

# Fine-Tuning based on 2000 drug examples from an Excel file

Special Topics: Generative AI-Driven Intelligent Apps  
Development

Ademilton Marcelo da Cruz Nunes (19679)

# Links

Github:

<https://github.com/ademiltonnunes/Machine-Learning/tree/main/ChatGPT/Fine-Tuning/2000%20Drug%20Examples>

# Table of Content

- Introduction
- Introduction - LLM
- Introduction - Fine-tuning
- Implementation
- Malady Excel File
- Preparing the Data
- Create a virtual development environment
- Setting openai api key
- Analyze and training the jsonl file
- Train the model - fine-tuning
- Checking Job Progress
- List fine-tuning models
- Completion of Fine-Tuning
- Testing the Fine Tuned Model Completion - Python
- Conclusion

# Introduction

This product exemplifies fine-tuning of a basic LLM, chatGPT 3.5. In order to fine-tune our project, we used an excel file with several examples of remedies and the malady they treat.

I demonstrated how to transform Excel data into the expected fine-tune data format and tested examples.

# Introduction - LLM

Large Language Model (LLM) refers to a type of generative AI model that is trained on a massive scale using large amounts of textual data. These models are designed to understand and generate human-like text. For example GPT-3, developed by OpenAI, is an example of a LLM.

While LLMs are powerful and versatile, they can sometimes provide general or ambiguous answers, especially if the input query is vague or lacks context.

# Introduction - Fine-tuning

Fine-tuning refers to the process of training a LLM on a specific task or domain to make it more specialized and effective for that particular use case.

Fine-tuning leverages the knowledge gained during LLM pre-training and transfers it to the specific task at hand. This transfer of knowledge helps the model generalize better and perform well on tasks even with limited task-specific data.

# Implementation

I will do the fine-tuning using an excel file containing:

- Drug\_name: Medicine name
- Reason: what malady the medicine treat
- Description: The description of the medicine.

Fine-tuning process will train LLM to answer which malady each remedy treats.

# Malady Excel File

	A	B	
1	Drug_Name	Reason	Description
2	A CN Gel(Topical) 20gmA CN Soap 75gm	Acne	Mild to moderate acne (spots)
3	A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA Ret 0.025% Gel 20gm	Acne	A RET 0.025% is a prescription medicine that is used to reduce fine wrinkles
4	ACGEL CL NANO Gel 15gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackhea
5	ACGEL NANO Gel 15gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackhea
6	Acleen 1% Lotion 25ml	Acne	treat the most severe form of acne (nodular acne)
7	Aclene 0.10% Gel 15gm	Acne	treat the most severe form of acne (nodular acne)
8	Acnay Gel 10gm	Acne	treat the most severe form of acne (nodular acne)
9	Acne Aid Bar 50gmAcne Aid Bar 100gm	Acne	treat acne vulgaris
10	Acne UV Gel 60gm	Acne	treat acne vulgaris
11	Acne UV SPF 30 Gel 30gm	Acne	treat mild to moderate acne(spots)
12	Acnecure Gel 20gm	Acne	treatment of dry scaly skin disorders of the scalp
13	Acnedap Gel 15gm	Acne	Mild to moderate acne (spots)
14	Acnedap Plus Gel 15gm	Acne	A RET 0.025% is a prescription medicine that is used to reduce fine wrinkles
15	Acnehit Gel 15gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackhea
16	Acnelak Soap 75gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackhea
17	Acnelak Clz Cream 15gm	Acne	treat the most severe form of acne (nodular acne)
18	Acnelak Z Lotion 15gm	Acne	treat the most severe form of acne (nodular acne)
19	Acnemoist Cream 60gm	Acne	treat the most severe form of acne (nodular acne)
20	Acnerex Soap 75gm	Acne	treat acne vulgaris
21	Acneril 1% Gel 10gmAcneril Tablet 10Acneril 0.10% Cream 20gm	Acne	treat acne vulgaris

Ready



Sheet1

Sheet2

Sheet3

+

:

<

>

100%



22:30

21/11/2023



# Preparing the Data

To fine-tuning using excel file, we have to put data in that format:

```
{"prompt": "Drug: <DRUG NAME>\nMalady:", "completion": " <MALADY NAME>"}
```


To prepare the data, we must install the following modules:

- pip install pandas
- pip install openpyxl
- pip install openai==0.28
- pip install openai[datalib]

# Preparing the Data

The pdf file has 22.483 rows with data, but to make this demonstrations we are extracting only 2000 medicine examples:

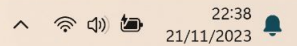
```
n = 2000
df = pd.read_excel('Medicine_description.xlsx', sheet_name='Sheet1', header=0, nrows=n)
df.head()
```

1 to 5 of 5 entries   

index	Drug_Name	Reason	Description
0	A CN Gel(Topical) 20gmA CN Soap 75gm	Acne	Mild to moderate acne (spots)
1	A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA Ret 0.025% Gel 20gm	Acne	A RET 0.025% is a prescription medicine that is used to reduce fine wrinkles
2	ACGEL CL NANO Gel 15gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackheads, white heads and pimple
3	ACGEL NANO Gel 15gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackheads, white heads and pimple
4	Acleen 1% Lotion 25ml	Acne	treat the most severe form of acne (nodular acne)

Show  per page

✓ 0s completed at 10:38 PM



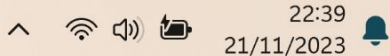
# Preparing the Data

The 2000 medicine examples treat those 7 types of malady. I list each disease by number so that our tests are more consistent.

```
reasons_dict = {reason: i for i, reason in enumerate(reasons)}  
print(reasons_dict)
```

```
➞ {'Acne': 0, 'Adhd': 1, 'Allergies': 2, 'Alzheimer': 3, 'Amoebiasis': 4, 'Anaemia': 5, 'Angina': 6}
```

✓ 0s completed at 10:39 PM



# Preparing the Data

I changed the first column to be:

Drug: <Drug Name> Malady:

```
df["Drug_Name"] = "Drug: " + df["Drug_Name"] + "\n" + "Malady:"  
df.head()
```



	Drug_Name	Reason	Description
0	Drug: A CN Gel(Topical) 20gmA CN Soap 75gm\nMa...	Acne	Mild to moderate acne (spots)
1	Drug: A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA...	Acne	A RET 0.025% is a prescription medicine that i...
2	Drug: ACGEL CL NANO Gel 15gm\nMalady:	Acne	It is used to treat acne vulgaris in people 12...
3	Drug: ACGEL NANO Gel 15gm\nMalady:	Acne	It is used to treat acne vulgaris in people 12...
4	Drug: Acleen 1% Lotion 25ml\nMalady:	Acne	treat the most severe form of acne (nodular ac...

✓ 0s completed at 10:42 PM



# Preparing the Data

Let's transform the name of each malady to its respective numeral value

```
df["Reason"] = " " + df["Reason"].apply(lambda x: "" + str(reasons_dict[x]))
```

```
[25] df.head()
```

	Drug_Name	Reason	Description
0	Drug: A CN Gel(Topical) 20gmA CN Soap 75gm\nMa...	0	Mild to moderate acne (spots)
1	Drug: A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA...	0	A RET 0.025% is a prescription medicine that i...
2	Drug: ACCEL CL NANO Gel 15gm\nMalady:	0	It is used to treat acne vulgaris in people 12...
3	Drug: ACCEL NANO Gel 15gm\nMalady:	0	It is used to treat acne vulgaris in people 12...
4	Drug: Acne 1% Lotion 25ml\nMalady:	0	treat the most severe form of acne (nodular ac...

