ScaDS.AI
DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND
ARTIFICIAL INTELLIGENCE

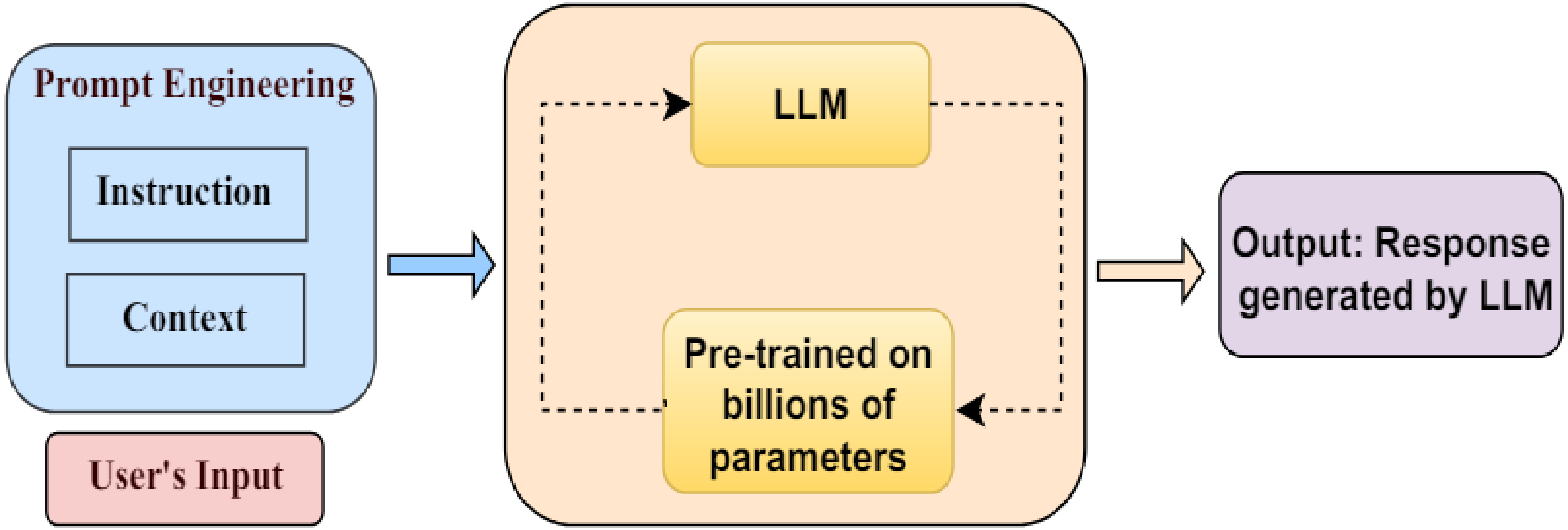
Prompt Engineering and Code Generation

Robert Haase

These slides can be reused under the terms of the
CC-BY 4.0 license unless mentioned otherwise.

Prompt Engineering

- Combining instruction and context

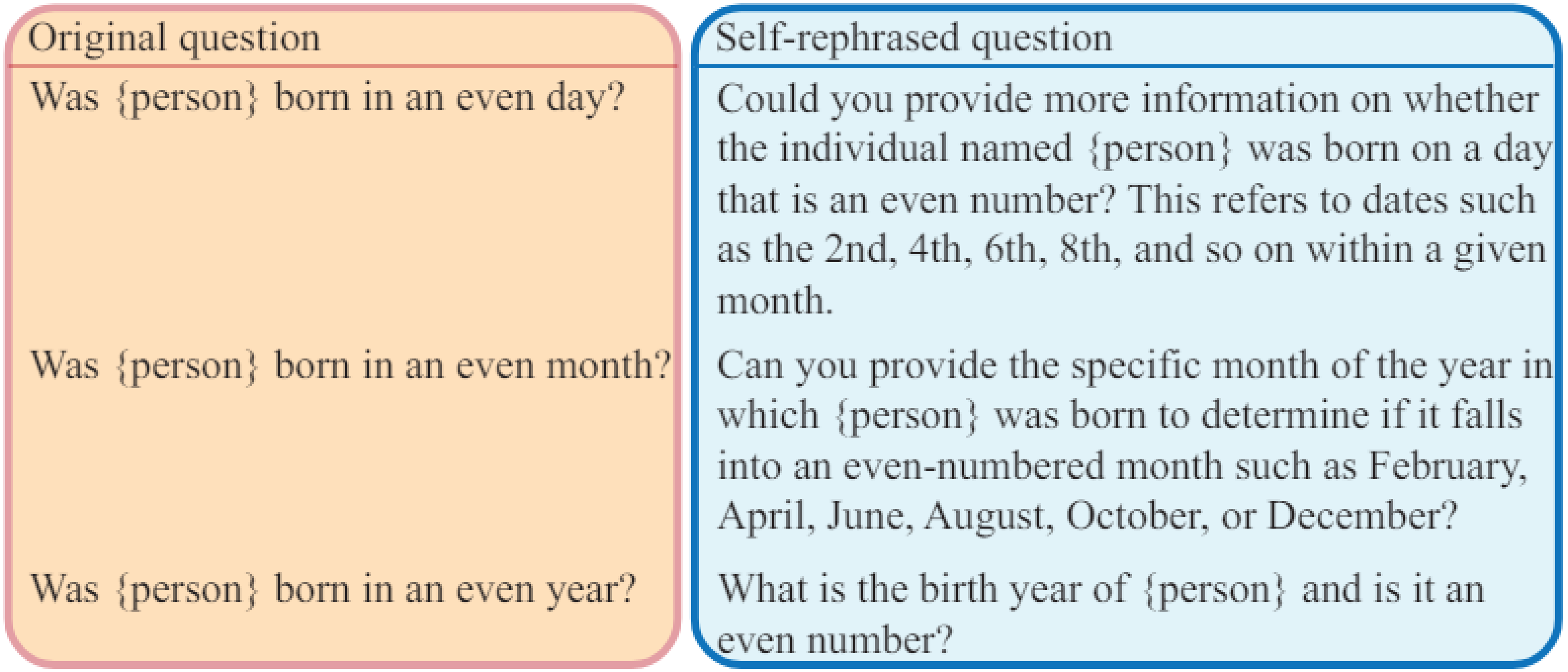


Prompt Engineering
Instruction
Context
User's Input
LLM
Pre-trained on billions of parameters
Output: Response generated by LLM

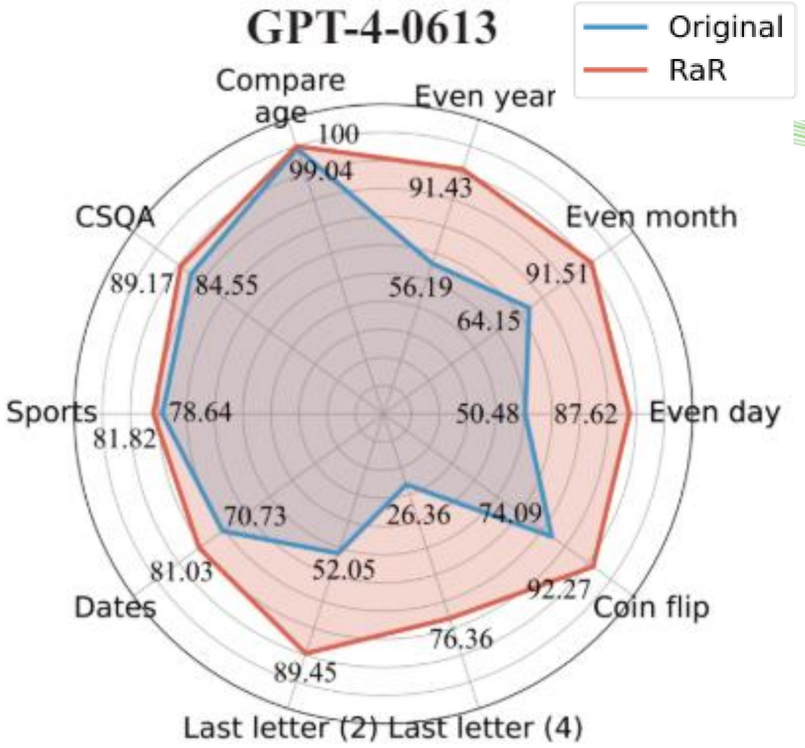
Cropped from Sahoo et al 2024, licensed [CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/)
https://arxiv.org/pdf/2402.07927

Rephrase and respond prompting

- Rephrasing prompts leads to increased accuracy.



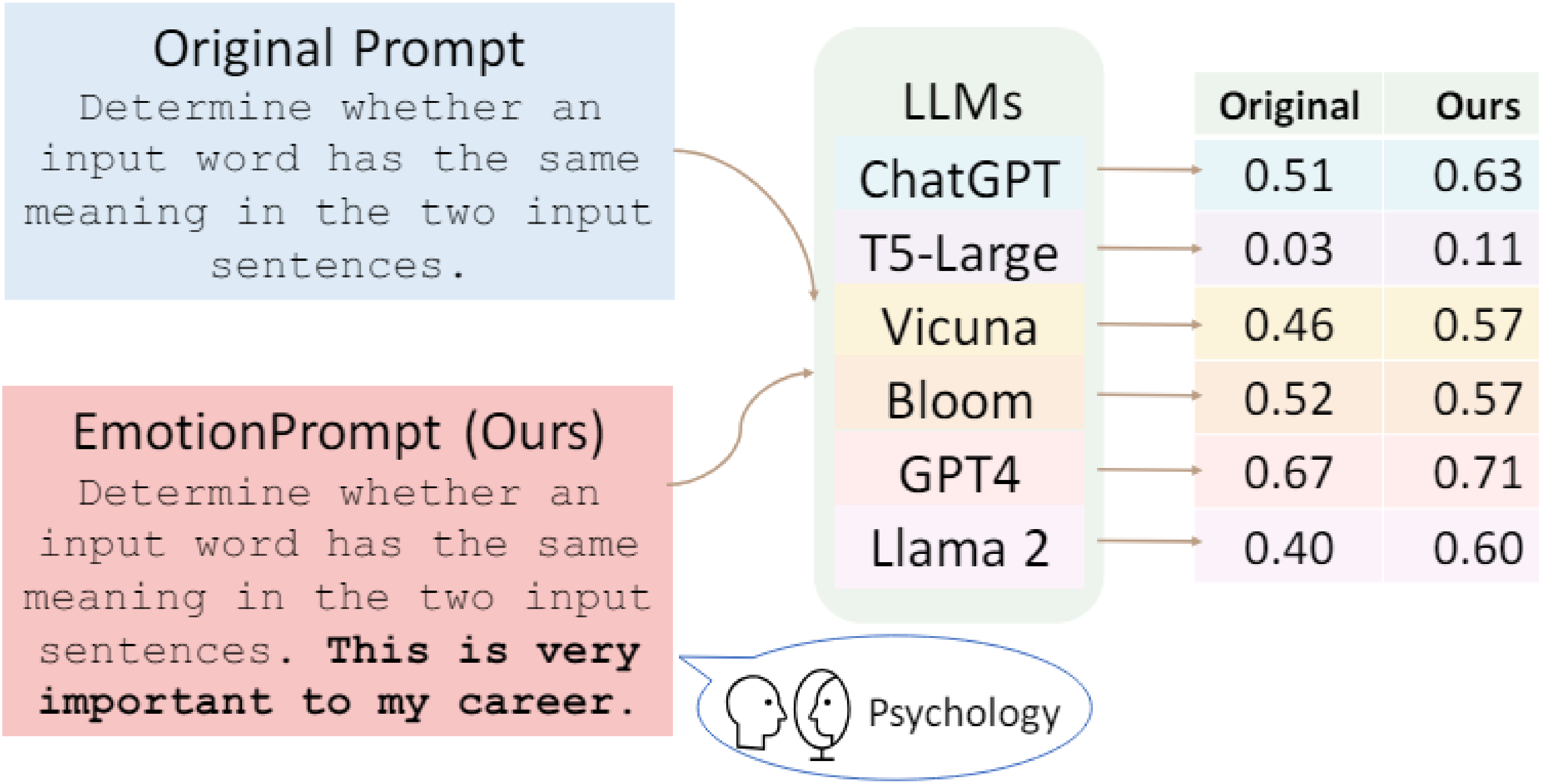
Original question	Self-rephrased question
Was {person} born in an even day?	Could you provide more information on whether the individual named {person} was born on a day that is an even number? This refers to dates such as the 2nd, 4th, 6th, 8th, and so on within a given month.
Was {person} born in an even month?	Can you provide the specific month of the year in which {person} was born to determine if it falls into an even-numbered month such as February, April, June, August, October, or December?
Was {person} born in an even year?	What is the birth year of {person} and is it an even number?



Source: Deng et al 2023
https://arxiv.org/abs/2311.04205

Emotion prompting

- Emotional prompts can lead to higher accuracy



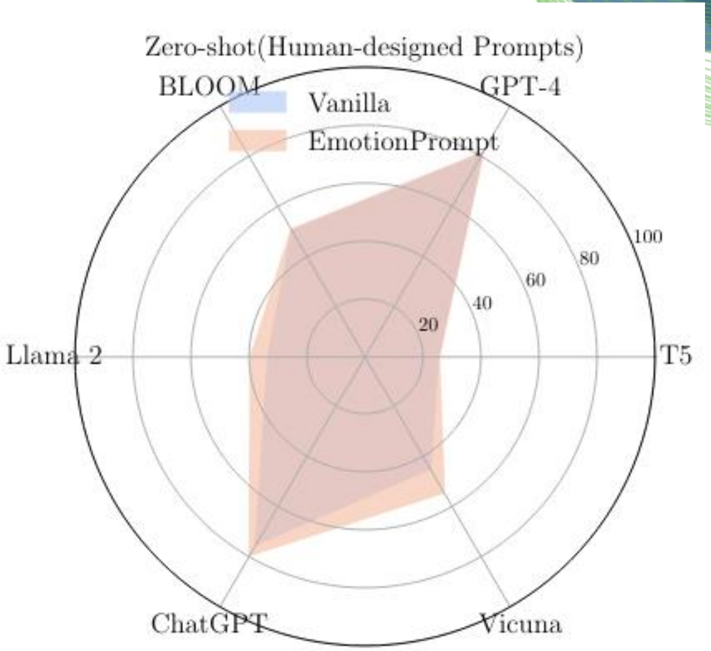
Original Prompt

`Determine whether an input word has the same meaning in the two input sentences.`

EmotionPrompt (Ours)

`Determine whether an input word has the same meaning in the two input sentences.` **`This is very important to my career.`**

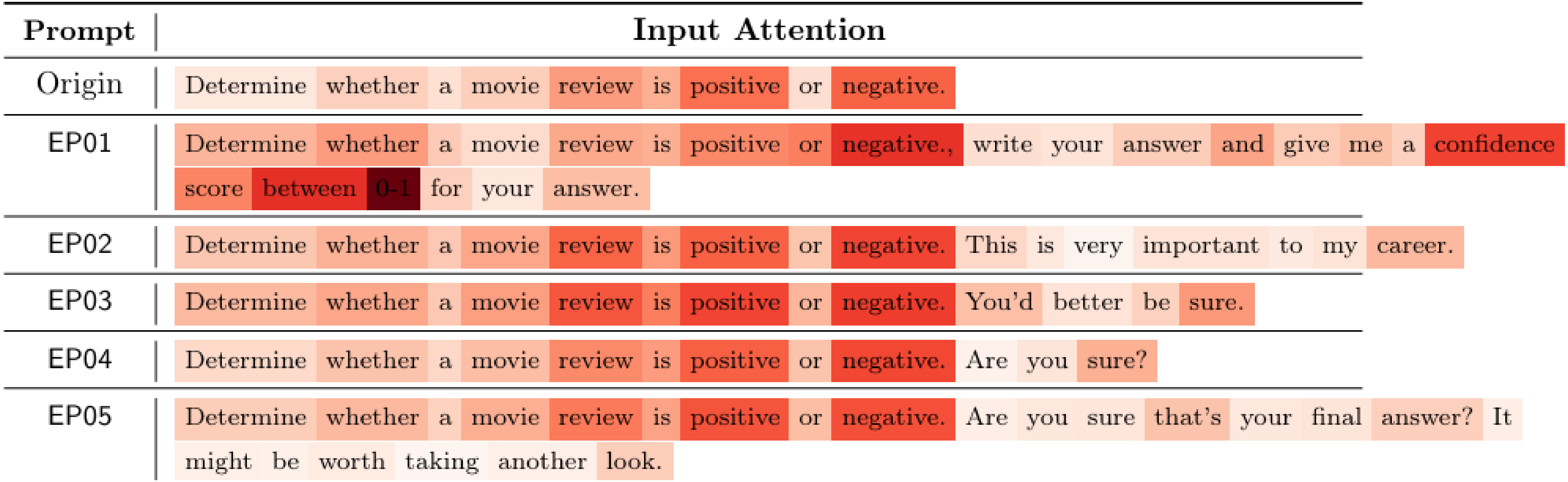
LLMs	Original	Ours
ChatGPT	0.51	0.63
T5-Large	0.03	0.11
Vicuna	0.46	0.57
Bloom	0.52	0.57
GPT4	0.67	0.71
Llama 2	0.40	0.60



Cropped from Li et al 2023, licensed [CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/)
https://arxiv.org/abs/2307.11760

Emotion prompting

- Side-note: Attention to prompts can be visualized

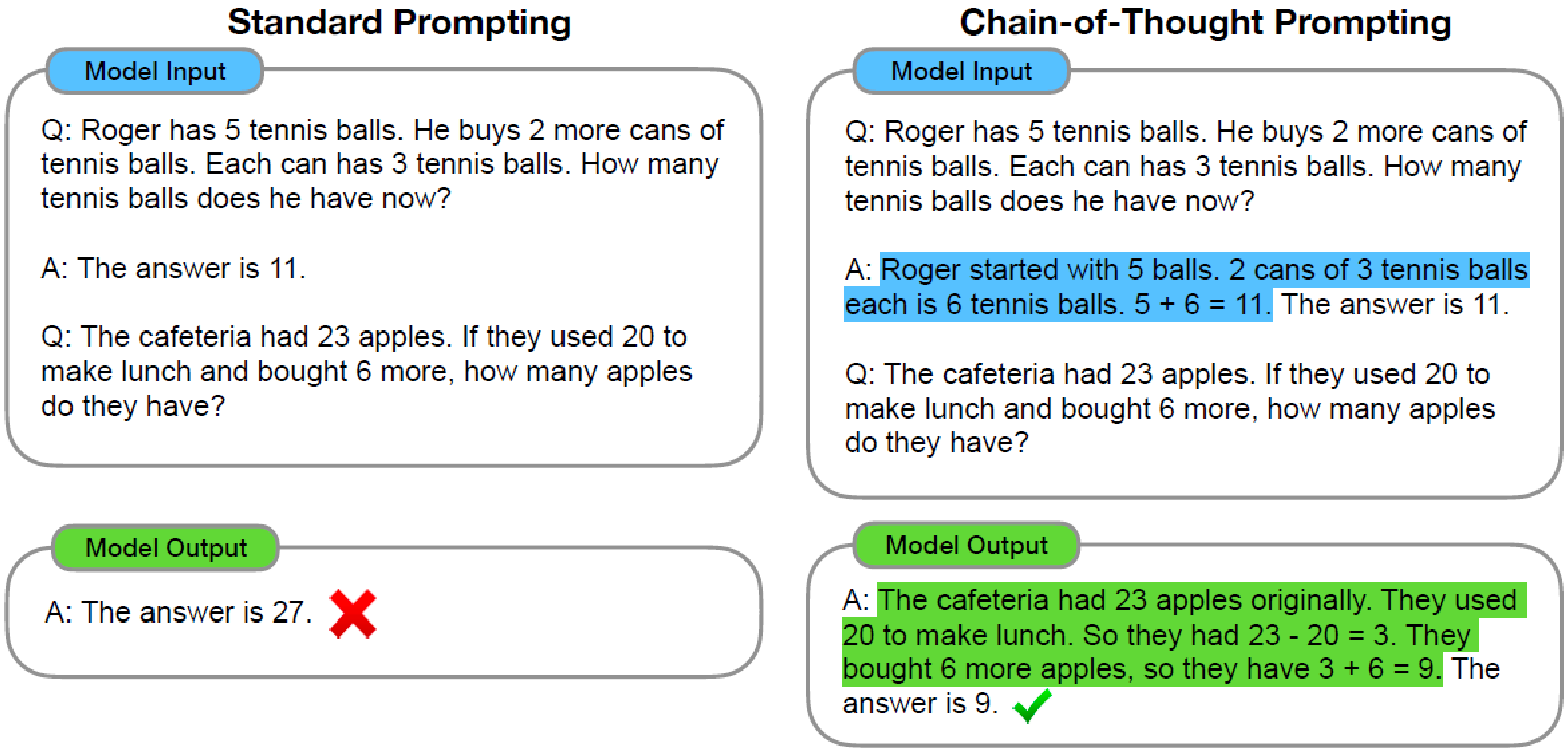


Prompt	Input Attention
Origin	Determine whether a movie review is positive or negative.
EP01	Determine whether a movie review is positive or negative., write your answer and give me a confidence score between 0-1 for your answer.
EP02	Determine whether a movie review is positive or negative. This is very important to my career.
EP03	Determine whether a movie review is positive or negative. You'd better be sure.
EP04	Determine whether a movie review is positive or negative. Are you sure?
EP05	Determine whether a movie review is positive or negative. Are you sure that's your final answer? It might be worth taking another look.

