

GPT + DesignSafe: Al-driven metadata extraction

Using AI for metadata extraction

- Access DesignSafe Data Depot using TAPIS API
- Read different file formats (csv / xlsx) and extract metadata:
 - Column names
 - Value ranges
 - File association
- Push the extracted metadata to a semantic search engine like Neo4j for context-specific searchers
- Example datasets considered:
 - LEAP liquefaction datasets (csv)
 - Liquefaction CPT data from Maurer (xlsx)

LEAP dataset



O Add My Data

My Projects Shared with Me

Dropbox.com Google Drive

Published Published (NEES)

Community Data

PRJ-3484 | LEAP-2022 - Effects of overburden stress, relative density, and static ♣ Download Dataset shear stress on cyclic strength of Ottawa F65 sand

Manzari, Majid Project Type Experimental

Q

Natural Hazard Type Earthquake Liquefaction Experiments and Analysis Projects | CMMI 1635524 Awards

> PRJ-1783: LEAP-2017 GWU Laboratory Tests PRJ-1780: LEAP-GWU-2015 Laboratory Tests

PRJ-2136: LEAP-Asia-2018: Stress-strain response of Ottawa sand in Cyclic Torsional Shear Tests

Stress-strain behavior and liquefaction strength characteristics of Ottawa F65 sand

Stress-strain behavior of Ottawa sand in cyclic direct simple shear and modeling of cyclic strength using Artificial

A

14

Copy Preview Preview Images Download

a

Confining Stress, Cyclic Direct Simple Shear Test, Liquefaction, Relative Density, Sand, Static Shear Stress, Stress-Strain Response

10.17603/ds2-q3fz-kq91 10.17603/ds2-z7zb-bq89

10.17603/ds2-52dm-s780 View Data Diagram | 🗠 View Project Metrics | 🗎 Leave Feedback

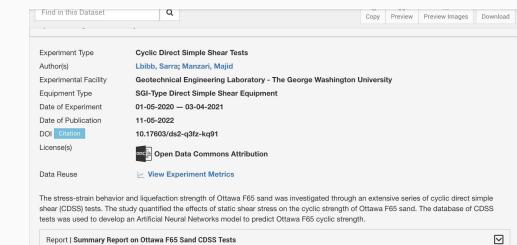
Description | The stress-strain-strength behavior of Ottawa F-65 sand is investigated through a series of stress-controlled Cyclic Direct Simple Shear (CDSS) tests conducted as part of the Liquefaction Experiments and Analysis Project (LEAP-2022). The tests were performed with two different overburden stresses on specimens prepared with different relative densities and subjected to different static (initial) shear stresses prior to cyclic shearing. The report included here presents the setup and experimental procedure, as well as the liquefaction strength curves and the results obtaine... **Show More**

PRJ-3484

Related Work

Keywords

DOIs in Project



Model Configuration | Direct Simple Shear Experimental Setup

Sensor Information | Direct Simple Shear Test Sensors

Name

Event | CDSS tests with Sig v = 100 kPa, Dr = 67%, Alpha = 0.3

P eo 0 601 sigv 100 CSR 0 150 Tau 30 .csv eo_0_601_sigv_100_CSR_0_200_Tau_30_.csv eo_0_601_sigv_100_CSR_0_250_Tau_30_.csv P eo 0 601 sigv 100 CSR 0 300 Tau 30 .csv $\overline{\mathbf{v}}$

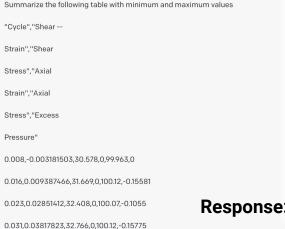
abla

 \triangle

Size

Trial 0: GPT-3 (DaVinci 003)

Prompt:



GPT-3 does a good job of extracting min and max values from the dataset presented. The summary also shows an accurate description of the behavior observed in the "limited" dataset shared with GPT-3

Response:

This table shows the relationship between shear strain, shear stress, axial strain, axial stress, and excess pressure for a cycle. It shows that as the shear strain increases, the shear stress, axial stress, and excess pressure increase, while the axial strain decreases. The minimum values for shear strain, shear stress, axial strain, axial stress, and excess pressure are 0.008, -0.003181503, 0, 99.963, and 0 respectively. The maximum values for shear strain, shear stress, axial strain, axial stress, and excess pressure are 0.133, 38.88, -0.0018094, 100.59, and -0.42067 respectively.

Trial 1: GPT-3 (DaVinci 003) - Computation

"Shear Strain": {

"mean": 0.077.

"Shear Stress": {
 "mean": 35.072.

