

# Week 2 – Data Collection & Extraction

## Tesseract OCR Guide

Tesseract is an open-source Optical Character Recognition (OCR) engine developed by Google. It is used to convert scanned images, PDFs, and images with text into machine-readable text. Tesseract supports more than 100 languages, including complex scripts like Arabic, Chinese, and many others.

---

---

## Installation

### Installing Tesseract on Different Platforms

1. **On macOS** (using Homebrew):

2. `brew install tesseract`

3. **On Ubuntu/Debian:**

4. `sudo apt update`

5. `sudo apt install tesseract-ocr`

6. **On Windows:**

- Download the Tesseract installer from the [Tesseract GitHub releases](#).

- Follow the installation instructions, and ensure to add Tesseract to your system's PATH.

User variables for idali	
Variable	Value
ChocolateyLastPathUpdate	133180374957958006
NVM_HOME	C:\Users\idali\AppData\Roaming\nvm
NVM_SYMLINK	C:\Program Files\nodejs
OneDrive	C:\Users\idali\OneDrive
OneDriveConsumer	C:\Users\idali\OneDrive
Path	C:\Program Files\RSA SecurID Token Common\;C:\App Installation\Scripts\;C:\App Installation\;C:\Users\idali\AppData\Local\Temp\;C:\Program Files\JetBrains\PyCharm Community Edition 2022.2.3\bin;
PyCharm Community Edit...	C:\Program Files\JetBrains\PyCharm Community Edition 2022.2.3\bin;
TEMP	C:\WINDOWS\TEMP

[New...](#) [Edit...](#) [Delete](#)

System variables	
Variable	Value
TEMP	C:\WINDOWS\TEMP
Tesseract	C:\Program Files\Tesseract-OCR\
TMP	C:\WINDOWS\TEMP
USERNAME	SYSTEM
VBOX_MSI_INSTALL_PATH	C:\Program Files\Oracle\VirtualBox\
windir	C:\WINDOWS
7ZC ENIADIE CVCMANI	1

[New...](#) [Edit...](#) [Delete](#)

```
Command Prompt
Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.

C:\Users\idali>tesseract -v
tesseract v5.5.0.20241111
leptonica-1.85.0
  libgif 5.2.2 : libjpeg 8d (libjpeg-turbo 3.0.4) : libpng 1.6.44 : libtiff 4.7.0 : zlib 1.3.1 : libwebp 1.4.0 : libopenjp2 2.5.2
  Found AVX2
  Found AVX
  Found FMA
  Found SSE4.1
  Found libarchive 3.7.7 zlib/1.3.1 liblzma/5.6.3 bz2lib/1.0.8 liblz4/1.10.0 libzstd/1.5.6
  Found libcurl/8.11.0 Schannel zlib/1.3.1 brotli/1.1.0 zstd/1.5.6 libidn2/2.3.7 libpsl/0.21.5 libssh2/1.11.0
```

## Basic Usage

Once Tesseract is installed, you can use it either from the command line or by using the Python wrapper `pytesseract`.

## Command Line Usage

To perform OCR on an image:

```
tesseract image.png output.txt
```

## Use Tesseract with python

```
from PIL import Image  
import pytesseract #pip install pytesseract first  
  
# Load an image using Pillow (PIL)  
image = Image.open('image.png')  
  
# Perform OCR on the image  
text = pytesseract.image_to_string(image)  
  
print(text)
```

Tesseract supports over 100 languages, and you can even train it for custom languages or fonts. To use a different language, you can download the corresponding trained data files and specify the language in the -l flag.

For example, to use Spanish (spa):

```
tesseract image.png output -l spa
```

### Tesseract Best Practices

**Preprocess Images:** Always preprocess images by converting to grayscale, adjusting brightness/contrast, and removing noise to improve Tesseract's accuracy.

**Use Correct --psm:** The Page Segmentation Mode (--psm) plays a crucial role in how Tesseract segments the image and interprets the text. Experiment with different modes for complex documents.

**Choose the Right Language:** Always specify the correct language (-l lang\_code) for better accuracy. Tesseract performs poorly when the language is incorrect.

**Example1-Preprocess Images:**

```

ocr_with_preprocessing.py ×
ocr_with_preprocessing.py > ...
20   cv2.imwrite(processed_image, img)
21   print(f"Image preprocessing completed. Saved as {processed_image}")
22
23 # ===== Step 2: OCR Recognition =====
24 # Open the preprocessed image
25 image = Image.open(processed_image)
26
27 # Perform OCR using Spacy
28 text = pytesseract.image_to_string(image, lang='spa')
29
30 # Print the recognized text
31 print("OCR Result:\n")
32 print(text)
33
34 # ===== Optional: Save the result to a text file =====
35 with open("output_food_descprition.txt", "w", encoding="utf-8") as f:
36     f.write(text)
37 print("\nText recognition completed. Result saved to output.txt")
38

```

Problems Output Debug Console Terminal Ports

PS C:\AI\_Coop\Homework\Week2\Project> & C:/Python313/python.exe c:/AI\_Coop/Homework/Week2/Project/ocr\_with\_preprocessing.py

- Image preprocessing completed. Saved as processed\_food\_description.png
- OCR Result:

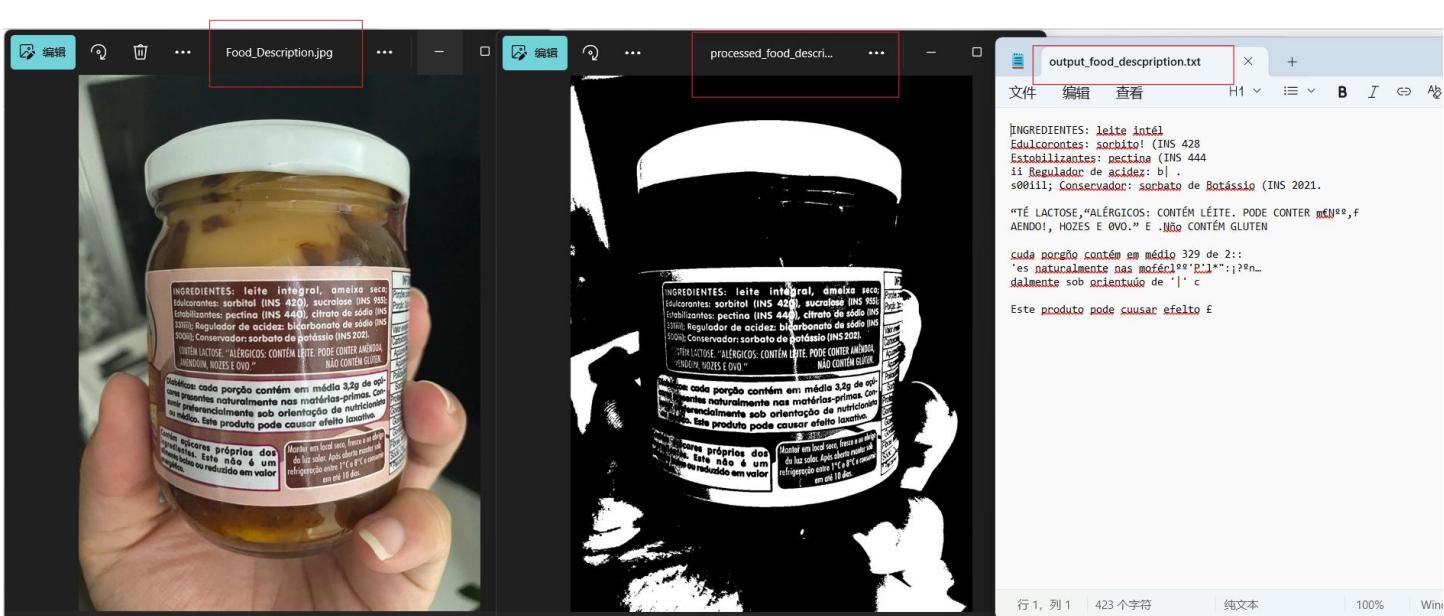
INGREDIENTES: leite intél  
 Edulcorantes: sorbito! (INS 428)  
 Estabilizantes: pectina (INS 444)  
 ii Regulador de acidez: b| .  
 s00iil; Conservador: sorbato de Botássio (INS 2021).

