305 – Unveiling the Illusion

Team Information

Team Name : ForensicGPT

Team Member Eungchang Lee, Donghyun Ha, Hyunwoo Shin, Jinhyeok Son

Email Address : forensicgpt@googlegroups.com

Instructions

Description DFC has been hosting an annual photography exhibition contest consistently each year. After last year's winning entry was revealed to be created by an AI, to avoid repeating the same mistake, they have entrusted you with the task of determining whether the submitted artworks are generated by human or AI. Please discern whether each submitted artwork is a human-generated or an AI-generated creation.

Target	Hash (MD5)
ImageSet.ad1	083C11A3D77C073AC1A4A32AF1230531

Questions

1) Explain the approach for detecting AI-generated creation and submit the code for it. (100 points)
   - We provide you with the information about the photos submitted for the 2022 DFC Contest, indicating which ones were revealed to be generated by human and generated by AI.
   - Scores will be assigned based on the "F1-Score" when using the submitted code to classify the photos of the artworks for the 2022 DFC contest.
   - Additional points may be awarded for the submitted code and

ideas.

2) Use the method devised in the previous question to determine whether each image has been generated by AI or not. (200 points)

   - We provide you with the photos of the artworks submitted for the 2023 DFC Contest.
   - Scores will be assigned based on "F1-Score" when using the submitted code to classify the photos of the artworks for the 2023 DFC contest.

Teams must:

- Develop and document the step-by-step approach used to solve this problem to allow another examiner to replicate team actions and results.
- Specify all tools used in deriving the conclusion(s).

Tools used:

Name:	Tensorflow	**Publisher:**	Google
Version:	2.10.0		
URL:	https://www.tensorflow.org		

Name:	Python	**Publisher:**	Python
Version:	3.9.17		
URL:	https://www.python.org		

Name:	HashTab	**Publisher:**	Implbits Software
Version:	6.0.0		
URL:	https://implbits.com		

Name:	FTK Imager	**Publisher:**	exterro
Version:	4.7.1.2		
URL:	https://www.exterro.com/ftk-imager		

Hardware:

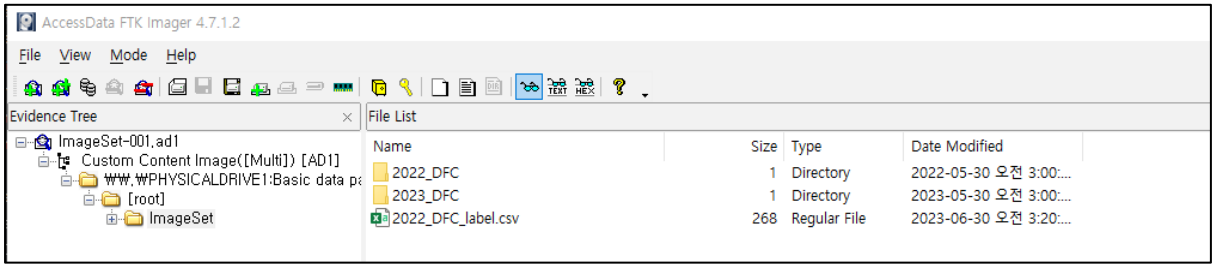
CPU:	Intel Core i9-9900KS
GPU:	NVIDIA GeForce RTX 3080
Memory:	32GB

Step-by-step methodology:

먼저 다운로드 받은 이미지셋 파일 ImageSet.ad1의 md5 해시 값이 원본과 일치함을 HashTab 도구를 통해 확인했습니다.

[그림 1] 분석 파일의 해시 값 확인

1) Explain the approach for detecting AI-generated creation and submit the code for it. (100 points)



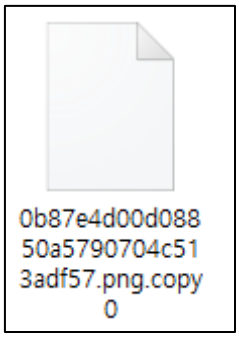
[그림 2] 파일 확인

주어진 파일을 FTK Imager를 이용해 확인해보면 위와 같이 2022년과 2023년의 이미지 파일, 2022년도 이미지 파일에 대한 레이블이 명시되어 있는 csv 파일이 존재하는 것을 확인할 수 있습니다.

