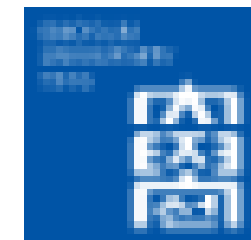


Project Progress Report

오픈소스SW 프로젝트 제안 발표

3조
고승우 박기정 서준
형



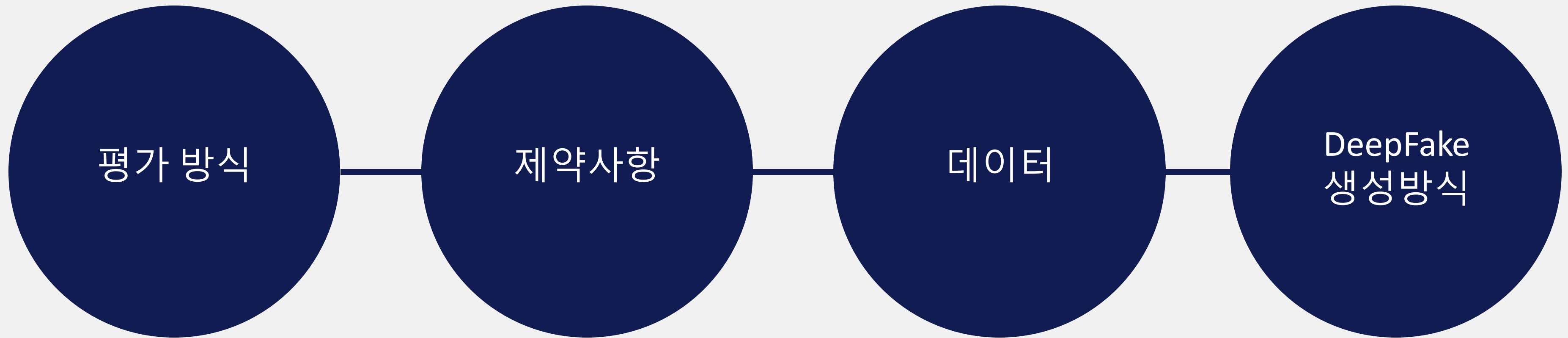
조선대학교
CHOSUN UNIVERSITY



Contents

- 01** 과제 개요
 - 평가 방식, 제약사항, 데이터, DeepFake 생성방식
- 02** EDA
 - 데이터 분포 확인, Real data, Fake data
- 03** 데이터 전처리 계획
 - Face labeling, 프레임 처리, 정규화, 불균형 해소방안
- 04** 분류 방식, 모델 선정
 - Face forensic, 주요 모델 선정, 앙상블

1.과제 개요



평가방식

Evaluation

Submissions are scored on log loss:

$$\text{LogLoss} = -\frac{1}{n} \sum_{i=1}^n [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)],$$

where

- n is the number of videos being predicted
- \hat{y}_i is the predicted probability of the video being FAKE
- y_i is 1 if the video is FAKE, 0 if REAL
- $\log()$ is the natural (base e) logarithm

A smaller log loss is better. The use of the logarithm provides extreme punishments for being both confident and wrong. In the worst possible case, a prediction that something is true when it is actually false will add infinite to your error score. In order to prevent this, predictions are bounded away from the extremes by a small value.

Submission File

For each `filename` in the test set, you must predict a probability for the `label` variable. The file should contain a header and have the following format:

```
filename,label
10000.mp4,0
10001.mp4,0.5
10002.mp4,1
etc.
```

평가 방법


- n : 예측하는 비디오의 수 (전체 샘플 수)
- \hat{y}_i : i 번째 비디오가 FAKE일 확률에 대한 예측값 (모델의 예측 확률, 0과 1 사이의 값)
- y_i : i 번째 비디오가 실제로 FAKE인지 여부 (정답값, FAKE이면 1, REAL이면 0)
- $\log()$: 자연 로그, 밑(base)이 e 인 로그
- Log Loss: 모델이 얼마나 확률적으로 정확하게 예측하는지, 특히 확률 예측에서의 신뢰도를 기반으로 평가하는 방법

제출 방법

- Format에 맞는 제출 파일을 생성하여 Kaggle에 업로드
- Comma separated variables (CSV) file 제출

제약사항

Code Requirements



This is a Code Competition

Submissions to this competition must be made through Notebooks. In order for the "Submit to Competition" button to be active after a commit, the following conditions must be met:

- CPU Notebook <= 9 hours run-time
- GPU Notebook <= 9 hours run-time on Kaggle's [P100 GPUs](#)
- No internet access enabled
- External data is allowed up to 1 GB in size. External data must be freely & publicly available, including pre-trained models.
- No using other Kaggle notebooks or utility scripts as inputs to your submission notebook. External data also cannot be externally-referenced (i.e. via BigQuery, Github, external URL). Instead, load external models or datasets directly as an external data source.
- No custom packages enabled in your submission notebook
- Submission file must be named "submission.csv"

Please see the [Code Competition FAQ](#) and [Getting Started](#) for specifics on this competition's unique design.

