**인공지능 이미지 분류도구**

세원아토스 기술연구소

조영탁

1. **목차**
   1. 목차
   2. 개요
   3. 방법론
   4. 학습치 시각화와 정확도 개선
   5. 결과
   6. 시스템 환경
2. **개요**

본 프로젝트는 패션 이미지를 CNN(Convolutional Neural Network)으로 분류하여 새로운 이미지에 대하여 인공지능이 자동으로 복종을 구분하는 모듈이다.

1. **방법론**
   1. **문제파악**

본 프로젝트(DEMO)에서는 세원아토스의 자사 쇼핑몰 모어댄베러-셀릭에 등록된 6개 카테고리로 분류(Classification) 하도록 설계하였다.

*Class*

Dress(원피스), Outer(아우터), Pants(하의), Skirt(치마), Suit(투피스), Top(상의)

6개 카테고리를 분류하기 위한 학습 데이터를 얻기 위해 파이썬의 라이브러리 Beautiful Soup 4, Selenium, tqdm, json, glob 등을 이용하여 DACT를 개발하였다.

* 1. **Dataset Crawling**

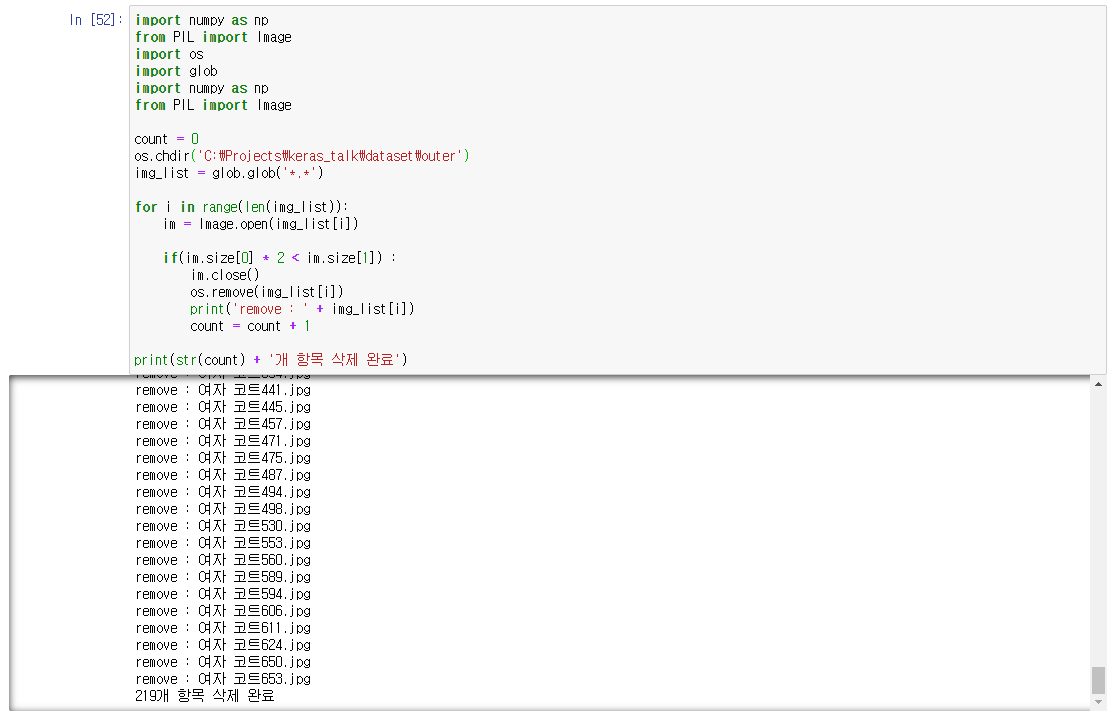
학습에 필요한 패션 이미지는 DACT 3.0을 이용하여 구글 크롤링하였다.

