How to fine-tune OpenAI GPT

Marcelo X · Follow
7 min read · Mar 15

Listen Share

Introduction

ChatGPT is an artificial intelligence chatbot developed by OpenAI in which you can have a conversation about any subject and you are answered with human-like coherent responses. It is built on top of a large language model trained on massive amounts of text data from all over the internet and several other sources. It was launched on November 30, 2022 and drew attention for its coherent, correct and articulate answers across many domains of knowledge.

Besides the main functionality of answering questions with coherent responses, OpenAI provides a way to customize their model to adjust the answers to fit a specific case. This feature is called fine-tuning and allows anyone to make use of GPT-3 technology to train a personalized model. This results in a model that answers questions that fits in your scenario based on the examples you provided. With a personalized model the quality of the responses related to your topic will improve and the latency of the requests will decrease.

This article provides you with the step by step on how to fine-tune the OpenAI GPT-3 model and have a trained model to fit your specific scenarios. Some example scenarios you can apply the fine tuning process are:

- Create a personalized chatbot for automatically answering frequently asked questions (FAQ) from your customers about your company's services
- Compilation of knowledge base documents from your company so any employee will be able to find information easily, faster and using natural language
- Summarization of text documents, discussion, notes or meetings transcriptions in succinct abstract with the important topics
- Programming code generation based on your company's code base
- And many more other cases...

The fine-tuning process

Gather and sanitize your dataset

First of all you need to gather a dataset with hundreds or thousands of examples that represents your scenario. You can use tools to help you to format these data like Apache Spark or any other MapReduce tool or framework. The dataset can be saved in the following format: **CSV**, **TSV**, **XLSX**, **JSON** or **JSONL**, and must have only two properties: "prompt" and "completion". You can see an example in CSV format bellow on how your dataset should be structured:

```
prompt,completion
How to create an Order?,To create an order you must do 5 things:…
How to create a new product?,To create a new product you must first…
```

```
How to create a promotion campaign?,To create a promotion campaign
...
```

When you have the data together, sanitized and in one of the supported formats, then you are ready to start the fine-tuning process.

Prerequisites

First of all you have to install the *OpenAI* Python library using *pip*. Run the following command to do so (ref link):

```
pip install - upgrade openai
```

Then you have to set the OpenAI API key to allow you to send commands to OpenAI on behalf of your account. You can find the api key for your account in the api-keys page. Run the follwing line to set environment variable to authorise commands to OpenAI:

```
export OPENAI_API_KEY=<your_key>
```

Format dataset

OpenAI fine-tuning process accepts only files in JSONL format, but don't worry, it provides a tool to transform any format mentioned above to JSONL. Just run the following command to format dataset to JSONL:

```
openai tools fine_tunes.prepare_data -f dataset.csv
```

If the command fails because it's missing some dependency run the following command to fix it and try again (click here for more details). Command to install required dependency to run openai package:

```
pip install OpenAI[dependency]
```

The command will analyse your dataset and will show you a list of sentences with important information about your dataset and some options for the fine tuning. Read it carefully! Then, based on the analysis it will suggest to perform some actions like, remove duplicates, add a suffix separator, add white space character at the beginning of completion, etc. I would suggest applying all recommended actions if there isn't any reason not to do so.

One important action to do is "*split the data into training and validation sets*" which will generate two files with a training set to be used for fine tuning (around ~95% of the examples) and a validation set to be used to assess the performance of the trained model (around ~5% of the examples). This action allows OpenAI to generate detailed results about the model performance later on.

Finally, you will have the two files with the dataset in the right format to be used in the fine tuning. Note that when the command finishes it will give you some tips and recommendations to run the fine tuning. For example, the suggested command to be used, like "*OpenAI api fine_tunes.create …*". Take note of that.

Run Fine-tuning

Now you have the data correctly formatted in JSONL format you are ready to run the fine tuning process for your custom model. Note that this is a paid functionality and the price varies based on the size of your dataset and the base model you use for the training (see pricing here).

You can run the suggested command from the previous command output or you can build the command yourself with options you would like to set. The command has several options you can set or change based on your needs (check it out fine-tune command details here). For example, the option "-m" you can set as "ada", "babbage", "curie" or "davinci", being one of the based models OpenAI provides having different advantages like performance, accuracy, price, etc. (see more about the models OpenAI provides).

Note that the command can change depending on your dataset values. For example, for classifier problems where each input in the prompt should be classified into one of the predefined classes you have to set the option " —

compute_classification_metrics". If there are only two classes you have to set ' — *classification_positive_class "class1"* '. If there are more classes you have to set " — *classification_n_classes 5*". See these examples below.