제약조건

CPU/GPU notebook에서 9 시간 이내 동작

-> Kaggle notebook 환경 내에서 각 모델 별 소요 시간 파악,
가능한 Epoch 및 모델 크기 분석

외부 데이터 1GB까지 허용, pre-trained model 허용

-> 외부 데이터 탐색 : 이번 수업에서는 미해당
(데이터 분포에 따른 불균형 라벨 해소방안)

데이터

데이터 상세

- Training dataset: 470GB 크기의 데이터
(본 수업에서는 0~7까지만 사용하므로 약 80GB)
-> 데이터 라벨 분포 확인, 영상 길이별 분포 확인
- Validation dataset
-> Train dataset을 일정 비율로 분할(8:2 or 7:3)
- Test dataset
-> test dataset에도 영상이 동일하게 분포하는 지 확인이 필요함.
- Fake : generative adversarial network (GAN), variational auto-encoder (VAE), flow model 등을 활용하여 생성된 동영상

Datasets:

There are 4 groups of datasets associated with this competition.

1. **Training Set:** This dataset, containing labels for the target, is available for download outside of Kaggle for competitors to build their models. It is broken up into 50 files, for ease of access and download. Due to its large size, it must be accessed through a GCS bucket which is only made available to participants after accepting the competition's rules. Please read the rules fully before accessing the dataset, as they contain important details about the dataset's permitted use. It is expected and encouraged that you train your models outside of Kaggle's notebooks environment and submit to Kaggle by uploading the trained model as an external data source.
2. **Public Validation Set:** When you commit your Kaggle notebook, the submission file output that is generated will be based on the small set of 400 videos/ids contained within this Public Validation Set. This is available on the Kaggle Data page as `test_videos.zip`
3. **Public Test Set:** This dataset is completely withheld and is what Kaggle's platform computes the public leaderboard against. When you "Submit to Competition" from the "Output" file of a committed notebook that contains the competition's dataset, your code will be re-run in the background against this Public Test Set. When the re-run is complete, the score will be posted to the public leaderboard. If the re-run fails, you will see an error reflected in your "My Submissions" page. Unfortunately, we are unable to surface any details about your error, so as to prevent error-probing. You are limited to 2 submissions per day, including submissions which error.
4. **Private Test Set:** This dataset is privately held outside of Kaggle's platform, and is used to compute the private leaderboard. It contains videos with a similar format and nature as the Training and Public Validation/Test Sets, but are real, organic videos with and without deepfakes. After the competition deadline, Kaggle transfers your 2 final selected submissions' code to the host. They will re-run your code against this private dataset and return prediction submissions back to Kaggle for computing your final private leaderboard scores.

Deepfake 생성방식

Deepfake 생성방식

- Face Synthesis

-> 가상의 얼굴 생성하여 합성, ex: Style GAN으로 학습된 이미지를 이용한 가짜 이미지 생성

- Face Attributes

-> 기존의 얼굴에 특정 속성만 바꿈(안경, 머리숱, 피부색 등) ex: Star GAN

- Facial Expression

-> 최근 자주 이용되는 방식, Source 얼굴에 Target 얼굴의 특정 표정등을 적용하여 생성
ex: Face2face, Synthesizing obama

- Face swap

-> 가장 보편적으로 이용되는 방식, 으특정한 사람과 다른사람의 얼굴의 교체. 피해자 사진의
특성을 소스 비디오에 입힘 ex: faceswap, deepfacelab

