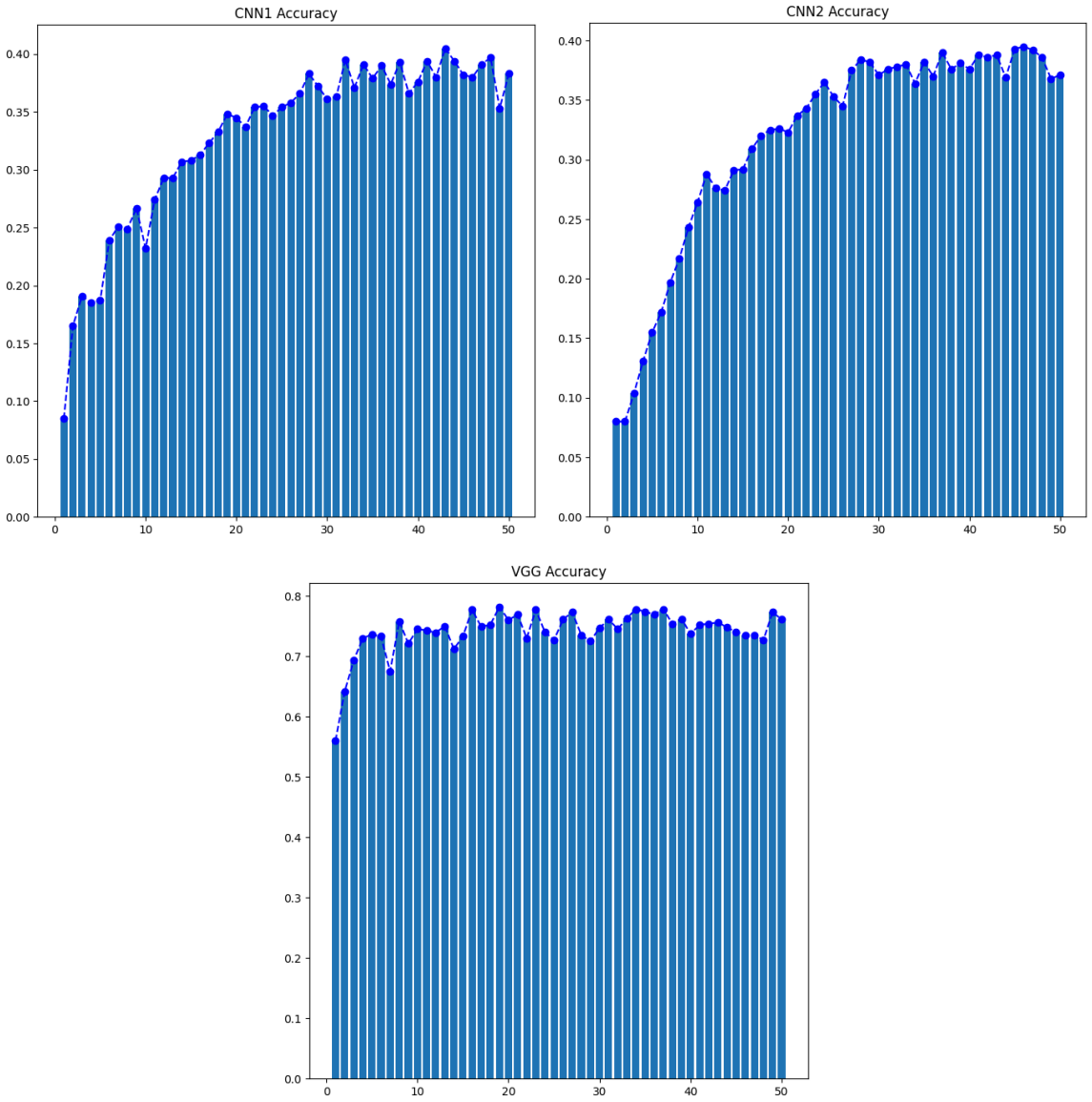

vanillaCNN1, vanillaCNN2, VGG19를 사용하였을 때 가장 잘 나온 validation score은 다음과 같다: vanillaCNN1은 43번째 epoch에서 0.405, vanillaCNN2는 46번째 epoch에서 0.395, VGG19는 19번째 epoch에서 0.782.

가장 좋은 성능을 보인 모델은 VGG19이며 가장 좋지 않은 성능을 보인 모델은 vanillaCNN2이다. VGG19가 가장 좋은 성능을 가장 큰 이유는 파라미터의 수이다. VGG19는 vanillaCNN1이나 vanillaCNN2에 비해서 훨씬 많은 파라미터 수를 갖고 있기 때문에 classification이나 localisation과 같은 task를 잘 수행할 수 있다. vanillaCNN2는 단순한 linear projection head를 사용해 분류성능이 좋지 않다는 vanillaCNN1의 단점을 보완하기 위해 head의 구성요소를 바꾸어 만들어졌지만 vanillaCNN1보다 정확도가 좋지 않았다. vanillaCNN2의 성능이 더 좋지 않은 이유는 사실 잘 모르겠지만, 랜덤에 의한 것이 아닐까 추측된다. 예를 들어, layer의 weight initialization이 어떻게 되었는가, 훈련 중 vanillaCNN2 head의 dropout layer에서 어떤 값이 dropout되었는가 등의 랜덤에 의한 결과가 오히려 안 좋은 결과를 끼쳤다고 생각한다.

Pytorch를 써야 하는 첫 번째 이유는 torch.nn이나 backpropagation과 같이 인공지능에서 주로 사용되는 툴을 제공해주기 때문이다. torch.nn에서는 linear, convolution, dropout 등의 layer와 ReLU와 같은 activation function 등을 제공해주며 torch.nn.Modele을 상속받아 사용자가

직접 모델을 직접 생성하여 사용해볼 수 있다는 장점이 있다. 두 번째 이유는 cuda를 사용할 수 있다는 장점이다. CPU를 이용하여 모델을 학습시키면 시간이 매우 걸린다. 실제로 이번 과제에서 local computer의 cpu를 이용하여 VGG 모델을 돌려보려고 했다가 시간이 너무 오래걸리는 것을 보고 바로 gpu로 돌렸었다. pytorch는 모델을 cpu 뿐만 아니라 gpu 환경에서도 사용가능하게 끔 하여 학습 시칸을 매우 단축시켜준다는 점에서 큰 장점을 가지고 있다.