

של מודלי שפה API - שימוש ב (LLM's) גדולים

▶ סדנה מעשית בהנחיית ישי שניידר

▶ Yishaishnayder@gmail.com



Yishai Shnayder



קצת עלי

- חוקח
- מוסמך בבריאות הציבור
- דאטה אנליסט

תודות

- תודות למורי, מאיה מלמוד ופרופ' לב מוצ'ניק, שנתנו לי כלים
- תודות לד"ר עמיחי פרלמן שבעצותיו הטובות ובחלקי קוד שהעמיד לרשותי נעזרתי בהרצאה הזו
- ותודה לנשים הטובות והאנשים הטובים מ-AI4R שבמסגרת הפרוייקט שלהם מועברת הסדנה הזו

איך נולדה הסדנה ומה ביקשתם?

באחת הקבוצות של קהילת AI4RI עלתה בקשה להראות כיצד משתמשים ב-API לגישה לכלי LLM

מה ביקשתם:

1. ללמוד עוד כלים לשימוש מחקרי והצגת מחקרים, מאמרים, עריכת מאמרים
2. לא השתמשתי ב-api ויהיה לי מעניין להבין את היכולות והטכניקה
3. אני מעוניינת ללמוד התממשקות למודלי שפה
4. אשמח שתראו שימוש עבור ניתוח איכותני
5. לקחת סט של שאלות, לתת לצ'אט סט של תשובות ולבקש מהבינה המלאכותית לבדוק את השאלות מתוך סט התשובות

עם מה תצאו מהסדנה?

1. נכיר API ותכונות של API
2. נלמד על היפרפרמטרים שמשנים את השאילתה
3. נלמד לשלוח בקשה ל- chatGPT דרך API
4. נלמד כיצד משלבים API עם משתנים שלנו, כולל משתנים ממסדי נתונים אחרים
5. נתמלל באמצעות API

חלק 1 - מבוא

□ מה זה בכלל API?

□ על מפתחות ואסימונים

□ איך מתחילים

ÉŃI



Application Programming Interface



מה נותן לנו API באופן כללי?

1. אבטחה משופרת- המשתמש לא חשוף למסד הנתונים שאיתו הוא עובד
2. גמישות- יכולת לקבל רק מה שצריך
3. מהירות – לא צריך לקרוא לכל מסד הנתונים אלא רק לחלק ממנו
4. שימוש בפחות משאבים- נגזר מסעיפים 2 ו-3

דוגמאות ל-API's:

מזג אויר, בורסה, שערי מטבע, מאגרי מידע ממשלתיים



למה להשתמש ב-API ב-LLM?

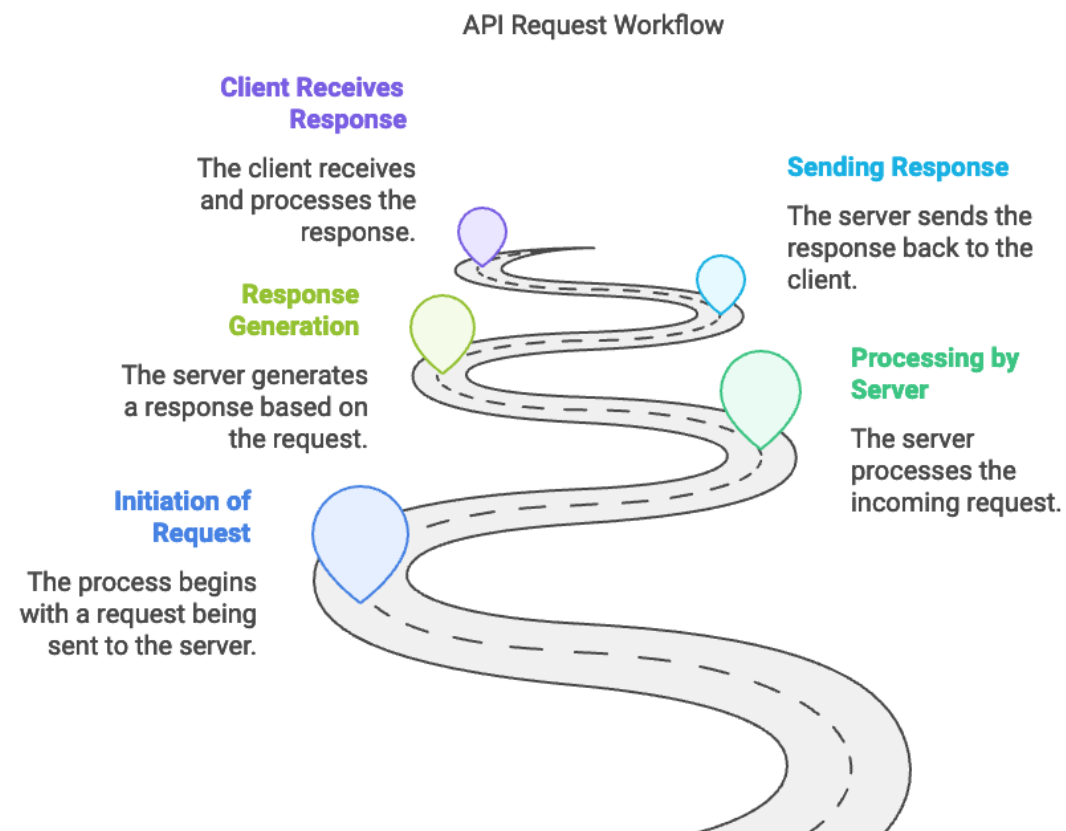
■ API משמש אותנו לביצוע משימות שלא היינו יכולים לבצע בלעדיו. לדוגמה, עיבוד מספר רב של קבצים באופן אוטומטי, הטמעת תהליכים בתוך התוכנה שלנו