	A	B	C
1	No	Name	AI_generat
2	1	68736f54b	0
3	2	96898eaa4	0
4	3	2cdd3a34(	0
5	4	24c6efda2	0
6	5	fb907fcf9a	0

[그림 3] csv 파일 내부

모델 생성을 위해 먼저 주어진 이미지 파일과 레이블을 기반으로 데이터셋을 제작했습니다. 먼저 해당 이미지를 덤프화는 과정에서 발생한 불필요한 데이터들을 확인할 수 있었습니다.



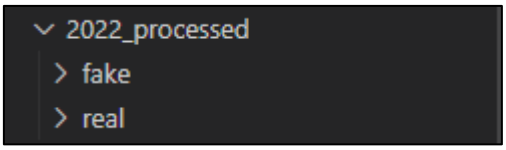
[그림 4] 불필요 데이터

데이터 전처리 과정을 위해 유효한 사진 파일만을 사용하기 위해 아래와 같은 스크립트를 작성했습니다. 이 코드는 디렉터리에 존재하는 파일 중 png 파일을 선별해 실제 인식이 가능한 이미지인지 확인해 저장합니다.

```python
import os
from PIL import Image

image_dir = "/home/hyunvis/dataset/2022_DFC"
output_dir = "/home/hyunvis/dataset/2022_DFC_fixed"

os.makedirs(output_dir, exist_ok=True)

png_files = [file for file in os.listdir(image_dir) if file.lower().endswith('.png')]

for png_file in png_files:
    input_path = os.path.join(image_dir, png_file)
    output_path = os.path.join(output_dir, png_file)

    try:
        img = Image.open(input_path)
        img.save(output_path)
        print(f"Processed: {png_file}")
    except Exception as e:
        print(f"Error processing {png_file}: {e}")
```

[그림 5] 데이터 전처리 코드

전처리가 완료된 이미지 파일을 데이터셋으로 만들기 위해 2022_DFC_label.csv파일을 이용해 fake 및 real 클래스로 분류하는 작업을 진행했습니다.



[그림 6] 데이터 분류

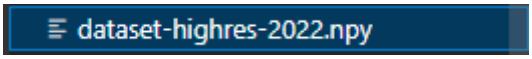
이미지 전처리 작업과 분류를 마친 뒤에는 해당 이미지를 데이터셋화 하는 작업을 진행했습니다. Real, fake로 구분된 디렉터리를 기반으로 기존에 존재하는 이미지들을 128*128 해상도를 가진 이미지로 정규화하고, 레이블을 지정했습니다. 테스트셋의 사이즈는 0.2로 지정했습니다.

```python
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split

imagePath = '/home/hyunvis/dataset/2022_processed'
categories = ["real", "fake"]
nb_classes = len(categories)
image_w = 128
image_h = 128
X = []
Y = []

for idx, cate in enumerate(categories):
    label = np.zeros(nb_classes)
    label[idx] = 1
    image_dir = os.path.join(imagePath, cate)

    for filename in os.listdir(image_dir):
        img = cv2.imread(os.path.join(image_dir, filename))
        img = cv2.resize(img, (image_w, image_h))
        X.append(img)
        Y.append(label)
```

[그림 7] 데이터셋 제작 1

```
X = np.array(X)
Y = np.array(Y)

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

dataset = {
    'X_train': X_train,
    'X_test': X_test,
    'Y_train': Y_train,
    'Y_test': Y_test
}

np.save("./dataset-highres-2022.npy", dataset)
```

[그림 8] 데이터셋 제작2



[그림 9] 데이터셋 파일

데이터셋은 다음과 같이 npy 파일 형태로 저장됩니다.

