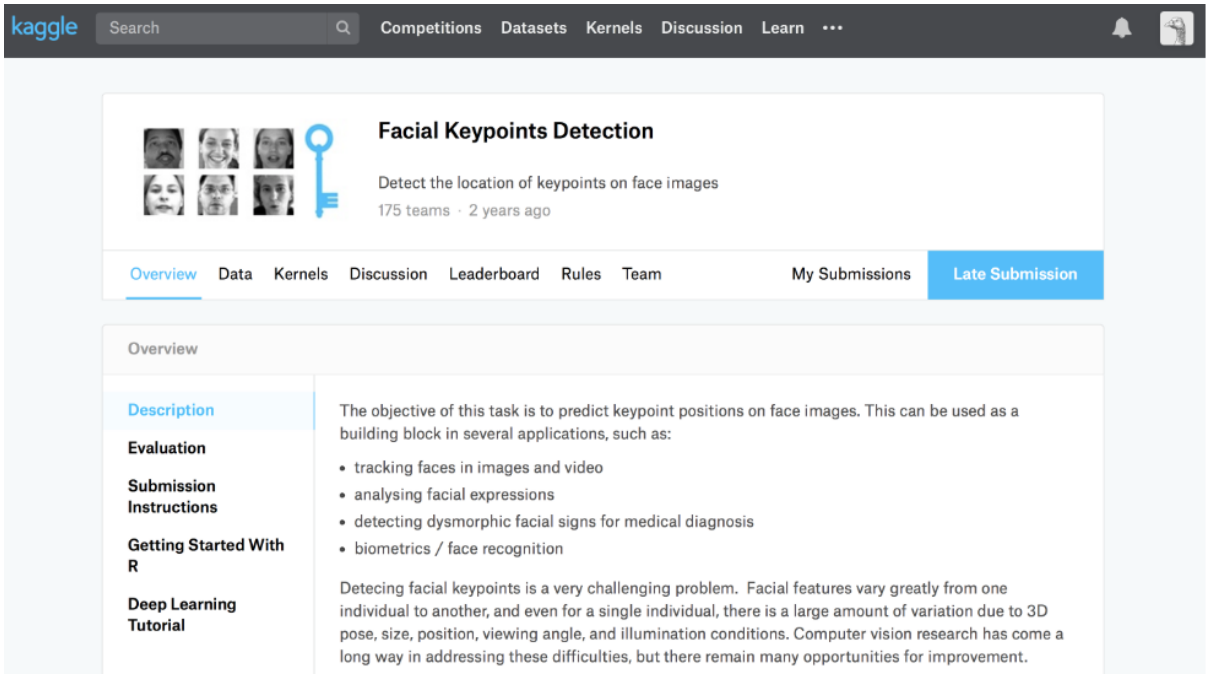
딥러닝 기반 핵심 산업별 빅데이터 분석 <머신러닝&딥러닝 파일럿 프로젝트>			
주 제	얼굴이미지의 키포인트 위치 감지	**링 크**	https://www.kaggle.com/c/facial-keypoints-detection
팀 명	JCJ(제이씨제이)	**일 자**	2018년 11월 23 - 29일
팀 장	장은경<sogang@gmail.com>	**팀 원**	정민호, 주상훈

1. 과제 개요

가. 주제: Facial Keypoints Detection

- 얼굴 이미지의 주요 포인트 위치 예측하기 ex. 눈, 코, 입 위치 등



나. 과제 선정 이유

1) 다양한 도메인에 광범위하게 응용 가능

- 이미지 혹은 영상 속 안면 인식
- 의학 진단에 활용
- 생체 인식을 활용한 보안 시스템에 적용

2) 딥러닝 수업 응용 기회

- 딥러닝, CNN 등 학습 내용을 복습하는 차원에서 적절

3) 이미지 인식에 대한 관심

- MNIST 데이터 분석을 바탕으로 이미지 분석 심화 학습

2. 데이터 설명

가. 학습데이터(training.csv)

- 7049 개의 이미지 목록으로 이뤄진 학습 데이터
- 각 행은 15 개의 키포인트에 대한 (x, y) 좌표와 행 순서의 픽셀 목록으로 된 이미지 데이터

나. 검증 데이터(test.csv)

- 1783 개의 테스트 이미지 목록으로 이뤄진 테스트 데이터
- 각 행에는 ImageId 와 이미지 데이터가 행 순서의 픽셀 목록으로 포함되어 있음