■ API מאפשר שליטה בהפרמטרים של המודל

■ שליטה רבה יותר ב-roles בתוך המודל



מסע השאילתה



אז מה צריך לעשות על מפתחות ואסימונים



API Keys

These are unique identifiers that grant you access to an API. They often have rate limits, which specify how many requests you can make per time period.



Authentication Tokens

These are short-lived tokens that verify your identity and authorize access to specific resources or functionalities.



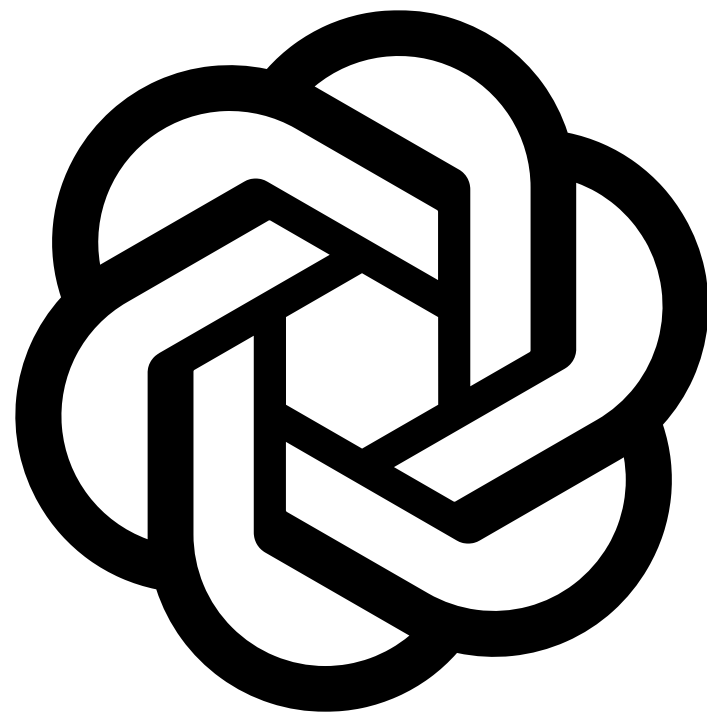
API Documentation

Detailed instructions on how to use the API, including available endpoints, parameters, and request/response formats.



מתי מקבלים את המפתח?

