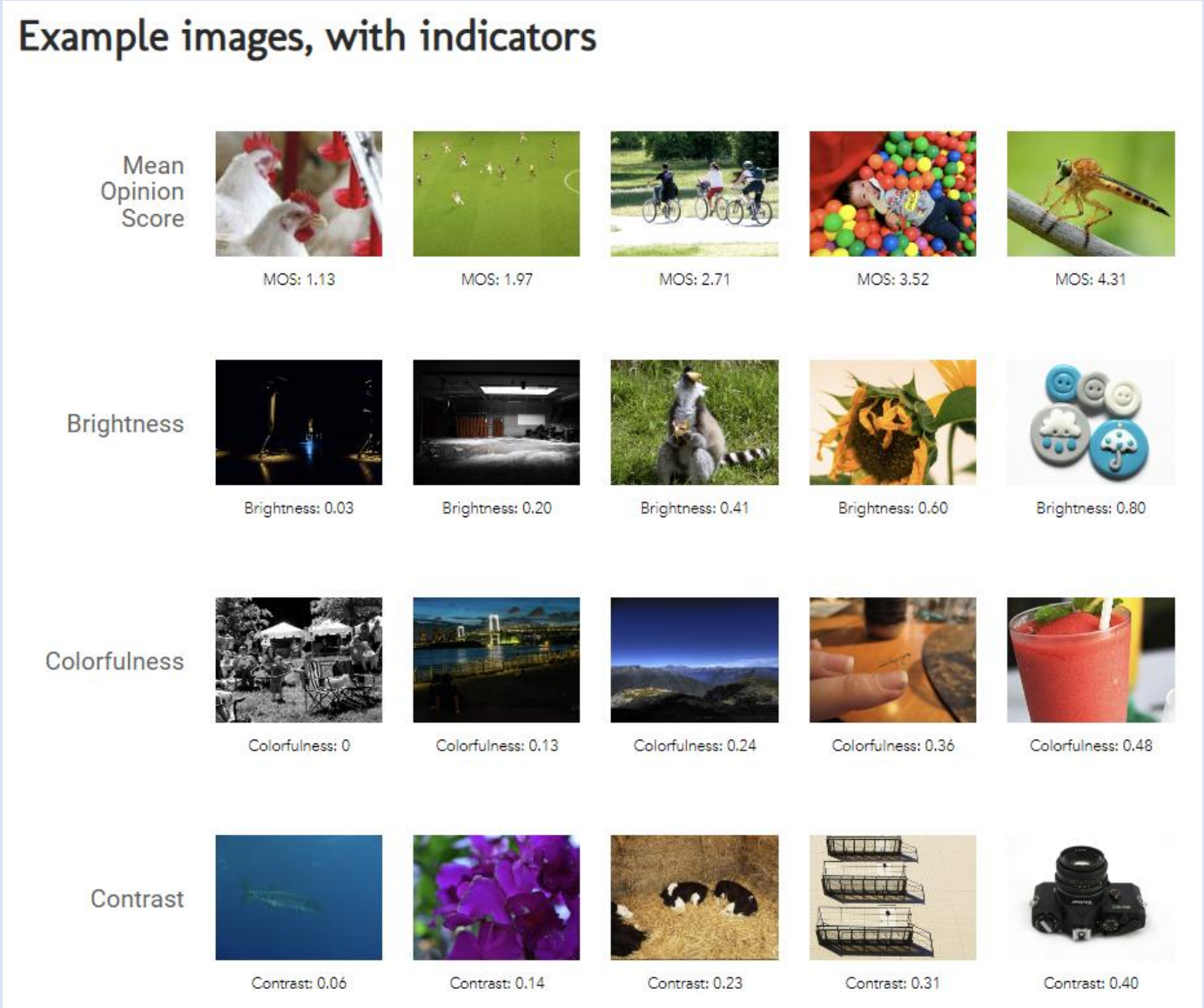
FastCampus
For Business

Blindly Assess Image Quality in the Wild Guided by A Self-Adaptive Hyper Network

[Implementation]

1. Dataset

- **Konlq-10k**



1. Dataset

- **KonIq-10k**
  - **총 10,073장의 이미지**
  - **512x384 px**
  - **우리가 자주 보는 사진 이미지와 유사하다.**

TABLE 1: Comparison of existing IQA databases with KonIQ-10k.

Database	Year	Content	No. of distorted images	Distortion type	No. of distortion types	No. of rated images	Ratings per image	Environment
IVC [17]	2005	10	185	artificial	4	185	15	lab
LIVE [18]	2006	29	779	artificial	5	779	23	lab
TID2008 [19]	2009	25	1,700	artificial	17	1,700	33	lab
CSIQ [20]	2009	30	866	artificial	6	866	5∼7	lab
TID2013 [21]	2013	25	3,000	artificial	24	3,000	9	lab
CID2013 [22]	2013	8	474	authentic	12∼14	480	31	lab
LIVE-itW [23]	2016	1,169	1,169	authentic	N/A	1,169	175	crowdsourcing
Waterloo Exploration [24]	2016	4,744	94,880	artificial	4	0	0	lab
MDID [25]	2017	20	1,600	artificial	5	1,600	33∼35	lab
KADID-10k [26]	2019	81	10,125	artificial	25	10,125	30	crowdsourcing
KonIQ-10k	2018	10,073	10,073	authentic	N/A	10,073	120	crowdsourcing

- **http://database.mmsp-kn.de/koniq-10k-database.html**

1. Dataset

- **KonIq-10k**
  - **koniq10k_indicators.csv**

	A	B	C	D	E	F	G	H	I
1	image_id	brightness	contrast	colorfulness	sharpness	quality_factor	bitrate	hxw	deep_feature
2	826373	0.390904834	0.253917609	0.432910646	22.76193439	79	0.576666667	3145728	100
3	2017266	0.268971143	0.184486318	0.274606168	15.92623475	73	0.247070313	3145728	10
4	2190310	0.046951947	0.126259043	0.096060146	8.148866586	90	0.121119748	5038848	42
5	2704811	0.44946323	0.310170134	0.365717795	7.452023631	97	0.340017731	4255763	188

