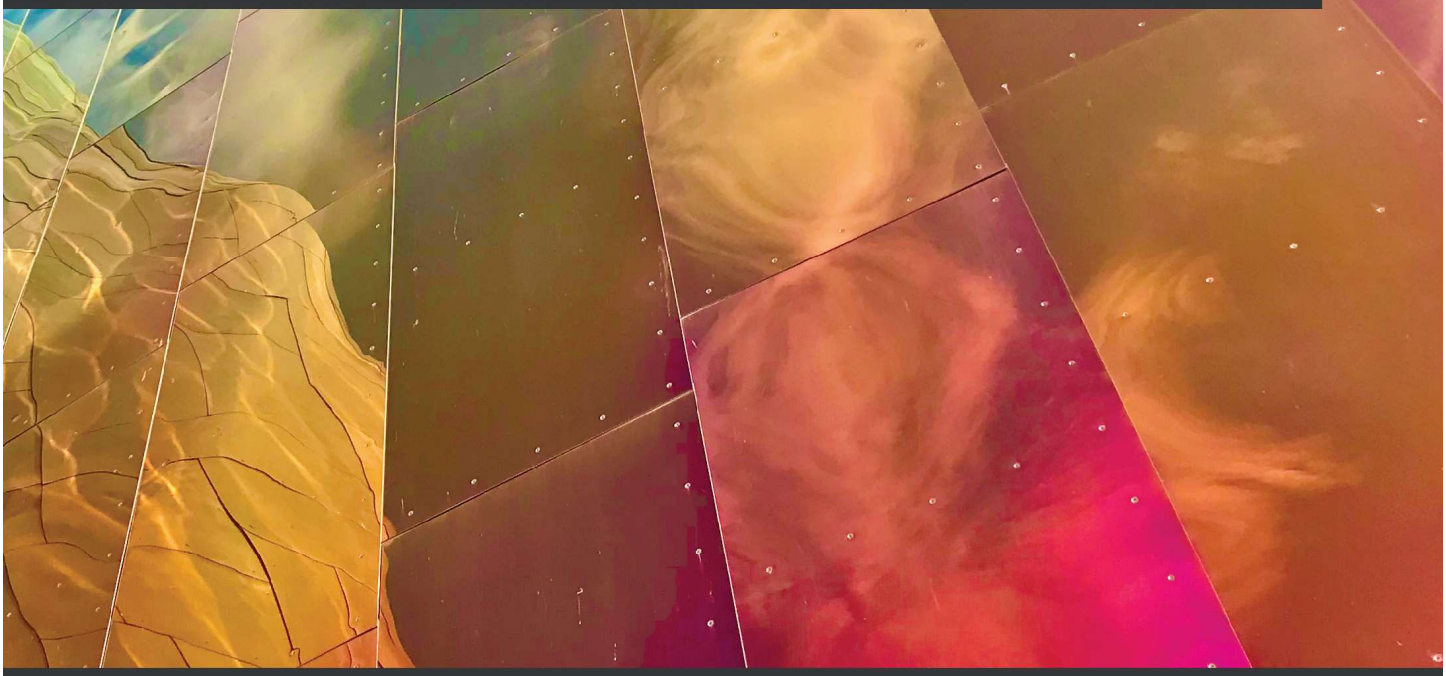
Exploring GPT-3

An unofficial first look at the general-purpose language processing API from OpenAI



Steve Tingiris

Foreword by Bret Kinsella (Founder and CEO of Voicebot.ai)

Exploring GPT-3

An unofficial first look at the general-purpose language processing API from OpenAI

Steve Tingiris

Packt

BIRMINGHAM—MUMBAI

Exploring GPT-3

Group Product Manager: Kunal Parikh

Publishing Product Manager: Sunith Shetty

Senior Editor: David Sugarman

Content Development Editor: Nathanya Dias

Technical Editor: Devanshi Ayare

Copy Editor: Safis Editing

Project Coordinator: Aparna Ravikumar Nair

Proofreader: Safis Editing

Indexer: Rekha Nair

Production Designer: Alishon Mendonca

First published: July 2021

Production reference: 1100621

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham

B3 2PB, UK.

ISBN 978-1-80056-319-3

www.packt.com

To my wife, Brigid, for putting up with my constant dabbling for all these years. To my son, Alex, for motivating me to push a little harder. There is no way I would have finished this book without you guys. Thank you!

– Steve

Foreword

What's all the fuss about? Maybe it's the demos. But that would be missing the point. GPT-3 burst into public consciousness in July 2020 not too long after OpenAI first introduced the technical breakthrough in a soberly presented blog post. Introduced in the middle of a global pandemic, rising social unrest, and a US presidential campaign, the message could have been lost entirely. The demos started trickling out in earnest a month later. They ensured that GPT-3 wouldn't be overlooked.

Technology Review claimed the GPT-3 was *shockingly good*. Many long-time AI researchers expressed both enthusiasm and surprise at its capabilities. But it was the demos that really captured everyone's attention. With a few lines of sample content and a request, the technology was generating poetry, website programming code, analogies, and answers to math questions, to name just a few examples. No one had ever seen a computer create and respond creatively to such a wide range of queries.

GPT-3 seemed to possess magical abilities. In many ways, GPT-3 is very simple. It predicts what words are the most likely to follow in a sequence. However, it is also the finest current example of the potential of **Generative Adversarial Networks** (**GANs**). And, it has shown that transformers can be applied to language models at an extremely large scale. GPT-3 provides an intriguing new technical capability while simultaneously resetting expectations about what is possible. As just one example, maybe chatbots don't have to choose between a set number of deterministic responses.

Natural Language Processing (**NLP**) has improved quickly in recent years. That has led to the more accurate recognition and understanding of speech. At the same time, synthetic speech engines have improved immensely and sound more humanlike each year. What has changed very little is how the systems respond to requests. They are all picking from a predetermined set of responses. GPT-3 offers a capability that enables developers to rethink that approach. But that requires developers to understand the new technology and how to use it.

Steve Tingiris is the first to take that task on with this book. With clear and precise presentation, Tingiris expertly walks new users through the journey from idea through production of a GPT-3 application. He lays out the principles and steps so developers can turn their ideas into (a new) reality. I look forward to seeing what you build.

Bret Kinsella

Founder and CEO of Voicebot.ai

Contributors

About the author

Steve Tingiris is the founder and managing director of Dabble Lab, a technology research and services company that helps businesses accelerate learning and adoption of natural language processing, conversational AI, and other emerging technologies. He has been designing and building automation solutions for over 20 years and has consulted on conversational AI projects for companies including Amazon, Google, and Twilio. He also publishes technical tutorials on Dabble Lab's YouTube channel— one of the most popular educational resources for conversational AI developers—and manages several open-source projects, including the Autopilot CLI, Twilio's recommended tool for building Autopilot bots. To connect with Steve, you can find him on `GitHub` `@tingiris`, via email to `steve@dabblelab.com`, or on Twitter `@tingiris`.

Acknowledgments

This book is the result of contributions from friends, colleagues, and many members of the OpenAI community. There are far too many people to mention everyone by name but for those who contributed directly or reviewed early drafts, I want to extend a special thank you.

