Microsoft

Fine-Tuning des modèles fondationnels avec Azure AI Foundry

Jeudi 24 avril 2025

global AI community

Virtuelle Francophone

Serge RETKOWSKY

France AI Global Black Belt
@Microsoft

Farid EL ATTAOUI

France AI Global Black Belt
@Microsoft

Fine-Tuning des modèles fondationnels avec Azure AI Foundry

Les modèles de GenAI sont extrêmement puissants, mais peuvent manquer de précision pour certaines tâches spécifiques. Le fine-tuning permet d'adapter un modèle fondationnel LLM ou SLM à vos besoins spécifiques en exploitant vos propres données.

Nicolas ROBERT

AI Architect @Expertime
Microsoft MVP IA

LinkedIn Live jeudi **24/04/2025 à** partir de 13h

Agenda

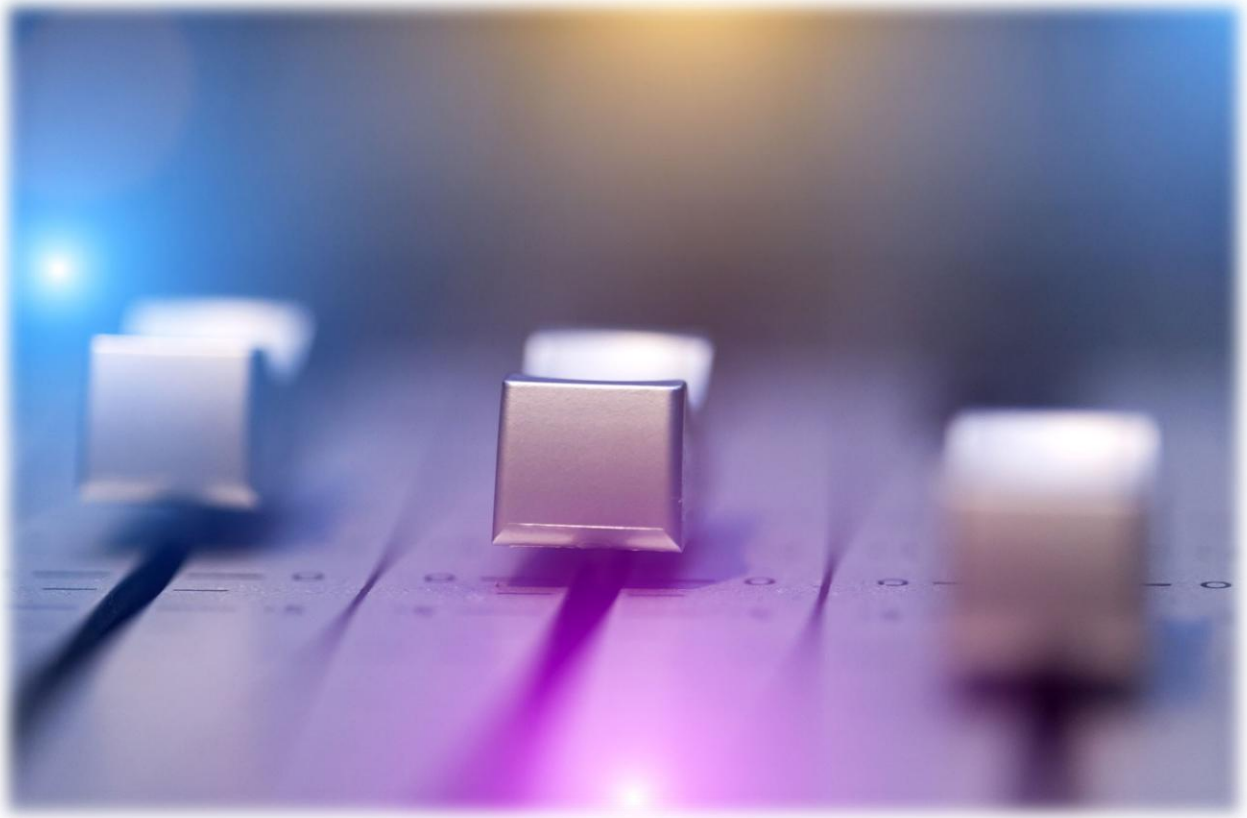
01
Why / When should we fine-tune a foundation model?

02
What is a foundation model?

03
Fine-Tuning techniques

04
What is available on Azure?

05
Demos



Enhance your foundation model with fine-tuning on Azure

01

Why / When should we fine-tune a foundation model?

02

What is a foundation model?

03

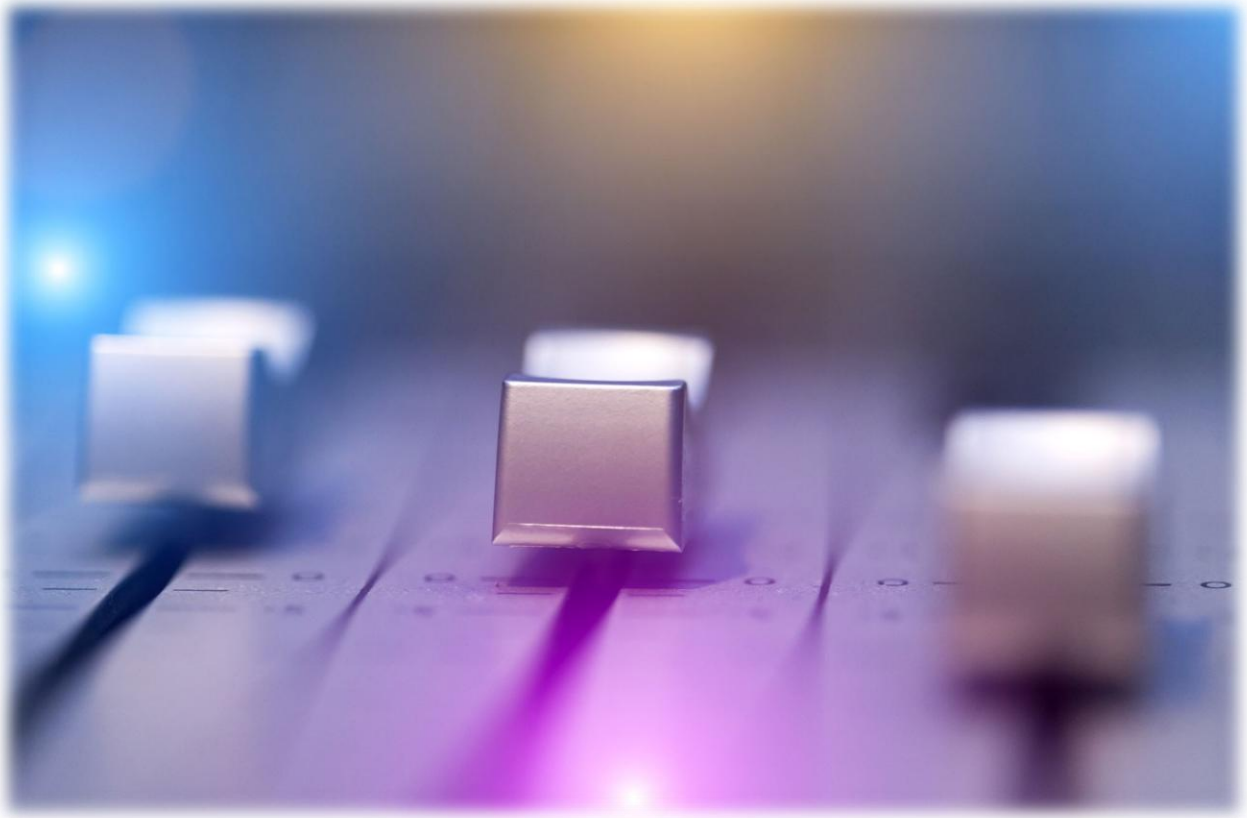
Fine-Tuning techniques

04

What is available on Azure?

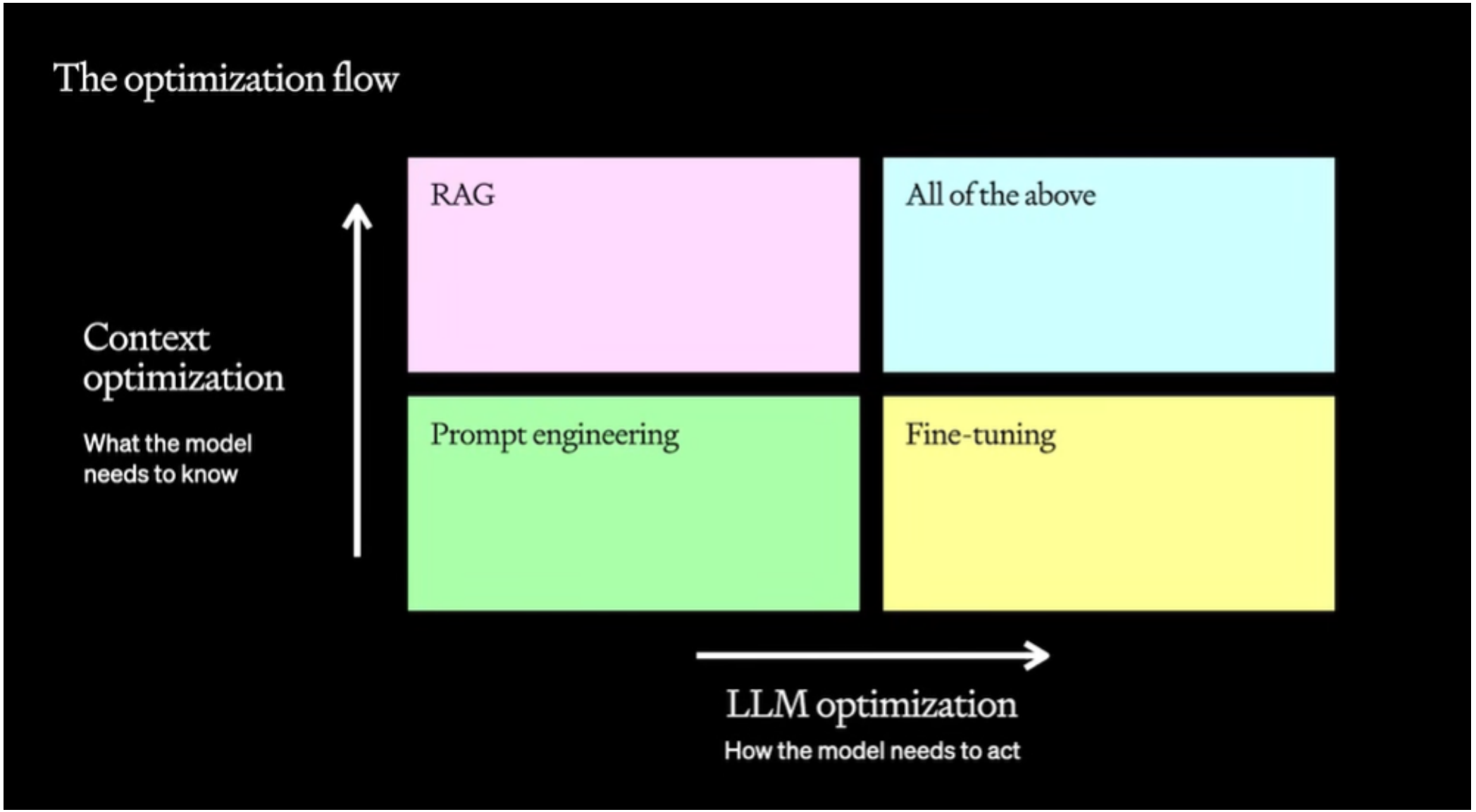
05

Demos



Gen AI journey

Plan an iterative path from basic to advanced GenAI leveraging your data



source

Why customize Gen AI models?

Scalability
Customization enables AI models to scale and adapt to specific enterprise needs.

Reducing hallucinations
Tailored models are less likely to produce inaccurate or irrelevant responses.

Increased reliability
Enhances the model's accuracy for domain-specific tasks.

Improved efficiency
Customization ensures faster and more precise results, saving time and resources.

Tailored solutions
Models are fine-tuned for specific use cases, providing more relevant and context-aware outcomes.

Enhance your foundation model with fine-tuning on Azure

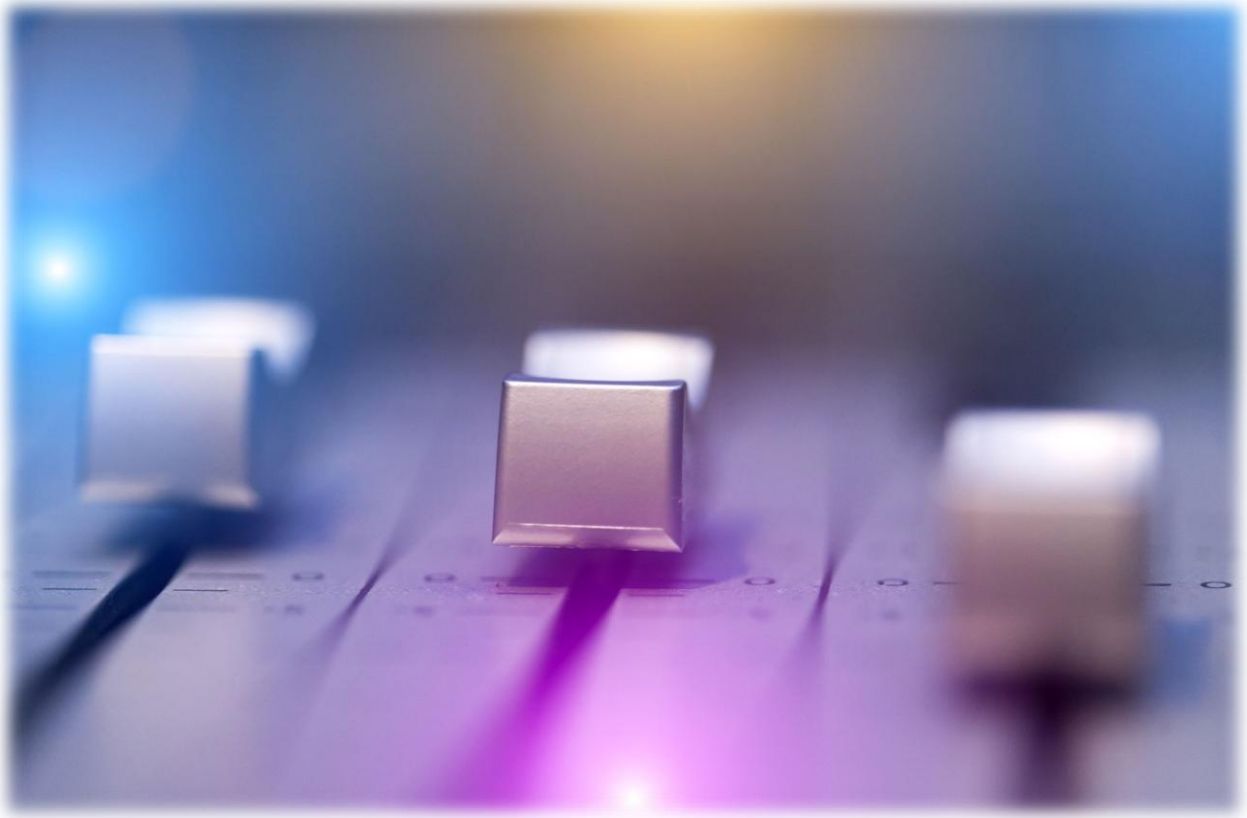
01
Why / When should we fine-tune a foundation model?

