

# Fine-Tuning OpenAI GPT for Drug-Malady Classification

Set up environment and install required packages:

```
.env X
venv > .env
1 # Define your OpenAI API key
2 OPENAI_API_KEY = "Replace with your actual API key"
```

And load the key using load\_dotenv and find\_dotenv:

```
Fine_Tuning_2000_Drugs.py X
Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > main
1 import pandas as pd
2 import openai, os
3 import time, json
4 from dotenv import load_dotenv, find_dotenv
5
6 # Load environment variables for OpenAI API
7 load_dotenv(find_dotenv())
8 openai.api_key = os.getenv("OPENAI_API_KEY")
9
```

Microsoft Word ribbon interface showing the 'Home' tab. The ribbon includes sections for Font, Paragraph, Styles, and Editing. The 'Font' section is expanded, showing options for font face (Calibri), size (11), bold, italic, underline, and color. The 'Paragraph' section shows alignment options (left, center, right, justified) and bullet points. The 'Styles' section shows the 'General' style. The 'Editing' section shows options for conditional formatting, insert, delete, and format. The status bar at the bottom indicates 'Page: 1' and 'Words: 100'.

**Step 2: Prepare the Data in JSONL file – Convert the excel file to jsonl file using python code.**

```
Fine_Tuning_2000_Drugs.py X
Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > ...

10 # Load the data from Excel (Medicine_description.xlsx)
11 def load_data_from_excel(file_path, n_rows=2000):
12     """
13     Loads the first 'n_rows' rows of data from the Excel file.
14     """
15     try:
16         df = pd.read_excel(file_path, sheet_name='Sheet1', header=0, n_rows=n_rows)
17         print(f"\nData loaded successfully from {file_path}")
18         return df
19     except Exception as e:
20         print(f"Error loading data from Excel: {e}")
21         return None
22
23 # Map each malady (Reason) to a unique identifier
24 def map_maladies_to_ids(df):
25     """
26     Maps each malady to a unique identifier.
27     """
28     reasons = df["Reason"].unique()
29     reasons_dict = {reason: i for i, reason in enumerate(reasons)}
30     return reasons_dict
31
32 # Convert the data to JSONL format
33 def create_chat_format(df, reasons_dict, output_jsonl_path):
34     """
35     Converts the DataFrame to chat format and saves it in JSONL format.
36     """
37     try:
38         chat_data = df.apply(create_chat_format_row, axis=1, reasons_dict=reasons_dict)
39
40         # Convert to JSONL format and save it
41         with open(output_jsonl_path, "w") as jsonl_file:
42             for record in chat_data:
43                 jsonl_file.write(json.dumps(record) + '\n')
44
45         print(f"\nConversion successful! Data saved as {output_jsonl_path}")
46     except Exception as e:
47         print(f"Error during chat format creation: {e}")
48
49 # Helper function for converting each row into chat format
50 def create_chat_format_row(row, reasons_dict):
51     user_message = f"Drug: {row['Drug_Name']}\nMalady:"
52     assistant_message = f"{reasons_dict[row['Reason']]}"
53
54     return {
55         "messages": [
56             {"role": "user", "content": user_message},
57             {"role": "assistant", "content": assistant_message}
58         ]
59     }
```

```
{ drug_malady_chat_data.jsonl } X
Drug_Classification_Fine_Tuning > {} drug_malady_chat_data.jsonl
1 {"messages": [{"role": "user", "content": "Drug: A CN Gel(Topical) 20gmA CN Soap 75gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
2 {"messages": [{"role": "user", "content": "Drug: A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA Ret 0.025% Gel 20gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
3 {"messages": [{"role": "user", "content": "Drug: ACGEL CL NANO Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
4 {"messages": [{"role": "user", "content": "Drug: ACGEL NANO Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
5 {"messages": [{"role": "user", "content": "Drug: Acleen 1% Lotion 25ml\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
6 {"messages": [{"role": "user", "content": "Drug: Aclene 0.10% Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
7 {"messages": [{"role": "user", "content": "Drug: Acnay Gel 10gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
8 {"messages": [{"role": "user", "content": "Drug: Acne Aid Bar 50gmAcne Aid Bar 100gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
9 {"messages": [{"role": "user", "content": "Drug: Acne UV Gel 60gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
10 {"messages": [{"role": "user", "content": "Drug: Acne UV SPF 30 Gel 30gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
