ITE4053 Deep Learning course

Practice 3

2016026080 Dajin Han

* **Experiment environments**

OS macOS Catalina

Memory 16GB

Language python3

Tools Jupyter notebook

* **Update Practice2 code**

The code used in Practice2 was modified and structured to create a network with a Layer classes. With Network class, you can create a network with multiple layer easily. You can create it by just putting together a list of layer when creating network object. Now Network class can have a layer that can manage stride option.

* + **Network Class**

스크린샷이(가) 표시된 사진

자동 생성된 설명

I made Forward(), Backward() functions operate continuously about layers inside the network object. We can check the inference time using predict function and can check training time using train function. Print\_layer function is made to make sure created network has accurate layers inside.

* + **Layer Class**

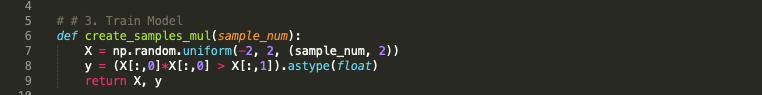
스크린샷이(가) 표시된 사진

자동 생성된 설명

In Practice\_2, I coded the program that perform backpropagation in neural network, with simple input shape. Shape of input is simple, It was (2,1). But in Practice\_3 I made Layer class that can manage every shape of nodes. I used np.prod()+np.sum() combination instead of np.dot() because it is easier to create code. But maybe it takes more time. So I plan to convert the code of np.prod() type to the code of np.dot() type in next assignment.

* **Practice\_3**
  + **Framework**

1. Create Samples



1. Make Network

그리기, 음식이(가) 표시된 사진

자동 생성된 설명

음식이(가) 표시된 사진

자동 생성된 설명

음식, 그리기이(가) 표시된 사진

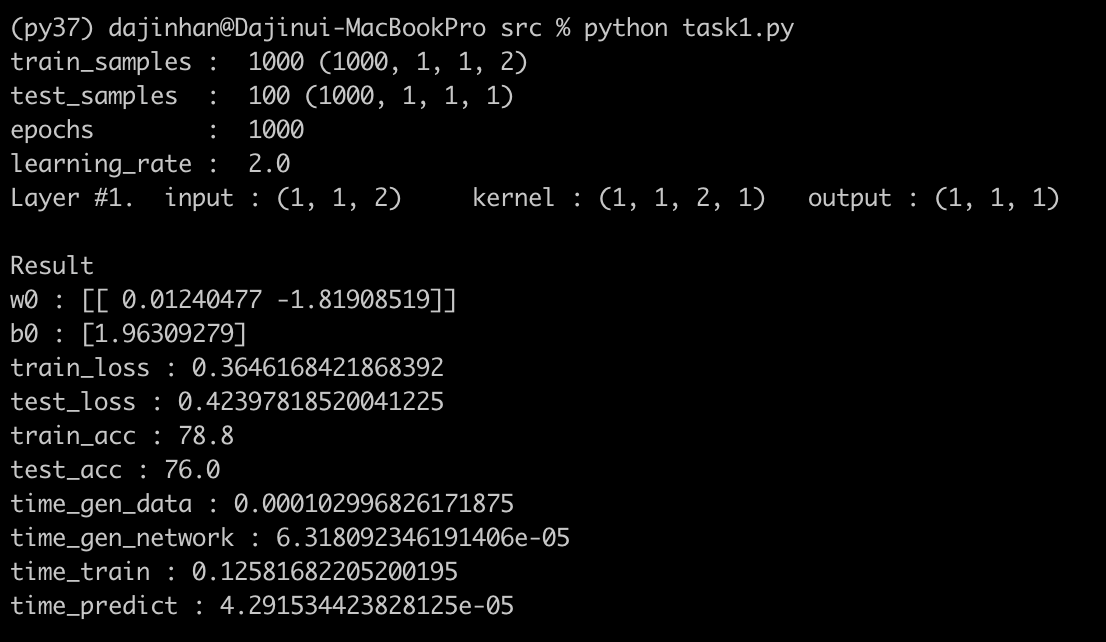
자동 생성된 설명

1. Run

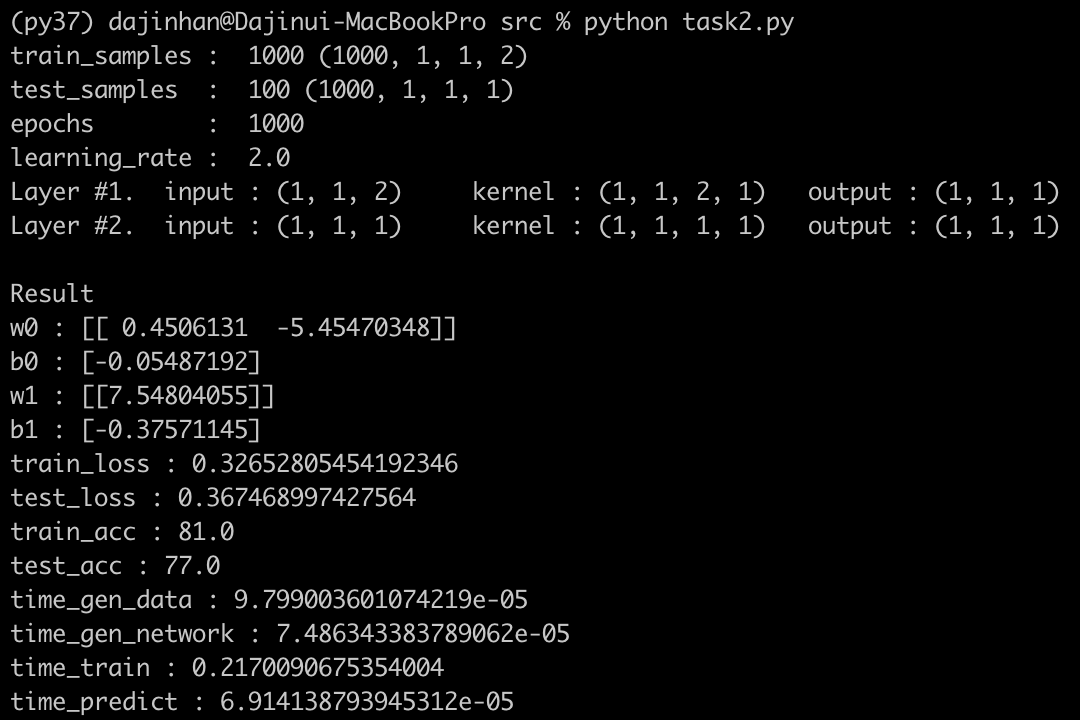
그리기이(가) 표시된 사진

자동 생성된 설명

* + **Result**
    - Task1



* + - Task2



* + - Task3

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

* + Comparison

|  |  |  |  |
| --- | --- | --- | --- |
|  | Task #1 | Task #2 | Task #3 |
| Accuracy\_train | 0.788 | 0.810 | 0.998 |
| Accuracy\_test | 0.760 | 0.770 | 1.000 |
| Time\_train | 0.126 | 0.217 | 0.458 |
| Time\_test | 0.00004 | 0.00007 | 0.0001 |

* **What I learned**

The process of figuring out how the interior of the Convolution layer was constructed was beneficial. Especially, the time to think about how to construct the internal backpropagation formula was helpful for me.

And when I first initialized the weight value in layer to zero, the accuracy was just 60%. But when I initialed it randomly, the accuracy increased to more than 90%. Some articles explain it is because of the activation function sigmoid. Initial distribution of weight value can cause the local minimum. Maybe I need to study more about this.