Fine-Tuning OpenAI GPT for Drug-Malady Classification

Using OpenAI to Classify Medical Maladies based on Drug Names

Presented by:

Vaishnavi Patil

Date: 11-20-2024

Table of Contents

Introduction

Overview:

- The purpose of this project is to fine-tune a GPT model to classify drugs and their associated maladies.
- This classification can help in understanding drug uses and medical conditions efficiently.

Objectives:

- Fine-tune OpenAI's GPT model.
- Map drug names to corresponding maladies using a structured dataset.
- Develop a system to predict medical conditions based on drug inputs.

Design

Why This Approach?

1. Identify and Understand Problems:

- The medical domain has an overwhelming number of drugs, and manually associating them with maladies is time-consuming.
- Automating this classification with AI can significantly reduce errors and increase speed.

2. Investigate Possible Solutions:

- Traditional ML algorithms (e.g., SVM, Random Forest) for classification.
- Fine-tuning GPT models for natural language understanding and classification.

3. Comparison and Selection:

- Traditional ML: May require extensive feature engineering, difficult to scale.
- Fine-Tuned GPT: Leverages pre-trained language models to understand context better, faster deployment, and scalability.
- Selected Solution: Fine-tuning OpenAI GPT for automatic drug-malady classification.

Implementation

How This Was Done

1. Data Loading and Preprocessing:

- Load data from an Excel sheet containing drug names and maladies.
- Preprocess and clean the data for model input.

2. Mapping Maladies to Unique Identifiers:

- Created a unique mapping for each malady to help the model classify them correctly.

3. Chat Format Conversion:

- Converted the dataset into a chat format for OpenAI’s fine-tuning API.

4. Fine-Tuning:

- Uploaded the preprocessed data to OpenAI’s API and initiated the fine-tuning process.

5. **Model Testing**:

- After fine-tuning, tested the model with sample drugs to verify accuracy.

Testing

Testing Process:

- Used a sample of 4 drug names to test the model’s predictions.
- The model was able to classify drugs correctly based on the fine-tuned data.

Sample Test Drugs:

- "A CN Gel(Topical) 20gmA CN Soap 75gm"
- "Coralan 5mg Tablet 14'S"
- "Carnisurge Syrup 100ml"
- "Strozina 250mg Injection 4mlStrozina Syrup 60ml"

Result:

- The model successfully predicted the associated maladies based on drug inputs.

Enhancement Ideas

Future Enhancements:

- Larger Dataset: Expand the dataset with more drugs and maladies for improved accuracy.
- Additional Features: Incorporate dosage, drug forms, and geographical variations to enhance prediction capabilities.
- Real-Time Prediction: Implement an API to make real-time predictions for drug-malady classification in a healthcare setting.

Conclusion

Summary:

- The project successfully demonstrates how to use GPT-based models to classify drugs and associated maladies.
- The fine-tuned model provides fast, accurate predictions based on the provided data.