"min": 30.578,

"max": 38.88

"Axial Strain": {

"min": 31.669, "max": 36.415

'Axial Stress": {
"mean": -0.15581,
"min": 30.578,
"max": 37.7

"mean": -0.001787

"min": -0.003181503.

"max": 0.2036224

Prompt:

```
Summarize the following table with summary statistics as a JSON file

"Cycle", "Shear --

Strain", "Shear

Stress", "Axial
```

Strain","Axial

Stress","Excess

Pressure'

0.008,-0.003181503,30.578,0,99.963,0 0.016,0.009387466,31.669,0,100.12,-0.15581

0.023,0.02851412,32.408,0,100.07,-0.1055

0.031,0.03817823,32.766,0.100.12,-0.15775

Response:

- GPT-3 got the column name wrong! (missing "Cycle" and "Excess Pressures" - due to badly formatted CSV file)
- GPT-3 does a good job of extracting min and max values correctly
- GPT-3 mean calculation was wrong!
 - Axial stress column is inconsistent

Mean values	GPT	Actual	Rel error (%)
Cycles	0.077	0.07035	9.45
Shear stress	35.072	35.708	1.78
Axial strain	-0.0018	-0.0005	237.17
Axial stress	-0.1558	100.303	100.16

Trial 2: ChatGPT - Generating code

Since GPT-3 is not so good at computations, why not generate code instead of the output directly. ChatGPT identified file as CSV and generated a code rather than the actual output. (we can also force it to generate a code using prompt engineering)

Prompt:

Generate a summary statistics for the following table as a JSON file: "Cvcle", "Shear Strain", "Shear Stress", "Axial Strain", "Axial Stress", "Excess Pressure" 0.008 -0.003181503 30.578 0 99.963 0 0.016 0.009387466 31.669 0 100.12 -0.15581 0.023 0.02851412 32.408 0 100.07 -0.1055 0.031 0.03817823 32.766 0 100.12 -0.15775 0.039 0.05090463 33.442 -0.0018094 100.23 -0.262920.047 0.06355149 34.476 -1.12E-05 100.23 -0.262920.055 0.07934018 35.16 0 100.33 -0.36678 0.063 0.0986296 35.576 0 100.33 -0.36809 0.1206195 36.604 -0.001787 100.38 -0.420030.078 0.1303195 36.415 -2.26E-05 100.33 -0.36875 0.086 0.1462469 37.372 0 100.49 -0.52485 0.094 0.1654432 37.7 0 100.44 -0.47326 0.102 0.1877144 38.537 0 100.59 -0.63101 0.109 0.1877144 38.322 -0.0018094 100.38 -0.42067 0.1971913 38.631 -1.12E-05 100.44 -0.472930.125 0.1972592 38.505 -0.0017981 100.33 -0.36875 0.133 0.2036224 38.88 -0.0018094 100.38 -0.4206

Response:

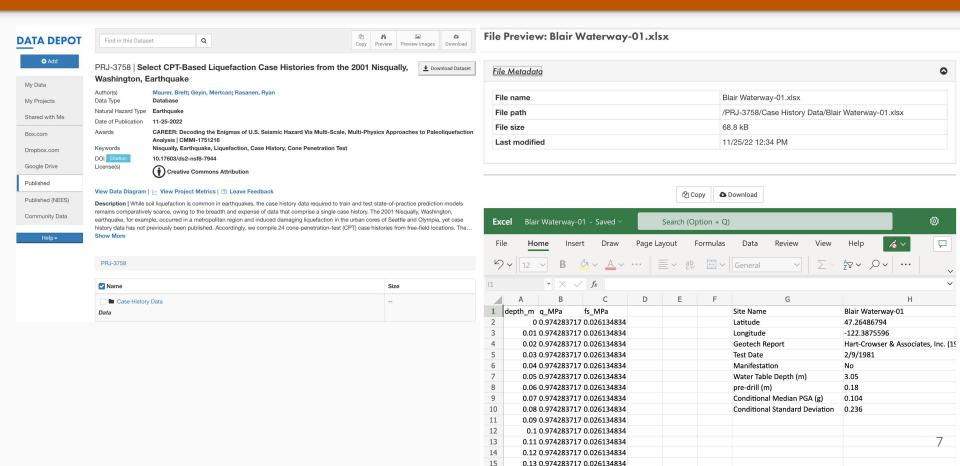
Here's a Python script that generates a JSON file containing summary statistics for the table you provided:



Note that in this script you need to have your table in a .csv file formate with the name table.csv and run the script on your machine.

It will generate a file summary_stats.json in your local directory which will have all the summary statistics like count, mean, std, min, max and quartiles for each column.

