CMPE 258, Deep Learning

Optimization

March 13, 2018

DMH 149A

Taehee Jeong

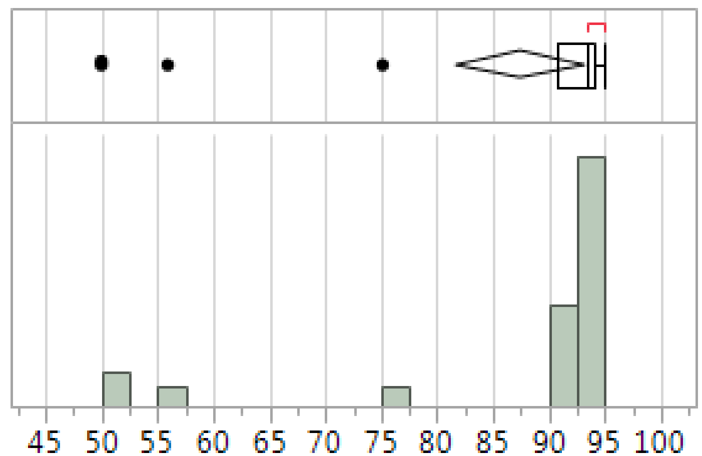
Ph.D., Data Scientist

SJSU SAN JOSÉ STATE UNIVERSITY

Exam_1 (mid term)

Max: 94.86

Median : 93.4



This score is not applied yet extra credit.

Grading policy for Exam_1

If the code can be executable without any extra effort and get reasonable result, I gave point based on the accuracy of the testing data.

If extra effort is needed to get reasonable result (whatever it is), I will take out 5 to 10 points from the accuracy of the testing data.

If the code cannot be executable, I take out 20 to 100 points depending on the error even though it looks reasonable.

Exam_2 (Mid-term)

4/5 for CNN

Assigiment_4

Deep Neural Network with tensorflow
Due day is Sunday, Marth 18th.
There is penalty for late submission or re-submission.

In Assignment_3 and mid-term exam, we used pandas and numpy.

For Assignment_4, we will use Tensorflow.

Please build Deep Neural Network with two hidden layers.

For activation function, please use relu or elu for hidden layers, sigmoid for output layer.

For weight initialization, please use xaiver initialization.

For optimization, please use adam optimization.

For regularization, please use dropout.

As the final output, please plot train accuracy and test accuracy with probability (0.1 ~ 0.9) of dropout.

Group Project

Proposal date

4/12

4/24

Participation credit

Extra credit will be posted soon for participation or contribution in discussion during class or on canvas.

Last lecture

- Regularization in Neural Network
- Activation functions in Neural Network
- Derivatives of activation functions
- Weight initialization
- Batch Normalization

Today’s topics

- Parameter update:

Momentum, Adagrad, Rmsprop, Adam

- Learning rate decay:

1/t, exponential decay, step decay

Parameter update

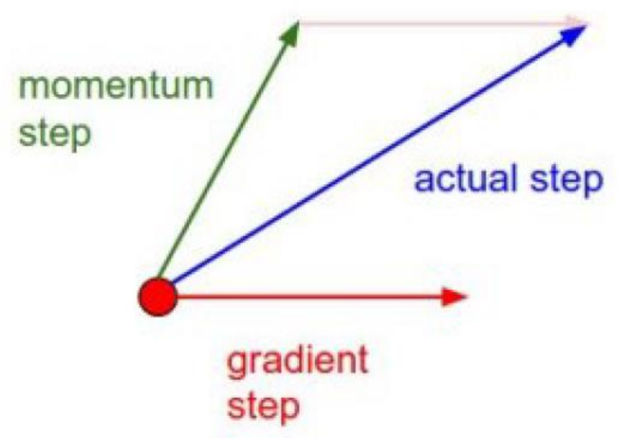
Faster optimization

- Nesterov momentum → not cover
- Second order method: based on Newton’s method, L-BFGS → not cover
- Momentum
- Adagrad
- Rmsprop
- Adam

Momentum update

Gradient descent update

x := x – learning rate*dx



Momentum update

v = β*v – learning rate*dx

x := x + v

v : velocity

β : momentum

SJSU SAN JOSÉ STATE UNIVERSITY

Adagrad

Adaptive learning rate method

v := v + dx**2

x := x – learning_rate *dx / (sqrt(v) + eps)

v : sum of squared gradients

Normalized the parameter update step, element-wise.

