딥러닝기초

**Dogs vs. Cats**

– Create an algorithm to distinguish dogs from cats

김준수, 이채별, 정원용, 홍민재

**1. 요약 Abstract**

딥러닝기초 수업의 과제로 cats&dogs 구분을 하게 되었다. 본 연구는 개와 고양이 이미지들을 분류하는 데 있어 최적의 데이터 전처리와 분류 모델을 제안한다. 데이터 전처리 과정에서 개와 고양이 데이터를 category로 분류하였고, augmentation을 활용하여 데이터를 증가시켰다. 분류 모델은 CNN을 활용하고, 각 layer 층의 파라미터 값을 조정함으로써 정확도를 향상시켰다.

**2. 서론 Introduction**

본 연구는 캐글에서 제공하는 데이터 세트를 이용한다. 3장에서는 개와 고양이 이미지 데이터의 특징을 파악하고 이를 활용하여 전처리하는 방법을 설명한 후 4조에서 개발한 이미지를 분류하는 딥러닝 모델에 대해 설명한다. 4장에서 학습 결과를 서술하고, 5장에서 결론과 향후 연구 과제를 서술하고자 한다.

**3. 방법론 Methods**

**3.1 데이터 분석**

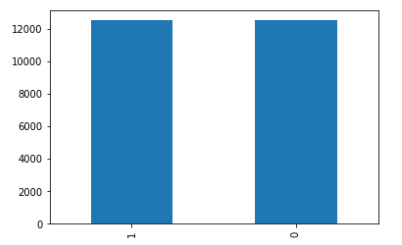
캐글에서 train, test 데이터를 받아 데이터에 대한 정보를 확인해본 결과 train 개와 고양이의 데이터 수는 12,500개로 서로 같았다. 그림[3-1]을 통해 확인할 수 있다.

그림 [3-1]

개와 고양이의 데이터를 보면 영상들은 single 영상으로, 개와 고양이를 제외한 다른 객체들은 포함되지 않았음을 확인했다.

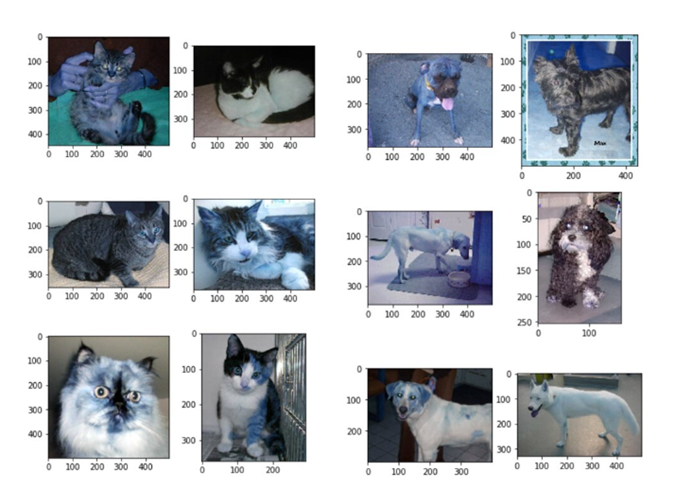


그림 [3-2]

**3.2 A1 모델**

A1 모델은 딥러닝기초 강의에서 배운 16장 내용을 최대한 활용하였다. 모델을 최대한 간단하게 구성하여 빠른 학습을 목표로 제작하였다.

**3.2.1 데이터 호출**

캐글 환경에서 연구를 진행하였으므로, 데이터는 자동으로 저장되어 있어 압축 파일만 풀어 연구를 진행했다.



그림[3-3]

**3.2.2 데이터 분류**

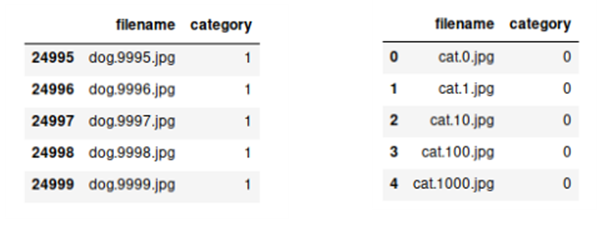
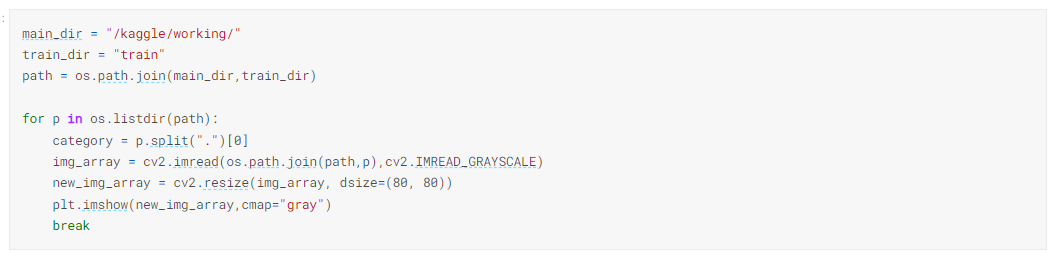
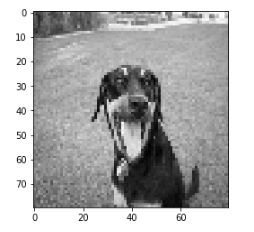
개와 고양이의 데이터가 train 폴더 안에 함께 들어 있으므로 개와 고양이로 분류 할 필요가 있다. 배열 두 개를 생성하여 한쪽에는 이미지 파일 이름, 다른 한쪽에는 개와 고양이를 구분할 클래스를 지정했다. 고양이는 0으로 강아지는 1로 설정하였다.

