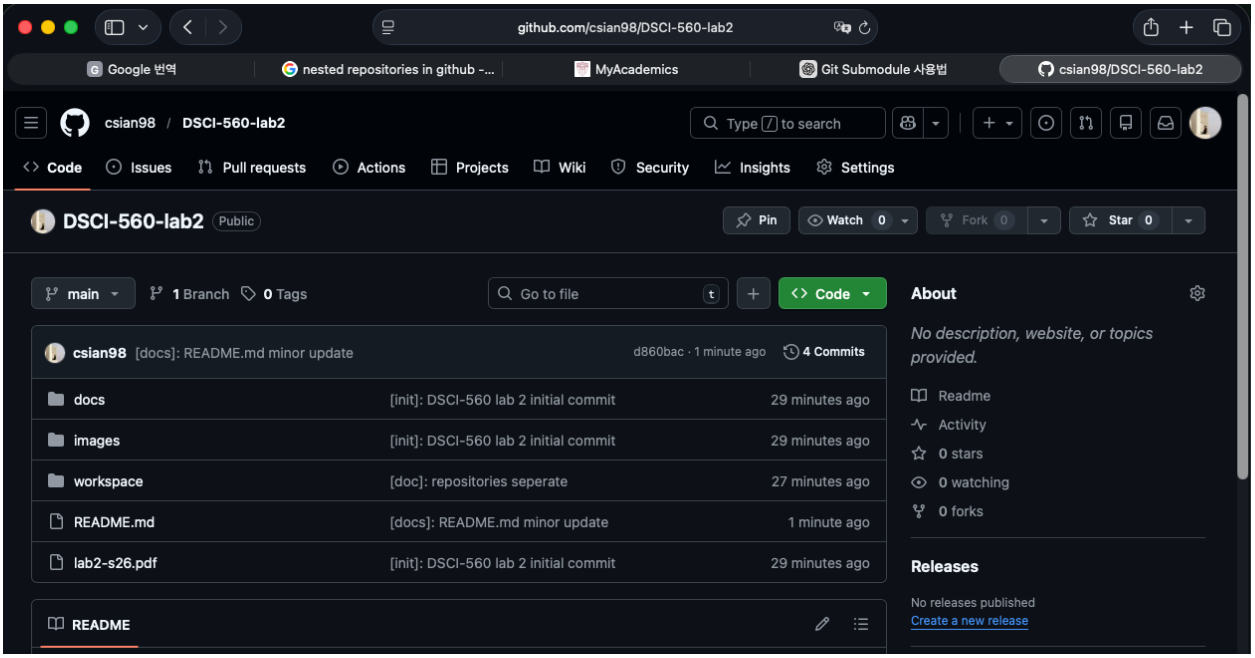
DSCI 560 USCID: 5023184813
Data Science Professional Practicum Jeong Hoon Choi

Laboratory Assignment 2



csian98 [docs]: README.md minor update d860bac · 1 minute ago 4 Commits

docs	[init]: DSCI-560 lab 2 initial commit	29 minutes ago
images	[init]: DSCI-560 lab 2 initial commit	29 minutes ago
workspace	[doc]: repositories seperate	27 minutes ago
README.md	[docs]: README.md minor update	1 minute ago
lab2-s26.pdf	[init]: DSCI-560 lab 2 initial commit	29 minutes ago

About

No description, website, or topics provided.

Readme
Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release

https://github.com/csian98/DSCI-560-lab2

1 Team Formation

Jeong Hoon Choi (USCID: 5023184813)
Viraj Bansal (USCID: 7366023502)
Chih-Yun Pai (USCID: 6777867067)

2 Real-World Applications and Domain-Specific Data

[Domain 1] System Log Normalization and Alert System (Jeong Hoon Choi)

[Domain Description] This system parses log(text) data stored in various formats using traditional parsing methods, Encoder-Decoder models, or LLM models.
It then performs anomaly detection (i.g. Autoencoder, Isolation Forests) and, based on the results, uses an LLM to generate an alert containing relevant information such as IP, user, error details, justification for the detection, and a risk score, which is then deliver to the appropriate user.

[Dataset] LogHub

loghub: https://github.com/logpai/loghub
Loghub provides logs for various services, along with labels that have been appropriately parsed using a Log Parser (drain).

[Reason]

Many commercial software solutions are available that collect and respond to various logs from systems such as SIEM, IDS/IPS, and EDR. Because these systems use different log formats depends on the services, normalization of these logs is crucial. These logs can then be processed using LLMs or encoder-decoder models, and an appropriate response/alert (message/chat) system can be applied based on the processed data.

