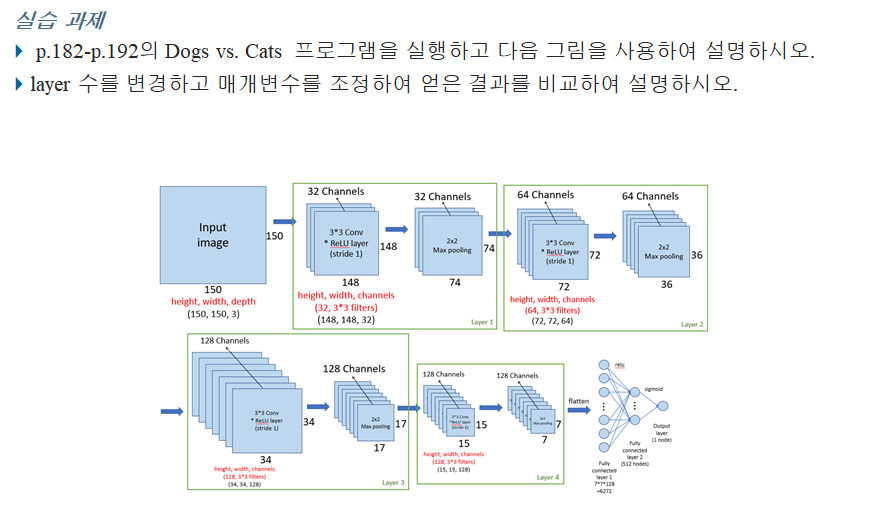
**CNN 실습**

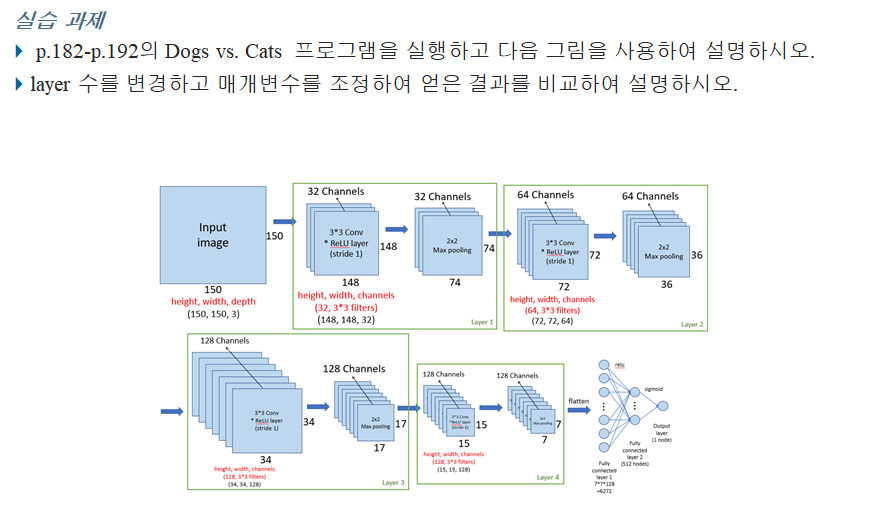
컴퓨터공학과 고은비

- Dogs vs. Cats 프로그램을 실행하고 다음 그림을 사용하여 설명한다.

- layer 수를 변경하고 매개변수를 조정하여 얻은 결과를 비교하여 설명한다.



