ITCS 4111/5111: Introduction to NLP

Working with Large Language Models: GPT, Llama-2, Mixtral

Razvan C. Bunescu

Department of Computer Science @ CCI

rbunescu@uncc.edu

OpenAI GPT models

- GPT = Generative Pre-trained Transformer.
- **Pre-trained** to "understand" natural language and code:
  - Using a language modeling (LM) objective.
- **Fine-tuned** to provide text outputs (answers) in response to their inputs (questions or **prompts**).
  - Instruction-based fine-tuning.
  - Reinforcement Learning through Human Feedback (RLHF).
- "Programming" with GPT, Llama-2, and other LMs:
  - Design a "prompt", usually by providing **instructions** or some examples of how to successfully complete a task (*zero-shot*, *few-shot* / *in-context learning, CoT*).

Three Options for Using the Chat API

1. OpenAI's ChatGPT Models (gpt-3.5-turbo and gpt-4):
   - Pay per intput and output token, see pricing.
   - Through the ChatGPT browser app.
   - Through the Chat completions API.
     - **Directly through Python** code or through the Playground.

2. Meta's Llama-2 model:
   - Free to use, quantized 70B version installed by Erfan on our HPC server.
   - Through an API endpoint that can accommodate ~ 40 concurrent requests. Link will be sent on Canvas.
   - Do not distribute.

Three Options for Using the Chat API

3. Mistrals's Mixtral-8x7B model:
   - Free to use, Mixtyure of Experts (MoE), 8 experts, 7B each.
   - Installed on CCI's HPC server by SIS.
   - Through an API endpoint that can accommodate multiple concurrent requests. Link will be sent on Canvas.
   - Do not distribute.

GPT, Llama2, Mixtral all use the same chat completion API.

GPT: Setting up the OpenAI API account

- Need to have an OpenAI account:
  - Same account for ChatGPT and the API.
    - Go to https://platform.openai.com, Log in / Signup.
    - **"Continue with Google", use your UNCC email**.
      - $5 is more than enough for the work in this class.
      - Go to billing overview, *Set payment* → input credit card, or *Add to credit balance*, input $5.
      - Go to billing history, *View* → *Download receipt*.

GPT: One-time Setup of API Key

- Create and store a secret API key:
  - Go to API keys and "+ *create new secret key*".
  - Copy the key and store it in a text file named **.env** as follows
    - OPENAI_API_KEY=…
  - Make sure you save the key, it will not be shown again.
- Place or copy the **.env** file in the folder you edit and run the notebook.
  - Other solutions exist, but this is what we will do in this course.
  - Do not put the secret key in your code!

Required Python Modules

- Install the **openai** module:
  - pip install openai (use pip3)
- Install the **python-dotenv** module, using one of:
  - pip install python-dotenv
  - conda install -c auto python-dotenv
- Alternatively, use Colab instead of Jupyter:
  - Has modules already installed.
  - But ensure the .env file is placed in the right Drive folder.

Chat Completion API: openai.ChatCompletion.create

- Chat models take a list of messages as input and return a model-generated message as output.
  - Designed to make multi-turn conversations easy, it's just as useful for single-turn tasks without any conversation.

```
from openai import OpenAI

client = OpenAI(api_key = os.environ['OPENAI_API_KEY'])
```

```
response =  client.chat.completions.create(
    model = "gpt-3.5-turbo",
    messages = [
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Who composed The Four Seasons?"},
        {"role": "assistant", "content": "Antonio Vivaldi composed The Four Seasons."},
        {"role": "user", "content": "For whom were most of his compositions written?"}
    ]
)

print(response.choices[0].message.content)
```

https://platform.openai.com/docs/guides/gpt
https://platform.openai.com/docs/api-reference/chat/create

Chat Completion API: openai.ChatCompletion.create

- 3 major roles in the **messages** parameter:
  - **System**: Optional first message, that indicates the LM persona.
    - Also called a *steering promp*, sets up the system behavior.
    - Default is "You are a helpful assistant".
  - **User**: Provides questions, requests, or comments to the assistant.
  - **Assistant**: Previous responses from the LM assistant, or example of desired LM response.
    - **Need to provide the conversation so far every time** we want to continue with a new user questions.
- Typical input (RE) is **system? user (assistant user)***

Chat Completion API: openai.ChatCompletion.create

- Other useful parameters:
  - **model**: gpt-3.5-turbo or gpt-4.
  - **temperature**: defaults to 1, but set it to **0 for greedy decoding**.
  - **top_p**: defaults to 1, use 0.1 if you want the LM to sample tokens only from the top 10% of probability mass, i.e. **nucleus sampling**.
  - **n** : defaults to 1, indicates # completions (alternatives) to generate.
  - **max_tokens**: defaults to ∞, maximum # of tokens to generate.
  - presence_penalty, frequence_penalty, logit_bias: penalize or favor repetitions, or certain tokens (later in this course).

Llama2: Chat Completion API

- **OpenAI.base_url**:
  - An attribute of the OpenAI class.
- **Model name**:
  - Specifies which version of Llama-2 is being utilized.
  - You must be on Eduroam to access the model directly. Off campus, you need connect through the educational cluster using VPN.

```
from openai import OpenAI

client = OpenAI(api_key = "aewndfoa1235123")

# Set the Llama API base URL
client.base_url = "http://cci-liqidnode1.uncc.edu/api/v1/"

model_name = "/mnt/llama/hf_models/TheBloke_Llama-2-70B-Chat-GPTQ"
```

Mixtral: Chat Completion API

- **OpenAI.base_url**:
  - An attribute of the OpenAI class.
- **Model name**:
  - Specifies which version of Mixtral is being utilized.
  - You must be on Eduroam to access the model directly. Off campus, you need connect through the educational cluster using VPN.

```
from openai import OpenAI

client = OpenAI(api_key = 'EMPTY')

# Set the Mixtral API base URL
client.base_url = "http://cci-siscluster1.charlotte.edu:5000/v1"

# Name of the Mixtral model being used.
model_name = "TheBloke/Mixtral-8x7B-v0.1-GGUF"
```

Examples with Open AI GPT and Llama-2

- Examples and homework refer to an HPC server with 6 V100 NVIDIA GPUs.
- We have also installed Llama2 and Mixtral on a faster HPC server with A5000 GPUs.

```
from openai import OpenAI

client = OpenAI(api_key = "OnuR-l5IlfYqF8HYoTOYHAcHOXCgL5xASQM5ooGHG6A")

# Set the OpenAI API base URL
client.base_url = "http://cci-llama1.charlotte.edu/api/v1"

model_name = "Llama-2-70B"
```

Examples with Open AI GPT and Llama-2

- Shown in the Jupyter notebook.

Supplementary Activities

- Take the Building Systems with the ChatGPT API short course (1 hour) from deeplearning.ai.