Face Synthesis



Facial Attributes



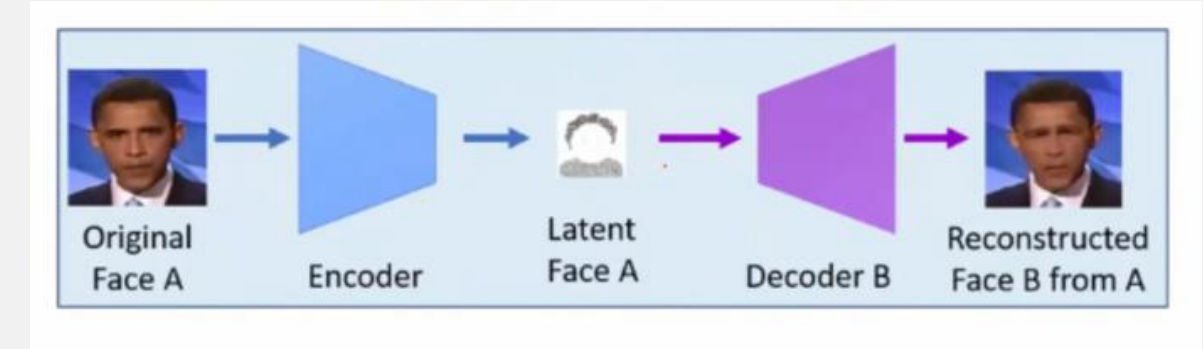
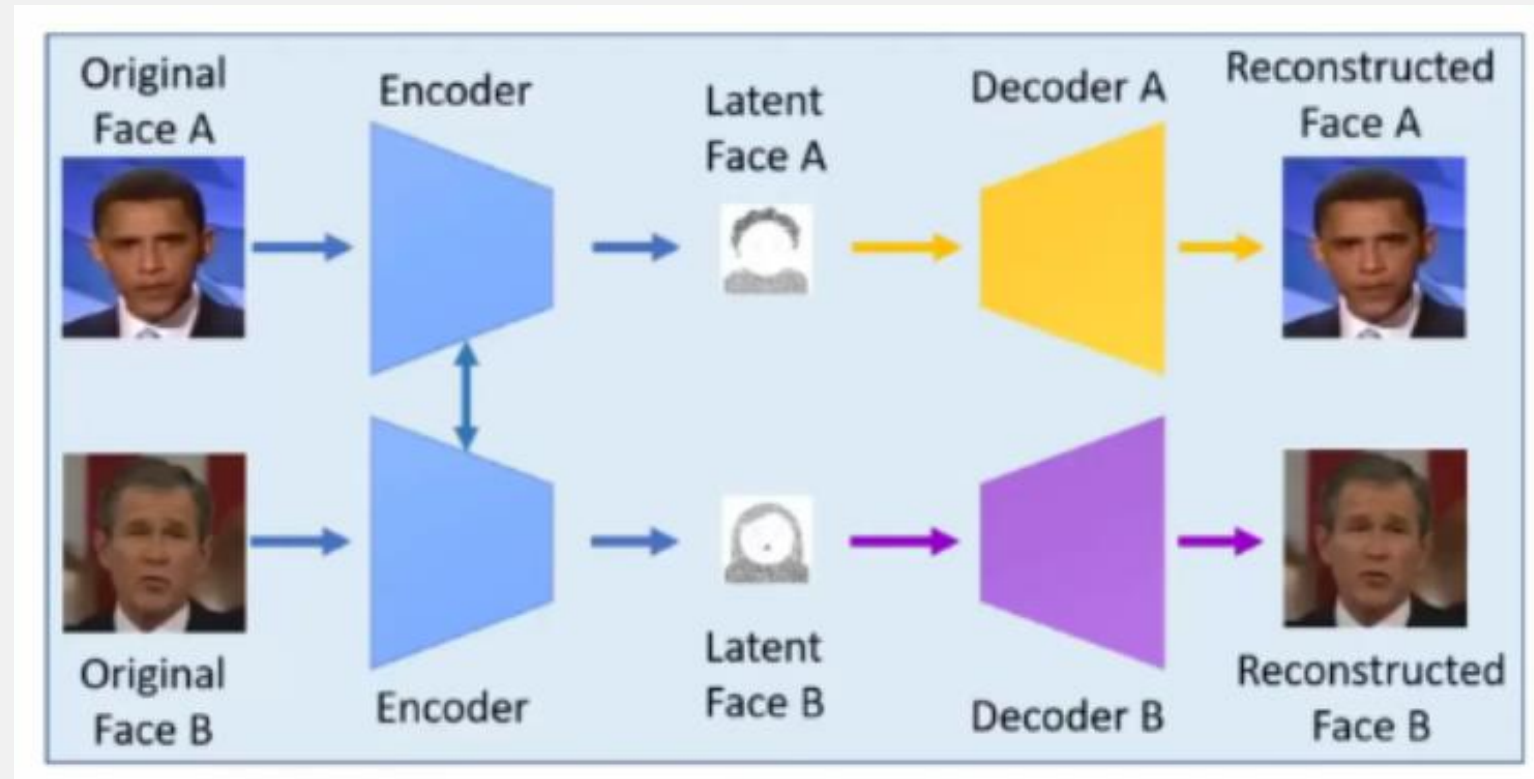
Facial Expression



Face Swap



Deepfake 생성방식



텐서플로우 기반으로 동작, 오토인코더를 기반으로 학습.
인코더 파트 공유, 디코더 파트는 서로 다르게 설정.
각 이미지의 특성 추출 → 공유함으로써 두 비디오의 공유되고 있는 특성이 학습됨.
여기서 공유되고 있는 특성: 눈코입위치, 유사한 특성이 학습되도록 유도
디코더에서는 피해자와 소스 특성이 각각 따로 학습되도록 유도 → 각 얼굴의 눈 생김새 코생김새 등
latent에서는 외곽이나 눈코입 위치만 학습