02
What is a foundation model?

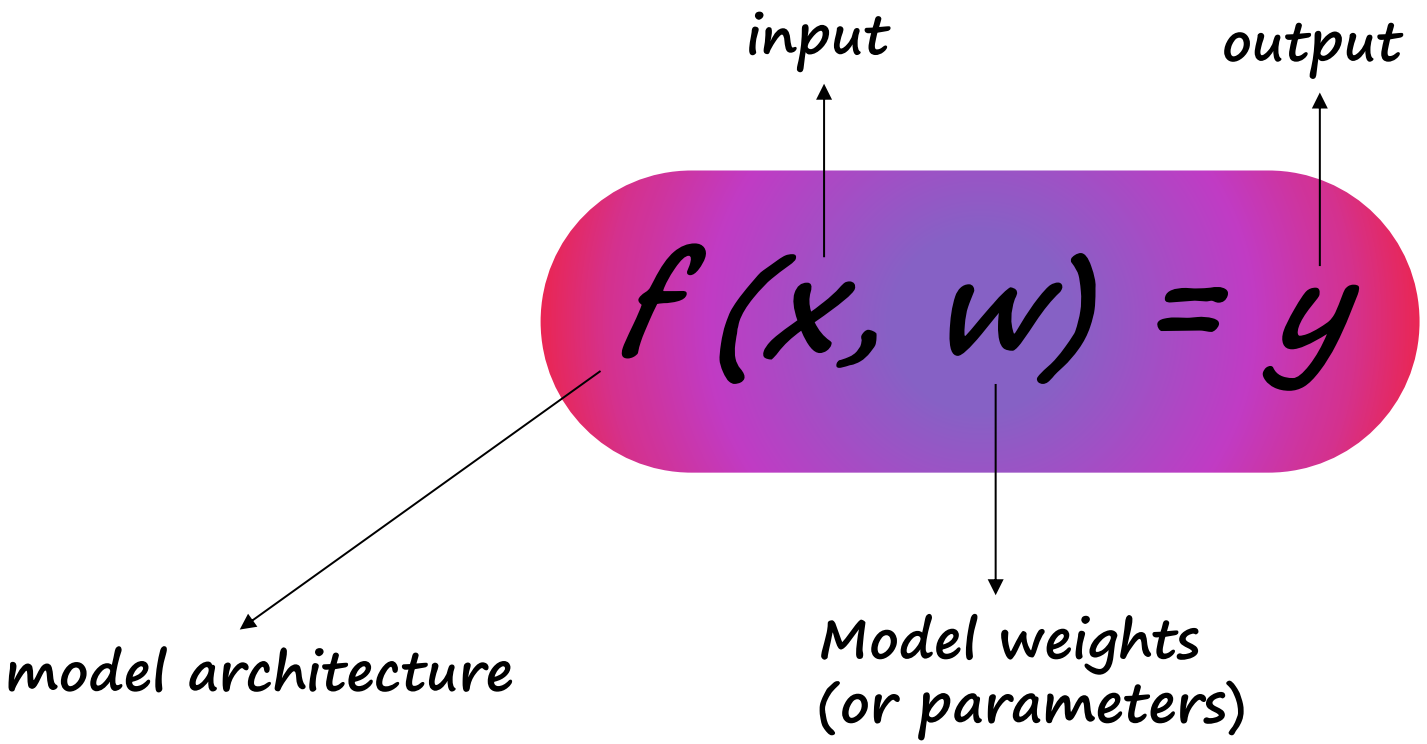
03
Fine-Tuning techniques

04
What is available on Azure?

05
Demos

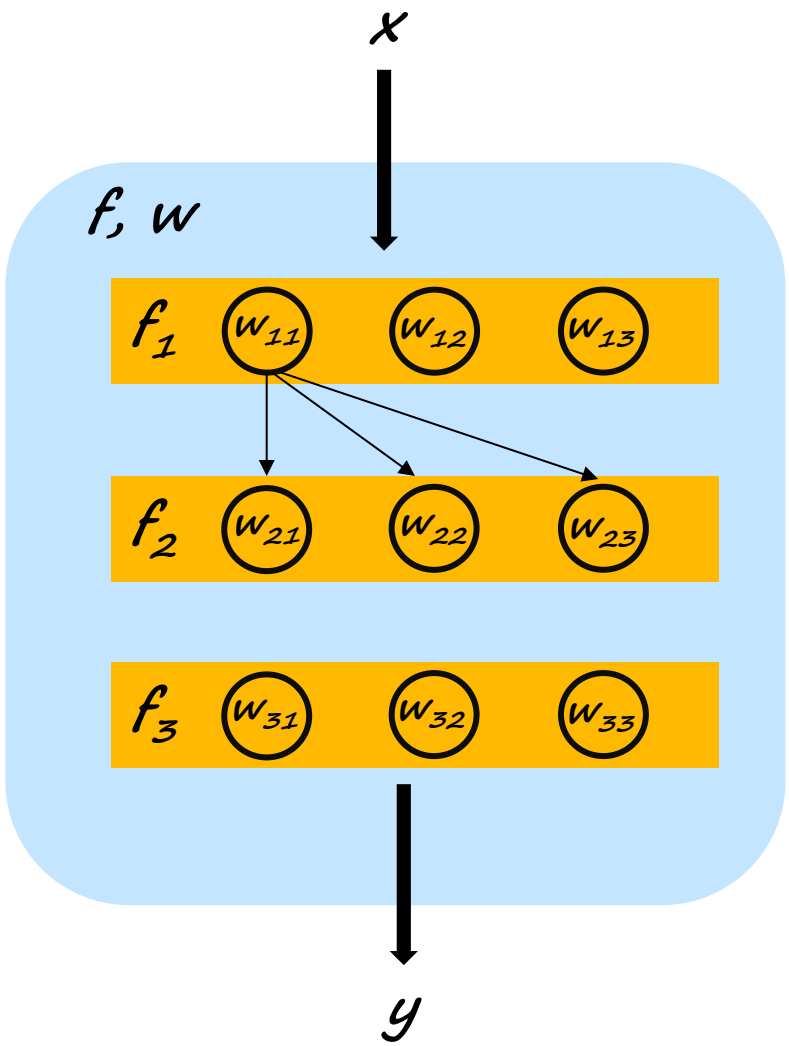


AI model as a function



AI model as a Network

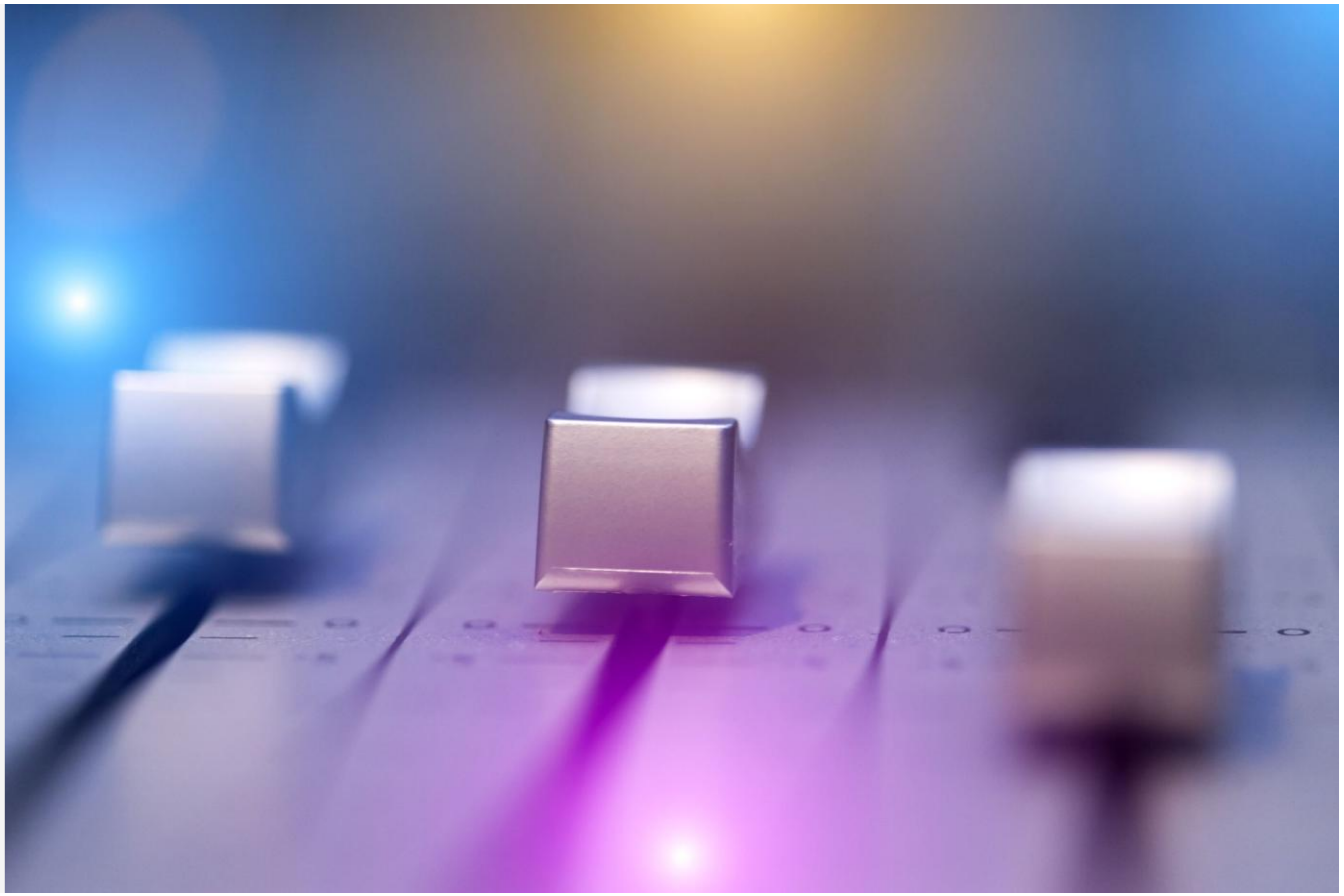
$f(x, w) = y$



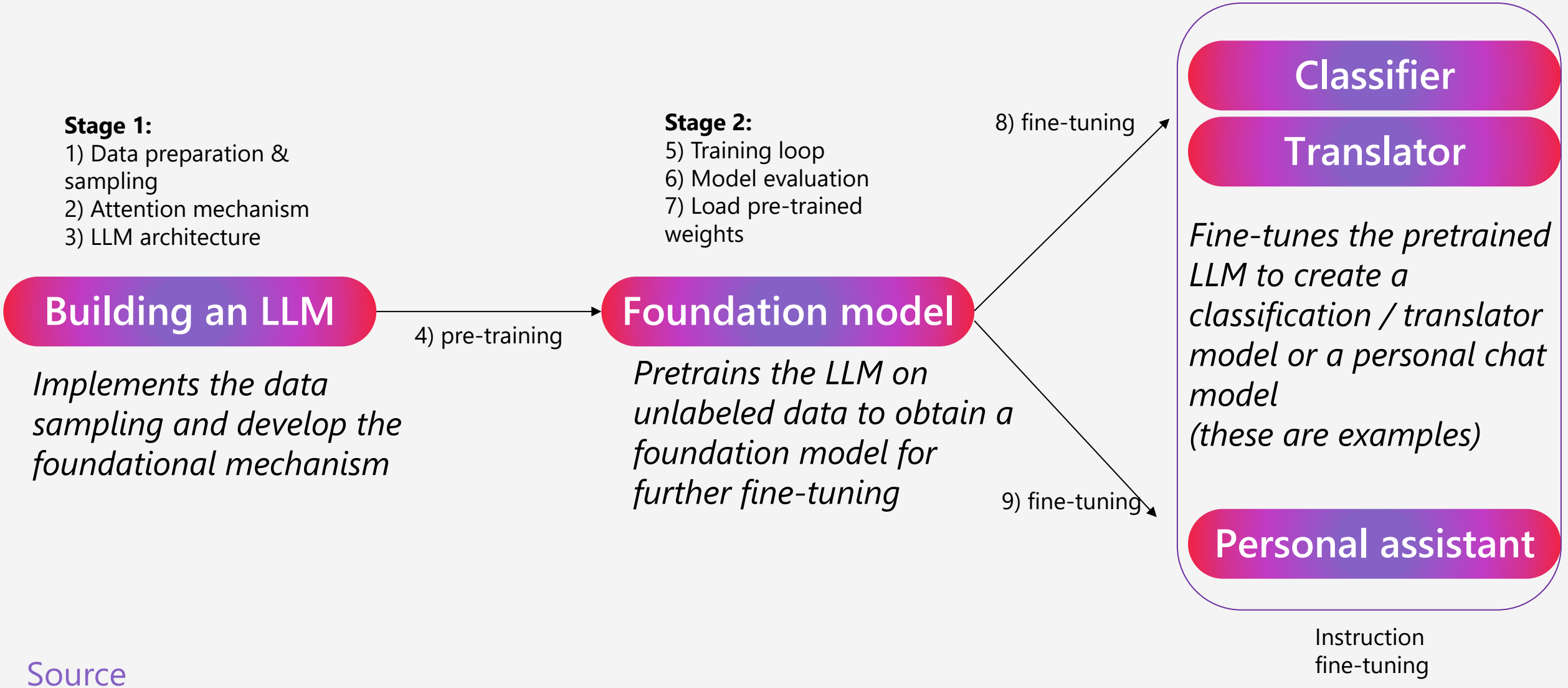
What is Fine-Tuning?