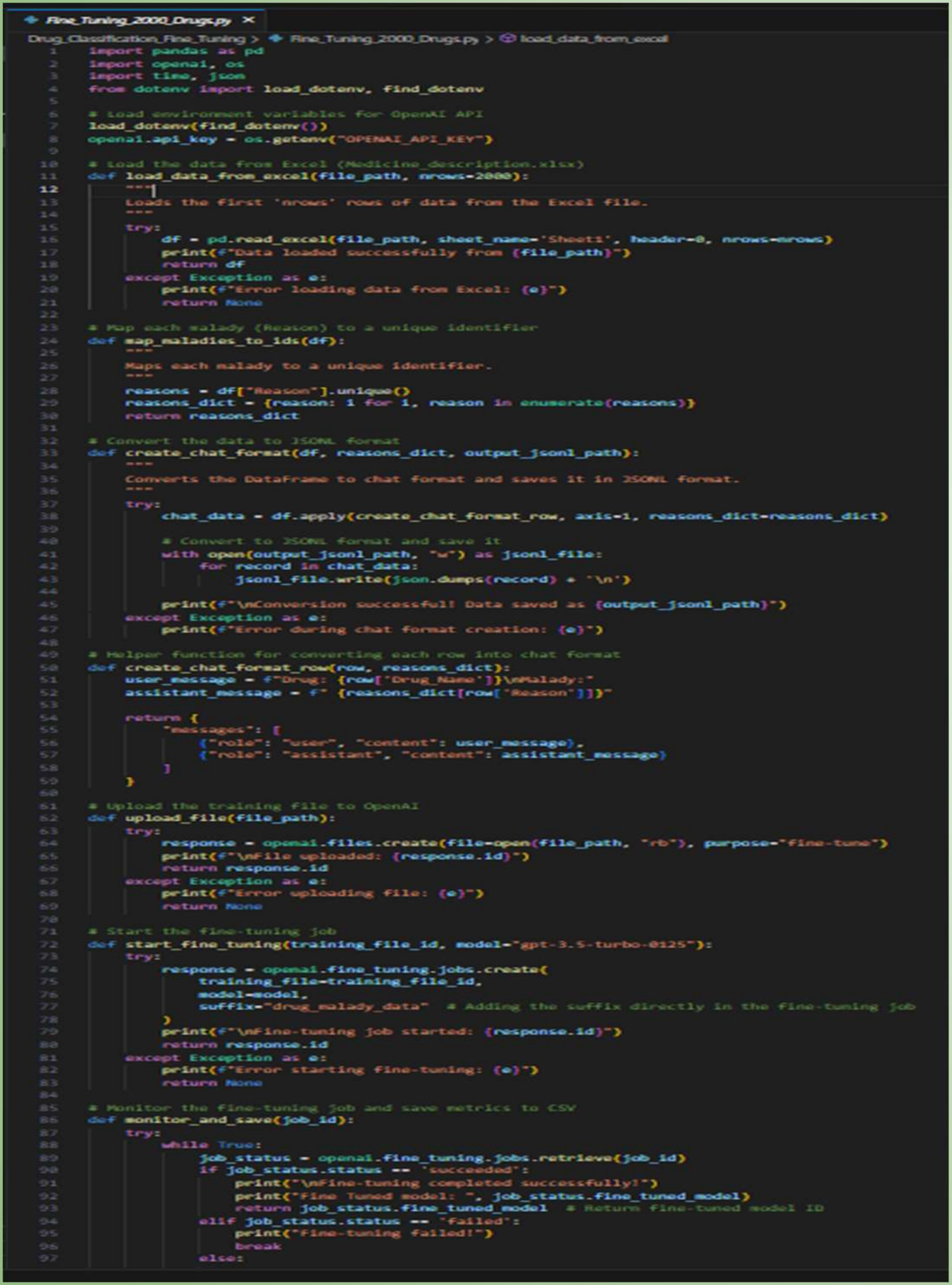
Next Steps:

- Further model testing and refinement with additional data.
- Implementation in healthcare applications for real-time drug classification.