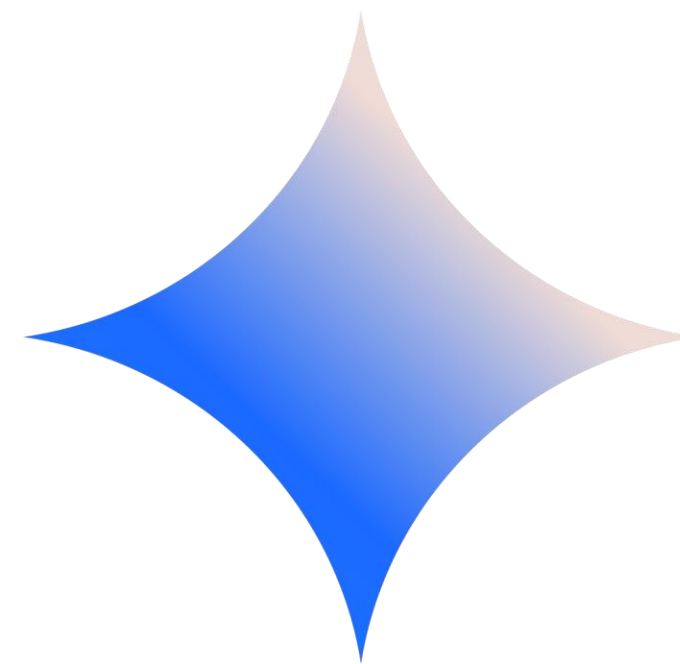
Open AI



Claude



Gemini



Storing Your API Key



API Key is confidential! Hence, you should store it separated from your code, in case you want to share your code

Where to store?

- In your environment
- In your Operation System



Storing Your API Key



API Key is confidential! Hence, you should store it separated from your code, in case you want to share your code

Where to store?

- In your environment
- In your Operation System

In terminal, create an .env file:

```
touch .env
```

Go to your .env file in the environment using file explorer, and paste:

```
API_KEY=your_api_key_here
```

Now, In python:

```
from dotenv import load_dotenv
import os

load_dotenv() # Load .env file
api_key = os.getenv("API_KEY")
```



Storing Your API Key



API Key is confidential! Hence, you should store it separated from your code, in case you want to share your code

Where to store?

- In your environment
- In your Operation System

<https://help.openai.com/en/articles/5112595-best-practices-for-api-key-safety>



Storing Your API Key



API Key is confidential! Hence, you should store it separated from your code, in case you want to share your code

Where to store?

- In your environment
- In your Operation System

In mac Keychain:

Bash:

```
security add-generic-password -a your_account -s  
your_service_name -w "your_api_key"
```



Storing Your API Key



API Key is confidential! Hence, you should store it separated from your code, in case you want to share your code

Where to store?

- In your environment
- In your Operation System

In mac Keychain:

Python:

```
import subprocess
```

```
def get_api_key(account, service):  
    result = subprocess.run(  
        ["security", "find-generic-password", "-a", account, "-s", service, "-w"],  
        stdout=subprocess.PIPE,  
        text=True,  
        check=True,  
    )  
    return result.stdout.strip()
```

```
api_key = get_api_key("your_account", "your_service_name")
```



Integrating the API into Your Application

1

Choose a Language

In our case: Python, but you can use JavaScript, Rust etc.