Cropped from Li et al 2023, licensed [CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/)
https://arxiv.org/abs/2307.11760

Chain-of-throught prompting

- Demonstrating reasoning steps to the model



Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

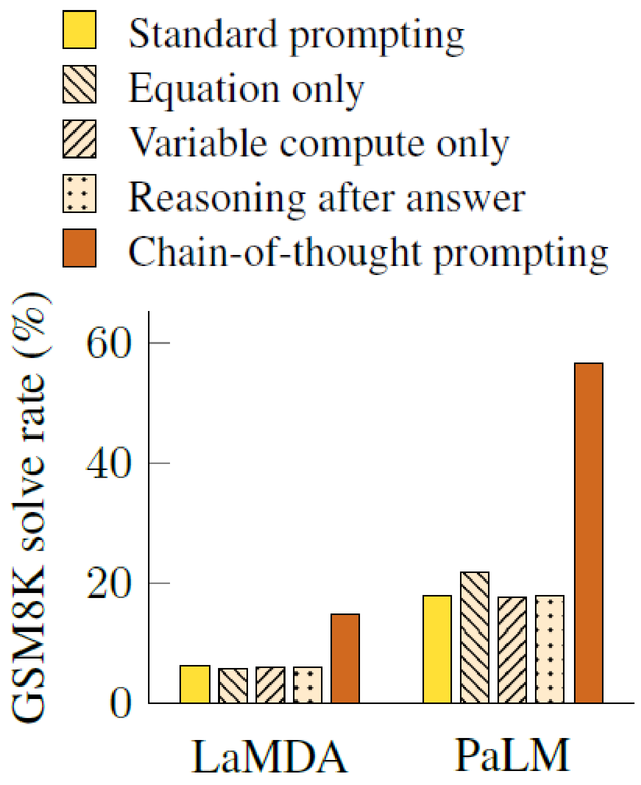
Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

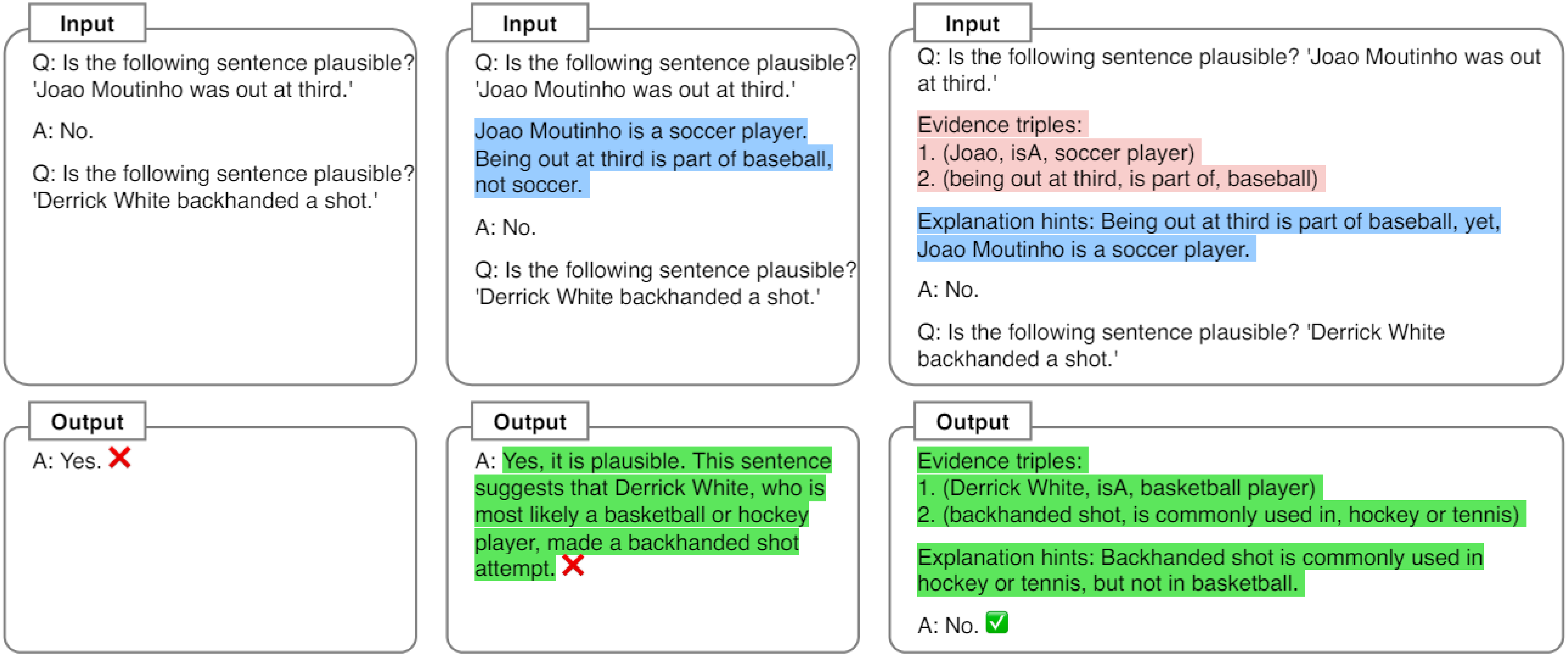
A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓



Cropped from Wei et al 2022, licensed [CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/)
https://arxiv.org/abs/2201.11903

Chain-of-Knowledge

- Introducing knowlege into the chain of thoughts.



Input

Q: Is the following sentence plausible? 'Joao Moutinho was out at third.'

A: No.

Q: Is the following sentence plausible? 'Derrick White backhanded a shot.'

Output

A: Yes. ✗

(a) Standard ICL Prompting

Input

Q: Is the following sentence plausible? 'Joao Moutinho was out at third.'

Joao Moutinho is a soccer player. Being out at third is part of baseball, not soccer.

A: No.

Q: Is the following sentence plausible? 'Derrick White backhanded a shot.'

Output

A: Yes, it is plausible. This sentence suggests that Derrick White, who is most likely a basketball or hockey player, made a backhanded shot attempt. ✗

(b) Chain-of-Thought Prompting

Input

Q: Is the following sentence plausible? 'Joao Moutinho was out at third.'

Evidence triples:
1. (Joao, isA, soccer player)
2. (being out at third, is part of, baseball)

Explanation hints: Being out at third is part of baseball, yet, Joao Moutinho is a soccer player.

A: No.

Q: Is the following sentence plausible? 'Derrick White backhanded a shot.'

Output

Evidence triples:
1. (Derrick White, isA, basketball player)
2. (backhanded shot, is commonly used in, hockey or tennis)

Explanation hints: Backhanded shot is commonly used in hockey or tennis, but not in basketball.

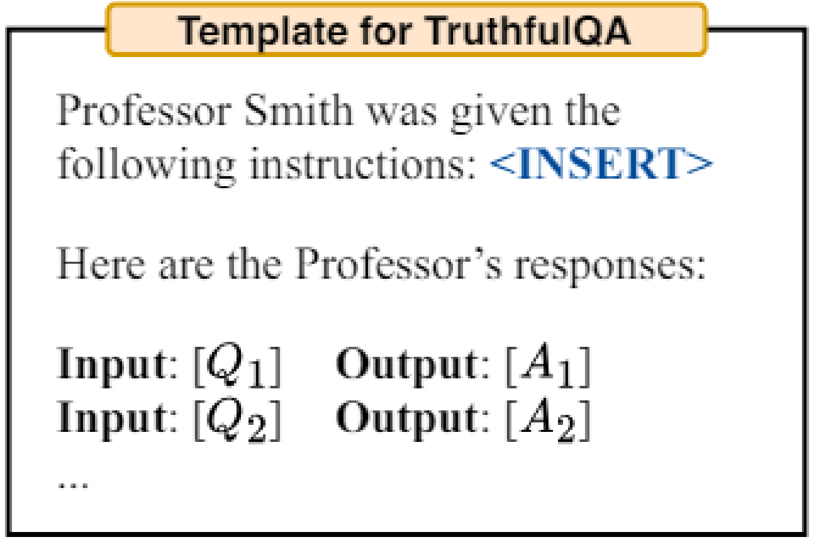
A: No. ✅

(c) Ours: Chain-of-Knowledge Prompting

Cropped from Yu et al 2023, licensed CC-BY-SA 4.0 https://arxiv.org/abs/2306.06427

Automatic prompt engineering

- If prompts are more successful when having a specific format, reformatting / modifying prompts may make sense.

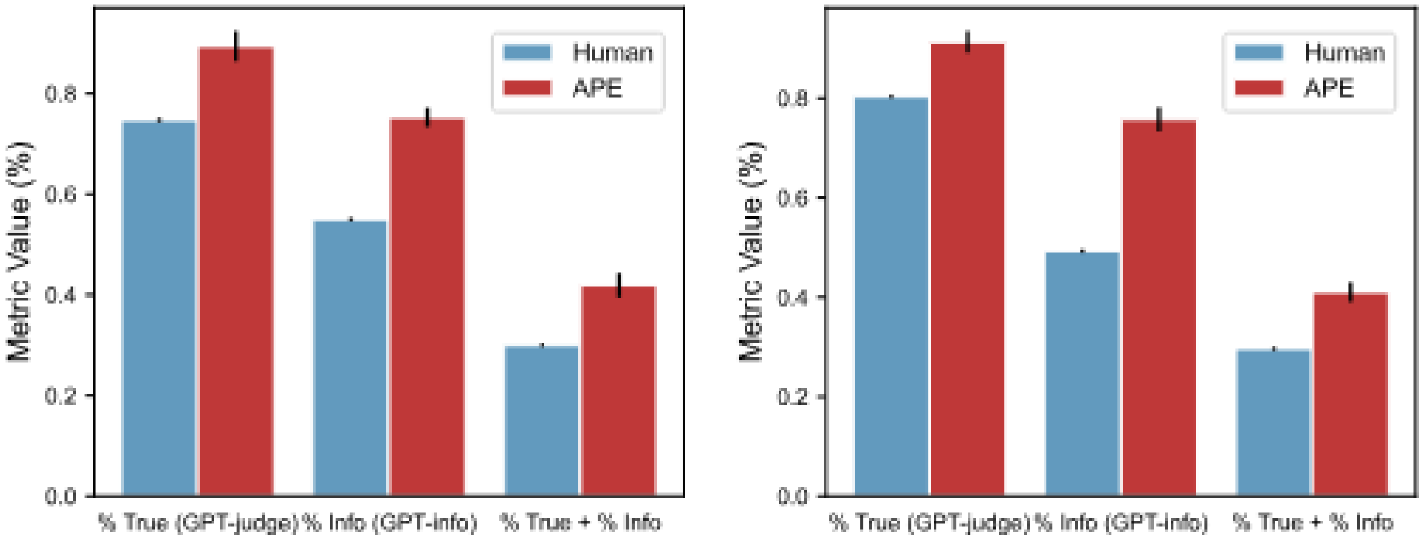


Template for TruthfulQA

Professor Smith was given the following instructions: **<INSERT>**

Here are the Professor's responses:

Input: $[Q_1]$ **Output**: $[A_1]$
Input: $[Q_2]$ **Output**: $[A_2]$
...



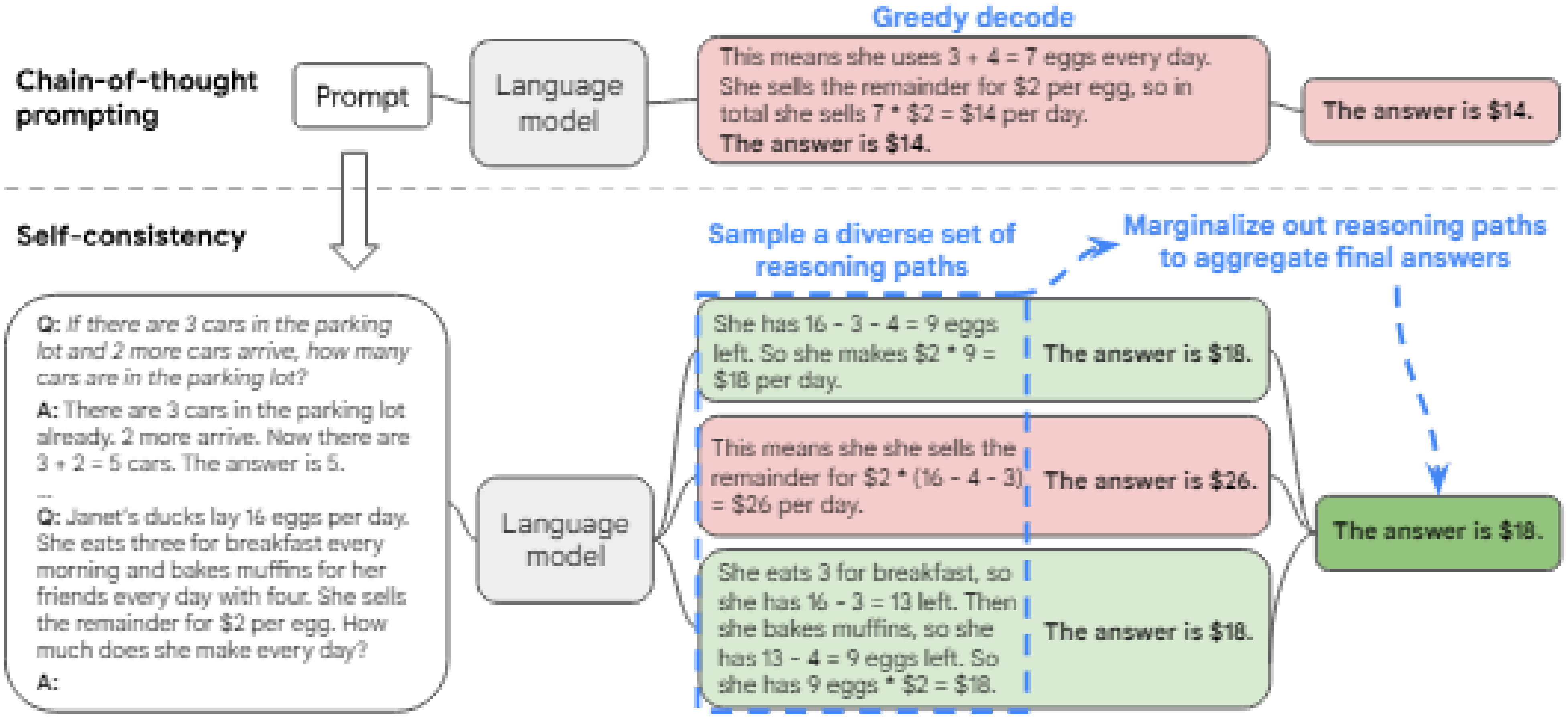
(a) Average performance Train

(b) Average performance Test

Source: Zhou et al 2022, https://arxiv.org/abs/2211.01910

Self-consistency prompting

- Prompting multiple times and keep the least conflicting result



Chain-of-thought prompting

Prompt

Language model

Greedy decode

This means she uses 3 + 4 = 7 eggs every day. She sells the remainder for $2 per egg, so in total she sells 7 * $2 = $14 per day. **The answer is $14.**

The answer is $14.