Extracting data from XLSX file



Trial 0: Extract metadata directly from text

Extract metadata from this file sample:

 depth_m
 q_MPa
 fs_MPa

 0 0.974283717
 0.026134834

 0.01
 0.974283717
 0.026134834

 0.02
 0.974283717
 0.026134834

 Associates, Inc. (1981)
 0.026134834

 0.03
 0.974283717
 0.026134834

 0.04
 0.974283717
 0.026134834

 0.05
 0.974283717
 0.026134834

 0.07
 0.974283717
 0.026134834

 0.08
 0.974283717
 0.026134834

 0.09
 0.974283717
 0.026134834

 0.1
 0.974283717
 0.026134834

 0.1
 0.974283717
 0.026134834

 0.1
 0.974283717
 0.026134834

 0.1
 0.974283717
 0.026134834

 0.1
 0.974283717
 0.026134834

 0.1
 0.974283717
 0.026134834

 0.1
 0.974283717
 0.026134834

Site Name Blair Waterway-01
Latitude 47.26486794
Longitude -122.3875596
Geotech Report Hart-Crowser &

Test Date 2/9/1981

Manifestation No

Water Table Depth (m) 3.05

pre-drill (m) 0.18

Conditional Median PGA (g) 0.104

Conditional Standard Deviation 0.236



The file appears to be a tab-separated data file that contains various metadata about a site. The metadata includes:

* Site Name: Blair Waterway-01

· Latitude: 47.26486794

· Longitude: -122.3875596

* Geotech Report: Hart-Crowser & Associates, Inc. (1981)

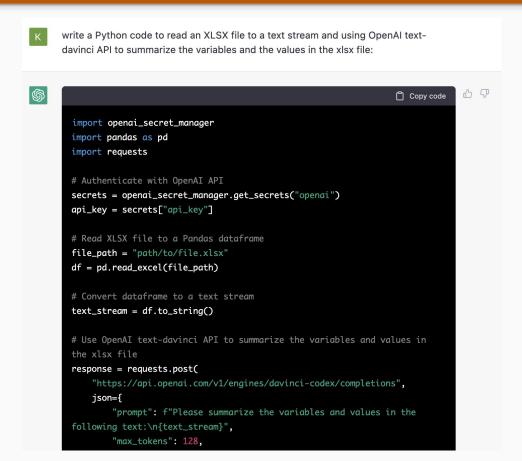
* Test Date: 2/9/1981

Manifestation: No

* Water Table Depth (m): 3.05

```
convert metadata to a JSON file
Here is an example of how the metadata from the file could be converted to a JSON
file:
                                                                     Copy code
      "Site Name": "Blair Waterway-01",
      "Latitude": 47.26486794,
      "Longitude": -122.3875596,
      "Geotech Report": "Hart-Crowser & Associates, Inc. (1981)",
      "Test Date": "2/9/1981",
      "Manifestation": "No",
      "Water Table Depth (m)": 3.05,
      "pre-drill (m)": 0.18.
      "Conditional Median PGA (g)": 0.104,
      "Conditional Standard Deviation": 0.236
It's important to notice that JSON does not accept tab as separator and the data in
the file may need to be preprocessed accordingly and also, JSON file format use
double guotes instead of single guotes.
```

Trial 1: ChatGPT to write code



Ask ChatGPT to write a code using OpenAl API to summarize XLSX file.

- ChatGPT only knows about Text-davinci-002 model as it was trained with data before 2021.
- It uses the package
 `openai_secret_manager`
 that didn't work.

Trial 2: human intervention in the code

```
chatgpt.py 2 X
                                                                                    human.py 3
 chatgpt.py > ...
                                                                                     human.py > ...
       import openai secret manager
                                                                                           # import openai secret manager
       import pandas as pd
                                                                                           import pandas as pd
       import requests
                                                                                           import openai
       # Authenticate with OpenAI API
       secrets = openai_secret_manager.get_secrets("openai")
                                                                                           # Authenticate with OpenAI API
       api_key = secrets["api_key"]
                                                                                           openai.api_key = "YOUR_API_KEY"
       # Read XLSX file to a Pandas dataframe
                                                                                           # read the xlsx file
       file_path = "path/to/file.xlsx"
                                                                                           file_path = "Blair Waterway-01.xlsx"
                                                                                           df = pd.read_excel(file_path)
       df = pd.read_excel(file_path)
       # Convert dataframe to a text stream
                                                                                           # Convert dataframe to a text stream
       text stream = df.to string()
                                                                                           text stream = df.to string()
       # Use OpenAI text-davinci API to summarize the variables and values in the
                                                                                           # pass the dataframe to GPT-3 to generate summary
       response = requests.post(
                                                                                           prompt = f"Please summarize the variables and the values in the \
           "https://api.openai.com/v1/engines/davinci-codex/completions",
                                                                                                      following xlsx file: {text_stream}"
                                                                                           response = openai.Completion.create(
           ison={
                                                                                               engine="text-davinci-003",
               "prompt": f"Please summarize the variables and values in the \
                           following text:\n{text_stream}",
                                                                                               prompt=prompt,
               "max tokens": 128,
                                                                                               max tokens=2048,
               "stop": ",",
                                                                                               n=1,
                                                        Close (#W)
                                                                                               stop=None,
           headers={"Authorization": f"Bearer {api_key}"},
                                                                                               temperature=0.5,
                                                                                           #print the summary
       # Print the summary
       print(response.json()["choices"][0]["text"])
                                                                                           print(response["choices"][0]["text"])
 30
```