  - **koniq10k_scores_and_distributions.csv**

	A	B	C	D	E	F	G	H	I	J
1	image_name	c1	c2	c3	c4	c5	c_total	MOS	SD	MOS_zscore
2	10004473376.jpg	0	0	25	73	7	105	3.828571429	0.527277894	77.38362069
3	10007357496.jpg	0	3	45	47	1	96	3.479166667	0.580003025	68.72857143
4	10007903636.jpg	1	0	20	73	2	96	3.78125	0.527219619	78.62857143

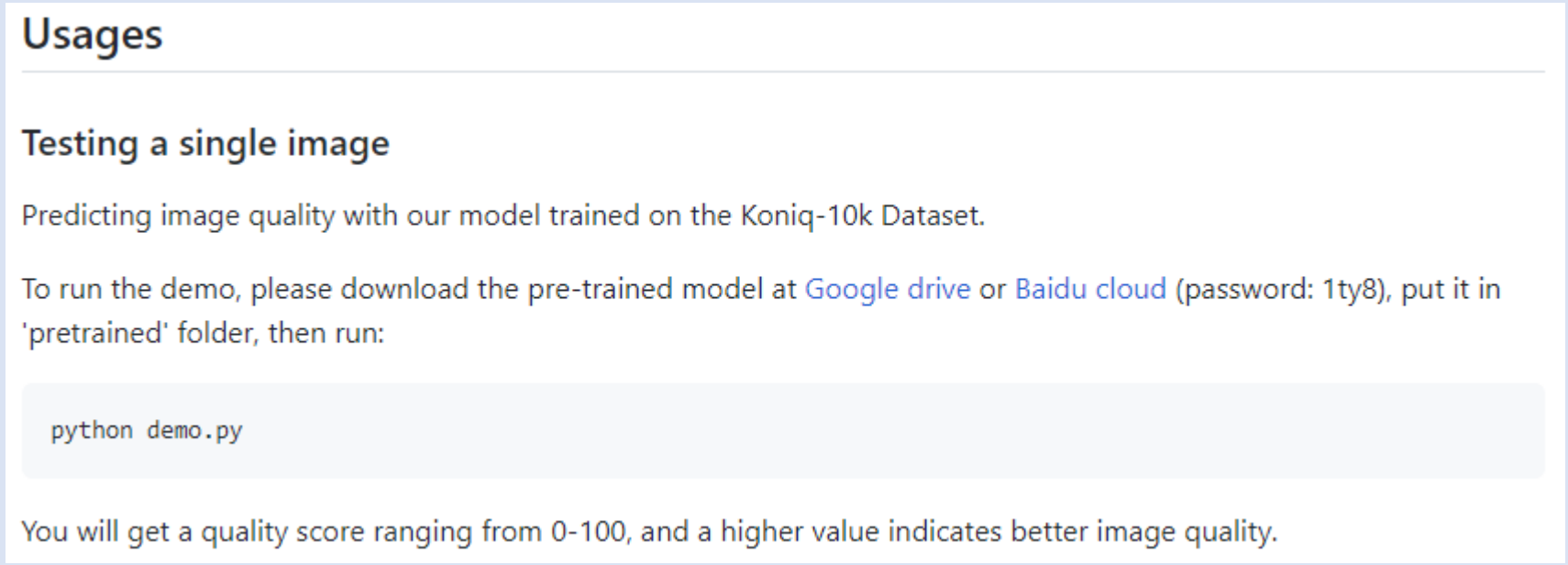
1. Dataset

- KonIq-10k
  - The scores file header
    - image_name : the image file name
    - c1-c5 : number of ratings for each ACR value
    - c_total : total number of judgments per image
    - MOS : Mean Opinion Scores of the 5-point ACR
    - SD : Standard Deviation of the MOS
    - MOS_zscore : ACR scores are first normalised for each user by z-scoring the user ratings

	A	B	C	D	E	F	G	H	I	J
1	image_name	c1	c2	c3	c4	c5	c_total	MOS	SD	MOS_zscore
2	10004473376.jpg	0	0	25	73	7	105	3.828571429	0.527277894	77.38362069
3	10007357496.jpg	0	3	45	47	1	96	3.479166667	0.580003025	68.72857143
4	10007903636.jpg	1	0	20	73	2	96	3.78125	0.527219619	78.62857143

1. Dataset

- **KonIq-10k**
  - **데이터셋 선정 이유**
    - **원본 이미지가 없고 Authentic image이기 때문에 NR-IQA에 적합하다.**
    - **이미지 개수가 많고(약 10,000장) labelling이 잘되어 있다.**
    - **HyperIQA 모델이 KonIq-10k으로 pretrained 되어 있다.**



Usages

Testing a single image

Predicting image quality with our model trained on the Koniq-10k Dataset.

To run the demo, please download the pre-trained model at Google drive or Baidu cloud (password: 1ty8), put it in 'pretrained' folder, then run:

```
python demo.py
```

You will get a quality score ranging from 0-100, and a higher value indicates better image quality.