Fine-Tuning

Technique that involves taking a pre-trained model (already been trained on a large, general dataset) and further training it on a specific, often smaller dataset
to make it more suitable for a particular task or domain.

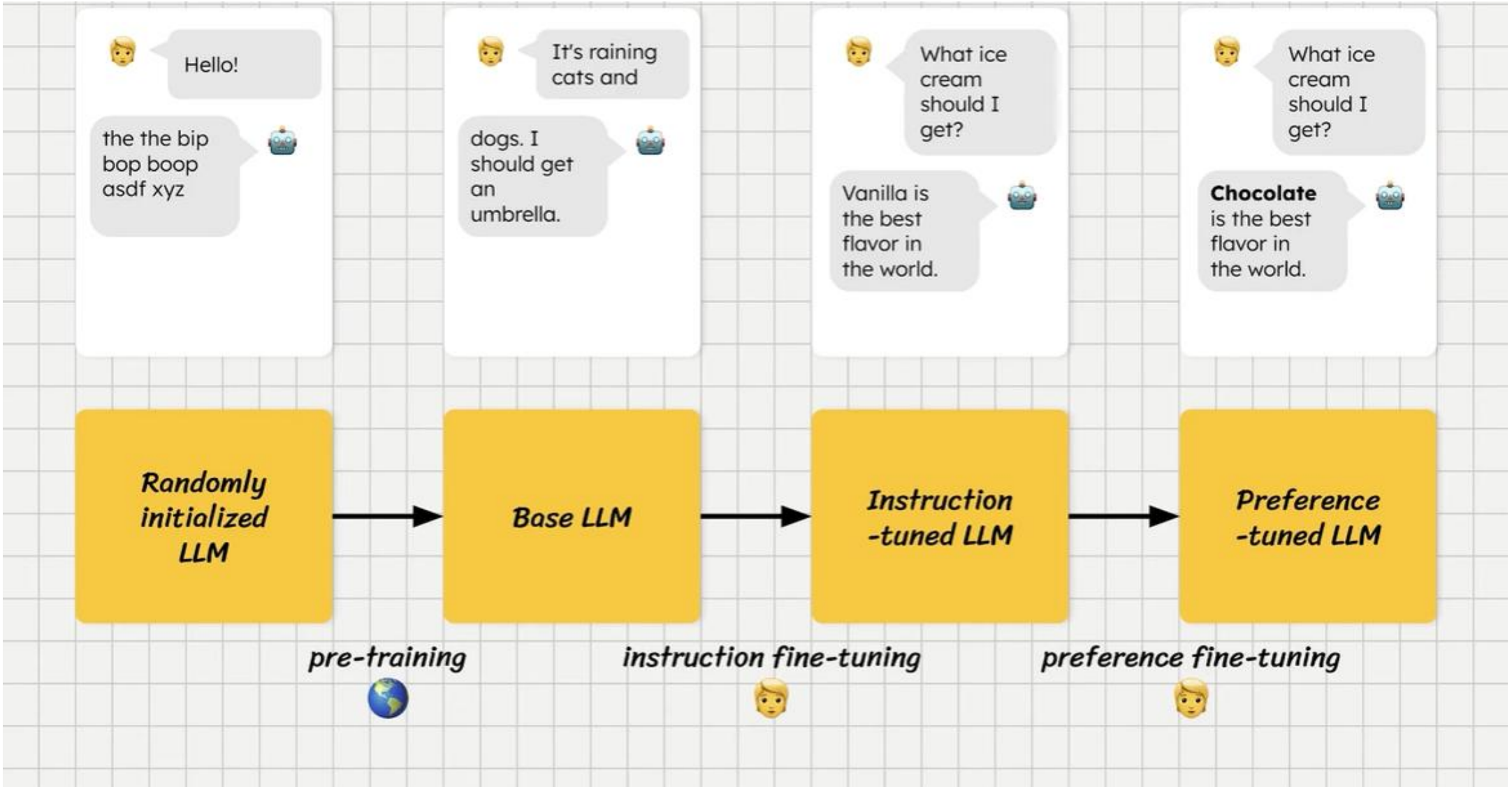


But an LLM is already fine-tuned ?