First, thank you to the entire Packt team. If Sunith had not reached out with the idea of creating a book on GPT-3, this project would never have started. If David, Gebin, Nathanya, Aishwarya, Roshan, and Devanshi weren't involved – the book might never have gotten finished. Thanks to each of you and everyone else on the Packt team who made this possible.

Thanks to Russell, Bakz, Ryan, and Bram for agreeing to be technical reviewers. You guys were so helpful to me in the OpenAI Slack channel when I first got started with GPT-3, I feel so fortunate to have had the opportunity to collaborate together on this book. Again, thank you!

Next, thank you to my *work family* at Dabble Lab. Kirk Owen for providing early feedback that helped refine the direction, and Mohamad Khalid, Manuel Fernandez, Mark Hovsepyan, Sohini Pattanayak, Shubham Prakash, and Daniela Ramirez, for picking up the slack while I spent much more time than I'd anticipated working on the book.

Finally, I'd like to thank the OpenAI team and the OpenAI community for all of the support and feedback throughout the project, including: Ashley Pilipiszyn, Mark Clintsman, Minal Chhatbar, Rene Diaz, Dan Shaw, Chris Fong, Dariusz Gross, Cristi Vlad, Jonathon Sauceda, Marc-Andre Schenk, Steven Kuo, Narendran Thillaisthanam, Matthew Benites, Manav Goel, Shubham Amraotkar, Mystici Mentis, Fred Zimmerman, Dmitry Kearo, CL Kim, Sudip Lingthep, Joakim Flink, Shubham Saboo, Pedro Ribeiro, Richard Klein, Steve Hoyt, Nicolas Garrel, Sebastian Derewicki, Vikram Pandya, Geoff Davis, Nelson Pereira, Heng Gu, Joey Bertschler, Surendra Reddy, James Morgan, Jon Oakes, Jeetendra K Sharma, Jim Taylor, Rebecca Johnson, Travis Barton, Herber Scrap, Pablo del Ser, Devin Bean, Nik K, Jason Boog, Mohak Agarwal, Sebastian Elliott, and Bjarne Carstensen.

About the reviewers

Russell Foltz-Smith has 20+ years of experience in tech as a developer, business development leader, executive, and researcher. He maintains a focus on search engines, scientific computing, and media platforms. Russ advises tech start-ups, mentors entrepreneurs, and is CTO of Maslo.ai, an empathetic computing platform. Russ is a visual artist and educator. He co-founded a k-12 school with his wife in 2012 in Venice, CA, where they live with their daughters and endless stacks of books.

Bakz Awan is an IT consultant and YouTube host of the channel *Bakz T. Future.* on YouTube. Bakz shares new concepts and ideas possible through GPT-3, while also providing tips and advice to beginners.

Table of Contents

Preface

Section 1: Understanding GPT-3 and the OpenAI API

1

Introducing GPT-3 and the OpenAI API

Technical requirements 4
Introduction to GPT-3 4
Simplifying NLP 4
What exactly is GPT-3? 5
Democratizing NLP 6
Understanding prompts, completions, and tokens 6
Prompts 6
Completions 12
Tokens 12
Introducing Davinci, Babbage, Curie, and Ada 15
Davinci 16
Curie 16
Babbage 16
Ada 16
Content filtering model 16
Instruct models 16
A snapshot in time 17
Understanding GPT-3 risks 17
Inappropriate or offensive results 17
Potential for malicious use 17
Summary 18

2

GPT-3 Applications and Use Cases

Technical requirements 20
Understanding general GPT-3 use cases 20
Introducing the Playground 20
Getting started with the Playground 20
Handling text generation and classification tasks 22

Text generation 22
Text classification 32
Understanding semantic search 37
The Semantic Search tool 38
Summary 40

Section 2: Getting Started with GPT-3

3
Working with the OpenAI Playground

Technical requirements 43
Exploring the OpenAI developer console 44
Developer documentation 44
Developer resources 46
Accounts and organizations 46
Pricing and billing 48
Usage reporting 49
Member management 50
Diving deeper into the Playground 50
Choosing the right engine 51
Response length 53
Temperature and Top P 54
Frequency and presence penalty 59
Best of 60
Stop sequence 61
Inject Start Text and Inject Restart Text 61
Show Probabilities 63
Working with presets 64
Grammatical Standard English 65
Text to command 67
Parse unstructured data 68
Summary 68

4
Working with the OpenAI API

Technical requirements 70
Understanding APIs 70
Getting familiar with HTTP 71
Uniform resource identifiers 72
HTTP methods 72
The HTTP body 72
HTTP headers 73
HTTP response status codes 73
Reviewing the OpenAI API endpoints 73
List Engines 74
Retrieve Engine 74
Create Completions 75
Semantic Search 75
Introducing CURL and Postman 75

Understanding API authentication 80
Keeping API keys private 82
Making an authenticated request to the OpenAI API 83
Working with multiple organizations 86
Introducing JSON 89
Using the Completions endpoint 90
Using the Semantic Search endpoint 94
Summary 98

5

Calling the OpenAI API in Code

Technical requirements 100
Choosing your programming language 100
Introducing replit 100
Creating a repl 102
Setting your OpenAI API key as an environment variable 104
Understanding and creating the .replit file 106
Using the OpenAI API with Node.js/JavaScript 107
Calling the engines endpoint 107
Calling the Completions endpoint 110
Calling the search endpoint 113
Using the OpenAI API in Python 115
Calling the completions endpoint 118
Calling the search endpoint 120
Using other programming languages 122
Summary 123

Section 3: Using the OpenAI API

6

Content Filtering

Technical requirements 127
Preventing inappropriate and offensive results 128
Understanding content filtering 129
Testing the content filtering process 133
Filtering content with JavaScript 136
Flagging unsafe words with Node.js/JavaScript 140
Filtering content with Python 142
Flagging unsafe words with Python 146
Summary 148

7
Generating and Transforming Text

Technical requirements 150
Using the examples 150
Generating content and lists 150
Dumb joke generator 150
Mars facts (in most cases) 154
Webinar description generator 157
Book suggestions 160
Children's book generator 163
Translating and transforming text 166
Acronym translator 166
English to Spanish 169
JavaScript to Python 172
Fifth-grade summary 175
Grammar correction 180
Extracting text 183
Extracting keywords 183
HTML parsing 187
Extracting a postal address 190
Extracting an email address 194
Creating chatbots 197
A simple chatbot 197
Summary 200

8
Classifying and Categorizing Text

Technical requirements 202
Understanding text classification 202
Using the completions endpoint for text classification 202
Content filtering is a text classification task 202
Introducing the classifications endpoint 203
Uploading files 203
Implementing sentiment analysis 204
Assigning an ESRB rating to text 208
Classifying text by language 212
Classifying text from keywords 217
Summary 221

9
Building a GPT-3-Powered Question-Answering App

Technical requirements 224
Introducing GPT Answers 224
GPT Answers technical overview 225
Hosting the app 225
Introducing the Answers endpoint 225
Setting up and testing Express 227
Creating the API endpoint for