|  |
| --- |
| Top |
| '여자 니트', 'tops for ladies', '여자 맨투맨', '여자 민소매', '여자 블라우스', '여자 셔츠', '여자 티셔츠', 'lady t-shirt', '여자 상의' |
| Outer |
| '여자 아우터', 'ladies outerwear', '여자 가디건', '여자 자켓', '여자 점퍼', '여자 집업', '여자 조끼 베스트', '여자 코트', '여자 외투' |
| Dress |
| '여자 원피스 -수영복', 'dress', '여자 롱 원피스', '여자 미니 원피스', '여성 원피스', '오버롤 원피스', '여자 드레스', '여성 원피스 패션', '원피스 신상' |
| Pants |
| 'women pants', '여성 바지', '여성 슬랙스', '여성 청바지', '여성 코튼팬츠', '여자 코튼팬츠', '여자 슬랙스', '여성 신상 팬츠', '여성 와이드팬츠' |
| Skirt |
| 'lady skirts', 'women skirt', '롱 스커트', '숏 스커트', '스커트 신상', '여성 스커트', '여성 치마', '여자 스커트', '치마 신상' |
| Suit |
| '여자 투피스 수트', '여성 수트', 'women suit', '여자 투피스 정장', '여성 정장', '여자 정장', '여자 블레이저', '여성 블레이저', 'lady formal suit' |

총 32,474개 이미지를 수집하였으며, Crawling 키워드는 다음과 같다.

* 1. **Data Cleansing**

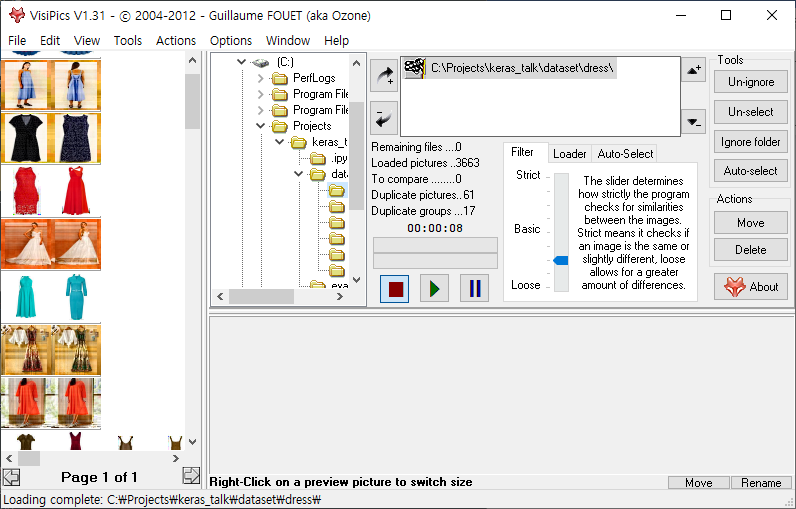
*Deleter*



데이터 클렌징은 1차로 가로 \* 세로 배열을 비교하여 비정상적 배열크기를 가진 이미지를 삭제하는 파이썬 모듈을 개발하여 진행하였다. deleter는 지정한 Dataset 폴더에 접근해 모든 이미지를 불러온(glob) 후 이미지의 가로, 세로 픽셀(im.size[0](http://solution.sewonatos.co.kr:3000/issues/3909#fn0), im.size[1](http://solution.sewonatos.co.kr:3000/issues/3909#fn1))을 비교하고 im.size[0](http://solution.sewonatos.co.kr:3000/issues/3909#fn0) \* 2 < im.size[1](http://solution.sewonatos.co.kr:3000/issues/3909#fn1)면 삭제하는 자동화 툴이다. DACT 3.0을 이용해 가져온 이미지 중 세로 크기가 비정상적으로 긴 이미지는 resize를 하더라도  
학습에 적합하지 않으며 데이터의 질이 좋지 않을 가능성이 크기 때문에 삭제한다.

Deleter를 사용하여 디렉토리 당 평균 200~300개의 오류 이미지 데이터를 추려냈다.

*Visipics*



이후 Visipics를 이용하여 이미지 배열을 검사하고 중복되는 데이터를 자동으로 삭제한다.