“TÉ LACTOSE, “ALÉRGICOS: CONTÉM LÉITE. PODE CONTER mENº,f  
 AENDO!, HOZES E ØVO.” E .Nº CONTÉM GLUTEN

cuda porgño contém em médio 329 de 2::  
 'es naturalmente nas moférlooo'P'l\*":j?on...  
 damente sob orientuúo de '| ' c

Este produto pode cuusar efeito f

Text recognition completed. Result saved to output.txt

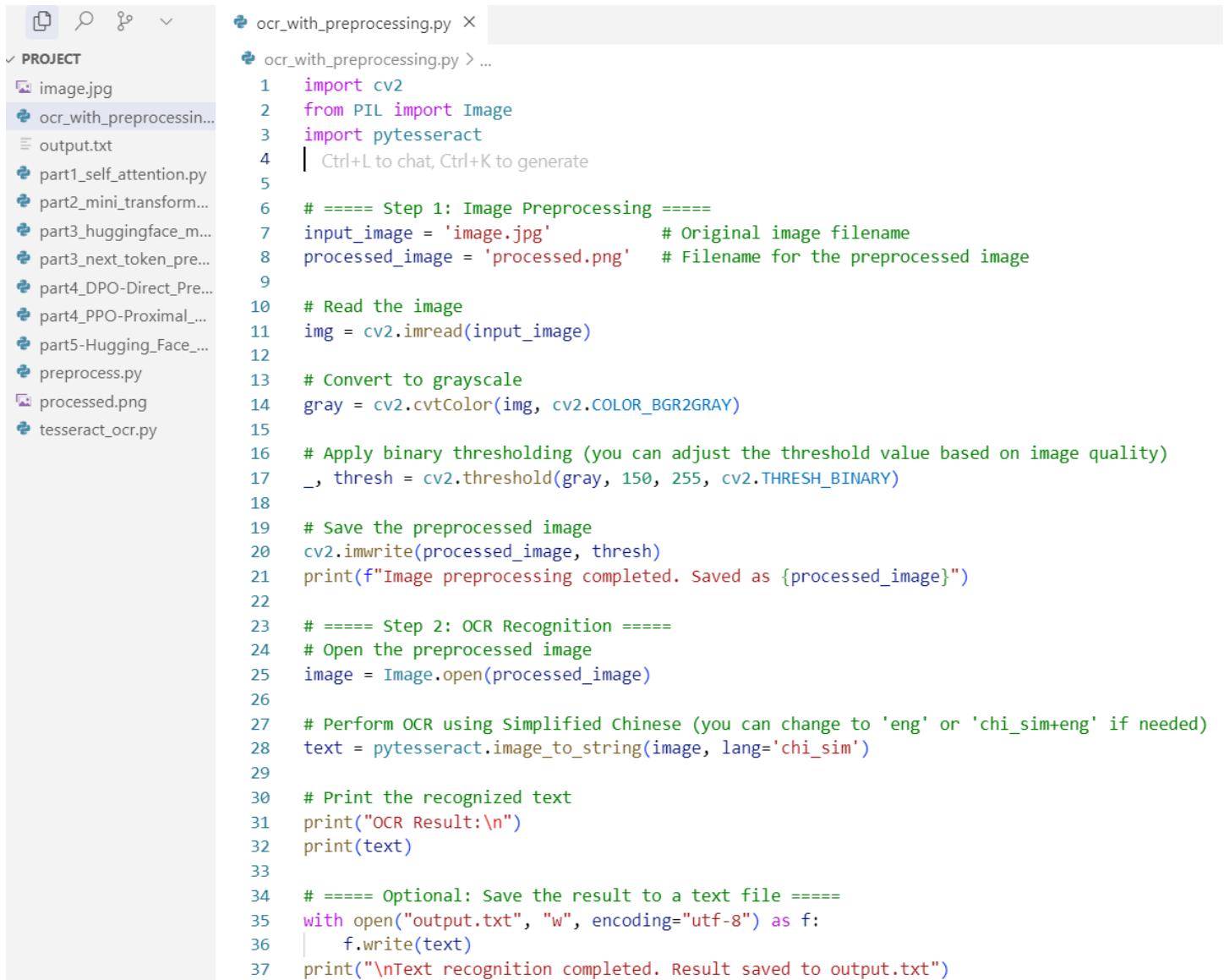


## Example 2 - Use Correct --psm:

```
ocr_with_preprocessing.py > ...
15 # Apply binary thresholding (you can adjust the threshold value based on image quality)
16 _, thresh = cv2.threshold(gray, 150, 255, cv2.THRESH_BINARY)
17
18 # Save the preprocessed image
19 cv2.imwrite(processed_image, thresh)
20 print(f"Image preprocessing completed. Saved as {processed_image}")
21
22 # ===== Step 2: OCR Recognition =====
23 # Open the preprocessed image
24 image = Image.open(processed_image)
25
26 # Test several PSM modes
27 for psm in [3, 6, 7, 11]:
28     print(f"\n--- OCR Result with PSM {psm} ---")
29     text = pytesseract.image_to_string(image, lang='spa', config=f'--psm {psm}')
30     print(text)
31
32 # ===== Optional: Save the result to a text file =====
33 with open(f"output_food_descprition_psm{psm}.txt", "w", encoding="utf-8") as f:
34     f.write(text)
35
36 print("\nText recognition completed. Result saved to output.txt")
37
```

Food_Description.jpg	8/2/2025 9:34 PM
image.jpg	8/2/2025 8:55 PM
ocr_with_preprocessing.py	8/2/2025 11:10 PM
output.txt	8/2/2025 9:22 PM
output_food_descpription.txt	8/2/2025 9:41 PM
output_food_descprition_psm3.txt	8/2/2025 11:11 PM
output_food_descprition_psm6.txt	8/2/2025 11:11 PM
output_food_descprition_psm7.txt	8/2/2025 11:11 PM
output_food_descprition_psm11.txt	8/2/2025 11:11 PM
part1_self_attention.py	7/29/2025 10:07 PM
part2_mini_transformer_block_in_pytorch.py	7/31/2025 7:27 PM
part3_huggingface_mistral_test.py	8/1/2025 8:47 PM
part3_next_token_prediction_using_huggingface.py	8/1/2025 10:27 PM
part4_DPO-Direct_Preference_Optimization.py	8/1/2025 9:27 PM
part4_PPO-Proximal_Policy_Optimization.py	8/1/2025 10:38 PM
part5-Hugging_Face_Transformers.py	8/1/2025 10:38 PM
preprocess.py	7/29/2025 9:10 PM
processed.png	8/2/2025 9:22 PM
processed_food_description.png	8/2/2025 9:41 PM
processed_food_description_multi.png	8/2/2025 11:11 PM
tesseract_ocr.py	7/29/2025 9:11 PM

### Example 3 - Choose the Right Language: (-l lang\_code)



The screenshot shows a code editor interface with a project sidebar on the left and a main code editor window on the right.

