**청년 AI-Big Data 아카데미**

컴퓨터 비전 실습 과제

담당교수 : 조민수

분반 : B4

이름 : 오홍석

POVIS ID : s\_1507

e-mail : iami0731@naver.com

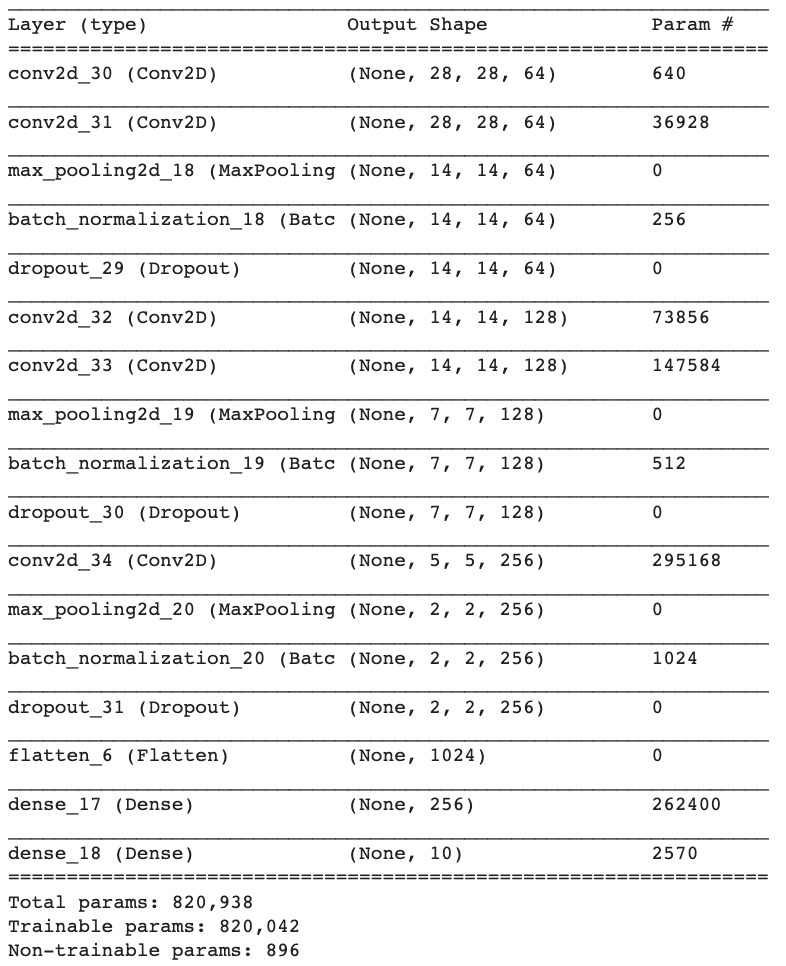
**1. 과제 설명**

해당 과제는 0~9까지 총 10개의 label로 이루어진 손글씨 이미지로 구성된 MNIST 데이터셋을 딥러닝 모델로 분류하는 것이다. 데이터셋은 각각의 레이블마다 28 x 28 크기 이미지인 60000개의 training 데이터셋, 10000개의 test 데이터셋이 존재하고, 결과적으로 총 70000개의 28 x 28 크기의 이미지로 데이터셋이 구성되어 있다.

**2. 진행환경**

과제 진행 환경은 실습시간에 생성한 가상환경, 구글의 코렙에서 진행하였다. 하지만 구글 코렙은 무료버젼의 사용량 제한과 런타임 연결 끊김으로 인해 최종적으로는 가상환경의 주피터 노트북으로 진행하였다.

**3. 진행전략**

3-1) 모델 구조

모델은 기본적으로 합성곱 신경망을 이용하였다. 전체적인 구조는 위의 CNN\_HS로 정의한 모델의 형태를 가진다. 첫번째로 모델에 들어가는 input 데이터의 shape이 28 x 28이기 때문에 연산을 수행할 때 여러 부분을 중복해서 탐색하기 위해 kernel\_size는 28과 비교해 상대적으로 작은 값인 3,3으로 설정하였다. 이는 모델의 모든 convolutional layer에서 동일하게 적용하였다. 또한 총 5번의 convolutional layer를 쌓았는데 이때에 위 4개의 convolutional layer에 padding을 same으로 설정하여 처음 데이터의 손실이 일어나지 않고 데이터 극단의 정보를 모두 활용하고자 하였다. 모델의 모든 활성화함수는 relu를 사용하였는데 이는 relu 활성화 함수의 특성 상 0보다 작은 값들을 모두 0으로 만들어주기 때문에 sparse한 형태로 만들어주어 연산량을 줄여주는 역할을 하기 때문에 연산속도가 빠르다. 또한 sigmoid 활성화 함수의 범위가 -1부터 1까지이므로 backpropagation으로 학습할 때에 기울기값이 사라지는 문제인 Vanishing Gradient Problem을 해결할 수 있기 때문이다. 각 convolutional layer마다 Maxpooling과 BatchNormalization, Dropout을 동일하게 적용하였다. 이 때에 pool\_size를 2,2로 설정하여 output shape를 합성곱신경망을 지날때마다 input shape의 1/2로 줄여나갔다. 각 합성곱 신경망마다 배치정규화를 진행하여 배치 간의 데이터 분포 차이를 줄이고자 하였다. 이후 Dropout으로 각 layer에서 모든 노드들을 연결하지 않고 75%의 노드들만 연결하여 overfitting을 피하고 학습 속도를 개선하고자 하였다. 마지막으로 추출된 특징들을 1차원 자료로 바꿔주기 위해 이미지 형태의 데이터를 배열형태로 flatten 한 뒤 한 개의 Fully connected Layer와 Output Layer를 통해 10개의 클래스를 분류하고 이때의 활성화함수는 다중분류를 하기 위해 softmax를 사용하였다.