GPT Answers 229
Creating the API endpoint 230
Testing our API with Postman 232
Creating the GPT Answers user interface 233
Integrating the Answers endpoint 238
Generating relevant and factual answers 242
Using files with the Answers endpoint 243
Summary 249

10

Going Live with OpenAI-Powered Apps

Technical requirements 252
Going live 252
Understanding use case guidelines 252
Addressing potential approval issues 253
Content filtering 254
Input and output lengths 255
Request rate limiting 257
Completing the pre-launch review request 258
High-level use case questions 259
Security and risk mitigation questions 260
Growth plan questions 261
Wrapping-up questions 263
Summary 264
Why subscribe? 265

Other Books You May Enjoy

Index

Preface

What if this book was written by artificial intelligence? Would you read it? I hope so because parts of it were. Yes, GPT-3 was used to create parts of this book. It's a bit meta I know, a book about GPT-3 written by GPT-3. But creating content is one of the many great uses for GPT-3. So why not? Also, for me, content generation was the use case that most piqued my interest. I wondered if GPT-3 could be used in a product I was working on to automate the generation of technical learning material.

You probably also have a specific reason why you're interested in GPT-3. Perhaps it's intellectual curiosity. Or maybe you have an idea that you think GPT-3 can enable. You've likely seen online demos of GPT-3 generating content, writing code, penning poetry, or something else, and you're wondering if GPT-3 could be used for an idea you have. If so, this book was written specifically for you.

My goal for this book is to provide a practical resource to help you get started with GPT-3, as quickly as possible, without any required technical background. That said, as I write this, GPT-3 is still in private beta. So, everyone is learning as they go. But the one thing I've learned for sure is that the possible applications for GPT-3 are vast and there is no way to know all of what's possible, let alone get it into a book. So, I hope this book makes getting started easy, but I also hope it's just the beginning of your journey *Exploring GPT-3*

Who this book is for

This book was written for anyone with an interest in NLP or learning GPT-3 – with or without a technical background. Developers, product managers, entrepreneurs, and hobbyists who want to learn about NLP, AI, and GPT-3 will find this book useful. Basic computer skills are all you need to get the most out of the book. While experience with a modern programming language is helpful, it's not required. The code examples provided are beginner friendly and easy to follow, even if you're brand new to writing code.

What this book covers

Chapter 1, *Introducing GPT-3 and the OpenAI API*, is a high-level introduction to GPT-3 and the OpenAI API.

Chapter 2, *GPT-3 Applications and Use Cases*, is an overview of core GPT-3 use cases: text generation, classification, and semantic search.

Chapter 3, *Working with the OpenAI Playground*, is a semi-deep dive into the OpenAI Playground and the developer portal.

Chapter 4, *Working with the OpenAI API*, is an introduction to calling the OpenAI API using Postman.

Chapter 5, *Calling the OpenAI API in Code*, is an introduction to using the OpenAI API with both Node.js/JavaScript and Python.

Chapter 6, *Content Filtering*, explains how to implement content filtering.

Chapter 7, *Generating and Transforming Text*, contains code and prompt examples for generating and transforming text.

Chapter 8, *Classifying and Categorizing Text*, takes a closer look at text classification and the OpenAI API Classification endpoint.

Chapter 9, *Building a GPT-3 Powered Question-Answering App*, explains how to build a functional GPT-3 powered web knowledge base.

Chapter 10, *Going Live with OpenAI-Powered Apps*, explains the OpenAI application review and approval process and discusses getting ready for a review.

To get the most out of this book

All of the code examples in this book were written using a web-based **Integrated Development Environment** (**IDE**) from `replit.com`. A free replit.com account is sufficient to follow the examples. To use replit.com, all that is required is a modern web browser and a replit.com account. The code has also been tested on macOS using Visual Studio Code, although it should work with any code editor and properly configured operating system. Code examples are provided in both Node.js/JavaScript and Python. For Node.js, version 12.16.1 is used and for Python, version 3.8.2 is used.

Software/hardware covered in the book	OS requirements
Node.js version 12.16.1	Windows, macOS, and Linux (Any)
Python version 3.8.2	Windows, macOS, and Linux (Any)

All of the code examples will require an OpenAI API Key and access to the OpenAI API. You can request access to the OpenAI API by visiting `https://openai.com/api`.

If you are using the digital version of this book, we advise you to type the code yourself or access the code via the GitHub repository (link available in the next section). Doing so will help you avoid any potential errors related to the copying and pasting of code.

Download the color images

We also provide a PDF file that has color images of the screenshots/diagrams used in this book. You can download it here: `https://static.packt-cdn.com/downloads/9781800563193_ColorImages.pdf`.

Conventions used

There are a number of text conventions used throughout this book.

`Code in text`: Indicates code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "However, suppose you don't want the completion to generate the human side of the conversation and you want to use the label `AI:` rather than `Assistant:`?"

A block of code is set as follows:

```
English: I do not speak Spanish
Spanish:
```

Bold: Indicates a new term, an important word, or words that you see onscreen. For example, words in menus or dialog boxes appear in the text like this. Here is an example: "Each subsequent time the **Submit** button is clicked."

> **Tips or important notes**
> Appear like this.

Get in touch

Feedback from our readers is always welcome.

General feedback: If you have questions about any aspect of this book, mention the book title in the subject of your message and email us at `customercare@packtpub.com`.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit `www.packtpub.com/support/errata`, selecting your book, clicking on the Errata Submission Form link, and entering the details.

Piracy: If you come across any illegal copies of our works in any form on the Internet, we would be grateful if you would provide us with the location address or website name. Please contact us at `copyright@packt.com` with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit `authors.packtpub.com`.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions, we at Packt can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about Packt, please visit `packt.com`.

Section 1: Understanding GPT-3 and the OpenAI API

The objective of this section is to provide you with a high-level introduction to GPT-3 and the OpenAI API and to show how easy it is to get started with. The goal is to engage you with fun examples that are quick and simple to implement.

This section comprises the following chapters:

- *Chapter 1*, *Introducing GPT-3 and the OpenAI API*
- *Chapter 2*, *GPT-3 Applications and Use Cases*

1
Introducing GPT-3 and the OpenAI API

The buzz about **Generative Pre-trained Transformer Version 3** (**GPT-3**) started with a blog post from a leading **Artificial Intelligence** (**AI**) research lab, OpenAI, on June 11, 2020. The post began as follows:

> *We're releasing an API for accessing new AI models developed by OpenAI. Unlike most AI systems which are designed for one use-case, the API today provides a general-purpose "text in, text out" interface, allowing users to try it on virtually any English language task.*

Online demos from early beta testers soon followed—some seemed too good to be true. GPT-3 was writing articles, penning poetry, answering questions, chatting with lifelike responses, translating text from one language to another, summarizing complex documents, and even writing code. The demos were incredibly impressive—things we hadn't seen a general-purpose AI system do before—but equally impressive was that many of the demos were created by people with a limited or no formal background in AI and **Machine Learning** (**ML**). GPT-3 had raised the bar, not just in terms of the technology, but also in terms of AI accessibility.