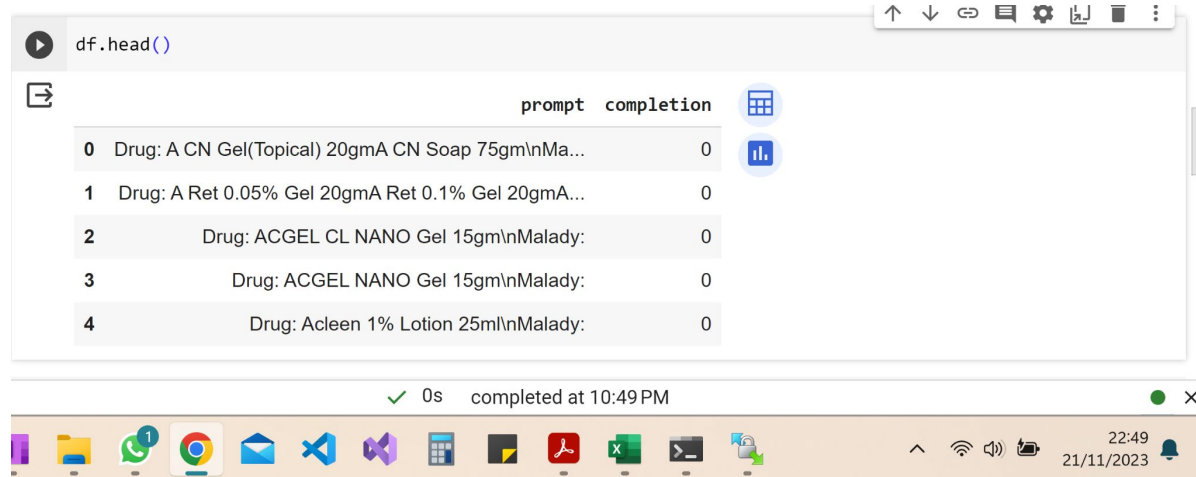
✓ 0s completed at 10:44 PM

# Preparing the Data

I removed the description column, as it will not be used for this example

```
df.drop(["Description"], axis=1, inplace=True)
```

Our values are like this



df.head()

	prompt	completion
0	Drug: A CN Gel(Topical) 20gmA CN Soap 75gm\nMa...	0
1	Drug: A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA...	0
2	Drug: ACGEL CL NANO Gel 15gm\nMalady:	0
3	Drug: ACGEL NANO Gel 15gm\nMalady:	0
4	Drug: Acleen 1% Lotion 25m\nMalady:	0

0s completed at 10:49 PM

22:49 21/11/2023

# Preparing the Data

I converted our data into a jsonl file, which is the expected format in fine-tuning

```
[14] # Convert the dataframe to jsonl format
      jsonl = df.to_json(orient="records", indent=0, lines=True)
      with open("drug_malady_data.jsonl", "w") as f:
          f.write(jsonl)
```

✓ [15] jsonl

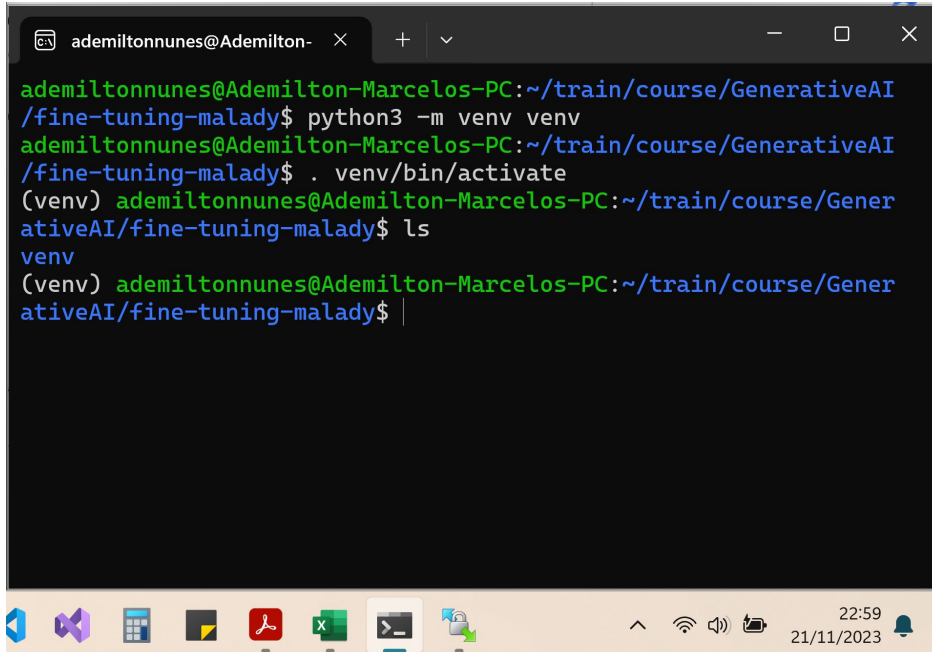
```
'{"prompt": "Drug: A CN Gel(Topical) 20gmA CN Soap 75gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA Ret 0.025% Gel 20gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: ACGEL CL NANO Gel 15gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: ACGEL NANO Gel 15gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acleen 1% Lotion 25ml\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Aclene 0.10% Gel 15gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acnay Gel 10gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acne Aid Bar 50gmAcne Aid Bar 100gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acne UV Gel 60gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acne UV SPF 30 Gel 30gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acnecure Gel 20gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acnedap Gel 15gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acnedap Plus Gel 15gm\\nMalady:", "completion": " 0"}\\n{"prompt": "Drug: Acnehit Gel 15gm\\nMalady..."
```