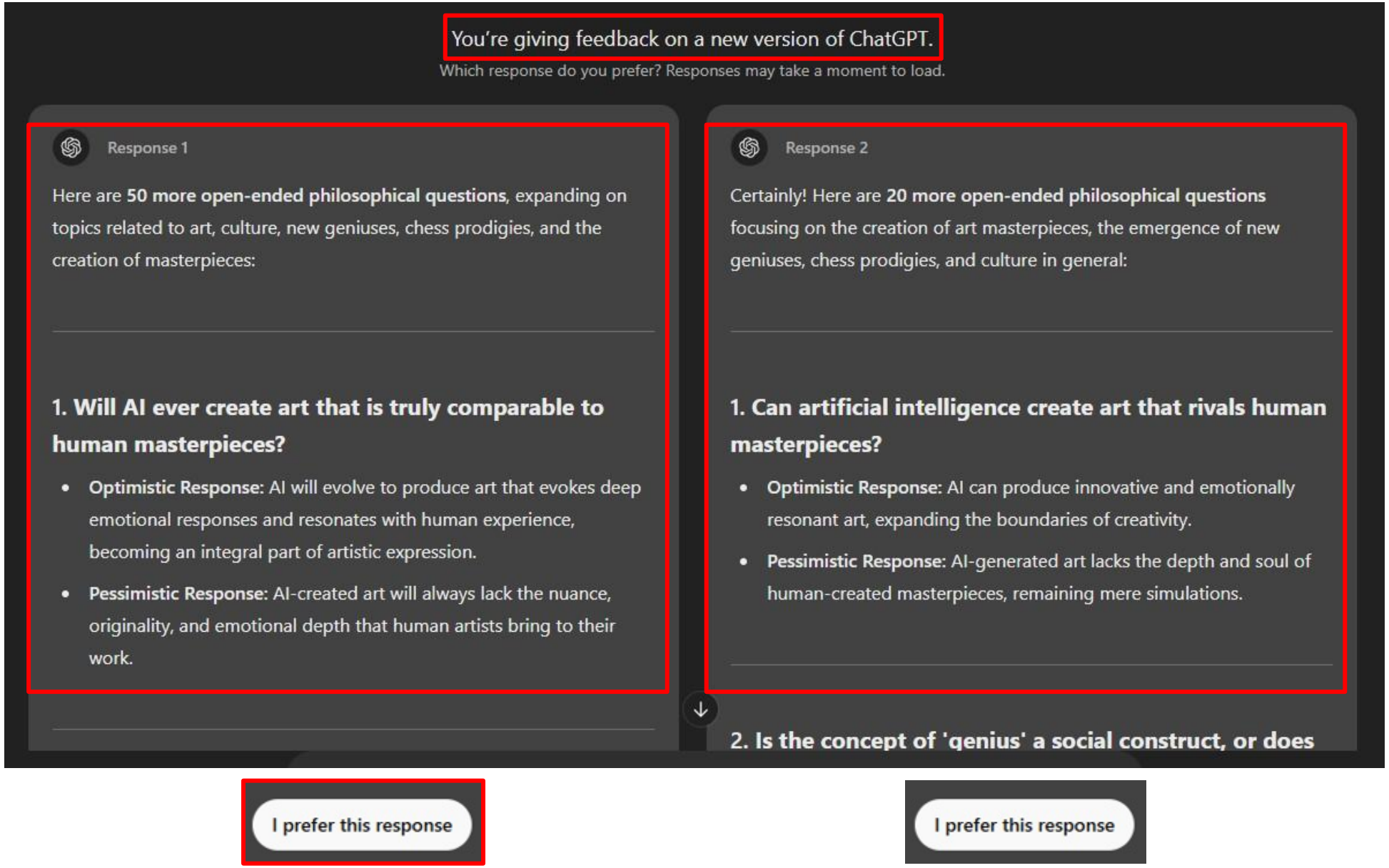
Source

LLM training the different steps

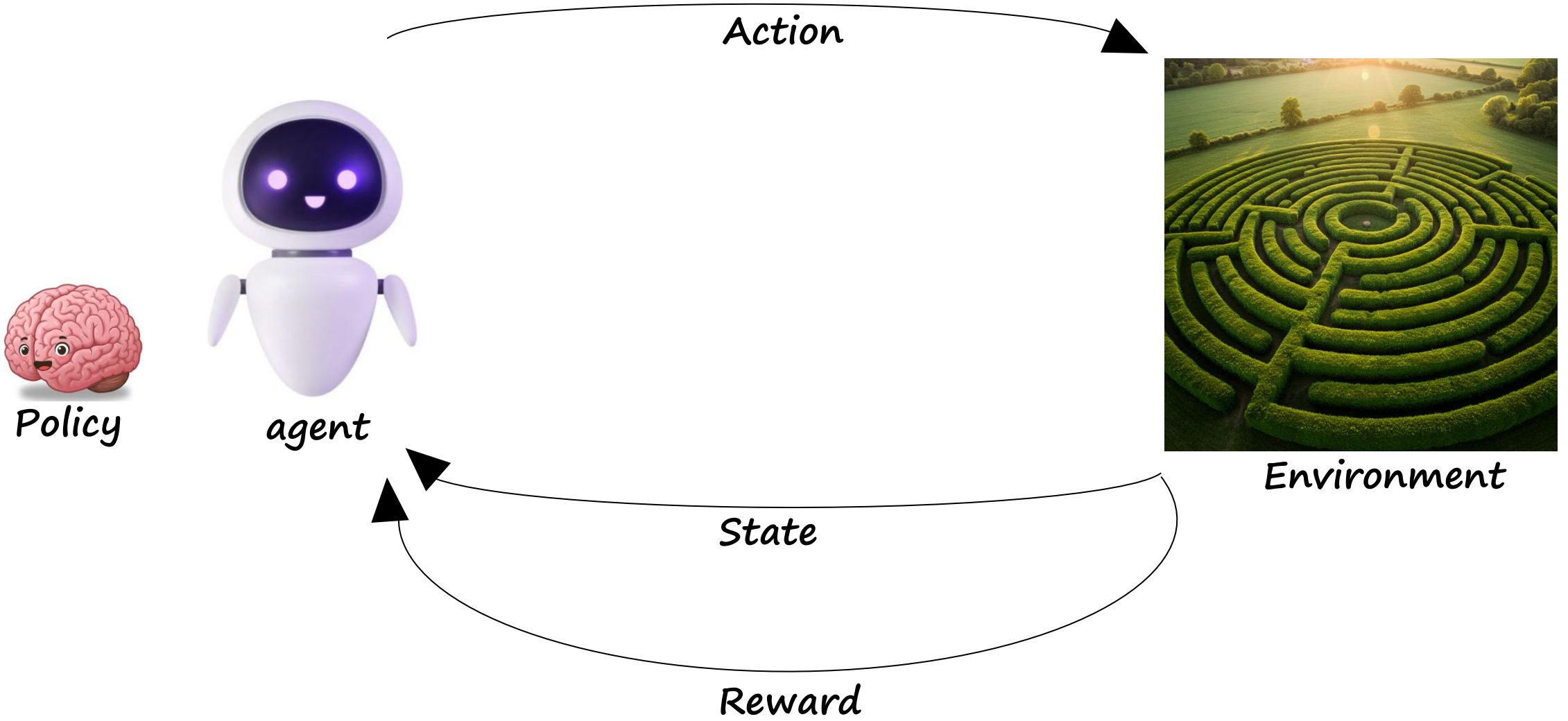


source

Preferences on ChatGPT

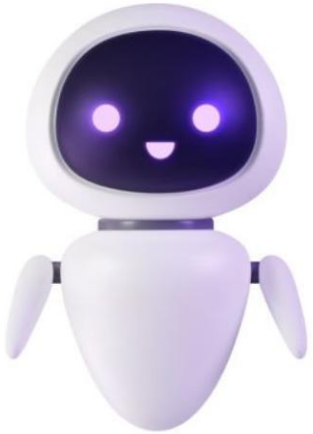


Reinforcement Learning in a Nutshell



Reinforcement Learning for LLMs

Action = next word prediction



Policy=LLM agent

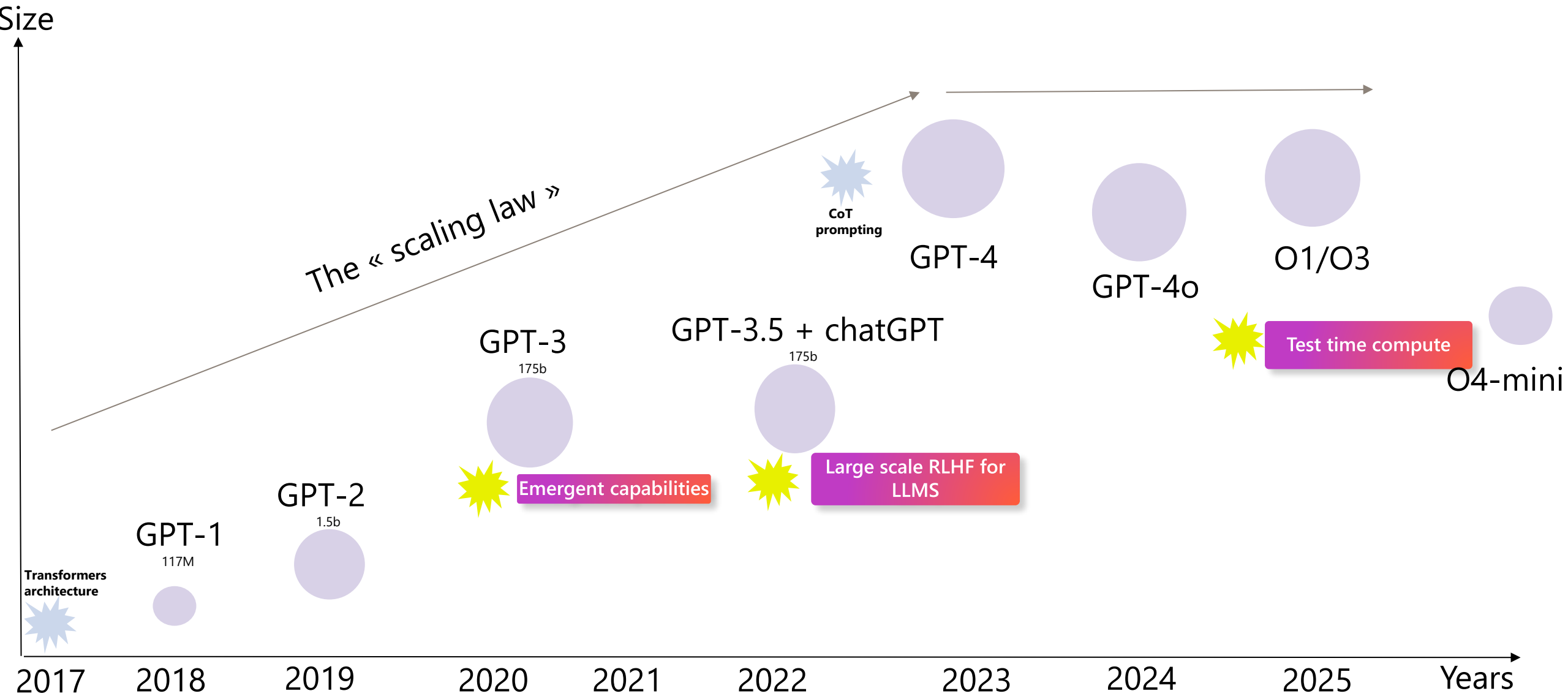
Environment

State = prompt + generated tokens

Reward += 1

Human/AI reward

Evolution of LLMs flagships models



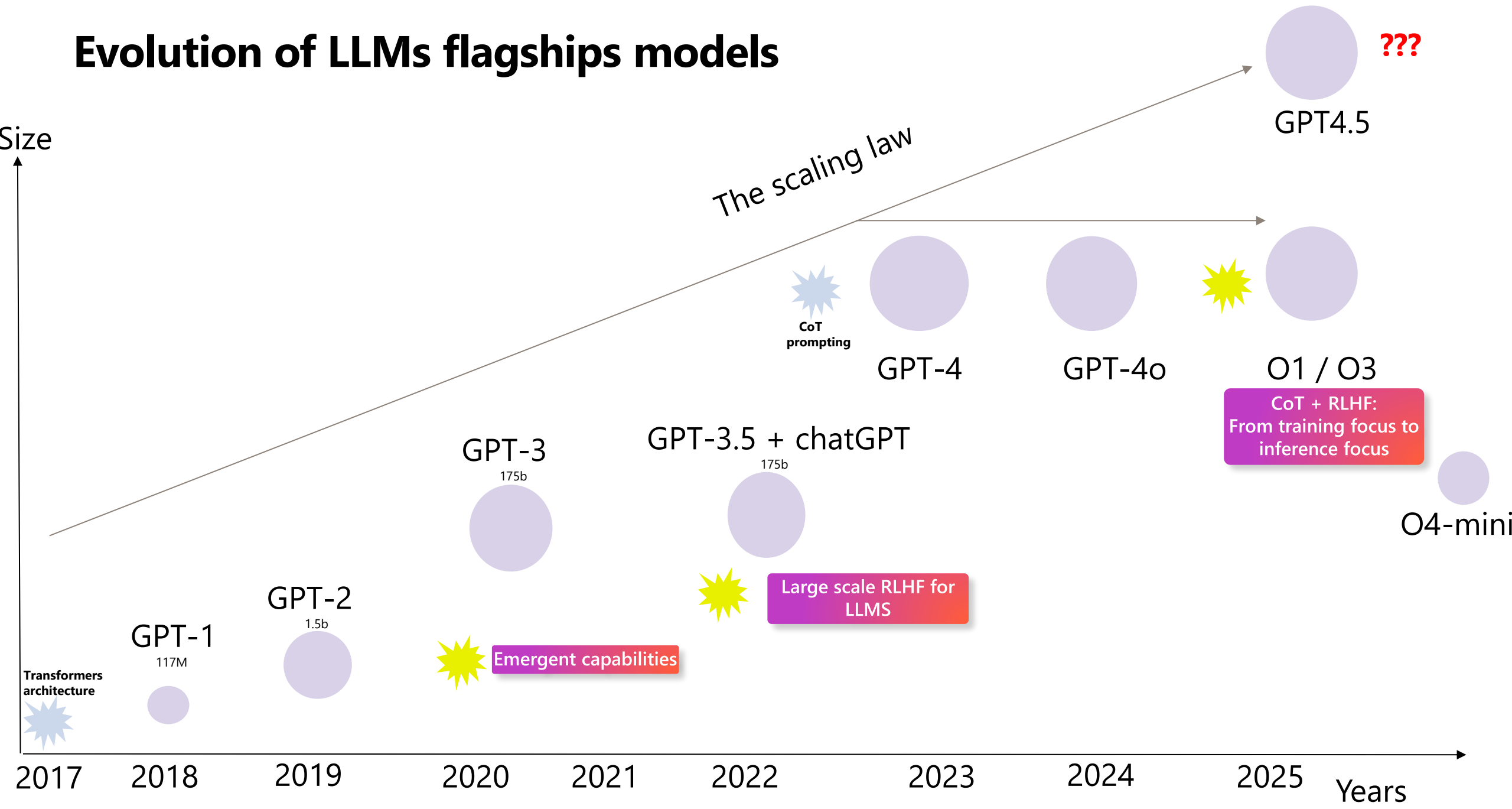
Is it the end of the scaling law?

"Pre-training as we know it will end. [...] Data is not growing: we have but one Internet. [...] You could even go as far as to say that data is the fossil fuel of AI."

Illya Sutskever,
Former Chief Scientist Officer at OpenAI

source

Evolution of LLMs flagships models



Enhance your foundation model with fine-tuning on Azure

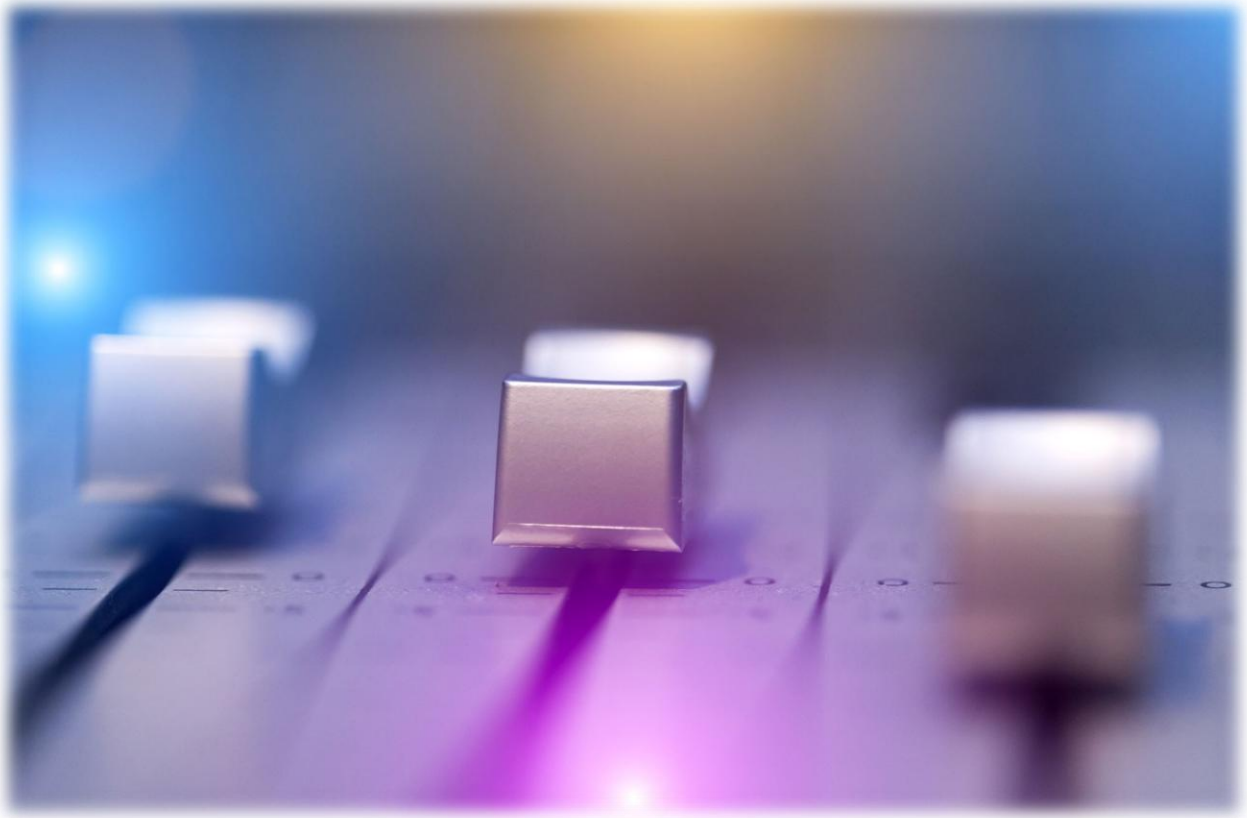
01
Why / When should we fine-tune a foundation model?

02
What is a foundation model?

03
Fine-Tuning techniques

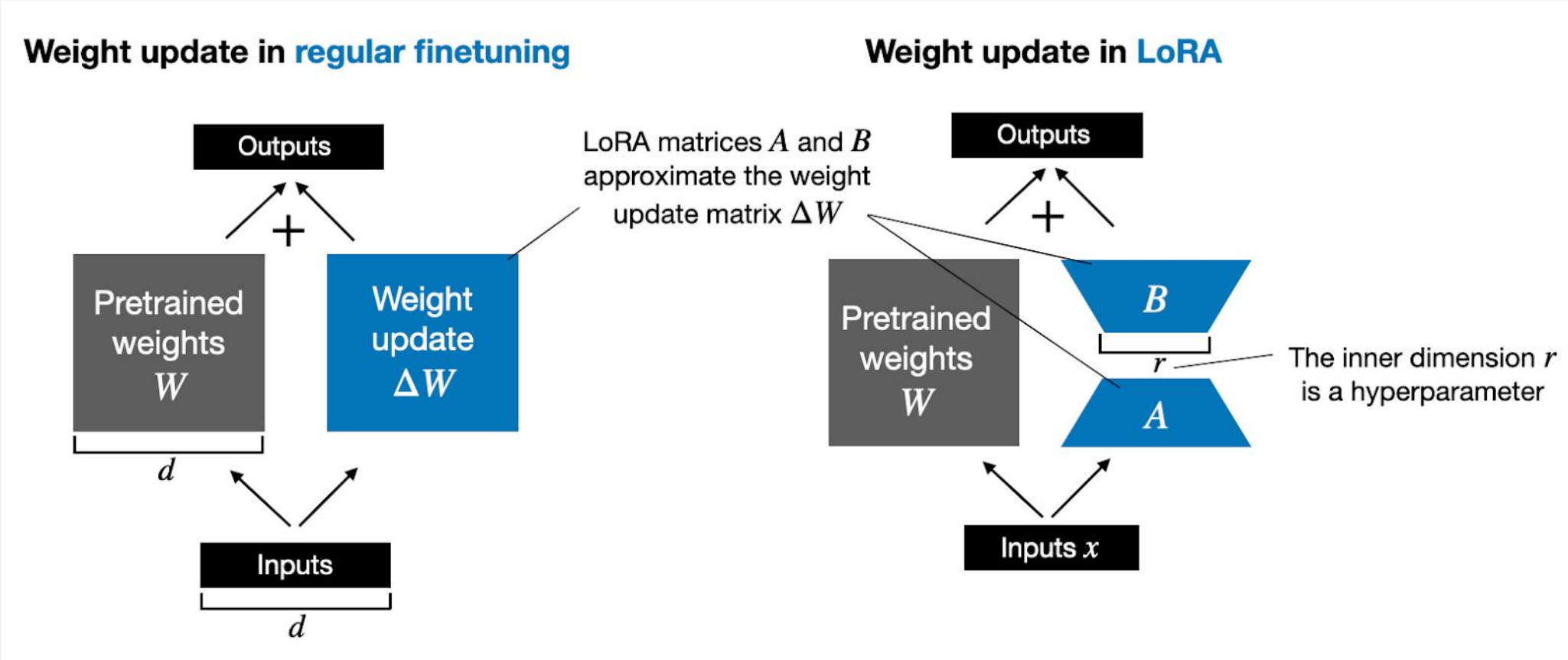
04
What is available on Azure?