Fine tuning command for binary classification:

```
openai api fine_tunes.create -t "dataset_train.jsonl" -v "dataset_valid.jsonl"
 -m "ada" --compute_classification_metrics --classification_positive_class " tr
```

Fine tuning command for multiclass classification:

```
openai api fine_tunes.create -t "dataset_train.jsonl" -v "dataset_valid.jsonl"
 -m "ada" --compute_classification_metrics --classification_n_classes 5
```

When you execute the command it will upload the dataset file(s) to OpenAI servers and the command will prompt you with a fine tune id such as "*ft-FpsHdkc836TLMi53FidOabcF*", save this id. At this point the request is received by OpenAI servers and the fine tuning is queued to be executed. The fine tuning execution would take some minutes or many hours depending on your dataset size and the model you chose. You don't need to wait for the command to finish, you can kill the process (ctrl-c) without hesitation and come back later to check the progress of the job. You can check the status of your process by running the following command with the id you received previously:

```
openai api fine_tunes.get -i <FINE_TUNING_ID>
```

When the fine tuning is done you will see the status of your fine tune process as "*processed*" meaning your custom model training is done and it's ready to be used. Note that to use your trained model you have to know its name which you can find in the property called "*fine_tuned_model*" (ex.: *"fine_tuned_model": "ada:ft-personal-2023–01–18–19–50–00"*).

To use your trained model when sending prompts to OpenAI just set the "*-m*" option with the id of your model like showing bellow:

```
openai api completions.create -m <FINE_TUNED_MODEL> -p "Your prompt here!"
```

You can also use the created model in your account playground link address. Now feel free to play, check, test and assess your model to make sure it responds with the expected answer for your scenario!

Performance assess results (Bonus)

Remember we generated two files for the dataset, one for the training and the other for the validation, OpenAI used the first one to train the model and the second one to run some assessment procedures to measure some stats like, for example, accuracy and loss. You can see the stats using the following command:

```
openai api fine_tunes.results -i <FINE_TUNING_ID> > results.csv
```

```
curl https://api.OpenAI.com/v1/files/$RESULTS_FILE_ID/content \
  -H "Authorization: Bearer $OPENAI_API_KEY" > results.csv
```

The output data is in a CSV file by default and can be saved in a file for better visualisation. The file structure is based on steps done in the training process which is represented by each line of the csv file. Each line shows valuable information about the step like: how many examples it processed, how many tokens, training loss, training accuracy, validation loss, and much more. See more about results in this link.

Conclusion

OpenAI is bringing one of the most revolutionary technological disruptions of recent times with the creation of GPT-3 and ChatGPT. The best part is that anyone can have access to use this incredible tool for free. Besides that, OpenAI provides the necessary tools for anyone to create personalized models to attend any specific scenario. This process is called fine-tuning and it's easily available to anyone by

OpeanAI. Fine tuning provides access to the cutting-edge technology of machine learning that OpenAI used in GPT-3. This provides endless possibilities to improve computer human interaction for companies, their employees and their customers.

With fine tuning anyone can create customized GPT-3 models specific for your scenario. For example, create a chatbot to interact with customers in a conventional manner to answer specific questions about your company, your services and your products. Another example is to help employees to find valuable information quickly and easily in a large knowledge base with several text articles where it's hard and time consuming to find what you need. The scenarios involving a high volume of text it can help are countless.

If you have any question feel free to contact me:
https://www.linkedin.com/in/marceloax/

OpenAI Gpt 3 ChatGPT Machine Learning Fine Tuning

Written by Marcelo X

4 Followers

More from Marcelo X

Marcelo X

Fake-GPT: Fake news classifier with GPT-3

Fine tuning GPT-3 model to create a fake news classifier (Fake GPT) using cutting-edge technology of OpenAI.