[Input & Layer 1]



텍스트이(가) 표시된 사진

자동 생성된 설명

케라스에서 models와 layers를 import하고, Sequential Model을 불러온다. Sequential Model은 신경망 모델을 만드는 방법 중 하나로, 순차적으로 layer를 더해나가는 방식이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

합성곱 계층은 CNN에서 핵심이자 가장 중요한 구성요소이다. 데이터의 형상을 유지하면서 학습시키기 위한 뉴런(filter/kernel)의 형태도 달라진다. 이미지 기준에서 합성곱은 이미지들의 특징맵(Feature Map)을 찾는 것을 의미한다. 이미지 데이터인 2차원 데이터를 다루고 있으므로 2차원 Convolutional layer인 Conv2D를 사용하여 합성곱 계층을 만들어준다. 3\*3 필터를 사용하여 32개의 채널로 하나의 이미지를 만든다. 필터는 합성곱 계층에서의 가중치에 해당하며, 학습단계에서 적절한 필터를 찾도록 학습하면서 업데이트된다.

stride는 입력 데이터에 필터를 적용할 때 이동할 간격을 조절해주는 파라미터로, 입력 데이터가 큰 경우 연산량을 줄이기 위한 목적으로 사용한다. stride 값을 크게 하면 데이터의 특징 일부를 잃어버릴 가능성이 크기 때문에 대부분 값을 1로 하고 이후에 Pooling 계층을 통해 이미지 크기를 줄이는 sub-sampling 과정을 거친다.

CNN에서 활성화 함수는 relu를 사용한다. relu 함수의 비선형성을 이용하여 사전훈련 없이도 신경망 훈련을 가능하게 하고, 복잡한 데이터 셋에서도 빠르고 효과적인 훈련을 할 수 있다.

input\_shape에서, 각 이미지 데이터마다 크기가 일정하지 않기 때문에 입력 사이즈를 동일하게 하기 위하여 이미지 크기는 150\*150으로 지정한다. RBG 색깔을 가지는 이미지는 채널의 개수가 3개이므로 각각의 채널마다 합성곱 연산을 한 결과를 모두 더한 값으로 특징 맵을 추출한다. 이때 입력 데이터의 채널 수와 필터의 채널 수가 같아야 한다.

3\*3 필터를 적용하여 이미지 크기는 148\*148이 되며, 이 과정을 통해 유사한 이미지의 영역을 강조하는 특성 맵(feature map)을 출력하여 다음 계층으로 전달한다.

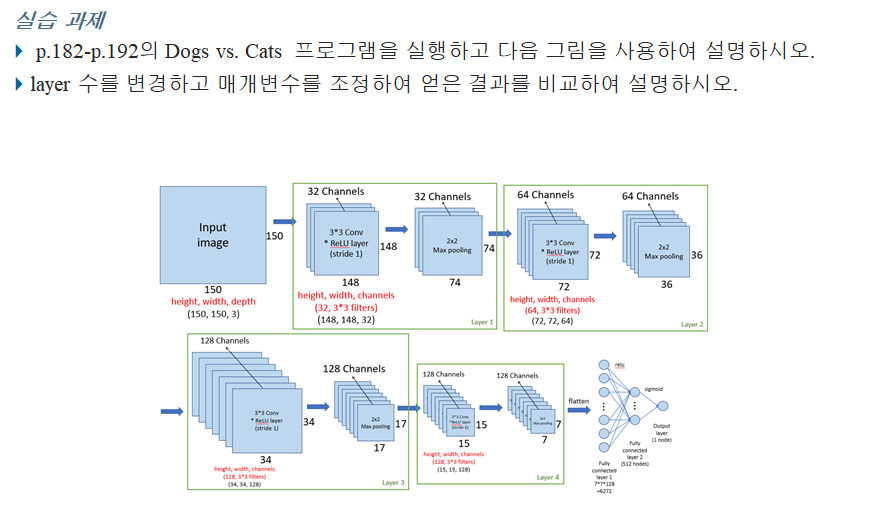
텍스트이(가) 표시된 사진

자동 생성된 설명

이미지 학습을 위한 CNN은 성능이 뛰어나 입력 데이터에 과도하게 맞춰지기도 한다. 풀링 계층은 이러한 과대적합(overfitting)을 줄이는데 도움을 주며, 이미지의 크기를 줄이면서 데이터의 손실을 막기 위한 역할로도 사용된다.