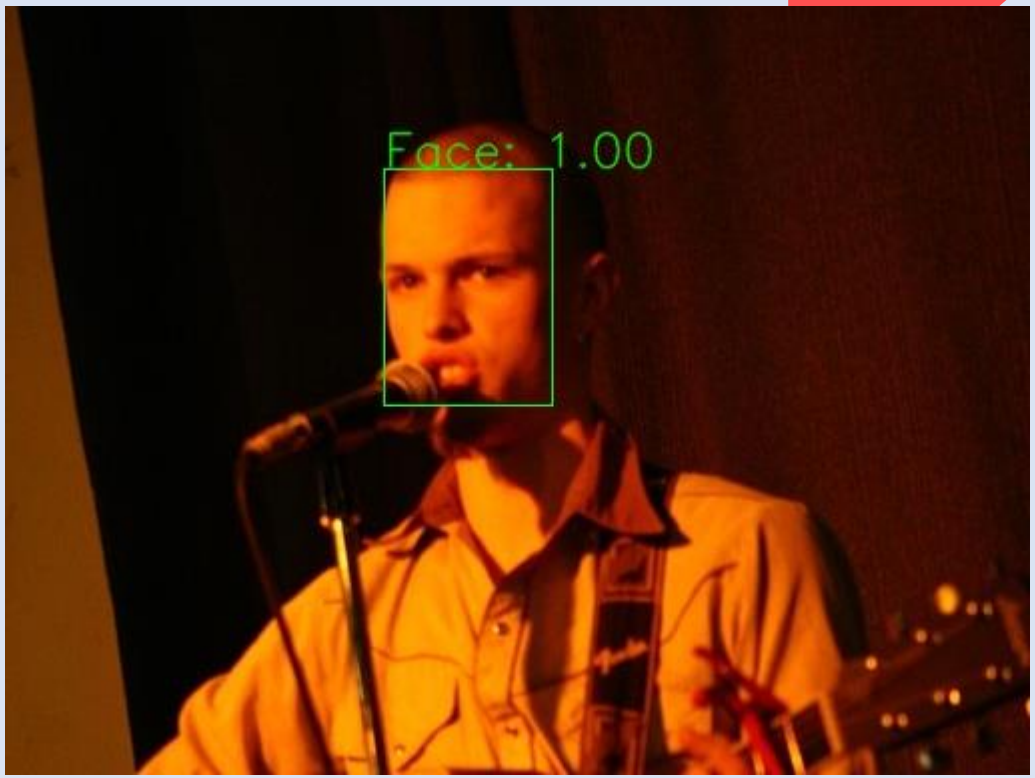
2. Dataset EDA 및 Preprocessing

- Face-detector 모델을 사용하여 사람이 있는 이미지만 추출
  - OpenCV 예제에서 DNN 모듈을 사용한 얼굴 검출 기능을 지원
  - SSD(Single Shot MultiBox Detector) 기반 얼굴 검출 네트워크
  - https://github.com/opencv/opencv/tree/master/samples/dnn/face_detector
  - 기존의 Haar-Cascade 방법보다 속도 & 정확도 면에서 더 좋은 성능을 나타냄

	Haar Cascade	DL
Size on disk	528KB	10MB (fp32), 5MB (fp16)
Efficiency @ 300x300**	30 ms	9.34 ms
Performance AP @ IoU = 0.5*	0.609 (FDDB) 0.149 (WIDER FACE, val.)	0.797 (FDDB) 0.173 (WIDER FACE, val.)

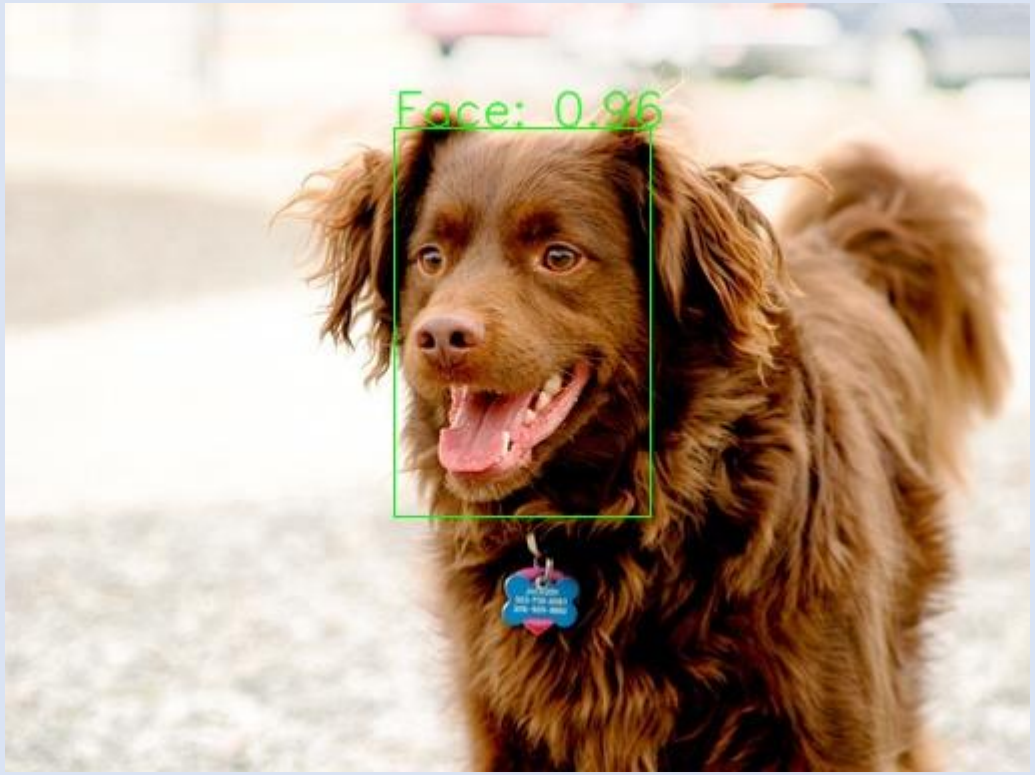
2. Dataset EDA 및 Preprocessing

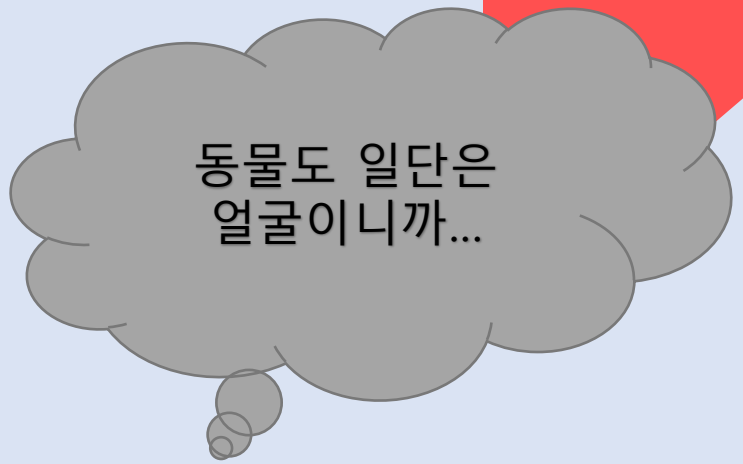
- Face-detector 모델을 사용하여 사람이 있는 이미지만 추출
  - 10,373장 ⟶ 1,563장
  - Confidence 값을 0.5 이상으로 설정