[Domain 2] Recipe (Chih-Yun Pai)
[Domain Description] A recipe-based food assistant that supports group chat communication by helping users decide what meals they can prepare based on available ingredients. Through messages, the AI agent can suggest suitable recipes, identify missing ingredients, and later assist with planning grocery purchases (if we are able to do that).

[Dataset]
https://www.kaggle.com/datasets/paultimothymooney/recipenlg/data

[Reason]
It's a use case where users can seek quick, practical assistance through messaging. A scenario I come up with is when we text "I want to have a quick dinner cuz I'm working heavily for the final project", based on previous message and the recipe database, AI agent may suggest a easy-to-prepare meal like spaghetti.

[Domain 3] LLM Stock Bot (Viraj Bansal)
[Domain Description] A message bot that can retrieve stock prices and some sort of news/forum posting about the stock and then allow users to discuss and talk about different stocks they are interested of investing. users can use messages and chatbot to propose and debate on different stocks while using the llm bot to fact check, learn, run analysis, buy/sell stocks, and generate reminders about their stocks' positions (individually or as groups).

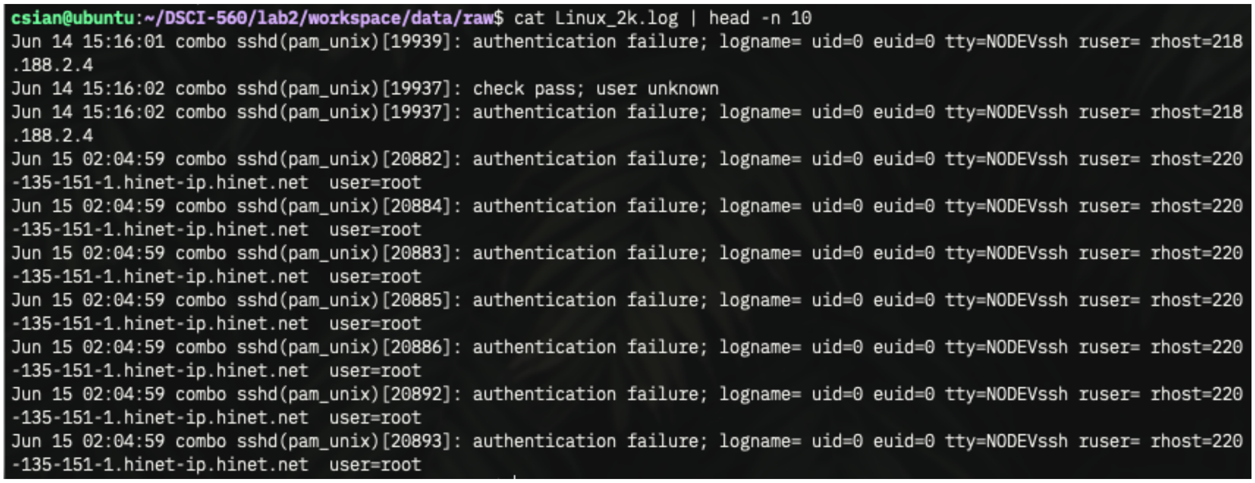
[Dataset]
yahoo finance (https://pypi.org/project/yfinance/) and stock news api (https://stocknewsapi.com/)

[Reason]
This can allow users to discuss, debate, and trade stocks with access to all real time data and news feed at there finger tips. they can also watch and praise/make of fun of each others portfolios position as they go up/down.