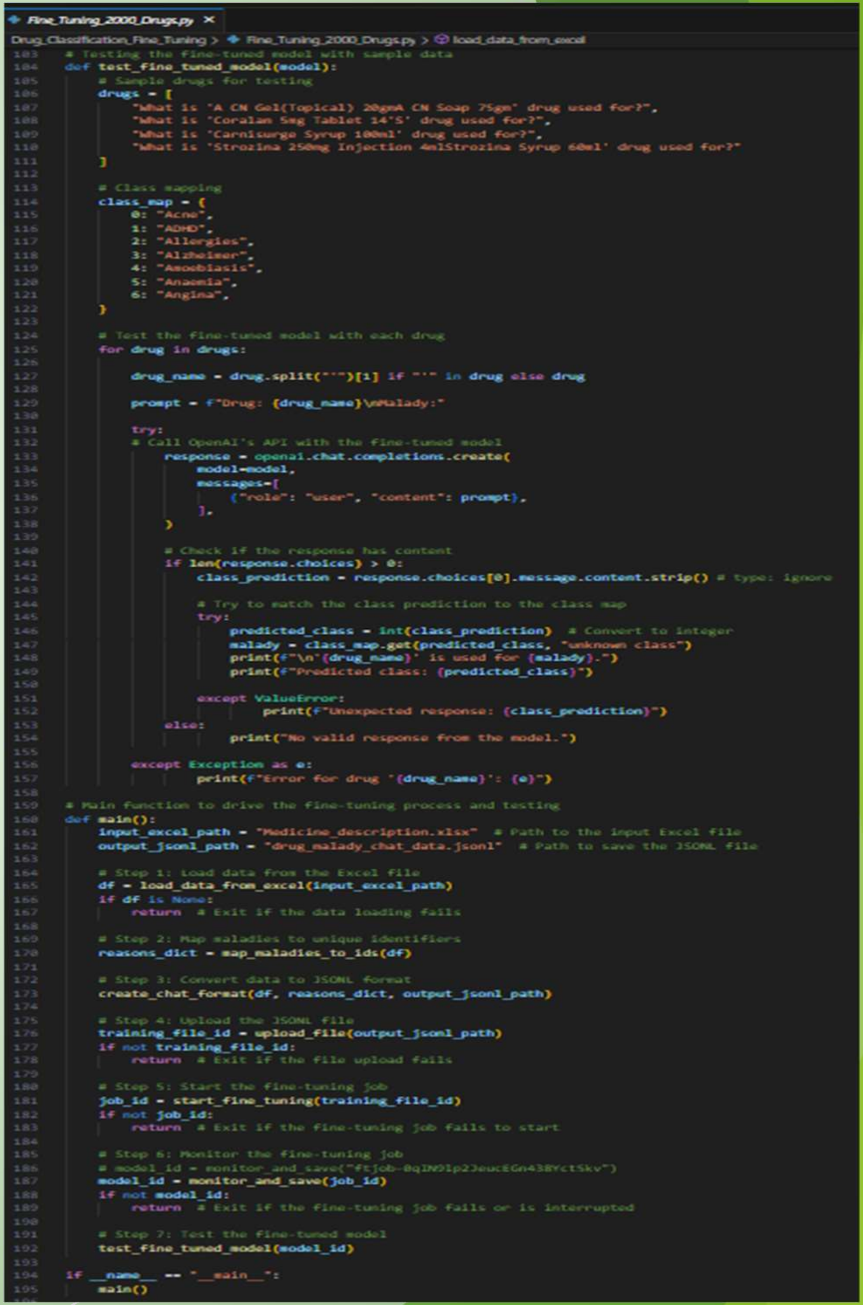
References

- **OpenAI GPT-3 Documentation:** https://beta.openai.com/docs/
- **Pandas Documentation:** https://pandas.pydata.org/pandas-docs/stable/
- **Fine-Tuning GPT Models for Custom Tasks:** https://openai.com/research/

