, reg=0.1
        iii. Solver: update rule: sgd_momentum, learning_rate=1, lr_decay=0.95, num_epochs=50, batch_size = 50
    b. With hidden units:
        i. Performance: train acc: 0.968000; val_acc: 0.928000
        ii. Model: input_dim=20, hidden_dim=32, weight_scale=1e-3, reg=0.3
        iii. Solver: update rule: sgd_momentum, learning_rate=1e-1, lr_decay=0.95, num_epochs=50, batch_size = 50

3. Softmax regression and beyond: multi-class classification with a softmax output layer
    a. No hidden units:
        i. Performance: train acc: 0.953000; val_acc: 0.935556
        ii. Model: input_dim=784, hidden_dim=None, weight_scale=1e-3,

reg=0.01

iii. Solver: update rule: rmsprop, learning_rate=5e-5, lr_decay=0.9, num_epochs=50, batch_size = 100

b. With hidden units:

i. Performance: train acc: 0.993000; val_acc: 0.977037

ii. Model: input_dim=784, hidden_dim=64, weight_scale=1e-3, reg=0.01

iii. Solver: update rule: rmsprop, learning_rate=1e-4, lr_decay=0.9, num_epochs=50, batch_size = 100

4. Convolutional Neural Network for multi-class classification

a. No batch normalization and drop out:

i. Performance: train acc: 0.984000; val_acc: 0.976200

ii. Model: input_dim=(1,28,28), num_filters=5, filter_size =7, hidden_dim=16, num_classes=10, weight_scale=1e-3, reg=0

iii. Solver: update rule: sgd_momentum, learning_rate=1e-3, lr_decay=0.9, num_epochs=50, batch_size = 100

b. With batch normalization and drop out:

i. Performance: train acc: 0.914000; val_acc: 0.901200

ii. Model: input_dim=(1,28,28), num_filters=8, filter_size =7, hidden_dim=16, num_classes=10, weight_scale=1e-3, reg=0

iii. Solver: update rule: sgd_momentum, learning_rate=1e-2, lr_decay=0.9, num_epochs=50, batch_size = 100

5. Convolutional Neural Networks for CIFAR10 image classification.

a. Performance: accuracy: 74%

b. Architecture Improvement:

i. Double the filter_num of first two conv layers

ii. Add batch normalization and relu after each conv layers

iii. Add drop out after first fc layer

c. Solver:

i. Learning_rate = 0.001

ii. Batch_size = 100

iii. Criterion = crossentropy loss

iv. Optimizer = adam

6. Short answer questions

a. Sigmoid projects large inputs to 0 or 1, causing its dsigmoid = sigmoid * (1 – sigmoid) to vanish. In DNN, the gradient vanishing is even more severe due to chain rule, resulting in difficulty in training.

b. When perform W in testing, multiply x by p to account for missing activations in training.
c. I will reduce learning rate.