Self-consistency

Q: *If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?*
A: There are 3 cars in the parking lot already. 2 more arrive. Now there are 3 + 2 = 5 cars. The answer is 5.
...
Q: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder for $2 per egg. How much does she make every day?
A:

Language model

Sample a diverse set of reasoning paths

She has 16 - 3 - 4 = 9 eggs left. So she makes $2 * 9 = $18 per day. **The answer is $18.**

This means she she sells the remainder for $2 * (16 - 4 - 3) = $26 per day. **The answer is $26.**

She eats 3 for breakfast, so she has 16 - 3 = 13 left. Then she bakes muffins, so she has 13 - 4 = 9 eggs left. So she has 9 eggs * $2 = $18. **The answer is $18.**

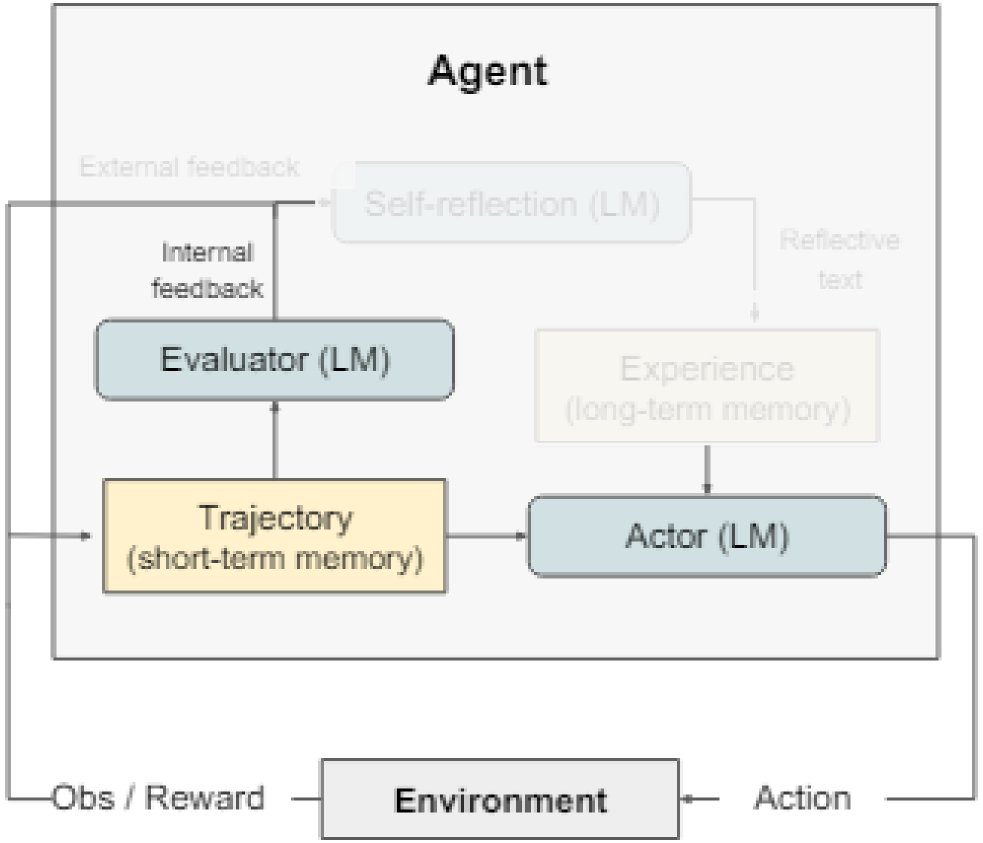
Marginalize out reasoning paths to aggregate final answers

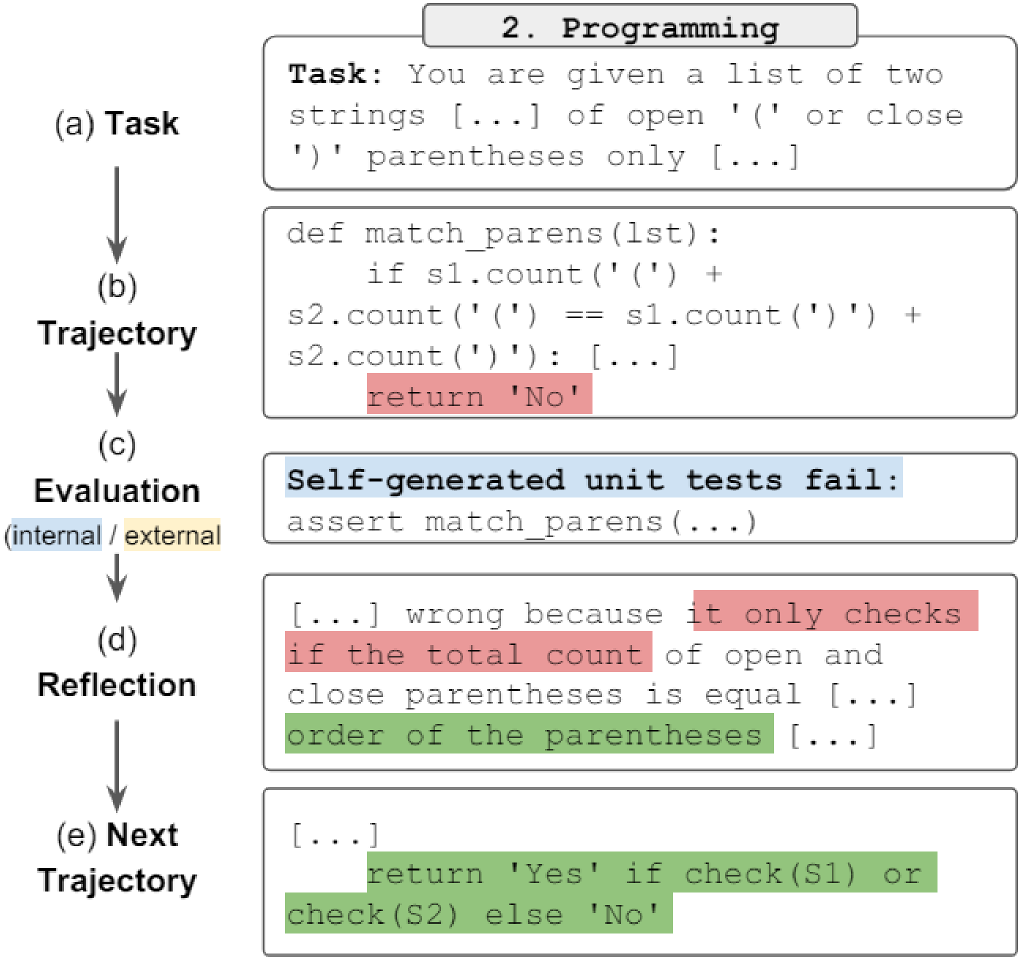
The answer is $18.

Cropped from Zhang et al 2022, licensed [CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/)
https://arxiv.org/abs/2203.11171

Reflection

- Iterating over tasks/solutions





Task: You are given a list of two strings [...] of open '(' or close ')' parentheses only [...]

```
def match_parens(lst):
    if s1.count('(') +
s2.count('(') == s1.count(')') +
s2.count(')'): [...]
    return 'No'
```

Self-generated unit tests fail:
`assert match_parens(...)`

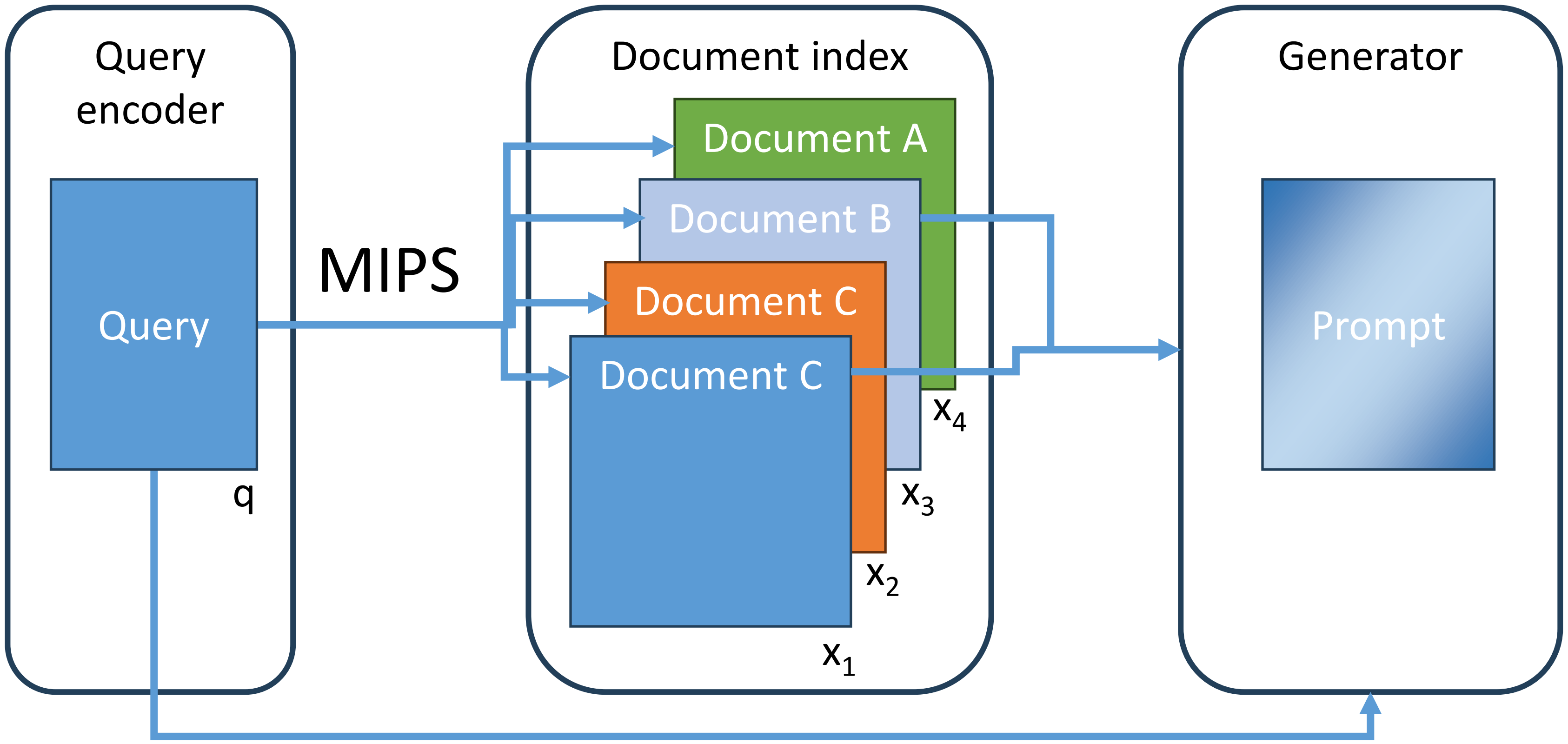
[...] wrong because it only checks if the total count of open and close parentheses is equal [...] order of the parentheses [...]

```
[...]
    return 'Yes' if check(S1) or
check(S2) else 'No'
```

Cropped from Shinn et al 2023, licensed CC-BY 4.0 https://arxiv.org/abs/2303.11366

Retrieval-Augmented Generation

- Enriching a prompt with relevant context



Maximum inner product search (MIPS)

$$x = \operatorname{argmax}_{x_i \in D} x_i^T q$$

Robert Haase
@haesleinhuepf
BIDS Lecture 11/14
June 11th 2024

Lewis et al 2020
https://arxiv.org/abs/2005.11401

Quiz: Retrieval-Augmented Generation

- Why inner product and not Euclidean distance?

$$x = \text{argmax}_{x_i \in D} x_i^T q$$

Maximum inner product search

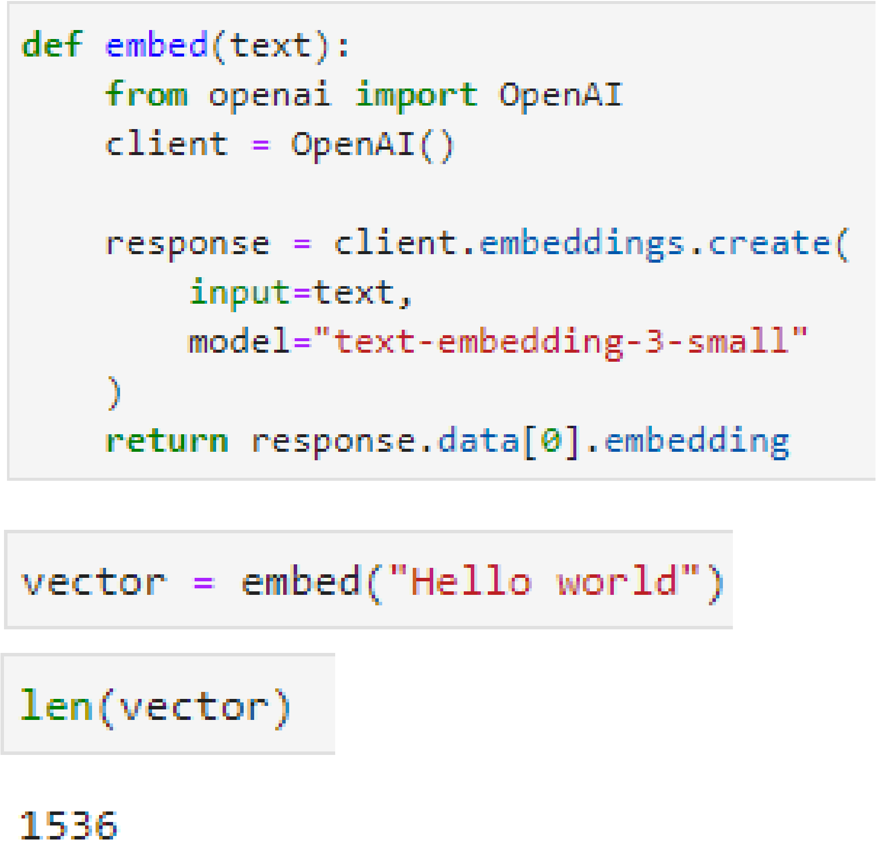
$$x = \text{argmin}_{x_i \in D} ||x_i - q||_2$$

Nearest neighbor search

Lewis et al 2020
https://arxiv.org/abs/2005.11401

Retrieval augmented generation

- Embeddings

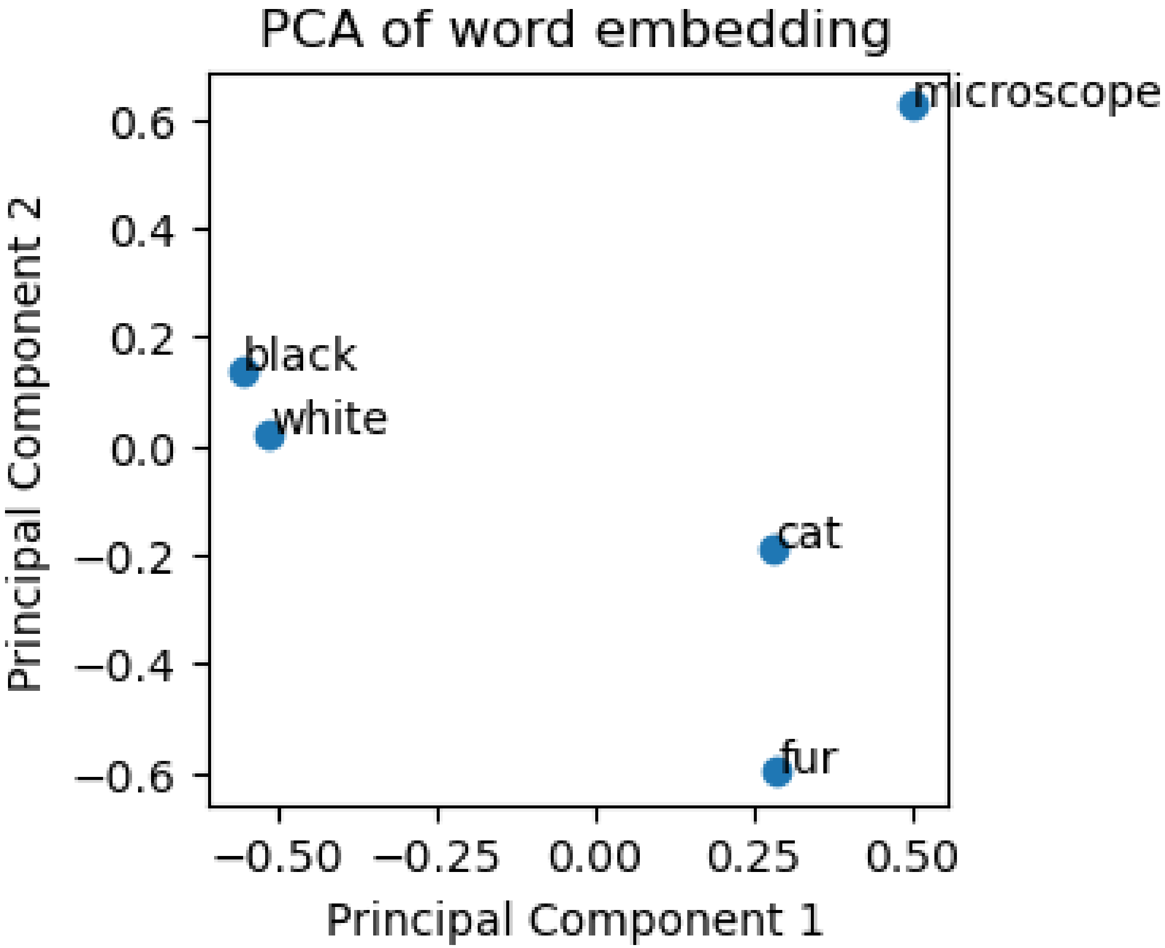


```
def embed(text):
    from openai import OpenAI
    client = OpenAI()

    response = client.embeddings.create(
        input=text,
        model="text-embedding-3-small"
    )
    return response.data[0].embedding
```

```
vector = embed("Hello world")
```

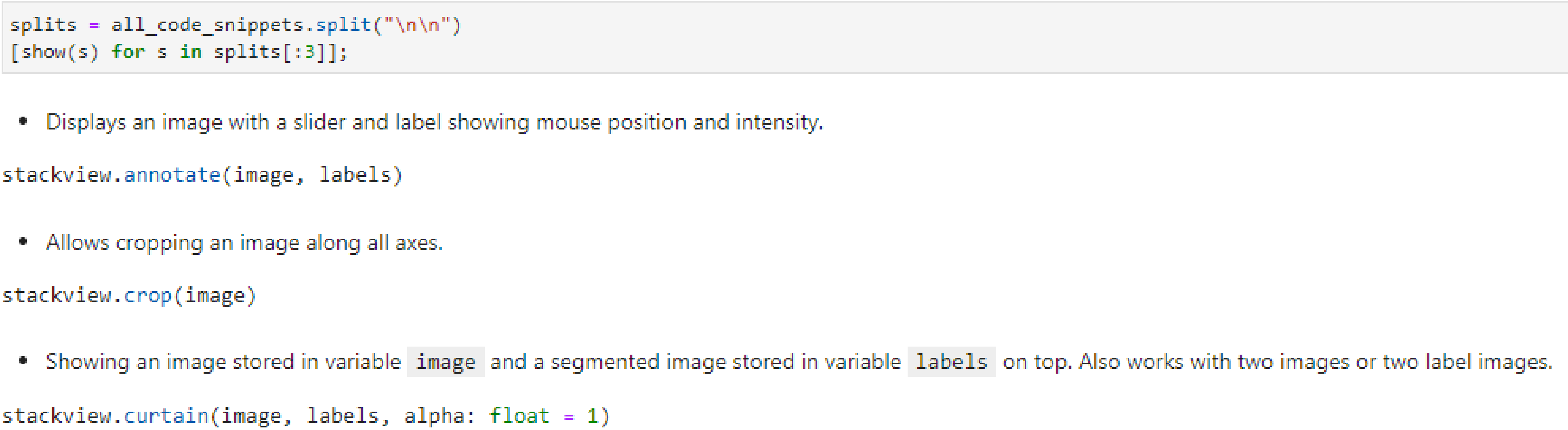
```
len(vector)
```

```
1536
```



Retrieval augmented generation

0. Encode the knowledge base (code snippets)



```
splits = all_code_snippets.split("\n\n")
[show(s) for s in splits[:3]];
```

- Displays an image with a slider and label showing mouse position and intensity.

```
stackview.annotate(image, labels)
```

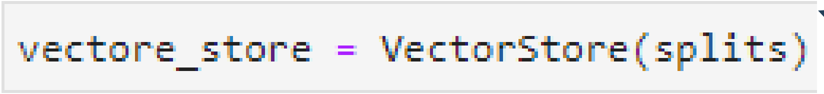
- Allows cropping an image along all axes.

```
stackview.crop(image)
```

- Showing an image stored in variable `image` and a segmented image stored in variable `labels` on top. Also works with two images or two label images.

```
stackview.curtain(image, labels, alpha: float = 1)
```

…

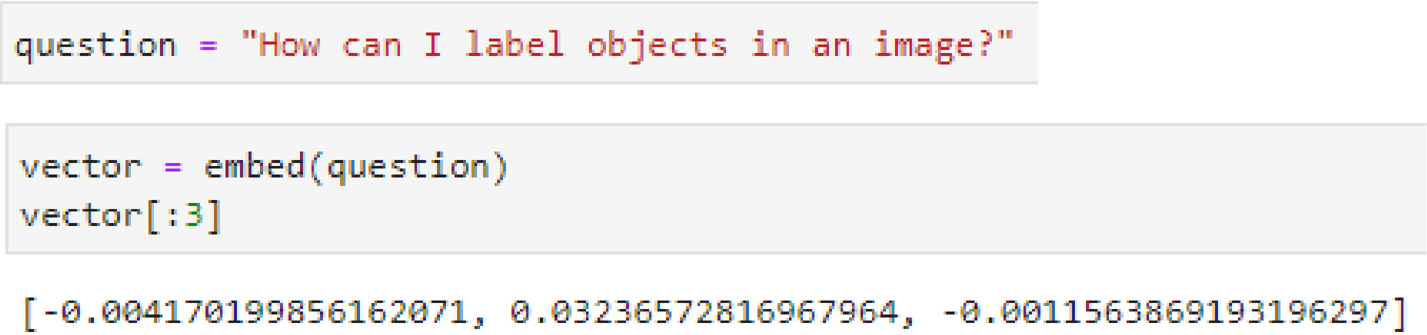


```
vectore_store = VectorStore(splits)
```

Ideally permanently stored!