Max pooling 방법은 대부분의 이미지 학습에 주로 사용된다. pooling 크기를 (2, 2)를 사용하여 MaxPooling2D를 수행하면 이미지 크기는 각 크기의 절반씩인 74\*74가 된다.

[Layer 2]



텍스트이(가) 표시된 사진

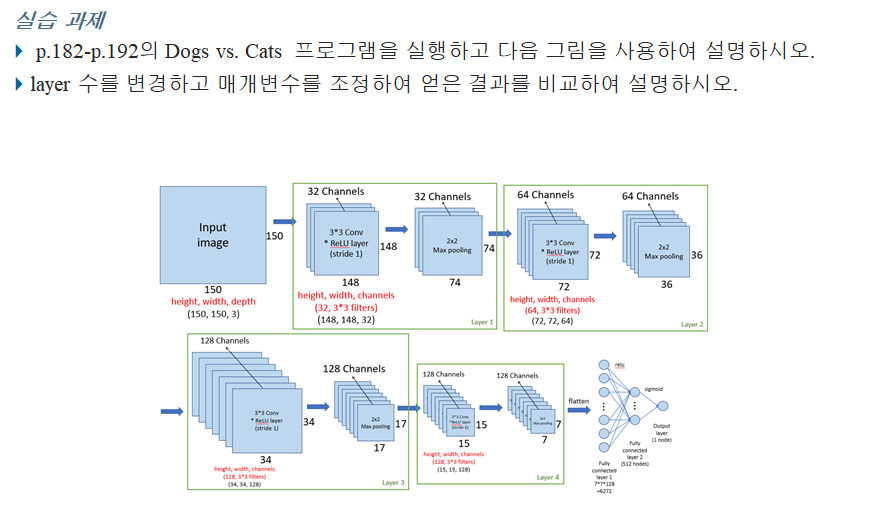
자동 생성된 설명

일반적으로 CNN에서는 레이어가 깊어지면 너비와 높이는 감소하고 채널(깊이)은 증가한다. 이때 각 채널은 서로 다른 특징(feature)을 적절히 추출하도록 학습되므로 다양한 특징들을 조합하여 적절히 분류를 수행한다.

Layer1에서 Layer2로 가면서 32개의 이미지에서 64개 이미지로 채널을 두배 늘렸다. 이 과정에서 이미지의 크기는 72\*72로 조금 줄어들었다.

위의 과정과 같이 Conv + MaxPooling 세트를 진행한다.

[Layer 3]

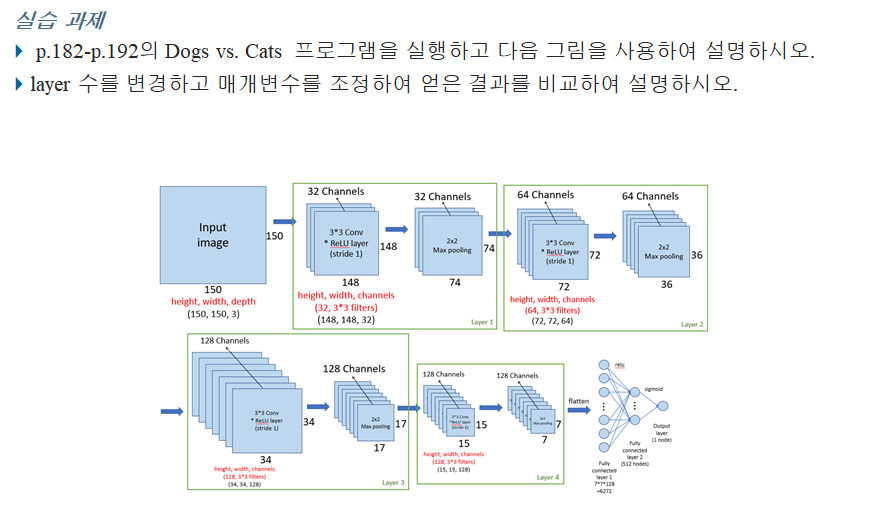


텍스트이(가) 표시된 사진

자동 생성된 설명

Layer3에서는 채널의 수가 두배 증가하여 128개로 구성되고, Conv + MaxPooling 세트를 수행한다.