GPT-3 is a general-purpose language processing AI model that practically anybody can understand and start using in a matter of minutes. You don't need a **Doctor of Philosophy** (**PhD**) in computer science—you don't even need to know how to write code. In fact, everything you'll need to get started is right here in this book. We'll begin in this chapter with the following topics:

- Introduction to GPT-3
- Democratizing NLP
- Understanding prompts, completions, and tokens
- Introducing Davinci, Babbage, Curie, and Ada
- Understanding GPT-3 risks

Technical requirements

This chapter requires you to have access to the **OpenAI Application Programming Interface** (**API**). You can register for API access by visiting `https://openapi.com`.

Introduction to GPT-3

In short, GPT-3 is a language model: a statistical model that calculates the probability distribution over a sequence of words. In other words, GPT-3 is a system for guessing which text comes next when text is given as an input.

Now, before we delve further into what GPT-3 is, let's cover a brief introduction (or refresher) on **Natural Language Processing** (**NLP**).

Simplifying NLP

NLP is a branch of AI that focuses on the use of natural human language for various computing applications. NLP is a broad category that encompasses many different types of language processing tasks, including sentiment analysis, speech recognition, machine translation, text generation, and text summarization, to name but a few.

In NLP, language models are used to calculate the probability distribution over a sequence of words. Language models are essential because of the extremely complex and nuanced nature of human languages. For example, *pay in full* and *painful* or *tee time* and *teatime* sound alike but have very different meanings. A phrase such as *she's on fire* could be literal or figurative, and words such as *big* and *large* can be used interchangeably in some cases but not in others—for example, using the word *big* to refer to an older sibling wouldn't have the same meaning as using the word *large*. Thus, language models are used to deal with this complexity, but that's easier said than done.

While understanding things such as word meanings and their appropriate usage seems trivial to humans, NLP tasks can be challenging for machines. This is especially true for more complex language processing tasks such as recognizing irony or sarcasm—tasks that even challenge humans at times.

Today, the best technical approach to a given NLP task depends on the task. So, most of the best-performing, **state-of-the-art** (**SOTA**) NLP systems are specialized systems that have been fine-tuned for a single purpose or a narrow range of tasks. Ideally, however, a single system could successfully handle any NLP task. That's the goal of GPT-3: to provide a general-purpose AI system for NLP. So, even though the best-performing NLP systems today tend to be specialized, purpose-built systems, *GPT-3 achieves SOTA performance on a number of common NLP tasks*, showing the potential for a future general-purpose NLP system that could provide SOTA performance for any NLP task.

What exactly is GPT-3?

Although GPT-3 is a general-purpose NLP system, it really just does one thing: it predicts what comes next based on the text that is provided as input. But it turns out that, with the right architecture and enough data, this *one thing* can handle a stunning array of language processing tasks.

GPT-3 is the third version of the GPT language model from OpenAI. So, although it started to become popular in the summer of 2020, the first version of GPT was announced 2 years earlier, and the following version, GPT-2, was announced in February 2019. But even though GPT-3 is the third version, the general system design and architecture hasn't changed much from GPT-2. There is one big difference, however, and that's the size of the dataset that was used for training.

GPT-3 was trained with a massive dataset comprised of text from the internet, books, and other sources, containing roughly 57 billion words and 175 billion parameters. That's 10 times larger than GPT-2 and the next-largest language model. To put the model size into perspective, the average human might read, write, speak, and hear upward of a billion words in an entire lifetime. So, GPT-3 has been trained on an estimated 57 times the number of words most humans will ever process.

The GPT-3 language model is massive, so it isn't something you'll be downloading and dabbling with on your laptop. But even if you could (which you can't because it's not available to download), it would cost millions of dollars in computing resources each time you wanted to build the model. This would put GPT-3 out of reach for most small companies and virtually all individuals if you had to rely on your own computer resource to use it. Thankfully, you don't. OpenAI makes GPT-3 available through an API that is both affordable and easy to use. So, anyone can use some of the most advanced AI ever created!

Democratizing NLP

Anyone can use GPT-3 with access to the OpenAI API. The API is a general-purpose *text in, text out* interface that could be used for virtually any language task. To use the API, you simply pass in text and get a text response back. The task might be to do sentiment analysis, write an article, answer a question, or summarize a document. It doesn't matter, as far as the API is concerned—it's all done the same way, which makes using the API easy enough for just about anyone to use, even non-programmers.

The text you pass in is referred to as a **prompt**, and the returned text is called a **completion**. A prompt is used by GPT-3 to determine how best to complete the task. In the simplest case, a prompt can provide a few words to get started with. For example, if the prompt was *If today is Monday, tomorrow is*, GPT-3 would likely respond with *Tuesday*, along with some additional text such as *If today is Tuesday, tomorrow is Wednesday*, and so on. This means that what you get out of GPT-3 depends on what you send to it.

As you might guess, the quality of a completion depends heavily on the prompt. GPT-3 uses all of the text in a prompt to help generate the most relevant completion. Each and every word, along with how the prompt is structured, helps improve the language model prediction results. So, *understanding how to write and test prompts is the key to unlocking GPT-3's true potential*.

Understanding prompts, completions, and tokens

Literally any text can be used as a prompt—send some text in and get some text back. However, as entertaining as it can be to see what GPT-3 does with random strings, the real power comes from understanding how to write effective prompts.

Prompts

Prompts are how you get GPT-3 to do what you want. It's like programming, but with plain English. So, you have to know what you're trying to accomplish, but rather than writing code, you use words and plain text.

When you're writing prompts, the main thing to keep in mind is that GPT-3 is trying to figure out which text should come next, so including things such as instructions and examples provides context that helps the model figure out the best possible completion. Also, quality matters— for example, spelling, unclear text, and the number of examples provided will have an effect on the quality of the completion.

Another key consideration is the prompt size. While a prompt can be any text, the prompt and the resulting completion must add up to fewer than 2,048 tokens. We'll discuss tokens a bit later in this chapter, but that's roughly 1,500 words.

So, a prompt can be any text, and there aren't hard and fast rules that must be followed like there are when you're writing code. However, there are some guidelines for structuring your prompt text that can be helpful in getting the best results.