그림 [3-4]

**3.2.3 데이터 전처리**

교재 16장을 살펴보면 글자를 인식하는 CNN을 설명하고 있다. 글자 데이터를 확인하면 RGB 값이 없는 흰색 바탕에 검은색 글인 영상이다. 따라서 A1 모델도 빠른 학습을 위해 고양이와 강아지의 데이터를 RGB값을 제외시킬 필요가 있었다. OpenCV를 통하여 데이터를 바꾸고 학습 속도를 위해 이미지의 크기도 줄였다. 영상의 크기는 80\*80 사이즈로 변경하여 모델에 적용하였다.



그림[3-5]

데이터들을 전처리한 후 강아지는 1, 고양이는 0으로 구분하여 나중에 정답을 확인할 때 사용하였다. 이미지들을 normalize를 하기 위해 255를 나누었다.





그림[3-6]

**3.2.3 A1 모델링**

모델은 Sequential()을 통해 층을 추가하면서 제작하였다. 총 7층으로 구성되어 있다.

1층은 Conv2D로 3\*3필터를 64개 생성하였고, activation으로는 ‘relu’ 함수를 사용하였다.

2층은 MaxPooling2D로 2\*2사이즈로 적용하였다.

3층은 Conv2D로 3\*3필터를 64개 생성하였고, activation으로는 ‘relu’ 함수를 사용하였다.

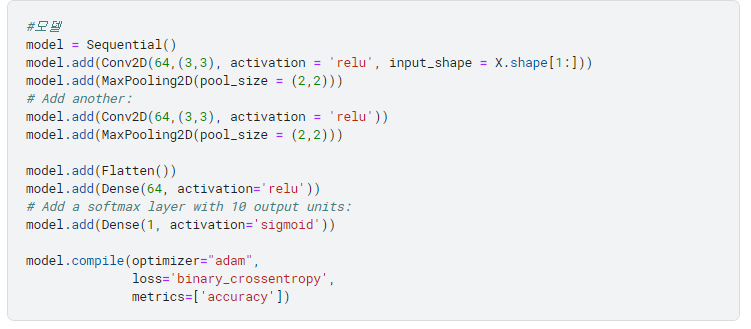
4층은 MaxPooling2D로 2\*2사이즈로 적용하였다.

5층은 Fallten을 사용하여 이미지를 하나로 정렬시켰다.

6층은 Dense로 64개로 사용하였고 activation으로는 ‘relu’ 함수를 사용하였다.

7층은 출력층으로 Dense를 1개로 사용하였고 activation으로는 ‘sigmoid’ 함수를 사용하였다.

모델의 compile 과정에서 optimizer는 ‘adam’을 손실 함수는 ‘binary\_crossentropy’을 사용하였다.



그림[3-7]

**3.3 A2 모델**

A2 모델은 A1 모델보다 정확도 측면에서 성능을 향상시키고자 했다. 데이터를 증폭하는 방법을 활용하여 모델의 정확도를 더욱 높일 수 있었다.

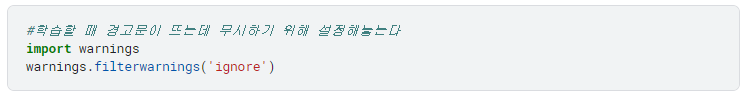
**3.3.1 데이터 준비**

필요한 라이브러리를 불러온 후 캐글 환경에서 zip파일을 다운로드 받아 압축을 풀고 디렉토리에 저장한다.



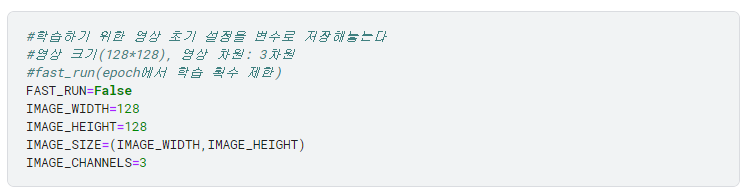
그림[3-8]

학습할 때 뜨는 경고문을 무시하기 위해 warnings 라이브러리를 사용하였다.



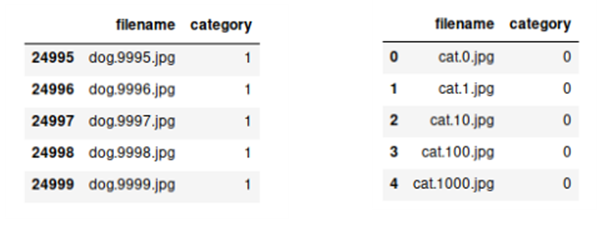
그림[3-9]

이미지 학습 및 생성에 필요한 변수를 미리 작성한다. FAST\_RUN은 epoch에서 학습 횟수를 제한하기 위한 변수다. 앞서 압축을 풀어놓은 이미지의 가로와 세로 길이, 사이즈(128\*128), 채널(컬러 사진이므로 R, G, B 3개)을 정의한다.