Deepfake 생성방식

- A sane number of images to use is anywhere between 1,000 and 10,000
- You want as many different angles, expressions and lighting conditions as possible
- The quality of training data should be of high quality (sharp and detailed)
- The original model can take anywhere from 12-48 hours to train on a Nvidia GTX 1080
- Original model: 64px input, 64px output

- Deepfake를 생성할때, 다양한 각도, 표정, 조명 조건의 사진이 있어야 좋은 결과물이 나온다는 가이드라인 확인.
- 반대로 각도, 표정, 조명, 프레임 유사도 조건을 고려한다면 Deepfake 탐지에 도움이 될것이라고 추정.

2.EDA



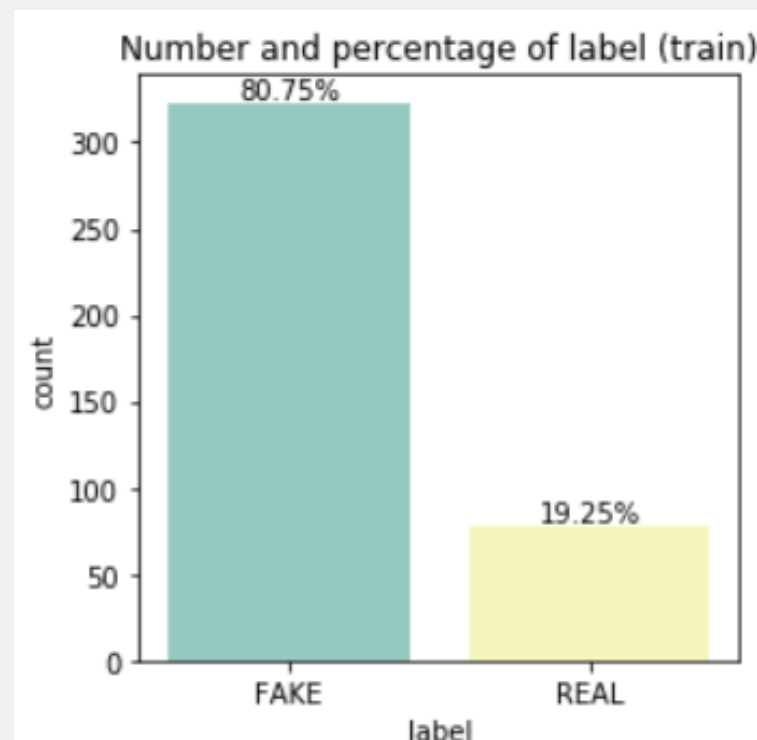
2.EDA

파일 유형 확인

```
train_list = list(os.listdir(os.path.join(DATA_FOLDER, TRAIN_SAMPLE_FOLDER)))
ext_dict = {}
for file in train_list:
    file_ext = file.split('.')[1]
    if (file_ext not in ext_dict):
        ext_dict.append(file_ext)
print(f"Extensions: {ext_dict}")
```

Extensions: ['mp4', 'json']