그 후, 제작된 데이터셋을 기반으로 AI 생성 이미지와 사람이 생성한 이미지를 구별하기 위해 CNN 모델 기반 이미지 분류 모델을 생성했습니다. CNN(Convolutional Neural Network) 모델은 이미지의 구조적 특성과 합성곱 연산을 통한 처리 및 분류 작업에 효과적인 모델이기 때문에, AI 생성 이미지 판단에 적용하였습니다.

```
import numpy as np
import tensorflow as tf
from sklearn.metrics import classification_report, f1_score

loaded_dataset = np.load("./dataset-highres-2022.npy", allow_pickle=True).item()
X_train = loaded_dataset['X_train']
X_test = loaded_dataset['X_test']
Y_train = loaded_dataset['Y_train']
Y_test = loaded_dataset['Y_test']

X_train = X_train / 255.0
X_test = X_test / 255.0
```

[그림 10] 데이터셋 불러오기 및 정규화

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(128, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(256, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(2, activation='softmax')
])
```

[그림 11] CNN 모델 정의

모델의 정확도 향상을 위해 다수의 레이어를 순차적으로 쌓았습니다. Conv2D 레이어는 2차원상의 합성곱 연산을 수행하고, MaxPooling2D 레이어는 MaxPooling 작업을 수행해 이미지의 크기를 줄입니다. Flatten 레이어는 다차원인 배열을 1차원으로 만들어 주는 역할을 하고, Dense 레이어는 Fully-Connected 레이어로서 이전 계층이 모든 뉴런과 연결되게 합니다. Dropout 레이어는 모델 학습시 과적합을 막기 위해 0.5의 값을 적용했습니다. 마지막으로 Softmax 활성함수를 사용해 확률을 계산하도록 설계했습니다.

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(X_train, Y_train, epochs=100, batch_size=128, validation_split=0.1)

test_loss, test_accuracy = model.evaluate(X_test, Y_test)
print("Test loss", test_loss)
print("Test accuracy:", test_accuracy)

model.save("model.h5")
```

[그림 12] 모델 컴파일 및 저장

Learning Rate를 자동으로 조절할 수 있는 Adam Optimizer를 사용했고, 학습횟수는 하드웨어 상황에 적절히 맞추어 100회, 배치 사이즈는 128을 적용했습니다. 검증을 위한 validation split은 0.1의 비율을 적용했습니다.

[그림 13] 모델 파일

모델은 다음과 같이 h5 파일 형태로 저장됩니다.

결과를 산출하기 위해 f1_score함수와 classification_report 함수를 활용했습니다.

```
Y_pred = model.predict(X_test)
Y_pred_classes = np.argmax(Y_pred, axis=1)
Y_true = np.argmax(Y_test, axis=1)

f1 = f1_score(Y_true, Y_pred_classes, average='weighted')

print(classification_report(Y_true, Y_pred_classes, target_names=['Real', 'Fake']))
print("F1-score: ", f1)
```

[그림 14] 결과 산출

```
42/42 [==============================] - 0s 7ms/step
              precision    recall  f1-score   support

        Real       0.94      0.98      0.96      1193
        Fake       0.80      0.50      0.62       147

    accuracy                           0.93      1340
   macro avg       0.87      0.74      0.79      1340
weighted avg       0.93      0.93      0.92      1340

F1-score:  0.9243791289454368
```

그림 15 결과 도출

위와 같은 절차를 거쳐 0은 6473개 1은 332개로 분류를 수행함으로써 결과적으로 0.924의 F1-Score를 얻을 수 있었습니다.

2) Use the method devised in the previous question to determine whether each image has been generated by AI or not. (200 points)

1번에서 진행한 것과 같이 2023 디렉터리에 존재하는 불필요한 파일을 지우는 전처리 과정을 진행한 뒤, 이전에 생성한 모델을 기반으로 2023 이미지 파일에 대한 추론을 진행했습니다.

```
import os
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import load_img, img_to_array
import csv

loaded_model = tf.keras.models.load_model("model.h5")
image_dir = "/home/hyunvis/dataset/2023_DFC"
image_files = [file for file in os.listdir(image_dir) if file.lower().endswith(('.png'))]
```

[그림 16] 모델 불러오기

.copy0 파일을 제외하고 png 파일만을 대상으로 추론을 진행합니다.