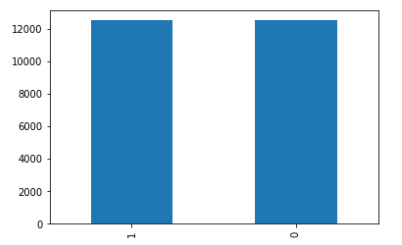
그림[3-10]

압축을 푼 디렉터리 중 train에는 dogxxx.jpg, catxxx.jpg 등 개와 고양이를 파일 이름으로 구분짓도록 이름이 저장되어 있다. 폴더 속 이미지 파일명을 split하여 개는 1, 고양이는 0으로 정답 레이블(category)을 생성한다. 아래는 데이터 셋 상위 5개를 샘플링한 결과다.



그림[3-11]

**3.3.2 데이터 전처리**

이미지 제너레이터에서 class\_mode를 “categorical”로 지정하기 위해 컬럼 카데고리를 문자열로 변경한다. df 안의 카테고리에서 0으로 표시된 것을 'cat'으로, 1로 표시된 것을 'dog'으로 변경한다.

그림[3-12]

train\_test\_split 모듈을 활용해 학습 데이터와 테스트 데이터를 분리한다. test\_size를 20%로 나눈 후 인덱스를 재정렬한다. train\_df와 validate\_df에서 인덱스로 세팅한 열을 df 내에서 삭제한다.

학습 데이터 및 검증 데이터의 형상을 확인하고 batch 크기를 설정한다.



그림[3-13]

학습 데이터의 제너레이터 설정은 신경망 모델의 성능을 높이기 위한 Data augmentation 과정이다. ImageDataGenerator 클래스는 이미지를 학습시킬 때 학습 데이터의 양이 적은 경우 학습 데이터를 조금씩 변형시켜서 학습 데이터의 양을 늘리는 방식이다. 이미지를 사용할 때마다 임의로 변형을 가함으로써 훨씬 더 많은 이미지를 보고 학습하는 것과 같은 효과를 낸다. 이를 통해 과적합(overfitting), 즉 모델이 학습 데이터에만 맞춰지는 것을 방지하고, 새로운 이미지도 잘 분류할 수 있게 한다.

\* rotation\_range = 30 : 지정된 각도(0도에서 30도) 범위 내에서 원본 이미지를 임의로 회전시킴

\* rescale = 1./255 : 값을 0과 1 사이로 변경

\* shear\_range = 0.2 : 층 밀리기 강도 20%

\* zoom\_range = 0.2 : 무작위 확대/축소 범위 20%

\* horizontal\_flip = True : 수평방향으로 뒤집기

\* width\_shift\_range = 0.1 : 지정된 수평 방향 이동 범위 내에서 임의로 원본 이미지 이동시키기

\* height\_shift\_range = 0.1 : 지정된 수직 방향 이동 범위 내에서 임의로 원본 이미지 이동시키기

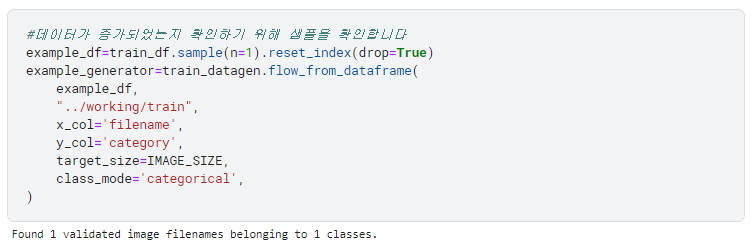
train\_datagen 클래스에 flow 함수를 사용해서 실제 데이터를 파라미터에 넣어주면 이미지 변형이 완료된다.  그림[3-14]

검증 데이터는 훈련 데이터와 다르게 rescale(정규화)만 해준다.



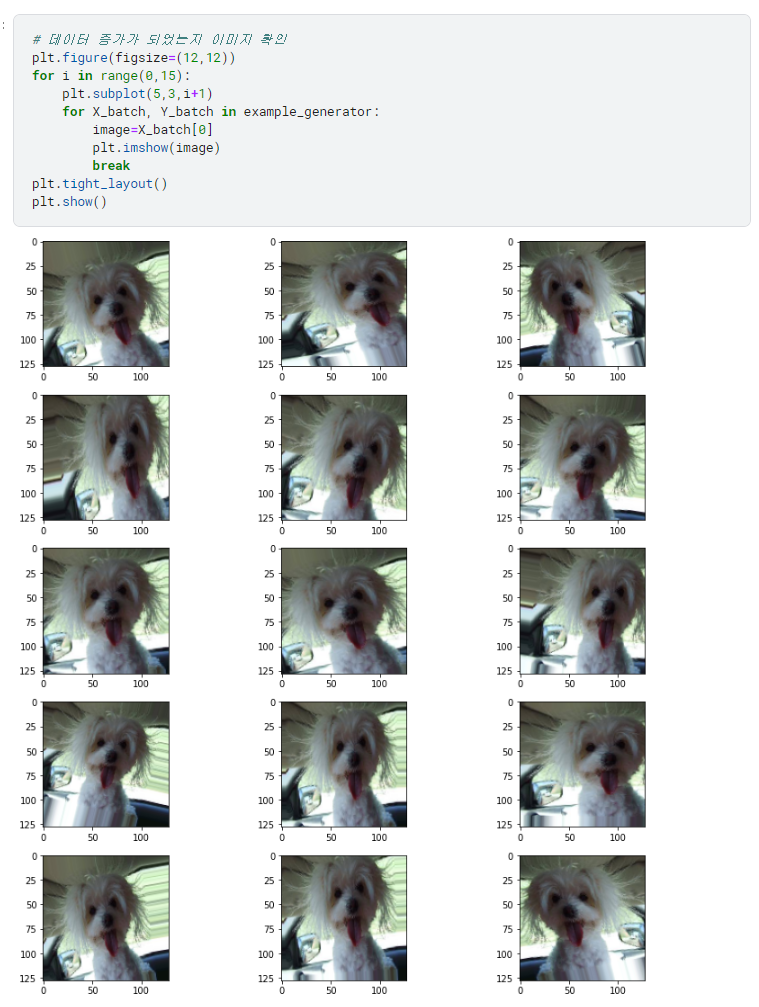
그림[3-15]