3-2) 학습전략

모델의 학습은 기본적으로 random\_seed를 고정한 상태로 진행하였다. 모델이 seed값에 의해 바뀌기 때문에 해당 모델을 재사용하기 위해 tensorflow의 random.set\_seed를 통해 고정하고 학습을 진행하였다. 모델 학습 시 손실 함숫값을 최소화하는 방향으로 모델의 가중치 값을 갱신하기 위한 전략을 결정하는 옵티마이저는 SGD, Momentum, Adam, RMSProp, AdaGrad등이 있다. 이 중 Adam은 학습률을 변경하는 RMSProp과 최적화에 의한 갱신 경로를 변경하는 Momentum을 조합하여 만들어진 optimizer이다. 현재 많은 딥러닝 모델에서 Adam을 사용하고 있고 2020년 9월에 게재된 '기계학습 옵티마이저 성능 평가' 논문에서도 MNIST, CIFAR-10, IRIS, TITANIC, Boston Housing Price 데이터셋으로 동일한 모델로 다른 옵티마이저로 평가하였을 때, Adam의 성능이 모든 데이터셋에서 좋은 성능을 보이고 있다. 그렇기 때문에 모델의 optimizer는 Adam으로 진행하였다. 또한 해당 과제에서 다루고 있는 데이터셋이 손글씨 이미지이기 때문에 글자를 쓴 사람에 따라 이미지 숫자의 크기가 다를 수도 있고, 중간에 글자가 존재하지 않을 수 있고, 뉘여져 있을 수도 있다. 이러한 데이터를 최대한 일반화시켜 학습하기 위해 데이터셋을 배치 단위로 augumentation하는 과정을 진행한다. 이 과정에서 무작위로 0~15도 범위로 회전시키고, 0~5% 확대 혹은 축소하고, 위아래좌우로 픽셀 기준 20% 이동시키도록 설정하였다. 이후 모델을 학습 시켰다. 모델이 최적의 파라미터를 찾기 위해 keras의 EarlyStopping과 ReduceLROnPlateau 기법을 사용하였다. EarlyStopping은 모델이 학습을 진행할 때 epoch에 따라 학습의 정도가 결정되는데 이때 너무 큰 epoch는 overfitting을 일으키고 너무 적은 epoch는 underfitting을 일으키기 때문에 특정시점에서 학습을 멈추게 하였다. 이때 기준은 validation의 정확도가 10번 이상 오르지 않으면 학습을 멈추도록 설정하였다. ReduceLROnPlateau는 모델이 학습을 진행할 때에 더이상 학습이 진행되지 않으면 learing rate를 감소시키면서 모델을 개선하는 방법이다. 이때 기준은 위와 동일하게 validation의 정확도로 확인하였으며 정확도가 5번이상 개선되지 않으면 learing rate를 0.2배 하여 학습을 진행하게 하였다.

