2018312856 KoSungwon

I have made 8 notebook files, corresponding to followings:

1. Low Quality data - face2face dataset / ResNet18

2. Low Quality data - face2face dataset / EfficientNetB0

3. Low Quality data - NeuralTexture dataset / ResNet18

4. Low Quality data - NeuralTexture dataset / EfficientNetB0

5. High Quality data - face2face dataset / ResNet18

6. High Quality data - face2face dataset / EfficientNetB0

7. High Quality data - NeuralTexture dataset / ResNet18

8. High Quality data - NeuralTexture dataset / EfficientNetB0

due to importation problem with timm, I used ResNet18 and EfficientNetB0. All eight files are conducted in same way, but they just differ from model selection, and data. I will explain my code for first Low Quality data – face2face dataset / ResNet18

1. import modules.

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

also, I checked for GPU resources, since I did it on my local.

2. dataset file

original dataset file doesn’t fit to PyTorch’s DataLoader. To use dataloader, I have to make class folders.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

so I made it into two class folders.

3. DataLoader

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

I defined dataloader. For training, I used data augmentation. It resizes original image, and filp image for 50% chance. Also, Normalize image data with given values.

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

I checked train dataset, and it correctly found dataset. (Real 600 + Fake 600)

Then, I checked train data.

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

result:

인간의 얼굴, 다채로움, 콜라주, 사람이(가) 표시된 사진

자동 생성된 설명

As you can see, images are distorted as I intended.

4. Model

텍스트, 스크린샷, 소프트웨어, 운영 체제이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

For ResNet, I used pretrained model in Torchvision. I froze weights in Feature extraction parts with convolutional layers, and added new classifier layer at the top, with 2 output for binary classification.

For EfficientNet, I used pretrained model from github.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

5. Train model

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

I used Adam optimizer, and CrossEntropyLoss function. when training, I track model’s accuracy and record the best one. After 100 epochs, I saved the best model, and store them in models folder.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

Training process looks like:

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

6. test model.

first, train loader. when training, It doesn’t distort images.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

plot loss, and accuracy of model training process.

텍스트, 스크린샷, 그래프, 디스플레이이(가) 표시된 사진

자동 생성된 설명

스크린샷, 텍스트, 그래프이(가) 표시된 사진

자동 생성된 설명

2. Accuracy, F1 score, precision, recall of each cases.

2-1. Low Quality data – face2face dataset / ResNet18

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2-2. Low Quality data - face2face dataset / EfficientNetB0

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2-3. Low Quality data – NeuralTexture dataset / ResNet18

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2-4. Low Quality data – NeuralTexture dataset / EfficientNetB0

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2-5. High Quality data – face2face dataset / ResNet18

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2-6. High Quality data - face2face dataset / EfficientNetB0

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2-7. High Quality data – NeuralTexture dataset / ResNet18

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2-8. High Quality data – NeuralTexture dataset / EfficientNetB0

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Summary. Each score is written in perspective of (0) – fake detection.

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명

Overall, models show better result in Low-Quality dataset. In perspective of models, ResNet performed better in Low Quality dataset. But Efficient net with High quality dataset showed very good results with accuracy of 91% .

I think too much resolution cause overfitting to models, referring from results above. But it is surprising that trained models can actually tell which is fake and real.