샘플 데이터는 그림[3-16]를 통해 확인해볼 수 있다.



그림[3-16]

그림[3-17]은 이미지 데이터를 확인해 본 결과다.

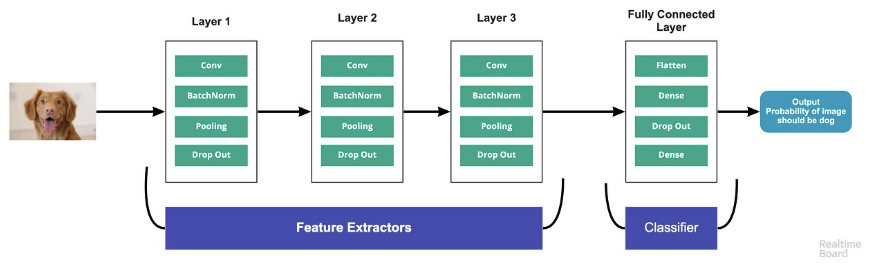


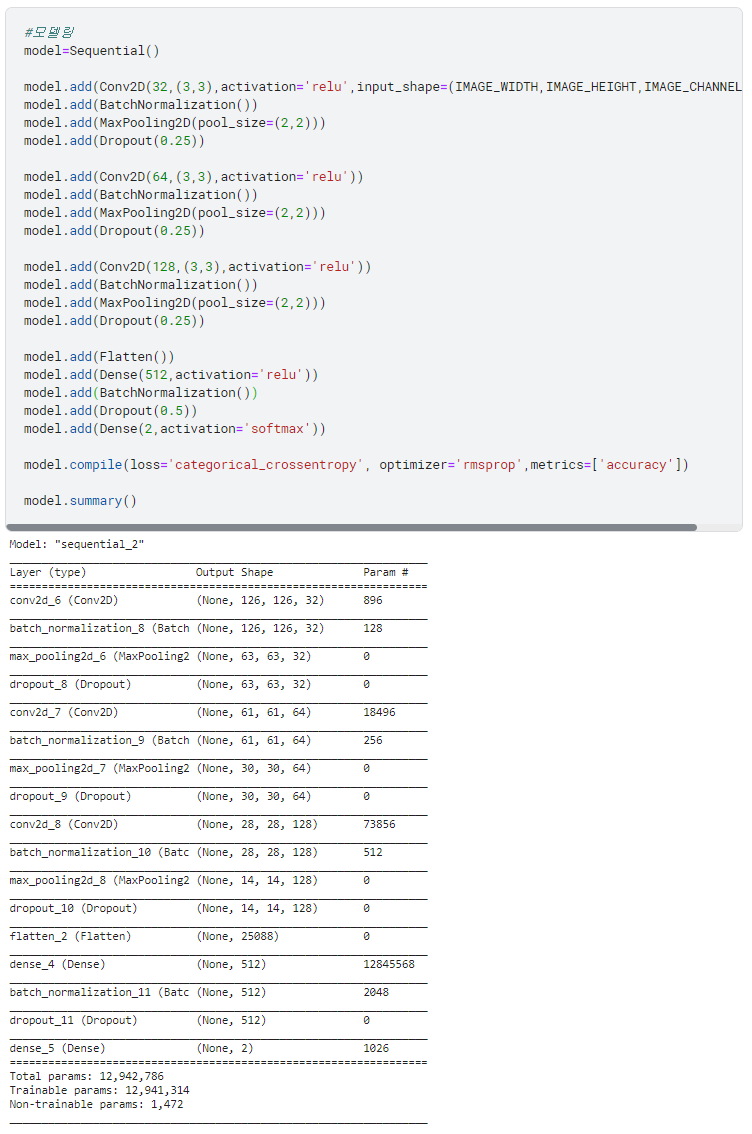


그림[3-17]