Appendix

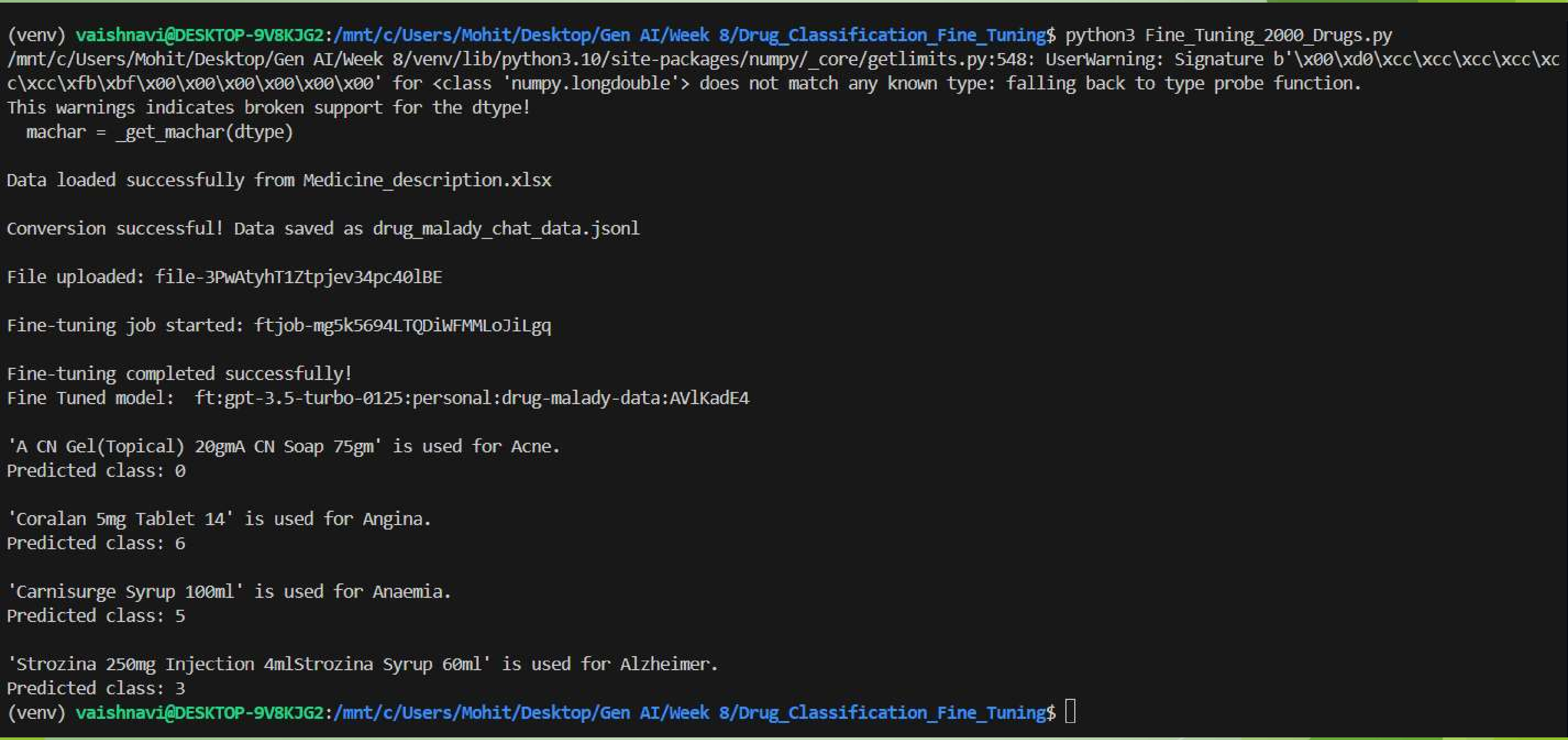

Fine_Tuning_2000_Drugs.py
Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > load_data_from_excel

```python
import pandas as pd
import openai, os
import time, json
from dotenv import load_dotenv, find_dotenv

# Load environment variables for OpenAI API
load_dotenv(find_dotenv())
openai.api_key = os.getenv("OPENAI_API_KEY")

# Load the data from Excel (Medicine_description.xlsx)
def load_data_from_excel(file_path, nrows=2000):
    """
    Loads the first 'nrows' rows of data from the Excel file.
    """
    try:
        df = pd.read_excel(file_path, sheet_name='Sheet1', header=0, nrows=nrows)
        print(f"Data loaded successfully from {file_path}")
        return df
    except Exception as e:
        print(f"Error loading data from Excel: {e}")
        return None

# Map each malady (Reason) to a unique identifier
def map_maladies_to_ids(df):
    """
    Maps each malady to a unique identifier.
    """
    reasons = df["Reason"].unique()
    reasons_dict = {reason: i for i, reason in enumerate(reasons)}
    return reasons_dict

# Convert the data to JSONL format
def create_chat_format(df, reasons_dict, output_jsonl_path):
    """
    Converts the DataFrame to chat format and saves it in JSONL format.
    """
    try:
        chat_data = df.apply(create_chat_format_row, axis=1, reasons_dict=reasons_dict)

        # Convert to JSONL format and save it
        with open(output_jsonl_path, "w") as jsonl_file:
            for record in chat_data:
                jsonl_file.write(json.dumps(record) + '\n')

        print(f"\nConversion successful! Data saved as {output_jsonl_path}")
    except Exception as e:
        print(f"Error during chat format creation: {e}")