✓ 0s completed at 10:49 PM



# Create a virtual development environment

This process is optional. However, it is recommended to create a virtual development environment:



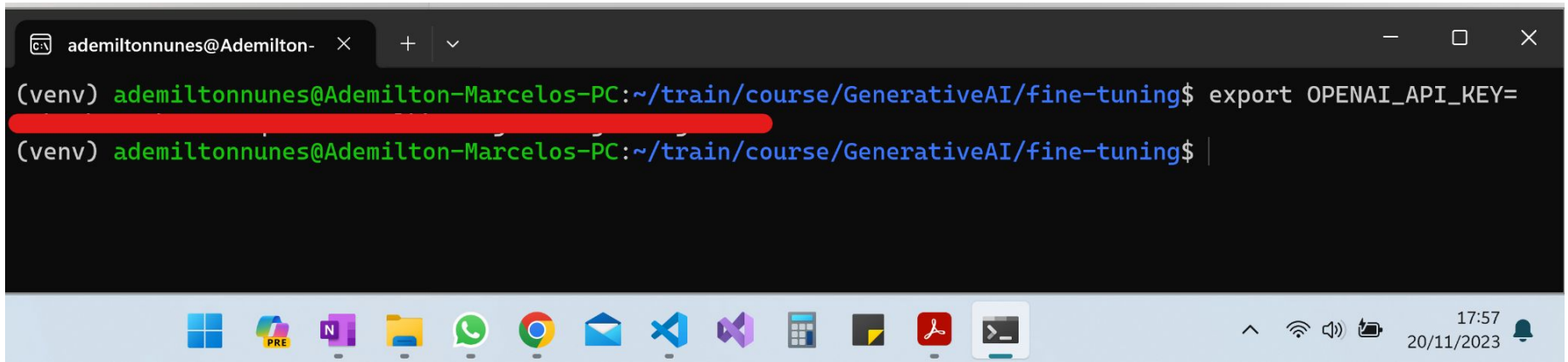
```
ademiltonnunes@Ademilton-PC:~/train/course/GenerativeAI/fine-tuning-malady$ python3 -m venv venv
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$ . venv/bin/activate
(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$ ls
venv
(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$
```

The image shows a terminal window with a dark background. The window title is 'ademiltonnunes@Ademilton-'. The terminal output shows the user running 'python3 -m venv venv' to create a virtual environment named 'venv'. Then, they run '. venv/bin/activate' to activate it, which changes the prompt to '(venv)'. Finally, they run 'ls' and the output shows 'venv' as a directory. The terminal window is part of a desktop environment with a taskbar at the bottom showing various application icons and system status information like '22:59' and '21/11/2023'.