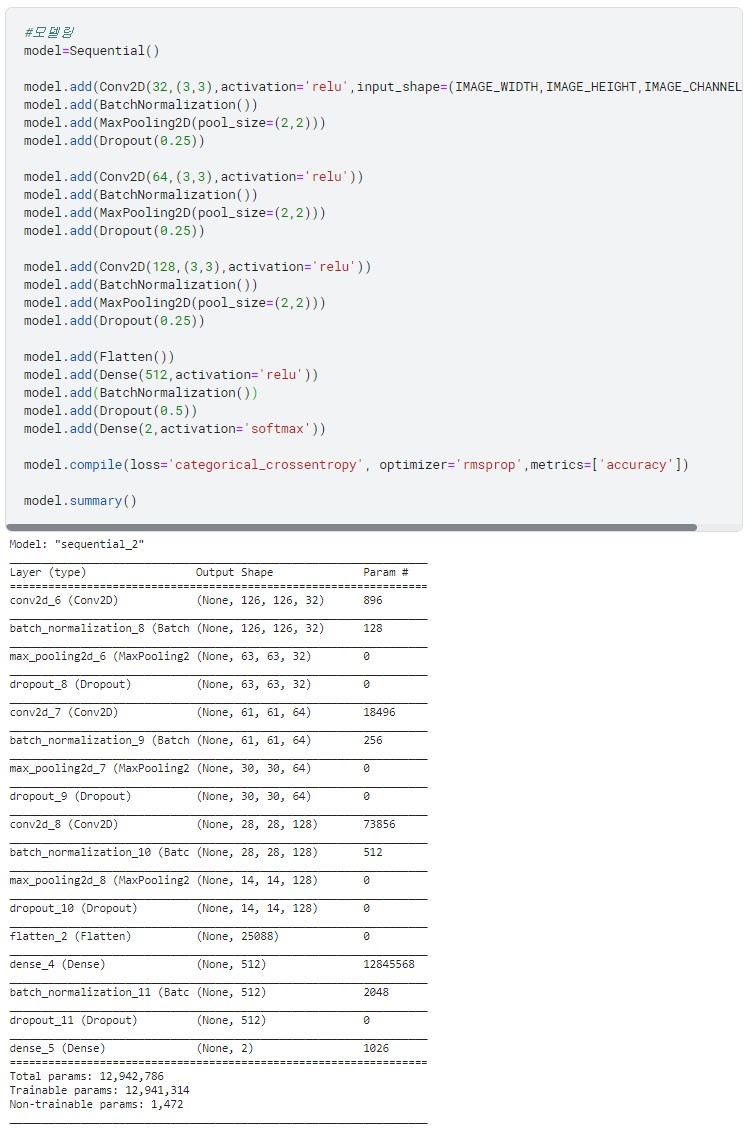
**3.3.2 A2 모델링 및 컴파일**

convolution 레이어 3개를 쌓아 놓은 간단한 형태로 ReLU 활성화 함수를 사용하며, 맥스풀링을 적용한다. 출력층에는 softmax 함수를 이용해 개와 고양이에 대한 퍼센티지를 제공하는 방식으로 분류하는 모델을 생성한다. Sequential 모델은 layer을 선형으로 연결해 구성한다. 가장 자주 사용하는 구조인 layer을 순서대로 쌓아 올린 네트워크다. 입력층의 32는 convolution filter 수이며, (3,3)은 커널 사이즈로 합성곱 커널의 (행,열)을 의미한다. 배치 정규화(Batch Normalization)는 신경망 입력 데이터를 평균: 0, 분산: 1로 정규화를 해서 학습이 잘 이루어지도록 하는 방법이다. 2\*2 filter로 맥스 풀링한 후 과적합을 방지하기 위해 activation을 0.25의 확률로 지운다. 이후 두 세번째 층에서 filter 수가 64, 128로 늘어난다. Flatten() 층에서는 학습되는 가중치가 없고 데이터를 1차원 vector로 변환한다. 하나의 layer에 있는 모든 뉴런을 또 다른 layer의 모든 뉴런과 연결 시켜준다. 이진 분류이므로 출력층의 활성화 함수는 softmax로 지정한다. 각 노드는 현재 이미지가 2개의 클래스 중 하나에 속할 확률을 출력한다. 그림[3-17]을 통해 요약된 정보를 볼 수 있다.

 모델 컴파일 단계에서 손실함수는 다중 클래스 분류에 적합한 categorical\_crossentropy를 사용하고, 옵티마이저는 RMSProp를 사용했다. RMSProp 알고리즘은 과거의 기울기들을 똑같이 더해가는 것이 아니라 먼 과거의 기울기는 조금 반영하고 최신의 기울기를 더 많이 반영해 훈련 과정 중에 학습률을 적절하게 변화시킨다.

****

그림[3-18]



그림[3-19]

과적합을 방지하기 위해 earlystopping을 사용한다. 10번의 epoch 동안 loss 값이 줄어들지 않거나 2번의 epoch 동안 accuracy가 상승하지 않으면 콜백해 학습을 중지시킨다. patience는 오차가 증가하는 즉, ‘개선이 없는 epoch 수’가 patience에 지정된 수만큼 지속될 경우 학습을 종료시킨다. reduceLROnPlateau는 callback 함수의 일종으로, learning rate가 더이상 업데이트 되지 않으면 학습을 중단한다. plateau는 ‘정체기’라는 뜻이며 모델의 정확도가 향상되지 않는 경우 learning rate(lr)를 줄여준다. ReduceLROnPlateau가 수량을 모니터링하며 patience 수의 epoch가 개선되지 않으면 학습률을 감소시킨다.