Weights with high gradients will have effective learning rate reduces,

Weights with low gradients will have effective learning rate increases.

Gradient descent with momentum

$V_{dW} = 0, V_{db} = 0$

For every iteration t:

Compute V_{dW}, V_{db} using dW and db

$$V_{dw} := \beta_1 \cdot V_{dw} + (1 - \beta_1) \cdot dW$$

$$V_{db} := \beta_1 \cdot V_{db} + (1 - \beta_1) \cdot db$$

$$W := W - \alpha \cdot V_{dw}$$

$$b := b - \alpha \cdot V_{db}$$

RMS prop

Root Mean square prop

$$S_{dW} \neq 0, S_{db} \neq 0, \epsilon = 10^{-8}$$

For every iteration t:

Compute S_{dW}, S_{db} using dW and db

$$S_{dW} := \beta_2 \cdot S_{dW} + (1 - \beta_2) \cdot dW^2$$

$$S_{db} := \beta_2 \cdot S_{db} + (1 - \beta_2) \cdot db^2$$

$$W := W - \alpha \cdot \frac{dW}{\sqrt{S_{dW} + \varepsilon}}$$

$$b := \mathrm{b} - \alpha \cdot \frac{db}{\sqrt{S_{db} + \varepsilon}}$$

Adam Optimization

Adaption moment estimation

$$V_{dW} = 0, V_{db} = 0, S_{dW} = 0, S_{db} = 0, \epsilon = 10^{-8}$$

For every iteration t:

Compute V_{dW}, V_{db} , S_{dW}, S_{db} using dW and db

$$V_{dW} := \beta_1 \cdot V_{dW} + (1 - \beta_1) \cdot dW$$

$$V_{db} := \beta_1 \cdot V_{db} + (1 - \beta_1) \cdot db$$

$$S_{dW} := \beta_2 \cdot S_{dW} + (1 - \beta_2) \cdot dW^2$$

$$S_{db} := \beta_2 \cdot S_{db} + (1 - \beta_2) \cdot db^2$$

$$W := W - \alpha \cdot V_{dW}$$

$$b := b - \alpha \cdot V_{db}$$

Adam Optimization

Bias correction

$$V_{dW}^{cor} = \frac{V_{dW}}{1-\beta_1^t}$$

$$V_{db}^{cor} = \frac{V_{db}}{1-\beta_1^t}$$

$$S_{dW}^{cor} = \frac{S_{dW}}{1-\beta_2^t}$$

$$S_{db}^{cor} = \frac{V_{db}}{1-\beta_2^t}$$

$$W := W - \alpha \cdot \frac{V_{dW}^{cor}}{\sqrt{S_{dW}^{cor}} + \varepsilon}$$

$$b := \mathrm{b} - \alpha \cdot \frac{V_{db}^{cor}}{\sqrt{S_{db}^{cor}} + \varepsilon}$$

Adam optimization

Hyper parameters choices

α : need to be tune

β_1 : 0.9 (momentum)

β_2 : 0.999 (RMS prop)

ε : 10^{-8}

Learning rate decay

Slowing decreasing α

If learning rate is large, parameters decay too aggressively and the parameters are unable to reach the local minimum.

1 epoch : 1 pass through data set, iteration number

- 1/t decay
- Exponentially decay
- Step decay: discrete staircase
- Manual decay

