**Image Processing & Vision Homework 5: Convolutional Neural Networks(CNNs)**

예술공학대학 컴퓨터예술학부

20190807 민정우

**Follow and understand the examples in the code**

Convolutional Neural Networks(CNNs)를 구현하기 위해 Neural Networks의 Layer를 구성해야 한다. 각 Layer는 Convolution Layer, Batch Normalization, ReLU 연산을 차례로 거친다.

CNNs 모듈은 nn.Module을 상속하는 클래스를 통해 제작할 수 있으며, CNNs의 Layer와 forward propagation 함수는 모듈 클래스 내에서 선언된다.

2D Convolution Layer는 다음 함수로 구현한다.

|  |
| --- |
| nn.Conv2d(in\_channels, out\_channels, kernel\_size, stride=1, padding=0, dilation=1) |
| in\_channels : 입력받은 이미지의 채널 수  out\_channels : convolution의 결과 채널 수  kernel\_size : convolution filter(kernel)의 크기  stride : convolution의 보폭  padding : 입력의 가장자리에 추가되는 패딩  dilation : kernel 요소 사이의 간격 |

Batch Normalization은 다음 함수로 구현한다.

|  |
| --- |
| nn.BatchNorm2d(num\_features) |
| num\_features : 예상 입력 크기의 채널 수 |

ReLU 함수는 다음과 같이 제공된다.

|  |
| --- |
| nn.ReLU() |

모델을 최적화하기 위해서 모델을 통해 출력한 결과를 Loss Function을 통해 분석한다. Loss Function은 Cross Entropy Loss Function을 사용했다.

|  |
| --- |
| nn.CrossEntropyLoss() |

Loss Function을 통한 Optimizer에는 Stochasctic Gradient Descent를 사용했다.

|  |
| --- |
| optim.SGD(params, lr) |
| params : model의 예측 결과  lr : Learning Rate, Gradient Descent의 이동 거리 |

전체 데이터셋에 대한 학습의 반복 횟수는 epochs 변수를 통해 조정한다.

제공된 CNNs 코드는 다음 과정으로 동작한다.

- nn.Module을 상속한 myConvNet 클래스에서 CNNs의 Layer와 Forward propagation 함수를 제작한다.

- 제작한 CNNs을 통해 데이터셋을 학습한다. 데이터셋의 학습은 다음 방법으로 진행한다.

- 현재 모델에 대해 입력 이미지를 Forward propagation로 분석한다.

- 분석한 결과에 대해 Loss Function을 적용한다.

- Loss Function의 결과를 이용해 Optimize하고, Backward propagation로 모델을 갱신한다.

- 학습이 완료된 모델을 테스트하고 정확도를 출력한다.

**Design my own CNNs**

제공된 코드에서 다음을 수정해 CNNs를 조정할 수 있다.

- ConvNet Class에서 layer를 추가하고 이를 ConvNet.forward()에 반영

- nn.Conv2d()의 매개변수 수정을 통한 채널 및 kernel 조정

- optim.SGD()의 learning rate 조정

- epoch 변수 조정

테스트를 거듭하면서 수정한 부분은 다음과 같다.

- Layer의 개수를 2개에서 4개로 조정하고 각각의 매개변수 조정. 이에 따라 forward propagation도 수정

각 Layer의 매개변수 값은 다음과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| Convolution  Input Channel : 3  Output Channel : 64  Kernel Size : 3  Stride = 1 | Convolution  Input Channel : 64  Output Channel : 128  Kernel Size : 3  Stride = 1 | Convolution  Input Channel : 128  Output Channel : 256  Kernel Size : 3  Stride = 2 | Convolution  Input Channel : 256  Output Channel : 512  Kernel Size : 3  Stride = 2 |
| Layer 1 | Layer 2 | Layer 3 | Layer 4 |

- Optimizer의 learning rate를 0.01에서 0.001로 조정

- epochs를 5에서 100으로 조정

**Train and test my CNNs model on CIFAR-10 dataset**