# Setting openai api key

We use the OpenAI API key to do the fine-tuning process. We can put this key in the environment variables

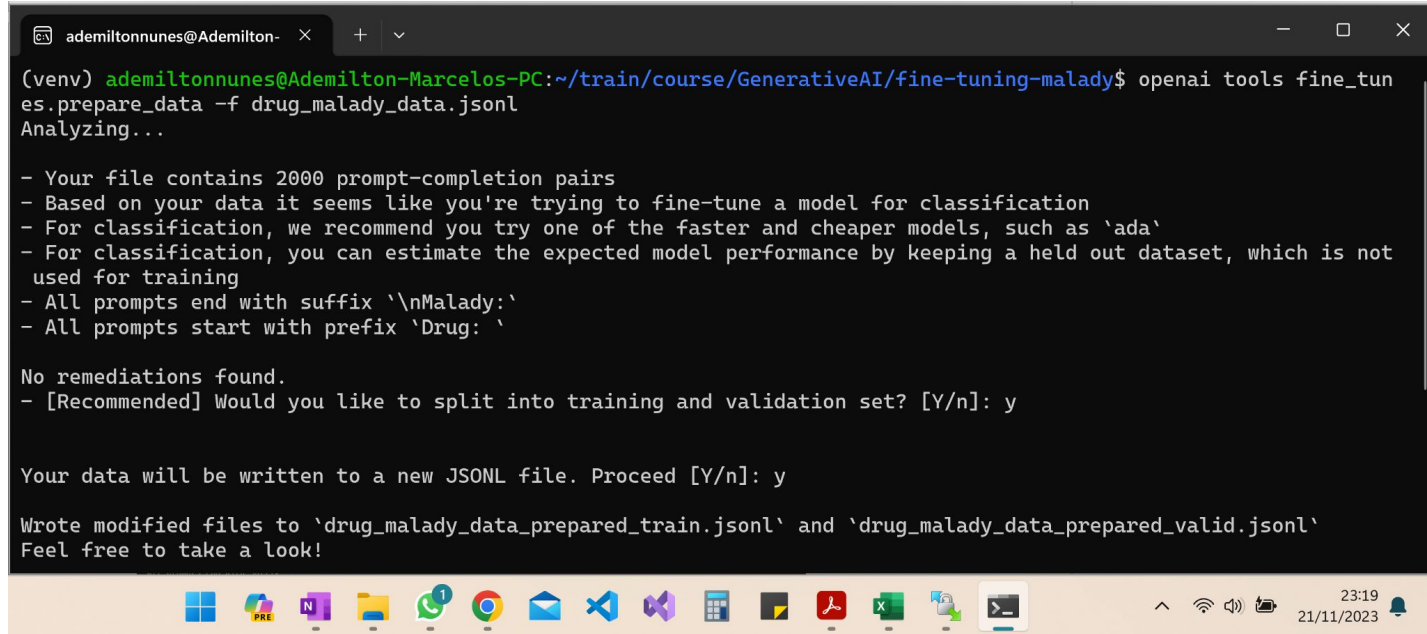


```
ademiltonnunes@Ademilton-PC: ~/train/course/GenerativeAI/fine-tuning$ export OPENAI_API_KEY=  
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning$
```

The screenshot shows a Windows terminal window with a dark background. The title bar at the top reads 'ademiltonnunes@Ademilton-'. The terminal content shows a user in a virtual environment (venv) at the directory ~/train/course/GenerativeAI/fine-tuning. The user has entered the command 'export OPENAI\_API\_KEY=' followed by a redacted API key (indicated by a red bar). The prompt returns to the shell. The Windows taskbar is visible at the bottom, showing various application icons and the system clock indicating 17:57 on 20/11/2023.