**Project Sidebar:**

- PROJECT
- image.jpg
- ocr\_with\_preprocessing.py
- output.txt
- part1\_self\_attention.py
- part2\_mini\_transform...
- part3\_huggingface\_m...
- part3\_next\_token\_pre...
- part4\_DPO-Direct\_Pre...
- part4\_PPO-Proximal\_...
- part5-Hugging\_Face...
- preprocess.py
- processed.png
- tesseract\_ocr.py

**Main Code Editor (ocr\_with\_preprocessing.py):**

```
 1 import cv2
 2 from PIL import Image
 3 import pytesseract
 4 # Ctrl+L to chat, Ctrl+K to generate
 5
 6 # ===== Step 1: Image Preprocessing =====
 7 input_image = 'image.jpg'          # Original image filename
 8 processed_image = 'processed.png' # Filename for the preprocessed image
 9
10 # Read the image
11 img = cv2.imread(input_image)
12
13 # Convert to grayscale
14 gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
15
16 # Apply binary thresholding (you can adjust the threshold value based on image quality)
17 _, thresh = cv2.threshold(gray, 150, 255, cv2.THRESH_BINARY)
18
19 # Save the preprocessed image
20 cv2.imwrite(processed_image, thresh)
21 print(f"Image preprocessing completed. Saved as {processed_image}")
22
23 # ===== Step 2: OCR Recognition =====
24 # Open the preprocessed image
25 image = Image.open(processed_image)
26
27 # Perform OCR using Simplified Chinese (you can change to 'eng' or 'chi_sim+eng' if needed)
28 text = pytesseract.image_to_string(image, lang='chi_sim')
29
30 # Print the recognized text
31 print("OCR Result:\n")
32 print(text)
33
34 # ===== Optional: Save the result to a text file =====
35 with open("output.txt", "w", encoding="utf-8") as f:
36     f.write(text)
37 print("\nText recognition completed. Result saved to output.txt")
```

● PS C:\AI\_Coop\Homework\Week2\Project> & C:/Python313/python.exe c:/AI\_Coop/Homework/Week2/Project/ocr\_with\_preprocessing.py  
Image preprocessing completed. Saved as processed.png  
OCR Result:

人生回报率最高的 21 件事

@睡前写感恩小事

加州大学教授 Emmons 实验：坚持记“小确幸”，  
3 个月后血清素（天然抗抑郁物质）水平涨 23%。  
不靠药，心情也能稳稳幸福，失眠、emo 退散

@工资到账先存 20%

巴菲特 2020 年给股东算账：每月定投指数基金，退休能多 300 万被动收入！延迟满足超赚，让“复利魔法”帮你攒大钱，养老不愁

@@年度目标贴冰箱上

多米尼加大学追踪 267 个创业者：把计划可视化，目标达成率比光心里想高 1.4 倍！视觉一刺激，大脑“行动区”直接激活，冲就完事儿

@@年度目标贴冰箱上

多米尼加大学追踪 267 个创业者：把计划可视化，目标达成率比光心里想高 1.4 倍！视觉一刺激，大脑“行动区”直接激活，冲就完事儿

人周末去公园吸“绿”