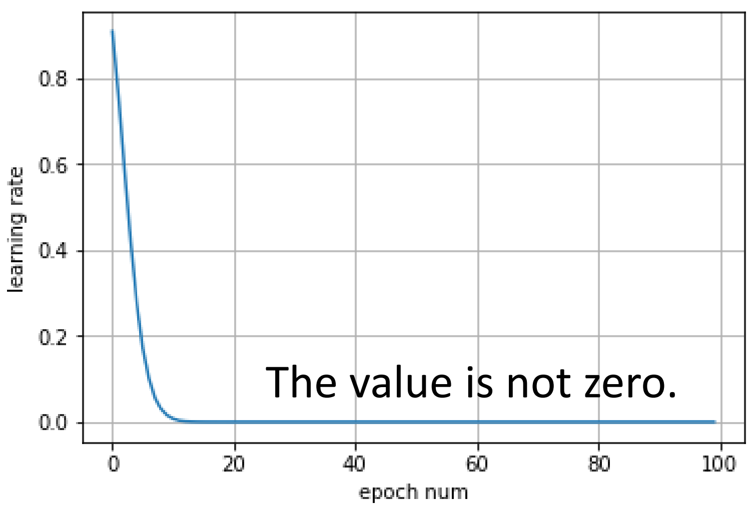
1/t decay

$$\boxed{\alpha = \frac{\alpha_0}{1 + decay - rate \times epoch - num}}$$

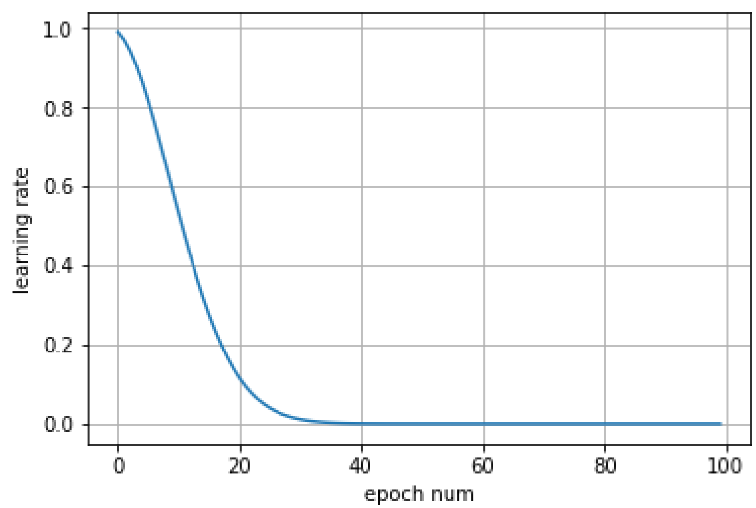
$$\boxed{\alpha = \frac{\alpha_0}{1 + kt}}$$