05
Demos



LoRA Fine-tuning

LoRA is a parameter-efficient fine-tuning technique that adapts a pre-trained model by learning small, low-rank matrices that represent the weight changes, while keeping the original model weights frozen.
[2106.09685] LoRA: Low-Rank Adaptation of Large Language Models (from Microsoft Research)



Source: https://magazine.sebastianraschka.com/p/practical-tips-for-finetuning-llms

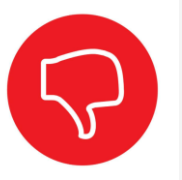
LoRA Fine-Tuning: Key Trade-offs

Advantages

Parameter-Efficient:
Drastically reduces trainable parameters (often <1%). Less memory, faster training, and lower compute costs. Enables fine-tuning on consumer hardware.

Preserves Knowledge:
Base model weights remain frozen, retaining general language understanding and reducing catastrophic forgetting.

Modular Customization:
Small, portable LoRA adapters facilitate easy task-switching & customization without retraining the entire model.



Considerations

Performance Trade-off:
Potential for slightly lower peak performance vs. full fine-tuning, depending on the task (but often negligible). Rank, the size of adapters, must be tuned.

Implementation Overhead:
Some code modification is needed to integrate LoRA layers into the model architecture and correctly apply gradients.

LoRA learns less and forgets less:

Direct Preference Optimization

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

source

Enhance your foundation model with fine-tuning on Azure

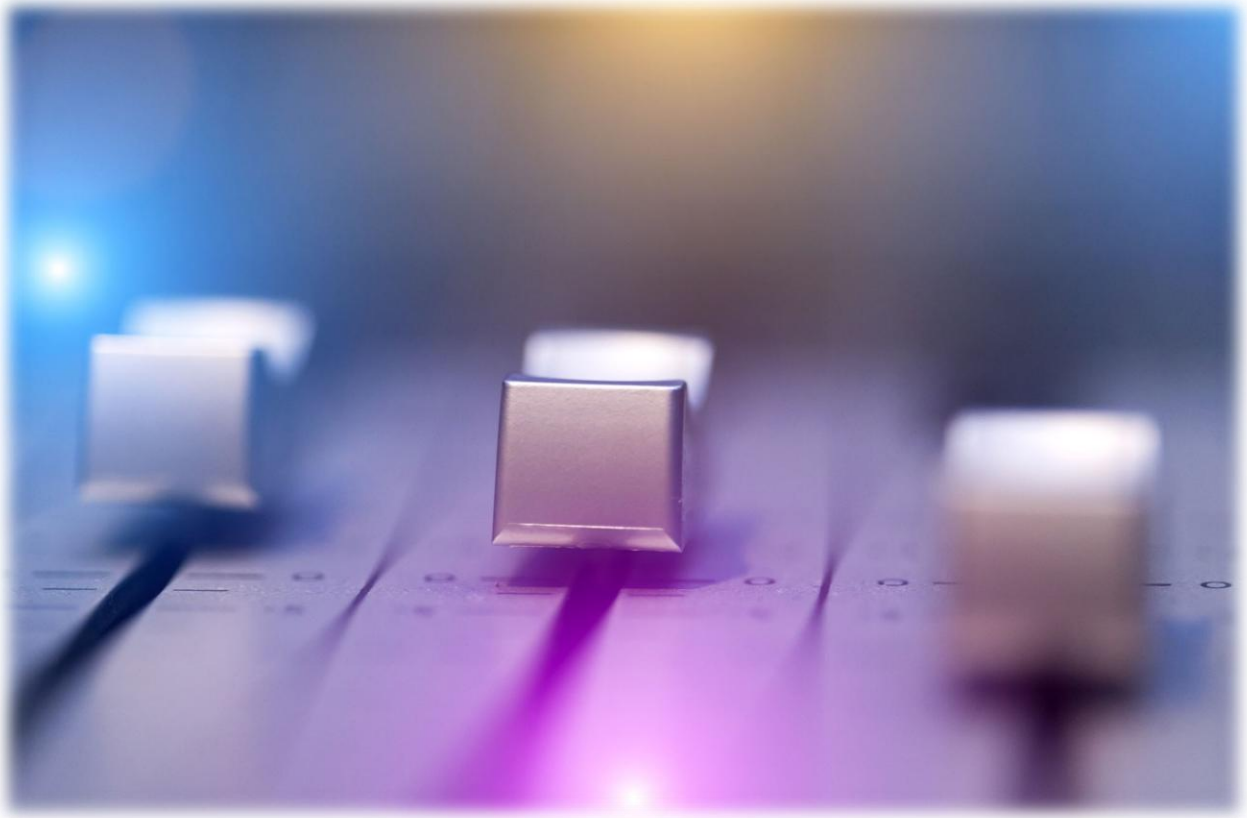
01
Why / When should we fine-tune a foundation model?

02
What is a foundation model?

03
Fine-Tuning techniques

04
What is available on Azure?

05
Demos



Azure Fine-tuning options

Azure OpenAI	Foundry Serverless FT (MaaS)	Foundry Managed FT (MaaP)	AzureML
Fully managed service - Only option for AOAI fine-tuning	Fully managed fine-tuning - no need for own GPU capacity	Bring your own GPU capacity for fine tuning and inference	Bring your own GPU capacity for fine tuning and inference
GPT-3.5-turbo, GPT-4, GPT-4o/4o-mini	Llama3, Phi3, Mistral, NTT	Supported for a larger number of models	All models including custom architectures
regions	regions	All regions supported	All regions supported
Fine-tuning billed per token. Model hosting by the hour. Inferencing is billed per token.	Fine-tuning jobs and model hosting is billed by the hour. Inferencing is billed per token.	Fine-tuning jobs and inference billed for GPUs.	Fine-tuning jobs and inference billed for GPUs.

Complexity & more knowledge required →

Overview does not include Azure AI services specific Fine-tuning options (e.g., Custom Neural Voice training).

What fine-tuning methods do we offer?

Supervised fine-tuning

- GPT-35-Turbo (0125) – 16K context
- GPT-4o (0806) – 128K context
- GPT-4o-mini (0718) – 64k context
- GPT-4.1
- GPT-4.1-mini

Direct Preference Optimization

- GPT-4o (0806)

Reinforcement Learning

- o3-mini

Regional

Global

Dataset Format for Supervised FineTuning

```
{
  "messages": [
    {
      "role": "system",
      "content": "Clippy is a factual chatbot that is also sarcastic."
    },
    {
      "role": "user",
      "content": "Who discovered Antarctica?"
    },
    {
      "role": "assistant",
      "content": "Some chaps named Fabian Gottlieb von Bellingshausen and Mikhail Lazarev, as if they don't teach that in every school!"
    }
  ]
}
```

Each example in your dataset should contain:

- **A system message** – Defines the assistant's behavior, tone, or constraints.
- **A user message** – Represents the input or question from the user.
- **An assistant message** – Contains the model's response to the user.

Dataset Format for DPO

Task instructions

User question

Preferred output

Non-preferred output

```
{
    "input": {
        "messages": [
            {
                "role": "system",
                "content": "You are an insightful assistant capable of exploring philosophical and forward-thinking topics.
                           When presented with a question, provide a succinct answer and focus on the implications of the topic"
            },
            {
                "role": "user",
                "content": "Can renewable energy fully replace fossil fuels?"
            }
        ]
    },
    "preferred_output": [
        {
            "role": "assistant",
            "content": "Technological breakthroughs and global efforts will lead to a sustainable energy future.",
        }
    ],
    “non_preferred_output": [
        {
            "role": "assistant",
            "content": "Dependence on fossil fuels will persist, delaying transitions and worsening climate change.",
        }
    ]
}
```

Each example in your dataset should contain:

- **A prompt**, like a user message.
- **A preferred output** (an ideal assistant response).
- **A non-preferred** output (a suboptimal assistant response).

DPO with Azure OpenAI Service

What is Direct Preference Optimization (DPO)?

- DPO fine-tuning allows you to fine-tune models based on prompts and pairs of responses.
- This approach enables the model to learn from human preferences, optimizing for outputs that are more likely to be favored.
- DPO adjusts model weights based on human preferences, making it computationally lighter and faster than RLHF while being equally effective at alignment.

Benefits of DPO

- **Simplicity**: Eliminates the need for a separate reward model, reducing complexity.
- **Stability**: Avoids instability associated with training multiple models, leading to consistent outcomes.
- **Efficiency**: Computationally efficient, allowing for faster convergence and lower overhead.
- **Bias Mitigation**: Directly incorporates human preferences, reducing unintended biases.

Why is DPO Useful?

- **Subjective Elements**: Beneficial in scenarios where tone, style, or specific content preferences are important.
- **Learning from Examples**: Allows the model to learn from both positive and negative examples.

Dataset Format for Vision Fine-Tuning

Task instructions

User question

Image data
URL or base64

True Answer

```
{
    "messages": [
        {
            "role": "system",
            "content": "You are a Vision Language Model specialized in interpreting visual data from chart images.
                        Your task is to analyze the provided chart image …"
        },
        {
            "role": "user",
            "content": "Between watching videos at dinner time and after dinner, which time had the bigger
                       difference between Pay TV and SVOD?"
        },
        {
            "role": "user",
            "content": [
                {"type": "image_url", "image_url": {"url": "https://example.com/charts/chart001.jpg"}},
              ]
        },
        {
            "role": "assistant",
            "content": "At dinner time"
        }
    ]
}
```

JSONL data file format for training and validation

Fine-tuning options

Azure's managed fine-tuning services streamline the customization process by taking care of the underlying infrastructure and optimization techniques. The compute is serverless, with charges based on hourly consumption. The managed option simplifies the process, allowing users to focus on their use case and data while Azure handles the complexities of LLMOps processes.

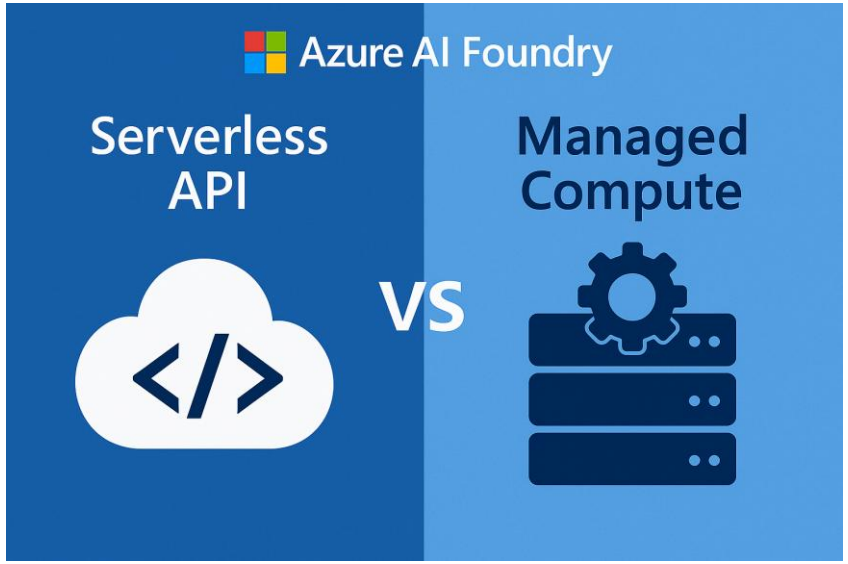
- **Azure OpenAI Service:** Lets users **fine-tune Azure OpenAI models**, enabling customization to specific datasets for enhanced performance.

- **Model-as-a-Service (MaaS):** Through Azure AI Foundry, users can access and fine-tune **various models from different providers**, such as Meta Llama and Microsoft's Phi series. This approach provides flexibility in choosing models that best suit specific use cases.

For MaaS models, there are two additional options: **Serverless API** and **Managed Compute**.

Unless explicitly prompted with a dialog when selecting fine-tuning for the supported model in AI Foundry, Managed Compute is the default option.