埃克塞特大学用手环监测：每周接触自然 2 小时，压力激素水平暴降 53%，比 80% 抗焦虑药还牛！绿色

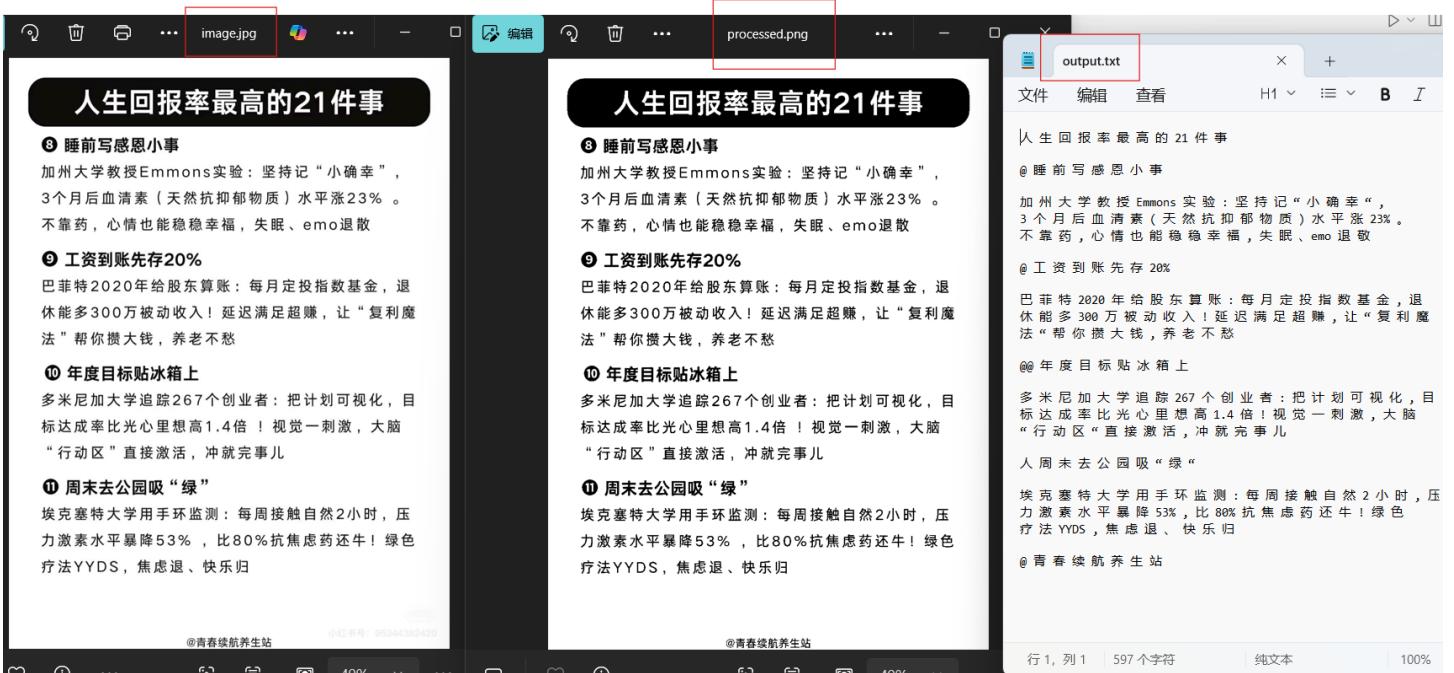
疗法 YYDS，焦虑退、快乐归

@青春续航养生站

@青春续航养生站

Text recognition completed. Result saved to output.txt

PS C:\AI\_Coop\Homework\Week2\Project> █



Custom Training: For specialized fonts or languages, you can train Tesseract to recognize custom fonts or languages. This is especially useful for handwriting or unusual fonts.

```

root@530399ee769c:/train  x  +  ~

Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.

C:\Users\idali>git clone https://github.com/guiem/train-tesseract.git
fatal: destination path 'train-tesseract' already exists and is not an empty directory.

C:\Users\idali>cd train-tesseract

C:\Users\idali\train-tesseract>docker build -t my-tesseract-trainer .
[+] Building 0.8s (14/14) FINISHED          docker:desktop-linux
=> [internal] load build definition from Dockerfile      0.0s
=> => transferring dockerfile: 1.59kB                  0.0s
=> [internal] load metadata for docker.io/library/ubuntu      0.5s
=> [auth] library/ubuntu:pull token for registry-1.docker      0.0s
=> [internal] load .dockerrcignore      0.0s
=> => transferring context: 2B      0.0s
=> [1/8] FROM docker.io/library/ubuntu:18.04@sha256:152      0.0s
=> => resolve docker.io/library/ubuntu:18.04@sha256:152      0.0s
=> [internal] load build context      0.0s
=> => transferring context: 38B      0.0s
=> CACHED [2/8] RUN apt-get update &&      apt-get insta      0.0s
=> CACHED [3/8] RUN apt-get install -y --reinstall make      0.0s
=> CACHED [4/8] WORKDIR /app      0.0s
=> CACHED [5/8] COPY requirements.txt ./      0.0s
=> CACHED [6/8] RUN mkdir src && cd /app/src &&      wge      0.0s
=> CACHED [7/8] RUN pip3 install -r requirements.txt      0.0s
=> CACHED [8/8] RUN apt-get install -y locales && local      0.0s
=> exporting to image      0.1s
=> => exporting layers      0.0s
=> => exporting manifest sha256:c81050894bc86d1794f9321      0.0s
=> => exporting config sha256:41a0f309cebc730f40c2df799      0.0s
=> => exporting attestation manifest sha256:22d33f70976      0.0s
=> => exporting manifest list sha256:06d550f01bd4fb891c      0.0s
=> => naming to docker.io/library/my-tesseract-trainer:      0.0s
=> => unpacking to docker.io/library/my-tesseract-train      0.0s

1 warning found (use docker --debug to expand):
- LegacyKeyValueFormat: "ENV key=value" should be used instead of legacy "ENV key value" format (line 5)

```

```
C:\Users\idali\train-tesseract>docker run -it -v C:\AI_Coop\Homework\Week2\Project\my_handwriting_train:/train my-tesseract-trainer bash
root@530399ee769c:/app# cd /train
root@530399ee769c:/train# ls
my_handwriting_sample.jpg training_text.txt
root@530399ee769c:/train# |
```

```
root@530399ee769c:/train# convert my_handwriting_sample.jpg my_handwriting_sample.tif
bash: convert: command not found
root@530399ee769c:/train# apt update && apt install -y imagemagick
Hit:1 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu bionic InRelease
Hit:3 http://archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
 fonts-droid-fallback fonts-noto-mono ghostscript gsfonts
 hicolor-icon-theme imagemagick-6-common imagemagick-6.q16
 libavahi-client3 libavahi-common-data libavahi-common3 libcups2
 libcupsfilters1 libcupsimage2 libdjvuibre-text libdjvuibre21
 libfftw3-double3 libgs9 libgs9-common libidn11 libijs-0.35
 libilmbase12 libjbig2dec0 liblcms2-2 liblqr-1-0
 libmagickcore-6.q16-3 libmagickcore-6.q16-3-extra
 libmagickwand-6.q16-3 libnetpbm10 libopenexr22 libpaper-utils
 libpaper1 libwmf0.2-7 netpbm poppler-data
Suggested packages:
 fonts-noto ghostscript-x imagemagick-doc autotrace cups-bsd
 | lpr | lprng curl enscript ffmpeg gimp gnuplot grads graphviz
 hp2xx html2ps libwmf-bin mplayer povray radiance sane-utils
 texlive-base-bin transfig ufraw-batch xdg-utils cups-common
 libfftw3-bin libfftw3-dev liblcms2-utils inkscape libjxr-tools
 libwmf0.2-7-gtk poppler-utils fonts-japanese-mincho
 | fonts-ipafont-mincho fonts-japanese-gothic
 | fonts-ipafont-gothic fonts-aphic-ukai fonts-aphic-uming
```

```
root@530399ee769c:/train# convert my_handwriting_sample.jpg my_handwriting_sample.tif
```

```
root@530399ee769c:/train# apt update && apt install -y imagemagick
Hit:1 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu bionic InRelease
Hit:3 http://archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  fonts-droid-fallback fonts-noto-mono ghostscript gsfonts
  hicolor-icon-theme imagemagick-6-common imagemagick-6.q16
  libavahi-client3 libavahi-common-data libavahi-common3 libcups2
  libcupsfilters1 libcupsimage2 libdjvu-libre-text libdjvu-libre21
  libfftw3-double3 libgs9 libgs9-common libidn11 libijs-0.35
  libilmbase12 libjbig2dec0 liblcms2-2 liblqr-1-0
  libmagickcore-6.q16-3 libmagickcore-6.q16-3-extra
  libmagickwand-6.q16-3 libnetpbm10 libopenexr22 libpaper-utils
  libpaper1 libwmf0.2-7 netpbm poppler-data
Suggested packages:
  fonts-noto ghostscript-x imagemagick-doc autotrace cups-bsd
  | lpr | lprng curl enscript ffmpeg gimp gnuplot grads graphviz
  hp2xx html2ps libwmf-bin mplayer povray radiance sane-utils
  texlive-base-bin transfig ufraw-batch xdg-utils cups-common
  libfftw3-bin libfftw3-dev liblcms2-utils inkscape libjxr-tools
  libwmf0.2-7-gtk poppler-utils fonts-japanese-mincho
  | fonts-ipafont-mincho fonts-japanese-gothic
  | fonts-ipafont-gothic fonts-archic-ukai fonts-archic-uming
  fonts-nanum
The following NEW packages will be installed:
  fonts-droid-fallback fonts-noto-mono ghostscript gsfonts
  hicolor-icon-theme imagemagick imagemagick-6-common
  imagemagick-6.q16 libavahi-client3 libavahi-common-data
  libavahi-common3 libcups2 libcupsfilters1 libcupsimage2
  libdjvu-libre-text libdjvu-libre21 libfftw3-double3 libgs9
```

```
cd train-tesseract
```

```
docker build -t my-tesseract-trainer .
```

```
docker run -it -v C:\AI_Coop\Homework\Week2\Project\my_handwriting_train:/train my-tesseract-trainer
bash
```

```
cd /train
```

```
ls
```

```
convert my_handwriting_sample.jpg my_handwriting_sample.tif
```

```
apt update && apt install -y imagemagick
```

Search jTessBc			
Name	Date modified	Type	Size
Yesterday			
program.log	8/3/2025 6:01 PM	文本文档	0 KB
.project	8/3/2025 5:22 PM	PROJECT File	1 KB
jTessBoxEditorFX.jar	8/3/2025 5:22 PM	Executable Jar File	311 KB
readme.html	8/3/2025 5:22 PM	Microsoft Edge HT...	7 KB
README.md	8/3/2025 5:22 PM	Markdown Source ...	2 KB
train	8/3/2025 5:22 PM	File	1 KB
train.bat	8/3/2025 5:22 PM	Windows Batch File	1 KB
versionchanges.txt	8/3/2025 5:22 PM	文本文档	3 KB
lib	8/3/2025 7:45 PM	File folder	
tesseract-ocr	8/3/2025 7:44 PM	File folder	
data	8/3/2025 5:23 PM	File folder	
samples	8/3/2025 5:23 PM	File folder	
tools	8/3/2025 5:23 PM	File folder	

I cannot open this jTessBoxEditorFX.jar to adjust the sample handwriting document and .box document and I got stuck.