3 Examples of Tools for Data Collection

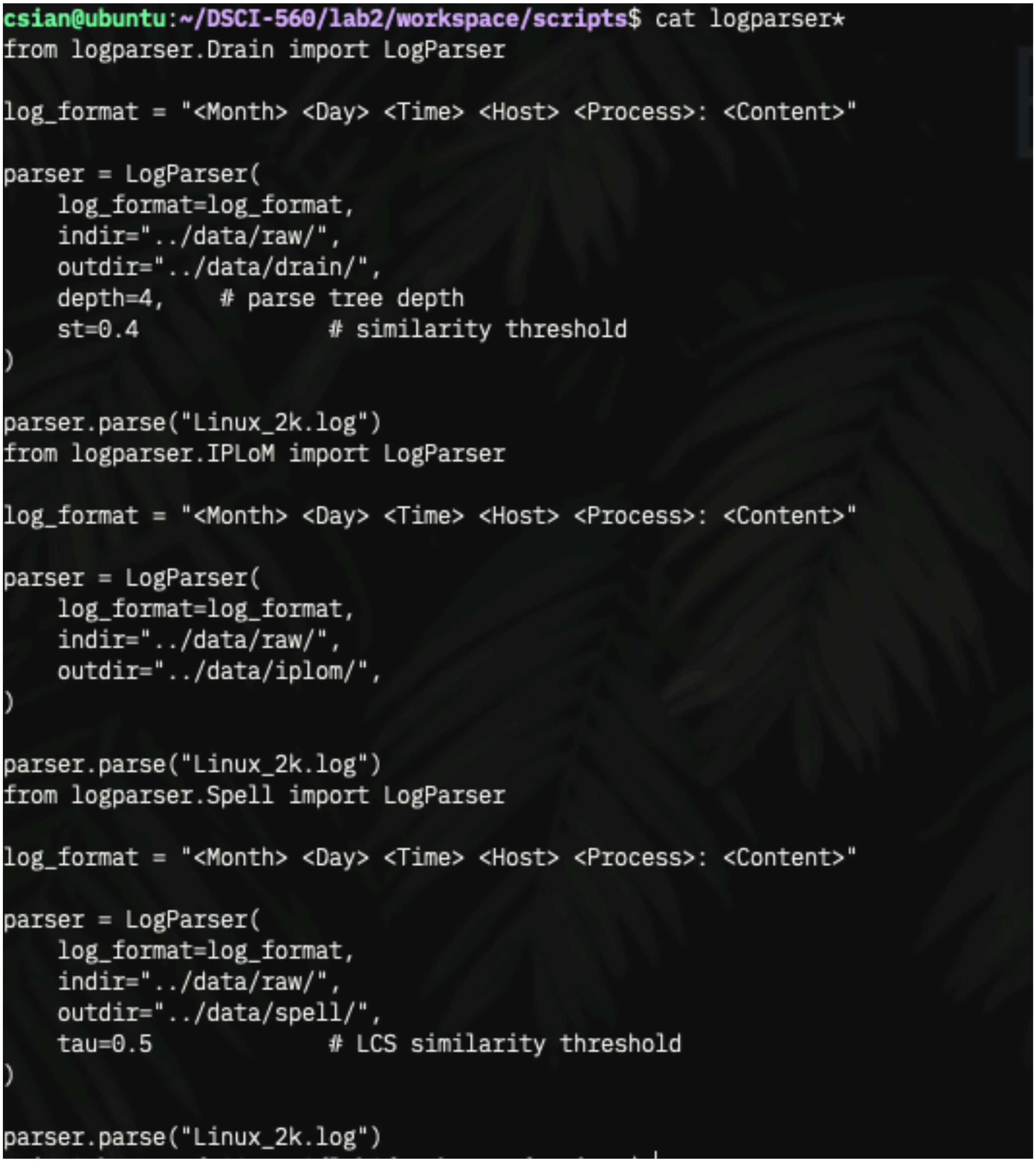
Log Parser:
logparser: https://github.com/logpai/logparser

Data Retrieve:
requests: https://pypi.org/project/requests/
beautifulsoup4: https://pypi.org/project/beautifulsoup4/
playwright: https://pypi.org/project/playwright/
pandas: https://pypi.org/project/pandas/
pytesseract: https://pypi.org/project/pytesseract/
pillow: https://pypi.org/project/pillow/
pdfplumber: https://pypi.org/project/pdfplumber/
python-docx: https://pypi.org/project/python-docx/
yfinance: https://pypi.org/project/yfinance/



```
csian@ubuntu:~/DSCI-560/lab2/workspace/data/raw$ cat Linux_2k.log | head -n 10
Jun 14 15:16:01 combo sshd(pam_unix)[19939]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=218
.188.2.4
Jun 14 15:16:02 combo sshd(pam_unix)[19937]: check pass; user unknown
Jun 14 15:16:02 combo sshd(pam_unix)[19937]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=218
.188.2.4
Jun 15 02:04:59 combo sshd(pam_unix)[20882]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=220
-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20884]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=220
-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20883]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=220
-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20885]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=220
-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20886]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=220
-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20892]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=220
-135-151-1.hinet-ip.hinet.net  user=root
Jun 15 02:04:59 combo sshd(pam_unix)[20893]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=220
-135-151-1.hinet-ip.hinet.net  user=root
```