2

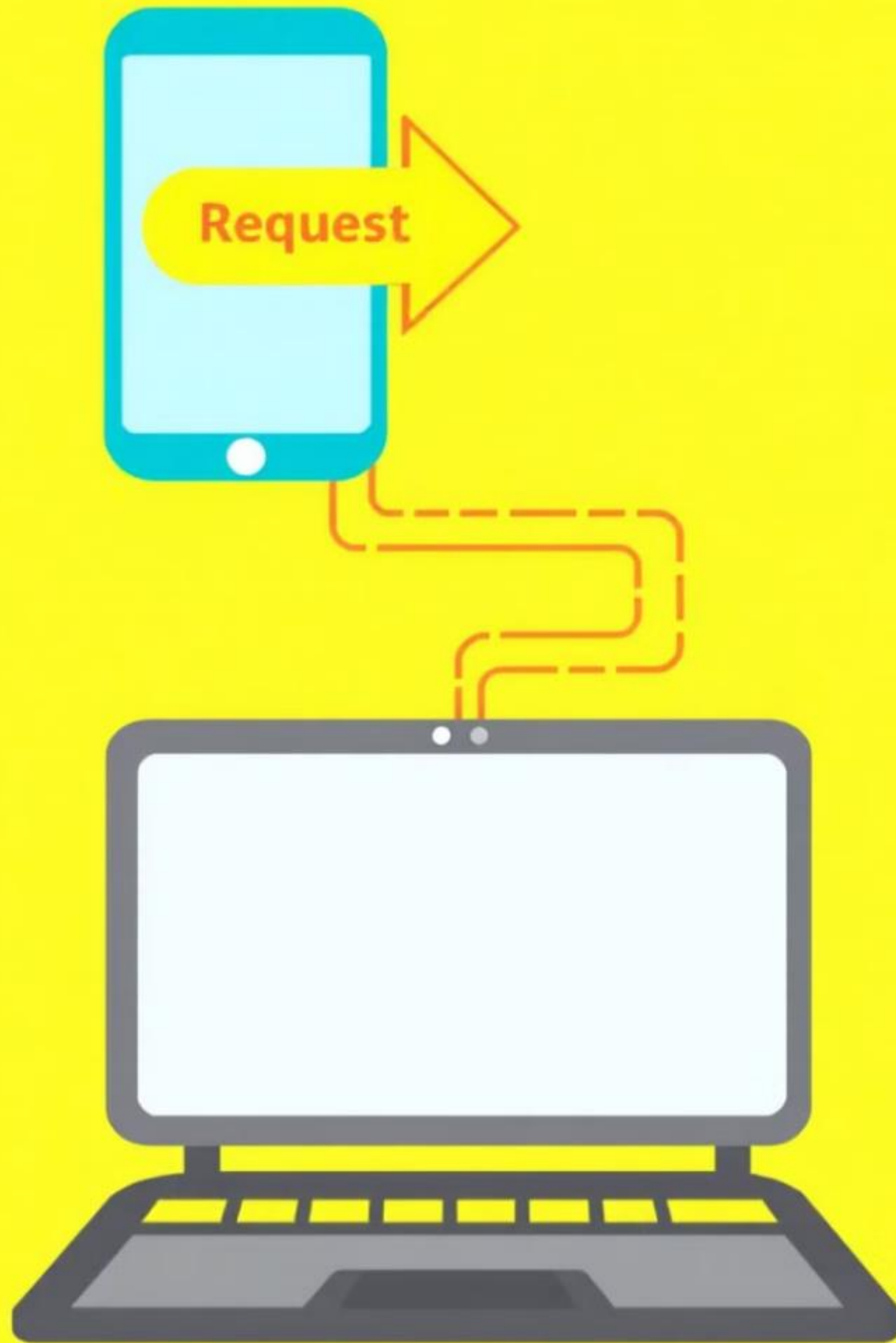
Install Libraries

Utilize libraries or SDKs to simplify API interactions and handle common tasks.

3

Configure Settings

Set up API credentials, endpoints, and any required parameters.



Making API Calls and Handling Responses

Formulate Requests

1

Send requests to the API using the appropriate HTTP methods (e.g., GET, POST, PUT, DELETE) and parameters.