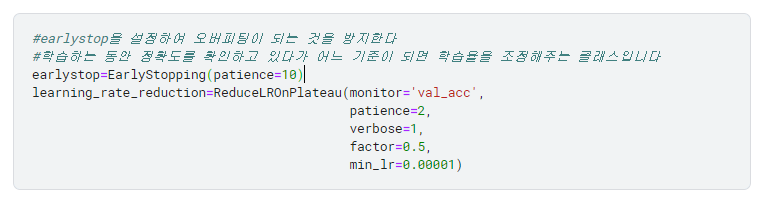
\* monitor: 모니터링 할 수량 (val\_acc: validation accuracy)

\* patience: 학습률이 감소한 후 개선되지 않은 epoch 수

\* verbose: 얼마나 자세하게 정보를 표시할 것인가 지정 (0~1)

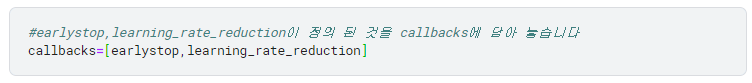
\* factor: 학습률이 감소하는 요인

\* min\_lr: 학습률의 하한



그림[3-20]

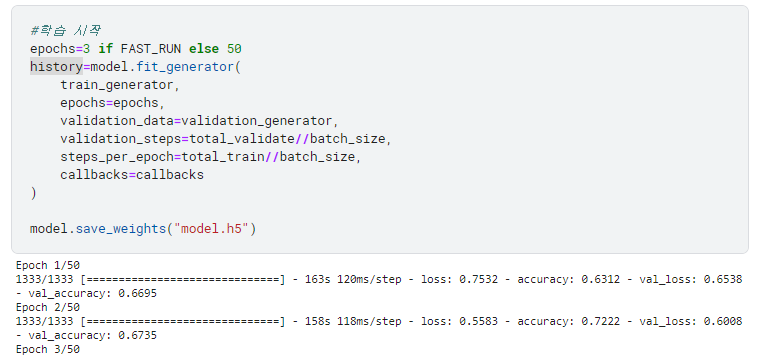
callbacks 변수에 earlystop과 learning\_rate\_reduction 함수를 저장한다.



그림[3-21]

**3.3.2 A2 모델 학습**

대용량 데이터를 학습시킨 뒤 모델을 저장하지 않으면 처음부터 학습을 다시 진행해야만 한다. 따라서 모델 가중치를 h5 파일 포맷으로 만들어 저장한다. 케라스에서는 모델과 가중치의 재사용을 위해 이를 파일 형태로 저장하는 라이브러리를 제공하며, 이를 통해 가중치를 파일 형태로 저장하고 불러올 수 있다.

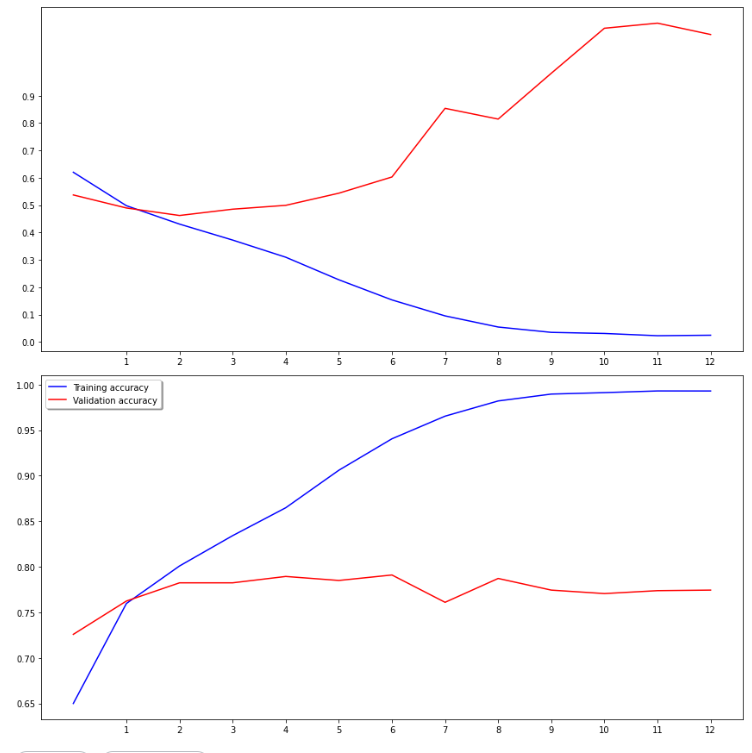


그림[3-22]

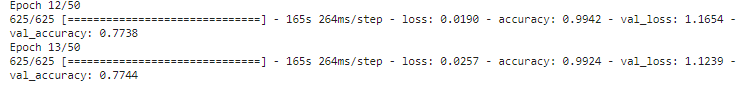
**4. 결과 Results**

A1 모델과 A2 모델은 동일한 데이터로 이루어져 있고, valIdation 값도 동일하다.

