

Google Cloud 와Hugging Face와 사례 비교

웹페이지의 " 리스트 불러오기"

박사학위대기중인 교신저자특

순서

- Google Drive vs. Hugging Face : 차이
- 바이오계열 연구자로서 이용 사례분석:
 - 구글드라이브에 저장된 파일의 목차 불러오기
 - Hugging Face 모델 목록 불러오기

가장 큰 차이

- Hugging Face는 기계 학습 모델과 데이터셋을 중심으로 하고,
Google Drive는 파일 저장과 협업에 중점을 둔다

Hugging Face

- 주로 자연어 처리에서 사용되는 기계 학습 모델의 허브.
- **핵심 기능:**
 - 수천 개의 사전 훈련된 모델에 대한 접근.
 - 연구원과 개발자들의 기여로 이루어진 커뮤니티 중심.
 - ML 모델의 훈련, 배포 및 협업을 위한 도구.

Google Drive

- 사용자가 온라인에 파일을 저장하고 어디서나 접근할 수 있게 해주는 클라우드 기반 저장 서비스
- 핵심 기능:
 - 협업을 위해 Google Docs, Sheets, Slides와 통합.
 - 파일 공유 및 권한 관리.

주요 차이점

- **사용:** Hugging Face는 ML 실무자와 연구자가 사용, Google Drive는 다양한 저장에 필요한 광범위한 사용자층
- **커뮤니티:** Hugging Face는 강력한 AI/ML 커뮤니티를 가지고 있으며, Google Drive는 그 외에도 다양한 사용자층을 대상.

바이오계열 연구자로서 이용 사례분석

구글 드라이브를 많이 씀

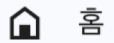
- 데이터 유출관련 (개인정보) 몹시 보안이 심함
- 외장하드 관련 승인 너무 어렵고 복잡
- 컴퓨터 간 이동은 구글 드라이브 쓸때가 많음



🔍 드라이브에서 검색



+ 신규



홈

▶ 내 드라이브

▶ 공유 드라이브

▶ 컴퓨터

👤 공유 문서함

🕒 최근 문서함

☆ 중요 문서함

⚠ 스팸

🗑 휴지통

☁ 저장용량

2TB 중 1.23TB 사용

추가 저장용량 구매

내 드라이브 ▾

유형 ▾

사람 ▾

수정 날짜 ▾

용량 많이 잡아먹는
대개 이미지 파일- 용량 매우 크다. 이름이 복잡하
며 틀리면 안됨, 의미가 있음

폴더



CORE



생활(비번등)



Colab Notebooks



case: my interesting ...



EBD



ENE_2023



LJW_ILC



연구



documents-똥컴



NGS



IHC_mrsx



IHC_mrsx



ihc_2



ENE (1)



Transformer



drive-download-202...



Patch



20230809



+ 신규

홈

내 드라이브

공유 드라이브

컴퓨터

공유 문서함

최근 문서함

중요 문서함

스팸

휴지통

저장용량

2TB 중 1.23TB 사용

추가 저장용량 구매

내 드라이브

유형

사람

수정 날짜

파일

이런 파일이 200-300개씩 있음 - 이름?

21SC007854;B;1;;FC...

21SC007792;B;1;;FC...

21SC007785;B;1;;FC...

21SC007780;B;1;;FC...

21SC008825

21SC008825;A;1;;FF...

23S_059633;A;1;;FA0...

23S_039796;A;6;;FA...

23S_044043;A;2;;FA...

23S_052780

원하던 것

- 이 파일들의 이름을 모두 추출해서 엑셀로 만들고 싶다.

접근과정

- 구글 드라이브 access 는 허깅페이스로 접근보다 Google Cloud 이용한 방법이 추천됨
- Google API Client 를 받아서 드라이브 페이지를 불러오면 가능
- 승인관련 문제: 매번 돌릴때마다 구글에서 이 App 이 내가 승인한 것이냐? 를 물어봄..

OAuth

- 오픈 스탠다드 인증 프로토콜
- 이는 사용자가 한 서비스(예: 웹사이트나 애플리케이션)에서 다른 서비스의 자원에 접근할 수 있도록 허용하는 안전한 인증 방식을 제공
- 사용자가 자신의 로그인 정보를 직접 제공하지 않아도 다른 애플리케이션에서 특정 정보에 접근하거나 기능을 수행할 수 있게 됨.

```

1  from googleapiclient.discovery import build
2  from google_auth_oauthlib.flow import InstalledAppFlow
3  from google.oauth2.credentials import Credentials
4  import pandas as pd
5  import re
6
7  # Define the scopes
8  SCOPES = ['https://www.googleapis.com/auth/drive.readonly']
9
10 # Define a function to clean file names
11 def clean_filename(filename):
12     # Replace or remove illegal characters here
13     # This example removes characters that are not alphanumeric, spaces, dots, or hyphens
14     cleaned = re.sub(r'^\w\s.-', '', filename)
15     return cleaned
16

