"AI News That's Fit to Print"

The AI Newsletter

Bimonthly

Late Edition

K-water연구원 AI연구센터에서 정기적으로 발간하는 뉴스레터입니다. AI 분야의 뉴스 뿐 아니라 코딩실습과 기술정보 등 다양한 내용과 난이도로 담아냈습니다.

AI Newsletter No.11B

AI LAB, THURSDAY, OCT 17, 2024

AI Tech

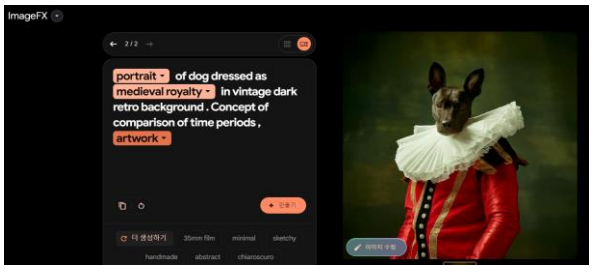
Tips

구글의 최신 AI 이미지 생성기 ImageFX

Google이 최신 AI 이미지 생성 모델인 Imagen 3를 기반으로 개발한 이미지 생성형 AI인 ImageFX가 최근 공개되었다. ImageFX는 Google 계정만 있으면 누구나 바로 사용할 수 있다. 텍스트 프롬프트에서 고품질 이미지를 생성하는 잠재확산 모델인 Imagen 3는 평가시점에 다른 최첨단(SOTA) 모델보다 우수성이 인정되었고, 특히 출력에 영향을 미치는 프롬프트의 특정 부분을 조정하여 사용자가 원하는 의도에 근접한 이미지를 생성하는 특징을 가지고 있다. 그러나 여전히 AI가 생성한 이미지 이상 징후를 보여준다. 일부 사진에 손가락이 많거나 얼굴이 왜곡되고 의미 없는 텍스트를 생성하기도 한다. 하지만 그동안 저작권이 있는 콘텐츠 생성을 거부하였지만 코카콜라, 마리오, 피카츄와 같은 상표가 있는 캐릭터 및 로고를 자유롭게 생성하고 활용할 수 있다. 또한 해당 사이트에서는 시범적으로 음악생성(MusicFX), 영상생성(VideoFX) 및 멀티모달 텍스트생성(TextFX) 기능도 활용해 볼 수 있다.

QUICK TIPS ImageFX를 사용하는간단한 방법

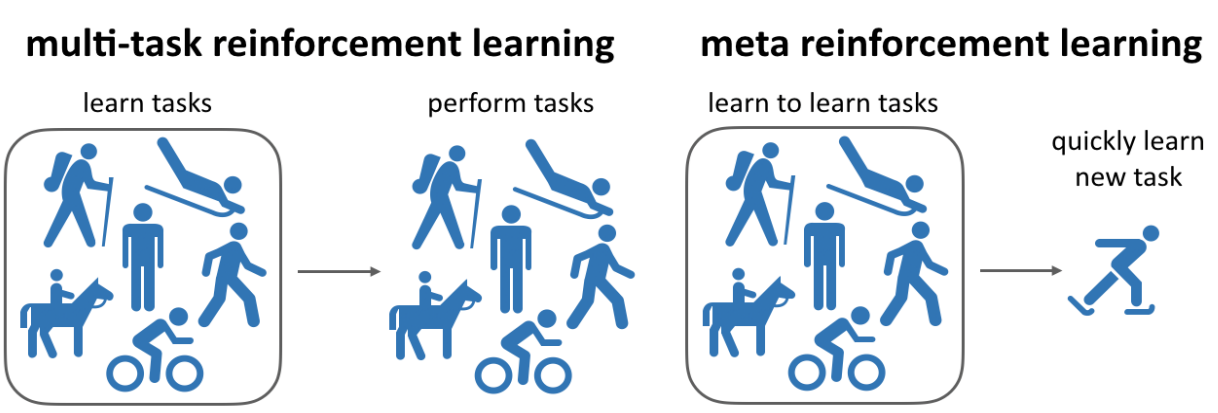
- Google의 AI Test Kitchen에서 ImageFX로 이동
- Google 계정 로그인 후 사용



ImageFX에서 텍스트 입력으로 이미지 생성 모습

메타러닝(Meta-Learning)