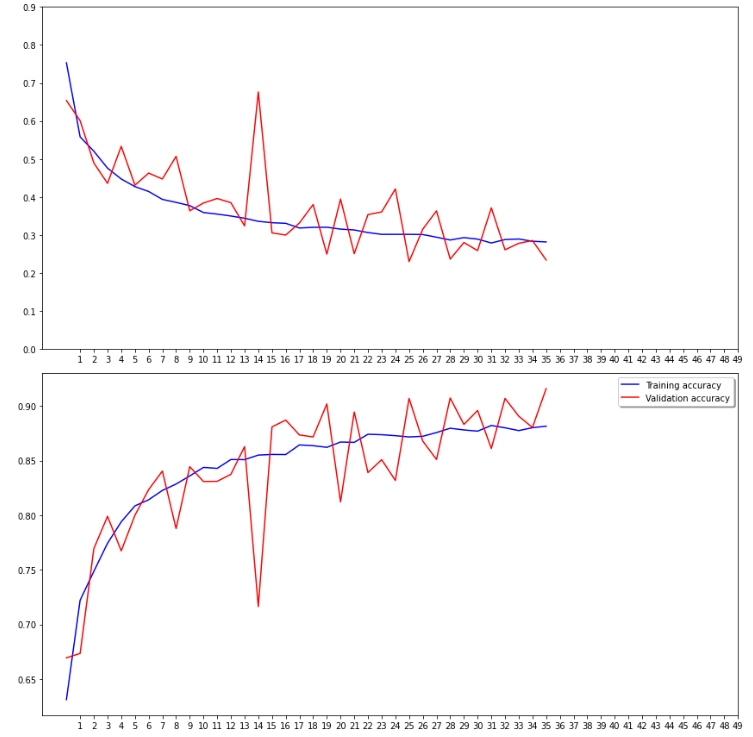
그림[4-1]과 그림[4-2]를 통해 A1의 모델을, 그림[4-3]과 그림[4-4]를 통해 M2 모델의 결과를 확인해볼 수 있다.



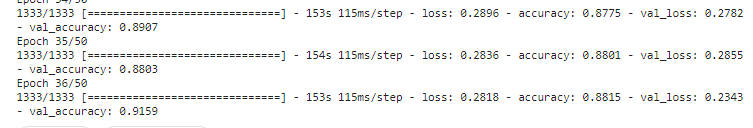
그림[4-1]



그림[4-2]



그림[4-3]



그림[4-4]

**5. 결론 Conclusion**

본 연구는 캐글의 dog vs cat 데이터 셋을 바탕으로 데이터를 전처리한 후, 전처리 한 데이터를 바탕으로 딥러닝 A1, A2 모델을 만들었다. 개와 고양이를 분류 예측하여 높은 성능을 내는 것을 목표로 하였다. A1 기준 알고리즘은 학습 시간을 최대한 줄이는 것에 집중했고, 이 모델에서 발생한 여러 문제점을 보완하며 효율적인 A2 팀별 알고리즘을 개발했다.

A1 모델의 학습 데이터 정확도는 ‘accuracy:0.9’로 높았지만, ‘val\_accuracy:0.77’로 과적합이 이루어진 것을 그림[4-4]를 통해 알 수 있다. A2 모델은 ‘accuracy:0.88’로 학습 데이터의 정확도는 낮으나 ‘val\_accuracy:0.91’로 모델 성능이 향상되었음을 그림[4-4]를 통해 알 수 있다.

향후 과제로 데이터에 더욱 알맞은 전처리 방법을 고안하고, 더 적합한 딥러닝 모델을 공부해 정확도가 더욱 향상될 수 있도록 할 것이다.

**6. 느낀점 Impression**

이채별: 처음 팀 프로젝트를 수행하는 데 있어 어떻게 진행해야 될지 막막했다. 학습해야 할 이미지 데이터의 양이 많아 학습 시간도 6시간 이상으로 오래 걸렸으며, CNN 모델을 어떻게 구성하느냐에 따라 성능이 천차만별이였기 때문에 모델을 선정하는 과정에 있어 어려움을 느끼기도 했다. 하지만 팀원들과 회의를 통해 부족한 부분은 서로 알려주고, 중요한 사항을 결정할 때는 충분히 상의해보면서 어려움을 극복할 수 있었다. 매주 시간이 지날 때마다 발전되는 알고리즘 성능에 뿌듯함을 느꼈다. 개인 프로젝트가 아니라 오히려 팀 프로젝트였기 때문에 좋은 결과물을 낼 수 있었다고 생각한다. 이번 프로젝트를 통해 인공지능에 대한 흥미도가 더욱 높아진 것 같다. 겨울 방학 때는 사람의 얼굴 이미지를 분류하는 딥러닝 알고리즘을 만들어보고 싶다.

홍민재: 1학기 때 기계학습기초 수업을 들은 경험이 있어 딥러닝에 대한 공부도 하고 싶어 딥러닝기초 수업을 들었는데, 기계학습기초 수업보다 더 어려운 것 같아서 당황스러웠다. 모델링에 대한 이해도가 내 자신도 부족하다고 느꼈고, 책에서도 심화해 서술하지는 않은 것 같다. 인공지능에 대해 완벽하게 이해하려면 수학 능력이 매우 높아야 한다고 느꼈고, 재미있고 주목받는 기술인만큼 열심히 노력해야겠다는 생각이 들었다.

김준수: 개와 고양이를 비교하는 딥러닝 프로그램 A1을 만들면서 배운 내용만으로는 부족하다는 것을 알게 되었다. 조금 더 딥러닝에 대해 자세히 공부해보고, 만든 A1에 대한 개선점을 보완하며 만든 A2를 개발하면서 인공지능에 대한 지식을 알아야한다면 책의 내용보다 더 깊게 들어가야한다는 것을 깨닫게 되었다.