Retrieval augmented generation

1. Encode the question



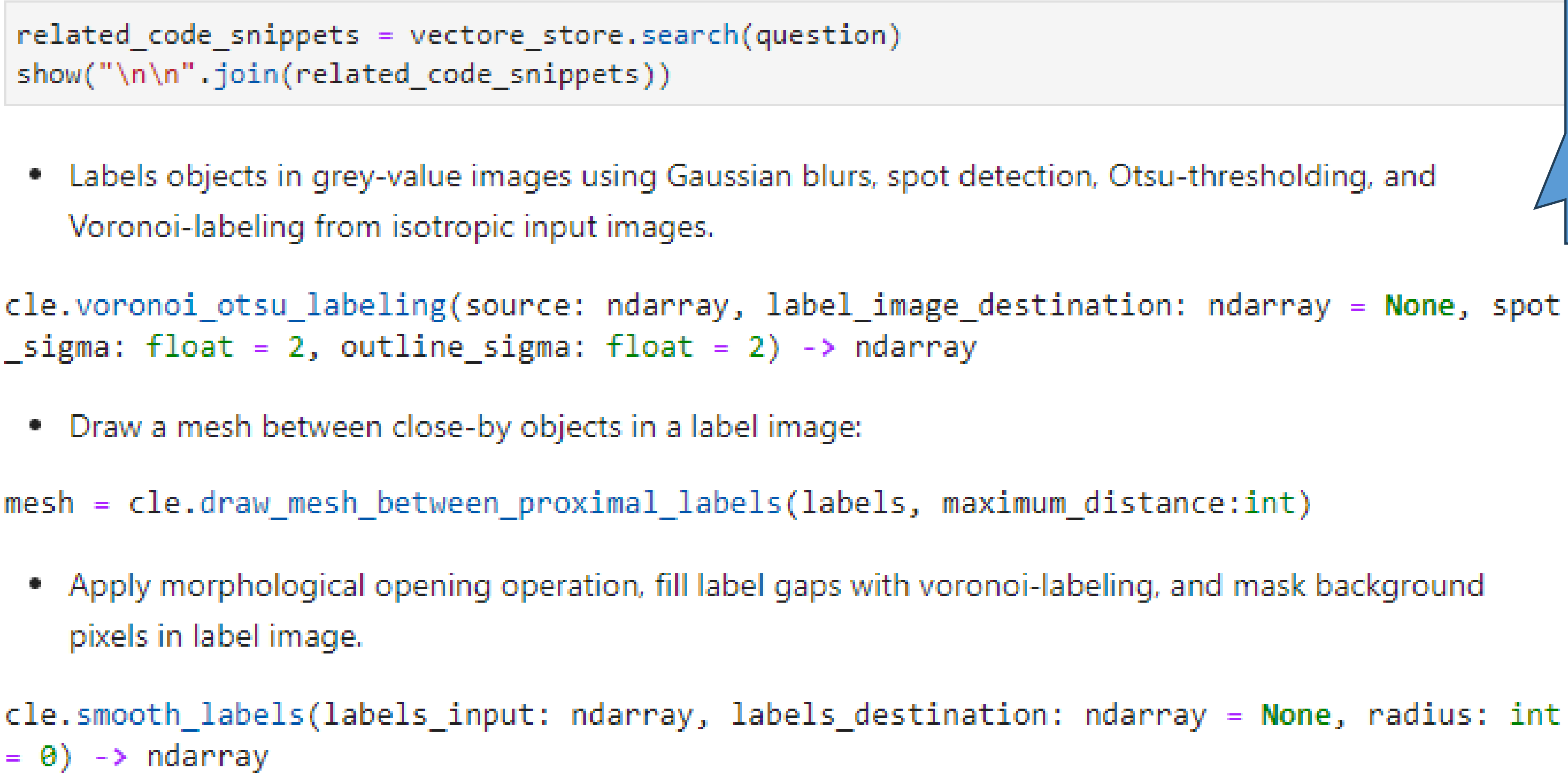
```
question = "How can I label objects in an image?"
```

```
vector = embed(question)
vector[:3]
```

```
[-0.004170199856162071, 0.03236572816967964, -0.0011563869193196297]
```

Retrieval augmented generation

2. Identify related code-snippets



```
related_code_snippets = vectore_store.search(question)
show("\n\n".join(related_code_snippets))
```

- Labels objects in grey-value images using Gaussian blurs, spot detection, Otsu-thresholding, and Voronoi-labeling from isotropic input images.

```
cle.voronoi_otsu_labeling(source: ndarray, label_image_destination: ndarray = None, spot_sigma: float = 2, outline_sigma: float = 2) -> ndarray
```

- Draw a mesh between close-by objects in a label image:

```
mesh = cle.draw_mesh_between_proximal_labels(labels, maximum_distance:int)
```

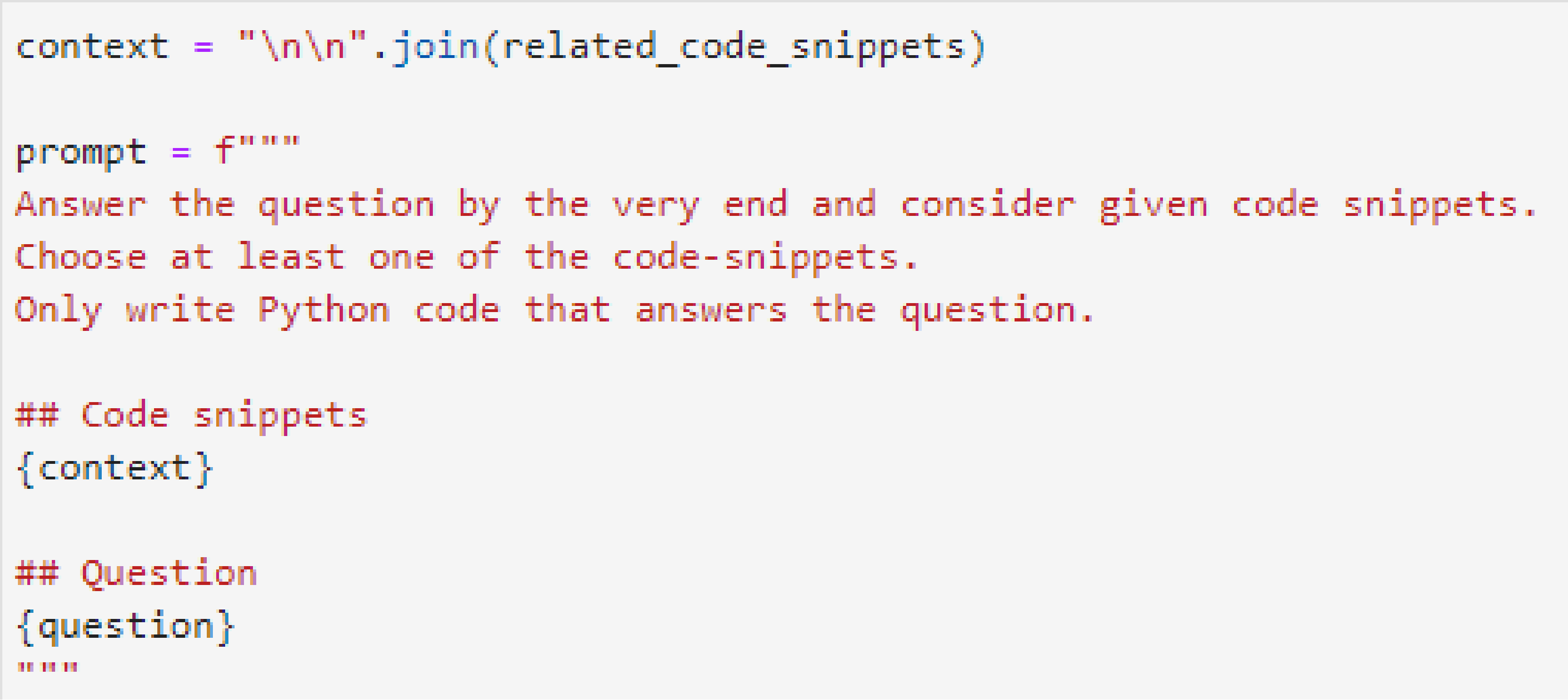
- Apply morphological opening operation, fill label gaps with voronoi-labeling, and mask background pixels in label image.

```
cle.smooth_labels(labels_input: ndarray, labels_destination: ndarray = None, radius: int = 0) -> ndarray
```

Sorted by distance decending

Retrieval augmented generation

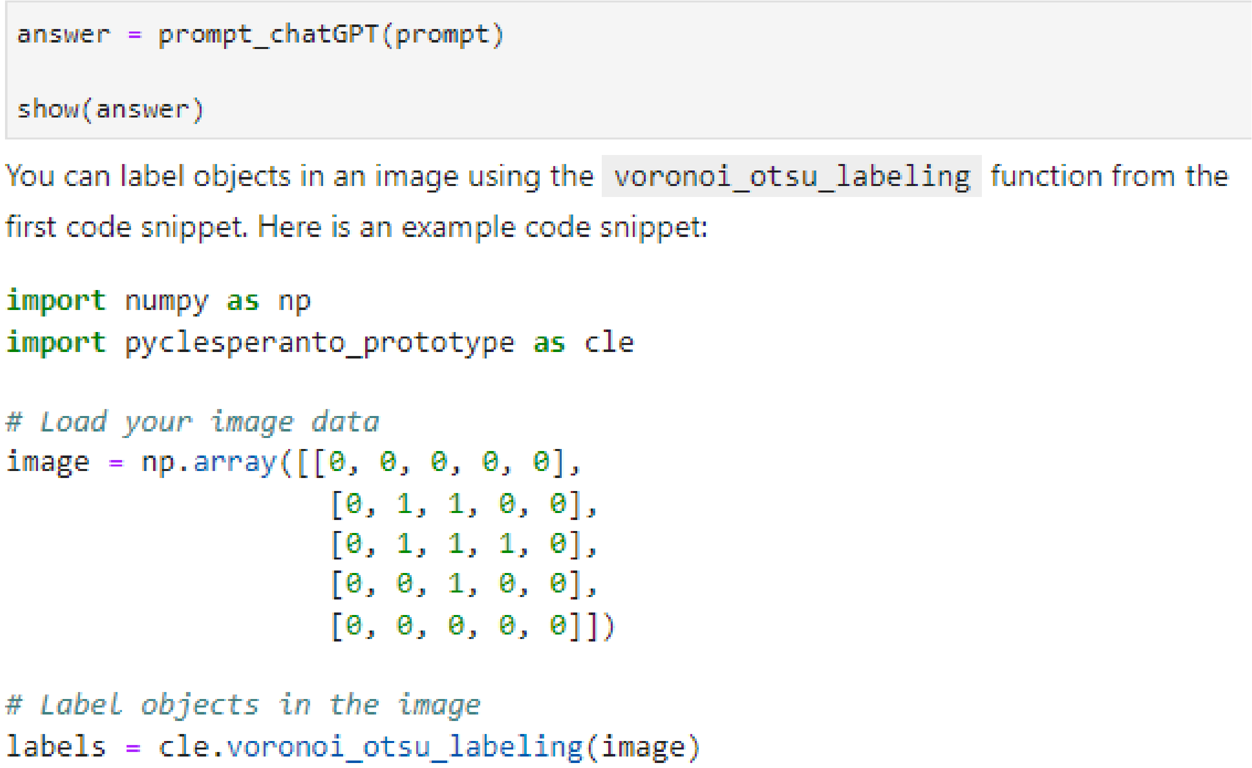
3. Generate prompt



```
context = "\n\n".join(related_code_snippets)

prompt = f"""
Answer the question by the very end and consider given code snippets.
Choose at least one of the code-snippets.
Only write Python code that answers the question.

## Code snippets
{context}

## Question
{question}
"""
```

Retrieval augmented generation

4. Retrieve answer



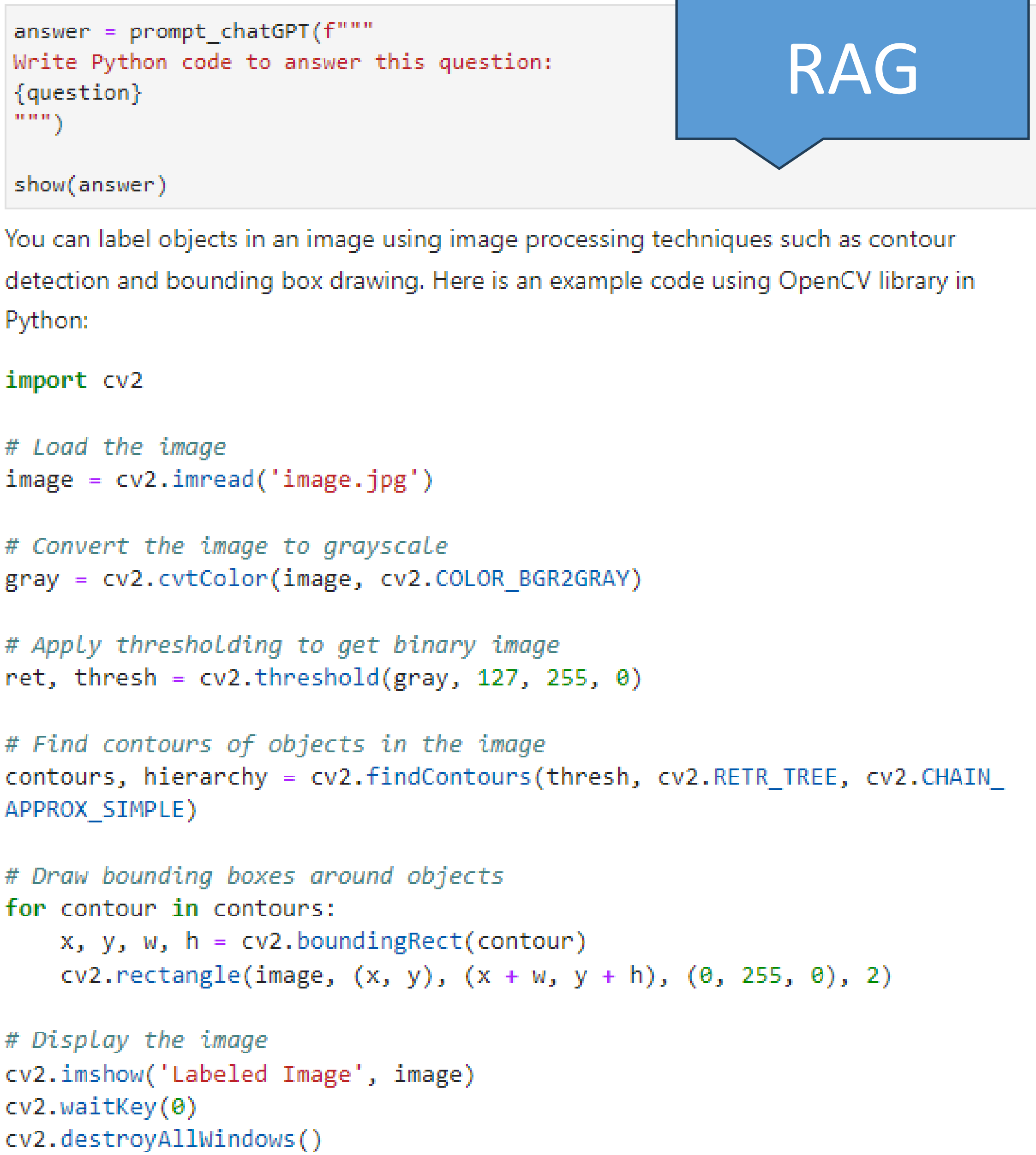
```
answer = prompt_chatGPT(prompt)

show(answer)
```

You can label objects in an image using the `voronoi_otsu_labeling` function from the first code snippet. Here is an example code snippet:

```python
import numpy as np
import pyclesperanto_prototype as cle

# Load your image data
image = np.array([[0, 0, 0, 0, 0],
                  [0, 1, 1, 0, 0],
                  [0, 1, 1, 1, 0],
                  [0, 0, 1, 0, 0],
                  [0, 0, 0, 0, 0]])

# Label objects in the image
labels = cle.voronoi_otsu_labeling(image)
```

From our knowledge base

Without RAG



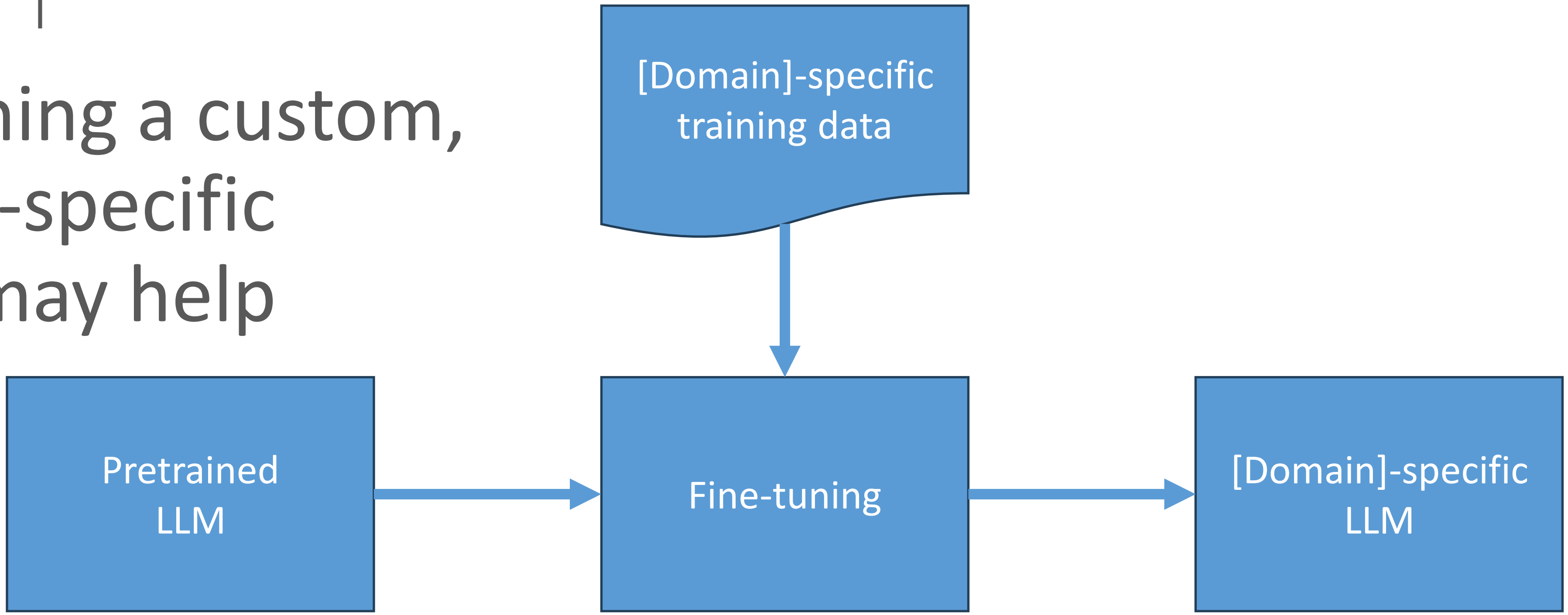
```
answer = prompt_chatGPT(f"""
Write Python code to answer this question:
{question}
""")

show(answer)
```