# Analyze and training the jsonl file

I analyzed and trained the jsonl file before being introduced to fine-tuning.



```
ademitonnunes@Ademilton-PC:~/train/course/GenerativeAI/fine-tuning-malady$ openai tools fine_tunes.prepare_data -f drug_malady_data.jsonl
Analyzing...

- Your file contains 2000 prompt-completion pairs
- Based on your data it seems like you're trying to fine-tune a model for classification
- For classification, we recommend you try one of the faster and cheaper models, such as 'ada'
- For classification, you can estimate the expected model performance by keeping a held out dataset, which is not used for training
- All prompts end with suffix '\nMalady:'
- All prompts start with prefix 'Drug: '

No remediations found.
- [Recommended] Would you like to split into training and validation set? [Y/n]: y

Your data will be written to a new JSONL file. Proceed [Y/n]: y

Wrote modified files to 'drug_malady_data_prepared_train.jsonl' and 'drug_malady_data_prepared_valid.jsonl'
Feel free to take a look!
```

# Analyze and training the jsonl file

The training process generated another jsonl file:

drug\_malady\_data\_prepared\_valid.jsonl

```

(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$ ls
drug_malady_data.jsonl  drug_malady_data_prepared_train.jsonl  drug_malady_data_prepared_valid.jsonl  venv
(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$ cat drug_malady_data_prepared_valid.jsonl
{"prompt":"Drug: A Ret 0.05% Gel 20gmA Ret 0.1% Gel 20gmA Ret 0.025% Gel 20gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acne UV Gel 60gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acnehit Gel 15gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acnelak Soap 75gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acnesol 1% Solution 25mlAcnesol Gel 20gmAcnesol Solution 45ml\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acnesol A Nano Gel 15gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acnetoin Plus Ointment 15gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acnil Soap 75gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acno 0.05% Gel 15gmAcno 20mg Capsule 10'S\nMalady:","completion":"" 0"}
{"prompt":"Drug: Acnovate Gel 15gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Adistar Gel 15gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Alene Gel 15gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Bengel Acra 2.5% Gel 30gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Benzonex 5% Gel 20gmBenzonex Gel 20gm\nMalady:","completion":"" 0"}
{"prompt":"Drug: Botanica 3D Serum 30ml\nMalady:","completion":"" 0"}
{"prompt":"Drug: Brite Opti Cream 10gm\nMalady:","completion":"" 0"}

```

# Train the model - fine-tuning

We train the model by the command:

```
openai api fine_tunes.create -t "drug_malady_data_prepared_train.jsonl" -v "drug_malady_data_prepared_valid.jsonl"  
--compute_classification_metrics --classification_n_classes 7 -m ada --suffix "drug_malady_data"
```

Explanation:

- `openai api fine_tunes.create`: initiates the process of creating a fine-tuned
- `t "drug_malady_data_prepared_train.jsonl"`: Specifies the training data file,
- `v "drug_malady_data_prepared_valid.jsonl"`: This specifies the validation data file, which is used to evaluate the performance of the model during training.
- `compute_classification_metrics`: This option indicates that the training process should compute classification metrics. This typically includes metrics such as accuracy, precision, recall, and F1 score, which are used to assess the model's performance.

# Train the model - fine-tuning

- `classification_n_classes 7`: This sets the number of classes for the classification task to 3. This suggests that the model is being trained for a classification problem with three distinct classes.
- `m ada`: This specifies the base model to use for fine-tuning. In this case, it's "ada."
- `suffix "drug_malady_data"`: This adds a suffix to the model name, possibly for easier identification.