[Layer 4]



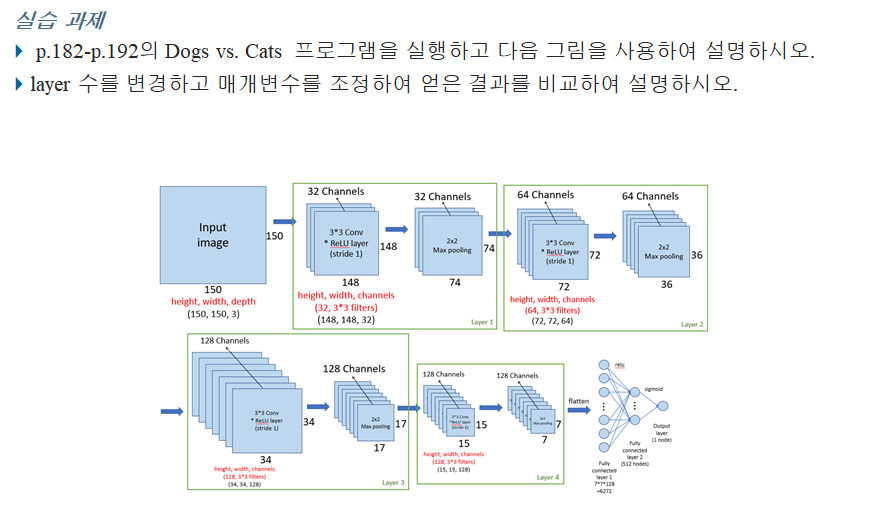
텍스트이(가) 표시된 사진

자동 생성된 설명

이미지가 크고 복잡한 문제이기 때문에 네트워크를 좀 더 크게 만들어준다. Conv + MaxPooling 단계를 하나 더 추가하면 네트워크의 용량을 늘리고 Flatten 층의 크기가 너무 커지지 않도록 특성 맵의 크기를 줄일 수 있다. 150×150 크기의 입력으로 시작해서 Flatten 층 이전에 7×7 크기의 특성 맵으로 줄어든다.

위에서 설명한 CNN의 패턴과 같이, 특성 맵의 깊이는 네트워크에서 점진적으로 증가하지만(32에서 128까지), 특성 맵의 크기는 감소한다(150×150에서 7×7까지).

[Classification & Output]



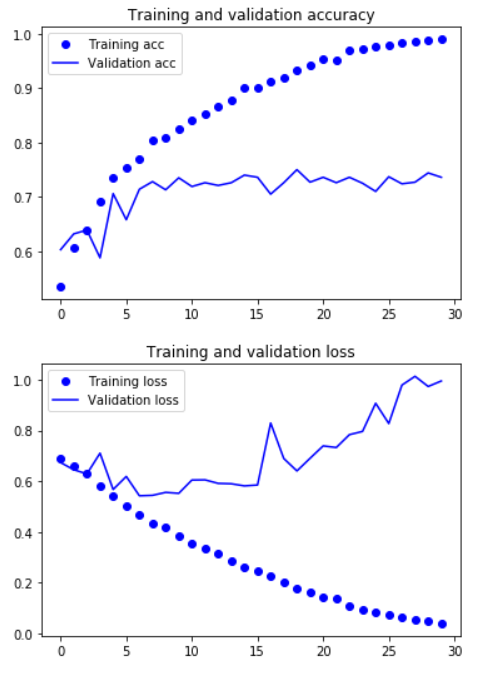
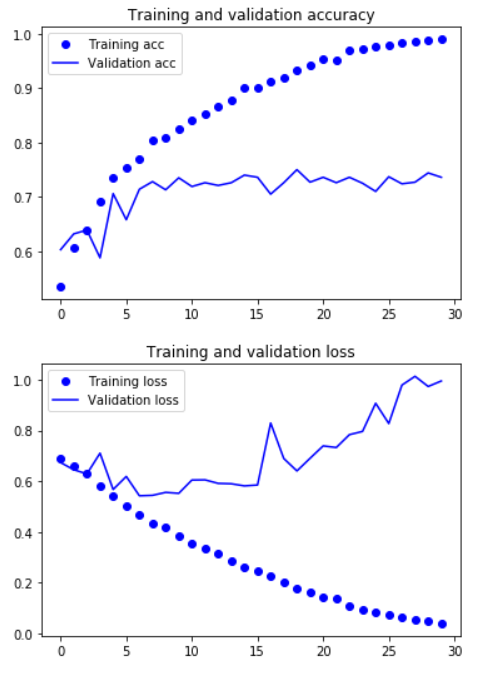
텍스트이(가) 표시된 사진