2. Dataset EDA 및 Preprocessing

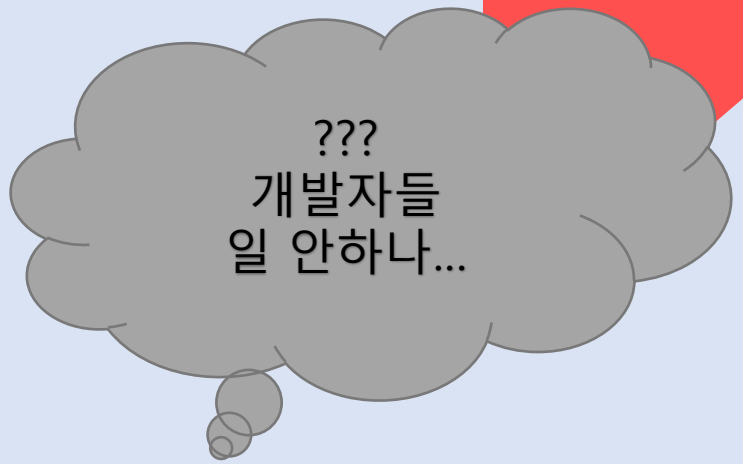
- Face-detector 모델을 사용하여 사람이 있는 이미지만 추출
  - 10,373장 ⟶ 1,563장
  - Confidence 값을 0.5 이상으로 설정





2. Dataset EDA 및 Preprocessing

- Face-detector 모델을 사용하여 사람이 있는 이미지만 추출
  - 10,373장 ⟶ 1,563장
  - Confidence 값을 0.5 이상으로 설정



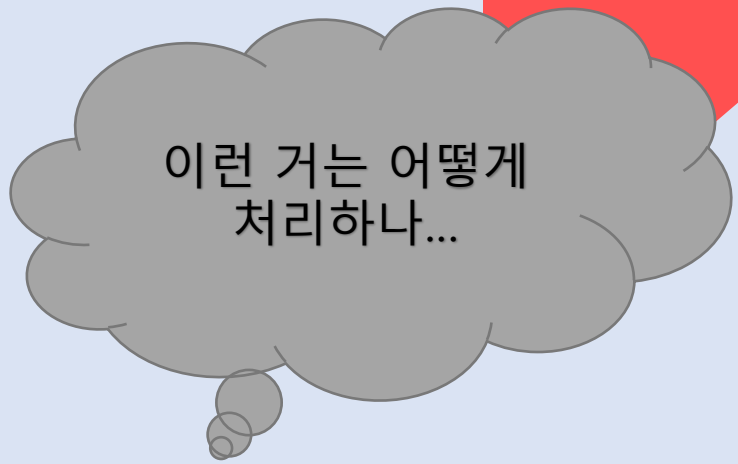
2. Dataset EDA 및 Preprocessing

- Face-detector 모델을 사용하여 사람이 있는 이미지만 추출
  - 10,373장 ⟶ 1,563장
  - Confidence 값을 0.5 이상으로 설정

0.5보다 높이면
안 잡히는 얼굴이
꽤 있는데...

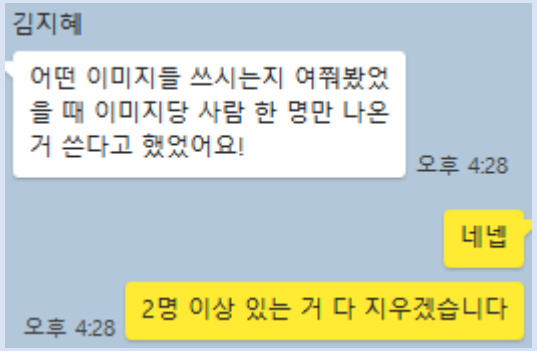
2. Dataset EDA 및 Preprocessing

- 모델을 사용하여 추출한 1,563장 중 사람이 아닌 이미지를 직접 제거
  - 1,563장 ⟶ 1,048장
  - 이미지에 다수의 사람이 있는 경우가 있음.



2. Dataset EDA 및 Preprocessing

- 모델을 사용하여 추출한 1,563장 중 사람이 아닌 이미지를 직접 제거
  - 1,563장 ⟶ 1,048장
  - 기업 화상 회의 당시 이미지 당 사람이 한명이라는 정보를 얻음