Different kinds of prompts

We'll dive deep into prompt writing throughout this book, but let's start with the different prompt types. These are outlined as follows:

- Zero-shot prompts
- One-shot prompts
- Few-shot prompts

Zero-shot prompts

A **zero-shot prompt** is the simplest type of prompt. It only provides a description of a task, or some text for GPT-3 to get started with. Again, it could literally be anything: a question, the start of a story, instructions—anything, but the clearer your prompt text is, the easier it will be for GPT-3 to understand what should come next. Here is an example of a zero-shot prompt for generating an email message. The completion will pick up where the prompt ends—in this case, after `Subject:`:

```
Write an email to my friend Jay from me Steve thanking him for
covering my shift this past Friday. Tell him to let me know if
I can ever return the favor.
Subject:
```

The following screenshot is taken from a web-based testing tool called the **Playground**. We'll discuss the Playground more in *Chapter 2*, *GPT-3 Applications and Use Cases,* and *Chapter 3*, *Working with the OpenAI Playground*, but for now we'll just use it to show the completion generated by GPT-3 as a result of the preceding prompt. Note that the original prompt text is bold, and the completion shows as regular text:

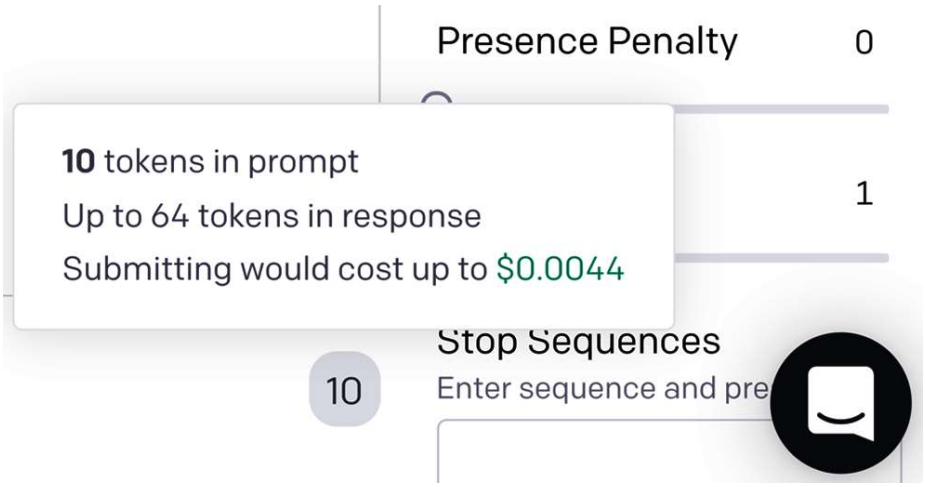


Figure 1.1 – Zero-shot prompt example

So, a zero-shot prompt is just a few words or a short description of a task without any examples. Sometimes this is all GPT-3 needs to complete the task. Other times, you may need to include one or more examples. A prompt that provides a single example is referred to as a one-shot prompt.

One-shot prompts

A **one-shot prompt** provides one example that GPT-3 can use to learn how to best complete a task. Here is an example of a one-shot prompt that provides a task description (the first line) and a single example (the second line):

```
A list of actors in the movie Star Wars
1. Mark Hamill: Luke Skywalker
```

From just the description and the one example, GPT-3 learns what the task is and that it should be completed. In this example, the task is to create a list of actors from the movie *Star Wars*. The following screenshot shows the completion generated from this prompt:

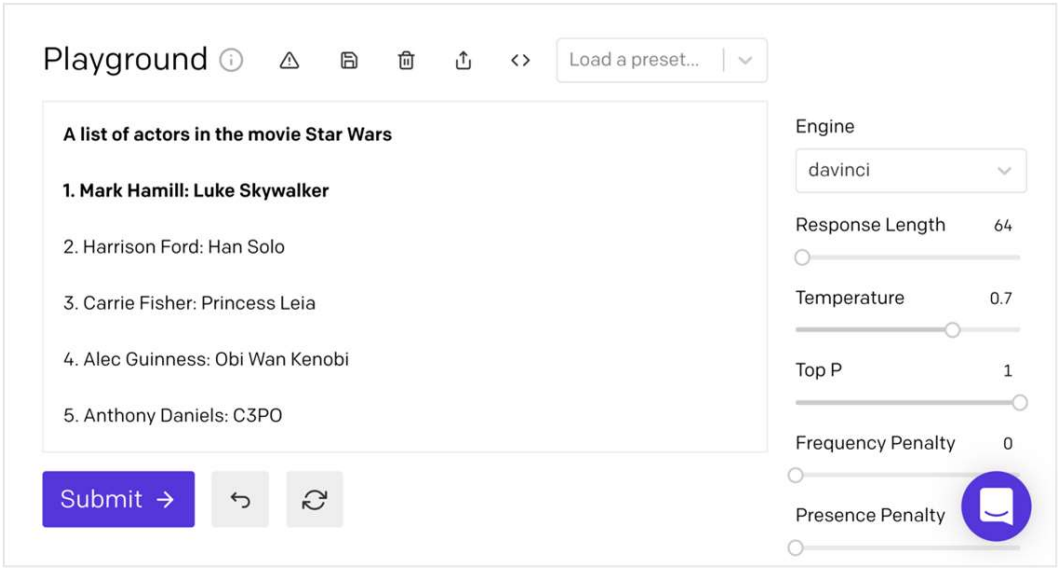


Figure 1.2 – One-shot prompt example

The one-shot prompt works great for lists and commonly understood patterns. But sometimes you'll need more than one example. When that's the case you'll use a few-shot prompt.

Few-shot prompts

A **few-shot prompt** provides multiple examples—typically, 10 to 100. Multiple examples can be useful for showing a pattern that GPT-3 should continue. Few-shot prompts and more examples will likely increase the quality of the completion because the prompt provides more for GPT-3 to learn from.

Here is an example of a few-shot prompt to generate a simulated conversation. Notice that the examples provide a back-and-forth dialog, with things that might be said in a conversation:

```
This is a conversation between Steve, the author of the book
Exploring GPT-3 and someone who is reading the book.
Reader: Why did you decide to write the book?
Steve: Because I'm super fascinated by GPT-3 and emerging
technology in general.
Reader: What will I learn from this book?
Steve: The book provides an introduction to GPT-3 from OpenAI.
You'll learn what GPT-3 is and how to get started using it.
Reader: Do I need to be a coder to follow along?
```

```
Steve: No. Even if you've never written a line of code before,
you'll be able to follow along just fine.
Reader:
```

In the following screenshot, you can see that GPT-3 continues the simulated conversation that was started in the examples provided in the prompt:

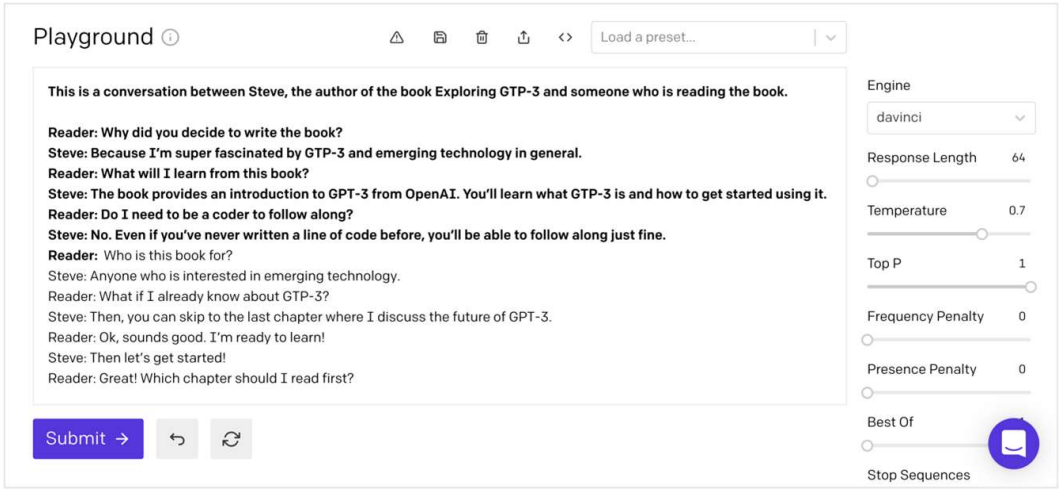


Figure 1.3 – Few-shot prompt example

Now that you understand the different prompt types, let's take a look at some prompt examples.