$\alpha_0 = 1$

decay rate = 0.1



decay rate = 0.01

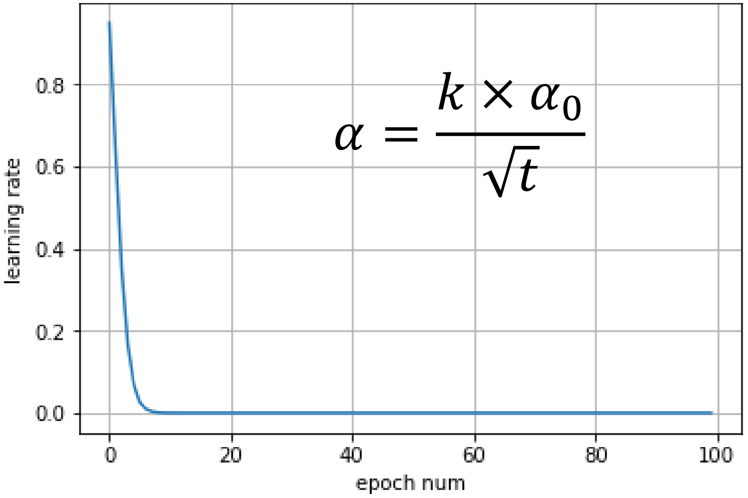


1/t decay

$$\boxed{\alpha = \frac{k \times \alpha_0}{\sqrt{epoch - num}}}$$

$\alpha_0 = 1$

$k = 0.95$

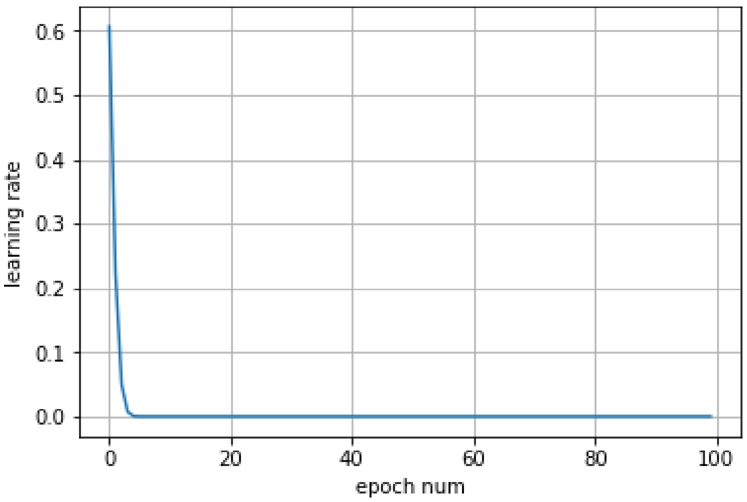


Exponential decay

$$\boxed{\alpha = \alpha_0 \cdot \exp(-kt)}$$

$\alpha_0 = 1$

$k = 0.5$

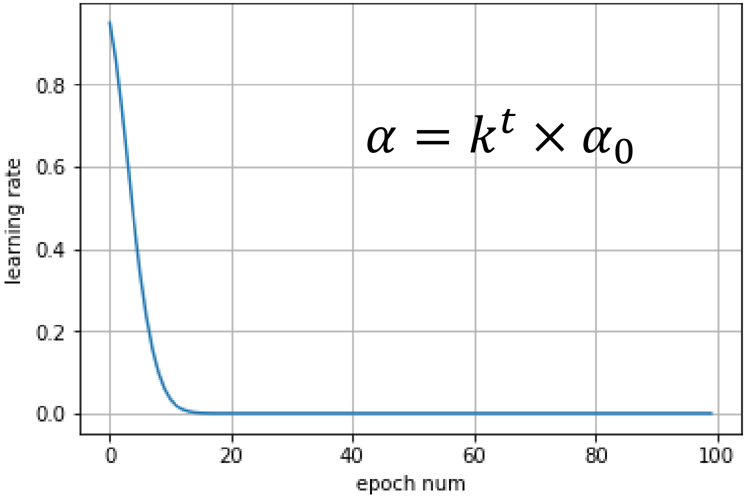


Exponentially decay

$$\boxed{\alpha = k^{epoch\_num} \times \alpha_0}$$

$\alpha_0 = 1$

$k = 0.95$



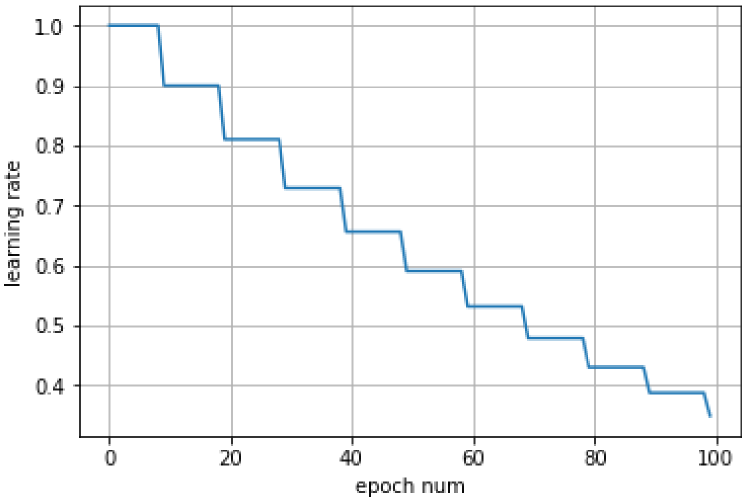
Discrete staircase

$\boxed{\alpha = k \times \alpha \text{ with every few epochs}}$

$\alpha_0 = 1$

k= 0.9

every 10 epochs



Learning rate decay

Annealing learning rate

Hyper parameter k: [0.01, 0.1, 0.5, 0.9, 0.95, 0.99]

$$\alpha = \frac{\alpha_0}{1 + kt}$$

$$\alpha = \frac{k \times \alpha_0}{\sqrt{t}}$$

$$\alpha = \alpha_0 \cdot \exp(-kt)$$

$$\alpha = k^t \times \alpha_0$$

$\alpha = k \times \alpha$ with every t

Summary

- Parameter update:

Momentum, Adagrad, Rmsprop, Adam

- Learning rate decay:

1/t, exponential decay, step decay