*Hand Cleansing*



이후 디렉토리에서 직접 이미지를 수작업으로 분류하여 오류 이미지를 삭제하였다. 초기 32,191개 이미지에서 10,010개 이미지를 수작업으로 분리하여 22,181개 데이터셋을 구축 완료하였다.

* 1. **Image Pre Processing(이미지 전 처리)**

*Filter out corrupted images*



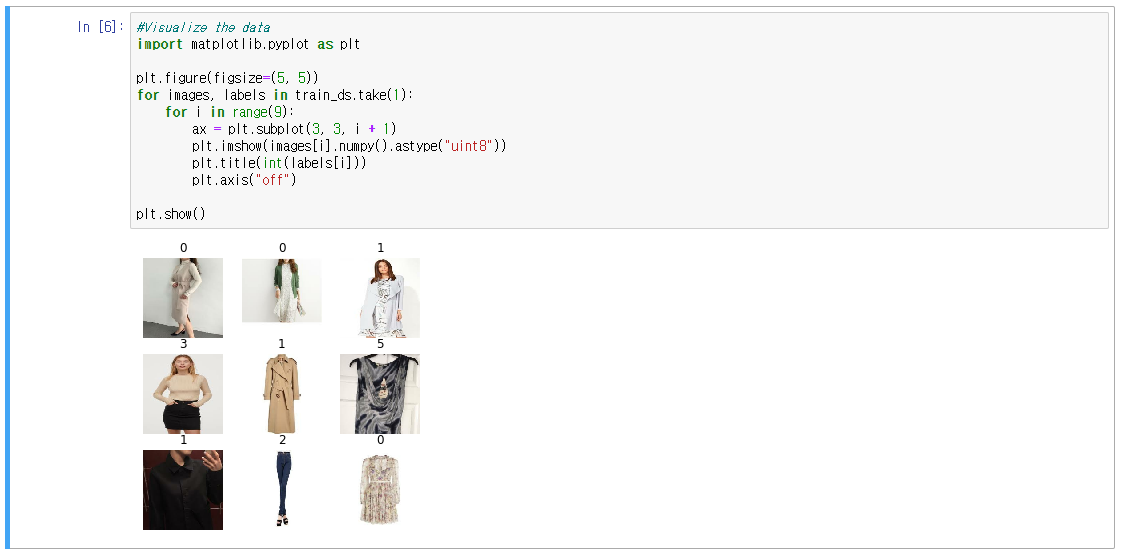
학습에 사용할 Image Dataset 디렉토리에 접근하여 파일형식을 확인하고 오류가 있는 이미지를 학습에서 제외한다.

*Generate a Dataset*



tf.keras.preprocessing.image\_dataset\_from\_directory 기능을 이용하여 Image Dataset 디렉토리에서 Dataset을 자동으로 가져오고 Train set, Test set으로 자동으로 나눈다. 이미지 사이즈는 (180, 180)으로 조정하고 배치 사이즈는 32로 지정하였다.

*Visualize the data*



이후 데이터 시각화 테스트를 진행하였다. Plt 라이브러리를 이용해 랜덤 이미지와 labels 데이터를 불러와 그리드하여 표시하였다. 정상적으로 결과가 출력되었다.

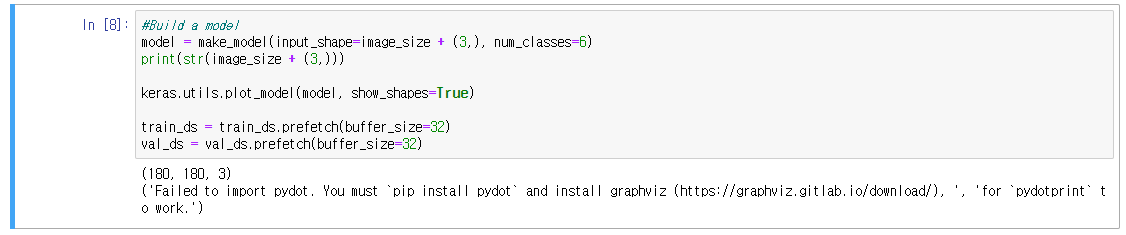
*image data augmentation*



데이터를 학습시키기 이전에 두 가지 문제를 해결해야 했다. 첫 번째는 데이터의 개수가 많지 않은 것, 두 번째는 옷의 촬영 각도가 일정하지 않을 것. 이 문제를 해경하기 위해 Keras에서 제공하는 Preprocessing의 RandomFlip을 이용하여 랜덤하게 좌우 반전을 실행하고, RandomRotation을 이용하여 이미지의 각도를 돌려보았다.