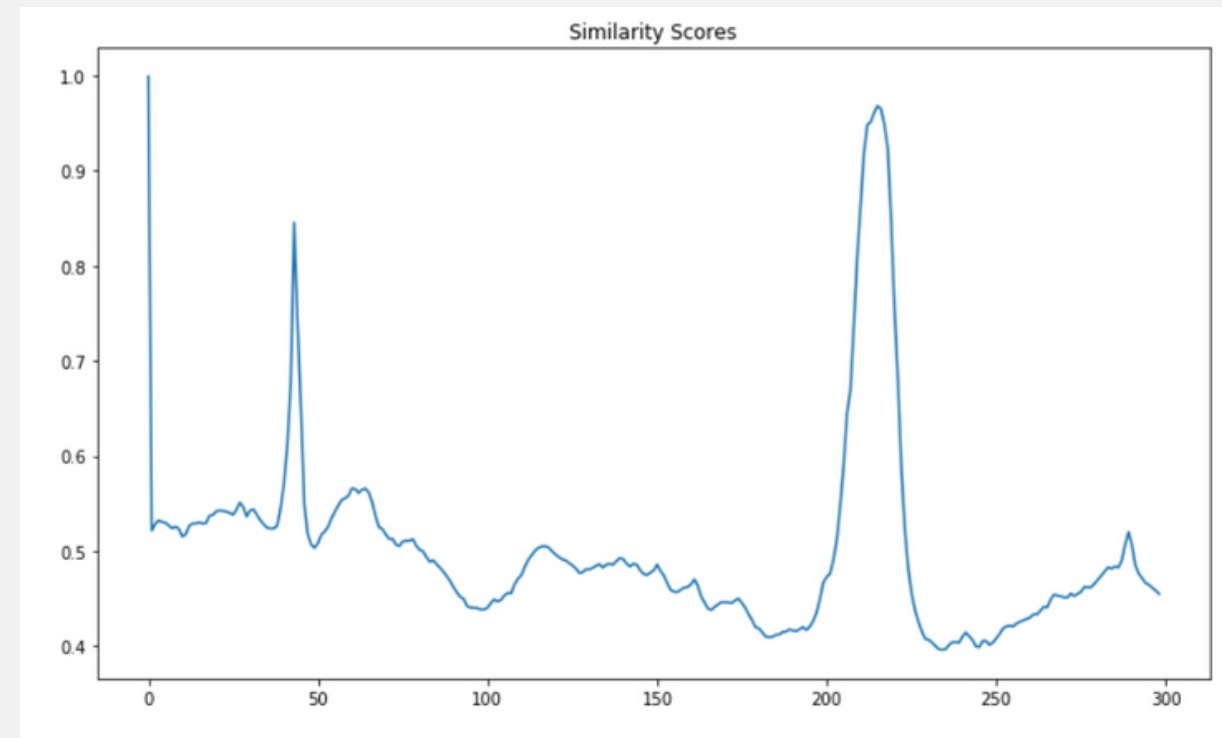
데이터 분포 확인



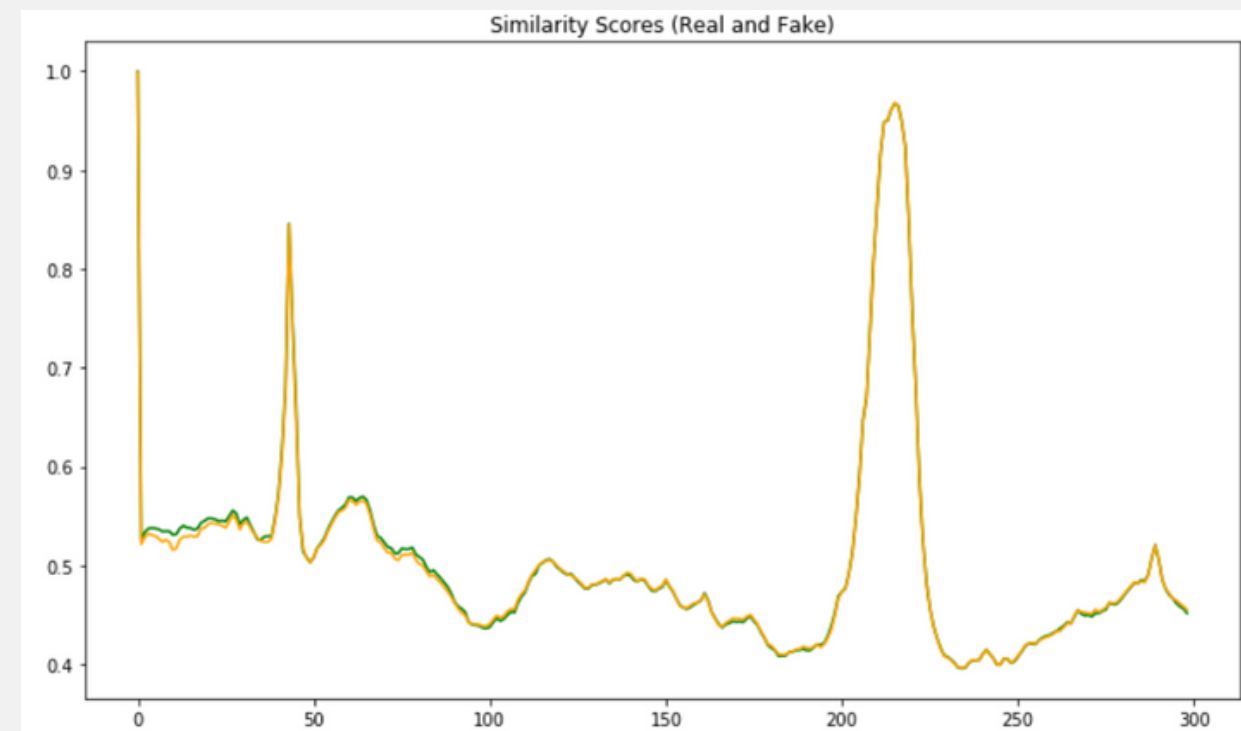
REAL 데이터의 분포가 현저히 적음을 확인.
-> real 데이터 증강 기법, 추가 데이터 삽입 고려(불가능)