### Bonus Hands-on Assignment Sheet

here is bonus part of our homework,which focuses on various aspects of data collection, extraction, and cleaning using OCR (Optical Character Recognition) technology like Tesseract, as well as other techniques such as Web Scraping and Automatic Speech Recognition (ASR). The goal is to apply different tools and methods to extract useful information from web pages, PDFs, audio files, and clean the data for further analysis.

### Task Overview

#	Module / Skill	Task Goal	Core Tools	Deliverables
1	Web Scraping &	arXiv Paper Abstract Scraper • Query any subcategory	trafilatura, pytesseract,	arxiv_clean.json (≤1MB) + scraper script

#	Module / Skill	Task Goal	Core Tools	Deliverables
---	----------------	-----------	------------	--------------

HTML Cleaning	(e.g., cs.CL) to fetch the latest 200 papers.	<ul style="list-style-type: none"> <li>• Scrape the /abs/ page and use <b>Trafilatura</b> to clean the content.</li> <li>• Use <b>Tesseract OCR</b> to extract abstract text from screenshots of the downloaded pages.</li> <li>• Save the data as JSON: {url, title, abstract, authors, date}</li> </ul>	
------------------	---	---	--

arxiv\_scraper.py X

▷ ⌂ ⌂ ⌂

```
arxiv_scraper.py > ...
1  # arxiv_scraper.py
2
3  from bs4 import BeautifulSoup
4  import requests
5  import trafilatura
6  from PIL import Image
7  import pytesseract
8  import time
9  import json
10 from selenium import webdriver
11
12 # Step 1: 获取 /abs/ 页面链接
13 category = "cs.CL"
14 url = f"https://arxiv.org/list/{category}/recent"
15 resp = requests.get(url)
16 soup = BeautifulSoup(resp.text, "html.parser")
17
18 abs_links = ["https://arxiv.org" + a['href'] for a in soup.select("dt a[href^='/abs/'])"]
19 abs_links = list(dict.fromkeys(abs_links)) # 去重
20 abs_links = abs_links[:200] # 限制最多200个
21
22 # Step 2: 提取结构化信息 + 用 Trafilatura 清洗摘要（备用）
23 all_data = []
```

Problems Output Debug Console Terminal Ports

+ ⌂ ⌂ ⌂ ⌂

```
I0000 00:00:1754268252.152370 30428 voice_transcription.cc:58] Registering VoiceTranscriptionCapability
[16992:30632:0803/204412.471:ERROR:google_apis\gcm\engine\registration_request.cc:291] Registration response error message: DEPRECATED_ENDPOINT
```

```
DevTools listening on ws://127.0.0.1:52668/devtools/browser/7c6946a5-e8f1-45c5-a8eb-62495dc9c80a
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
I0000 00:00:1754268281.109236 31348 voice_transcription.cc:58] Registering VoiceTranscriptionCapability
[31252:31360:0803/204441.406:ERROR:google_apis\gcm\engine\registration_request.cc:291] Registration response error message: DEPRECATED_ENDPOINT
[31252:31360:0803/204441.476:ERROR:google_apis\gcm\engine\mcs_client.cc:700] Error code: 401 Error message: Authentication Failed: wrong_secret
[31252:31360:0803/204441.476:ERROR:google_apis\gcm\engine\mcs_client.cc:702] Failed to log in to GCM, resetting connection.
[31252:31360:0803/204502.798:ERROR:google_apis\gcm\engine\registration_request.cc:291] Registration response error message: DEPRECATED_ENDPOINT
Created TensorFlow Lite XNNPACK delegate for CPU.
Attempting to use a delegate that only supports static-sized tensors with a graph that has dynamic-sized tenso
```

[ ] pow...

[ ] Pyth...

```

Hi, I'm Cline
can do all kinds of tasks thanks to breakthroughs in Claude 3 Sonnet's agentic coding capabilities and access to tools that let me create & edit files, explore complex projects, use a browser, and execute terminal commands with your permission, of course! I can even use MCP to create new tools and extend my own capabilities.

Sign up for an account to get started for free, or use an API key that provides access to models like Claude 3.7 Sonnet.

Get Started for Free

Use your own API key

```

arxiv\_scraped.py arxiv\_clean.json

```

735  {
744      "date": "[ 24 Jul 2025]",
745      "abstract": "We gratefully acknowledge the search, navigation, and analysis of arXiv:2507.22930 in all fields. Help | Advance
746  },
747  {
748      "url": "https://arxiv.org/abs/2507.22929",
749      "title": "EH-Benchmark: Ophthalmic Hallucination Benchmark and Agent-Driven Top-Down Traceable Reasoning W
750      "authors": [
751          "Xiaoyu Pan",
752          "Yang Bai",
753          "Ke Zou",
754          "Yang Zhou",
755          "Jun Zhou",
756          "Huazhu Fu",
757          "Yih-Chung Tham",
758          "Yong Liu"
759      ],
760      "date": "[ 24 Jul 2025]",
761      "abstract": "We gratefully acknowledge the search, navigation, and analysis of arXiv:2507.22929 in all fields. Help | Advance
762  },
763  {
764      "url": "https://arxiv.org/abs/2507.22928",
765      "title": "How does Chain of Thought Think? Mechanistic Interpretability of Chain-of-Thought Reasoning wit
766      "authors": [
767          "Xi Chen",
768          "Aske Plaat",
769          "Niki van Stein"
770      ],
771      "date": "[ 24 Jul 2025]",
772      "abstract": "We gratefully acknowledge the search, navigation, and analysis of arXiv:2507.22928 in all fields. Help | Advance
773  }

```

Problems Output Debug Console Terminal Ports + ^ x

DevTools listening on ws://127.0.0.1:49923/devtools/browser/4a351cd5-6fff-4ea2-8977-dad5e21deb67  
 WARNING: All log messages before absl::InitializeLog() is called are written to STDERR  
 I0000 00:00:1754272060.846575 4480 voice\_transcription.cc:58] Registering VoiceTranscriptionCapability  
 WARNING: All log messages before absl::InitializeLog() is called are written to STDERR  
 I0000 00:00:1754272060.846575 4480 voice\_transcription.cc:58] Registering VoiceTranscriptionCapability  
 [14924:19188:0803/214741.147:ERROR:google\_apis\gcm\engine\registration\_request.cc:291] Registration response e  
 I0000 00:00:1754272060.846575 4480 voice\_transcription.cc:58] Registering VoiceTranscriptionCapability

## Batch OCR for arXiv

### PDFs (same paper set as Task)

1).