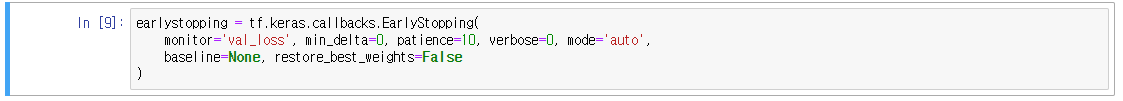
패션 이미지는 대부분 수평을 이룬 각도로 촬영하기 때문에 많은 값을 회전시키면 오히려 좋지 않다. 실제로 0.1의 값을 Rotation 한 결과 정확도가 기존보다 떨어졌다.

*Build a model*



다음으로 모델을 디자인하였다. Make\_model 함수를 이용하여 모델을 만드는데 이미지의 형태는 3차원 이미지(RGB)로 지정하고 num\_classes는 6으로 총 6가지 분류를 제공하도록 디자인하였다.

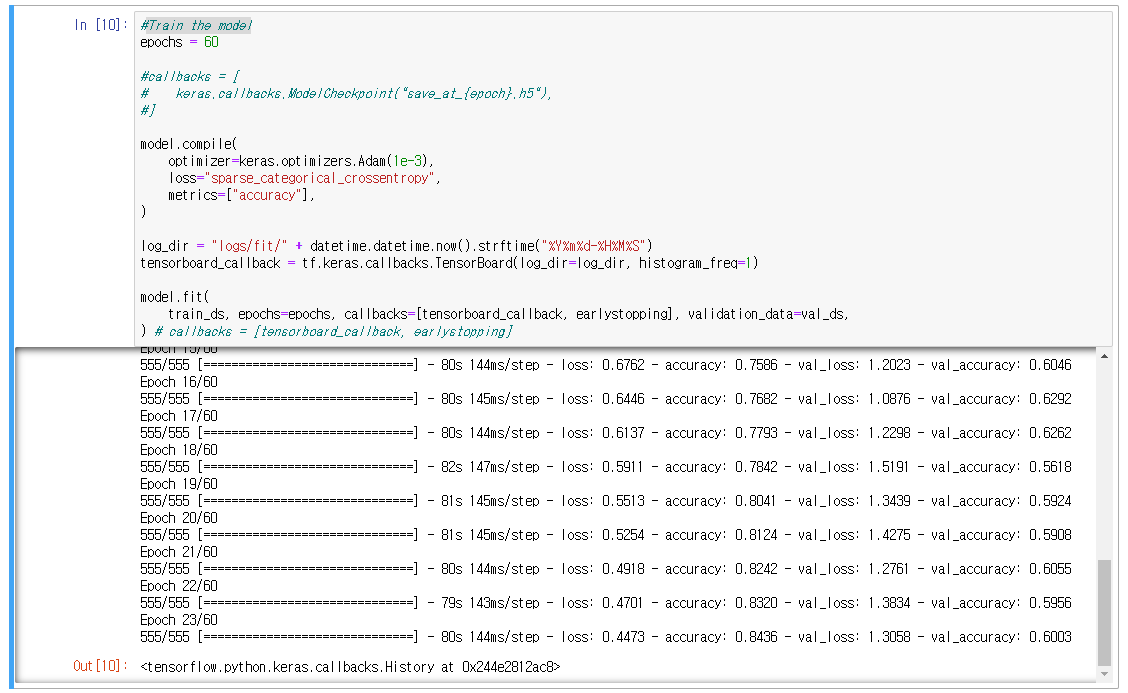
*Earlystopping*



데이터 학습 이전에 Overfitting(과적합)을 방지하기 위해 EarlyStopping Callback 함수를 디자인하였다. 검증 데이터셋의 손실값을 모니터링하며 과적합이 일어나는 지점을 자동으로 찾아 학습을 중단하도록 설정했다. 최소 10회의 Epoch를 학습하고 학습치 delta값이 0만큼 변동되면 중단지점으로 판단하여 학습을 중단한다.