Prompt examples

The OpenAI API can handle a variety of tasks. The possibilities range from generating original stories to performing complex text analysis, and everything in between. To get familiar with the kinds of tasks GPT-3 can perform, OpenAI provides a number of prompt examples. You can find example prompts in the Playground and in the OpenAI documentation.

In the Playground, the examples are referred to as **presets**. Again, we'll cover the Playground in detail in *Chapter 3*, *Working with the OpenAI Playground*, but the following screenshot shows some of the presets that are available:

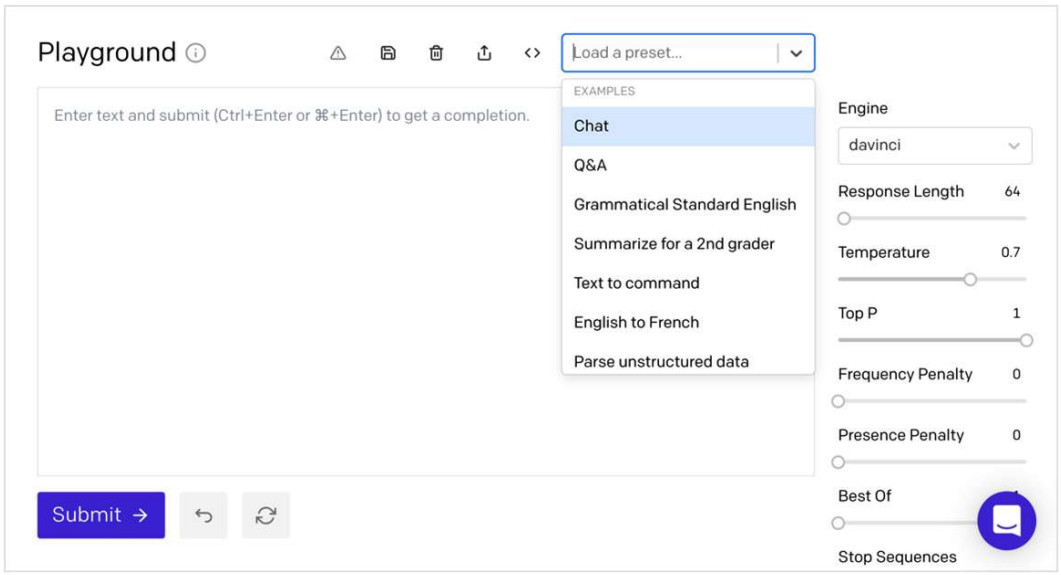


Figure 1.4 – Presets

Example prompts are also available in the OpenAI documentation. The OpenAI documentation is excellent and includes a number of great prompt examples, with links to open and test them in the Playground. The following screenshot shows an example prompt from the OpenAI documentation. Notice the **Open this example in Playground** link below the prompt example. You can use that link to open the prompt in the Playground:

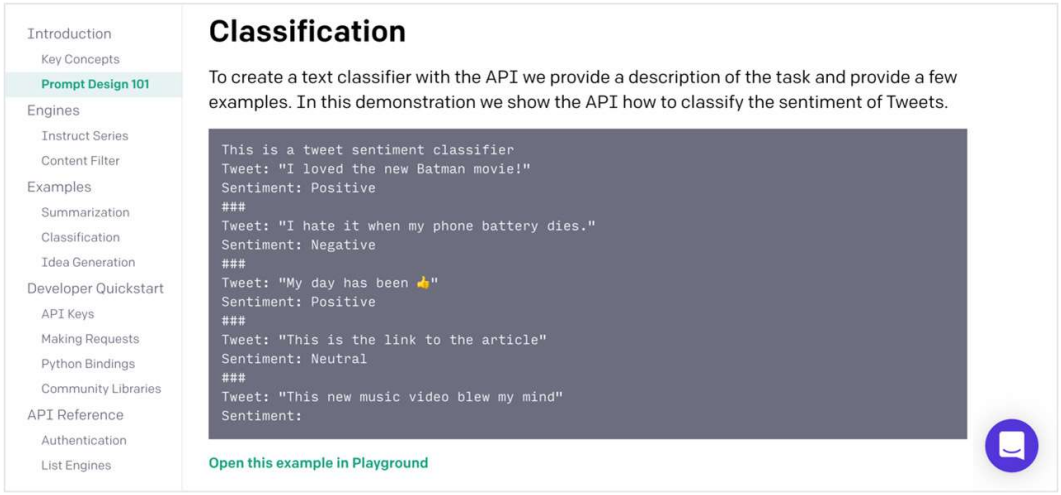


Figure 1.5 – OpenAI documentation provides prompt examples

Now that you have an understanding of prompts, let's talk about how GPT-3 uses them to generate a completion.

Completions

Again, a completion refers to the text that is generated and returned as a result of the provided prompt/input. You'll also recall that GPT-3 was not specifically trained to perform any one type of NLP task—it's a general-purpose language processing system. However, GPT-3 can be shown how to complete a given task using a prompt. This is called meta-learning.

Meta-learning

With most NLP systems, the data used to teach the system how to complete a task is provided when the underlying ML model is trained. So, to improve results for a given task, the underlying training must be updated, and a new version of the model must be built. GPT-3 works differently, as it isn't trained for any specific task. Rather, it was designed to recognize patterns in the prompt text and to continue the pattern(s) by using the underlying general-purpose model. This approach is referred to as **meta-learning** because the prompt is used to *teach* GPT-3 how to generate the best possible completion, without the need for retraining. So, in effect, the different prompt types (zero-shot, one-shot, and few-shot) can be used to *program* GPT-3 for different types of tasks, and you can provide a lot of instructions in the prompt—up to 2,048 tokens. Alright—now is a good time to talk about tokens.

Tokens

When a prompt is sent to GPT-3, it's broken down into tokens. **Tokens** are numeric representations of words or—more often—parts of words. Numbers are used for tokens rather than words or sentences because they can be processed more efficiently. This enables GPT-3 to work with relatively large amounts of text. That said, as you've learned, there is still a limit of 2,048 tokens (approximately ~1,500 words) for the combined prompt and the resulting generated completion.

You can stay under the token limit by estimating the number of tokens that will be used in your prompt and resulting completion. On average, for English words, every four characters represent one token. So, just add the number of characters in your prompt to the *response length* and divide the sum by four. This will give you a general idea of the tokens required. This is helpful if you're trying to get an idea of how many tokens are required for a number of tasks.