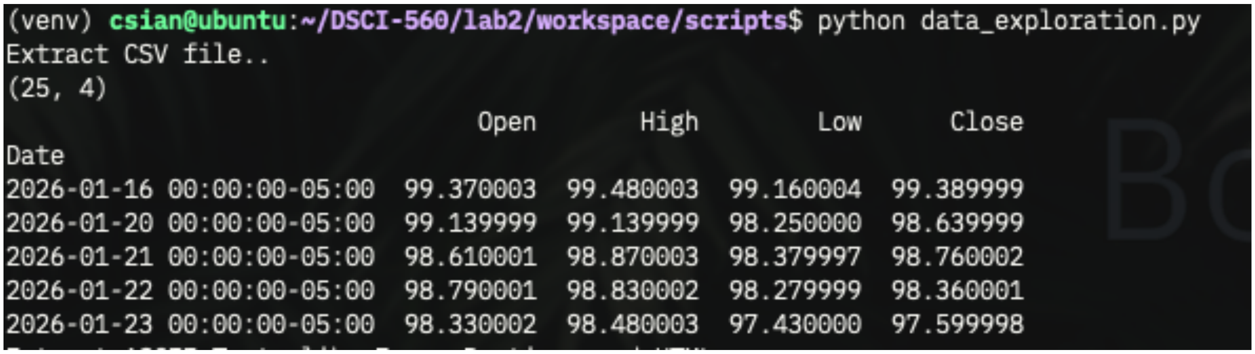
As an example of the data to be used in the project, I downloaded and used Linux syslog data from Loghub.



```
csian@ubuntu:~/DSCI-560/lab2/workspace/scripts$ cat logparser*
from logparser.Drain import LogParser

log_format = "<Month> <Day> <Time> <Host> <Process>: <Content>"

parser = LogParser(
    log_format=log_format,
    indir="../data/raw/",
    outdir="../data/drain/",
    depth=4,    # parse tree depth
    st=0.4              # similarity threshold
)

parser.parse("Linux_2k.log")
from logparser.IPLoM import LogParser

log_format = "<Month> <Day> <Time> <Host> <Process>: <Content>"

parser = LogParser(
    log_format=log_format,
    indir="../data/raw/",
    outdir="../data/iplom/",
)

parser.parse("Linux_2k.log")
from logparser.Spell import LogParser

log_format = "<Month> <Day> <Time> <Host> <Process>: <Content>"

parser = LogParser(
    log_format=log_format,
    indir="../data/raw/",
    outdir="../data/spell/",
    tau=0.5             # LCS similarity threshold
)

parser.parse("Linux_2k.log")
```

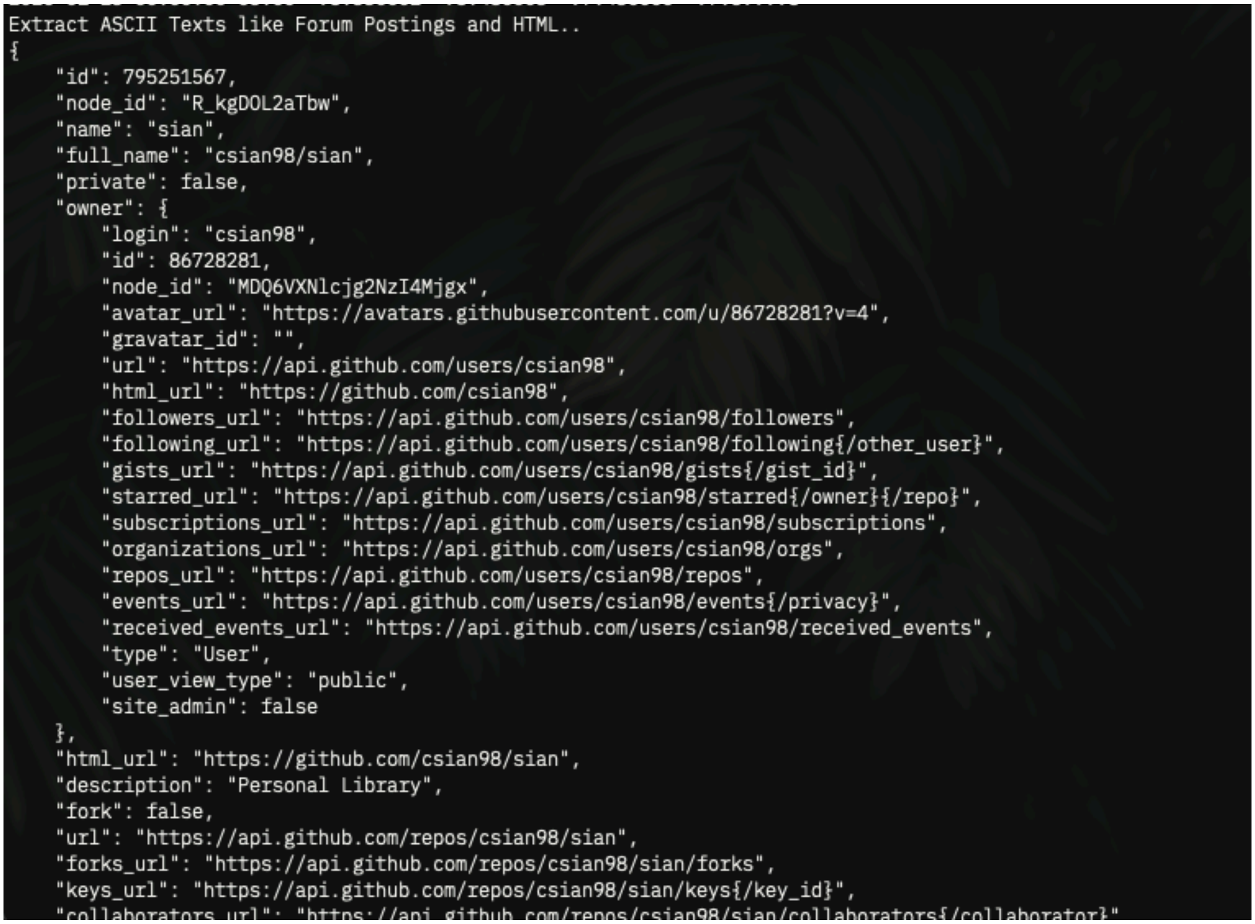
Using logparser3, I parsed the logs using the Drain, IPLoM, and Spell methods, and I was able to easily parse the syslog, which maintains a relatively consistent format.