```

→ Breakdown.. (GPT)

python

 Copy code

```
from googleapiclient.discovery import build
from google_auth_oauthlib.flow import InstalledAppFlow
from google.oauth2.credentials import Credentials
import pandas as pd
import re
```

- 여기서는 필요한 라이브러리를 가져옵니다. `googleapiclient.discovery`는 Google API와 상호작용하기 위해, `google_auth_oauthlib.flow`와 `google.oauth2.credentials`는 OAuth 2.0 인증을 처리하기 위해, `pandas`는 데이터를 조작하고 Excel 파일로 저장하기 위해, 그리고 `re`는 정규 표현식을 사용하여 문자열을 처리하기 위해 사용됩니다.

python

 Copy code

```
SCOPES = ['https://www.googleapis.com/auth/drive.readonly']
```

”

- 사용할 Google Drive API의 범위를 정의합니다. 여기서는 'drive.readonly'를 사용하여 파일 읽기 권한만 요청합니다.

python

 Copy code

```
def clean_filename(filename):  
    cleaned = re.sub(r'^\w\s.-]', '', filename)  
    return cleaned
```

- 파일 이름에서 알파벳, 숫자, 공백, 점, 하이픈을 제외한 모든 문자를 제거하는 함수를 정의합니다. 이는 파일 이름을 정리하여 파일 시스템에서 문제를 일으키지 않게 하기 위함입니다.

python

 Copy code

```
def main():
```

- 메인 함수를 정의합니다. 프로그램의 진입점입니다.

python

 Copy code

```
flow = InstalledAppFlow.from_client_secrets_file('/Users/mac/Documents/clien
```

- OAuth 2.0 인증 흐름을 시작합니다. 여기서는 클라이언트 비밀 파일의 위치와 사용할 스코프를 인자로 제공합니다. 이 파일에는 Google API에 접근하기 위한 클라이언트 ID와 비밀이 포함되어 있습니다.

python

 Copy code

```
creds = flow.run_local_server(port=0)
```

- 로컬 서버를 사용하여 사용자 인증을 처리하고, 인증이 완료되면 인증 정보를 `creds` 변수에 저장합니다. `port=0`은 사용 가능한 임의의 포트에서 서버를 실행하라는 의미입니다.

python

 Copy code

```
service = build('drive', 'v3', credentials=creds)
```

- 인증 정보를 사용하여 Google Drive 서비스를 초기화합니다. 여기서는 Google Drive API 버전 3을 사용합니다.

python

 Copy code

```
folder_id = '10YmL2NXWIF1vLmEcvwgEft09glWOQAH0'
```

- 검색할 특정 폴더의 ID를 지정합니다.

python

 Copy code

```
results = service.files().list(  
    pageSize=100,  
    fields="nextPageToken, files(id, name)",  
    q=f"'{folder_id}' in parents and mimeType='application/vnd.google-apps.folder'"  
).execute()
```

- 지정된 폴더 안의 폴더들을 나열하기 위한 API 호출을 실행합니다. 여기서는 최대 100개의 폴더를 나열하도록 하고, 폴더의 ID와 이름만 결과에 포함되도록 요청합니다. 또한, ``mimeType='application/vnd.google-apps.folder'``를 사용하여 폴더만 필터링합니다.

python

 Copy code

```
items = results.get('files', [])
```

- API 호출 결과에서 폴더 목록을 추출합니다. 결과가 없는 경우 빈 리스트를 반환합니다.

python

 Copy code

```
if not items:
    print('No folders found.')
else:
    print('Folders:')
```

- 폴더가 하나도 발견되지 않으면, "No folders found."를 출력합니다. 그렇지 않으면 "Folders:"

+ 신규

홈

내 드라이브

공유 드라이브

컴퓨터

공유 문서함

최근 문서함

중요 문서함

스팸

휴지통

저장용량

2TB 중 1.23TB 사용

추가 저장용량 구매

내 드라이브 >

유형

사람

수정 날짜

파일

이런 파일이 200-300개씩 있음 - 이름?

21SC007854;B;1;;FC...

21SC007792;B;1;;FC...

21SC007785;B;1;;FC...

21SC007780;B;1;;FC...

21SC008825...

21SC008825;A;1;;FF...

23S_059633;A;1;;FA0...

23S_039796;A;6;;FA...

23S_044043;A;2;;FA...

23S_052780...

Hugging Face 에서는?

