Machine Learning: 2124095
Zhejiang University
Professor Deng Cai

Nov 19, 2020
Homework 3

Homework 3

Collaborators: /
Name: Chen Zhibo
Student ID: 3190100923

Problem 3-1. Neural Networks

In this problem, we will implement the entire process of the neural networks training, such as feedforward, backpropagation and optimizer.

(a) Affine layer

Answer:
forward: $y = wx + b$
difference: 9.77×10^{-10}
backward: The backpropagation is

$$\frac{\partial y}{\partial x} = \omega \times \frac{\partial l}{\partial y}$$

$$\frac{\partial y}{\partial \omega} = x \times \frac{\partial l}{\partial y}$$

$$\frac{\partial y}{\partial b} = \frac{\partial l}{\partial y}$$

difference:

error of dx is 9.83×10^{-11}
error of dw is 6.09×10^{-10}
error of db is 9.2276×10^{-12}

(b) Relu layer

Answer:
forward: y=max{x,0}
difference: 4.99×10^{-8}
backward: The backpropagation is $\begin{cases} x & (x > 0) \\ 0 & (x <= 0) \end{cases}$
difference: 3.28×10^{-12}

Inline Question 1:
the Sigmoid function will be getting zero gradient flow during backpropagation because the max value of Sigmoid is $\frac{1}{4}$ and when $\omega\sigma'(x)$ always >1 or <1, it will lead to gradient vanish or explosion.
Considering the 1 dimension case, when ω always <1, the gradient will vanish.

(c) Solver

Answer:

1. TwoLayerNet: train accuracy is 98.1% val accuracy is 96.3%
2. Three-layer Net to overfit: train accuracy is 100% val accuracy is 53.14%
3. Five-layer Net to overfit: train accuracy is 100% val accuracy is 50.1%
4. Inline Question 2:
   the network with more layers is harder to converge. Because a network with more layers means greater times of implementations of activation, which will cause shift of the data and made the the function easier to have gradient disappearance.

(d) Update relus

Answer:
with the same times of iterations, we find the val accuracy of SGD is 24.8%.And the val accuracy of SGD+momentum is 88.4%
the optimzation of SGD+momentum update rule converge faster.

(e) Conv layer

Answer:
forward difference: 2.21×10^{-8}
backward difference:
$dx = 1.16 \times 10^{-8}$
$dw = 2.25 \times 10^{-10}$
$db = 3.37 \times 10^{-11}$

(f) Pooling layer

Answer:
forward difference:4.17×10^{8}
backward difference:$dx = 3.28 \times 10^{-12}$

(g) Experiment

Answer:
I adjust the batchsize of the network from 5 to 500 and found batchsize at around 15 have the best performance. Then I set the filter size from 3 to 11 but find the initial 7 has the best performance. Next I adjust the filter num and find filter num that is approximately close to the batch size perfomance relatively better.

Also I tried changing the learning rate, weight scale. In the process I discovered that with smaller lr it's harder to converge, but with larger epoch, the value will be better getting closer to the best optimized point. So I set the learning rate as 10^{-3} and epoch as 10, finally get approximately 98.5%.
Also I tried other update rules like adam and rmsprop, but it doesn't seems to converge too fast and it's hard to find a approprate params to reach the best performance. So the actual performance doesn't seem to be better than the sgd+momentum optimizer.

Problem 3-2. Batch Normalization

The backpropagation of batch normalization.

(a) Answer:

$$\frac{\partial l}{\partial \gamma} = \sum_{i=1}^{m} (\widehat{x_i} \times \frac{\partial l}{\partial y_i})$$

$$\frac{\partial l}{\partial \beta} = \sum_{i=1}^{m} \frac{\partial l}{\partial y_i}$$

$$\frac{\partial l}{\partial x_i} = \frac{\partial l}{\partial y_i} (\frac{\partial y}{\partial \widehat{x_i}} \times \frac{\partial \widehat{x_i}}{\partial x_i})$$

$$= \frac{\partial l}{\partial y_i} \times \frac{\partial y}{\partial \widehat{x_i}} \times (\frac{\partial \widehat{x_i}}{\partial x_i} + \frac{\partial \widehat{x_i}}{\partial \sigma_B^2} (\frac{\partial \sigma_B^2}{\partial x_i} + \frac{\partial \sigma_B^2}{\partial \mu_B} \frac{\partial \mu_B}{\partial x_i}) + \frac{\partial \widehat{x_i}}{\partial \mu_B} \frac{\partial \mu_B}{\partial x_i})$$

$$= \frac{\partial l}{\partial y_i} \times \gamma \times (\frac{-1}{\sqrt{\sigma_B^2 + \epsilon}} + (-\frac{1}{2}) \times (\sigma_B^2 + \epsilon)^{\frac{-3}{2}} \times (x_i - \mu_B) \times (\frac{2(x_i - \mu_B)}{m} + \frac{\sum_{i=1}^{m} [-2(x_i - \mu_B)]}{m} \times \frac{1}{m})$$

$$+ (-\frac{1}{\sqrt{\sigma_B^2 + \epsilon}}) \times \frac{1}{m})$$