4 Data Collection



```
(venv) csian@ubuntu:~/DSCI-560/lab2/workspace/scripts$ python data_exploration.py
Extract CSV file..
(25, 4)
                                Open       High        Low      Close
Date
2026-01-16 00:00:00-05:00  99.370003  99.480003  99.160004  99.389999
2026-01-20 00:00:00-05:00  99.139999  99.139999  98.250000  98.639999
2026-01-21 00:00:00-05:00  98.610001  98.870003  98.379997  98.760002
2026-01-22 00:00:00-05:00  98.790001  98.830002  98.279999  98.360001
2026-01-23 00:00:00-05:00  98.330002  98.480003  97.430000  97.599998
```

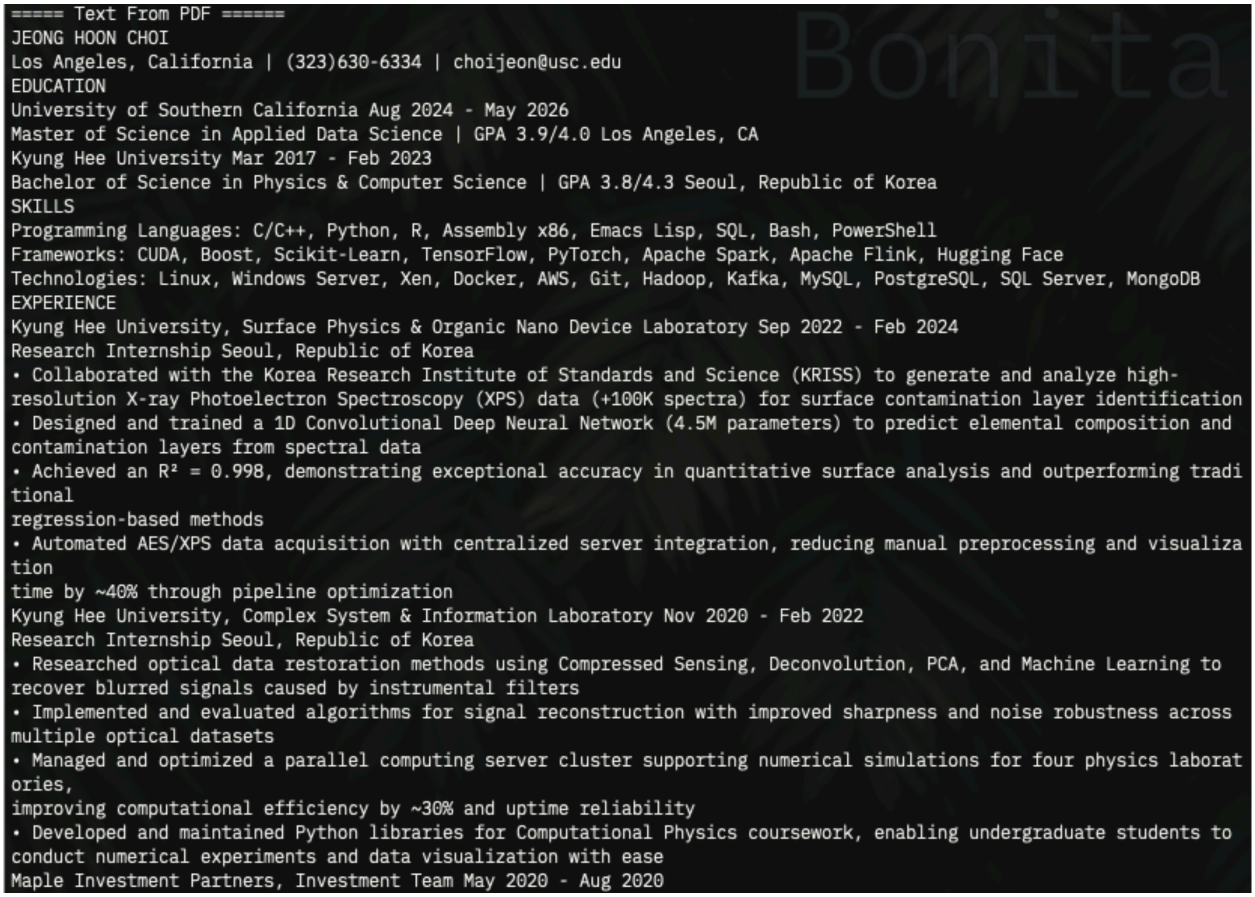
Using the yfinance Python library provided by Yahoo Finance, I collected the dollar index (DX-Y.NYB) data for the past 30 days and saved and loaded it as a CSV file.



```
Extract ASCII Texts like Forum Postings and HTML..
{
    "id": 795251567,
    "node_id": "R_kgDOL2aTbw",
    "name": "sian",
    "full_name": "csian98/sian",
    "private": false,
    "owner": {
        "login": "csian98",
        "id": 86728281,
        "node_id": "MDQ6VXNlcjg2NzI4Mjgx",
        "avatar_url": "https://avatars.githubusercontent.com/u/86728281?v=4",
        "gravatar_id": "",
        "url": "https://api.github.com/users/csian98",
        "html_url": "https://github.com/csian98",
        "followers_url": "https://api.github.com/users/csian98/followers",
        "following_url": "https://api.github.com/users/csian98/following{/other_user}",
        "gists_url": "https://api.github.com/users/csian98/gists{/gist_id}",
        "starred_url": "https://api.github.com/users/csian98/starred{/owner}{/repo}",
        "subscriptions_url": "https://api.github.com/users/csian98/subscriptions",
        "organizations_url": "https://api.github.com/users/csian98/orgs",
        "repos_url": "https://api.github.com/users/csian98/repos",
        "events_url": "https://api.github.com/users/csian98/events{/privacy}",
        "received_events_url": "https://api.github.com/users/csian98/received_events",
        "type": "User",
        "user_view_type": "public",
        "site_admin": false
    },
    "html_url": "https://github.com/csian98/sian",
    "description": "Personal Library",
    "fork": false,
    "url": "https://api.github.com/repos/csian98/sian",
    "forks_url": "https://api.github.com/repos/csian98/sian/forks",
    "keys_url": "https://api.github.com/repos/csian98/sian/keys{/key_id}",
    "collaborators_url": "https://api.github.com/repos/csian98/sian/collaborators{/collaborator}",
```