# Helper function for converting each row into chat format
def create_chat_format_row(row, reasons_dict):
    user_message = f"Drug: {row['Drug_Name']}\nMalady:"
    assistant_message = f" {reasons_dict[row['Reason']]}"

    return {
        "messages": [
            {"role": "user", "content": user_message},
            {"role": "assistant", "content": assistant_message}
        ]
    }

# Upload the training file to OpenAI
def upload_file(file_path):
    try:
        response = openai.files.create(file=open(file_path, "rb"), purpose="fine-tune")
        print(f"\nFile uploaded: {response.id}")
        return response.id
    except Exception as e:
        print(f"Error uploading file: {e}")
        return None

# Start the fine-tuning job
def start_fine_tuning(training_file_id, model="gpt-3.5-turbo-0125"):
    try:
        response = openai.fine_tuning.jobs.create(
            training_file=training_file_id,
            model=model,
            suffix="drug_malady_data"  # Adding the suffix directly in the fine-tuning job
        )
        print(f"\nFine-tuning job started: {response.id}")
        return response.id
    except Exception as e:
        print(f"Error starting fine-tuning: {e}")
        return None

# Monitor the fine-tuning job and save metrics to CSV
def monitor_and_save(job_id):
    try:
        while True:
            job_status = openai.fine_tuning.jobs.retrieve(job_id)
            if job_status.status == 'succeeded':
                print("\nFine-tuning completed successfully!")
                print("Fine Tuned model: ", job_status.fine_tuned_model)
                return job_status.fine_tuned_model  # Return fine-tuned model ID
            elif job_status.status == 'failed':
                print("Fine-tuning failed!")
                break
            else:
```