2.EDA

프레임 유사도 분석(vudstovrck.mp4)



Real, Fake 프레임 유사도 분석



-> 전체 프레임을 촬영하면 원본 이미지의 유사성 점수는 거의 동일.
얼굴 프레임만 촬영했을때 프레임 유사도가 다른지 확인해야함

2.EDA

Fake 영상 샘플 확인



-> 얼굴쪽의
부자연스러운 현상 확
인



-> 얼굴쪽의
급격한 프레임 변화 확
인

3.데이터 전처리 계

획

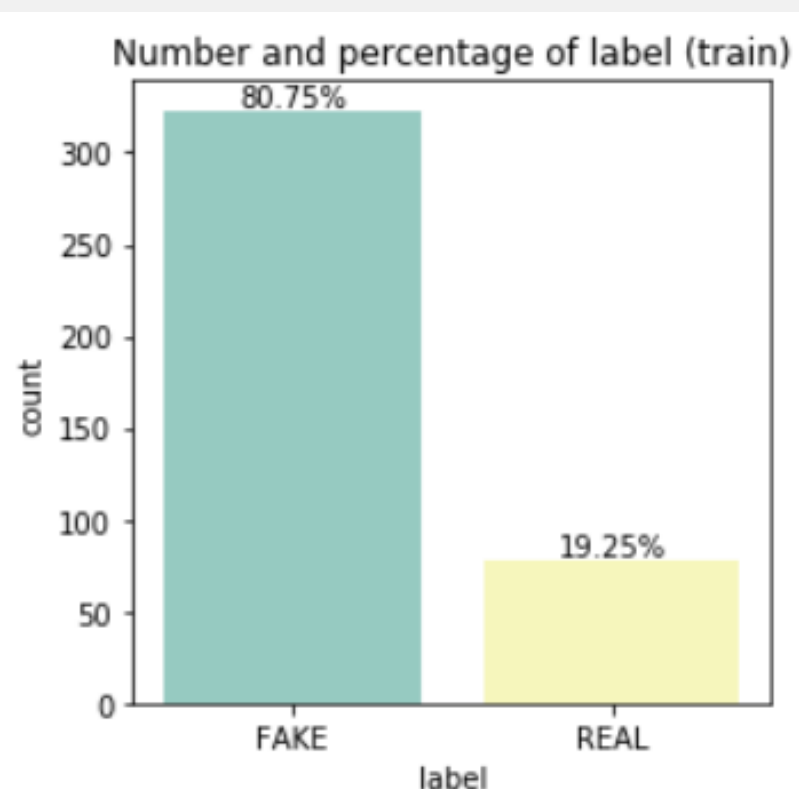
Face Labeling

프레임 처리

정규화

불균형
해소 방안

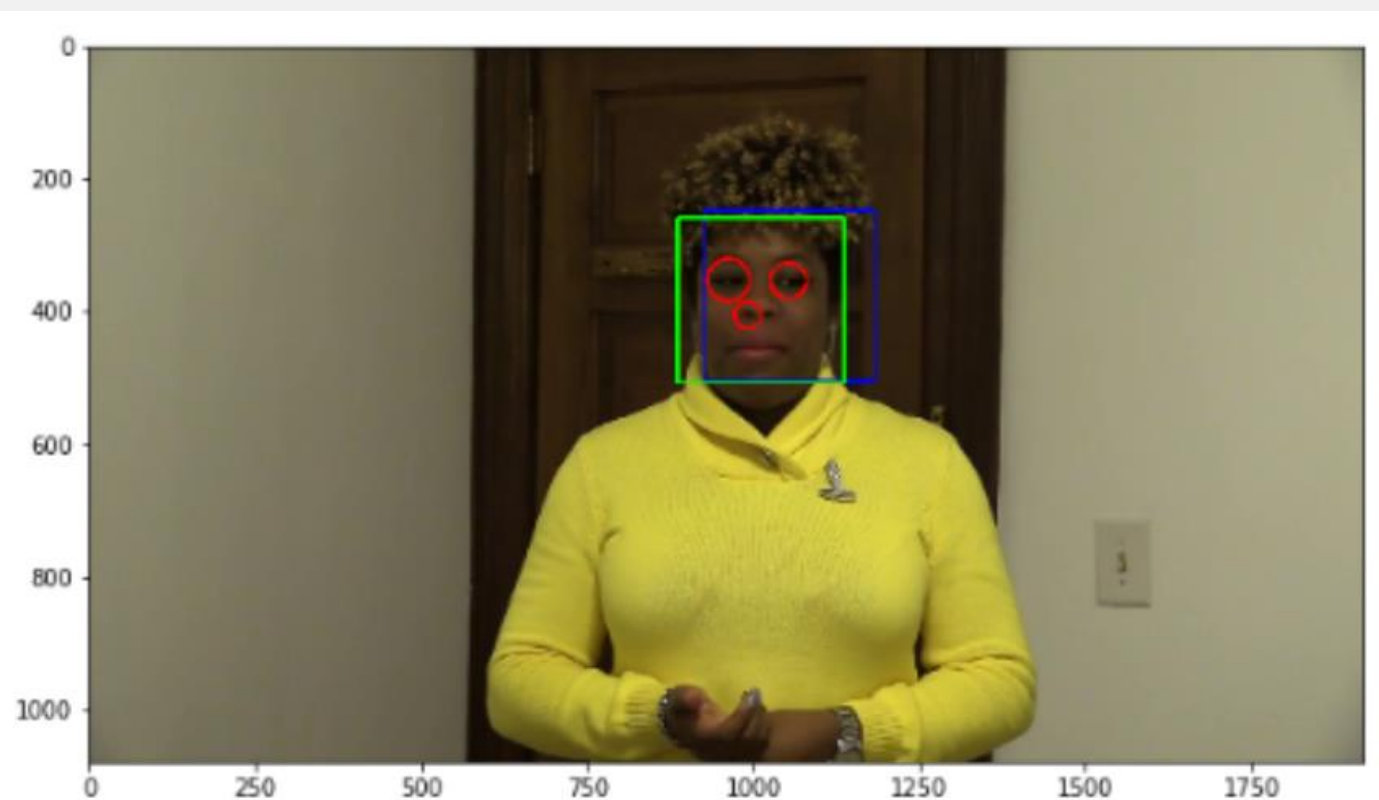
3.데이터 전처리



Real 데이터가 부족함에 따른 데이터 증강 기법 탐색:

1.Flip, Rotate와 같은 위치 변환에 따른 증강

2.Brightness 밝기 조절 , Saturation색조 조절에 따른 증강



Face Detection

OpenCV에서 얼굴(정면,측면), 눈, 웃음을 추출하는 기능 가져와서 활용

.

- 정면 : 녹색 직사각형
- 눈 : 빨간색 원;
- 미소: 빨간색 사각형;
- 측면: 파란색 직사각형.

데이터 전처리

얼굴 라벨링:

- 목적: 데이터셋에서 각 이미지 또는 비디오의 얼굴에 대한 라벨을 할당하여 모델이 딥페이크인지 실제인지 구분할 수 있도록 함.

프레임 처리:

- 프레임 추출: OpenCV를 사용하여 비디오에서 일정한 간격으로 프레임을 추출.
- 얼굴 크롭: MTCNN 또는 Dlib을 사용하여 각 프레임에서 얼굴을 감지하고, 해당 얼굴 영역만을 자름.