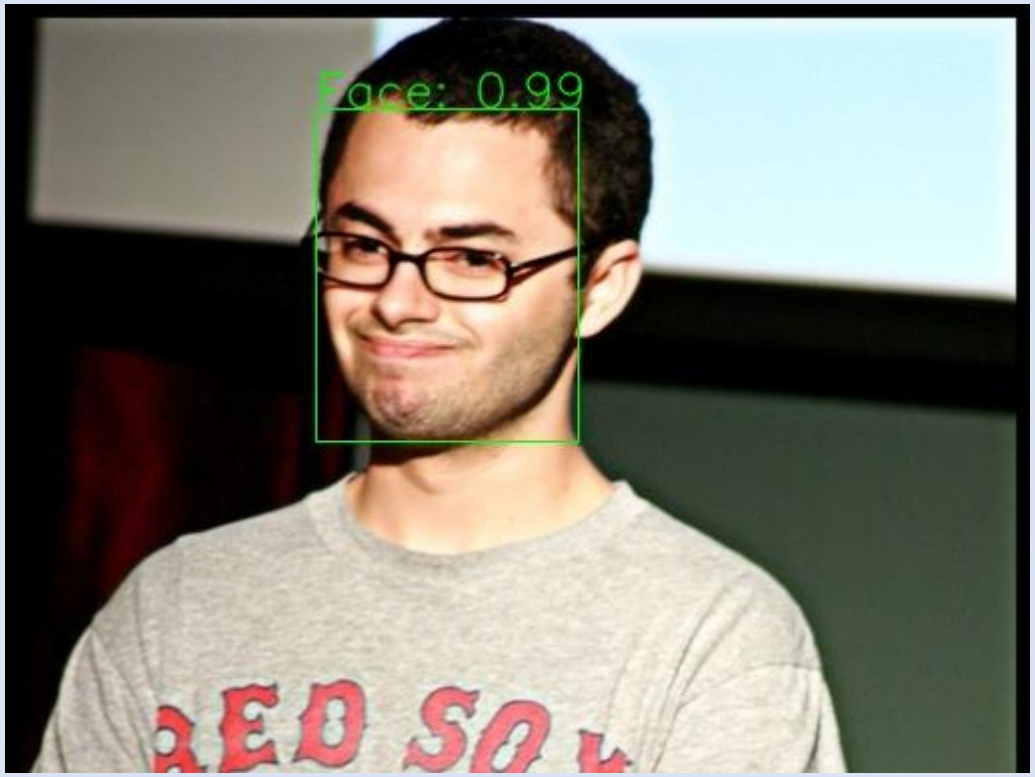


2. Dataset EDA 및 Preprocessing

- 1,048장의 이미지 중 사람이 2명 이상인 이미지를 직접 제거
  - 1,048장 ⟶ 509장
  - ~~(사실 이건 모델 돌렸어도 되는데...)~~

2. Dataset EDA 및 Preprocessing

- 1,048장의 이미지 중 사람이 2명 이상인 이미지를 제거
  - 1,048장 ⟶ 509장
  - 최종적으로 사람이 한 명 있고 confidence값이 0.5 이상인 이미지만 추출



2. Dataset EDA 및 Preprocessing

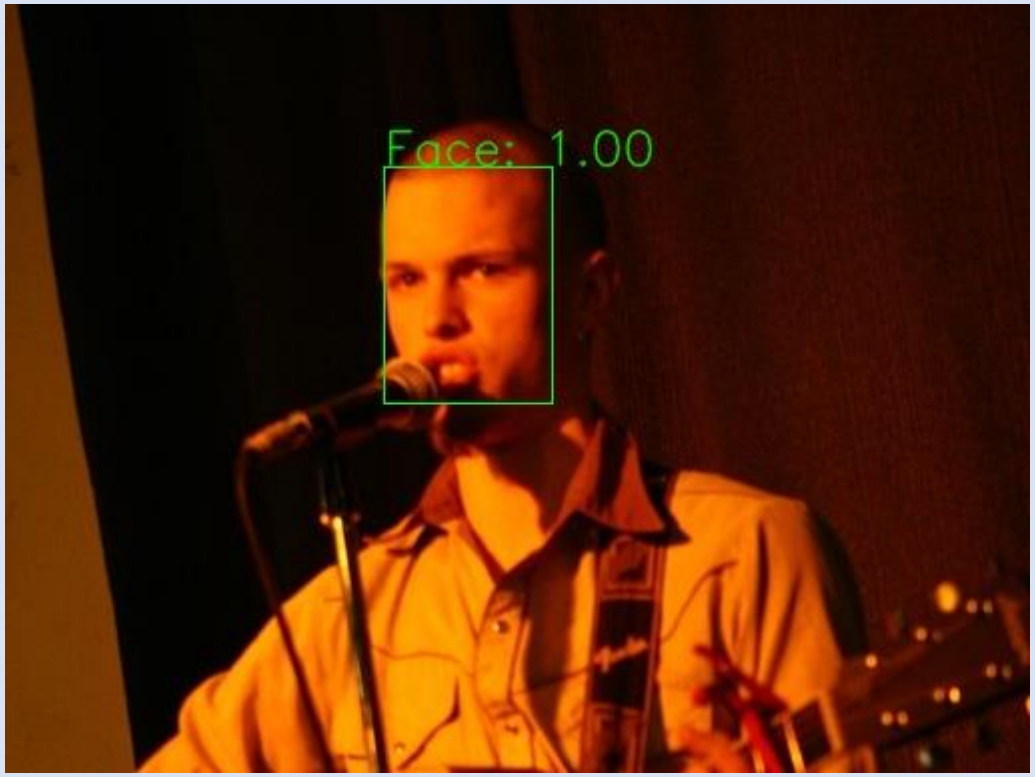
- 509장의 이미지와 CSV 파일 매칭
  - 이미지는 있지만 CSV 데이터가 없는 경우가 존재
  - 데이터가 없는 이미지 20장 삭제
  - 509장 ⟶ 489장