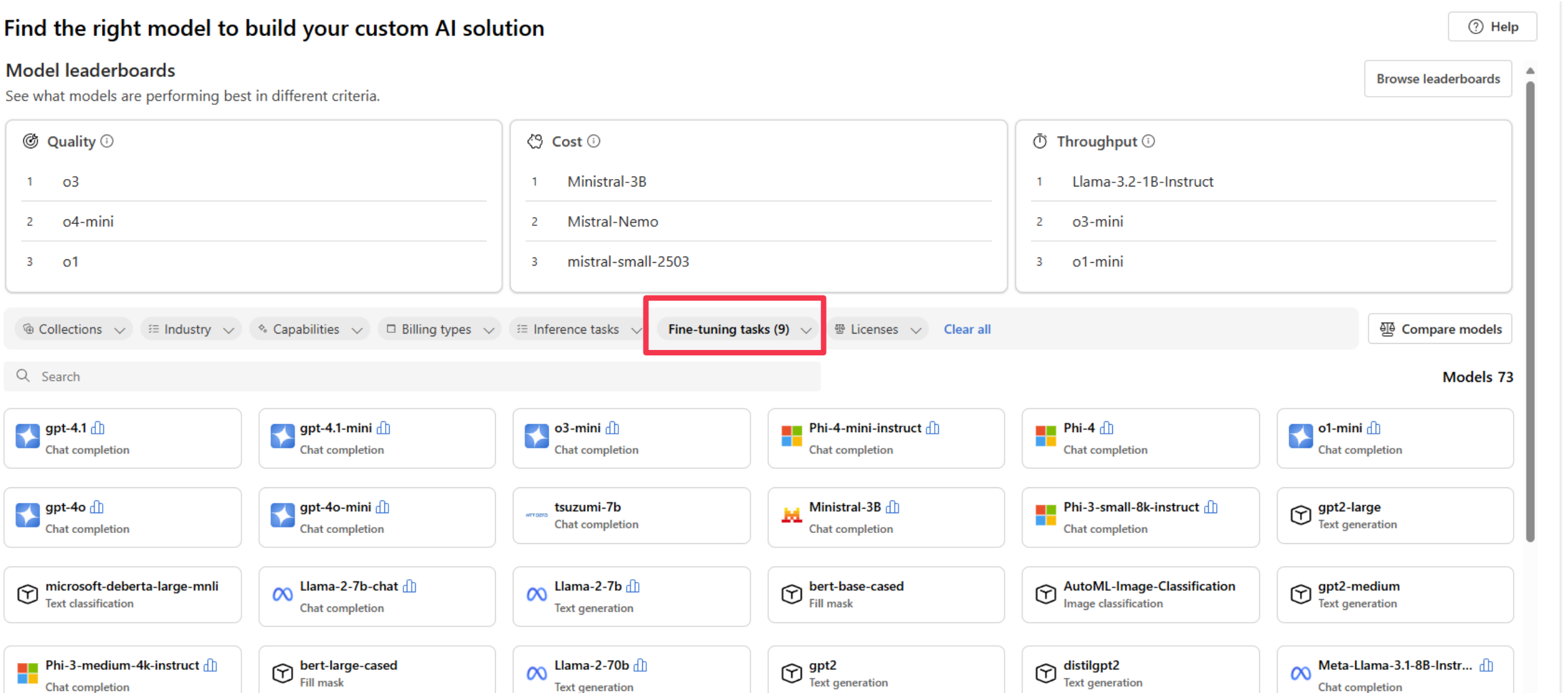
Fine-tuning options



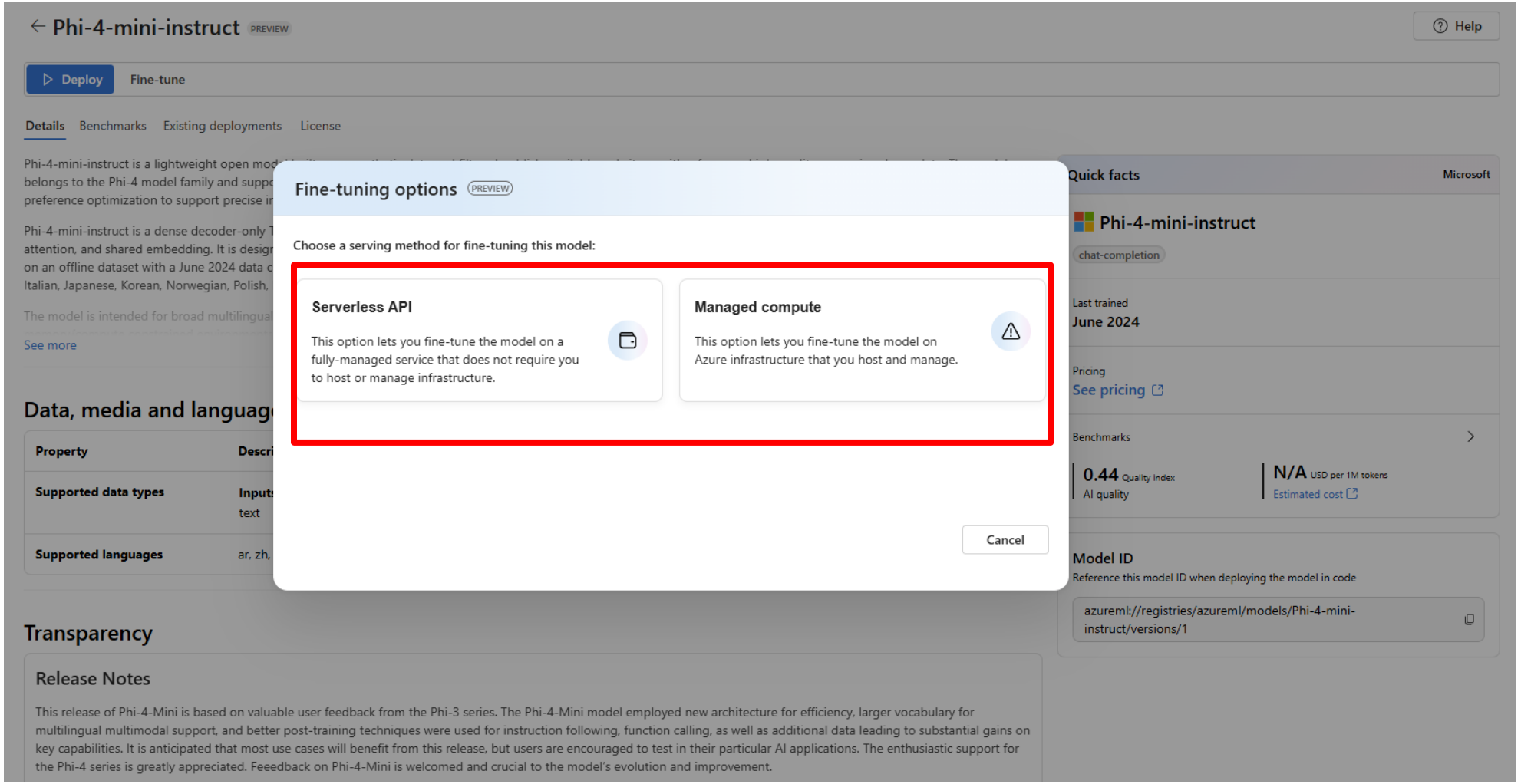
Serverless API means the fine-tuned model is deployed on fully managed infrastructure, priced on a pay-per-token basis, along with a fixed hosting fee. This option eliminates infrastructure management concerns for both fine-tuning and model deployment.

Managed compute means that the portal fine-tuning option for the model is supported, however the actual infrastructure for training and hosting the fine-tuned model needs to be managed by the user. Most of the models are supported via this option.

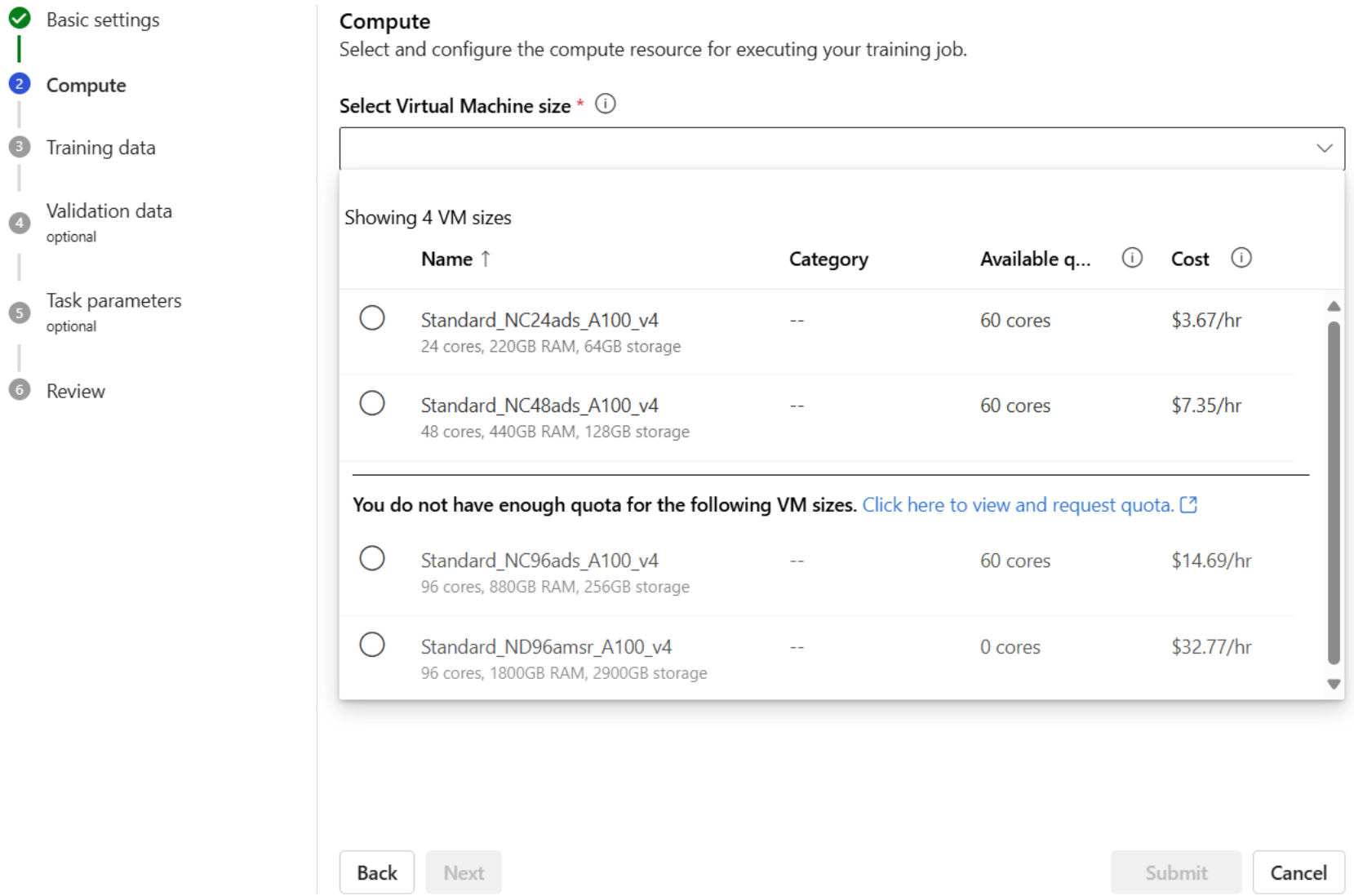
Azure AI Foundry Fine-tuning tasks



Serverless / Managed compute



Managed compute



Pricing

[Azure OpenAI Pricing page]

Region: Sweden Central

Currency: Euro Zone – Euro (€) EUR

1 USD = 0.9261 EUR

Fine-tuning models

Model		Pricing
GPT-4o-2024-08-06	Regional	Input: €2.547/1M tokens Cached Input: €1.402/1M tokens Output: €10.19/1M tokens Training: €25.5/1M tokens Hosting: €1.6/hour
	Global	Input: €2.316/1M tokens Cached Input: €1.158/1M tokens Output: €9.27/1M tokens Training: use regional Hosting: €1.6/hour
GPT-4o-mini	Regional	Input: €0.153/1M tokens Cached Input: €0.085/1M tokens Output: €0.62/1M tokens Training: €3.1/1M tokens Hosting: €1.6/hour
	Global	Input: €0.139/1M tokens Cached Input: €0.070/1M tokens Output: €0.56/1M tokens Training: use regional Hosting: €1.6/hour
GPT-4-0613 (8K)	Regional	Input: €27.783/1M tokens Output: €55.57/1M tokens Training: €74.1/1M tokens Hosting: €4.7/hour
GPT-3.5-Turbo (16K)	Regional	Input: €0.5/1M tokens Output: €1.4/1M tokens Training: €7.41/1M tokens

Enhance your foundation model with fine-tuning on Azure

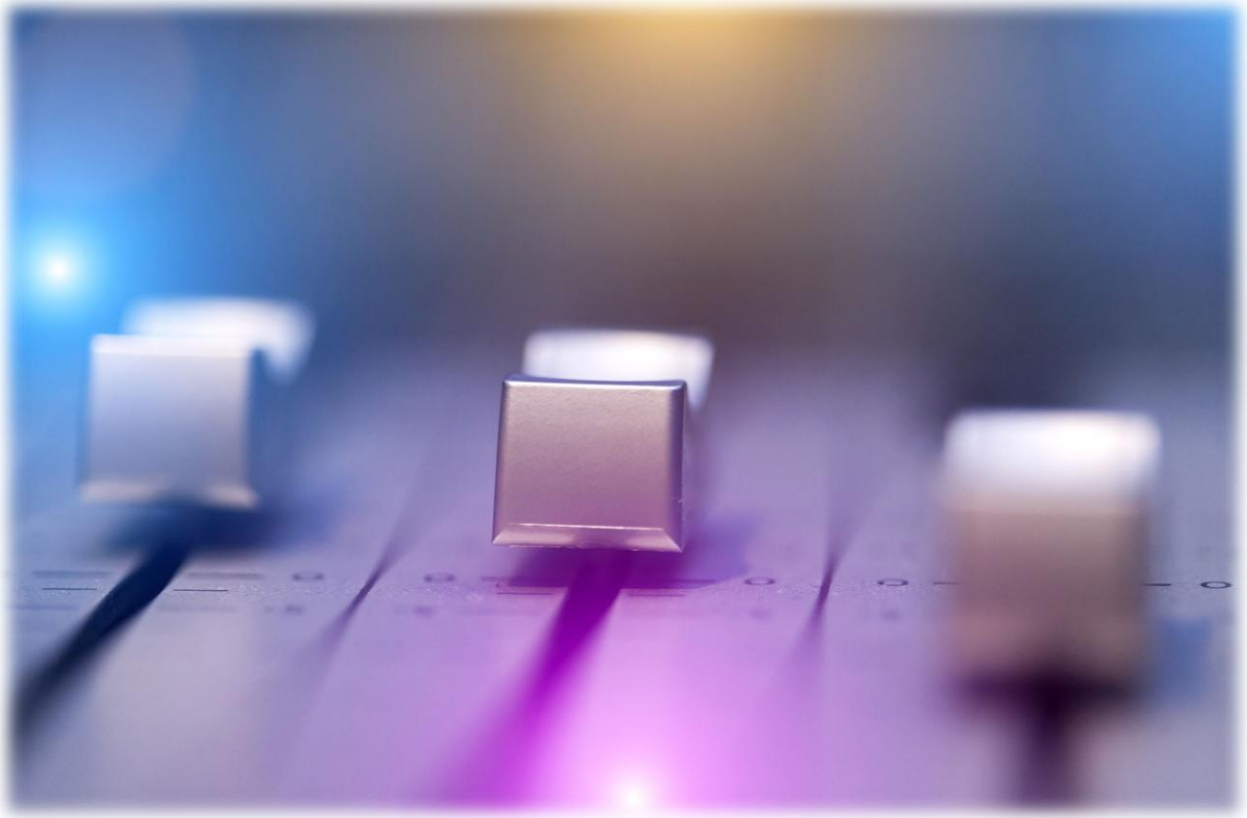
01
Why / When should we fine-tune a foundation model?