8 min read · Apr 24

See all from Marcelo X

Recommended from Medium

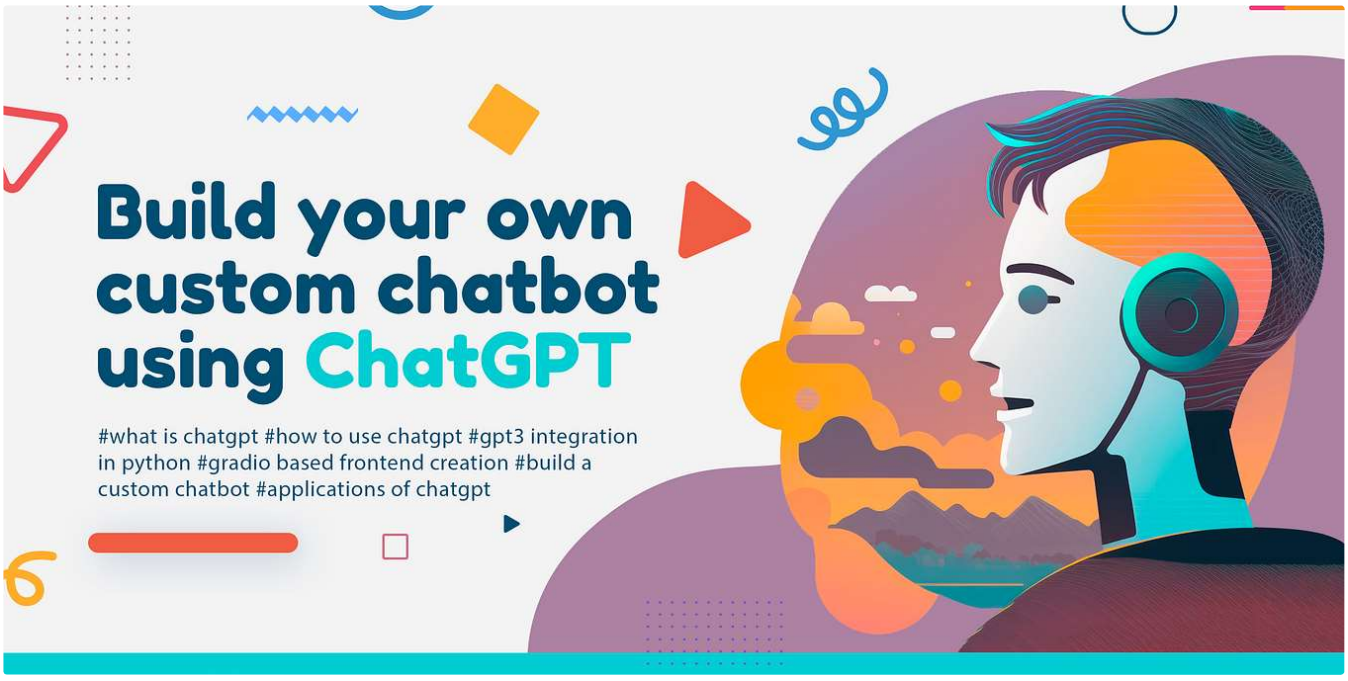
Sushwanth Nimmagadda

How to train ChatGPT with your custom data and create your own chatbot

With ChatGPT API's advent, you can now create your own AI-based simple chat app by training it with your custom data.

5 min read · Jun 2

13 4



Vishnu Sivan in Coinmonks

Build your custom chatbot using chatgpt

ChatGPT is the most powerful language model ever built. It is an amazing tool that can be used in various ways to enhance your productivity...

12 min read · Mar 26

55 2

Lists

The New Chatbots: ChatGPT, Bard, and Beyond

13 stories · 94 saves

Now in AI: Handpicked by Better Programming

266 stories · 106 saves

What is ChatGPT?