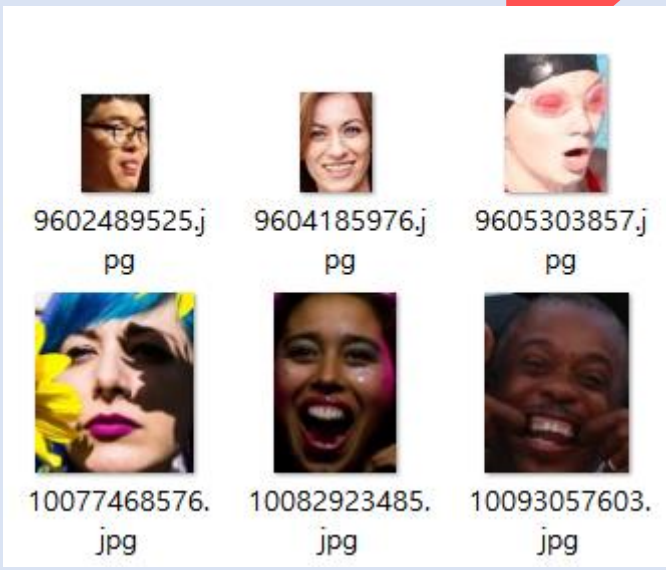
	image_name	c1	c2	c3	c4	c5	c_total	MOS	SD	MOS_zscore
3	10009096245.jpg	0	0	21	75	13	109	3.926606	0.556218	77.243750
13	10027545463.jpg	0	0	31	66	5	102	3.745098	0.539275	72.827982
14	10027582235.jpg	0	0	24	78	5	107	3.822430	0.491687	77.137712
33	10077468576.jpg	0	0	38	66	2	106	3.660377	0.514306	73.815217
36	10082923485.jpg	0	16	77	11	0	104	2.951923	0.509708	53.226087
...	...	...	...	...	...	...	...	...	...	...
9994	9808126155.jpg	8	71	21	0	0	100	2.130000	0.525222	29.080275
9998	9821930934.jpg	0	1	43	57	3	104	3.596154	0.566358	68.337838
10016	9859562873.jpg	3	61	37	0	0	101	2.336634	0.534364	36.283482
10044	9932883805.jpg	0	0	45	61	1	107	3.588785	0.513098	70.466387
10049	9935622.jpg	2	56	43	5	0	106	2.481132	0.620770	39.872881

489 rows × 10 columns

2. Dataset EDA 및 Preprocessing

- 얼굴 이미지 잘라내기
  - 총 489장의 face_cropped_image를 얻음





2. Dataset EDA 및 Preprocessing

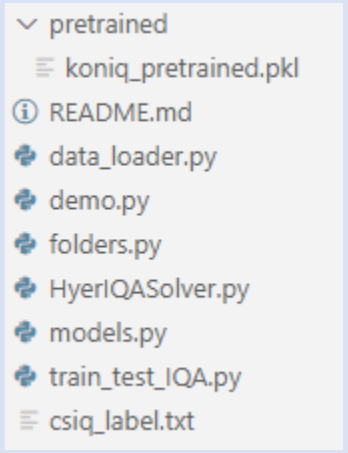
- 불필요한 CSV 컬럼 삭제 및 컬럼 추가
  - Image_predic_score : 원본 이미지로 측정한 predicted quality
  - Face_predic_score : 얼굴만 자른 이미지로 측정한 predicted quality

	image_name	MOS_zscore	image_predic_score	face_predic_score
0	10009096245.jpg	77.243750	0	0
1	10027545463.jpg	72.827982	0	0
2	10027582235.jpg	77.137712	0	0
3	10077468576.jpg	73.815217	0	0
4	10082923485.jpg	53.226087	0	0
...	...	...	...	...
484	9808126155.jpg	29.080275	0	0
485	9821930934.jpg	68.337838	0	0
486	9859562873.jpg	36.283482	0	0
487	9932883805.jpg	70.466387	0	0
488	9935622.jpg	39.872881	0	0

489 rows × 8 columns

3. Implementation

- 제공된 demo.py와 pre-trained model을 사용



Usages

Testing a single image

Predicting image quality with our model trained on the Koniq-10k Dataset.

To run the demo, please download the pre-trained model at Google drive or Baidu cloud (password: 1ty8), put it in 'pretrained' folder, then run:

```
python demo.py
```

You will get a quality score ranging from 0-100, and a higher value indicates better image quality.

3. Implementation

- 제공된 demo.py와 pre-trained model을 사용

```
etach() or sourceTensor.clone().detach().requires_grad_(True), rather than torch.tensor(sourceTensor).
  img = torch.tensor(img.cpu()).unsqueeze(0)
Predicted quality score: 76.86
```

- 제공된 demo.py와 pre-trained model이 정상 작동하는 것을 확인
- 처음에는 GPU가 없다고 하여 로컬에서 작동이 되지 않음
- 구글 colab으로 이식하여 모델을 작동
- ~~근데 오늘 다시 해보니 CPU로 설정되어 잘 돌아감.~~

3. Implementation

- 제공된 demo.py와 pre-trained model을 사용
- 원본 이미지 489장을 test data로 Image_predic_score를 추출

	image_name	MOS_zscore	image_predic_score	face_predic_score
0	10009096245.jpg	77.243750	68.595401	0
1	10027545463.jpg	72.827982	68.649963	0
2	10027582235.jpg	77.137712	68.441750	0
3	10077468576.jpg	73.815217	69.222231	0
4	10082923485.jpg	53.226087	47.766008	0
...	...	...	...	...
484	9808126155.jpg	29.080275	27.906944	0
485	9821930934.jpg	68.337838	63.600168	0
486	9859562873.jpg	36.283482	35.754680	0
487	9932883805.jpg	70.466387	63.188913	0
488	9935622.jpg	39.872881	37.430399	0

489 rows × 8 columns

3. Implementation

- 제공된 demo.py와 pre-trained model을 사용
- 얼굴을 자른 이미지 489장을 test data로 Face_predic_score를 추출

	image_name	MOS_zscore	image_predic_score	face_predic_score
0	10009096245.jpg	77.243750	68.595401	32.174589
1	10027545463.jpg	72.827982	68.649963	31.567229
2	10027582235.jpg	77.137712	68.441750	47.628218
3	10077468576.jpg	73.815217	69.222231	39.306854
4	10082923485.jpg	53.226087	47.766008	33.829192
...	...	...	...	...
484	9808126155.jpg	29.080275	27.906944	22.466578
485	9821930934.jpg	68.337838	63.600168	35.479411
486	9859562873.jpg	36.283482	35.754680	34.887408
487	9932883805.jpg	70.466387	63.188913	37.021549
488	9935622.jpg	39.872881	37.430399	30.018513