다. 제출파일 형식(submissionFileFormat.csv)

- 예측할 27124 키포인트 목록
- 컬럼은 RowId, ImageId, FeatureName, Location(예측할 항목)

예측할 항목: 얼굴에서 15 개 키포인트

left_eye_center, right_eye_center, left_eye_inner_corner, left_eye_outer_corner, right_eye_inner_corner, right_eye_outer_corner, left_eyebrow_inner_end, left_eyebrow_outer_end, right_eyebrow_inner_end, right_eyebrow_outer_end, nose_tip, mouth_left_corner, mouth_right_corner, mouth_center_top_lip, mouth_center_bottom_lip

3. 과제 수행 내역

1. 데이터 탐색
   A. 7049개의 데이터 중 2140개만 온전한 데이터
   B. Image 컬럼의 데이터의 타입이 문자열로 되어있음
2. 데이터 전처리
   A. 4909개의 결측값 제거
   B. 이미지 픽셀값 데이터(str) → int 변환
      i. ['123 0 0 2 ... 3 15 11'] → [123, 0, 0, 2, ... , 3, 15, 11]
3. 픽셀 이미지 시각화
4. 이미지 분석(케라스 & 텐서플로우)
   A. CNN
      i. Convolution layer
         1. 입력된 이미지에서 다시 한번 특징을 추출하기 위해 마스크를 도입
         2. 마스크 크기, 개수, 이동 거리 정의
      ii. Max_pooling

1. 컨볼루션 레이어를 통해 생성된 이미지를 축소
2. Pooling mask 크기, 개수, 이동 거리 정의

iii. Flattening
1. 앞서 컨볼루션한 다차원 데이터를 이후 학습을 위해 평탄화(2차원으로)
2. [24*24 * 128, 30]

iv. Fully Connected Layer
1. 이전 계층의 모든 레이어를 결합된 형태의 레이어를 만든다

B. 모델 학습
i. Cost 값 계산
ii. Optimization을 통해 cost값을 최소화

5. 새로운 조건의 모델을 만들고 검증 → 반복

4. 결과 보고

1. 분석 결과

모델	케라스 활용 여부	Learning rate	Batch size	컨볼루션 Layer 개수	컨볼루션 Layer 사이즈	Pooling mask 사이즈	Training epochs	최종 COST 값	Kaggle 점수
1	X	0.001	20	32	(3, 3)	(2, 2)	263	4.86	10.6
				64	(3, 3)	(2, 2)			
2	X	0.001	100	64	(3, 3)	(2, 2)	500	10.14	11.81626
				128	(3, 3)	(2, 2)			
3	O	0.001	100	64	(4, 4)	(2, 2)	100	18.26	**4.58835**
4	O	0.001	100	64	(5, 5)	(2, 2)	100(68)	17.24	5.20052
				32	(4, 4)	(2, 2)			

2. 배운 점

A. 오차율이 아주 아주 큰 경우, learning rate를 조정하거나 cost를 산정하는 방식을 변경해보자!
i. 예) Softmax_cross_entropy_with_logits → **MSE**

B. Convolution 수나 Epoch 수를 마냥 늘린다고 cost가 줄어드는 것은 아니다

C. 모델이 아주 큰 경우 학습 결과를 꼭 저장하자(saver)

D. 이미지 데이터 학습의 경우 학습된 사이즈보다 더 큰 값으로 예측하기도 한다
i. 좌표값의 최댓값이 96인데 **96 이상으로** 예측하기도 함

E. Batch size, Convolution Layer, Pooling mask 개수 및 사이즈 혹은 learning rate 등을

지속적으로 변경해보는 것이 COST값의 개선에 유의미할 수 있다.

F. 이미지 파일이 색상값으로 이뤄진 경우, normalizing을 해줄지 말지 여부를 고려하자
G. 과적합 검증의 필요성
   i. Train 데이터에서는 잘 적용됐던 모델이 test 데이터에서 적용이 잘 안 되는 현상
   ii. Train set을 학습데이터와 검증데이터로 나누어서 학습시킨 결과가 더 좋았음
H. 무에서 유를 창조하기보다 유에서 더 나은 결과를 도출하는 것도 하나의 방법이다
   i. 분석 전에 목표를 달성하기 위한 적합한 방법을 찾아 벤치마킹
   ii. 현재 나와 있는 가장 효율적인 툴을 찾아 적용

.

감사합니다