02
What is a foundation model?

03
Fine-Tuning techniques

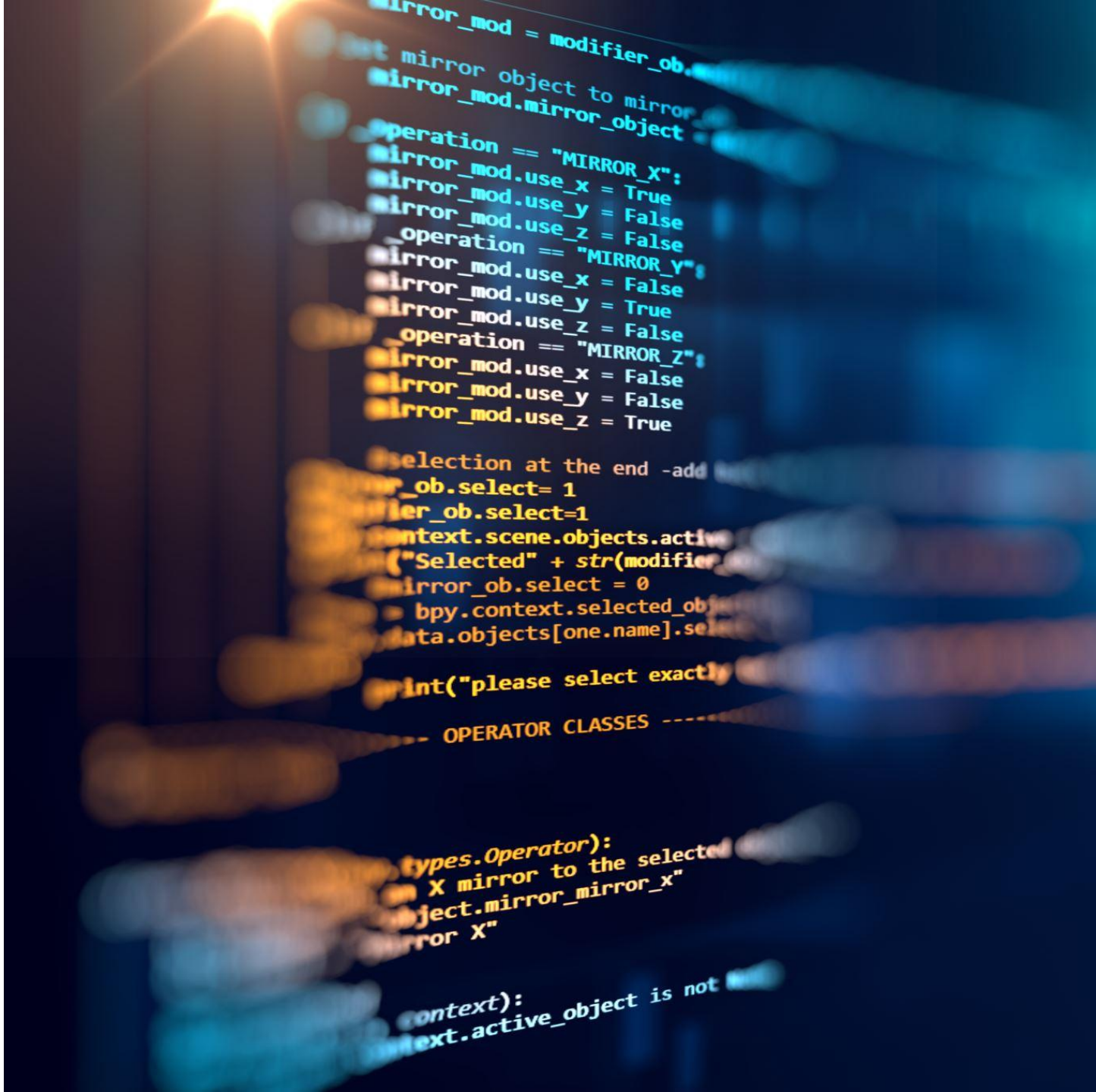
04
What is available on Azure?

05
Demos

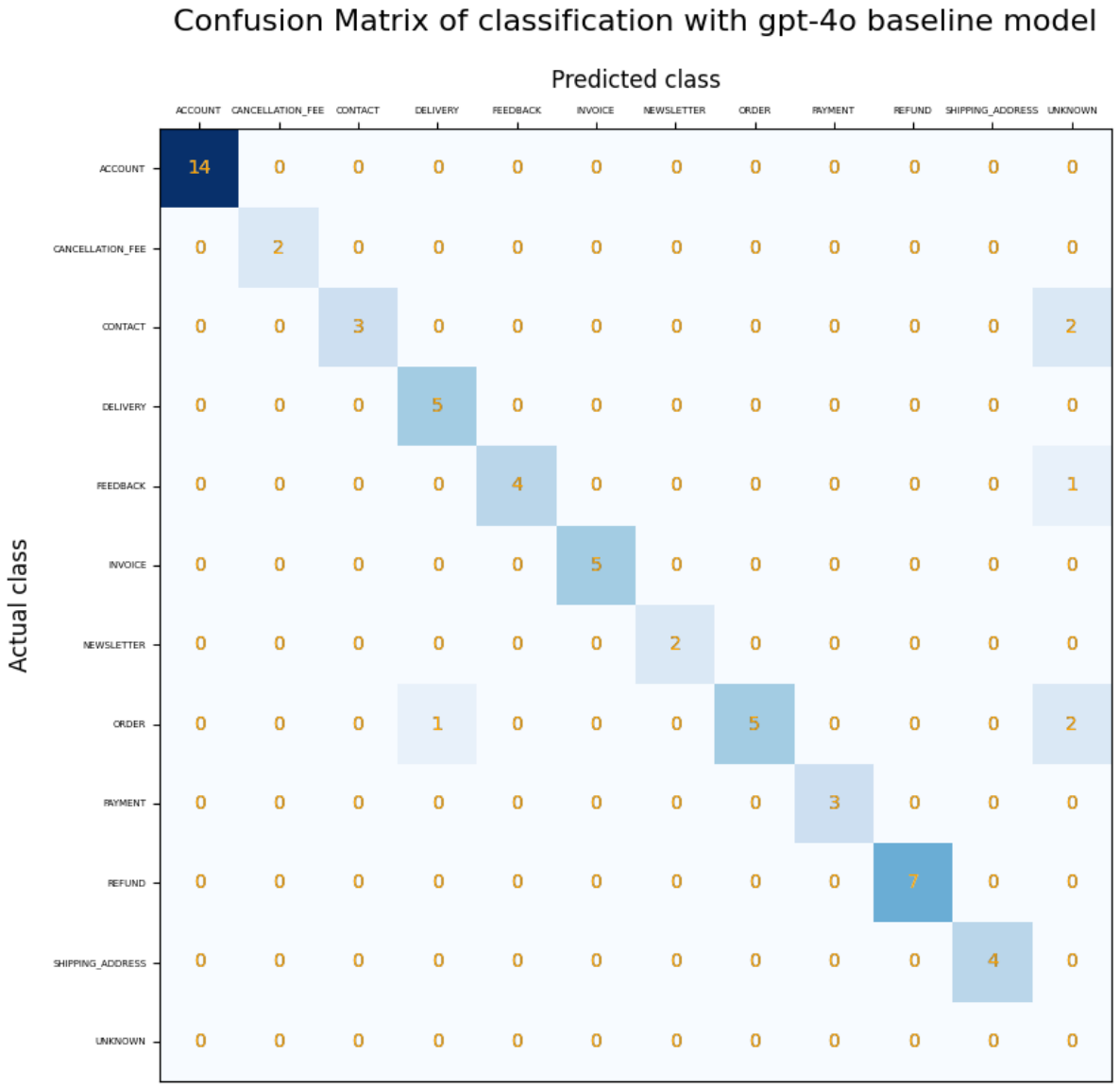


Demo 1

Supervised fine-tuning of GPT-4o for text classification



Supervised fine-tuning of GPT-4o for text classification



Fine-tuning using the Python SDK

Supervised fine-tuning of GPT-4o for text classification
1. Jsonl files

```
training_jsonl = 'customer_training.jsonl'
```

```
json_train_list = []

for index, row in ds_train.iterrows():
    json_obj = {
        "messages": [
            {
                "role": "system", "content": str(prompt)
            },
            {
                "role": "user", "content": str(row["text"])
            },
            {
                "role": "assistant", "content": str(row["category"])
            },
        ]
    }
    json_train_list.append(json_obj)

# Saving into the jsonl file
with open(training_jsonl, "w") as f:
    for json_obj in json_train_list:
        f.write(json.dumps(json_obj) + "\n")
```

Text classification with gpt-4o fine tuned model

Supervised fine-tuning of GPT-4o for text classification
2. Uploading jsonl files into Azure AI Foundry

Uploading the jsonl files into Azure AI Foundry

```
train_file = aoai_client.files.create(file=open(training_jsonl, "rb"), purpose="fine-tune")
```

```
val_file = aoai_client.files.create(file=open(validation_jsonl, "rb"), purpose="fine-tune")
```

```
train_file
```

```
FileObject(id='file-0c98ff00ebdb4054aff6edce70249315', bytes=145399, created_at=1738837265, filename='customer_training.jsonl', object='file', purpose='fine-tune', status='pending', status_details=None)
```

```
val_file
```

```
FileObject(id='file-008d6e3a091a4ba1b3de649d3c396407', bytes=31138, created_at=1738837266, filename='customer_validation.jsonl', object='file', purpose='fine-tune', status='pending', status_details=None)
```

```
print(f"File name:  {train_file.filename}")
print(f"File size:  {(train_file.bytes / (1024 * 1024)):.2f} MB")
print(f"Created at: {datetime.fromtimestamp(train_file.created_at)}")
```

```
File name:  customer_training.jsonl
File size:  0.14 MB
Created at: 2025-02-06 10:21:05
```

```
print(f"File name:  {val_file.filename}")
print(f"File size:  {(val_file.bytes / (1024 * 1024)):.2f} MB")
print(f"Created at: {datetime.fromtimestamp(val_file.created_at)}")
```

```
File name:  customer_validation.jsonl
File size:  0.03 MB
Created at: 2025-02-06 10:21:06
```

Text classification with gpt-4o fine tuned model

Supervised fine-tuning of GPT-4o for text classification
3. Fine-tuning training

3. Running the fine tuning job

```
# Create fine tuning job
file_train = train_file.id
file_val = val_file.id

ft_job = aoai_client.fine_tuning.jobs.create(
    suffix=project_name,
    training_file=file_train,
    validation_file=file_val,
    model=baseline_model,
    method="sft",  # sft=Supervised Fine tuning (default method). DPO is available as well.
    seed=None,  # Seed to ensure the same randomization and consistent training results across runs. If no seed is specified, one will be generated.
    hyperparameters={
        "n_epochs": None,  # Number of epochs (passes through the entire dataset). 'Auto' by default.
        "batch_size": None,  # Number of examples per batch. 'Auto' by default.
        "learning_rate_multiplier": None  # Scale factor for the learning rate. 'Auto' by default.
    })
```

Text classification with gpt-4o fine tuned model

Supervised fine-tuning of GPT-4o for text classification
4. End of training

```
# Retrieve the state of a fine-tune job
ft_job = aoai_client.fine_tuning.jobs.retrieve(job_id)
ft_job.to_dict()
```


```
{'id': 'ftjob-c40e7b411fae4e75a63042e6a4cbcf6d',
 'created_at': 1738837274,
 'fine_tuned_model': 'gpt-4o-2024-08-06.ft-c40e7b411fae4e75a63042e6a4cbcf6d-callcenter',
 'finished_at': 1738843690,
 'hyperparameters': {'batch_size': 1,
  'learning_rate_multiplier': 1.0,
  'n_epochs': 3},
 'model': 'gpt-4o-2024-08-06',
 'object': 'fine_tuning.job',
 'result_files': ['file-14529be0515648e8bccdb77e620f1f0b'],
 'seed': 1718536147,
 'status': 'succeeded',
 'trained_tokens': 108837,
 'training_file': 'file-0c98ff00ebdb4054aff6edce70249315',
 'validation_file': 'file-008d6e3a091a4ba1b3de649d3c396407',
 'estimated_finish': 1738838190,
 'suffix': 'callcenter'}
```


```
start_dt = datetime.fromtimestamp(ft_job.to_dict()['created_at'])
end_dt = datetime.fromtimestamp(ft_job.to_dict()['finished_at'])

print(f"{start_dt} Model training starts")
print(f"{end_dt} End of model training")
print(f"Training duration: {(end_dt - start_dt)}")
trained_tokens = ft_job.to_dict()["trained_tokens"]
print(f"Number of trained tokens: {trained_tokens:,}")
```

```
2025-02-06 10:21:14 Model training starts
2025-02-06 12:08:10 End of model training
Training duration: 1:46:56
Number of trained tokens: 108,837
```

Text classification with gpt-4o fine tuned model

Supervised fine-tuning of GPT-4o for text classification
5. Metrics

You can check the metrics of the Azure AI Foundry portal as well.

```
data_io = StringIO(aoai_client.files.content(ft_job.to_dict()['result_files'][0]).content.decode())
results_df = pd.read_csv(data_io)
results_df
```

	step	train_loss	train_mean_token_accuracy	train_error_rate	valid_loss	valid_mean_token_accuracy	valid_error_rate	full_valid_loss
0	1	0.002886	1.000000	NaN	0.001381	1.000000	NaN	NaN
1	2	0.001554	1.000000	NaN	0.003025	1.000000	NaN	NaN
2	3	0.028144	1.000000	NaN	0.001523	1.000000	NaN	NaN
3	4	0.020215	1.000000	NaN	0.159286	1.000000	NaN	NaN
4	5	0.000238	1.000000	NaN	0.000490	1.000000	NaN	NaN
...	...	...	...	...	...	...	...	...
835	836	0.000902	0.000000	NaN	0.000491	0.500000	NaN	NaN
836	837	0.000120	0.600000	NaN	0.000221	0.333333	NaN	NaN
837	838	0.000199	0.333333	NaN	0.000250	0.500000	NaN	NaN
838	839	0.000206	0.333333	NaN	0.000160	0.333333	NaN	NaN
839	840	0.000112	0.750000	NaN	0.000260	0.000000	NaN	0.024622