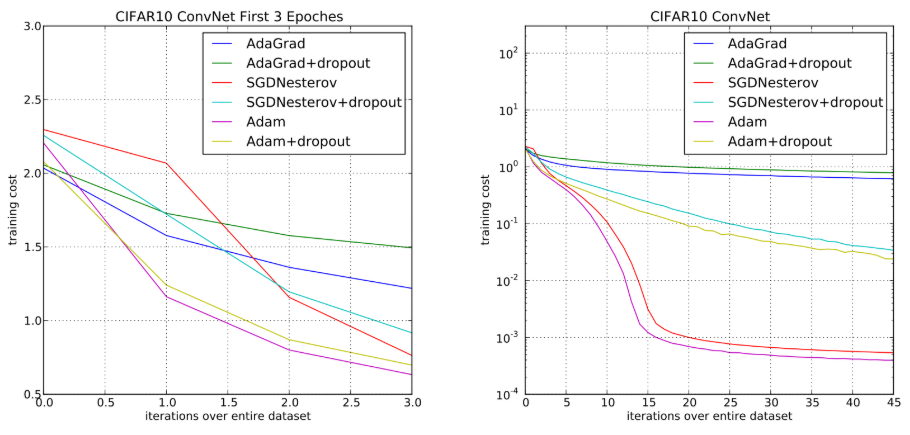
머신 러닝 (machine learning)에서 overfitting은 학습데이터를 과하게 잘 학습하는 것을 뜻한다. 일반적으로 학습 데이터는 실제 데이터의 부분집합인 경우가 대부분이다. 따라서, 아래의 그래프처럼 학습 데이터에 대해서는 오차가 감소하지만, 실제 데이터에 대해서는 오차가 증가하는 지점이 존재할 수 있다.

*Train the model*



학습 Epoch는 60으로 지정하였다. 신경망 옵티마이저는 Adam Optimization을 이용하였다. Adam method는 Adagrad + RMSProp의 장점을 섞어 놓은 것이다. Adam method의 의 주요 장점은 stepsize가 gradient의 rescaling에 영향 받지 않는다는 것이다. gradient가 커져도 stepsize는 bound되어 있어서 어떠한 objective function을 사용한다 하더라도 안정적으로 최적화를 위한 하강이 가능하다. 게다가 stepsize를 과거의 gradient 크기를 참고하여 adapted시킬 수 있다

*ADAM CNN*

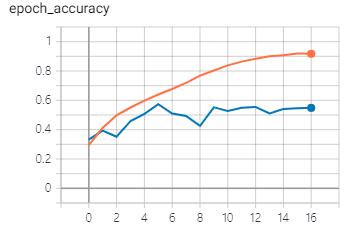
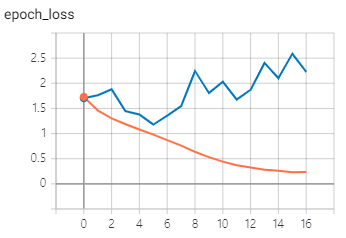


본 프로젝트에서 이미지 분류에 사용하는 CNN 기법에도 ADAM Optimization이 좋은 성과를 내는 것을 확인할 수 있다.

1. **학습치 시각화와 정확도 개선**

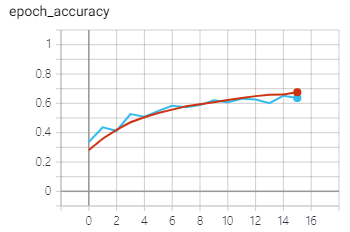
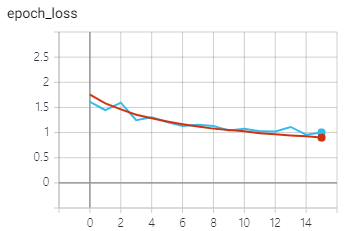
Jupyter notebook 환경에서 %tensorboard --logdir logs/fit명령을 실행하여 콜백함수에 저장된 학습 매개변수 그래프를 그려보았다.