Receive Responses

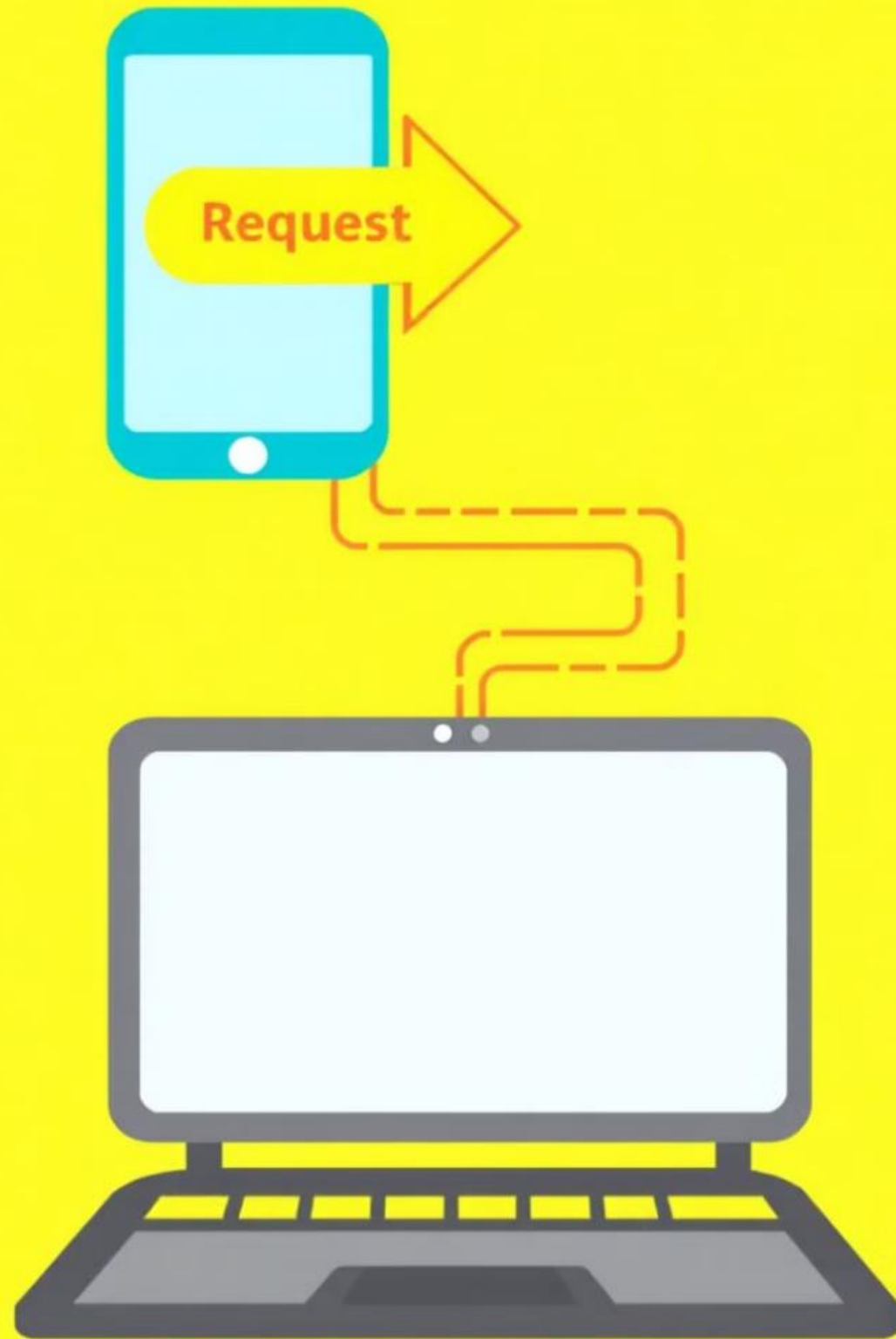
2

Handle responses, typically in JSON or XML format, and parse the data to extract the information you need.

Process Data

3

Use the retrieved data to update your application's state, display information, or perform other actions.