3-3) 학습결과

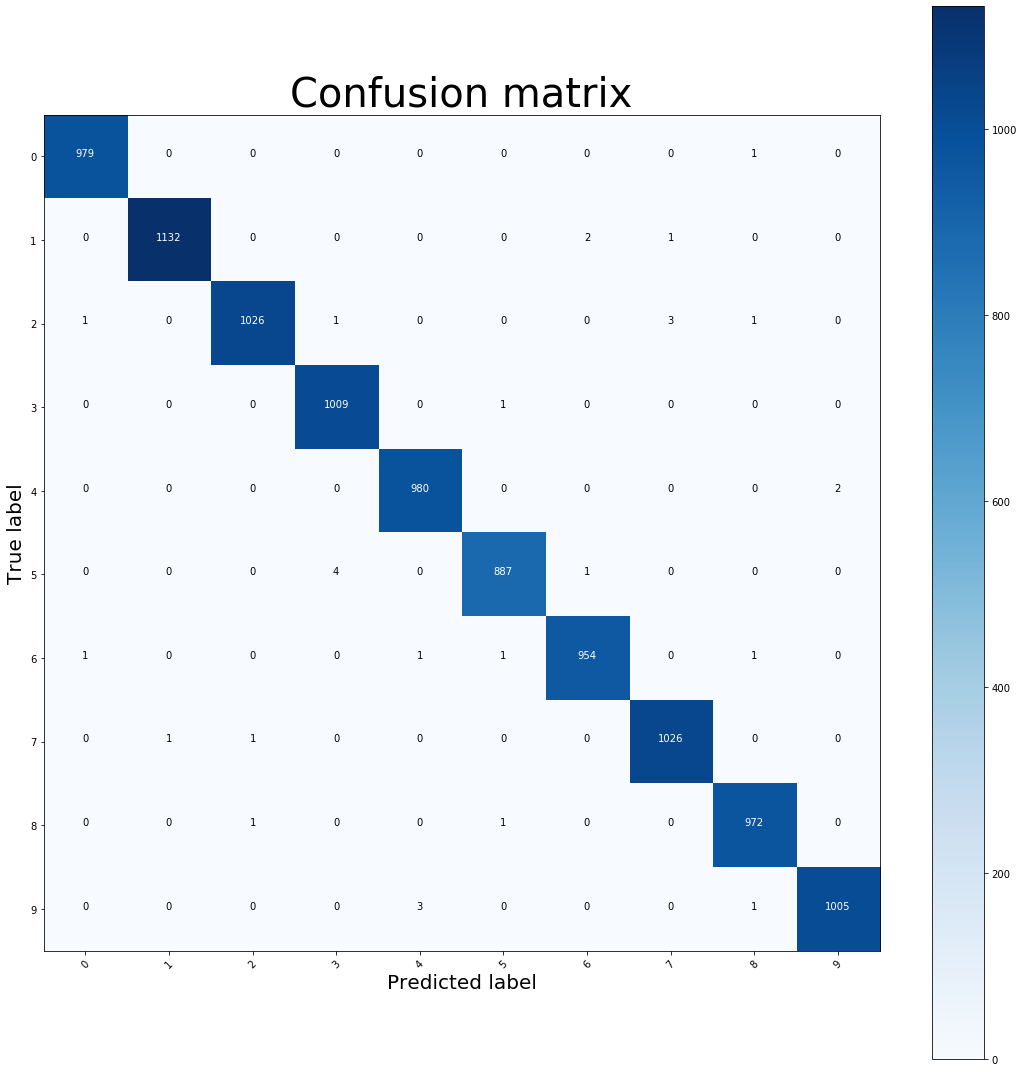
위의 모델구조와 학습전략으로 모델을 학습한 결과 최종적으로 test 데이터셋에 대한 loss와 정확도가 각각 0.0108, 0.9970이 나옴을 확인 할 수 있었다.



**4. 성능 분석**

4-1) 잘못된 분류 케이스 확인 및 현실적인 최대 정확도 제안

총 10000개의 test 데이터셋의 0.3%인 30개의 잘못된 분류케이스를 확인하였다.



|  |  |
| --- | --- |
| 실제 숫자 | 오분류 개수(개) |
| 0 | 1 |
| 1 | 3 |
| 2 | 6 |
| 3 | 1 |
| 4 | 2 |
| 5 | 5 |
| 6 | 4 |
| 7 | 2 |
| 8 | 2 |
| 9 | 4 |

잘못된 분류케이스를 모델이나 데이터 전처리를 통해 개선할 수 있는 케이스, 개선할 수 없는 케이스로 나누어 보았다. 개선할 수 없는 케이스는 다음과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

해당 케이스들은 True값을 보았을 때 실제로 사람이 잘못 쓴 경우이다. 일반적인 형태의 숫자가 아니고 아예 잘 못 작성하였거나 어느 부분이 흐리게 작성되거나 제거된 케이스들이다. 그렇기 때문에 해당 케이스들은 모델로 분류할 수 없다고 판단되어 현실적인 최대 정확도는 0.988이라고 판단한다.

4-2) 개선할 수 있는 방향 제안

개선할 수 있는 케이스의 예시는 다음과 같다.

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

첫 번째로는 True라벨이 1인데 7이라고 예측하였다. 1을 저런 형식으로 쓰는 경우도 있기 때문에 1클래스에 대한 저 모양의 데이터셋을 추가하여 학습한다면 개선 가능 할 것이다. 두 번째로는 True 라벨이 6인데 5라고 예측하였다. 데이터의 회전을 랜덤으로 진행했음에도 불구하고 정확한 클래스로 예측을 하지 못한 것을 볼 수 있다. 이 또한 데이터 augmentation을 진행하여 클래스별로 더욱 다양한 데이터셋을 확보한다면 개선 가능 할 것이다. 나머지 예시 또한 첫 번째와 두번째와 동일하게 클래스별로 더욱 다양한 표기법 데이터셋을 확보하고 회전과 이동 등 다양한 augmentation을 진행한다면 성능을 개선 가능 할 것이라고 판단된다.