*1차 학습*

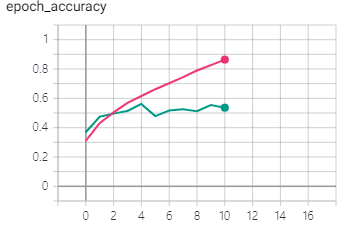
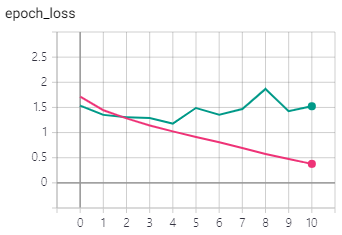
학습 결과 만족스러운 결과가 나타나지 않았다. Train의 경우 정확도가 90%이상을 상회하였지만 검증데이터에 대하여 정확도가 50프로를 조금 넘는 수준이였다. 이유는 5회차 학습에 과적합이 발생하였다. 이를 해결하기 위해 학습 조기중단(early stopping)를 도입하였다. 또한 데이터셋의 클렌징 작업이 이루어지지 않아 잘못된 데이터가 디렉토리에 들어있었다. 이를 해결하기 위해 데이터를 클렌징하였다.

*2차 학습*

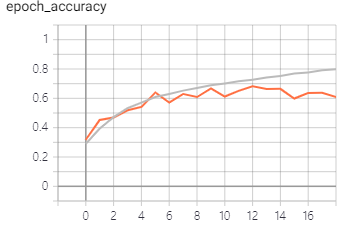
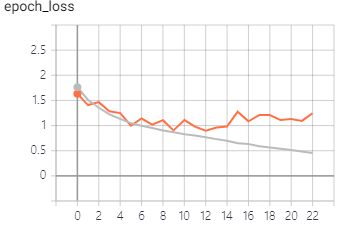
데이터를 클렌징하고 학습 조기중단을 사용하여 정확도를 크게 올릴수 있었다. 15회 학습하며 오버피팅이 일어나지 않았지만 손실률이 높고 정확도는 70% 이하 수준이였다.

*3차 학습*

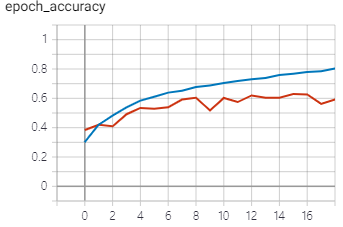
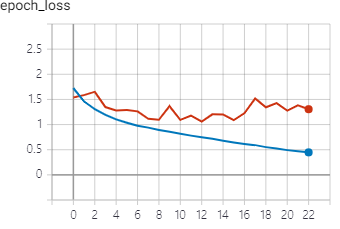
효율적인 데이터 전처리가 진행되는지 알아보기 위해 preprocessing 기능을 off하고 기본 데이터로만 학습 하였다. 결과적으로 Train에 대한 성과는 눈에띄게 좋아졌으나, Vailidation set의 정확도가 떨어지고 과적합이 발생하였다. 이후 Early Stopping 함수를 수정하여 더욱 민감하게 반응하도록 수정하였다.

*4차 학습*

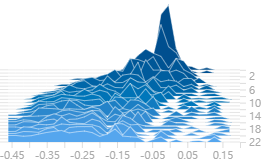
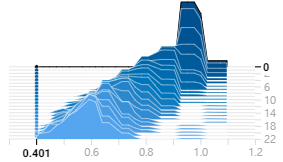
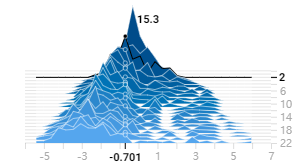
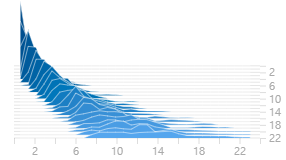
Preprocessing을 수정하였다. 기존에 0.1의 값만큼 Rotate하던 전처리를 0.02로 수정하여 패션이미지에 최적화되도록 수정하였다. 또한, 추가로 클렌징을 시도하여 정확도를 조금 끌어올렸다. 하지만 early Stopping이 정확히 동작하지 않아 과적합 현상이 발생하였다. Early Stopping을 정확히 연구하여 다시 시도하였다.