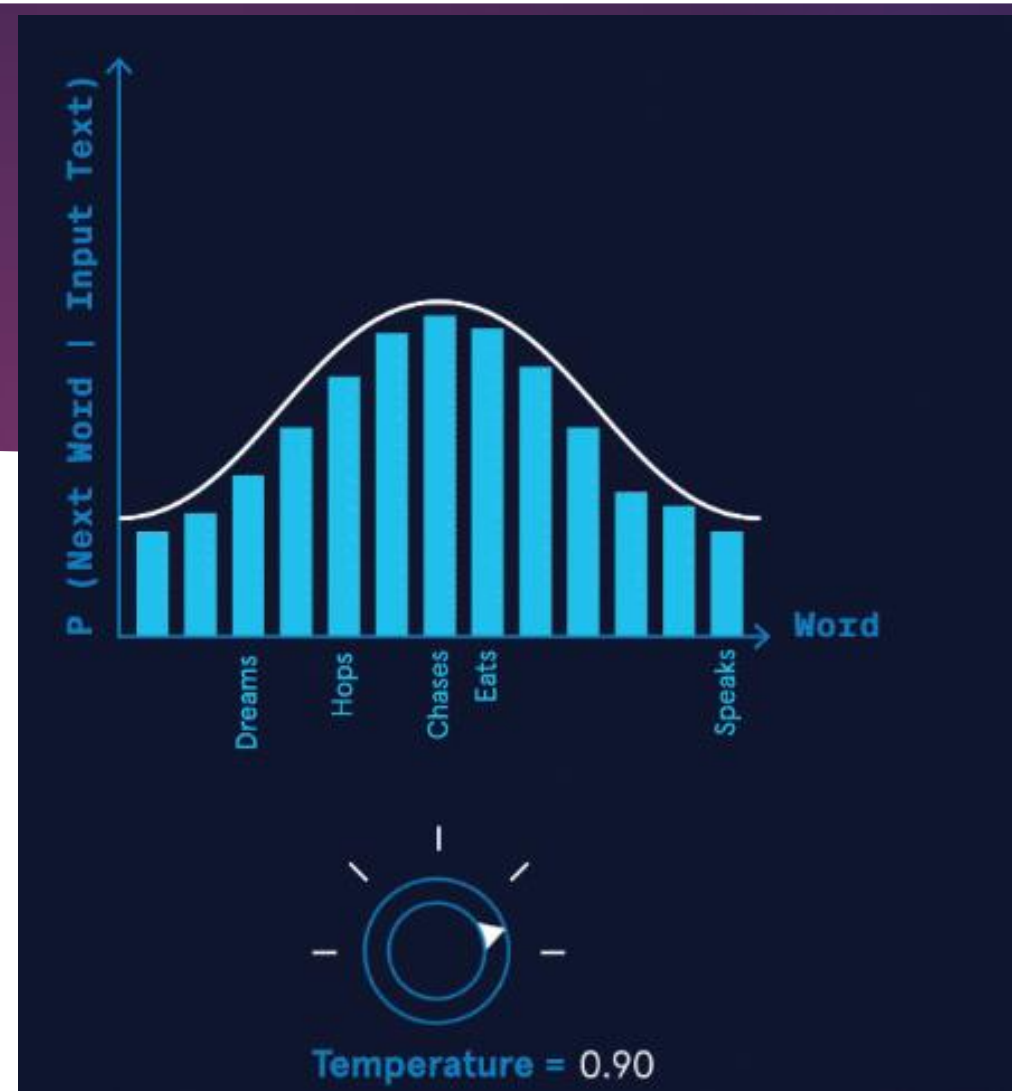
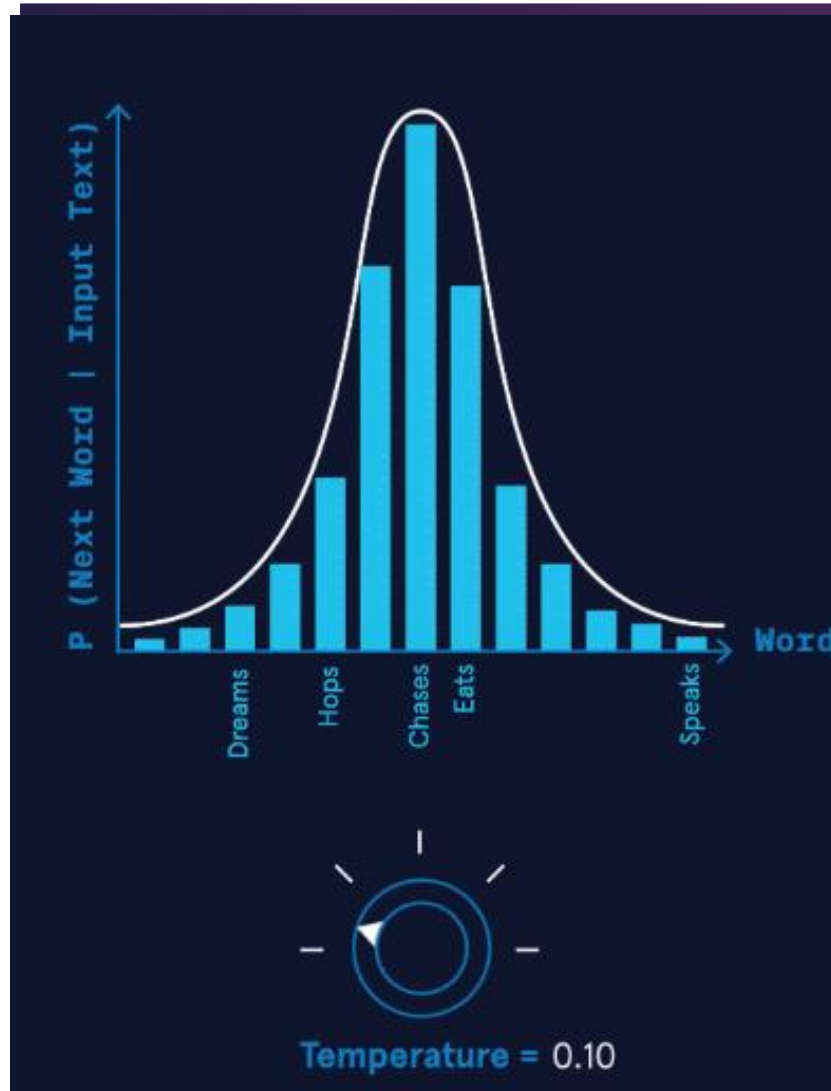


Hyperparameters (Do not confuse with Parameters) Parameters)

Parameters are the weights of the model and are part of the model.