- Use **Tesseract** to convert PDFs to text.
- Retain OCR layout (e.g., titles, sections) if needed.

pytesseract, pdf2image

pdf\_ocr/ folder with TXT files + code notebook

2 PDF to Text OCR

The screenshot shows a code editor interface with two tabs: 'arxiv\_scraping.py' and 'arxiv\_pdf\_ocr.py'. The 'arxiv\_pdf\_ocr.py' tab is active, displaying a Python script for processing PDF files. The script imports os, json, requests, pdf2image, PIL, and pytesseract. It sets paths for Tesseract and Poppler, creates output folders, loads paper links from 'arxiv\_clean.json', and loops through papers to download PDFs and extract IDs. The script then performs OCR on each PDF. A terminal window at the bottom shows a list of successful file operations and OCR completions.

```
import os
import json
import requests
from pdf2image import convert_from_path
from PIL import Image
import pytesseract
# ----- Step 0: Set the path and configuration -----
# Set the path of tesseract.exe (required only for Windows)
pytesseract.pytesseract.tesseract_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"
# If you haven't set the Poppler environment variable, manually specify the path (Windows)
POPPLER_PATH = r"C:\Users\idali\Downloads\poppler-24.08.0\Library\bin" # Modify it to your poppler decompression path
# Create an output folder
os.makedirs("pdfs", exist_ok=True)
os.makedirs("pdf_ocr", exist_ok=True)
# ----- Step 1: Load the paper link from the JSON file -----
with open("arxiv_clean.json", "r", encoding="utf-8") as f:
    papers = json.load(f)
# ----- Step 2: Download the PDF file -----
for paper in papers:
    arxiv_id = paper["url"].split("/")[-1] # Extract the ID, such as "2408.01234"
    PDF download successfully: 2507.22937
    PDF download successfully: 2507.22936
    PDF download successfully: 2507.22935
    PDF download successfully: 2507.22934
    PDF download successfully: 2507.22933
    PDF download successfully: 2507.22932
    PDF download successfully: 2507.22931
    PDF download successfully: 2507.22930
    PDF download successfully: 2507.22929
    PDF download successfully: 2507.22928
    PDF download successfully: 2507.22927
    OCR Completed: 2507.23776
    OCR Completed: 2507.23740
    OCR Completed: 2507.23661
    OCR Completed: 2507.23588
    OCR Completed: 2507.23577
```

The screenshot shows a file explorer interface with a breadcrumb navigation bar: Week2 > Project\_Bonus >. The main area displays a list of files and folders:

Name	Date modified	Type	Size
pdf_ocr	8/3/2025 11:01 PM	File folder	
pdfs	8/3/2025 10:22 PM	File folder	
arxiv_clean.json	8/3/2025 9:50 PM	JSON Source File	
arxiv_pdf_ocr.py	8/3/2025 11:09 PM	Python Source File	
arxiv_scraping.py	8/3/2025 11:06 PM	Python Source File	
page.png	8/3/2025 9:50 PM	PNG File	
paper_details.json	8/3/2025 8:32 PM	JSON Source File	

pdfs				
	Name	Date modified	Type	Size
Ho	2507.22928.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	5,176 KB
Ga	2507.22929.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	2,936 KB
Ida	2507.22930.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	452 KB
	2507.22931.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	1,183 KB
	2507.22932.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	576 KB
	2507.22933.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	1,416 KB
	2507.22934.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	5,497 KB
	2507.22935.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	826 KB
	2507.22936.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	924 KB
	2507.22937.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	15,270 KB
	2507.22938.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	3,620 KB
	2507.22939.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	1,676 KB
	2507.22940.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	1,474 KB
	2507.22941.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	802 KB
	2507.22943.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	632 KB
	2507.22944.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	3,320 KB
	2507.22968.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	34,222 KB
	2507.23063.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	192 KB
	2507.23082.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	225 KB
	2507.23083.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	178 KB
	2507.23095.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	3,662 KB
	2507.23104.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	1,004 KB
	2507.23121.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	1,449 KB
	2507.23135.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	1,768 KB
	2507.23158.pdf	8/3/2025 10:22 PM	Adobe Acrobat D...	887 KB

50 items

pdf_ocr				
	Name	Date modified	Type	Size
Ho	2507.22928.txt	8/3/2025 11:01 PM	文本文档	73 KB
Ga	2507.22929.txt	8/3/2025 11:00 PM	文本文档	84 KB
Ida	2507.22930.txt	8/3/2025 10:58 PM	文本文档	46 KB
	2507.22931.txt	8/3/2025 10:57 PM	文本文档	62 KB
	2507.22932.txt	8/3/2025 10:56 PM	文本文档	36 KB
	2507.22933.txt	8/3/2025 10:56 PM	文本文档	130 KB
	2507.22934.txt	8/3/2025 10:54 PM	文本文档	136 KB
	2507.22935.txt	8/3/2025 10:52 PM	文本文档	103 KB
	2507.22936.txt	8/3/2025 10:50 PM	文本文档	67 KB
	2507.22937.txt	8/3/2025 10:49 PM	文本文档	56 KB
	2507.22938.txt	8/3/2025 10:49 PM	文本文档	41 KB
	2507.22939.txt	8/3/2025 10:48 PM	文本文档	43 KB
	2507.22940.txt	8/3/2025 10:48 PM	文本文档	109 KB
	2507.22941.txt	8/3/2025 10:47 PM	文本文档	47 KB
	2507.22943.txt	8/3/2025 10:46 PM	文本文档	49 KB
	2507.22944.txt	8/3/2025 10:45 PM	文本文档	85 KB
	2507.22968.txt	8/3/2025 10:44 PM	文本文档	79 KB
	2507.23063.txt	8/3/2025 10:43 PM	文本文档	36 KB
	2507.23082.txt	8/3/2025 10:43 PM	文本文档	33 KB
We	2507.23083.txt	8/3/2025 10:42 PM	文本文档	16 KB
my	2507.23095.txt	8/3/2025 10:42 PM	文本文档	121 KB
Prc	2507.23104.txt	8/3/2025 10:41 PM	文本文档	68 KB
	2507.23121.txt	8/3/2025 10:40 PM	文本文档	65 KB
	2507.23135.txt	8/3/2025 10:39 PM	文本文档	41 KB
	2507.23158.txt	8/3/2025 10:39 PM	文本文档	55 KB

50 items

### Whisper Transcription

#### Bot for 10 short NLP

conference talks (~3 minutes

each).

yt-

talks\_transcripts.jsonl +

transcription script

3 Automatic  
Speech  
Recognition (ASR)

- Use yt-dl to fetch YouTube audio.

yt-dlp, pytesseract

• Transcribe

with Tesseract for any OCR-based text in the transcript

images.