Another way to get the token count is with the token count indicator in the Playground. This is located just under the large text input, on the bottom right. The magnified area in the following screenshot shows the token count. If you hover your mouse over the number, you'll also see the total count with the completion. For our example, the prompt **Do or do not. There is no try.**—the wise words from Master Yoda—uses **10** tokens and **74** tokens with the completion:

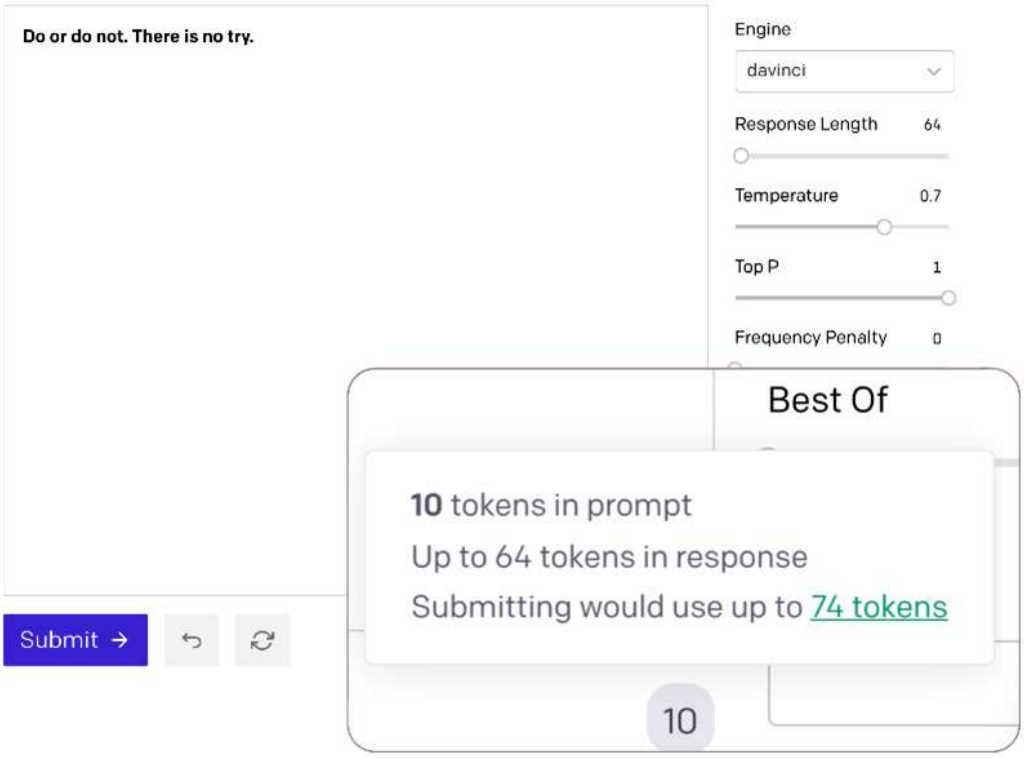


Figure 1.6 – Token count

While understanding tokens is important for staying under the 2,048 token limit, they are also important to understand because tokens are what OpenAI uses as the basis for usage fees. Overall token usage reporting is available for your account at `https://beta.openai.com/account/usage`. The following screenshot shows an example usage report. We'll discuss this more in *Chapter 3*, *Working with the OpenAI Playground*:

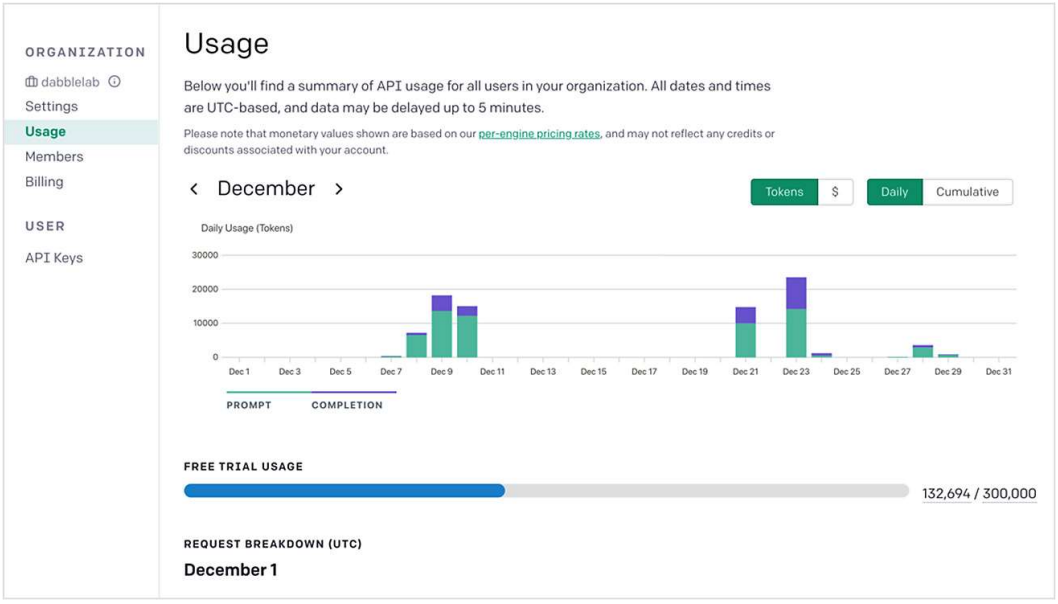


Figure 1.7 – Usage statistics

In addition to token usage, the other thing that affects the costs associated with using GPT-3 is the engine you choose to process your prompts. The engine refers to the language model that will be used. The main difference between the engines is the size of the associated model. Larger models can complete more complex tasks, but smaller models are more efficient. So, depending on the task complexity, you can significantly reduce costs by using a smaller model. The following screenshot shows the model pricing at the time of publishing. As you can see, the cost differences can be significant:

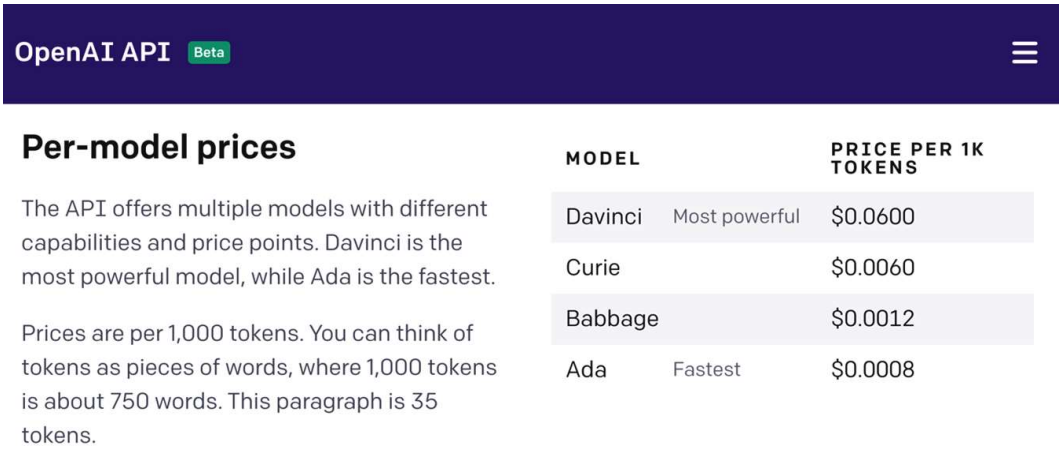


Figure 1.8 – Model pricing

So, the engines or models each has a different cost but the one you'll need depends on the task you're performing. Let's look at the different engine options next.