11 {"messages": [{"role": "user", "content": "Drug: Acnecure Gel 20gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
12 {"messages": [{"role": "user", "content": "Drug: Acnedap Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
13 {"messages": [{"role": "user", "content": "Drug: Acnedap Plus Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
14 {"messages": [{"role": "user", "content": "Drug: Acnehit Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
15 {"messages": [{"role": "user", "content": "Drug: Acnelak Soap 75gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
16 {"messages": [{"role": "user", "content": "Drug: Acnelak Clz Cream 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
17 {"messages": [{"role": "user", "content": "Drug: Acnelak Z Lotion 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
18 {"messages": [{"role": "user", "content": "Drug: Acnemoist Cream 60gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
19 {"messages": [{"role": "user", "content": "Drug: Acnerex Soap 75gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
20 {"messages": [{"role": "user", "content": "Drug: Acneril 1% Gel 10gmAcneril Tablet 10Acneril 0.10% Cream 20gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
21 {"messages": [{"role": "user", "content": "Drug: Acnesol 1% Solution 25mlAcnesol Gel 20gmAcnesol Solution 45ml\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
22 {"messages": [{"role": "user", "content": "Drug: Acnesol A Nano Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
23 {"messages": [{"role": "user", "content": "Drug: Acnesol CL Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
24 {"messages": [{"role": "user", "content": "Drug: Acnestal Soap 75gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
25 {"messages": [{"role": "user", "content": "Drug: Acnestar 10mg Capsule 10'SAcnestar 2.5% Soap 75gmAcnestar S Soap 75gmAcnestar 20mg Capsule 10'SAcnestar 2.5% Soap 75gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
26 {"messages": [{"role": "user", "content": "Drug: Acnetoin 20mg Capsule 10'SAcnetoin Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
27 {"messages": [{"role": "user", "content": "Drug: Acnetoin Plus Ointment 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
28 {"messages": [{"role": "user", "content": "Drug: Acnetor AD 1% Ointment 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
29 {"messages": [{"role": "user", "content": "Drug: Acnetor AD Cream 15Acnetor AD Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
30 {"messages": [{"role": "user", "content": "Drug: Acnewar Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
31 {"messages": [{"role": "user", "content": "Drug: Acnewar Plus Gel 15gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
32 {"messages": [{"role": "user", "content": "Drug: Acnex 10mg Capsule 10'SAcnex 20mg Capsule 10'SAcnex Bar 75gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
33 {"messages": [{"role": "user", "content": "Drug: Acnezyl Gel(Topical) 10gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
34 {"messages": [{"role": "user", "content": "Drug: Acnicin Gel 15gmAcnicin 1/1% Solution 25ml\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
35 {"messages": [{"role": "user", "content": "Drug: Acnil Soap 75gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
36 {"messages": [{"role": "user", "content": "Drug: Acnin Cream 50gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
37 {"messages": [{"role": "user", "content": "Drug: Acnis Bimla Cream Face Pack 50gm\\nMalady:"}, {"role": "assistant", "content": " 0"}]}
```

### Step 3: Use this JSONL file as training file for creating a fine-tuning model.

```
Fine_Tuning_2000_Drugs.py X
Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > ...
61 # Upload the training file to OpenAI
62 def upload_file(file_path):
63     try:
64         response = openai.files.create(file=open(file_path, "rb"), purpose="fine-tune")
65         print(f"\nFile uploaded: {response.id}")
66         return response.id
67     except Exception as e:
68         print(f"Error uploading file: {e}")
69         return None
70
71 # Start the fine-tuning job
72 def start_fine_tuning(training_file_id, model="gpt-3.5-turbo-0125"):
73     try:
74         response = openai.fine_tuning.jobs.create(
75             training_file=training_file_id,
76             model=model,
77             suffix="drug_malady_data" # Adding the suffix directly in the fine-tuning job
78         )
79         print(f"\nFine-tuning job started: {response.id}")
80         return response.id
81     except Exception as e:
82         print(f"Error starting fine-tuning: {e}")
83         return None
84
```