Fine_Tuning_2000_Drugs.py
Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > load_data_from_excel

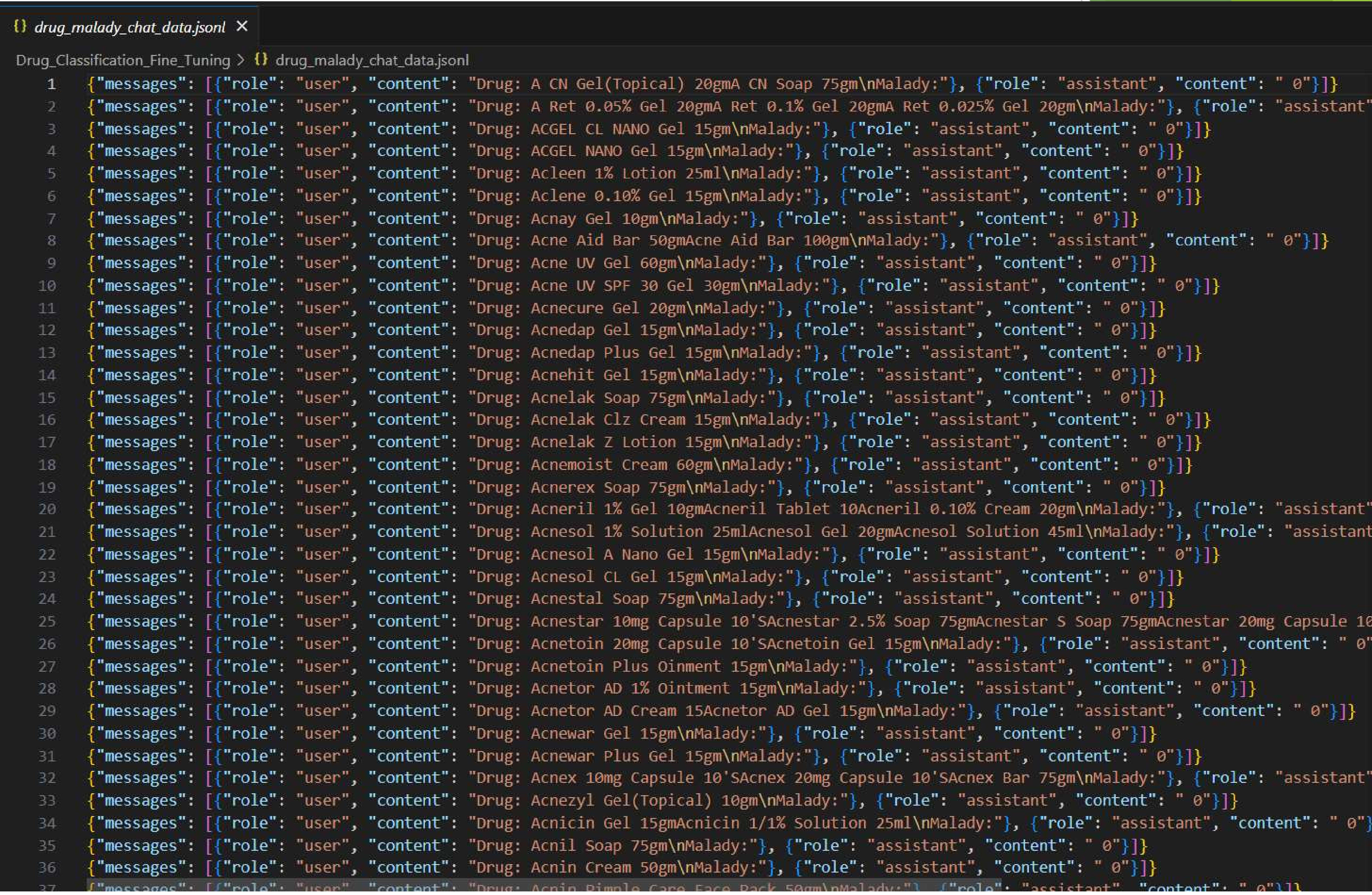
```python
# Testing the fine-tuned model with sample data
def test_fine_tuned_model(model):
    # Sample drugs for testing
    drugs = [
        "What is 'A CN Gel(Topical) 20gmA CN Soap 75gm' drug used for?",
        "What is 'Coralan 5mg Tablet 14'S' drug used for?",
        "What is 'Carnisurge Syrup 100ml' drug used for?",
        "What is 'Strozina 250mg Injection 4mlStrozina Syrup 60ml' drug used for?"
    ]

    # Class mapping
    class_map = {
        0: "Acne",
        1: "ADHD",
        2: "Allergies",
        3: "Alzheimer",
        4: "Amoebiasis",
        5: "Anaemia",
        6: "Angina",
    }

    # Test the fine-tuned model with each drug
    for drug in drugs:

        drug_name = drug.split("'")[1] if "'" in drug else drug

        prompt = f"Drug: {drug_name}\nMalady:"

        try:
        # Call OpenAI's API with the fine-tuned model
            response = openai.chat.completions.create(
                model=model,
                messages=[
                    {"role": "user", "content": prompt},
                ],
            )

            # Check if the response has content
            if len(response.choices) > 0:
                class_prediction = response.choices[0].message.content.strip() # type: ignore

                # Try to match the class prediction to the class map
                try:
                    predicted_class = int(class_prediction)  # Convert to integer
                    malady = class_map.get(predicted_class, "unknown class")
                    print(f"\n'{drug_name}' is used for {malady}.")
                    print(f"Predicted class: {predicted_class}")

                except ValueError:
                        print(f"Unexpected response: {class_prediction}")
            else:
                    print("No valid response from the model.")

        except Exception as e:
                print(f"Error for drug '{drug_name}': {e}")

# Main function to drive the fine-tuning process and testing
def main():
    input_excel_path = "Medicine_description.xlsx"  # Path to the input Excel file
    output_jsonl_path = "drug_malady_chat_data.jsonl"  # Path to save the JSONL file

    # Step 1: Load data from the Excel file
    df = load_data_from_excel(input_excel_path)
    if df is None:
        return  # Exit if the data loading fails

    # Step 2: Map maladies to unique identifiers
    reasons_dict = map_maladies_to_ids(df)

    # Step 3: Convert data to JSONL format
    create_chat_format(df, reasons_dict, output_jsonl_path)

    # Step 4: Upload the JSONL file
    training_file_id = upload_file(output_jsonl_path)
    if not training_file_id:
        return  # Exit if the file upload fails

    # Step 5: Start the fine-tuning job
    job_id = start_fine_tuning(training_file_id)
    if not job_id:
        return  # Exit if the fine-tuning job fails to start

    # Step 6: Monitor the fine-tuning job
    # model_id = monitor_and_save("ftjob-0qIN91p2JeucEGn438YctSkv")
    model_id = monitor_and_save(job_id)
    if not model_id:
        return  # Exit if the fine-tuning job fails or is interrupted

    # Step 7: Test the fine-tuned model
    test_fine_tuned_model(model_id)

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

Appendix