Introducing Davinci, Babbage, Curie, and Ada

The massive dataset that is used for training GPT-3 is the primary reason why it's so powerful. However, bigger is only better when it's necessary—and more power comes at a cost. For those reasons, OpenAI provides multiple models to choose from. Today there are four primary models available, along with a model for content filtering and **instruct models**.

The available models or engines (as they're also referred to) are named `Davinci`, `Babbage`, `Curie`, and `Ada`. Of the four, `Davinci` is the largest and most capable. `Davinci` can perform any tasks that any other engine can perform. `Babbage` is the next most capable engine, which can do anything that `Curie` or `Ada` can do. `Ada` is the least capable engine, but the best-performing and lowest-cost engine.

When you're getting started and for initially testing new prompts, you'll usually want to begin with Davinci , then try, `Ada`, `Babbage`, or `Curie` to see if one of them can complete the task faster or more cost-effectively. The following is an overview of each engine and the types of tasks that might be best suited for each. However, keep in mind that you'll want to test. Even though the smaller engines might not be trained with as much data, they are all still general-purpose models.

Davinci

`Davinci` is the most capable model and can do anything that any other model can do, and much more—often with fewer instructions. `Davinci` is able to solve logic problems, determine cause and effect, understand the intent of text, produce creative content, explain character motives, and handle complex summarization tasks.

Curie

`Curie` tries to balance power and speed. It can do anything that `Ada` or `Babbage` can do but it's also capable of handling more complex classification tasks and more nuanced tasks like summarization, sentiment analysis, chatbot applications, and Question and Answers.

Babbage

`Babbage` is a bit more capable than `Ada` but not quite as performant. It can perform all the same tasks as `Ada`, but it can also handle a bit more involved classification tasks, and it's well suited for semantic search tasks that rank how well documents match a search query.

Ada

`Ada` is usually the fastest model and least costly. It's best for less nuanced tasks—for example, parsing text, reformatting text, and simpler classification tasks. The more context you provide `Ada`, the better it will likely perform.

Content filtering model

To help prevent inappropriate completions, OpenAI provides a content filtering model that is fine-tuned to recognize potentially offensive or hurtful language.

Instruct models

These are models that are built on top of the `Davinci` and `Curie` models. **Instruct models** are tuned to make it easier to tell the API what you want it to do. Clear instructions can often produce better results than the associated core model.

A snapshot in time

A final note to keep in mind about all of the engines is that they are all a *snapshot in time*, meaning the data used to train them cuts off on the date the model was built. So, GPT-3 is not working with up-to-the-minute or even up-to-the-day data—it's likely weeks or months old. OpenAI is planning to add more continuous training in the future, but today this is a consideration to keep in mind.

All of the GPT-3 models are extremely powerful and capable of generating text that is indistinguishable from human-written text. This holds tremendous potential for all kinds of potential applications. In most cases, that's a good thing. However, not all potential use cases are good.

Understanding GPT-3 risks

GPT-3 is a fantastic technology, with numerous practical and valuable potential applications. But as is often the case with powerful technologies, with its potential comes risk. In GPT-3's case, some of those risks include inappropriate results and potentially malicious use cases.

Inappropriate or offensive results

GPT-3 generates text so well that it can seem as though it is aware of what it is saying. It's not. It's an AI system with an excellent language model—it is not conscious in any way, so it will never willfully say something hurtful or inappropriate because it has no will. That said, it can certainly generate inappropriate, hateful, or malicious results—it's just not intentional.

Nevertheless, understanding that GPT-3 can and will likely generate offensive text at times needs to be understood and considered when using GPT or making GPT-3 results available to others. This is especially true for results that might be seen by children. We'll discuss this more and look at how to deal with it in *Chapter 6*, *Content Filtering*.

Potential for malicious use

It's not hard to imagine potentially malicious or harmful uses for GPT-3. OpenAI even describes how GPT-3 could be *weaponized* for misinformation campaigns or for creating fake product reviews. But OpenAI's declared mission is to *ensure that artificial general intelligence benefits all of humanity*. Hence, pursuing that mission includes taking responsible steps to prevent their AI from being used for the wrong purposes. So, OpenAI has implemented an application approval process for all applications that will use GPT-3 or the OpenAI API.

But as application developers, this is something we also need to consider. When we build an application that uses GPT-3, we need to consider if and how the application could be used for the wrong purposes and take the necessary steps to prevent it. We'll talk more about this in *Chapter 10*, *Going Live with OpenAI-Powered Apps*.

Summary

In this chapter, you learned that GPT-3 is a general-purpose language model for processing virtually any language processing task. You learned how GPT-3 works at a high level, along with key terms and concepts. We introduced the available models and discussed how all GPT-3 applications must go through an approval process to prevent potentially inappropriate or harmful results.

In the next chapter, we'll discuss different ways to use GPT-3 and look at specific GPT-3 use case examples.

2
GPT-3 Applications and Use Cases

GPT-3 was designed to be a general-purpose language processing model, meaning it wasn't explicitly trained for any one type of language processing task. So, possible use cases include virtually any natural language processing task you can imagine and others that probably haven't been imagined yet. New use cases for GPT-3 are constantly being discovered and that is a big part of the allure for many users. Sure, it does better with some tasks than others, but still, there are hundreds of possible uses. In this chapter, we'll break down some general use cases, and see how you can get started testing prompts of your own.

Our topics for this chapter are as follows:

- Understanding general GPT-3 use cases
- Introducing the Playground
- Handling text generation and classification tasks
- Understanding semantic search

Technical requirements

This chapter requires you to have access to the OpenAI API. You can register for API access by visiting `https://openapi.com`.

Understanding general GPT-3 use cases

In the last chapter, you learned that the OpenAI API is a *text in, text out* interface. So, it always returns a text response (called a **completion**) to a text input (called a **prompt**). The completion might be generating new text, classifying text, or providing results for a semantic search. The general-purpose nature of GPT-3 means it could be used for almost any language processing task. To keep us focused, we're going to look at the following general use cases: text generation, classification, and semantic search:

- **Text generation**: Text generation tasks are tasks for creating new, original text content. Examples include article writing and chatbots.
- **Classification**: Classification tasks tag or classify text. Examples of classification tasks include things such as sentiment analysis and content filtering.
- **Semantic search**: Semantic search tasks match a query with documents that are semantically related. For example, the query might be a question that gets matched to one or more documents that provide answers.

To illustrate different use cases, we'll be using the OpenAI **Playground**. So, before we dive into different example use cases, let's get acquainted with the Playground.

Introducing the Playground

To get started with GPT-3, OpenAI provides the Playground. The Playground is a web-based tool that makes it easy to test prompts and get familiar with how the API works. Just about everything you could do by calling the API (which we'll discuss in more detail later), you can also do in the Playground. Best of all, with the Playground, you can start using GPT-3 without writing a single line of code – you just provide a text input (the prompt) in plain English.

Getting started with the Playground

To access the Playground, you log in at `https://openai.com`. After you've authenticated, you'll be able to navigate to the Playground from the main menu.