**Step 4:** Then, monitor this fine-tuning process. This may take few minutes to complete.

```
Fine_Tuning_2000_Drugs.py X
Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > ...

85 # Monitor the fine-tuning job and save metrics to CSV
86 def monitor_and_save(job_id):
87     try:
88         while True:
89             job_status = openai.fine_tuning.jobs.retrieve(job_id)
90             if job_status.status == 'succeeded':
91                 print("\nFine-tuning completed successfully!")
92                 print("Fine Tuned model: ", job_status.fine_tuned_model)
93                 return job_status.fine_tuned_model # Return fine-tuned model ID
94             elif job_status.status == 'failed':
95                 print("Fine-tuning failed!")
96                 break
97             else:
98                 print(f"Fine-tuning in progress... (status: {job_status.status})")
99                 time.sleep(90) # Wait for 120 seconds before checking again
100     except Exception as e:
101         print(f"Error monitoring job: {e}")
102
```

**Step 5: We will test our fine-tuning model for few sample drugs and print the class for each drug.**

```
Fine_Tuning_2000_Drugs.py X
Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > ...

104 def test_fine_tuned_model(model):
105     # Sample drugs for testing
106     drugs = [
107         "What is 'A CN Gel(Topical) 20gmA CN Soap 75gm' drug used for?",
108         "What is 'Coralan 5mg Tablet 14'S' drug used for?",
109         "What is 'Carnisurge Syrup 100ml' drug used for?",
110         "What is 'Strozina 250mg Injection 4mlStrozina Syrup 60ml' drug used for?"
111     ]
112
113     # Class mapping
114     class_map = {
115         0: "Acne",
116         1: "ADHD",
117         2: "Allergies",
118         3: "Alzheimer",
119         4: "Amoebiasis",
120         5: "Anaemia",
121         6: "Angina",
122     }
123
124     # Test the fine-tuned model with each drug
125     for drug in drugs:
126
127         drug_name = drug.split("")[1] if "" in drug else drug
128
129         prompt = f"Drug: {drug_name}\nMalady:"
130
131         try:
132             # Call OpenAI's API with the fine-tuned model
133             response = openai.chat.completions.create(
134                 model=model,
135                 messages=[
136                     {"role": "user", "content": prompt},
137                 ],
138             )
139
140             # Check if the response has content
141             if len(response.choices) > 0:
142                 class_prediction = response.choices[0].message.content.strip() # type: ignore
143
144                 # Try to match the class prediction to the class map
145                 try:
146                     predicted_class = int(class_prediction) # Convert to integer
147                     malady = class_map.get(predicted_class, "unknown class")
148                     print(f"\n'{drug_name}' is used for {malady}.")
149                     print(f"Predicted class: {predicted_class}")
150
151                 except ValueError:
152                     print(f"Unexpected response: {class_prediction}")
153             else:
154                 print("No valid response from the model.")
155
156         except Exception as e:
157             print(f"Error for drug '{drug_name}': {e}")
158     ...
```

**Step 6: Once the fine tuning is successful, it will print the predicted class for each drug.**

```
Fine-tuning completed successfully!  
Fine Tuned model: ft:gpt-3.5-turbo-0125:personal:drug-malady-data:AVSIQ4zG  
  
'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
```

### Output of the entire script:

[illegible]

```
Fine-tuning completed successfully!
Fine Tuned model: ft:gpt-3.5-turbo-0125:personal:drug-malady-data:AVSIQ4zG

'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
```

**GitHub URL:**

<https://github.com/vaishnavi477/Machine-Learning/tree/main/Generative%20AI/Drug%20Classification%20Fine%20Tuning%20OpenAI>

**Google slide URL:**

[https://docs.google.com/presentation/d/1Z\\_HBdy6WH5qhlnFkl0hzczMFaluY09ap4ZEKowU0-g/edit?usp=sharing](https://docs.google.com/presentation/d/1Z_HBdy6WH5qhlnFkl0hzczMFaluY09ap4ZEKowU0-g/edit?usp=sharing)