9 stories · 160 saves

Predictive Modeling w/ Python

20 stories · 310 saves

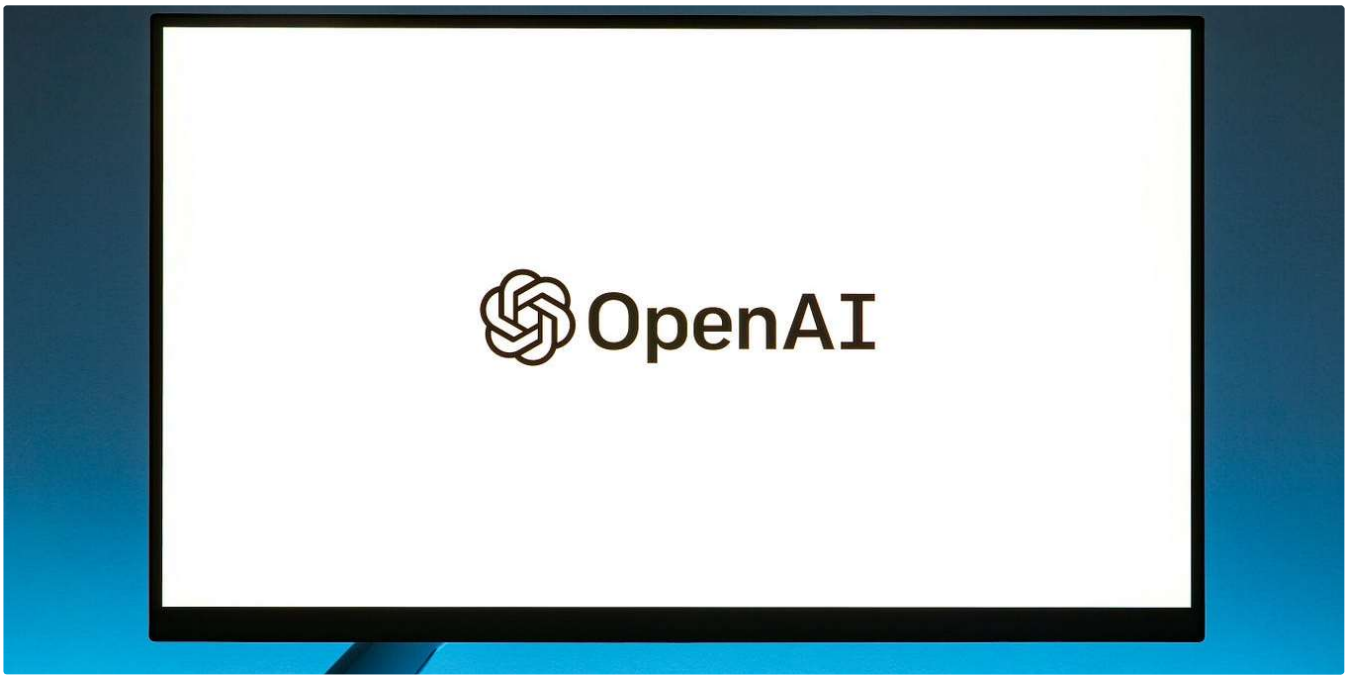
Maximilian Vogel in MLearning.ai

The ChatGPT list of lists: A collection of 3000+ prompts, examples, use-cases, tools, APIs...

Updated Aug 20, 2023. Added prompt design courses, masterclasses and tutorials.