- 구글 드라이브에 직접 접속하는 방법: 가능한 하나 잘 모르겠음
- 아쉬운대로 Hugging Face model 이름중 처음 검색되는 100개
이름 끌어오기

```
from huggingface_hub import HfApi, HfFolder
import pandas as pd
import re

# Define a function to clean model names or dataset names
def clean_name(name):
    # Replace or remove illegal characters here
    # This example removes characters that are not alphanumeric, spaces, dots, or hyphens
    cleaned = re.sub(r'^\w\s.-', '', name)
    return cleaned

def main():
    # Authentication with Hugging Face
    api = HfApi()
    token = HfFolder.get_token()
    if token is None:
        print("No Hugging Face token found. Please log in using `huggingface-cli login`.")
        return

    # List models (you can also list datasets with list_datasets())
    models = api.list_models()

    if not models:
        print('No models found.')
    else:
        print('Models:')
        model_data = [{'Name': clean_name(model.modelId), 'ID': model.sha} for model in models[:100]] # Limit to first 100 for demonstration
        df = pd.DataFrame(model_data)
```

7 / Mac / Desktop / huggingface_notipy / ...

```
def main():
    # Authentication with Hugging Face
    api = HfApi()
    token = HfFolder.get_token()
    if token is None:
        print("No Hugging Face token found. Please log in using `huggingface-cli login`.")
        return

    # List models (you can also list datasets with list_datasets())
    models = api.list_models()

    if not models:
        print('No models found.')
    else:
        print('Models:')
        model_data = [{'Name': clean_name(model.modelId), 'ID': model.sha} for model in models[:100]] # Limit to first 100 for demonstration
        df = pd.DataFrame(model_data)

        # Attempt to write to Excel
        try:
            df.to_excel('huggingface_models.xlsx', index=False)
            print("Model names saved to Excel successfully.")
        except Exception as e:
            print(f"Failed to write to Excel: {e}")

if __name__ == '__main__':
    main()
```

python

 Copy code

```
from huggingface_hub import HfApi, HfFolder
import pandas as pd
import re
```

- 필요한 라이브러리를 가져옵니다. `huggingface_hub`은 Hugging Face Hub와 상호작용하기 위해, `pandas`는 데이터를 조작하고 Excel 파일로 저장하기 위해, `re`는 정규 표현식을 사용하여 문자열을 처리하기 위해 사용됩니다.

python

 Copy code

```
def clean_name(name):
    cleaned = re.sub(r'^\w\s.-]', '', name)
    return cleaned
```

python

 Copy code

```
def main():
```

- 메인 함수를 정의합니다. 이 스크립트의 진입점입니다.

python

 Copy code

```
api = HfApi()
```

- `HfApi` 클래스의 인스턴스를 생성하여 Hugging Face API와의 상호작용을 준비합니다.

python

 Copy code

```
token = HfFolder.get_token()
```

- `HfFolder.get_token()` 메소드를 사용하여 로컬 환경에서 Hugging Face API 토큰을 검색합니다. 이 토큰은 사용자 인증을 위해 사용됩니다.

python

 Copy code

```
if token is None:
    print("No Hugging Face token found. Please log in using `huggingface-cli`")
    return
```

- 만약 API 토큰을 찾을 수 없으면, 사용자에게 `huggingface-cli login`을 사용하여 로그인하라는 메시지를 출력하고 스크립트를 종료합니다.

python

 Copy code

```
models = api.list_models()
```

- `list_models()` 메소드를 호출하여 Hugging Face Hub에서 사용 가능한 모델의 목록을 가져옵니다.

python

 Copy code

```
if not models:
    print('No models found.')
else:
    print('Models:')
```

python

 Copy code

```
model_data = [{ 'Name': clean_name(model.modelId), 'ID': model.sha} for model
```

- 가져온 모델 목록에서 처음 100개의 모델에 대해, 각 모델의 ID와 SHA 해시를 정리하여 `model_data` 리스트에 딕셔너리 형태로 저장합니다. 여기서 모델 이름은 `clean_name` 함수를 통해 정리됩니다.

python

 Copy code

```
df = pd.DataFrame(model_data)
```

python

 Copy code

```
try:
    df.to_excel('huggingface_models.xlsx', index=False)
    print("Model names saved to Excel successfully.")
except Exception as e:
    print(f"Failed to write to Excel: {e}")
```

- DataFrame을 Excel 파일로 저장을 시도합니다. 성공하면 "Model names saved to Excel successfully." 메시지를 출력하고, 실패하면 에러 메시지를 출력합니다.

python

 Copy code

```
if __name__ == '__main__':
    main()
```

 Office Update To keep up-to-date with security updates, fixes, and improvements, choose Check for Updates.

[illegible]

결론

- ML: Hugging Face.
- Data scientist/Researcher in various fields using big data:
Google Cloud
- 구글드라이브 공유, 데이터분석은 google cloud