489 rows × 8 columns

4. Result - HyperIQA

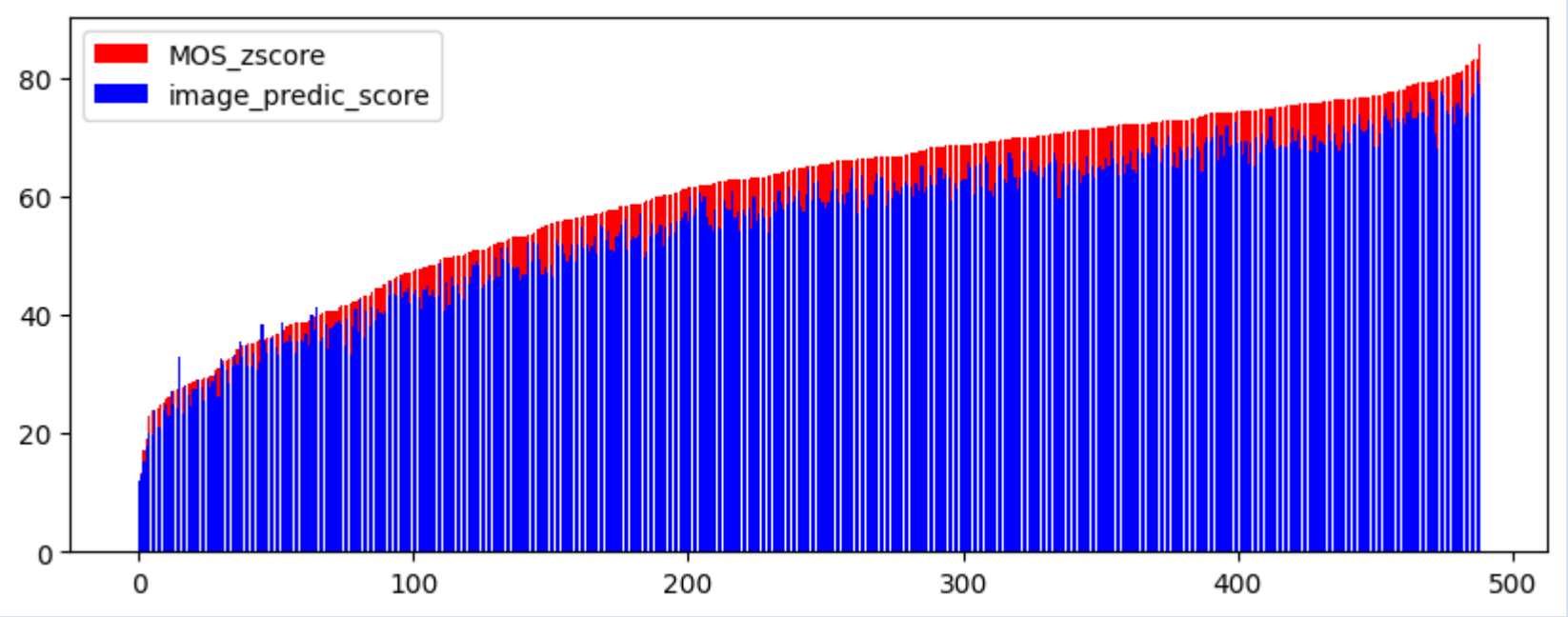
- 원본 이미지 score에 비해 얼굴을 자른 이미지 score가 매우 낮다.
- 원래 score가 낮은 경우 하락폭이 크지 않지만, score가 높은 이미지의 경우 50% 가까이 점수가 낮게 나온다.

	image_name	MOS_zscore	image_predic_score	face_predic_score
0	10009096245.jpg	77.243750	68.595401	32.174589
1	10027545463.jpg	72.827982	68.649963	31.567229
2	10027582235.jpg	77.137712	68.441750	47.628218
3	10077468576.jpg	73.815217	69.222231	39.306854
4	10082923485.jpg	53.226087	47.766008	33.829192
...	...	...	...	...
484	9808126155.jpg	29.080275	27.906944	22.466578
485	9821930934.jpg	68.337838	63.600168	35.479411
486	9859562873.jpg	36.283482	35.754680	34.887408
487	9932883805.jpg	70.466387	63.188913	37.021549
488	9935622.jpg	39.872881	37.430399	30.018513