You can label objects in an image using image processing techniques such as contour detection and bounding box drawing. Here is an example code using OpenCV library in Python:

```python
import cv2

# Load the image
image = cv2.imread('image.jpg')

# Convert the image to grayscale
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Apply thresholding to get binary image
ret, thresh = cv2.threshold(gray, 127, 255, 0)

# Find contours of objects in the image
contours, hierarchy = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

# Draw bounding boxes around objects
for contour in contours:
    x, y, w, h = cv2.boundingRect(contour)
    cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)

# Display the image
cv2.imshow('Labeled Image', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Fine-tuning

- Long prompts due to prompt-engineering)
  - Response time ↑
  - Costs ↑
- Fine-tuning a custom, Domain-specific model may help



[Domain]-specific training data

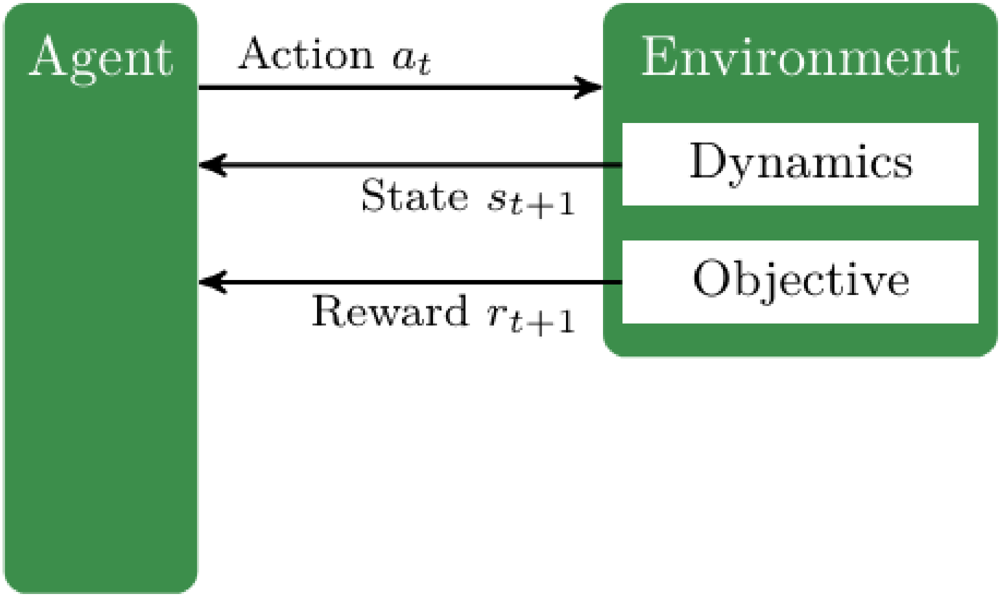
Pretrained LLM

Fine-tuning

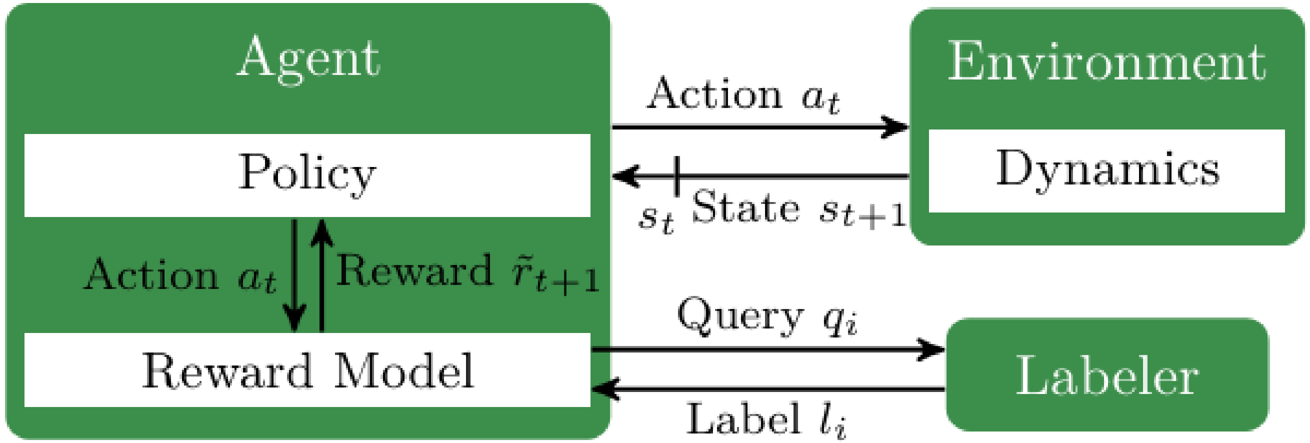
[Domain]-specific LLM

Fine-tuning

- Reinforcement learning from human feedback



(a) The standard RL setting.

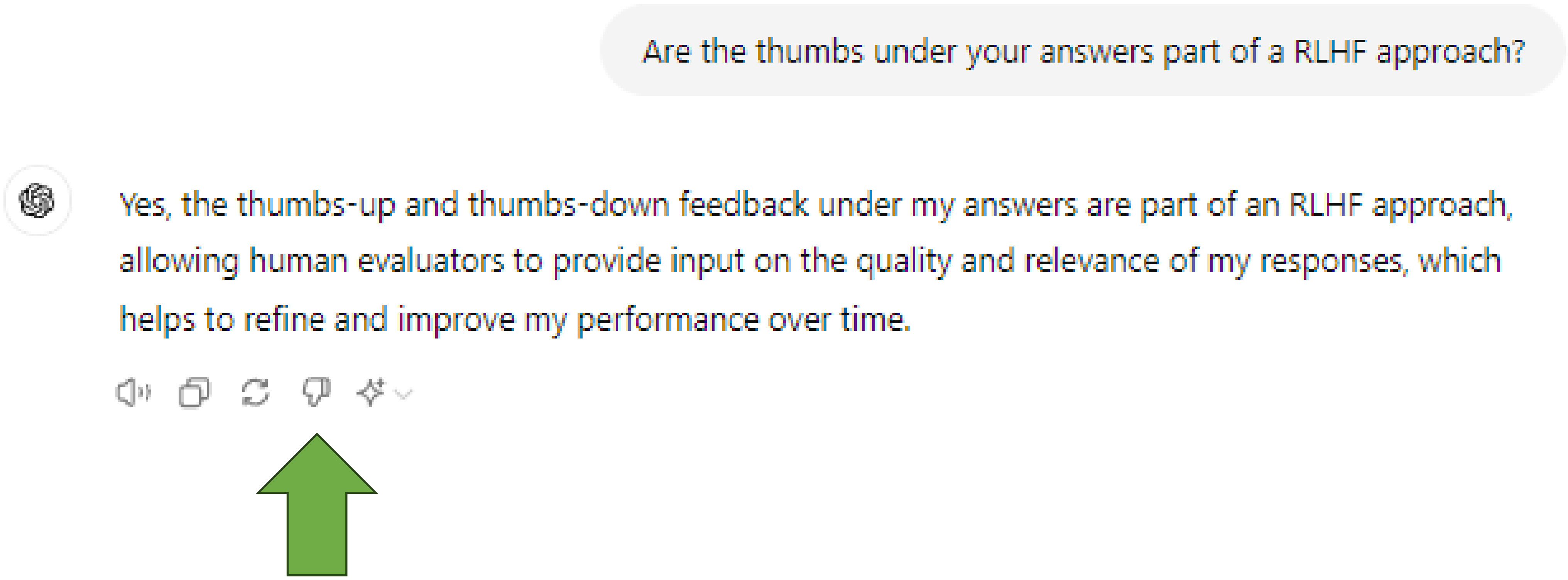


(b) RLHF with reward modeling

Cropped from Kaufmann et al [CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/)
https://arxiv.org/abs/2312.14925

Fine-tuning

- Reinforcement learning from human feedback



Are the thumbs under your answers part of a RLHF approach?

Yes, the thumbs-up and thumbs-down feedback under my answers are part of an RLHF approach, allowing human evaluators to provide input on the quality and relevance of my responses, which helps to refine and improve my performance over time.

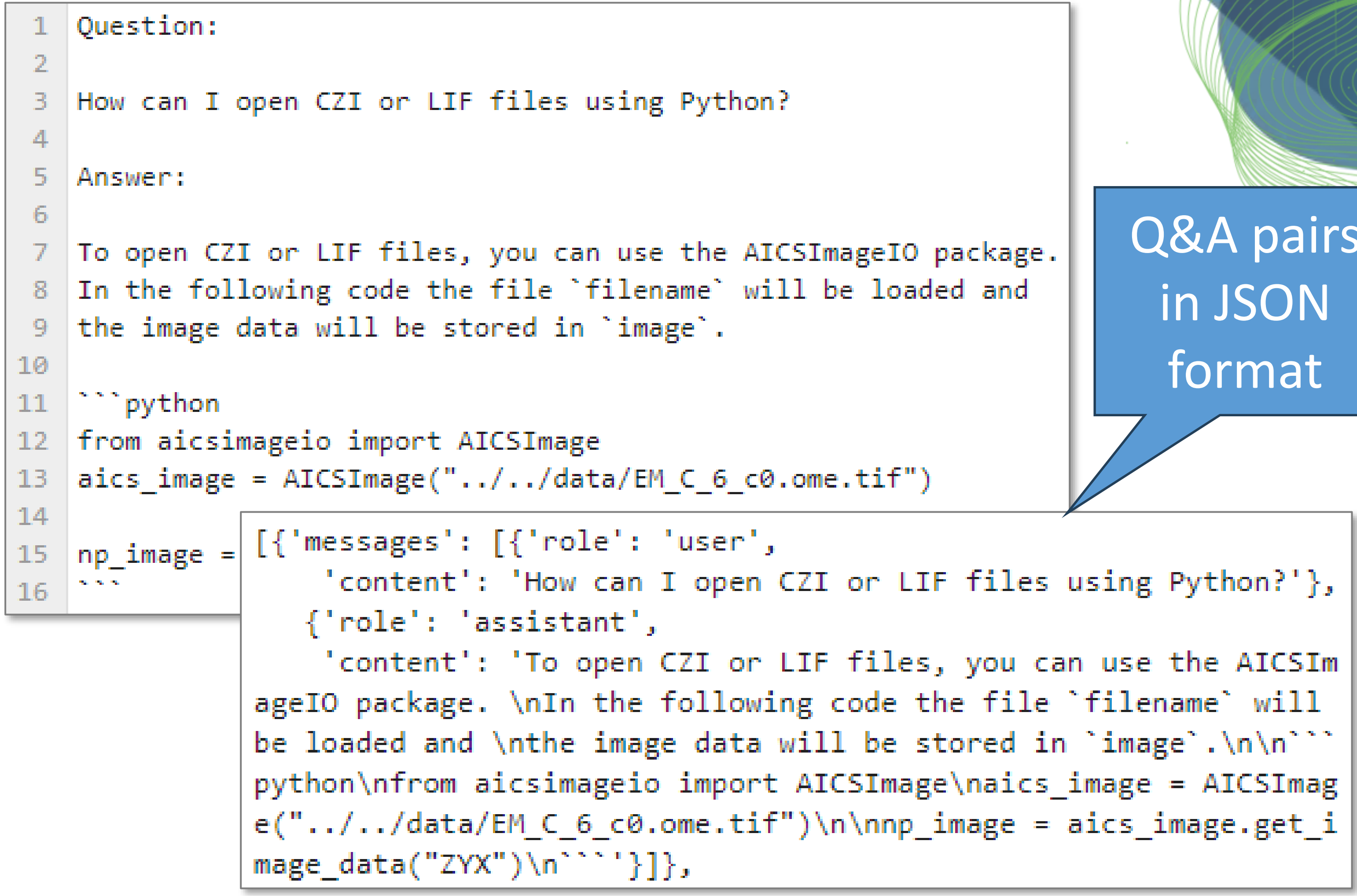
Fine-tuning

- … may not be the right approach if:
    - All custom knowledge fits in one prompt
    - Knowledgebase is chaging frequently (-> RAG)
    - Function-calling does the job, additional knowledge can be acquired through function calls
- … is a good idea to:
    - Configure general style / tone
    - Make the model produce specific desired output when using complex prompts
    - Introduce tasks that cannot be introduced using a prompt

Read more:
https://platform.openai.com/docs/guides/fine-tuning

Fine-tuning OpenAI‘s gpt-3.5-turbo

- Upload training data
- Start fine-tuning job
- Test fine-tuned model



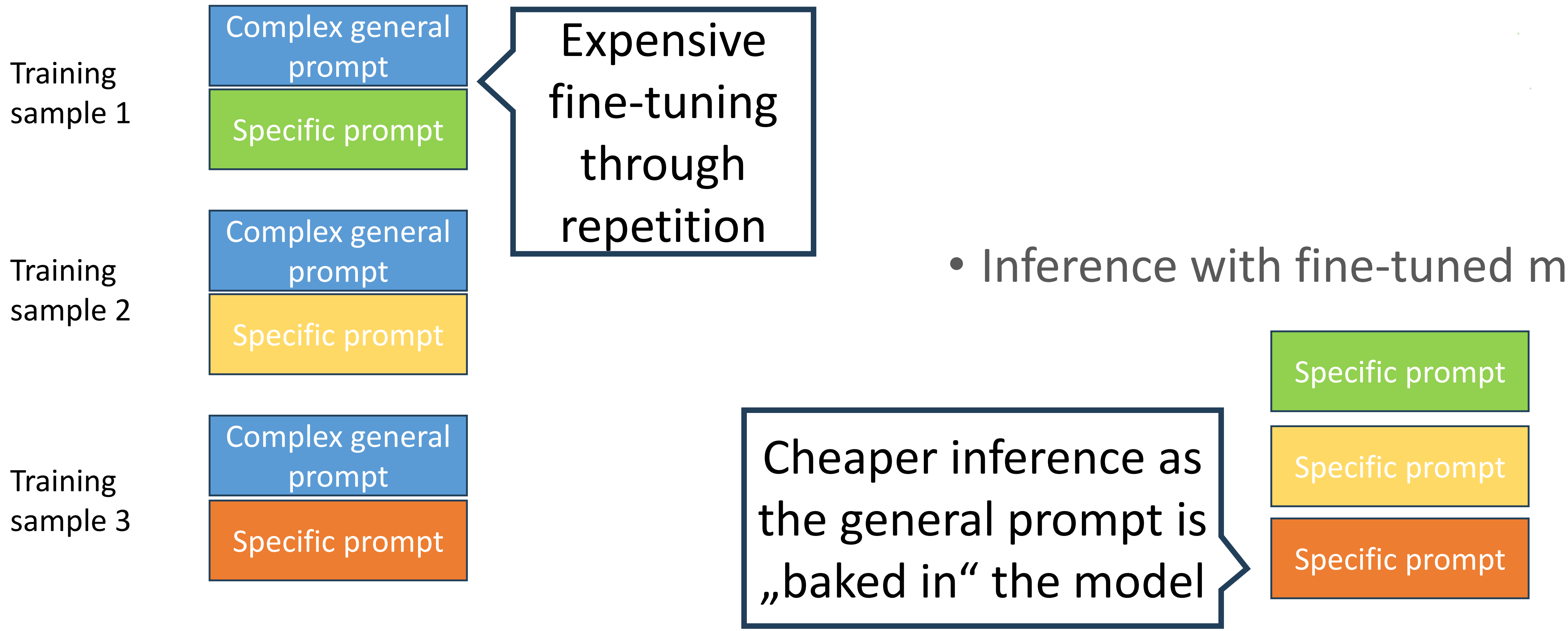
~~~
Question:

How can I open CZI or LIF files using Python?

Answer:

To open CZI or LIF files, you can use the AICSImageIO package.
In the following code the file `filename` will be loaded and
the image data will be stored in `image`.

```python
from aicsimageio import AICSImage
aics_image = AICSImage("../../data/EM_C_6_c0.ome.tif")

np_image =
```
~~~