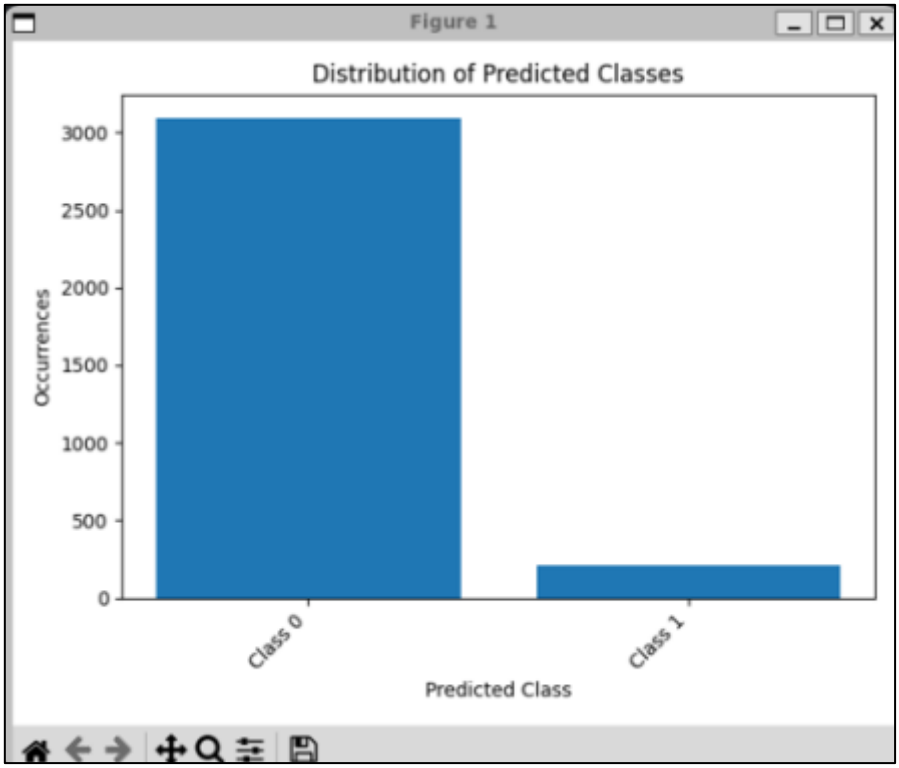
```
preprocessed_images = []
for image_file in image_files:
    image_path = os.path.join(image_dir, image_file)
    img = load_img(image_path, target_size=(128, 128))
    img_array = img_to_array(img) / 255.0
    preprocessed_images.append(img_array)

preprocessed_images = np.array(preprocessed_images)
predictions = loaded_model.predict(preprocessed_images)
predicted_classes = np.argmax(predictions, axis=1)
output_file = "predictions_2023.csv"

with open(output_file, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(["Image", "Predicted Class"])
    for i, image_file in enumerate(image_files):
        writer.writerow([image_file, predicted_classes[i]])

print(f"Predictions saved to {output_file}")
```