- 이미지 크기 조정: CNN 모델의 입력 크기에 맞추어 각 프레임을 일정한 크기로 조정.
(예: 128x128)

데이터 전처리

정규화:

- 픽셀 값 정규화: 각 이미지의 픽셀 값을 0~1 범위로 정규화하거나, 평균을 0, 표준편차를 1로 맞추는 방식으로 정규화.
- 데이터 증강과 함께 적용: 정규화를 데이터 증강 과정에서도 함께 적용하여 일관성을 유지.

불균형 해소:

- 데이터 증강: 딥페이크 이미지의 수를 늘리기 위해 회전, 크기 조정, 노이즈 추가 등의 기법을 사용.
- 오버샘플링: 부족한 클래스(예: 진짜 이미지)의 데이터를 복제하거나, 작은 변형을 추가.
- 언더샘플링: 과잉 클래스(예: 딥페이크 이미지)에서 일부 데이터를 랜덤으로 제거하여 균형을 맞춤.
- 가중치 조정: 손실 함수에서 각 클래스의 가중치를 조정하여, 덜 대표적인 클래스에 더 많은 중요성을 부여.

4.분류 방식, 모델 선정



FaceForensics++ 선택 이유

다양한 모델의 검증된 성능:

FaceForensics++는 Xception, EfficientNet, ResNet 등이 학습되었으며, 딥페이크 탐지에서 높은 성능을 보인 것으로 입증. 검증된 알고리즘을 활용함으로써 데이터셋의 탐지 성능을 더욱 향상시킴.

전처리와 데이터 구조 참고

얼굴 검출 및 크롭, 데이터 증강 방법등의 전처리 과정은 FaceForensics++에서 효율적으로 구현된 방식. 이를 기반으로 데이터셋에 적용할 수 있고, 데이터셋에 맞춘 수정도 가능.
예시로 비디오에서 Dlib, MTCNN, OpenCV 같은 라이브러리를 활용해서 프레임을 추출하여 얼굴을 감지하고, 필요에 따라 크기를 조정하는 등의 전처리 기법이 존재.