- Save .jsonl with timestamps.

```

PROJE... asr_ocr_transcriber.py > ...
> frames
> pdf_ocr
> pdfs
> transcripts
> videos
arxiv_clean.json
arxiv_pdf_ocr.py
arxiv_scraping.py
asr_ocr_transcriber.py
page.png
paper_details.json
talks_transcripts.jsonl
Youtube_links_3_minu...
asr_ocr_transcriber.py > ...
22 |     "https://www.youtube.com/watch?v=9It06-/0118",
23 ]
24
25 os.makedirs("videos", exist_ok=True)
26 os.makedirs("frames", exist_ok=True)
27 os.makedirs("transcripts", exist_ok=True)
28
29 # Step 1: Download full video (video + audio)
30 def download_video(url):
31     cmd = [
32         "yt-dlp",
33         "-f", "bestvideo+bestaudio/best",
34         "--merge-output-format", "mp4", # force output to mp4
35         "-o", "videos/%(id)s.%s%(ext)s",
36         url
37     ]
38     subprocess.run(cmd, check=True)
39
40 # Step 2: Transcribe audio using Whisper
41 def transcribe_video(video_path):
42     model = whisper.load_model("medium") # or 'large' if you have GPU and need better accuracy
43     result = model.transcribe(video_path)
44     return result
45

```

Problems Output Debug Console Terminal Ports

[youtube] 9It06-7018: Downloading ios player API JSON  
[youtube] 9It06-7018: Downloading m3u8 information  
[info] 9It06-7018: Downloading 1 format(s): 136+251  
[download] Destination: videos\9It06-7018.f136.mp4  
[download] 100% of 8.18MiB in 00:00:01 at 6.24MiB/s  
[download] Destination: videos\9It06-7018.f251.webm  
[download] 100% of 2.29MiB in 00:00:00 at 10.44MiB/s  
[Merger] Merging formats into "videos\9It06-7018.mp4"  
Deleting original file videos\9It06-7018.f251.webm (pass -k to keep)  
Deleting original file videos\9It06-7018.f136.mp4 (pass -k to keep)  
Transcribing video: videos\9It06-7018.mp4 ...  
Extracting frames from: videos\9It06-7018.mp4 ...  
Running OCR for 9It06-7018 ...  
Saving transcription and OCR output for 9It06-7018 ...  
Cleaning up temporary frames for 9It06-7018 ...  
All done!

OUTLINE

PS C:\AI Coop\Homework\Week2\Project Bonus>

**4 Data Cleaning & Deduplication**

**End-to-End Cleaner:**

- Merge the outputs from Tasks 1-3 into one dataset.
- Steps: language detection → strip HTML noise → use MinHash for deduplication (similarity  $\geq 0.7$ ) → remove PII (emails, credit card numbers, phone numbers) →

clean\_corpus.txt + stats.md (token count, removal percentage)

## remove repetitive

n-grams.

## class\_2\_lecture

### Part 1: Attention Mechanism (Self-Attention)

```
part1_attention_mechanism.py X
C: > AI_Coop > Homework > Week2 > Project > part1_attention_mechanism.py > self_attention
1 import numpy as np
2
3 # Random Q, K, V matrices
4 def generate_random_qkv(seq_len=4, d_model=8):
5     return [np.random.rand(seq_len, d_model) for _ in range(3)]
6
7 # Scaled dot-product attention
8 def self_attention(Q, K, V):
9     d_k = Q.shape[-1]
10    scores = np.dot(Q, K.T) / np.sqrt(d_k)
11    weights = softmax(scores)
12    output = np.dot(weights, V)
13    return output, weights
14
15 def softmax(x):
16     exp_x = np.exp(x - np.max(x, axis=-1, keepdims=True))
17     return exp_x / np.sum(exp_x, axis=-1, keepdims=True)
18
19 Q, K, V = generate_random_qkv()
20 out, attn_weights = self_attention(Q, K, V)
21 print("Attention Output:\n", out)
22 print("Attention Weights:\n", attn_weights)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\idali> & C:/Python313/python.exe c:/AI\_Coop/Homework/Week2/Project/part1\_attention\_mechanism.py

● Attention Output:

```
[[0.51834168 0.36705926 0.42507334 0.37301351 0.54690054 0.42639484
 0.6993746 0.80091463]
[0.51944028 0.36302342 0.41606231 0.36579824 0.54249989 0.42096458
 0.70112831 0.79805703]
[0.51580608 0.35286444 0.40977176 0.3749141 0.54439827 0.42541188
 0.69883605 0.8045721 ]
[0.51727463 0.36269226 0.41930202 0.37298043 0.54056164 0.42292799
 0.69904642 0.8013006611]
```

Attention Weights:

```
-- 
[[0.27602408 0.25294813 0.20237117 0.26865662]
[0.2902997 0.24662881 0.19095182 0.27211967]
[0.29429571 0.25946855 0.19451864 0.2517171 ]
[0.28984034 0.24785633 0.20092363 0.26137971]]
```

PS C:\Users\idali>

### Part 2: Mini Transformer Block in PyTorch

```
part2_mini_transformer_block_in_pytorch.py X
part2_mini_transformer_block_in_pytorch.py > ...
1 import torch
2 import torch.nn as nn
3
4 class MiniTransformerBlock(nn.Module):
5     def __init__(self, embed_dim):
6         super().__init__()
7         self.attn = nn.MultiheadAttention(embed_dim, num_heads=2, batch_first=True)
8         self.ffn = nn.Sequential(
9             nn.Linear(embed_dim, embed_dim * 4),
10            nn.ReLU(),
11            nn.Linear(embed_dim * 4, embed_dim)
12        )
13        self.norm1 = nn.LayerNorm(embed_dim)
14        self.norm2 = nn.LayerNorm(embed_dim)
15
16    def forward(self, x):
17        attn_output, _ = self.attn(x, x, x)
18        x = self.norm1(x + attn_output)
19        ffn_output = self.ffn(x)
20        x = self.norm2(x + ffn_output)
21        return x
22
23 x = torch.randn(1, 5, 16) # batch_size=1, seq_len=5, embed_dim=16
```

Problems Output Debug Console Terminal Ports

PS C:\AI\_Coop\Homework\Week2\Project> & C:/Python313/python.exe c:/AI\_Coop/Homework/Week2/Project/part2\_mini\_transformer\_block\_in\_pytorch.py

● torch.size([1, 5, 16])

## Part 3: Next Token Prediction using HuggingFace

### Option 1: Use a Publicly Available Model

Problems 1 Output Debug Console Terminal Ports

P  
P

⚠ Warning: 'huggingface-cli login' is deprecated. Use 'hf auth login' instead.  
The token has not been saved to the git credentials helper. Pass `add\_to\_git\_credential=True` in this function directly or `--add-to-git-credential` if using via 'hf' CLI if you want to set the git credential as well.  
Token is valid (permission: fineGrained).  
The token `test` has been saved to C:\Users\idali\.cache\huggingface\stored\_tokens  
Your token has been saved to C:\Users\idali\.cache\huggingface\token  
Login successful.  
The current active token is: `test`  
PS C:\AI\_Coop\Homework\Week2\Project> █

