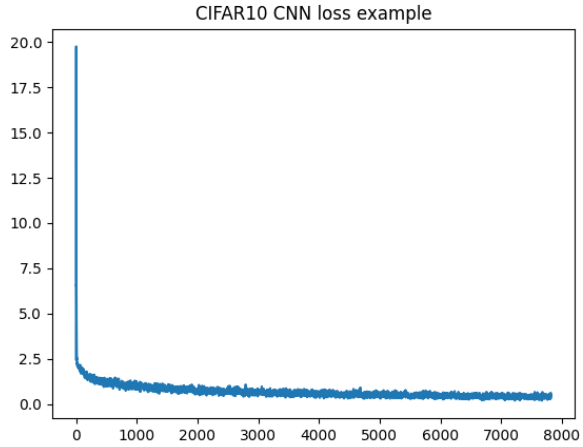
HW2 : CIFAR10 Classifier with CNN

20191611 유종선

1. 기준 모델 learning curve & 결과




```
Accuracy : 76.28 %
```

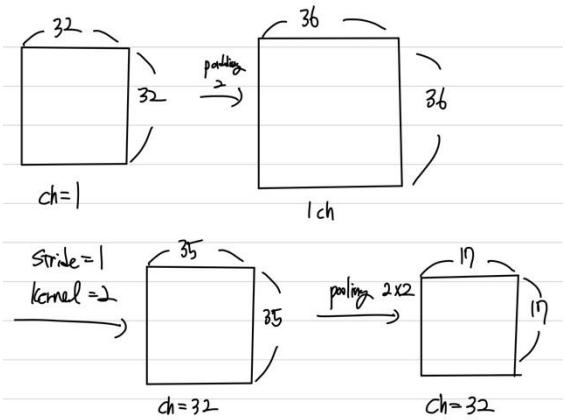
2. 기준 모델 hyperparameter

   Batch_size = 128

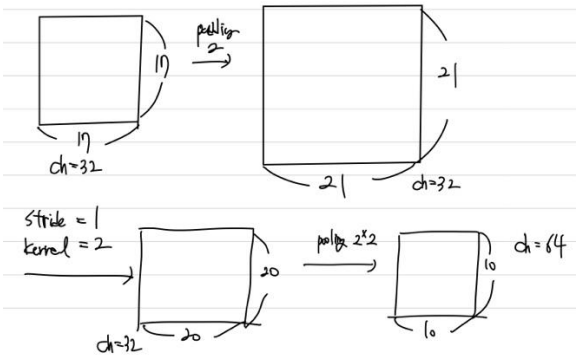
   Learning_rate = 0.005

   Epoch = 20

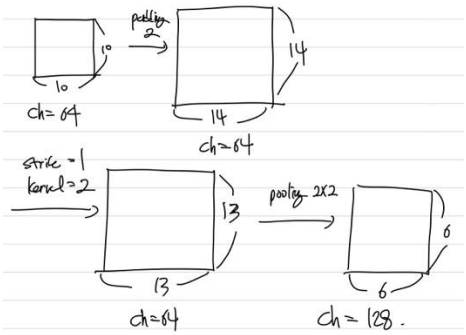
3. Output shape
   - Output size : {(input – kernel + 2 * padding) / stride} + 1
   - Input : 32 * 32 * 3channels
   - Convolution 1 : out = 32, kernel = 5, stride = 1, padding = 2, max_pool = 2



- Convolution 2 : out = 64, kernel = 5, stride = 1, padding = 2, max_pool = 2



- Convolution 3 : out = 128, kernel = 5, stride = 1, padding = 2, max_pool = 2



Convolution layer output : 128 * 6 * 6

- fc1 = Linear(128 * 6 * 6 -> 128)
- fc2 = Linear(128 -> 64)
- fc3 = Linear(64 -> 10)

4. For accuracy

i. Learning_rate(이하 lr)를 정하는 것이 상당히 어려웠다. Epoch을 올리는 것은 상당한 제약이 따르기 때문에, learning_rate를 올리고 싶었지만 0.0001에서 시작해서, 0.005보다 올라가게 되면, Loss가 튀는 현상이 자주 발생하였다. 그래서 0.005를 기준 learning_rate로 정하였다.

ii. Epoch은 최대 50까지 밖에 돌려보지 못했다. 당연히 epoch이 늘어날수록 loss 미세하게나마 감소하였으나, 정작 test accuracy에서는 유의미한 차이를

보기 힘들었다. 그래서 50 epoch에서 줄여나갔는데, 10에서 20은 차이가 나고, 20에서 50은 차이가 크지 않고 accuracy가 유지되어서, 20으로 epoch을 정하고 수행하였다. 더 좋은 환경에서 epoch을 크게 하여 돌리지 못한 것이 아쉽다.

iii. Dropout 함수는 오버피팅이나 노이즈 감소를 위해서 사용되는데, 기준 모델에서는 다음과 같이 dropout 함수를 한번만 사용하였다.

```
self.layer3 = nn.Sequential(
    nn.Conv2d(in_channels=64, out_channels=128, kernel_size=5, stride=1, padding=2),
    nn.BatchNorm2d(num_features=128),
   nn.ReLU(),
   nn.Dropout(0.25),
   nn.MaxPool2d(kernel_size=max_pool_kernel)
)
```

이렇게 사용한 이유는 layer1과 2에서도 동일하기 dropout을 사용하였을 때보다 이렇게 마지막 Layer에서만 적용했을 때 가장 좋은 결과값이 도출되었다. Linear 함수를 수행하고 난 다음에도 dropout 함수를 적용해보았는데 loss가 튀는 현상이 더 자주 발생하여서 하지 않았다. 0.25 또한 0에 가까운 경우와 0.25, 0.5에 가까운 경우 모두 해본 결과 0.25일 때 가장 좋은 결과값이 나왔다.

iv. Batch_size는 기본으로 64로 설정하고, 32와 128 이렇게 3가지로 테스트 해보았는데, 128일때가 가장 좋은 결과 값이 나왔다. 32와 64인 경우보다 128에서 소폭 더 좋은 accuracy를 도출하였다.

v. Activation function으로 relu와 leaky_relu 2가지를 모두 적용해보았는데, 두 함수 성능의 차이가 거의 나지 않았다. 그래서 relu 함수를 적용하였다.

vi. Weight initialization의 경우에, He initialization을 사용하였다. Conv2d layer에서는 nonlinearity = 'relu'로, Linear layera에서는 nonlinearity = 'linear'로 설정하여 수행하였다. 가중치 초기화를 하지 않는 것보다 하는 것이 정확도에서 크지는 않지만, 1~ 2% 정도 소폭의 상승이 있었다.