최근 AI기술 특히 딥러닝기술이 많은 분야의 혁신에 영향을 미치고 있으나 딥러닝기술의 성능 향상을 위해서는 여전히 크고 다양한 데이터셋과 성능이 좋은 하드웨어를 필요로 하는 것이 현실이다. 이러한 딥러닝기술의 단점을 보완하고, 사람과 같이 적은 데이터를 활용하여 어떤 개념을 상대적으로 빠르게 학습 할수 있는 방법에 대해서 연구하는 분야가 "메타러닝(Meta-Learning)"이다. 메타러닝이라는 용어는 컴퓨터 공학, 인공지능 분야뿐만 아니라 심리학, 행동 경제학 등 다양한 분야에서 사용되고 있으며, 인공지능 분야에서는 새로운 개념 또는 태스크를 빠르게 학습하기 위해 "어떻게 학습해야 할지를 학습(learning to learn)"하는 방법으로서, 새로운 태스크를 더 빨리 학습하기 위해 이전의 학습 경험을 적극적으로 활용하는 방법이다. 예를 들면(왼쪽 그림 참조), 기존의 학습 방법은 하나의 모델이 다양한 여러 태스크를 잘 학습하고 테스트시 학습한 것과 같은 여러 태스크를 잘 수행하는 것을 목표로 한다. 반면에 메타러닝은 학습시 멀티 태스크 러닝과 같이 다양한 여러 태스크를 학습하긴 하지만, 학습시 그 태스크를 외우기만 하는 것이 아니라 학습하는 방법을 학습하여, 테스트시 학습 때 보지 못했던 새로운 태스크가 주어졌을 때 이를 빠르게 학습할 수 있는 방법이다. 최근 인공지능에서는 거대언어모델과 같이 생성형 인공지능모델에서 인공지능의 성능 향상을 위해 "Few-Shot Learning" 등을 적용하는데 이것이 메타러닝의 좋은 예시라고 할 수 있다.



멀티 Tasks러닝과 메타 러닝의 차이점 (그림: https://meta-world.github.io)

정창훈, 이승현, 이동민, 장성은, 이승재, 윤승제 (2022) 모두를 위한 메타러닝, 위키북스

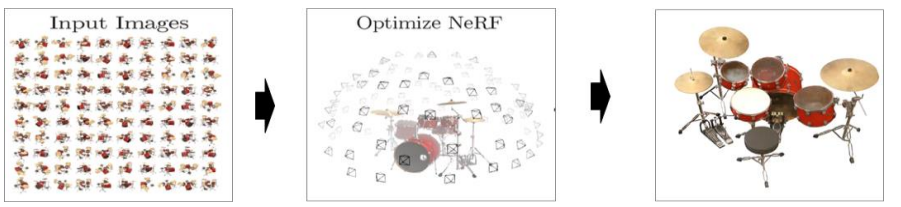
가성비 높은 3차원 영상 제작, NeRF와 3D 가우시안 스플래팅

딥러닝을 활용한 3차원화, NeRF

NeRF(Neural Radiance Field)는 신경망을 활용한 빛의 장을 가시화하는 기술로 사람의 눈으로 보이는 가시영역을 AI를 활용해 3D로 구현하는 기술이며, 2020년 개발되어 지속적으로 발전 중이다. 여러 장의 2D 이미지를 활용하여 새로운 시점에서의 물체 이미지를 만들어 3D를 생성하는 방법으로, N개의 시점에서의 불연속적인 2D 이미지를 입력받아 이미지가 연속적으로 구성될 수 있도록 임의의 시점에서의 새로운 이미지를 만들어 낸다.

수학적 3차원화, 가우시안 스플래팅

3D Gaussian Splatting은 2022년 인공신경망을 활용한 NeRF의 성능을 개선하여 수학적으로 3D를 구현하는 방법이다. 사진을 통해 3D의 Point Cloud정보를 취득하고, 이를 중심으로 3D 공간상에 수많은 3D Gaussian을 생성하며, 이를 객체간 경계를 기준으로 Clone과 Split을 반복함으로써 실제와 유사한 3D 공간을 생성한다. K-water연구원 AI연구센터는 이러한 기술을 활용해 정수장, 댐 등의 시설물에 적용하는 연구를 수행 중이다.



<NeRF를 이용한 3차원 영상 제작 과정>

<3D 가우시안 스피래팅을 이용한 3차원 영상 제작 과정>

<공주정수장 막여과동>

<보현산댐>

Hands-on AI Project

통계적 방법에 의한 이상치 탐색

이상치(Outlier)는 데이터의 수집, 입력, 측정과정과 이 외 다양한 원인으로 발생할 수 있으며, 머신러닝 모델 성능에 악영향을 미칠 수 있다. 따라서 데이터 전처리(Data Processing) 과정에서 반드시 처리해주어야 하는데, 이번 Hands-On에서는 다양한 이상치 탐색(Outlier Detection) 방법 중에서 변수가 일변량(uni-variate)이며, 모수적(parametric)일 때 적합한 통계적 방법에 대해 다룰 것이다.

 # 필요 라이브러리 설치

- `%pip install pandas`
- `%pip install matplotlib`
- `%pip install statsmodels`

※ 위 3개의 라이브러리 설치 및 전체 Source 코드 및 결과는 분량상 생략, 아래 링크를 참고해주세요.