```
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week 8/Drug_Classification_Fine_Tuning$ python3 Fine_Tuning_2000_Drugs.py
/mnt/c/Users/Mohit/Desktop/Gen AI/Week 8/venv/lib/python3.10/site-packages/numpy/_core/getlimits.py:548: UserWarning: Signature b'\x00\xd0\xcc\xcc\xcc\xcc\xc
c\xcc\xfb\xbf\x00\x00\x00\x00\x00\x00' for <class 'numpy.longdouble'> does not match any known type: falling back to type probe function.
This warnings indicates broken support for the dtype!
  machar = _get_machar(dtype)

Data loaded successfully from Medicine_description.xlsx

Conversion successful! Data saved as drug_malady_chat_data.jsonl

File uploaded: file-3PwAtyhT1Ztpjev34pc40lBE

Fine-tuning job started: ftjob-mg5k5694LTQDiWFMMLoJiLgq

Fine-tuning completed successfully!
Fine Tuned model:  ft:gpt-3.5-turbo-0125:personal:drug-malady-data:AVlKadE4

'A CN Gel(Topical) 20gmA CN Soap 75gm' is used for Acne.
Predicted class: 0

'Coralan 5mg Tablet 14' is used for Angina.
Predicted class: 6

'Carnisurge Syrup 100ml' is used for Anaemia.
Predicted class: 5

'Strozina 250mg Injection 4mlStrozina Syrup 60ml' is used for Alzheimer.
Predicted class: 3
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week 8/Drug_Classification_Fine_Tuning$
```

Appendix



drug_malady_chat_data.jsonl ×

Drug_Classification_Fine_Tuning > drug_malady_chat_data.jsonl

```
{"messages": [{"role": "user", "content": "Drug: A CN Gel(Topical) 20gmA CN Soap 75gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA Ret 0.025% Gel 20gm\nMalady:"}, {"role": "assistant"
{"messages": [{"role": "user", "content": "Drug: ACGEL CL NANO Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: ACGEL NANO Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acleen 1% Lotion 25ml\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Aclene 0.10% Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnay Gel 10gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acne Aid Bar 50gmAcne Aid Bar 100gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acne UV Gel 60gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acne UV SPF 30 Gel 30gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnecure Gel 20gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnedap Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnedap Plus Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnehit Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnelak Soap 75gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnelak Clz Cream 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnelak Z Lotion 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnemoist Cream 60gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnerex Soap 75gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acneril 1% Gel 10gmAcneril Tablet 10Acneril 0.10% Cream 20gm\nMalady:"}, {"role": "assistant"
{"messages": [{"role": "user", "content": "Drug: Acnesol 1% Solution 25mlAcnesol Gel 20gmAcnesol Solution 45ml\nMalady:"}, {"role": "assistant
{"messages": [{"role": "user", "content": "Drug: Acnesol A Nano Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnesol CL Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnestal Soap 75gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnestar 10mg Capsule 10'SAcnestar 2.5% Soap 75gmAcnestar S Soap 75gmAcnestar 20mg Capsule 10
{"messages": [{"role": "user", "content": "Drug: Acnetoin 20mg Capsule 10'SAcnetoin Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"
{"messages": [{"role": "user", "content": "Drug: Acnetoin Plus Oinment 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnetor AD 1% Ointment 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnetor AD Cream 15Acnetor AD Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnewar Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnewar Plus Gel 15gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnex 10mg Capsule 10'SAcnex 20mg Capsule 10'SAcnex Bar 75gm\nMalady:"}, {"role": "assistant"
{"messages": [{"role": "user", "content": "Drug: Acnezyl Gel(Topical) 10gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnicin Gel 15gmAcnicin 1/1% Solution 25ml\nMalady:"}, {"role": "assistant", "content": " 0"}
{"messages": [{"role": "user", "content": "Drug: Acnil Soap 75gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnin Cream 50gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
{"messages": [{"role": "user", "content": "Drug: Acnin Pimple Care Face Pack 50gm\nMalady:"}, {"role": "assistant", "content": " 0"}]}
```

Thank You!