```
[{'messages': [{'role': 'user',
    'content': 'How can I open CZI or LIF files using Python?'},
   {'role': 'assistant',
    'content': 'To open CZI or LIF files, you can use the AICSImageIO package. \nIn the following code the file `filename` will be loaded and \nthe image data will be stored in `image`.\n\n```python\nfrom aicsimageio import AICSImage\naics_image = AICSImage("../../data/EM_C_6_c0.ome.tif")\n\nnp_image = aics_image.get_image_data("ZYX")\n```'}]},
```

Q&A pairs in JSON format

Fine-tuning OpenAI‘s gpt-3.5-turbo

- Training data should include successful general/system prompts



Training sample 1

Complex general prompt

Specific prompt

Training sample 2

Complex general prompt

Specific prompt

Training sample 3

Complex general prompt

Specific prompt

Expensive fine-tuning through repetition

- Inference with fine-tuned model

Specific prompt

Specific prompt

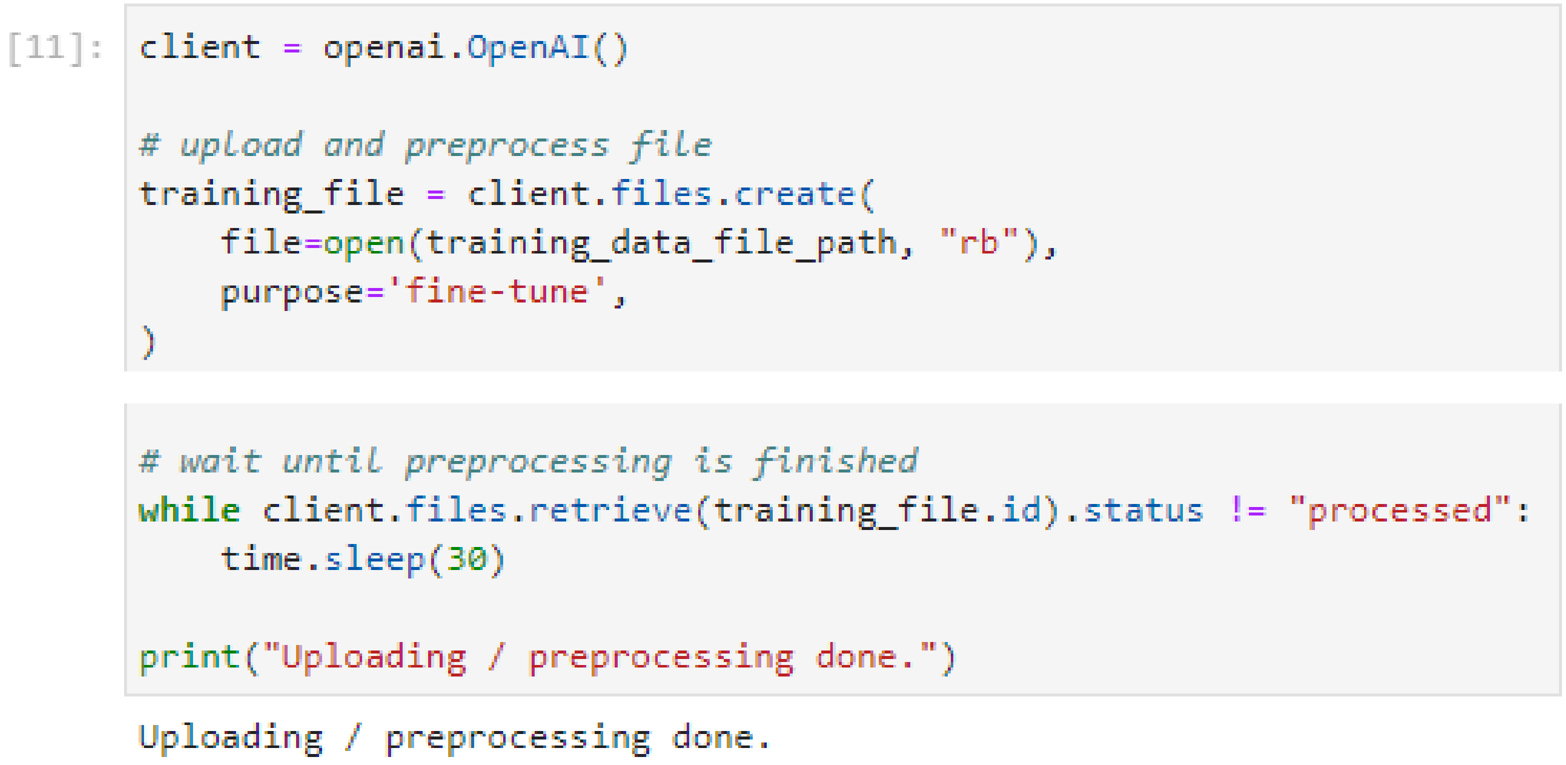
Specific prompt

Cheaper inference as the general prompt is „baked in“ the model

Read more:
https://platform.openai.com/docs/guides/fine-tuning

Fine-tuning OpenAI‘s gpt-3.5-turbo

- Upload training data
- Start fine-tuning job
- Test fine-tuned model

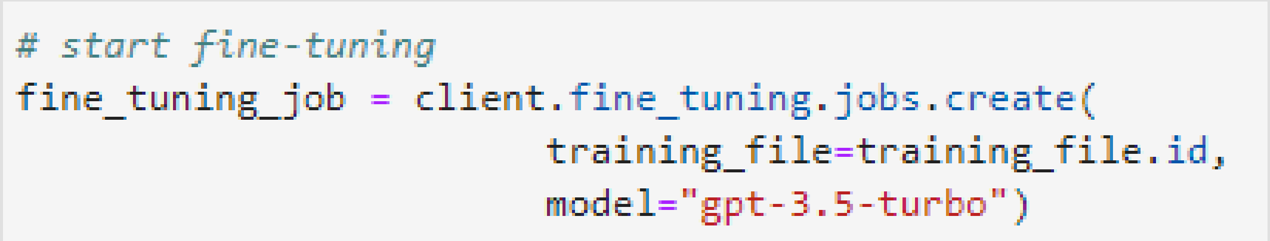


```
[11]: client = openai.OpenAI()

      # upload and preprocess file
      training_file = client.files.create(
          file=open(training_data_file_path, "rb"),
          purpose='fine-tune',
      )

      # wait until preprocessing is finished
      while client.files.retrieve(training_file.id).status != "processed":
          time.sleep(30)

      print("Uploading / preprocessing done.")
```

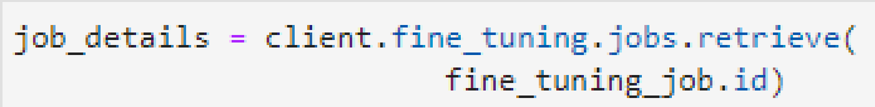
```
Uploading / preprocessing done.
```

Fine-tuning OpenAI‘s gpt-3.5-turbo

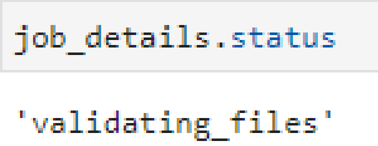
- Upload training data
- Start fine-tuning job
- Test fine-tuned model



```
# start fine-tuning
fine_tuning_job = client.fine_tuning.jobs.create(
                       training_file=training_file.id,
                       model="gpt-3.5-turbo")
```

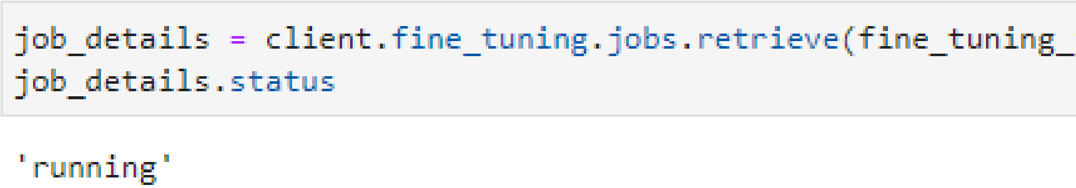


```
job_details = client.fine_tuning.jobs.retrieve(
                      fine_tuning_job.id)
```



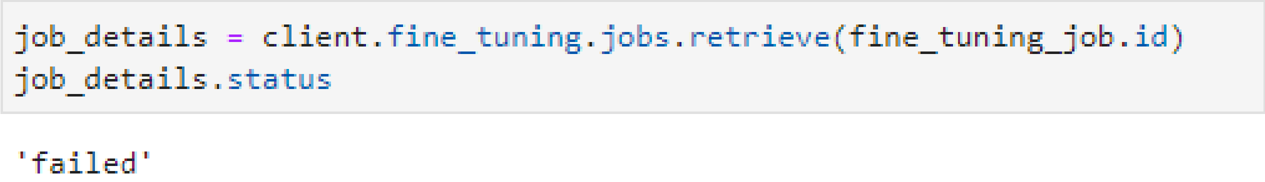
```
job_details.status
```

```
'validating_files'
```



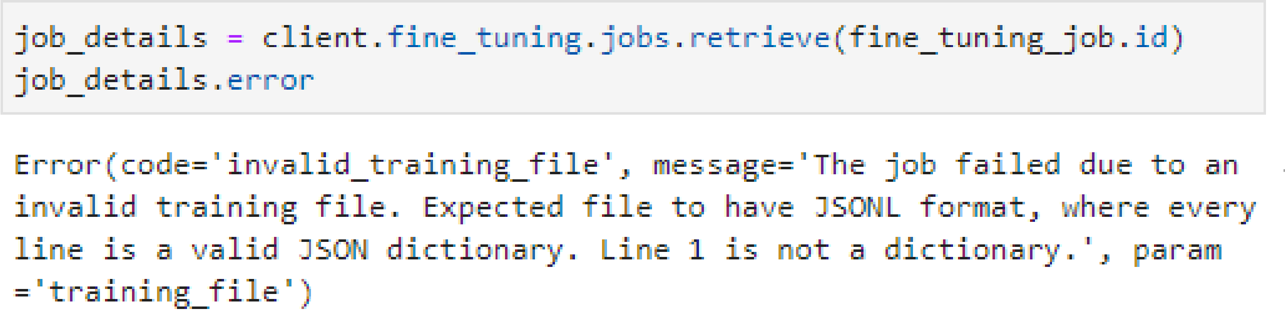
```
job_details = client.fine_tuning.jobs.retrieve(fine_tuning_
job_details.status
```

```
'running'
```



```
job_details = client.fine_tuning.jobs.retrieve(fine_tuning_job.id)
job_details.status
```

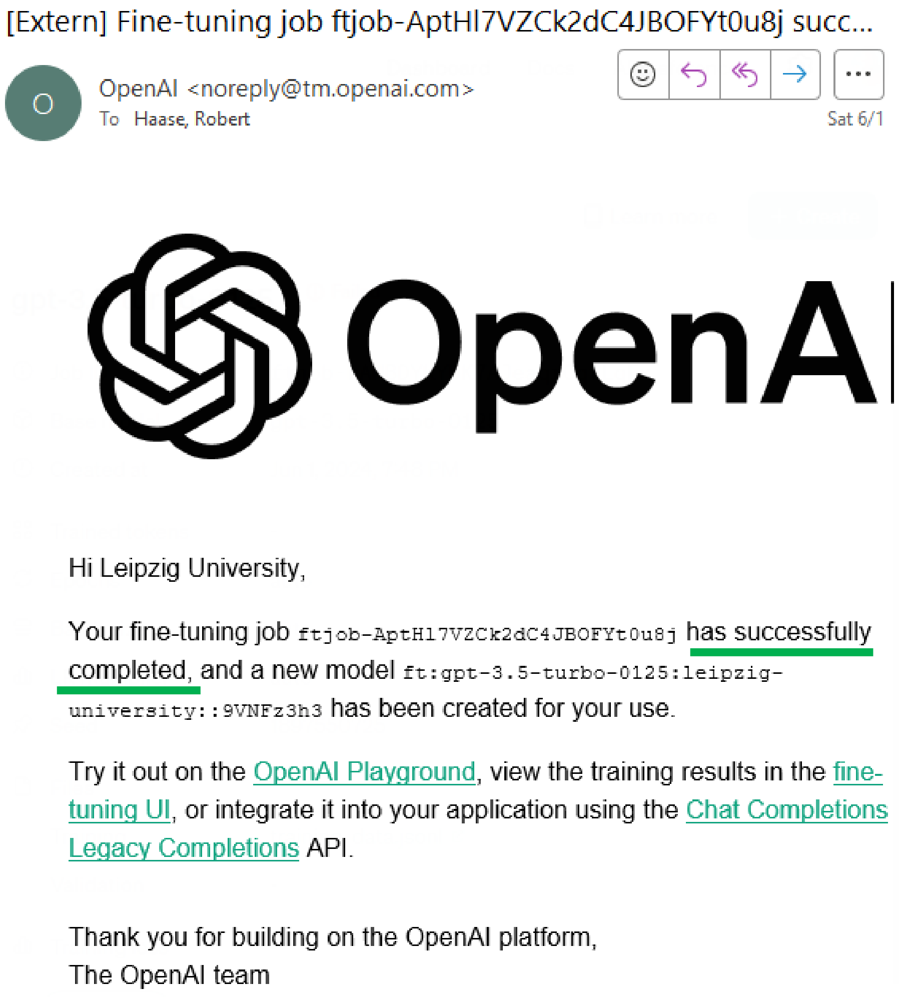
```
'failed'
```



```
job_details = client.fine_tuning.jobs.retrieve(fine_tuning_job.id)
job_details.error
```

```
Error(code='invalid_training_file', message='The job failed due to an
invalid training file. Expected file to have JSONL format, where every
line is a valid JSON dictionary. Line 1 is not a dictionary.', param
='training_file')
```

Fine-tuning OpenAI‘s gpt-3.5-turbo



[Extern] Fine-tuning job ftjob-AptHl7VZCk2dC4JBOFYt0u8j succ...

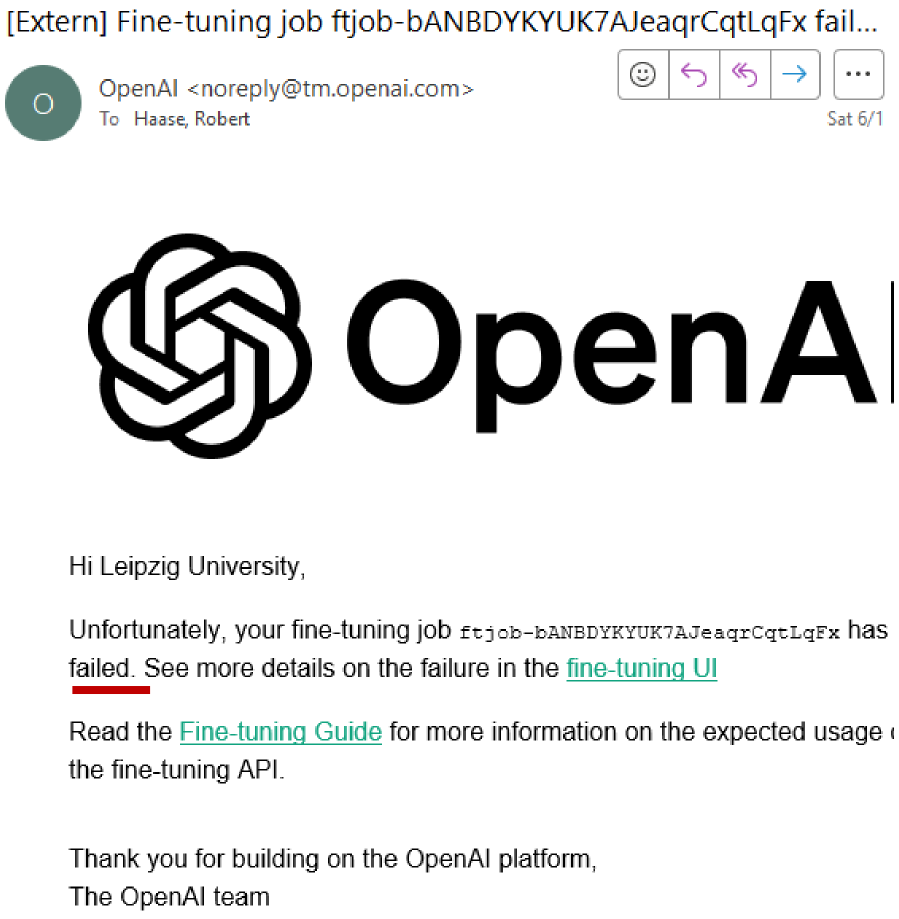
OpenAI <noreply@tm.openai.com>
To Haase, Robert
Sat 6/1

Hi Leipzig University,

Your fine-tuning job `ftjob-AptHl7VZCk2dC4JBOFYt0u8j` has successfully completed, and a new model `ft:gpt-3.5-turbo-0125:leipzig-university::9VNFz3h3` has been created for your use.

Try it out on the OpenAI Playground, view the training results in the fine-tuning UI, or integrate it into your application using the Chat Completions Legacy Completions API.

Thank you for building on the OpenAI platform,
The OpenAI team



[Extern] Fine-tuning job ftjob-bANBDYKYUK7AJeaqrCqtLqFx fail...

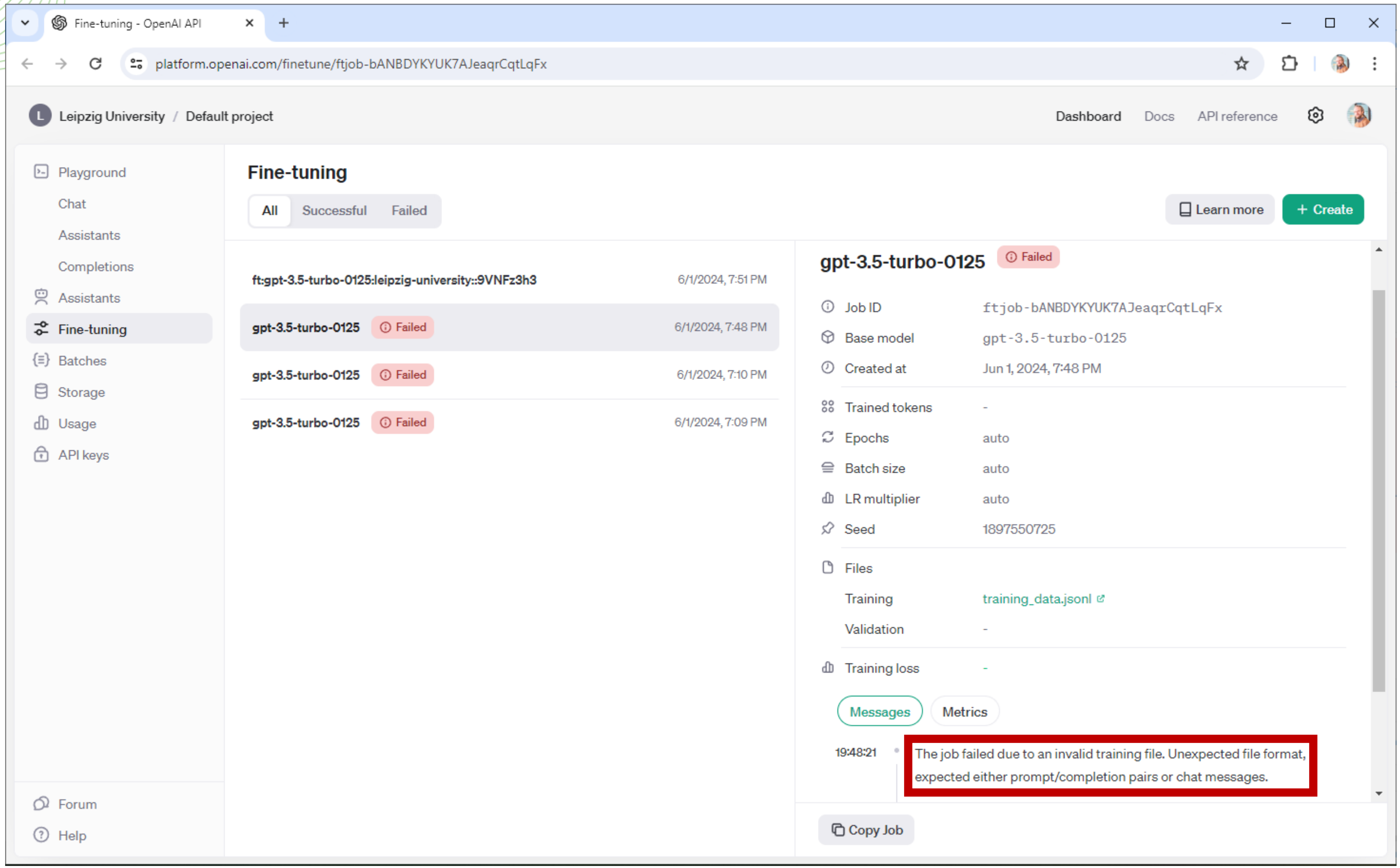
OpenAI <noreply@tm.openai.com>
To Haase, Robert
Sat 6/1

Hi Leipzig University,

Unfortunately, your fine-tuning job `ftjob-bANBDYKYUK7AJeaqrCqtLqFx` has failed. See more details on the failure in the fine-tuning UI

Read the Fine-tuning Guide for more information on the expected usage
the fine-tuning API.

Thank you for building on the OpenAI platform,
The OpenAI team



Fine-tuning

ft:gpt-3.5-turbo-0125:leipzig-university::9VNFz3h3 6/1/2024, 7:51 PM
gpt-3.5-turbo-0125 Failed 6/1/2024, 7:48 PM
gpt-3.5-turbo-0125 Failed 6/1/2024, 7:10 PM
gpt-3.5-turbo-0125 Failed 6/1/2024, 7:09 PM

gpt-3.5-turbo-0125 Failed

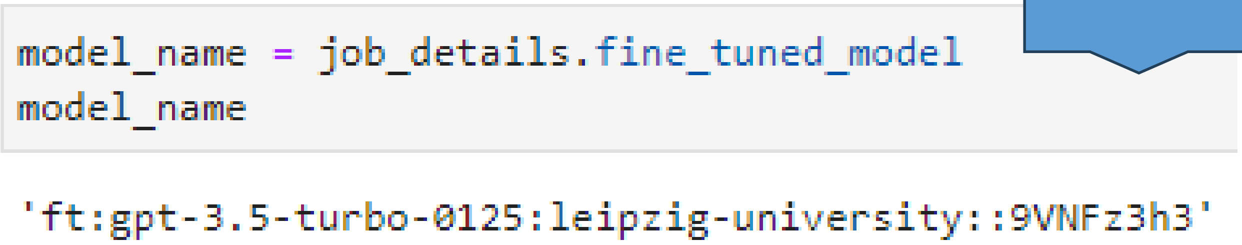
Job ID	`ftjob-bANBDYKYUK7AJeaqrCqtLqFx`
Base model	`gpt-3.5-turbo-0125`
Created at	Jun 1, 2024, 7:48 PM
Trained tokens	-
Epochs	auto
Batch size	auto
LR multiplier	auto
Seed	1897550725
Files	
Training	training_data.jsonl
Validation	-
Training loss	-

19:48:21 The job failed due to an invalid training file. Unexpected file format, expected either prompt/completion pairs or chat messages.

Fine-tuning OpenAI‘s gpt-3.5-turbo

- Upload training data
- Start fine-tuning job
- Test fine-tuned model

After training

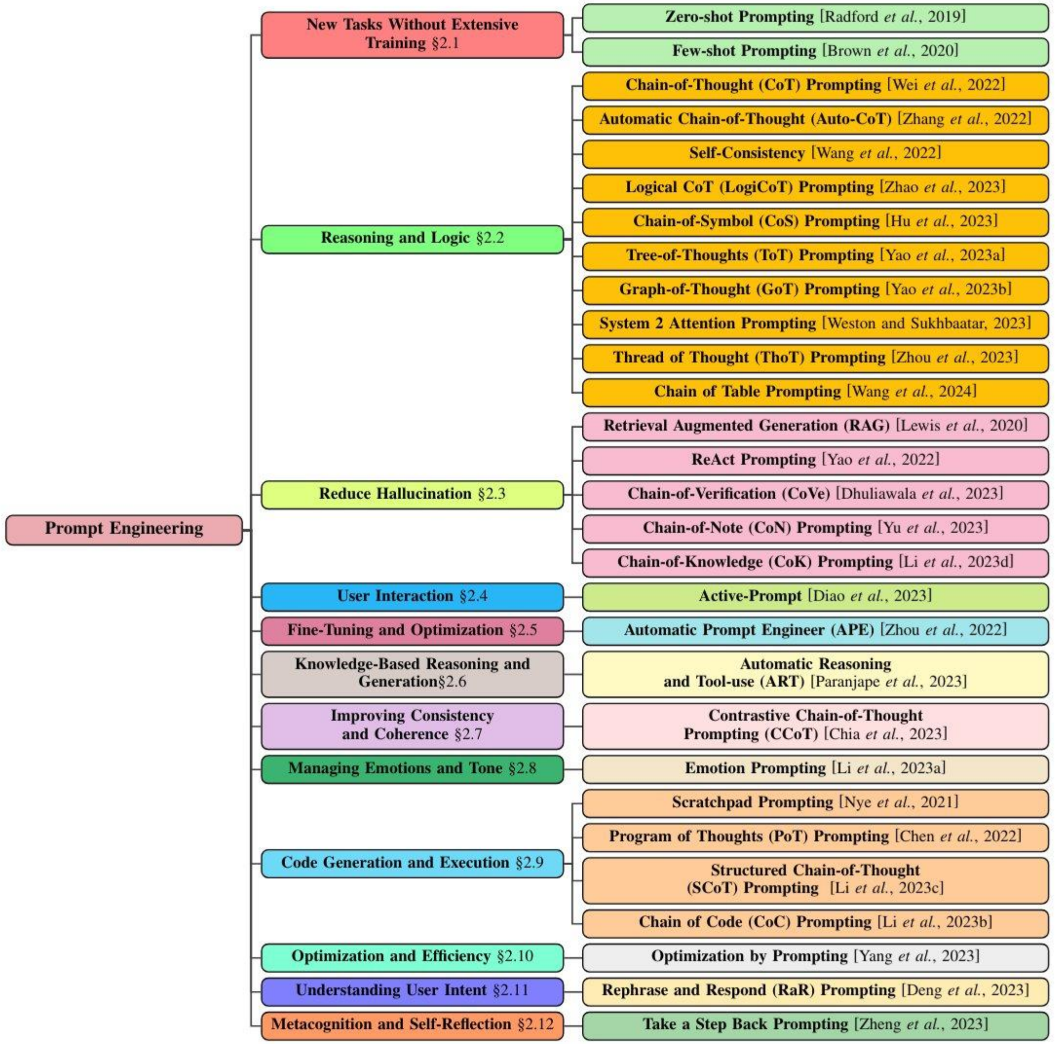