Trial 2: Human + Al code -> Output

```
(xai) krishna@kks32 metadata-extraction % python3 metadata.py
The xlsx file contains the following variables and values:
depth_m: 0.00-0.22
g MPa: 0.974284-0.911027
fs_MPa: 0.026135-0.026977
Unnamed: 3: No values
Unnamed: 4: No values
Unnamed: 5: No values
Site Name: Blair Waterway-01
Latitude: 47.264868
Longitude: -122.38756
Geotech Report: Hart-Crowser & Associates, Inc. (1981)
Test Date: 1981-02-09 00:00:00
Manifestation: No
Water Table Depth (m): 3.05
pre-drill (m): 0.18
Conditional Median PGA (g): 0.104
Conditional Standard Deviation: 0.236
```

Trial 2b: Extend GPT to extract it as a JSON

K

Extract variables and values from the following text output as a JSON file:

```
depth_m: 0.00 - 0.22 (increments of 0.01)
```

q_MPa: 0.974284 - 0.911027 (increments of 0.05 - 0.06) fs_MPa: 0.026135 - 0.026977 (increments of 0.001 - 0.002)

Unnamed: 3: Empty Unnamed: 4: Empty Unnamed: 5: Empty

Site Name: Blair Waterway-01

Latitude: 47.264868 Longitude: -122.38756

Geotech Report: Hart-Crowser & Associates, Inc. (1981)

Test Date: 1981-02-09 00:00:00

Manifestation: No

Water Table Depth (m): 3.05

pre-drill (m): 0.18

Conditional Median PGA (g): 0.104 Conditional Standard Deviation: 0.236

Generate output as JSON

```
"depth m": {
        "min": 0,
        "max": 0.22,
        "increment": 0.01
      "a MPa": {
        "min": 0.974284.
        "max": 0.911027,
10 -
        "increment": {
11
          "min": 0.05.
12
          "max": 0.06
13
14
15 -
      "fs_MPa": {
16
        "min": 0.026135,
        "max": 0.026977,
17
18 -
        "increment": {
19
          "min": 0.001,
20
           "max": 0.002
21
22
      "Unnamed: 3": "Empty",
      "Unnamed: 4": "Empty",
      "Unnamed: 5": "Empty".
      "Site Name": "Blair Waterway-01",
      "Latitude": 47.264868,
      "Longitude": -122.38756,
      "Geotech Report": "Hart-Crowser & Associates, Inc. (1981)",
      "Test Date": "1981-02-09 00:00:00".
31
      "Manifestation": "No",
      "Water Table Depth (m)": 3.05,
      "pre-drill (m)": 0.18,
      "Conditional Median PGA (a)": 0.104.
34
      "Conditional Standard Deviation": 0.236
36
```

12

Trial 2c: Can we write a code to convert to JSON?

Write a generic python code to extract variables and values and output as a JSON file using the following text stream only as an example: The xlsx file contains the following variables and values: depth_m: 0.00-0.22 q_MPa: 0.974284-0.911027 fs MPa: 0.026135-0.026977 Unnamed: 3: No values Unnamed: 4: No values Unnamed: 5: No values Site Name: Blair Waterway-01 Latitude: 47.264868 Longitude: -122.38756 Geotech Report: Hart-Crowser & Associates, Inc. (1981) Test Date: 1981-02-09 00:00:00