자동 생성된 설명

모델을 완성하려면 마지막 합성곱 층의 출력 텐서를 하나 이상의 Dense 층에 주입하여 분류를 수행해야 한다.

CNN에서 Convolution layer와 Pooling layer를 반복적으로 거치면 주요 특징만 추출된다. 추출된 주요 특징은 2차원 데이터로 이루어져 있기 때문에 Dense와 같이 분류를 위한 학습 레이어에서는 1차원 데이터로 바꾸어서 학습이 되어야 한다. Flatten layer는 CNN의 데이터 타입을 Fully Connected Neural Network 형태로 변경하는 레이어로, 2차원 데이터를 1차원 데이터로 바꾸는 역할을 한다. 7\*7 사이즈의 이미지 128개를 1차원 데이터로 변환하여 (7, 7, 128) 출력을 7\*7\*128=6272 크기의 벡터로 펼쳤다.

그 다음 Dense 층을 추가하고, relu 함수를 사용하여 512개를 출력한다. 이진 분류 문제이므로 네트워크는 하나의 유닛(크기가 1인 Dense 층)과 sigmoid 활성화 함수로 끝난다. 이 유닛은 한 클래스에 대한 확률을 인코딩할 것이다.



이 그래프는 과대적합의 특성을 보여준다. 비교적 훈련 샘플의 수(2,000개)가 적기 때문에 과대적합이 가장 중요한 문제이다.

[추가실험]

텍스트이(가) 표시된 사진

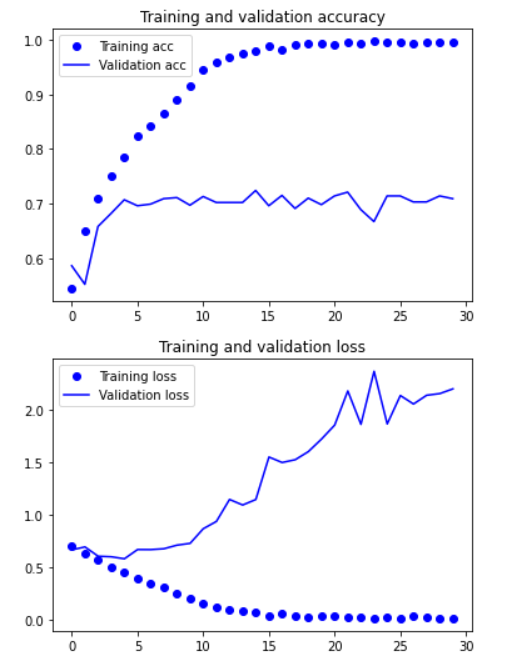
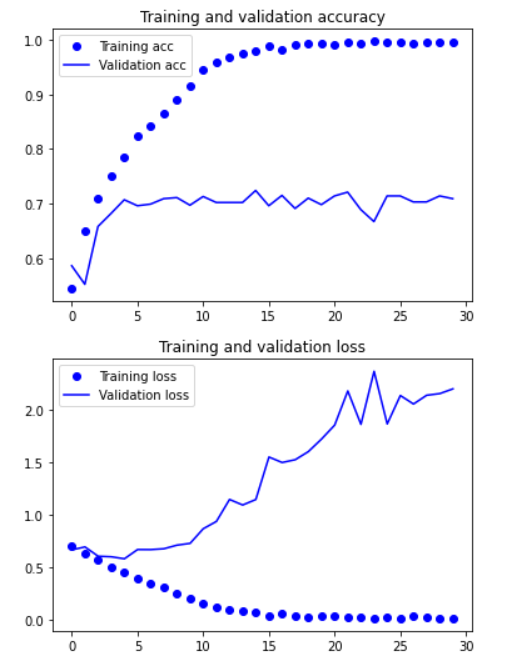
자동 생성된 설명기존코드