```

from transformers import AutoTokenizer, AutoModelForCausalLM
from huggingface_hub import login
#login(token="Your_token") # optional if already logged in via CLI
# you have to visit https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1 to sign the agreement in order to use this model
model_name = "mistralai/Mistral-7B-Instruct-v0.1"
tokenizer = AutoTokenizer.from_pretrained(model_name, use_auth_token=True)
model = AutoModelForCausalLM.from_pretrained(model_name, use_auth_token=True, device_map="auto")

```

C:\Python313\lib\site-packages\huggingface\_hub\file\_download.py:143: UserWarning: 'huggingface\_hub' cache-system uses symlinks by default to efficiently store duplicated files but your machine does not support them in C:\Users\idali\.cache\huggingface\hub\models-mistralai-Mistral-7B-Instruct-v0.1. Caching files will still work but in a degraded version that might require more space on your disk. This warning can be disabled by setting the `HF\_HUB\_DISABLE\_SYMLINKS\_WARNING` environment variable. For more details, see https://huggingface.co/docs/huggingface\_hub/how-to-cache#limitations.  
To support symlinks on Windows, you either need to activate Developer Mode or to run Python as an administrator. In order to activate developer mode, see this article: https://docs.microsoft.com/en-us/windows/apps/get-started/enable-your-device-for-development  
warnings.warn(message)  
Fetching 2 files: 0% | 0/2 [00:00<?, ?it/s]  
Storage is enabled for this repo, but the 'hf\_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install huggingface\_hub[hf\_xet]` or `pip install hf\_xet`  
Xet Storage is enabled for this repo, but the 'hf\_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install huggingface\_hub[hf\_xet]` or `pip install hf\_xet`  
Error while downloading from https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1/resolve/ec5deb64f2c6e6fa90c1abf74a91d5c93a9669ca/model-00001-of-00002.safetensors: HTTPSConnectionPool(host='cas-bridge.xethub.hf.co', port=443): Read timed out.  
Trying to resume download...  
Error while downloading from https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1/resolve/ec5deb64f2c6e6fa90c1abf74a91d5c93a9669ca/model-00001-of-00002.safetensors: HTTPSConnectionPool(host='cas-bridge.xethub.hf.co', port=443): Read timed out.  
Trying to resume download...  
model-00001-of-00002.safetensors: 100% | 9.94G/9.94G [02:50<00:00, 52.1MB/s] | 1.04G/9.94G [42:22<6:03:26, 408kB/s] | 4.54G/4.54G [02:52<00:00, 18.1MB/s] | 1.43G/4.54G [42:23<1:32:35, 561kB/s]  
model-00001-of-00002.safetensors: 10% | 2/2 [42:23<00:00, 1271.95s/it] | 2/2 [00:10<00:00, 5.11s/it]  
model-00002-of-00002.safetensors: 100% | 116/116 [00:00<00:00, 453kB/s]  
Fetching 2 files: 100% | 2/2 [42:23<00:00, 1271.95s/it]  
Loading checkpoint shards: 100% | 2/2 [00:10<00:00, 5.11s/it]  
generation\_config.json: 100% | 116/116 [00:00<00:00, 453kB/s]  
Some parameters are on the meta device because they were offloaded to the disk and cpu.

```

from transformers import AutoTokenizer, AutoModelForCausalLM
from huggingface_hub import login
import torch
# Optional if already logged in via CLI
# login(token="your_hf_token")
# -----
# 🚧 Device Selection: CUDA > MPS > CPU
# -----
if torch.cuda.is_available():
    device = torch.device("cuda")
    print("✅ Using CUDA (GPU)")
elif torch.backends.mps.is_available():
    device = torch.device("mps")
    print("🟡 Using MPS (Apple Silicon GPU)")
else:
    device = torch.device("cpu")
    print("🔴 Using CPU")
# -----
# 🚧 Load Model from Hugging Face
# -----
model_name = "mistralai/Mistral-7B-Instruct-v0.1"
# -----
tokenizer = AutoTokenizer.from_pretrained(model_name, use_auth_token=True)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
)
```

● Using CPU  
C:\Python313\lib\site-packages\transformers\models\auto\tokenization\_auto.py:999: FutureWarning: The `use\_auth\_token` argument is deprecated and will be removed in v5 of Transformers. Please use `token` instead.  
warnings.warn(  
C:\Python313\lib\site-packages\transformers\models\auto\auto\_factory.py:492: FutureWarning: The `use\_auth\_token` argument is deprecated and will be removed in v5 of Transformers. Please use `token` instead.  
warnings.warn(  
Loading checkpoint shards: 100% | 2/2 [00:24<00:00, 12.07s/it]  
Setting `pad\_token\_id` to `eos\_token\_id` for open-end generation.  
Generated Output: The Eiffel Tower is located in Paris, France, and is one of the most

## Part 4: DPO vs PPO – Side-by-Side Educational Example

## DPO: Direct Preference Optimization

A screenshot of a code editor interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and part4\_DPO-Direct\_Preference\_Optimization.py - Project - Cursor. The left sidebar shows a project structure with files like image.png, part1\_self\_attention.py, part2\_mini\_transformer\_block\_in\_py..., part3\_huggingface\_mistral\_test.py, part3\_next\_token\_prediction\_using\_h..., part4\_DPO-Direct\_Preference\_Optimi..., part4\_PPO-Proximal\_Policy\_Optimiz..., preprocess.py, and tesseract\_ocr.py. The main code editor tab is part4\_DPO-Direct\_Preference\_Optimization.py, containing the following Python code:

```
import torch
import torch.nn.functional as F

# Simulated log-probs of chosen vs rejected completions
chosen_logp = torch.tensor([[[-1.0]]])
rejected_logp = torch.tensor([[[-2.0]]])

def dpo_loss(chosen_logp, rejected_logp, beta=0.1):
    return -F.logsigmoid((chosen_logp - rejected_logp) / beta).mean()

print("DPO Loss:", dpo_loss(chosen_logp, rejected_logp).item())
```

The terminal tab shows the output: DPO Loss: 4.5398901420412585e-05. The status bar at the bottom indicates PS C:\AI\_Coop\Homework\Week2\Project>, Cursor Tab, Ctrl+K to generate a command, Ln 11, Col 64, Spaces: 4, UTF-8, CRLF, Python 3.13.5 64-bit, Go Live, and a file icon.

## PPO: Proximal Policy Optimization (simplified for in-class demo)

A screenshot of a code editor interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and part4\_PPO-Proximal\_Policy\_Optimization.py - Project - Cursor. The left sidebar shows a project structure with files like image.png, part1\_self\_attention.py, part2\_mini\_transformer\_block\_in\_py..., part3\_huggingface\_mistral\_test.py, part3\_next\_token\_prediction\_using\_h..., part4\_DPO-Direct\_Preference\_Optimi..., part4\_PPO-Proximal\_Policy\_Optimiz..., preprocess.py, and tesseract\_ocr.py. The main code editor tab is part4\_PPO-Proximal\_Policy\_Optimization.py, containing the following Python code:

```
import torch
import torch.nn.functional as F

# Simulated old and new policy log-probs (log pi_theta(a|s) and log pi_theta_old(a|s))
old_log_prob = torch.tensor([[[-1.0]]]) # from reference policy (e.g. GPT-4 before PPO step)
new_log_prob = torch.tensor([[[-0.8]]]) # from updated policy
reward = torch.tensor([[1.0]]) # reward from human or reward model
epsilon = 0.2 # PPO clipping parameter

# Compute ratio of new to old policy
log_ratio = new_log_prob - old_log_prob
ratio = torch.exp(log_ratio)

# Unclipped and clipped advantages
advantage = reward # assume reward ~ advantage for simplicity
clipped_ratio = torch.clamp(ratio, 1 - epsilon, 1 + epsilon)

# PPO loss (negative of the clipped surrogate objective)
ppo_loss = -torch.min(ratio * advantage, clipped_ratio * advantage).mean()
print("PPO Loss:", ppo_loss.item())
```

The terminal tab shows the output: PPO Loss: -1.2000000476837158. The status bar at the bottom indicates mization.py, Python 3.13.5 64-bit, Go Live, and a file icon.

## Bonus: Inference with Quantization (O1 & O3)

```
# Run model with torch_dtype=torch.float16 for O1
```

```
model = AutoModelForCausalLM.from_pretrained(model_name, torch_dtype=torch.float16,
device_map="auto")
```

Loading checkpoint shards: 100%  | 2/2 [00:10<00:00, 5.32s/it]

Concept: Explain how FP16/O1 optimizes memory and speed at inference.

要在 gpu 上跑一下