Manifestation: No

Water Table Depth (m): 3.05

pre-drill (m): 0.18

Conditional Median PGA (g): 0.104 Conditional Standard Deviation: 0.236

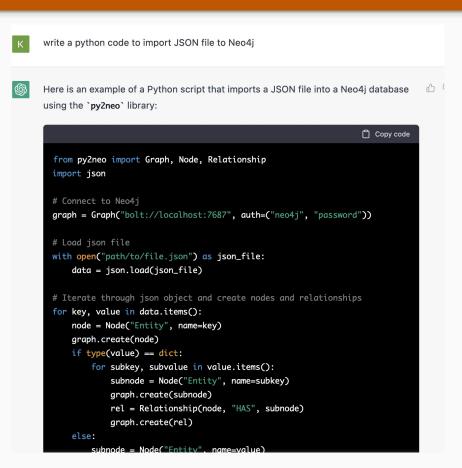
Trial 2c: Text to JSON output

```
chatgpt.py > ...
      import ison
      data = {}
      text = """The xlsx file contains the following variables and values: \n depth_m: 0.00-0.
      q_MPa: 0.974284-0.911027 \n fs_MPa: 0.026135-0.026977 \n Unnamed: 3: No values \n Unr
      Site Name: Blair Waterway-01 \n Latitude: 47.264868 \n Longitude: -122.38756
      Geotech Report: Hart-Crowser & Associates, Inc. (1981) \n Test Date: 1981-02-09 00:00:0
      Manifestation: No \n Water Table Depth (m): 3.05 \n pre-drill (m): 0.18
      Conditional Median PGA (g): 0.104 \n conditional Standard Deviation: 0.236"""
10
      lines = text.split("\n")
      for line in lines:
          parts = line.split(":")
          variable = parts[0].strip()
          value = parts[1].strip()
          data[variable] = value
      with open("data.json", "w") as f:
20
          ison.dump(data, f)
```

The code above will take the text stream as input and extract the variables and values from it. It then stores them in a dictionary, and finally writes that dictionary to a JSON file named "data.json". The resulting JSON file will have the following format:

```
Copy code
"depth_m": "0.00-0.22",
"q_MPa": "0.974284-0.911027",
"fs_MPa": "0.026135-0.026977",
"Unnamed: 3": "No values",
"Unnamed: 4": "No values".
"Unnamed: 5": "No values".
"Site Name": "Blair Waterway-01",
"Latitude": "47.264868",
"Longitude": "-122.38756",
"Geotech Report": "Hart-Crowser & Associates, Inc. (1981)",
"Test Date": "1981-02-09 00:00:00",
"Manifestation": "No",
"Water Table Depth (m)": "3.05",
"pre-drill (m)": "0.18",
"Conditional Median PGA (g)": "0.104",
"Conditional Standard Deviation": "0.236"
```

Building a semantic search engine from JSON



```
else:
    subnode = Node("Entity", name=value)
    graph.create(subnode)
    rel = Relationship(node, "HAS", subnode)
    graph.create(rel)

print("Data imported successfully!")
```

The script uses the `Graph` class from `py2neo` to connect to a Neo4j database running on the localhost at port 7687. It loads the JSON file using the `json` package, then it iterates through the objects in the JSON file, creating `Node` and `Relationship` objects for each key-value pair. It creates the nodes and relationships in the Neo4j database using the `create` method.

It's important to notice that this script assumes that you have Neo4j installed locally and running on the default port 7687 and also, you will need to have `py2neo` package installed in your environment, if not, you can install it with the following command:



Also, you may need to adjust the credentials and the connection url to match your Neo4j setup.

Challenges: GPT is a probabilistic language model

Outputs will be different for the same input!

Output 1:

```
(xai) krishna@kks32 metadata-extraction % python3 metadata.py
```

The xlsx file contains six columns and 23 rows. The first three columns are 'depth_m', 'q_MPa', and 'fs_MPa', which contain values of depth in meters, q in MPa, and fs in MPa, respectively. The remaining columns are unnamed and contain values of site name, latitude, longitude, geotech report, to st date, manifestation, water table depth in meters, pre-drill in meters, conditional median PGA in g, and conditional standard deviation, respectively.

Output 2:

```
(xai) krishna@kks32 metadata-extraction % python3 metadata.py
The xlsx file contains the following variables and values:
depth m: 0.00-0.22
q_MPa: 0.974284-0.911027
fs MPa: 0.026135-0.026977
Unnamed: 3: No values
Unnamed: 4: No values
Unnamed: 5: No values
Site Name: Blair Waterway-01
Latitude: 47.264868
Longitude: -122.38756
Geotech Report: Hart-Crowser & Associates, Inc. (1981)
Test Date: 1981-02-09 00:00:00
Manifestation: No
Water Table Depth (m): 3.05
pre-drill (m): 0.18
Conditional Median PGA (g): 0.104
Conditional Standard Deviation: 0.236
```