텍스트이(가) 표시된 사진

자동 생성된 설명변경코드

기존 코드에서는 4개의 Conv + MaxPooling 세트로 CNN이 구성되었다. 이미지 분석 과정에서 일부 기능을 추출할 수 있는 경우에는 MaxPooling을 수행하는 것이 좋다. 하지만 CNN의 초기 단계에서는 커널이 edges와 gradients를 추출하는 단계에 있기 때문에 MaxPooling을 수행하는 것이 권장되지 않는다(Max Pooling in Convolutional Neural Network and Its Features, 2020.3). 따라서 첫 번째 레이어에서는 MaxPooling을 수행하지 않았다.

특성 맵의 깊이를 128에서 256으로 2배 증가시키고, 추출된 특징들을 서브 샘플링(sub sampling)하는 것보다는 특징을 보다 정교하게 추출하는 것이 더 중요하다고 판단하여 네 번째 레이어에서도 MaxPooling을 수행하지 않았다. 또한 마지막에 Conv + MaxPooling 단계를 하나 더 추가하여 네트워크의 용량을 늘리고 Flatten 층의 크기가 너무 커지지 않게 특성 맵의 크기를 줄일 수 있도록 하였다.



하지만 학습 성능이 더 좋아질 것이라는 예상과 달리, 더욱 미흡한 결과가 나왔다. 훈련과 검증 정확도는 훈련 정확도가 기존보다 더 빨리 높아졌다는 것 외에는 기존 학습과 거의 차이가 없다. 위의 코드는 전부 훈련 정확도를 높이는 방향으로 수정한 코드이기에 당연히 훈련 정확도가 높게 나올 수밖에 없다.

