|  |
| --- |
| **ML Assignment**  Image classification CIFAR-10 |
| 2020년 06월 14일 |
| 한국항공대학교  소프트웨어학과  이상헌 |

**목 차**

1. **개요**
2. **성능 향상을 위한 방법**
3. **최종 모델**
4. **결론 및 느낀점**
5. **참고 문헌**
6. **개요**

2020년 1학기 머신러닝 11주차 과제는 pytorch 및 torchvision을 이용하여 cifar-10 네트워크를 훈련시키는 것이다. Baseline model을 그대로 훈련시킨 결과 정확도 약 60%를 얻을 수 있었다.

1. **성능 향상을 위한 방법**

기존 코드에서 사용하던 방식은 LeNet-5 네트워크 모델에 SGD 옵티마이저를 사용하였고, batch size는 4를 선택하였다. LeNet-5 네트워크 모델과 SGD 옵티마이저는 각각 네트워크 모델과 옵티마이저 중 가장 기본에 해당하는 방식이고, batch size 역시 개선의 여지가 보였으므로 제출한 코드에서는 이 셋을 수정하였다.

* 1. 네트워크 모델

LeNet-5 모델이 아닌 다른 모델을 선택하던 도중, torchvision 라이브러리에서 여러 종류의 네트워크 모듈을 제공한다는 사실을 알아냈다. torchvision에서 제공하는 여러 모듈 중에는 ResNeXt라는 모듈이 있는데, 이는 Andrew Ng 교수님의 강의에서 나오는 ResNet을 개선한 모델로써 pytorch 홈페이지에 따르면 가장 정확도가 높은 모델이다.

* 1. 옵티마이저

옵티마이저는 Adam을 사용했으며, 학습률은 epoch 0부터 19까지는 1e-4를, 20부터 39까지는 1e-5를 사용하였다. 이러한 방법을 사용한 이유는 하드웨어적 자원이 모자랐기 때문으로, Ng 교수님의 강의에서 Babysitting에 빗대어 소개된 방식을 사용할 수밖에 없었다. 또한 학습률을 위와 같이 설정한 이유는, 각각의 경우에서 학습률이 조금 더 높아진 경우 정확도가 서서히 감소하는 모습을 보였기 때문이다.

* 1. Batch size

Cifar-10의 train set의 크기는 50000으로, 이는 로 나타낼 수 있다. 이 때 batch size는 꼴일 때 최대의 효율을 발휘하고, 또한 대부분의 경우 각 mini batch의 크기가 동일할 때 학습 시간이 동일하므로 최적의 batch size를 , 즉 16으로 판단하였다.

1. **최종 모델**

위 세 부분을 수정한 결과, 최종 모델은 평균적으로 정확도 약 75%를 얻을 수 있었다.

1. **결론 및 느낀점**

첫째로 느낀 것은 머신러닝에서 수학이 차지하는 부분이 매우 크다는 것이고, 둘째로는 머신러닝을 위해서는 어마어마한 양의 하드웨어적 자원이 필요하다는 것이다.

1. **참고 문헌**

<https://tutorials.pytorch.kr/beginner/blitz/cifar10_tutorial.html> (예제 분석)

<https://pytorch.org/docs/stable/optim.html> (optimizer 변경)

<https://pytorch.org/docs/stable/torchvision/models.html> (학습 모델 네트워크 변경)

<https://everyday-deeplearning.tistory.com/entry/%EC%B4%88-%EA%B0%84%EB%8B%A8-%EB%85%BC%EB%AC%B8%EB%A6%AC%EB%B7%B0ResNext> (ResNeXt 모델 해설)

<https://medium.com/hyunjulie/%EA%B5%AC%EA%B8%80-colab-%EC%97%90%EC%84%9C-pytorch-%EC%82%AC%EC%9A%A9%ED%95%98%EA%B8%B0-e065902d4a91> (colab에서 pytorch를 이용한 머신러닝 실행시키기)