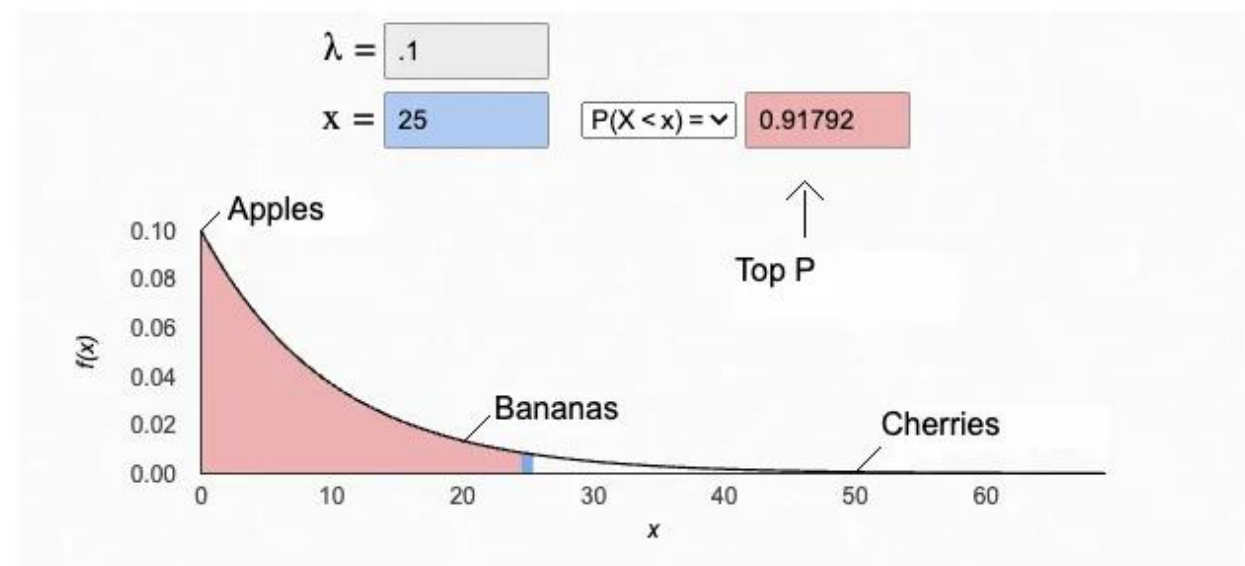
Hyperparameters are values passed to a query in order to fine tune the answer





<https://www.codecademy.com/courses/intro-to-open-ai-gpt-api/lessons/open-ai-api/exercises/token-sampling>

Top P



https://medium.com/@_b/transforming-chaos-into-creativity-with-top-p-top-k-and-temperature-55808ac90314

Try it yourself:

https://colab.research.google.com/drive/1yeLM1LoaEqTAS6D_Op9_L2pLA06uUGW1

Temperature Vs. top-p Rule of thumbs

Use Case	Temperature	Top_p	Description
Code Generation	0.2	0.1	Generates code that adheres to established patterns and conventions. Output is more deterministic and focused. Useful for generating syntactically correct code.
Creative Writing	0.7	0.8	Generates creative and diverse text for storytelling. Output is more exploratory and less constrained by patterns.
Chatbot Responses	0.5	0.5	Generates conversational responses that balance coherence and diversity. Output is more natural and engaging.
Code Comment Generation	0.3	0.2	Generates code comments that are more likely to be concise and relevant. Output is more deterministic and adheres to conventions.
Data Analysis Scripting	0.2	0.1	Generates data analysis scripts that are more likely to be correct and efficient. Output is more deterministic and focused.
Exploratory Code Writing	0.6	0.7	Generates code that explores alternative solutions and creative approaches. Output is less constrained by established patterns.

What are the components of an API request?

id: Identifier unique to each completion

object: Completion endpoint used


created: Timestamp of the completion

model: The model used for the completion

system_fingerprint: An identifier of the backend system configuration

choices: List of completions, covered in depth below


usage: Token usage, split up into input, output, and total tokens used

 Identifier

 Endpoint

 Timestamp

 Model

 Fingerprint-
Behind the
scene

 Choices



Usage- What happend
to my credit