



The screenshot shows a code editor window with the file 'fetch_and_convert.py' open. The code is a Python script for fetching WordPress posts and converting them to QLORA format. It uses the requests library to make API calls, json to parse responses, and BeautifulSoup to extract post content. The script defines two functions: 'fetch_wordpress_posts' which fetches posts from a base URL across multiple pages, and 'convert_to_qlora_format' which takes a list of posts and returns them in QLORA format. The code is well-structured with comments explaining the logic.

```
#!/usr/bin/env python
# coding: utf-8
# In[1]:
import requests
import time
import json
from bs4 import BeautifulSoup

def fetch_wordpress_posts(base_url, max_pages=50, delay=0.5):
    all_posts = []
    page = 1

    while page <= max_pages:
        url = f"{base_url}/wp-json/wp/v2/posts?per_page=100&page={page}"
        print(f"Fetching: {url}")
        r = requests.get(url)
        if r.status_code != 200:
            break
        posts = r.json()
        if not posts:
            break
        for post in posts:
            all_posts.append({
                "id": post["id"],
                "title": post["title"]["rendered"],
                "content": post["content"]["rendered"]
            })
        page += 1
        time.sleep(delay)

    return all_posts

def convert_to_qlora_format(posts):
    # Implementation of convert_to_qlora_format function
    pass
```

ã??ã?¼ã?¿ã?»ã??ã??ã??ä½?ã??ã?³ã?¼ã??ã•®ä¿®æ£ç??

Description

å?•å??å•~å?ã•®å??å?ã?~å•®è~?äº?ã??ã?~å•?ã?¹ã•|å?å¾?ã•?ã??ã?ã?°ã?©ã? å•~å•?å¾?ã•?ã•~é~?äº?ã?~ã??ã?~ã?¼ã?¿ã?»ã??ã??ã??ä½?æ?~ã?~ã??ã?ã?°ã?©ã? å•?å?¥ã•§ã?ã•?ã?ã?~ã??ã??ã??ï¼?ã•¤ã•®ã??ã?ã?°ã?©ã? å•«ã¾ã?~ã?~ã?~ã•®ã?~ã»¥ã?~ã?§ã?ã??

import requests
import time
import json
from bs4 import BeautifulSoup

def fetch_wordpress_posts(base_url, max_pages=50, delay=0.5):
 all_posts = []
 page = 1

 while page <= max_pages:
 url = f"{base_url}/wp-json/wp/v2/posts?per_page=100&page={page}"
 print(f"Fetching: {url}")
 r = requests.get(url)
 if r.status_code != 200:
 break
 posts = r.json()
 if not posts:
 break
 for post in posts:
 all_posts.append({
 "id": post["id"],
 "title": post["title"]["rendered"],
 "content": post["content"]["rendered"]
 })
 page += 1
 time.sleep(delay)

 return all_posts

def convert_to_qlora_format(posts):
 # Implementation of convert_to_qlora_format function
 pass