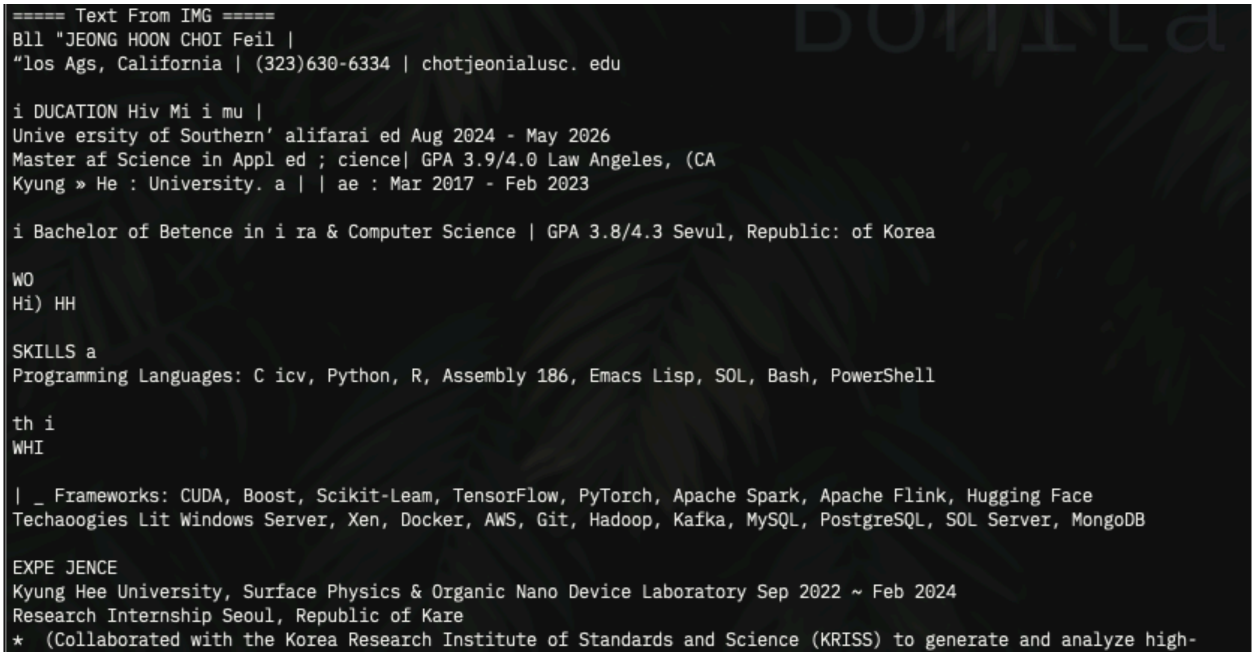
Additionally, I used the API provided by GitHub to scrape data about users and repositories, converted it to JSON format, and saved it.