489 rows × 8 columns

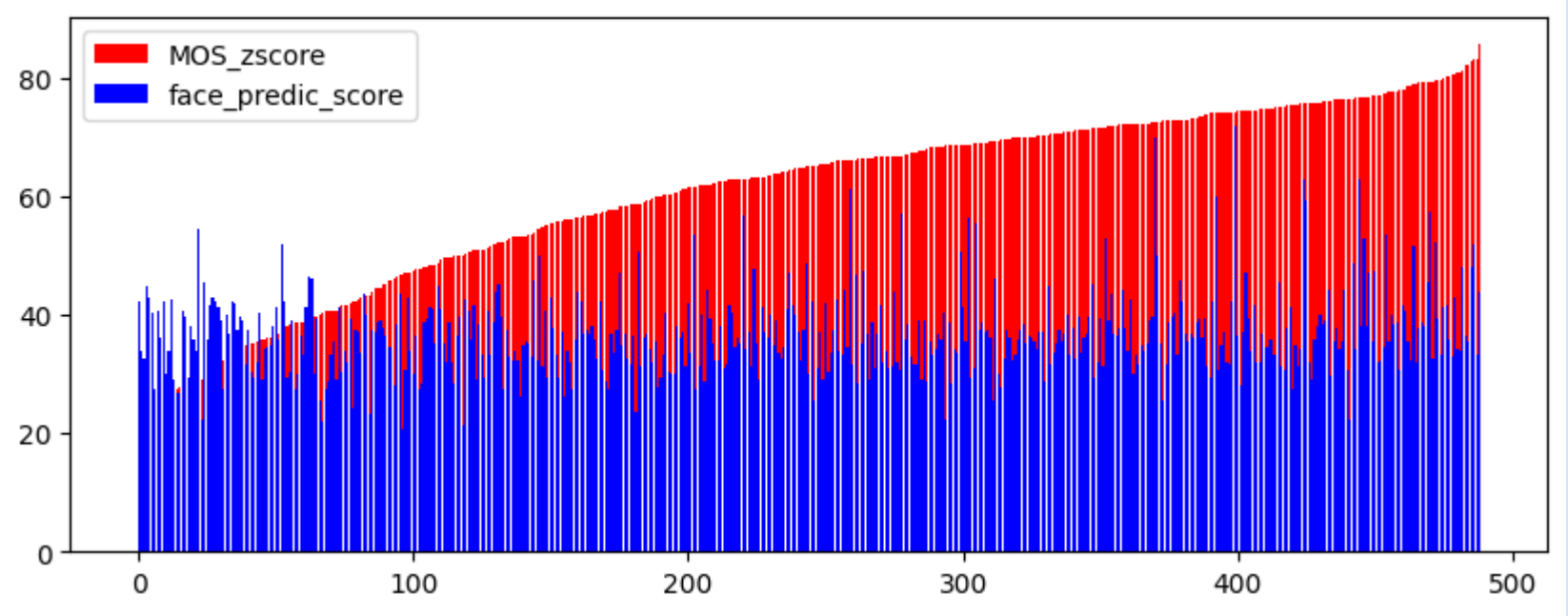
4. Result - HyperIQA

- 원본 이미지 MOS_zscore와 원본 이미지 predict_score 비교
  - 매우 유사한 경향을 보인다.
  - predict_score가 MOS_zscore에 비해 조금 낮지만 큰 차이는 없다.



4. Result - HyperIQA

- 원본 이미지 MOS_zscore와 얼굴 자른 이미지 face_predict_score 비교
  - 매우 큰 차이를 보인다.
  - face_predict_score가 대부분 20 ~ 40점 사이 값을 갖는다.
  - MOS_zscore가 높을 수록 차이가 심해진다.



4. Result - HyperIQA

- PLCC, SRCC, MSE 비교

	Model_name	PLCC	SRCC	face_PLCC	face_SRCC	MSE	face_MSE
0	HyperIQA	0.990063	0.985667	0.123718	0.104186	28.658855	818.429342

  - 원본 이미지의 MOS_zscore와 모델이 예측한 score로 PLCC, SRCC 값을 계산했을 때, 0.99, 0.98 이라는 매우 높은 값을 보인다.
    - 이것은 이 모델이 사람이 보는 것과 매우 유사하게 이미지의 quality를 측정한다는 것을 뜻한다.
  - 하지만, 얼굴만 자른 이미지로 예측한 score로 PLCC, SRCC 값을 계산했을 때, 0.12, 0.10 이라는 매우 낮은 값을 보인다.
    - 이것은 이 모델이 사람이 보는 것과 거의 연관성이 없게 이미지의 quality를 측정한다는 것을 뜻한다.

4. Result - HyperIQA

- PLCC, SRCC, MSE 비교

	Model_name	PLCC	SRCC	face_PLCC	face_SRCC	MSE	face_MSE
0	HyperIQA	0.990063	0.985667	0.123718	0.104186	28.658855	818.429342

  - 원본 이미지의 MOS_zscore와 모델이 예측한 score로 MSE 값을 계산했을 때 28.65라는 값을 보인다.

  - 하지만, 얼굴만 자른 이미지로 예측한 score로 MSE 값을 계산했을 때 818.42라는 매우 높은 값을 보인다.

    - MSE 비교를 통해 얼굴 이미지만 예측한 score는 error가 매우 심한 것을 알 수 있고, 이 score는 신뢰할 수 없다는 결론을 낼 수 있다.

5. Problem and Solution

- **Summary**
  - **이번에 구현한 HyperIQA 모델은 KonIq-10k로 pre-trained 되어 있고,**
  - **사람이 들어간 이미지의 quality를 평가하는데 훌륭한 결과를 보인다.**
  - **하지만, 우리가 원하는 얼굴 이미지의 quality를 평가하기에는 매우 안좋은 결과를 보이고 있다.**

5. Problem and Solution

- MOS_zscore가 정확하지 않다.
  - 모델 평가에 활용한 MOS_zscore값은 원본 이미지에 대한 값으로, 우리가 평가에 활용하기에 적합하지 않다.
  - 얼굴만 자른 이미지의 경우 원본 이미지의 특정 부분에 속하기 때문에 이 부분만을 특정한 MOS_zscore값이 필요하다.
  - 하지만 MOS_zscore값은 labelling이 매우 어렵기 때문에 현실적으로 불가능하다.
- Pre-trained 모델을 사용했다.
  - 이번 predict score를 계산하는데 사용한 모델은 KonIq-10k로 pre-trained 된 모델이고, 이 모델은 사람의 얼굴을 평가하는데 적합하지 않다.
  - 처음부터 학습을 다시 하거나, 얼굴 이미지로 학습한 weights 값이 필요하다.

5. Problem and Solution

- 얼굴 이미지의 크기가 너무 작다.
  - 모델이 input 이미지를 resize(512 * 384)하는데 당연히 화질이 안 좋아진다.

- 개인적인 문제
  - 구현이 어렵고, 모델을 수정하는 것이 불가능할 것 같다.
  - Pytorch에 대한 학습이 부족하고,
    source code를 분석하고 이해하는 역량이 부족하다.
  - 이 모델의 경우, demo.py라는 실행 파일이 있어 어렵지 않았지만
    demo.py가 없다고 가정했을 때 시도한 구현은 실패했다.