# Train the model - fine-tuning

```
ademiltonnunes@Ademilton-PC: ~/train/course/GenerativeAI/fine-tuning-malady$ openai api fine_tunes.create -t "drug_malady_data_prepared_train.jsonl" -v "drug_malady_data_prepared_valid.jsonl" --compute_classification_metrics --classification_n_classes 7 -m ada --suffix "drug_malady_data"
Found potentially duplicated files with name 'drug_malady_data_prepared_train.jsonl', purpose 'fine-tune' and size 128249 bytes
file-bV30XY2PtAAgJetasH4eEDmo
Enter file ID to reuse an already uploaded file, or an empty string to upload this file anyway:
Upload progress: 100%|██████████| 128k/128k [00:00<00:00, 323Mit/s]Uploaded file from drug_malady_data_prepared_train.jsonl: file-U3ItVITSLV6bpSgQtcfZW5Uy
Found potentially duplicated files with name 'drug_malady_data_prepared_valid.jsonl', purpose 'fine-tune' and size 32007 bytes
file-zqcmHd6nWMni29CxnKp1lSbP
Enter file ID to reuse an already uploaded file, or an empty string to upload this file anyway:
Upload progress: 100%|██████████| 32.0k/32.0k [00:00<00:00, 32.0Mit/s]Uploaded file from drug_malady_data_prepared_valid.jsonl: file-KHdGqjMFwSQXx0952Qo67mqM
Created fine-tune: ft-ke1XFvp6FtkKZZ5o2wFd8t5t
Streaming events until fine-tuning is complete...

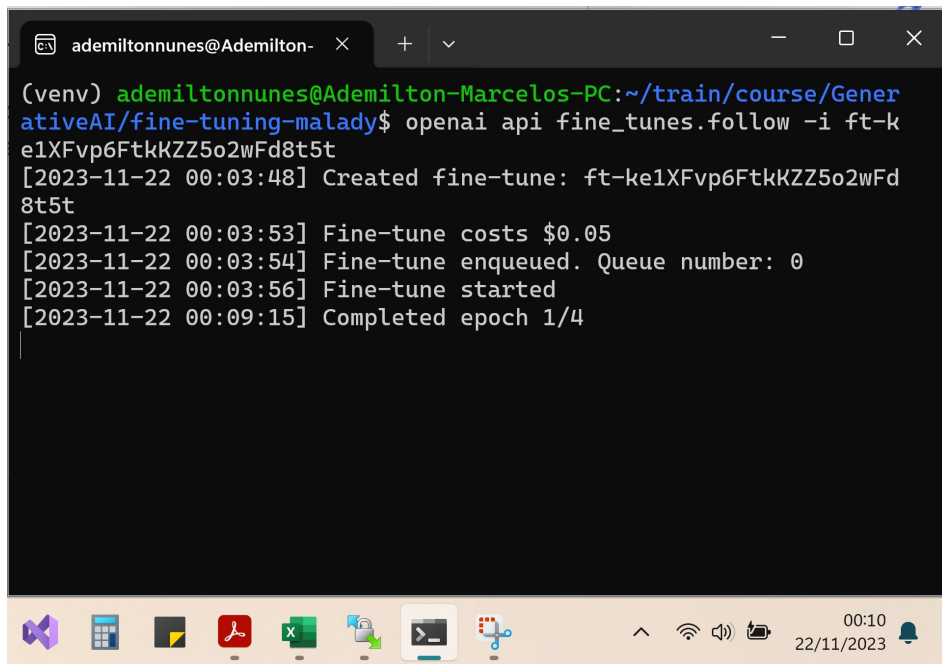
(CTRL-C will interrupt the stream, but not cancel the fine-tune)[2023-11-22 00:03:48] Created fine-tune: ft-ke1XFvp6FtkKZZ5o2wFd8t5t

[2023-11-22 00:03:53] Fine-tune costs $0.05
[2023-11-22 00:03:54] Fine-tune enqueued. Queue number: 0
[2023-11-22 00:03:56] Fine-tune started

(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$
(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$
```

# Checking Job Progress

The fine-tuning process can be time-consuming, so we can check the Job Progress of creating this.

A terminal window with a dark background and light-colored text. The window title bar shows 'ademiltonnunes@Ademilton-'. The prompt is '(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady\$'. The command entered is 'openai api fine\_tunes.follow -i ft-ke1XFvp6FtkKZZ5o2wFd8t5t'. The output shows a series of status updates from the OpenAI API, including the creation of the fine-tune, its cost, being enqueued, starting, and completing epoch 1/4.

```
ademiltonnunes@Ademilton-PC:~/train/course/GenerativeAI/fine-tuning-malady$ openai api fine_tunes.follow -i ft-ke1XFvp6FtkKZZ5o2wFd8t5t
[2023-11-22 00:03:48] Created fine-tune: ft-ke1XFvp6FtkKZZ5o2wFd8t5t
[2023-11-22 00:03:53] Fine-tune costs $0.05
[2023-11-22 00:03:54] Fine-tune enqueued. Queue number: 0
[2023-11-22 00:03:56] Fine-tune started
[2023-11-22 00:09:15] Completed epoch 1/4
```