Finally, I extracted text from three images, a Microsoft Word document, and a PDF file. I used the same resume for all three.



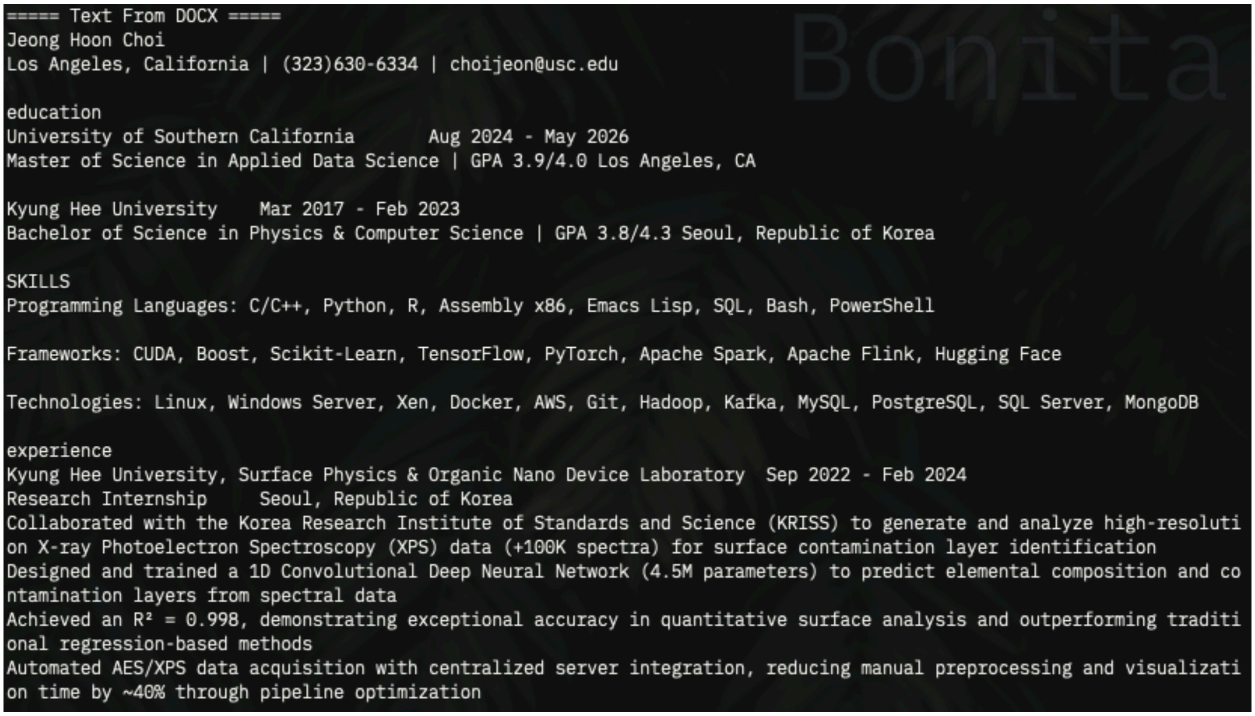
```
===== Text From PDF ======
JEONG HOON CHOI
Los Angeles, California | (323)630-6334 | choijeon@usc.edu
EDUCATION
University of Southern California Aug 2024 - May 2026
Master of Science in Applied Data Science | GPA 3.9/4.0 Los Angeles, CA
Kyung Hee University Mar 2017 - Feb 2023
Bachelor of Science in Physics & Computer Science | GPA 3.8/4.3 Seoul, Republic of Korea
SKILLS
Programming Languages: C/C++, Python, R, Assembly x86, Emacs Lisp, SQL, Bash, PowerShell
Frameworks: CUDA, Boost, Scikit-Learn, TensorFlow, PyTorch, Apache Spark, Apache Flink, Hugging Face
Technologies: Linux, Windows Server, Xen, Docker, AWS, Git, Hadoop, Kafka, MySQL, PostgreSQL, SQL Server, MongoDB
EXPERIENCE
Kyung Hee University, Surface Physics & Organic Nano Device Laboratory Sep 2022 - Feb 2024
Research Internship Seoul, Republic of Korea
• Collaborated with the Korea Research Institute of Standards and Science (KRISS) to generate and analyze high-
resolution X-ray Photoelectron Spectroscopy (XPS) data (+100K spectra) for surface contamination layer identification
• Designed and trained a 1D Convolutional Deep Neural Network (4.5M parameters) to predict elemental composition and
contamination layers from spectral data
• Achieved an R² = 0.998, demonstrating exceptional accuracy in quantitative surface analysis and outperforming tradi
tional
regression-based methods
• Automated AES/XPS data acquisition with centralized server integration, reducing manual preprocessing and visualiza
tion
time by ~40% through pipeline optimization
Kyung Hee University, Complex System & Information Laboratory Nov 2020 - Feb 2022
Research Internship Seoul, Republic of Korea
• Researched optical data restoration methods using Compressed Sensing, Deconvolution, PCA, and Machine Learning to
recover blurred signals caused by instrumental filters
• Implemented and evaluated algorithms for signal reconstruction with improved sharpness and noise robustness across
multiple optical datasets
• Managed and optimized a parallel computing server cluster supporting numerical simulations for four physics laborat
ories,
improving computational efficiency by ~30% and uptime reliability
• Developed and maintained Python libraries for Computational Physics coursework, enabling undergraduate students to
conduct numerical experiments and data visualization with ease
Maple Investment Partners, Investment Team May 2020 - Aug 2020
```