	Raw				Compressed 23				Compressed 40			
	DF	F2F	FS	NT	DF	F2F	FS	NT	DF	F2F	FS	NT
Steg. Features + SVM [27]	99.03	99.13	98.27	99.88	77.12	74.68	79.51	76.94	65.58	57.55	60.58	60.69
Cozzolino <i>et al.</i> [17]	98.83	98.56	98.89	99.88	81.78	85.32	85.69	80.60	68.26	59.38	62.08	62.42
Bayar and Stamm [10]	99.28	98.79	98.98	98.78	90.18	94.93	93.14	86.04	80.95	77.30	76.83	72.38
Rahmouni <i>et al.</i> [51]	98.03	98.96	98.94	96.06	82.16	93.48	92.51	75.18	73.25	62.33	67.08	62.59
MesoNet [5]	98.41	97.96	96.07	97.05	95.26	95.84	93.43	85.96	89.52	84.44	83.56	75.74
XceptionNet [14]	99.59	99.61	99.14	99.36	98.85	98.36	98.23	94.5	94.28	91.56	93.7	82.11

Table 4: Accuracy of manipulation-specific forgery detectors. We show the results for raw and the compressed datasets of all four manipulation methods (DF: DeepFakes, F2F: Face2Face, FS: FaceSwap and NT: NeuralTextures).

다양한 모델

XceptionNet:

심층 분리 합성곱을 통해 연산 효율성을 극대화하고, 얼굴 영역의 텍스처 변화를 잘 감지.

EfficientNet:

네트워크의 크기를 균형적으로 확장하는 방식으로, 크기, 깊이, 너비 등을 확장하면서도 연산 효율성을 유지.

ResNet:

미세한 조작 패턴을 안정적으로 학습, 딥페이크의 섬세한 변화를 잘 탐지.

VGG16/VGG19:

매우 깊은 CNN 구조로서 특징 추출에 강하고, 고해상도 이미지에서의 변조 감지에 효과적.

다양한 모델

DenseNet:

각 레이어가 이전 모든 레이어의 출력을 입력으로 사용하는 구조로, 정보가 효과적으로 전달되고 특징이 재사용. 상대적으로 적은 파라미터로도 높은 성능을 낼 수 있으며, 과적합을 줄이는 데 유리.

Yolo_v2:

전체 이미지를 한 번에 처리하여 객체를 탐지하므로 속도가 빠름.
다양한 크기의 객체를 인식할 수 있도록 앵커 박스를 도입하여, 높은 탐지 정확도를 제고

MTCNN:

얼굴 감지, 얼굴 랜드마크 감지, 얼굴 인식을 동시에 수행할 수 있는 구조, 다양한 포즈, 각도 및 조명 변화에 강인.

다양한 모델

InceptionNet_V2:

다양한 크기의 커널을 사용하여 서로 다른 수준의 특징을 동시에 추출할 수 있는 구조, 다양한 합성곱 레이어와 풀링 레이어를 조합하여 각 단계에서 보다 정교한 특징을 학습 가능.

MobileNet_V3:

경량화된 모델로, 높은 연산 효율성을 제공, 제한된 자원에서도 잘 작동.
하이퍼 파라미터 튜닝을 통해 최적화된 네트워크 구조를 제공하여, 성능과 속도 간의 균형을 잘 맞춘다.

앙상블 모델

- 일반적으로 3~5개의 모델을 사용.
 - 3개 모델: 효율적이며, 적절한 성능 향상을 기대 가능. 예를 들어, Xception, EfficientNet, ResNet과 같은 모델을 결합 가능.
 - 5개 모델: 더 다양한 모델을 사용하여 성능을 높일 수 있지만, 연산 자원이 더 많이 필요하고 모델 간 상관성을 잘 관리가 필요.
- 선택 기준:
 - 성능: 각 모델이 얼마나 딥페이크 탐지에 적합한지 평가.
 - 다양성: 동일한 유형의 모델보다 서로 다른 구조를 가진 모델을 선택하여 다양한 특징을 학습하게 하는 것이 중요.
 - 연산 자원: 앙상블 학습에서는 여러 모델을 병렬적으로 학습시키고 예측을 결합하기 때문에, 연산 자원이 충분한지 확인.
- Stacking 앙상블:
 - 여러 모델을 학습한 후, 이들의 예측 결과를 입력으로 사용하여 메타 모델(예: 로지스틱 회귀, 작은 신경망)을 학습하는 방식. 서로 다른 모델이 서로의 장점을 보완하여 성능을 극대화하는 데 효과적.

위의 기법을 통해 5개 정도의 딥페이크 탐지 모델을 결합하여 탐지의 정확성과 안정성을 높이는 것 목표



감사합니다