L1 Language Models, the Chat Format and Tokens

Setup

Load the API key and relevant Python libaries.

In this course, we've provided some code that loads the OpenAI API key for you.

In [33]:
```
import os
import openai
import tiktoken
from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file

openai.api_key  = os.environ['OPENAI_API_KEY']
```

helper function

This may look familiar if you took the earlier course "ChatGPT Prompt Engineering for Developers" Course

In [34]:
```
def get_completion(prompt, model="gpt-3.5-turbo"):
    messages = [{"role": "user", "content": prompt}]
    response = openai.ChatCompletion.create(
        model=model,
        messages=messages,
        temperature=0,
    )
    return response.choices[0].message["content"]
```

Prompt the model and get a completion

In [35]:
```
response = get_completion("What is the capital of France?")
```

In [36]:
```
print(response)
```

```
The capital of France is Paris.
```

Tokens

In [37]:
```
response = get_completion("Take the letters in lollipop \
and reverse them")
print(response)
```

```
ppilolol
```

"lollipop" in reverse should be "popillol"

In [38]:
```
response = get_completion("""Take the letters in \
l-o-l-l-i-p-o-p and reverse them""")
print(response)
```

```
p-o-p-i-l-l-o-l
```

In [39]:
```
print(response)
```

```
p-o-p-i-l-l-o-l
```

Helper function (chat format)

Here's the helper function we'll use in this course.

In [*]:
```
def get_completion_from_messages(messages,
                                 model="gpt-3.5-turbo",
                                 temperature=0,
                                 max_tokens=500):
    response = openai.ChatCompletion.create(
        model=model,
        messages=messages,
        temperature=temperature, # this is the degree of randomness of the m
        max_tokens=max_tokens, # the maximum number of tokens the model can
    )
    return response.choices[0].message["content"]
```

In [42]:
```
messages =  [
{'role':'system',
 'content':"""You are an assistant who\
 responds in the style of Dr Seuss."""},
{'role':'user',
 'content':"""write me a very short poem\
 about a happy carrot"""},
]
response = get_completion_from_messages(messages, temperature=1)
print(response)
```

```
Oh, happy carrot with a bright orange hue,
So crispy and delicious, it's true.
With every bite, you make us smile,
Eating veggies has never been so worthwhile!
```

In [43]:
```
# length
messages =  [
{'role':'system',
 'content':'All your responses must be \
one sentence long.'},
{'role':'user',
 'content':'write me a story about a happy carrot'},
]
response = get_completion_from_messages(messages, temperature =1)
print(response)
```

```
Once upon a time, there was a happy carrot named Carl who grew in a warm and
sunny garden, and he often enjoyed chatting with the friendly ladybugs and wi
ggly earthworms who lived alongside him.
```

In [44]:
```
# combined
messages =  [
{'role':'system',
 'content':"""You are an assistant who \
responds in the style of Dr Seuss. \
All your responses must be one sentence long."""},
{'role':'user',
 'content':"""write me a story about a happy carrot"""},
]
response = get_completion_from_messages(messages,
                                        temperature =1)
print(response)
```

```
In the garden grew a happy carrot, bright orange and oh so fair-like-a-prince
ss, who smiled and sang all day long, until she was plucked from the ground w
ith care and turned into a crunchy snack that made everyone's day much bette
r.
```

In [45]:
```
def get_completion_and_token_count(messages,
                                   model="gpt-3.5-turbo",
                                   temperature=0,
                                   max_tokens=500):

    response = openai.ChatCompletion.create(
        model=model,
        messages=messages,
        temperature=temperature,
        max_tokens=max_tokens,
    )

    content = response.choices[0].message["content"]

    token_dict = {
'prompt_tokens':response['usage']['prompt_tokens'],
'completion_tokens':response['usage']['completion_tokens'],
'total_tokens':response['usage']['total_tokens'],
    }

    return content, token_dict
```

In [46]:
```
messages = [
{'role':'system',
 'content':"""You are an assistant who responds\
 in the style of Dr Seuss."""},
{'role':'user',
 'content':"""write me a very short poem \
 about a happy carrot"""},
]
response, token_dict = get_completion_and_token_count(messages)
```

In [47]:
```
print(response)
```

```
Oh, the happy carrot, so bright and so bold,
With a smile on its face, and a story untold.
It grew in the garden, with sun and with rain,
And now it's so happy, it can't help but exclaim!
```

In [48]:
```
print(token_dict)
```

```
{'prompt_tokens': 39, 'completion_tokens': 52, 'total_tokens': 91}
```

Notes on using the OpenAI API outside of this classroom

To install the OpenAI Python library:

```
!pip install openai
```

The library needs to be configured with your account's secret key, which is available on the website (https://platform.openai.com/account/api-keys).

You can either set it as the `OPENAI_API_KEY` environment variable before using the library:

```
!export OPENAI_API_KEY='sk-...'
```

Or, set `openai.api_key` to its value:

```
import openai
openai.api_key = "sk-..."
```

A note about the backslash

- In the course, we are using a backslash `\` to make the text fit on the screen without inserting newline '\n' characters.
- GPT-3 isn't really affected whether you insert newline characters or not. But when working with LLMs in general, you may consider whether newline characters in your prompt may affect the model's performance