*5차 학습*

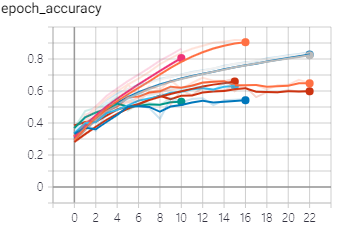
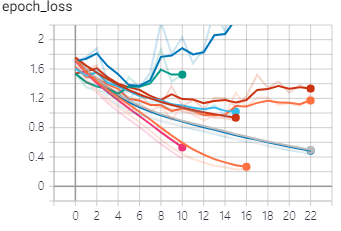
 

바지 클래스에서 잦은 오류가 발생하는 것을 확인하고 클렌징을 시도하였다. Early Stopping이 정상적으로 작동하여 학습이 적절하게 조기종료 되었으며 22 Epoch까지 학습을 진행하였다. 기존에 학습 주기가 짧았던데 비해 더 길게 학습하였고 정확도가 높아졌으며 loss도 줄었다.

총 5회까지 정확도를 높이는 연구를 진행하며 CNN 분류의 정확도를 향상시키는 방법을 조금 알게 되었다.

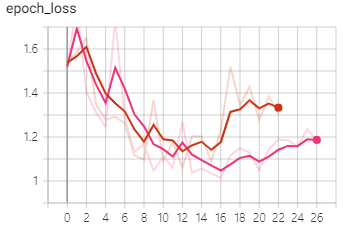
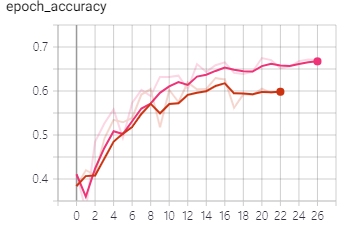
   

다음 4개의 히스토그램을 확인할 수 있었다.

5회의 학습 개선 이력의 그래프는 다음과 같다.

*5차 학습*



기존 Dress와 Skirts 구분에 문제를 인식하고 중점적으로 데이터 클렌징 작업을 실행하였다. 6차 학습 결과 5차학습 대비 크게 정확도를 상승시켰다.

1. **결과**

****

*5차 학습 결과*

Validation Image 외에 새로운 이미지로 신경망이 잘 작동하는지 확인하기 위해 모어댄베러의 상품 이미지를 입력하였다. 5차 학습 결과는 다음과 같이 skirt와 dress를 정확히 구분해내지 못했다.



*6차 학습 결과*



**Skirt 85.93**

6차 학습 결과 dress와 skirt를 정확히 구분하였다. 해당 사진에서 보이고자 하는 상품이 skirt임을 정확히 찾아내고, 적은 확률로 dress이거나 top일 가능성을 염두에 두고 있다.

1. **시스템 환경**

*기본환경*

디바이스 이름 : DESKTOP-PM0CJFG  
프로세서 : Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz 3.60GHz  
설치된 RAM : 32GB  
디스플레이 어댑터 : NVIDIA GeForce RTX 2080 SUPER  
메인보드 : GIGABYTE Z390 AORUS ELITE  
SSD : 삼성전자 PM981 512GB NVMe  
POWER : GreatWall E750 80PLUS GOLD  
CPU 쿨러 : DEEPCOOL GAMER STORM CASTLE 360EX

*Windows 사양*

에디션 : Windows 10 Pro  
버전 : 1903  
설치 날짜 : 2020-04-29  
OS 빌드 : 18362.778

*GPU Driver*

Geforce Experience Game Ready Driver  
CUDA 10.1, CUDNN, cudart64\_101

*사용 기술*

AI : TensorFlow, Keras

Language : Python 3.8.2

ETC : Numpy, PLT, Tensorboard