구글 코랩에서 열기(인터넷망)

* 입력데이터(Data.csv) : 대청댐 수질자료(2016~2023년)

① 라이브러리 선언

```
import pandas as pd
import matplotlib.pyplot as plt
import math
from statsmodels.stats.stattools import medcouple  # 수정 4분위수 방법
```

② 데이터 불러오기

```
# csv 파일을 데이터프레임으로 읽어들입니다.
Raw_Data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Data.csv',
index_col=0, sep=',',encoding='utf-8-sig',parse_dates=True)
display(Raw_Data) # 불러온 데이터파일을 나타낸다.
```

③ 이상값 탐색(Outlier Detection) 방법

```
# 이상치 탐색 방법들을 함수로 정의

def Z_Score_Detection(df):  # ① Z-score 방법
    threshold = 3  # 한계값 설정
    z_scores = (df - df.mean()).abs() / df.std() # Z-score 계산
    outliers = df.loc[z_scores > threshold] # 한계값을 초과한 이상치 슬라이싱
    return outliers
```

```
def Modified_Z_Score_Detection(df):  # ② 수정 Z-score 방법
    threshold = 3.5  # 한계값 설정
    med = df.median()  # 중앙값 계산
    MAD = ((df - med).abs()).median() #중앙값 절대편차(Median Absolute Deviation) 계산
    modified_z_scores = (0.6745 * (df - med) / MAD).abs()  # 수정 Z-score 계산
    outliers= df.loc[modified_z_scores > threshold]  #한계값을초과한이상치슬라이싱
    return outliers
```

```
def Quartiles_Detection(df):  # ③ 4분위수 방법
    threshold = 1.5  # 한계값 설정(1.5~3.0)
    q1 = df.quantile(0.25)  # Q1 계산
    q3 = df.quantile(0.75)  # Q2 계산
    iqr = q3-q1  # IQR 계산
   # 한계값을 초과한 이상치 슬라이싱
    outliers =  df.loc[(df < (q1 - threshold * iqr)) | (df > q3 + threshold * iqr)]
    return outliers
```

```
def Modified_Quartiles_Detection(df):  # ④ 수정 4분위수 방법
    threshold = 1.5  # 한계값 설정
   # medcouple(MC)가 NaN을 중앙값 계산에 포함시키지 않도록 NaN 삭제
    df.dropna(axis=0, inplace=True)
    q1 = df.quantile(0.25)  # Q1 계산
    q3 = df.quantile(0.75)  # Q2 계산
    iqr = q3-q1  # IQR 계산
    MC = medcouple(df)  # statsmodels 패키지를 이용해 medcouple(MC) 계산
    # MC값에 따라 한계값을 초과한 이상치 슬라이싱
    if MC >=0:
        outliers =  df.loc[(df < (q1 - threshold * math.exp(-4*MC) * iqr)) | (df > (q3 +
threshold * math.exp(3*MC)*iqr))]
    else:
        outliers =  df.loc[(df < (q1 - threshold * math.exp(-3*MC) * iqr)) | (df > (q3 +
threshold * math.exp(4*MC)*iqr))]
    return outliers
```

④ 이상값 탐색 실행

```
# 이상치 탐색 할 컬럼명과 단위를 딕셔너리로 만든다.
Detection_List = {'Water Temperature(Mun-Ui)':'℃','Water Temperature(Chu-
Dong)':'℃','Water Temperature(Hoe-Nam)':'℃','pH(Mun-Ui)':'pH','pH(Chu-
Dong)':'pH','pH(Hoe-Nam)':'pH','DO(Mun-Ui)':'mg/L','DO(Chu-Dong)':'mg/L','DO(Hoe-
Nam)':'mg/L','Transparency(Mun-Ui)':'Transparency','Transparency(Chu-
Dong)':'Transparency','Transparency(Hoe-Nam)':'Transparency'}

Mode = 1  # 1: Z-score, 2: 수정 Z-score, 3: 4분위법, 4: 수정 4분위법
# Column_List의 열을 차례로 Z-Score 함수에 넣어 받은 이상치를 Remove 데이터
  프레임에 차례로 붙여넣는다.
Remove = pd.DataFrame()
for col_name, ylable_name in Detection_List.items():
    if Mode == 1:  # Z-Score 함수를 호출한다.
        Remove = pd.concat([Remove, Z_Score_Detection(Raw_Data[col_name])], axis=1,
        join = 'outer')
    elif Mode == 2:  # 수정 Z-Score 함수를 호출한다.
        Remove = pd.concat([Remove,
Modified_Z_Score_Detection(Raw_Data[col_name])], axis=1, join = 'outer')
    elif Mode == 3:  # 4분위수 함수를 호출한다.
        Remove = pd.concat([Remove, Quartiles_Detection(Raw_Data[col_name])],
        axis=1, join = 'outer')
    elif Mode == 4:  # 수정 4분위수 함수를 호출한다.
        Remove = pd.concat([Remove,
        Modified_Quartiles_Detection(Raw_Data[col_name])], axis=1, join = 'outer')
```

⑤ 그래프로 나타내기

```
plt.figure(figsize=(15,5))
    plt.plot(Raw_Data[col_name], label='Data', color='gray', marker='o', markersize=1,
linestyle='-',linewidth=0.2)
    plt.scatter(Remove.index, Remove[col_name], color='red', label='Outliers', s=5,
zorder=2)
    plt.title(col_name)
    plt.xlabel('day')
    plt.ylabel(ylable_name)
    plt.legend(loc='best')
    plt.show()
```