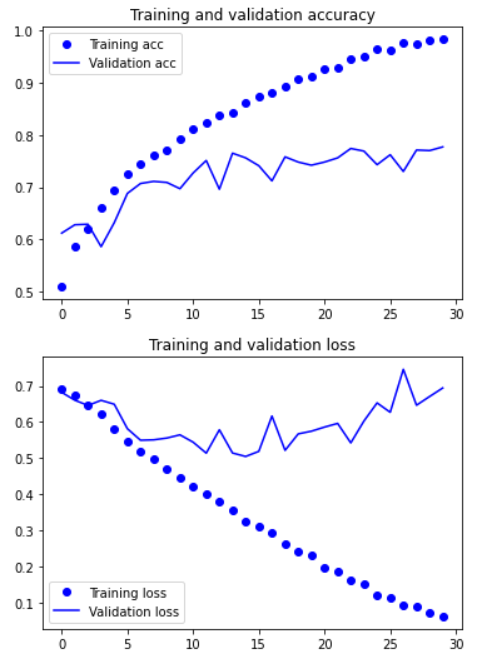
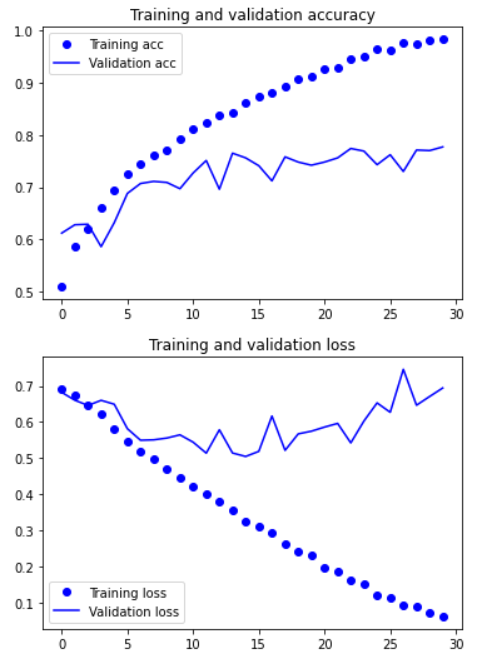
더 심각한 문제는 검증 손실이다. 훈련을 거듭할수록 검증 손실이 2.0이 훌쩍 넘는 값이 나오고 있다. 이는 과대적합이 더욱 크게 발생하였음을 보여주고 있으며, 위의 예측은 잘못된 예측임을 알게 되었다. Convolutional Layer를 진행할수록 과적합이 발생한다는 것을 인지하지 못하고, 합성곱 계층의 수만 늘렸다. 위의 방법은 훈련에서의 정확도만 높이는 방법일 뿐, 전반적인 학습능력을 높이는 방법이 아니다. 이 추가실험의 가장 큰 실수는 MaxPooling 연산을 두 번이나 제외한 것이다. MaxPooling은 과대적합을 줄여줄 수 있는 좋은 기능이기에, 함부로 Pooling 연산을 제외하는 것이 바람직하지 않음을 알 수 있었다.

[추가실험2]

텍스트이(가) 표시된 사진

자동 생성된 설명수정코드2

위의 문제점들을 개선하여 다시 학습을 수행하였다. 4개의 Conv + MaxPooling 세트로 CNN을 구성하고, 4번째 레이어에서 특성 맵의 깊이를 256으로 두 배 증가시켰다. 그리고 마지막 레이어에에만 MaxPooling을 추가하였다. MaxPooling으로 오버피팅이 좀 더 줄어들 것으로 예상한다.



예측한 대로 비교적 좋은 학습 결과가 나왔다. 처음 학습 결과보다 훨씬 개선된 모습을 보이고 있다. 물론 훈련과 검증의 폭이 상당히 줄어들었으며, 오버피팅이 발생하는 시점 또한 많이 밀려났다.

이를 통해 특성 맵의 깊이 증가와 MaxPooling 수행은 오버피팅을 줄여주는 데 많은 도움을 준다는 것을 알 수 있다.

[참고자료]

프랑소와 숄레, 『케라스 창시자에게 배우는 딥러닝』, 길벗, 2018.

「CNN (Convolutional Neural Network) 요약 정리」, https://ndb796.tistory.com/477

「Convolutional Neural Network (CNN)」, https://www.tensorflow.org/tutorials/images/cnn?hl=en

「Max Pooling in Convolutional Neural Network and Its Features」, https://analyticsindiamag.com/max-pooling-in-convolutional-neural-network-and-its-features/

「딥러닝 (6) - CNN (Convolutional Neural Network)」, https://davinci-ai.tistory.com/29

「딥러닝 - 초보자를 위한 컨볼루셔널 네트워크를 이용한 이미지 인식의 이해」, https://bcho.tistory.com/1149

「완전 연결 계층, Fully connected layer」, https://dsbook.tistory.com/59?category=780563

「폴링 계층, Pooling Layer」, https://dsbook.tistory.com/79

「합성곱 계층, Convolution Layer」, https://dsbook.tistory.com/72?category=780563

「합성곱 신경망, Convolutional Neural Network (CNN)」, https://dsbook.tistory.com/71?category=780563