정원용: 1학기 기계학습 때의 팀 프로젝트는 팀원들의 참여율이 적어 아쉬웠던 기억이 있었는데, 이번 딥러닝기초에서는 경험이 많으신 팀원분들과 참여도 다들 적극적으로 해주셔서 많은 것을 보고 배울 수 있었다. 하나의 딥러닝 코드를 보고 여러 관점으로 토론이 진행 되니까 비록 딥러닝이라는 과목 자체에 탐구하 것이 정말 많이 남아 있구나라고 생각했고, 어려움을 느끼면서도 하나씩 알아가는 과정에 즐거움을 느낄 수 있었다. 팀 과제를 진행하면서 ‘이 부분은 더 개선할 수 없을까?’ 라는 의문이 많이 생겼는데, 앞으로는 이 의문을 해결하는 방향으로 공부해 나가야겠다고 느꼈다.

**7. 참고자료 Reference Material**

우준세, <Python – OpenCV (11) : Grayscale 함수 비교>, <<준세 단칸방>, 2021. 07. 16,

<https://wjunsea.tistory.com/76>

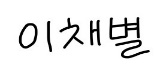
혜민스님, <[Deep Learning] CNN : Convolutional Neural Network - 강아지와 고양이 이미지 분류 [Colab]**>**, 2020. 03. 08,

<https://blog.naver.com/PostView.naver?blogId=gogsally&logNo=221842733299&parentCategoryNo=&categoryNo=39&viewDate=&isShowPopularPosts=true&from=search>

**회 의 록**

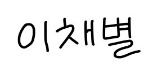
|  |  |
| --- | --- |
| **회의일시** | **날짜 : 2022년 11월 20일 (일요일)**  **시간 : 21:00 ∼ 22:30** |
| **회의 목표** | **AI 모델 선정** |
| **참 석 자** | **이채별, 홍민재, 정원용**  **(김준수 – 주말 아르바이트로 인해 불참)** |
| **-회의내용** | **회의 시작 전 각자 구현한 A1 모델 소스 코드를 공유했다.**  **한 명씩 A1(기준 알고리즘) 모델 구현 과정에 대해 발표하고,**  **가장 정확도가 높은 모델을 선정하였다. (홍민재 학생 모델)**  **과적합을 방지하기 위한 earlystopping 추가 등 알고리즘 성능 향상을**  **위한 모델 개선이 필요해 보인다.**   |  |  |  | | --- | --- | --- | | **이채별** | **홍민재** | **이준수** | |  |  |  | |

**2022년 11월 20일**

**작성자 : 이채별 (서명**

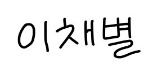
|  |  |
| --- | --- |
| **회의일시** | **날짜 : 2022년 11월 27일 (일요일)**  **시간 : 21:00 ∼ 23:00** |
| **회의 목표** | **A2 모델 선정** |
| **참 석 자** | **이채별, 홍민재, 정원용**  **(김준수 – 주말 아르바이트로 인해 불참)** |
| **-회의내용** | **회의 시작 전 각자 구현한 A2 모델 소스 코드를 공유했다.**  **한 명씩 A2(기준 알고리즘) 모델 구현 과정에 대해 발표하고,**  **가장 정확도가 높은 모델을 선정하였다. (이채별 학생 모델)**  **학습 횟수를 더 늘리는 방식으로 모델을 개선해나가기로 했다.**  **앞으로 진행 할 보고서 작성, ppt 작성, 알고리즘 보완으로 파트를 나누었다.**   |  |  |  | | --- | --- | --- | | **이채별** | **홍민재** | **이준수** | |  |  |  | |

**2022년 11월 27일**

**작성자 : 이채별 (서명**

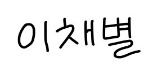
|  |  |
| --- | --- |
| **회의일시** | **날짜 : 2022년 12월 2일 (금요일)**  **시간 : 22:00 ∼ 02:30** |
| **회의 목표** | **알고리즘 보완 및 보고서/ppt 작성** |
| **참 석 자** | **이채별, 홍민재, 정원용, 김준수** |
| **-회의내용** | **김준수 학생은 A1 알고리즘을 보완하고 ppt 초안을 작성 하였으며, 정원용 학생은**  **A2 모델에 submission 파일을 생성할 수 있는 소스 코드를 작성하였다.**  **홍민재 학생과 이채별 학생은 A1과 A2 알고리즘을 나누어 보고서를 작성하였다..**  **각자 분업화를 하며 팀원이 다같이 상의해 결정해야 될 사항에서는 음성 통화**  **를 통해 피드백을 해주며 결과물을 완성해 나갔다.**  **ppt와 보고서 같은 경우 각자 보완하여 3일 토요일 회의에서 최종으로 마무**  **리 하기로 결론 지었다.** |

**2022년 12월 2일**

**작성자 : 이채별 (서명**

|  |  |
| --- | --- |
| **회의일시** | **날짜 : 2022년 12월 3일 (토요일)**  **시간 : 22:00 ∼ 24:00** |
| **회의 목표** | **최종 발표 자료 보완 및 발표 준비** |
| **참 석 자** | **이채별, 홍민재, 정원용, 김준수** |
| **-회의내용** | **최종 발표 자료를 수정하고, 발표 파트를 분배하였다.**  **(A1 모델 발표: 홍민재, A2 모델 발표: 정원용, 이채별, 개선점 발표: 김준수)**  **발표를 어떻게 할지 개인적으로 연습할 시간을 충분히 가진 후,**  **발표 시연을 해보고 회의를 마무리 하였다.** |

**2022년 12월 3일**

**작성자 : 이채별 (서명**