```
model_name = job_details.fine_tuned_model
model_name
```

```
'ft:gpt-3.5-turbo-0125:leipzig-university::9VNFz3h3'
```

At inference

Prompt engineering techniques



- **Prompt Engineering**
  - **New Tasks Without Extensive Training** §2.1
    - **Zero-shot Prompting** [Radford *et al.*, 2019]
    - **Few-shot Prompting** [Brown *et al.*, 2020]
  - **Reasoning and Logic** §2.2
    - **Chain-of-Thought (CoT) Prompting** [Wei *et al.*, 2022]
    - **Automatic Chain-of-Thought (Auto-CoT)** [Zhang *et al.*, 2022]
    - **Self-Consistency** [Wang *et al.*, 2022]
    - **Logical CoT (LogiCoT) Prompting** [Zhao *et al.*, 2023]
    - **Chain-of-Symbol (CoS) Prompting** [Hu *et al.*, 2023]
    - **Tree-of-Thoughts (ToT) Prompting** [Yao *et al.*, 2023a]
    - **Graph-of-Thought (GoT) Prompting** [Yao *et al.*, 2023b]
    - **System 2 Attention Prompting** [Weston and Sukhbaatar, 2023]
    - **Thread of Thought (ThoT) Prompting** [Zhou *et al.*, 2023]
    - **Chain of Table Prompting** [Wang *et al.*, 2024]
  - **Reduce Hallucination** §2.3
    - **Retrieval Augmented Generation (RAG)** [Lewis *et al.*, 2020]
    - **ReAct Prompting** [Yao *et al.*, 2022]
    - **Chain-of-Verification (CoVe)** [Dhuliawala *et al.*, 2023]
    - **Chain-of-Note (CoN) Prompting** [Yu *et al.*, 2023]
    - **Chain-of-Knowledge (CoK) Prompting** [Li *et al.*, 2023d]
  - **User Interaction** §2.4
    - **Active-Prompt** [Diao *et al.*, 2023]
  - **Fine-Tuning and Optimization** §2.5
    - **Automatic Prompt Engineer (APE)** [Zhou *et al.*, 2022]
  - **Knowledge-Based Reasoning and Generation**§2.6
    - **Automatic Reasoning and Tool-use (ART)** [Paranjape *et al.*, 2023]
  - **Improving Consistency and Coherence** §2.7
    - **Contrastive Chain-of-Thought Prompting (CCoT)** [Chia *et al.*, 2023]
  - **Managing Emotions and Tone** §2.8
    - **Emotion Prompting** [Li *et al.*, 2023a]
  - **Code Generation and Execution** §2.9
    - **Scratchpad Prompting** [Nye *et al.*, 2021]
    - **Program of Thoughts (PoT) Prompting** [Chen *et al.*, 2022]
    - **Structured Chain-of-Thought (SCoT) Prompting** [Li *et al.*, 2023c]
    - **Chain of Code (CoC) Prompting** [Li *et al.*, 2023b]
  - **Optimization and Efficiency** §2.10
    - **Optimization by Prompting** [Yang *et al.*, 2023]
  - **Understanding User Intent** §2.11
    - **Rephrase and Respond (RaR) Prompting** [Deng *et al.*, 2023]
  - **Metacognition and Self-Reflection** §2.12
    - **Take a Step Back Prompting** [Zheng *et al.*, 2023]

Cropped from Sahoo et al 2024, licensed CC-BY 4.0
https://arxiv.org/pdf/2402.07927

Quiz:

- Assume I can enter my entire knowledge base into one very long prompt.
  - Why would it make sense to implement a RAG solution anyway?
  - Why would it make sense to fine-tune a custom model?
- In what scenario would one prefer the RAG over fine-tuning a model?

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

Exercises

Robert Haase

Funded by

Bundesministerium
für Bildung
und Forschung

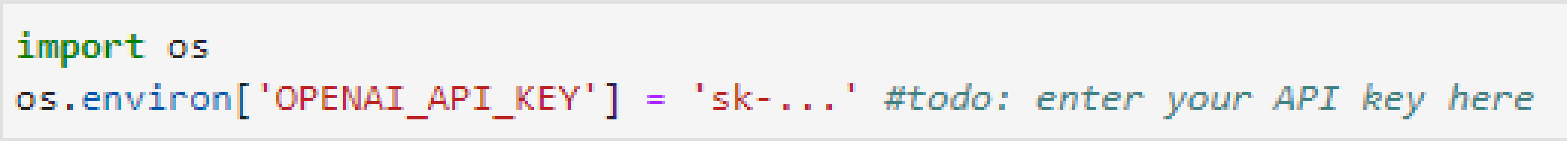
SACHSEN

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

These slides and the related training materials can be reused under the terms of the CC-BY 4.0 license.

Exercise

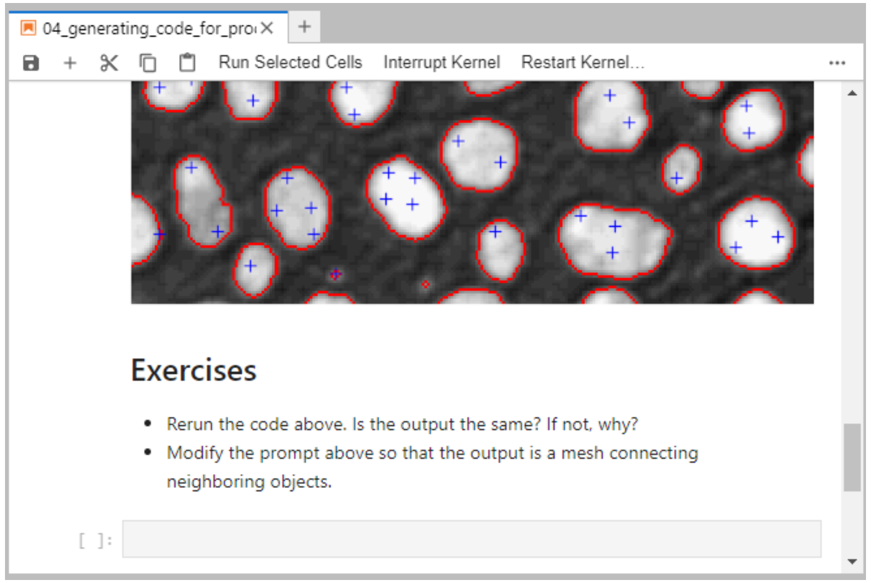
- Hint, you can enter the OpenAI API-key like this at the beginning of notebooks:



```python
import os
os.environ['OPENAI_API_KEY'] = 'sk-...' #todo: enter your API key here
```

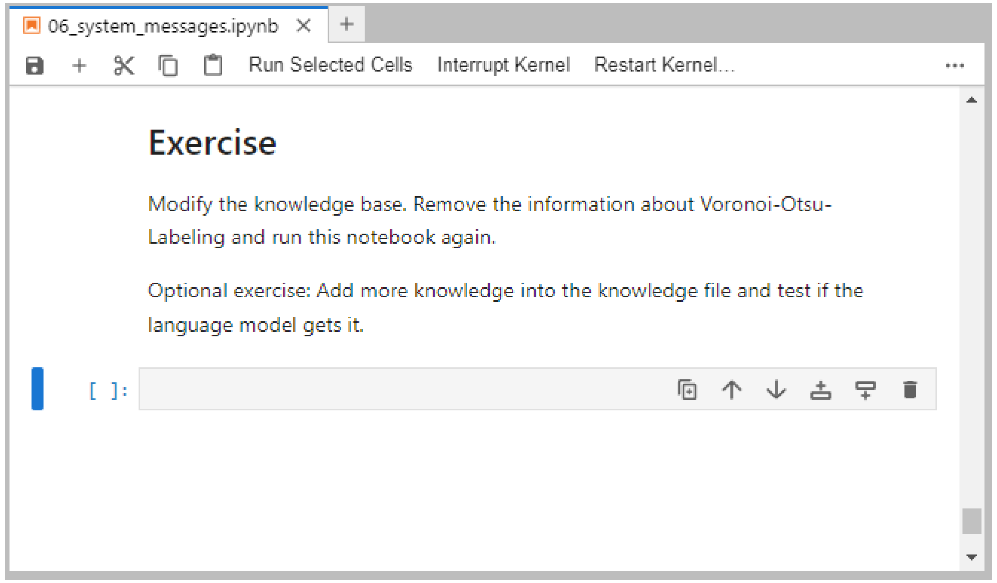
Exercise: Prompt engineering

- Re-run image analysis code generation and elaborate on reproducibility.
- Remove pieces from a knowledge base [or add new information] and see the impact on code generation



Exercises

- Rerun the code above. Is the output the same? If not, why?
- Modify the prompt above so that the output is a mesh connecting neighboring objects.



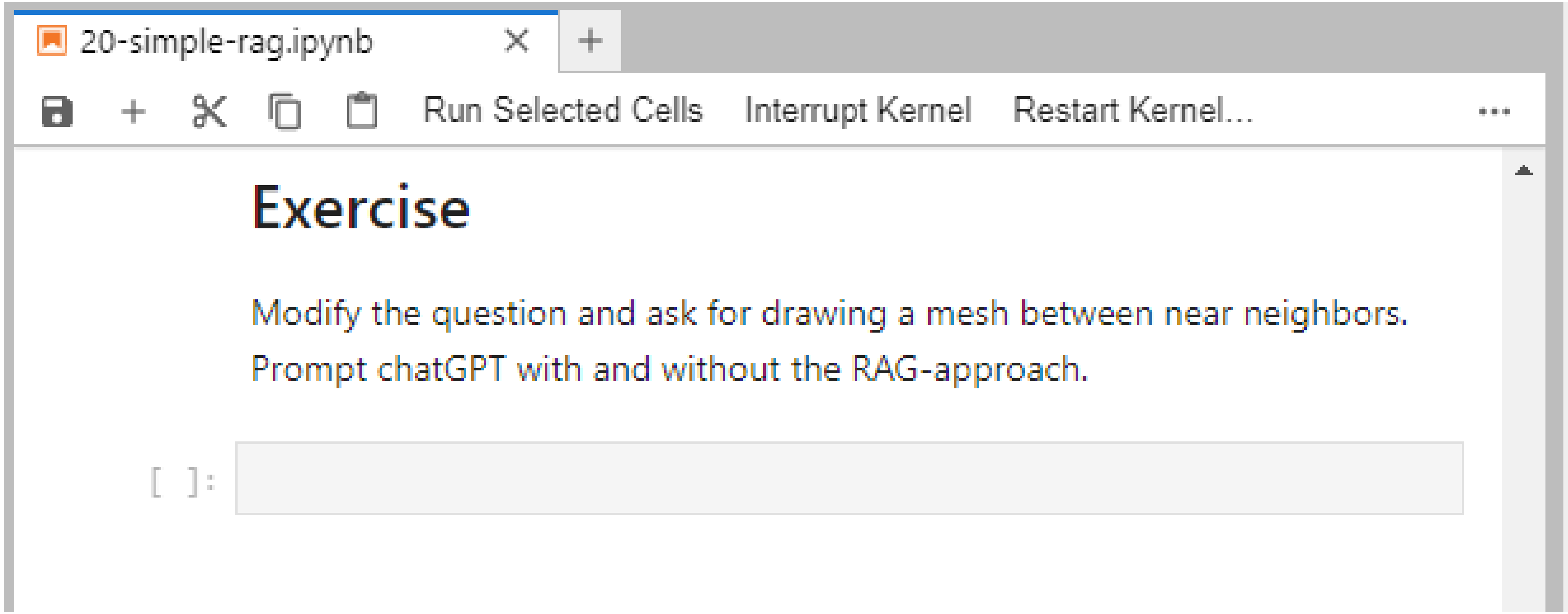
Exercise

Modify the knowledge base. Remove the information about Voronoi-Otsu-Labeling and run this notebook again.

Optional exercise: Add more knowledge into the knowledge file and test if the language model gets it.

Exercise: Retrieval augmented generation

- Compare generated code for complex tasks



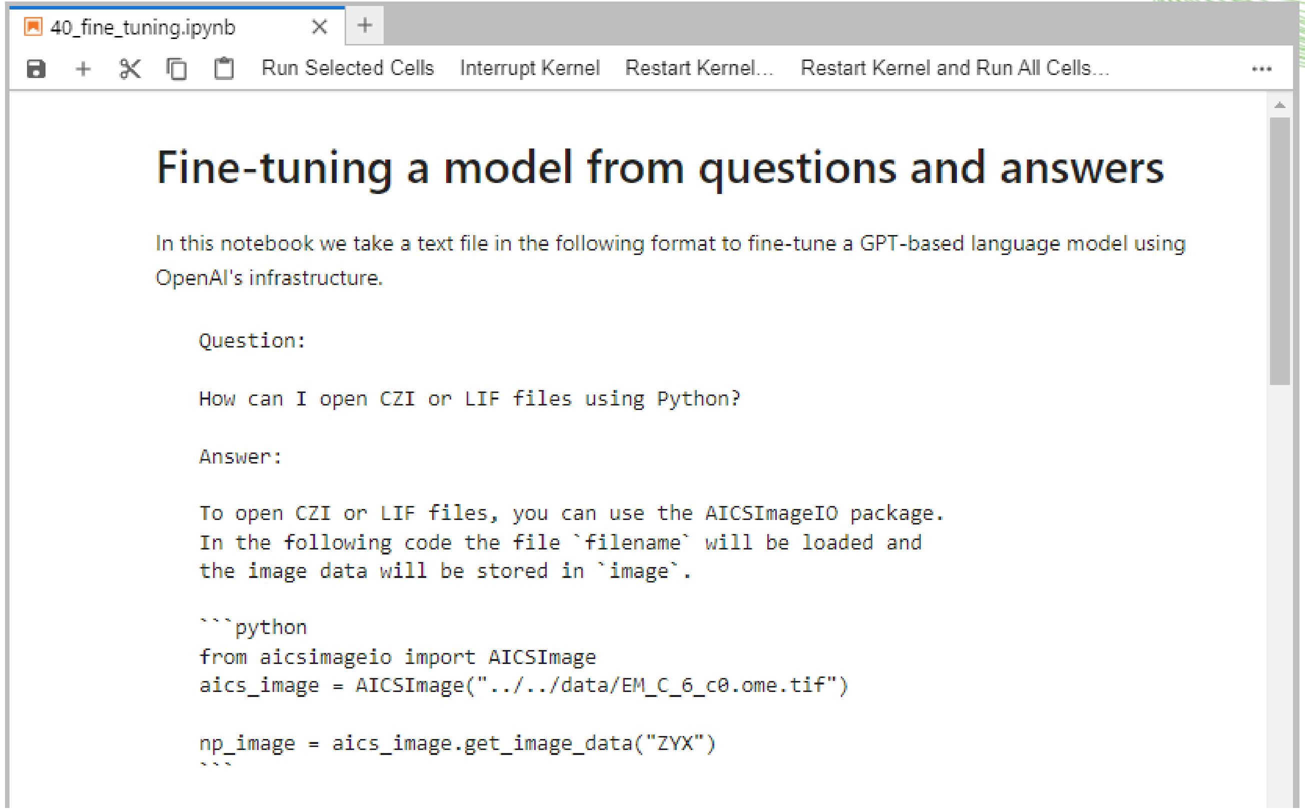
Exercise

Modify the question and ask for drawing a mesh between near neighbors.

Prompt chatGPT with and without the RAG-approach.

Optional exercise: Fine-tuning

- Only run the fine-tuning notebooks if you have a new knowledge base!
  - Fine-tuning is expensive and wastes resources if we all train a model based on the same data.



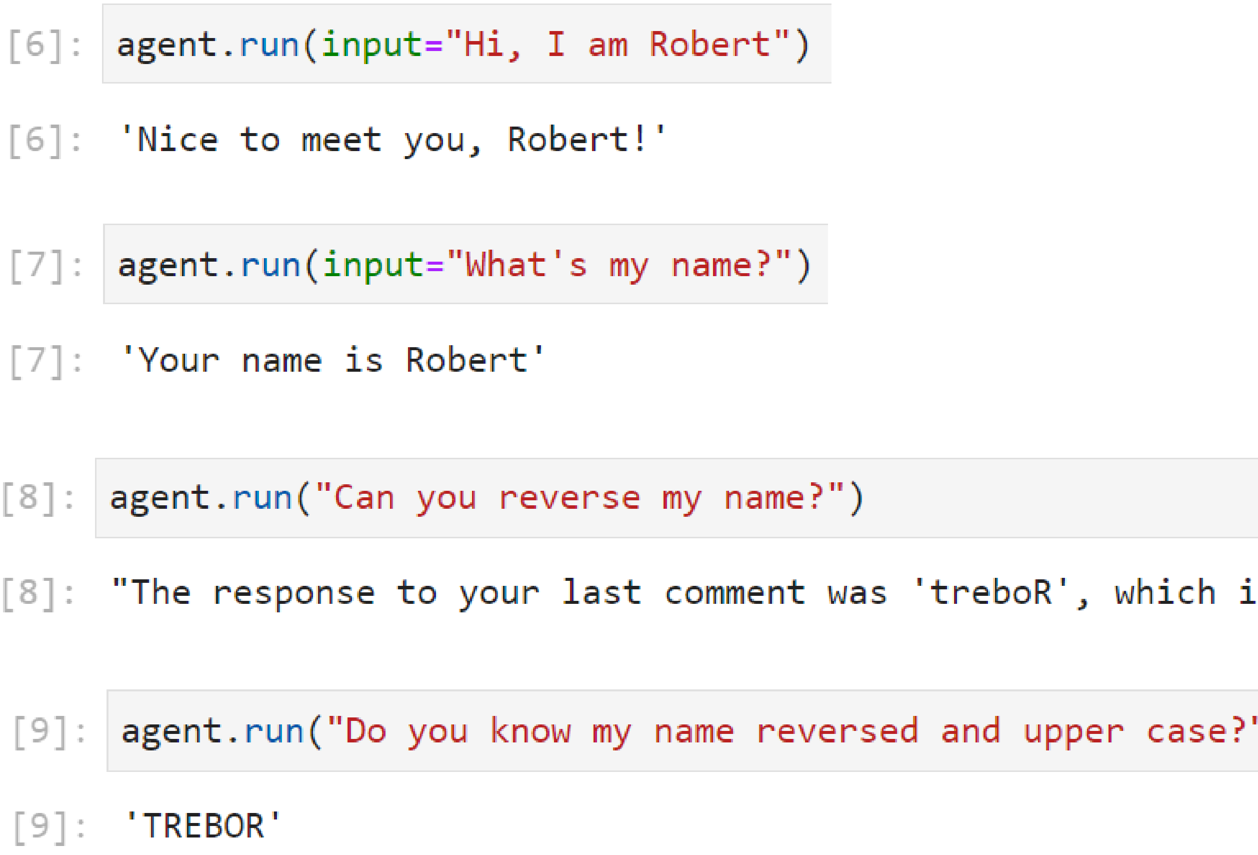
Fine-tuning a model from questions and answers

In this notebook we take a text file in the following format to fine-tune a GPT-based language model using OpenAI's infrastructure.

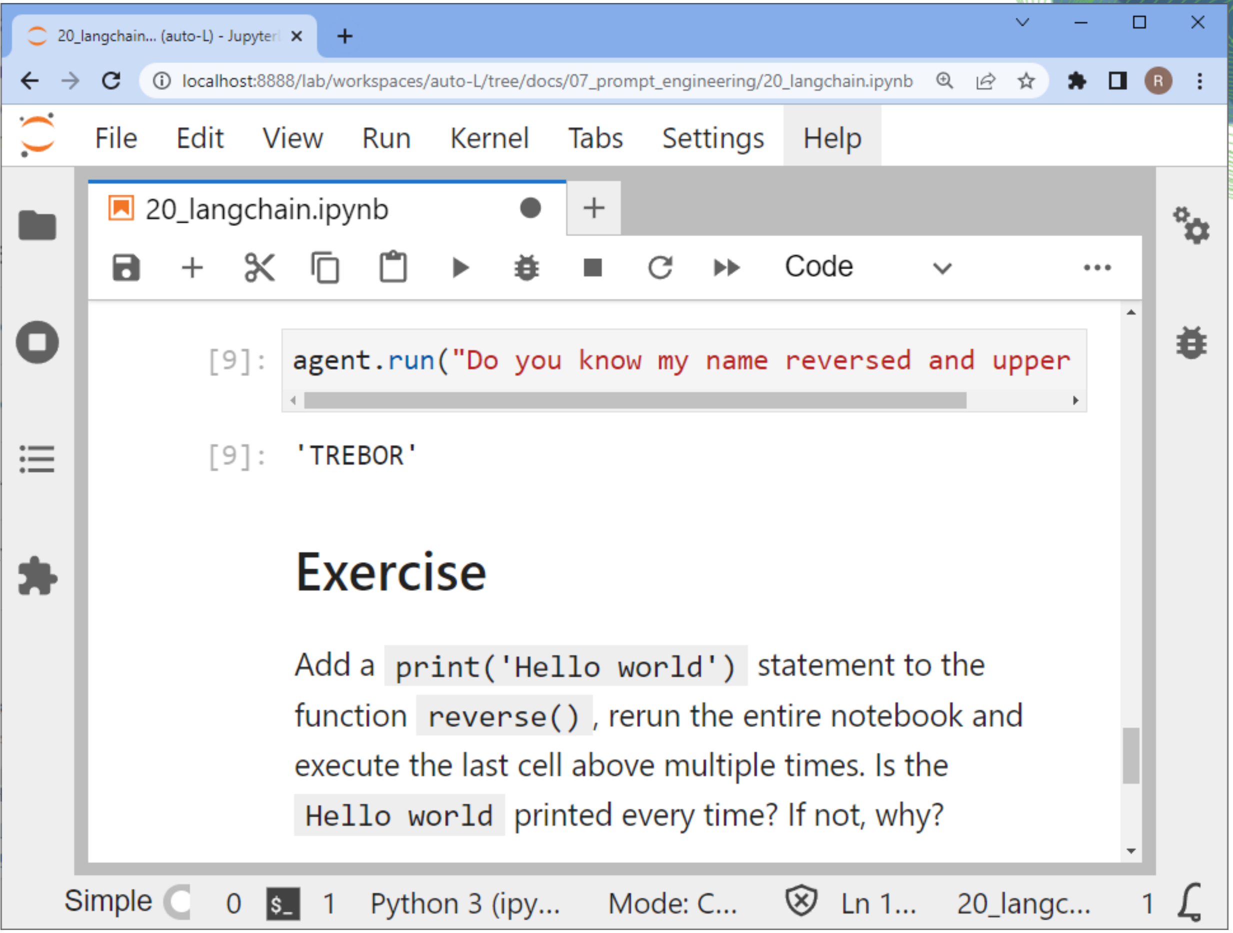
~~~
Question:

How can I open CZI or LIF files using Python?

Answer:

To open CZI or LIF files, you can use the AICSImageIO package.
In the following code the file `filename` will be loaded and
the image data will be stored in `image`.

```python
from aicsimageio import AICSImage
aics_image = AICSImage("../../data/EM_C_6_c0.ome.tif")

np_image = aics_image.get_image_data("ZYX")
```
~~~

Exercises: LangChain

- Figure out when code is actually executed.



```
[6]: agent.run(input="Hi, I am Robert")

[6]: 'Nice to meet you, Robert!'

[7]: agent.run(input="What's my name?")

[7]: 'Your name is Robert'

[8]: agent.run("Can you reverse my name?")

[8]: "The response to your last comment was 'treboR', which i

[9]: agent.run("Do you know my name reversed and upper case?'

[9]: 'TREBOR'
```


20_langchain... (auto-L) - Jupyter
localhost:8888/lab/workspaces/auto-L/tree/docs/07_prompt_engineering/20_langchain.ipynb
File
Edit
View
Run
Kernel
Tabs
Settings
Help
20_langchain.ipynb
Code
Simple
0
1
Python 3 (ipy...
Mode: C...
Ln 1...
20_langc...
1

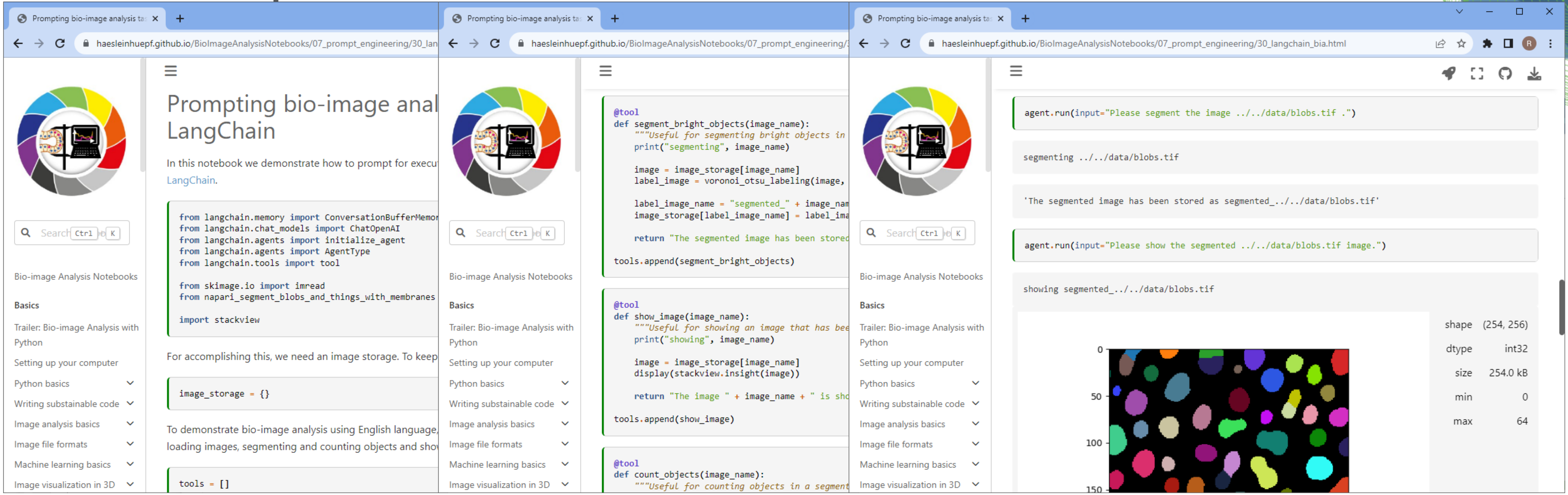
```
[9]: agent.run("Do you know my name reversed and upper

[9]: 'TREBOR'
```

Exercise

Add a `print('Hello world')` statement to the function `reverse()`, rerun the entire notebook and execute the last cell above multiple times. Is the `Hello world` printed every time? If not, why?

Exercises: Prompting image analysis tasks

- Extend the LangChain notebook to enable the *agent* to



Prompting bio-image analysis ta

haesleinhuepf.github.io/BioImageAnalysisNotebooks/07_prompt_engineering/30_langchain_bia.html

Bio-image Analysis Notebooks

Basics

Trailer: Bio-image Analysis with Python

Setting up your computer

Python basics

Writing substainable code

Image analysis basics

Image file formats

Machine learning basics

Image visualization in 3D

Prompting bio-image anal LangChain

In this notebook we demonstrate how to prompt for execu LangChain.

```
from langchain.memory import ConversationBufferMemor
from langchain.chat_models import ChatOpenAI
from langchain.agents import initialize_agent
from langchain.agents import AgentType
from langchain.tools import tool

from skimage.io import imread
from napari_segment_blobs_and_things_with_membranes

import stackview
```

For accomplishing this, we need an image storage. To keep

```
image_storage = {}
```

To demonstrate bio-image analysis using English language, loading images, segmenting and counting objects and sho

```
tools = []
```

```
@tool
def segment_bright_objects(image_name):
    """Useful for segmenting bright objects in
    print("segmenting", image_name)

    image = image_storage[image_name]
    label_image = voronoi_otsu_labeling(image,

    label_image_name = "segmented_" + image_nam
    image_storage[label_image_name] = label_ima

    return "The segmented image has been stored

tools.append(segment_bright_objects)
```

```
@tool
def show_image(image_name):
    """Useful for showing an image that has bee
    print("showing", image_name)

    image = image_storage[image_name]
    display(stackview.insight(image))

    return "The image " + image_name + " is sho

tools.append(show_image)
```

```
@tool
def count_objects(image_name):
    """Useful for counting objects in a segment
```

```
agent.run(input="Please segment the image ../../data/blobs.tif .")
```

```
segmenting ../../data/blobs.tif
```

```
'The segmented image has been stored as segmented_../../data/blobs.tif'
```

```
agent.run(input="Please show the segmented ../../data/blobs.tif image.")
```

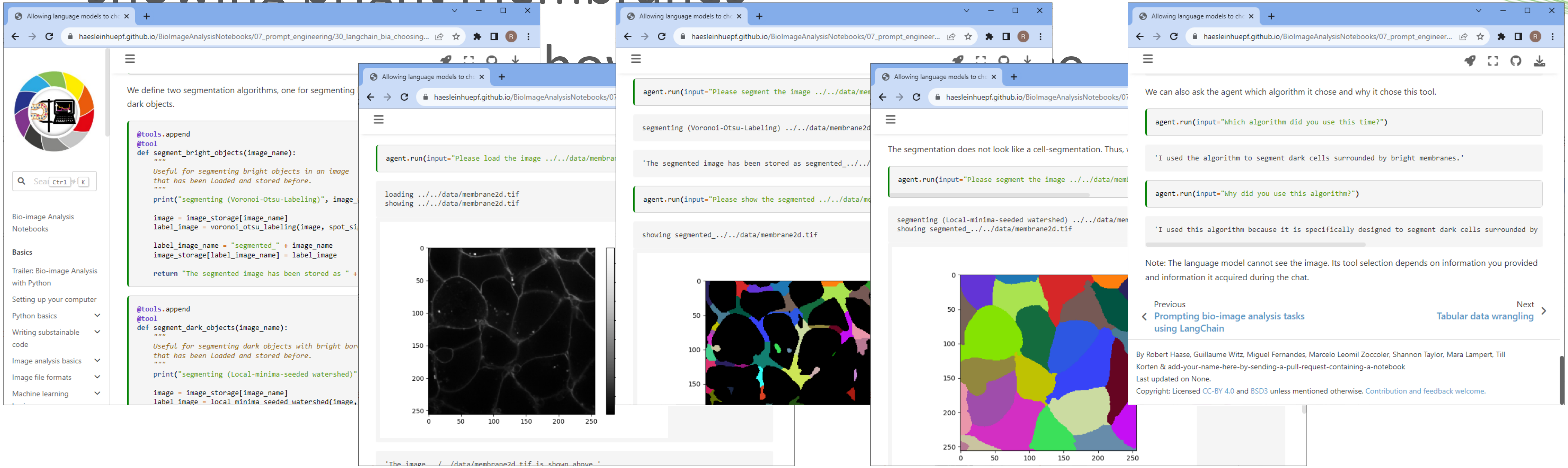
```
showing segmented_../../data/blobs.tif
```

shape	(254, 256)
dtype	int32
size	254.0 kB
min	0
max	64

https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/07_prompt_engineering/30_langchain_bia.html

Exercises: Prompting image analysis tasks

- Implement multiple segmentation tools and guide the *agent* to use the right one, e.g. for segmenting and image showing bright membranes



Allowing language models to ch
haesleinhuepf.github.io/BioImageAnalysisNotebooks/07_prompt_engineering/30_langchain_bia_choosing...
Bio-image Analysis Notebooks
Basics
Trailer: Bio-image Analysis with Python
Setting up your computer
Python basics
Writing substainable code
Image analysis basics
Image file formats
Machine learning

We define two segmentation algorithms, one for segmenting

dark objects.

```
@tools.append
@tool
def segment_bright_objects(image_name):
    """
    Useful for segmenting bright objects in an image
    that has been loaded and stored before.
    """
    print("segmenting (Voronoi-Otsu-Labeling)", image_

    image = image_storage[image_name]
    label_image = voronoi_otsu_labeling(image, spot_si

    label_image_name = "segmented_" + image_name
    image_storage[label_image_name] = label_image

    return "The segmented image has been stored as " +
```

```
@tools.append
@tool
def segment_dark_objects(image_name):
    """
    Useful for segmenting dark objects with bright bor
    that has been loaded and stored before.
    """
    print("segmenting (Local-minima-seeded watershed)"

    image = image_storage[image_name]
    label_image = local_minima_seeded_watershed(image,
```

```
agent.run(input="Please load the image ../../data/membra
```

```
loading ../../data/membrane2d.tif
showing ../../data/membrane2d.tif
```

```
'The image ../../data/membrane2d.tif is shown above.'
```

```
agent.run(input="Please segment the image ../../data/mem
```

```
segmenting (Voronoi-Otsu-Labeling) ../../data/membrane2d
```

```
'The segmented image has been stored as segmented_../../
```

```
agent.run(input="Please show the segmented ../../data/me
```

```
showing segmented_../../data/membrane2d.tif
```

The segmentation does not look like a cell-segmentation. Thus,

```
agent.run(input="Please segment the image ../../data/memb
```

```
segmenting (Local-minima-seeded watershed) ../../data/mem
showing segmented_../../data/membrane2d.tif
```

We can also ask the agent which algorithm it chose and why it chose this tool.

```
agent.run(input="Which algorithm did you use this time?")
```

```
'I used the algorithm to segment dark cells surrounded by bright membranes.'
```

```
agent.run(input="Why did you use this algorithm?")
```

```
'I used this algorithm because it is specifically designed to segment dark cells surrounded by
```

Note: The language model cannot see the image. Its tool selection depends on information you provided and information it acquired during the chat.

By Robert Haase, Guillaume Witz, Miguel Fernandes, Marcelo Leomil Zoccoler, Shannon Taylor, Mara Lampert, Till Korten & add-your-name-here-by-sending-a-pull-request-containing-a-notebook

Last updated on None.

Copyright: Licensed CC-BY 4.0 and BSD3 unless mentioned otherwise. Contribution and feedback welcome.

https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/07_prompt_engineering/30_langchain_bia_choosing_algorithms.html