I used pdfplumber to extract text from the PDF file.



```
===== Text From IMG =====
Bll "JEONG HOON CHOI Feil |
“los Ags, California | (323)630-6334 | chotjeonialusc. edu

i DUCATION Hiv Mi i mu |
Unive ersity of Southern’ alifarai ed Aug 2024 - May 2026
Master af Science in Appl ed ; cience| GPA 3.9/4.0 Law Angeles, (CA
Kyung » He : University. a | | ae : Mar 2017 - Feb 2023

i Bachelor of Betence in i ra & Computer Science | GPA 3.8/4.3 Sevul, Republic: of Korea

WO
Hi) HH

SKILLS a
Programming Languages: C icv, Python, R, Assembly 186, Emacs Lisp, SOL, Bash, PowerShell

th i
WHI

| _ Frameworks: CUDA, Boost, Scikit-Leam, TensorFlow, PyTorch, Apache Spark, Apache Flink, Hugging Face
Techaoogies Lit Windows Server, Xen, Docker, AWS, Git, Hadoop, Kafka, MySQL, PostgreSQL, SOL Server, MongoDB

EXPE JENCE
Kyung Hee University, Surface Physics & Organic Nano Device Laboratory Sep 2022 ~ Feb 2024
Research Internship Seoul, Republic of Kare
*  (Collaborated with the Korea Research Institute of Standards and Science (KRISS) to generate and analyze high-
```

Tesseract was used to extract text from relatively low-resolution image files of resumes taken with a mobile phone.



```
===== Text From DOCX =====
Jeong Hoon Choi
Los Angeles, California | (323)630-6334 | choijeon@usc.edu

education
University of Southern California       Aug 2024 - May 2026
Master of Science in Applied Data Science | GPA 3.9/4.0 Los Angeles, CA

Kyung Hee University    Mar 2017 - Feb 2023
Bachelor of Science in Physics & Computer Science | GPA 3.8/4.3 Seoul, Republic of Korea

SKILLS
Programming Languages: C/C++, Python, R, Assembly x86, Emacs Lisp, SQL, Bash, PowerShell

Frameworks: CUDA, Boost, Scikit-Learn, TensorFlow, PyTorch, Apache Spark, Apache Flink, Hugging Face

Technologies: Linux, Windows Server, Xen, Docker, AWS, Git, Hadoop, Kafka, MySQL, PostgreSQL, SQL Server, MongoDB

experience
Kyung Hee University, Surface Physics & Organic Nano Device Laboratory  Sep 2022 - Feb 2024
Research Internship     Seoul, Republic of Korea
Collaborated with the Korea Research Institute of Standards and Science (KRISS) to generate and analyze high-resoluti
on X-ray Photoelectron Spectroscopy (XPS) data (+100K spectra) for surface contamination layer identification
Designed and trained a 1D Convolutional Deep Neural Network (4.5M parameters) to predict elemental composition and co
ntamination layers from spectral data
Achieved an R² = 0.998, demonstrating exceptional accuracy in quantitative surface analysis and outperforming traditi
onal regression-based methods
Automated AES/XPS data acquisition with centralized server integration, reducing manual preprocessing and visualizati
on time by ~40% through pipeline optimization
```

Finally, I used python-docx to extract text directly from a Microsoft Word file with the *.docx extension.