# List fine-tuning models

By the command: `openai api fine_tunes.list`, we list all fine-tuning models created. In our example, the model name is

```
"validation_files": [  
  {  
    "object": "file",  
    "id": "file-KHdGqjMFwSQXx0952Qo67mqM",  
    "purpose": "fine-tune",  
    "filename": "drug_malady_data_prepared_valid.jsonl",  
    "bytes": 32007,  
    "created_at": 1700640228,  
    "status": "processed",  
    "status_details": null  
  }  
],  
"result_files": [],  
"created_at": 1700640228,  
"updated_at": 1700640555,  
"status": "running",
```



00:15  
22/11/2023





# List fine-tuning models

```
] ,  
  "created_at": 1700641437,  
  "updated_at": 1700642809,  
  "status": "succeeded",  
  "fine_tuned_model": "ada:ft-personal:drug-malady-data-2023  
-11-22-08-46-47"  
}  
],  
  "next_starting_after": null  
}  
(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/Gener  
ativeAI/fine-tuning-malady$ |
```



# Completion of Fine-Tuning

With the fine-tuning model created, we can make completions by the command:

```
“openai api completions.create -m <MODEL ID> -p <YOUR_PROMPT>”
```

Where:

- -m: fine-tuning model
- -p: prompt

# Completion of Fine-Tuning

```
(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$ openai api completions.create -m ada:ft-personal:drug-malady-data-2023-11-22-08-46-47 -p "AP Gel 10gm"
AP Gel 10gm
Malady 0 0 0 0

0 0 0 0 0 0 0(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/fine-tuning-malady$
```



# Testing the Fine Tuned Model Completion - Python

I implemented a Python test that test 3 medicines/drugs options:

- "A CN Gel(Topical) 20gmA CN Soap 75gm", # Class 0
- "Addnok Tablet 20'S", # Class 1
- "ABICET M Tablet 10's", # Class 2

# Testing the Fine Tuned Model Completion - Python

```
# Configure the model ID. Change this to your model ID.
model = "ada:ft-personal:drug-malady-data-2023-11-22-08-46-47"

# Returns a drug class for each drug
for drug_name in drugs:
    prompt = "Drug: {}\nMalady:".format(drug_name)

    response = openai.Completion.create(
        model=model,
        prompt=prompt,
        temperature=1,
        max_tokens=1,
    )

    # Print the generated text
    drug_class = response.choices[0].text
    # The result should be 0, 1, and 2
    print(f"Drug Name:{drug_name}, class: {drug_class}")
```

0

✓ 0s completed at 8:16 PM



# Testing the Fine Tuned Model Completion - Python

## Result

```
Drug Name:A CN Gel(Topical) 20gmA CN Soap 75gm, class: 0
```

```
Drug Name:Addnok Tablet 20'S, class: 1
```

```
Drug Name:ABICET M Tablet 10's, class: 2
```

✓ 0s completed at 8:22 PM



20:22  
22/11/2023



# Conclusion

In summary, this project has been a thorough exploration of fine-tuning a model, illuminating the intricacies involved in the process. From meticulously preparing the data, training the model, to creating and monitoring the fine-tuning process, each step contributed to a comprehensive understanding.

The initial data preparation involved reading from an Excel file, transforming it, and saving it in a jsonl format. The heart of the project revolved around model training, including validation to ensure effective learning. The creation and tracking of the fine-tuning process allowed for a nuanced understanding of model evolution.

The successful completion of fine-tuning resulted in a model capable of providing accurate answers. Demonstrations via command line interfaces and Python examples illustrated the real-world utility of the fine-tuned model. This project not only showcased technical prowess but also emphasized the importance of a systematic approach, demystifying the process of fine-tuning a model for those navigating the complexities of machine learning.