[그림 17] 추론 및 csv 출력

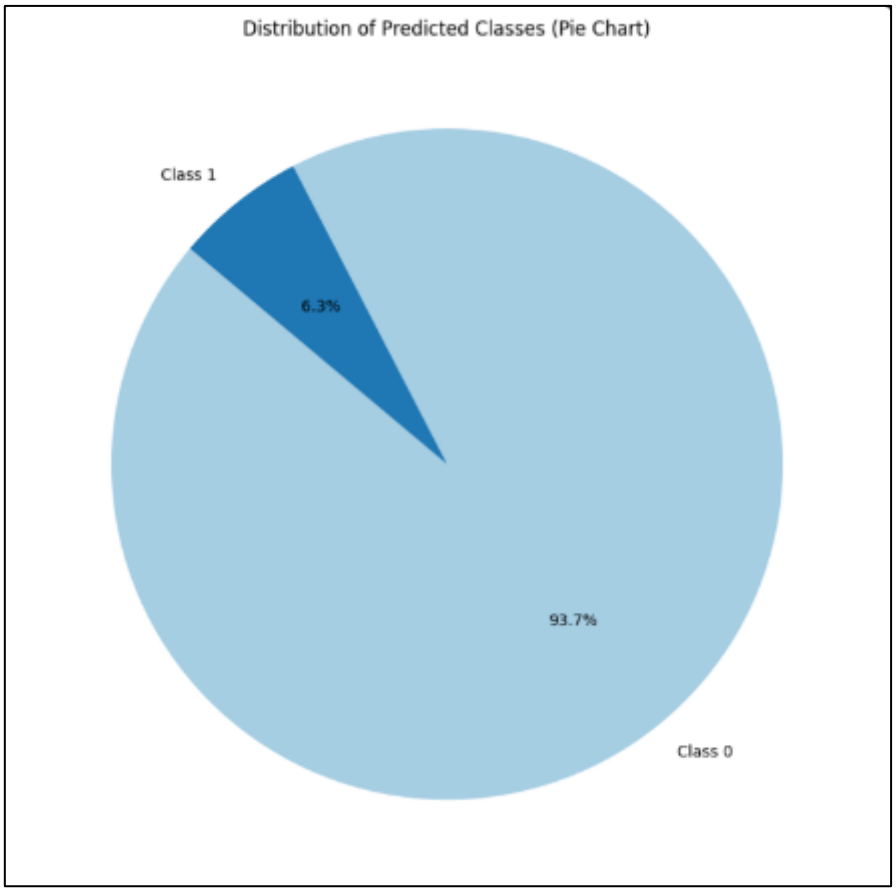
저장한 모델을 불러오고 2023 데이터셋에 대한 이미지 사이즈 조절과 같은 전처리 작업을 진행했습니다. 모델을 기반으로 추론을 진행한 뒤, 예측한 클래스를 csv파일로 저장해 알기 쉽도록 수행하였습니다.

Image	Predicted Class
31f42752cb7a22404f198d589841ae71.png	0
fb9e4585424f4b41cc639b41c0fc6f63.png	0
ff55fdb209405c167f591d435159783f.png	0
86a6e3f295023f40f86184ace9df483f.png	1
aa427f530cb7c4cff6fec686bdb18a9b.png	0
ca5f2edfb4730bb8a59fe4c55f65d51c.png	1
3d37468a030f941024045c9344c03351.png	0
207aad93e824e1e312746c9b1f60c47d.png	1
c9e8e6be94436065c4a489c43b32bf86.png	0

[그림 18] 예측 csv



[그림 19] 0, 1 예측 시각화 - 1



[그림 20] 0, 1 예측 시각화 - 2

이처럼 2023 데이터셋에 대해 추론을 진행하였을 때, 시각화 결과를 통해서도 알 수 있듯이 0이 3024개, 1이 276개로 이는 2022 데이터 셋에 대한 예측 비율과 비슷함을 알 수 있습니다.

따라서, 2023 데이터셋의 경우 Ground truth 값이 존재하지 않기 때문에, 해당 단계에서는 직접 F1 Score를 산출할 수 없지만, 성능 평가 시 2022 데이터셋에서 산출한 F1 score와 비슷한 결과값을 가질 수 있을 것으로 기대됩니다.

1번 문항과 2번 문항에 대해 작성한 코드는 함께 첨부하였으며, Readme.md를 통해 2022 데이터 셋에 대한 f1-score를 산출할 수 있는 process와, ground truth가 있다면 2023 데이터 셋에 대해 f1-score를 산출할 수 있는 process를 확인할 수 있습니다.