10 min read · Feb 8

7.1K 75



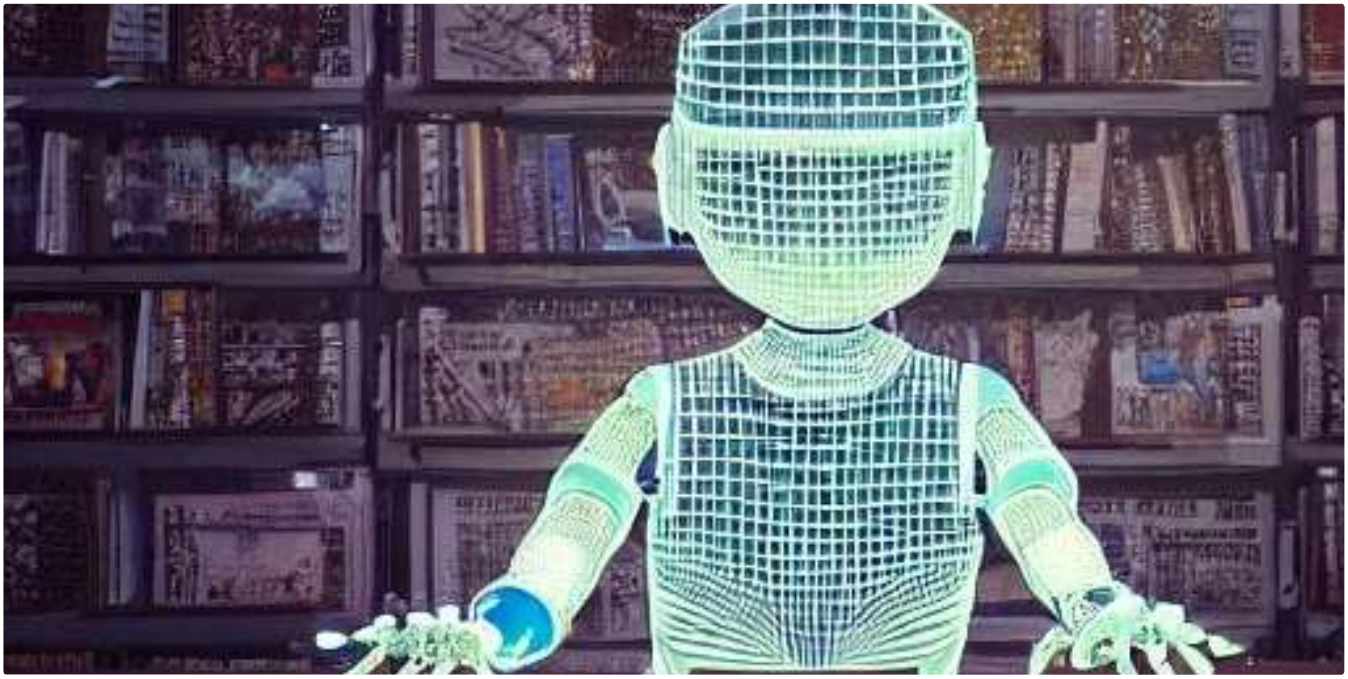
Paula Ceccon Ribeiro in GoPenAI

GPT-3: Lessons from Fine-Tuning

Step-by-step guide for GPT-3 fine-tuning

· 7 min read · May 16

28 1



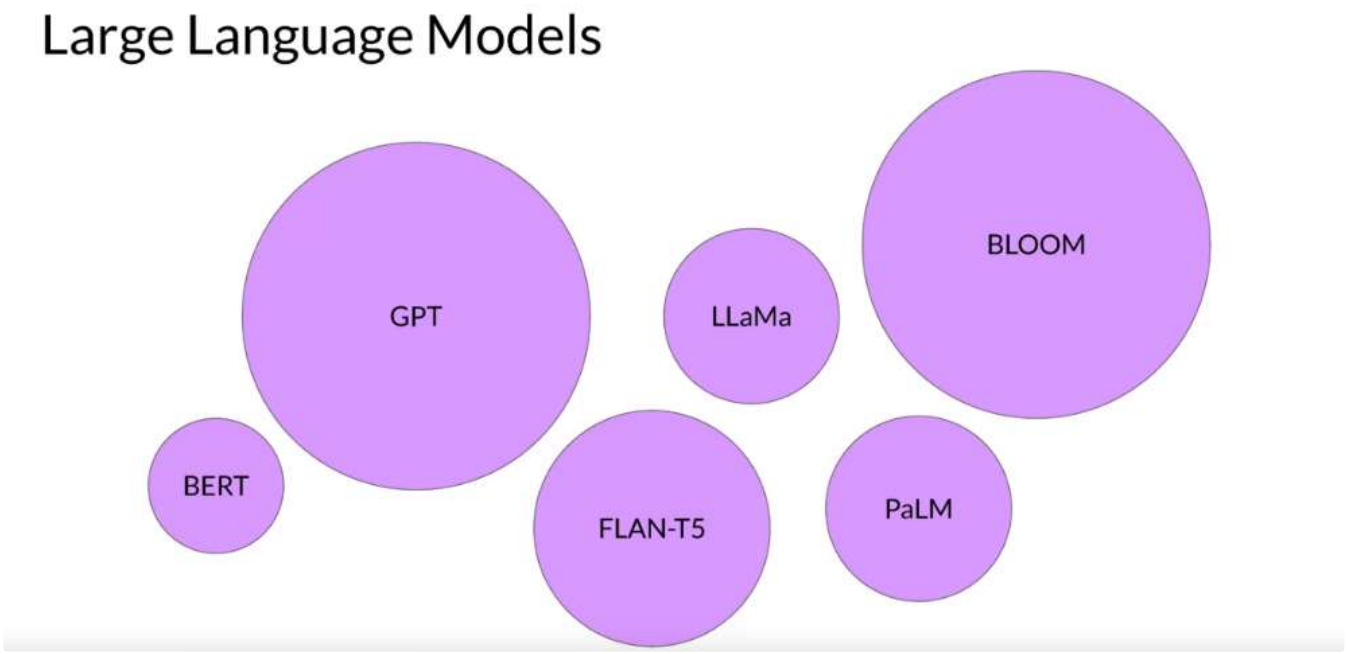
Mohit Soni

How to use Chat GPT and LangChain to get responses from your documents

In the previous post, I explained methods that you can use to train Chat GPT on your data. In this post, we will use one of those methods...

4 min read · Jun 17

6



Yash Bhaskar

Introduction to LLMs and the generative AI : Part 1- LLM Architecture, Prompt Engineering and LLM...

Large language models (LLMs) have revolutionized the field of artificial intelligence (AI) development, offering developers unprecedented...

11 min read · Jul 16

113

See more recommendations