840 rows × 12 columns

[Text classification with gpt-4o fine tuned model]

Supervised fine-tuning of GPT-4o for text classification
6. Model deployment

6. Model deployment

```
my_models = aoai_client.models.list().to_dict()

models_df = date_sorted_df(my_models['data'])
cols = ['status', 'capabilities', 'lifecycle_status', 'id', 'created_at', 'model']
bold_start, bold_end = '\033[1m', '\033[0m'
print(f"Models of Azure OpenAI resource {bold_start}{resource_name}{bold_end}:")
display(models_df[cols].head())
```

Models of Azure OpenAI resource **azureopenai-sweden-sr**:

	status	capabilities	lifecycle_status	id	created_at	model
88	succeeded	{'fine_tune': False, 'inference': True, 'compl...	generally-available	gpt-4o-2024-08-06.ft-c40e7b411fae4e75a63042e6a...	2025-02-06 10:21:14	gpt-4o-2024-08-06
87	succeeded	{'fine_tune': False, 'inference': True, 'compl...	generally-available	gpt-4o-2024-08-06.ft-c40e7b411fae4e75a63042e6a...	2025-02-06 10:21:14	gpt-4o-2024-08-06
86	succeeded	{'fine_tune': False, 'inference': True, 'compl...	generally-available	gpt-4o-2024-08-06.ft-c40e7b411fae4e75a63042e6a...	2025-02-06 10:21:14	gpt-4o-2024-08-06
32	succeeded	{'fine_tune': False, 'inference': True, 'compl...	preview	gpt-4o-mini-audio-preview-2024-12-17	2025-02-03 00:00:00	NaN
85	succeeded	{'fine_tune': True, 'inference': True, 'comple...	generally-available	gpt-4o-mini-2024-07-18.ft-82d7922a32084532a1d9...	2025-01-23 16:53:53	gpt-4o-mini-2024-07-18

[Text classification with gpt-4o fine tuned model]

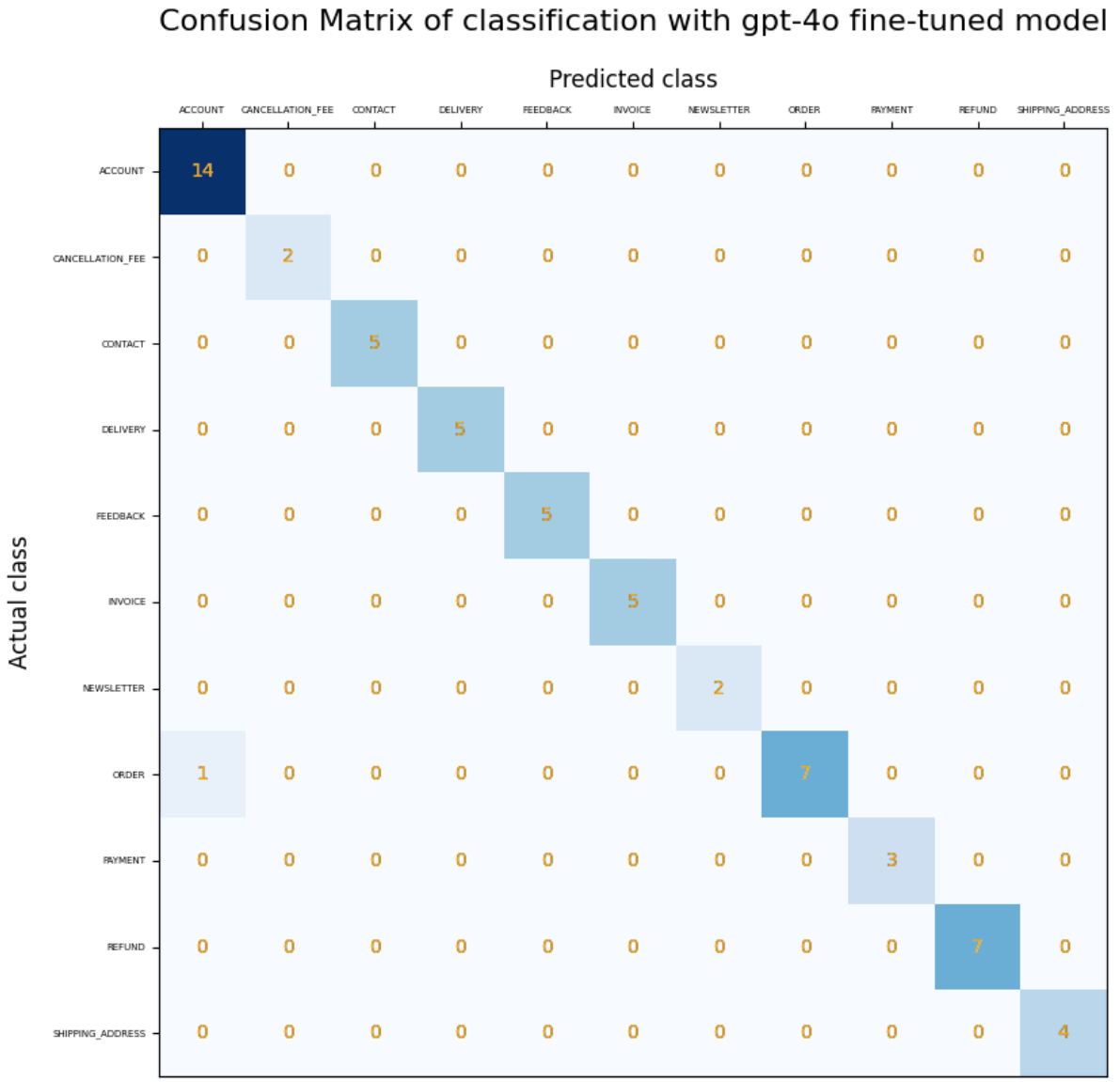
Supervised fine-tuning of GPT-4o for text classification
7. Use of the deployed model

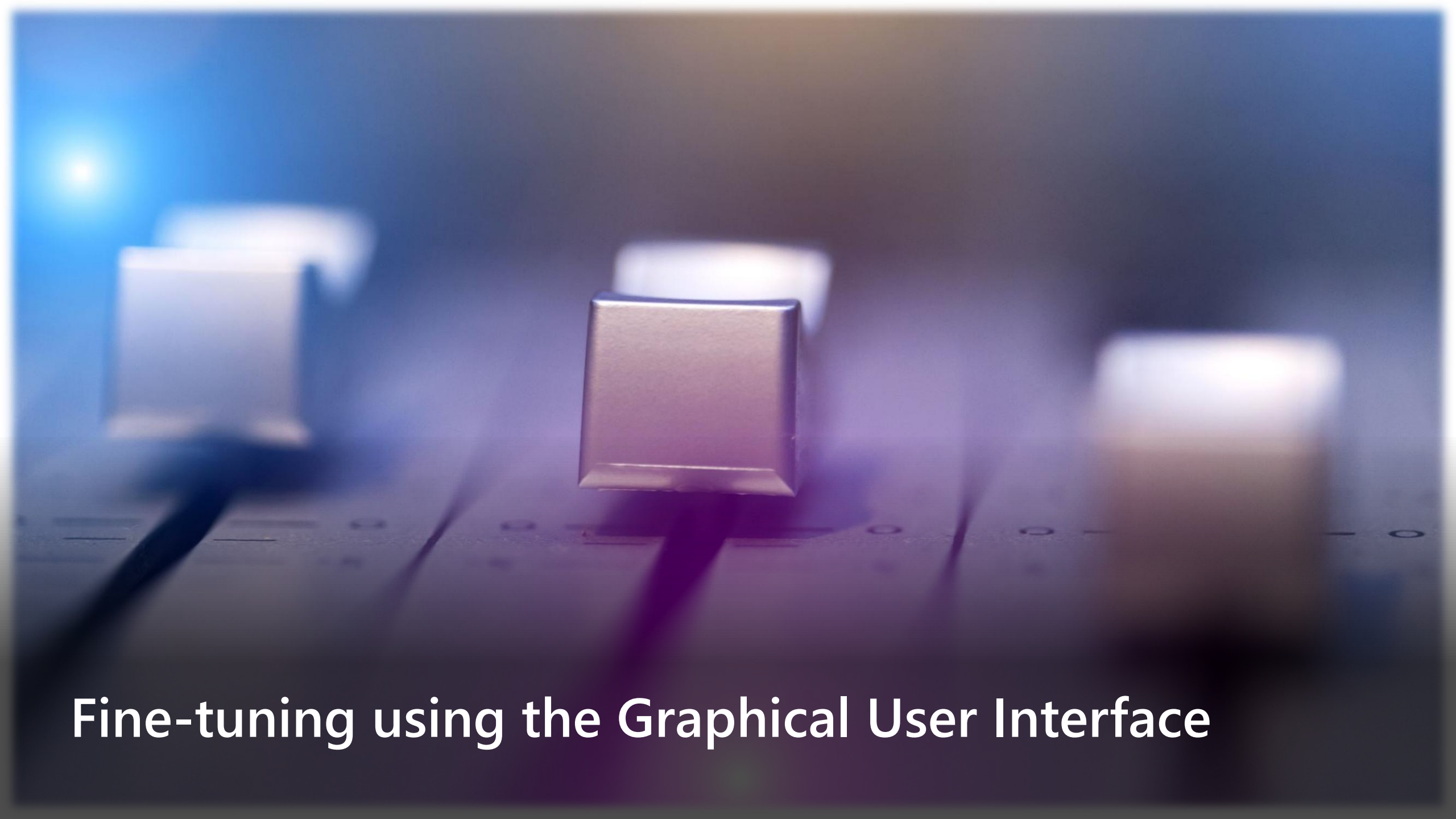
```
# Process test df with the fine-tuned model
print("Fine-tuned model:", project_name)

tqdm.pandas()
ds_test['gpt-4o-ft-pred'] = ds_test.progress_apply(
    lambda row: calling_classification_model(f"{row['text']}", project_name),
    axis=1
)
```

```
Fine-tuned model: callcenter
100%|██████████| 60/60 [01:50<00:00,  1.85s/it]
```

[Text classification with gpt-4o fine tuned model]

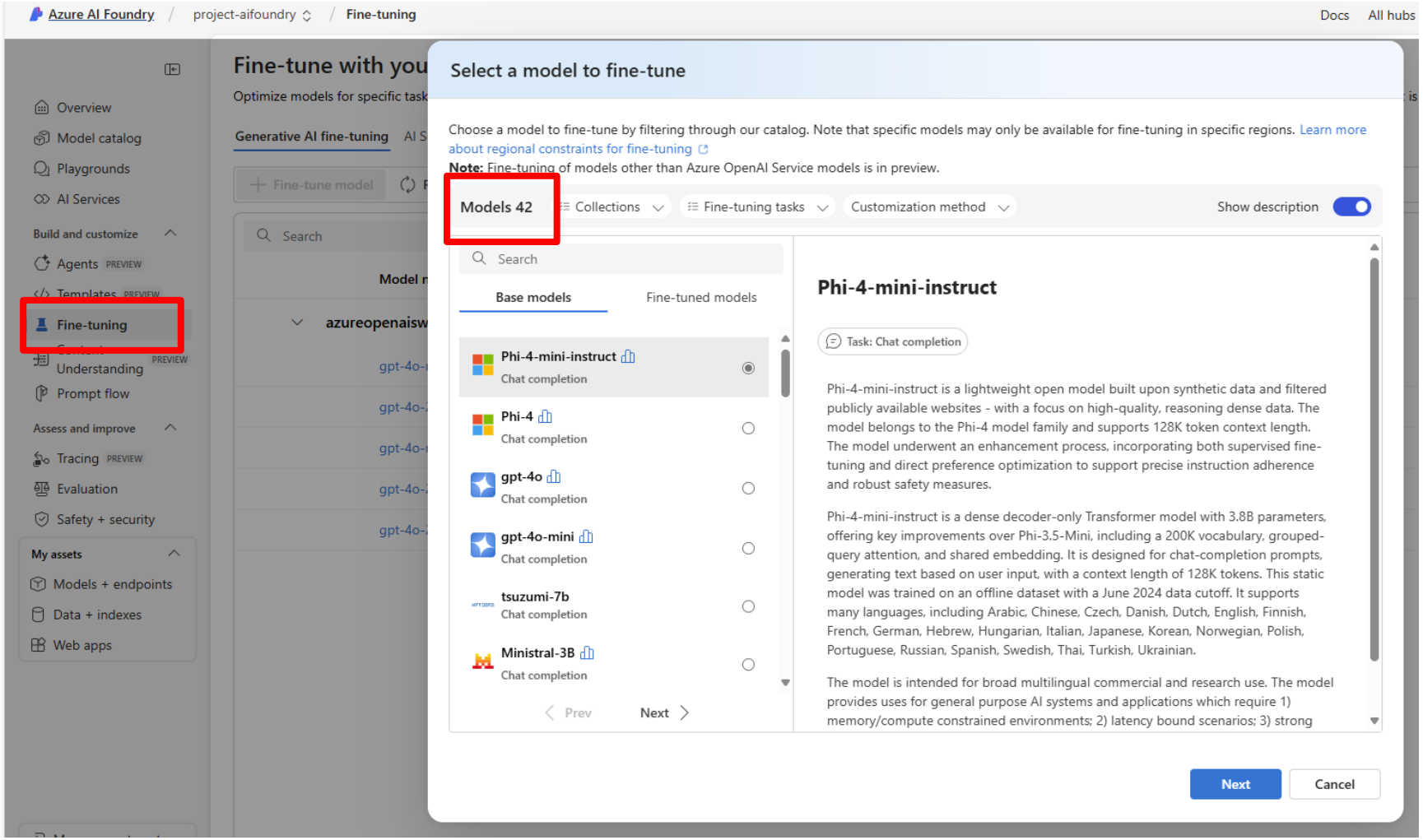
Supervised fine-tuning of GPT-4o for text classification



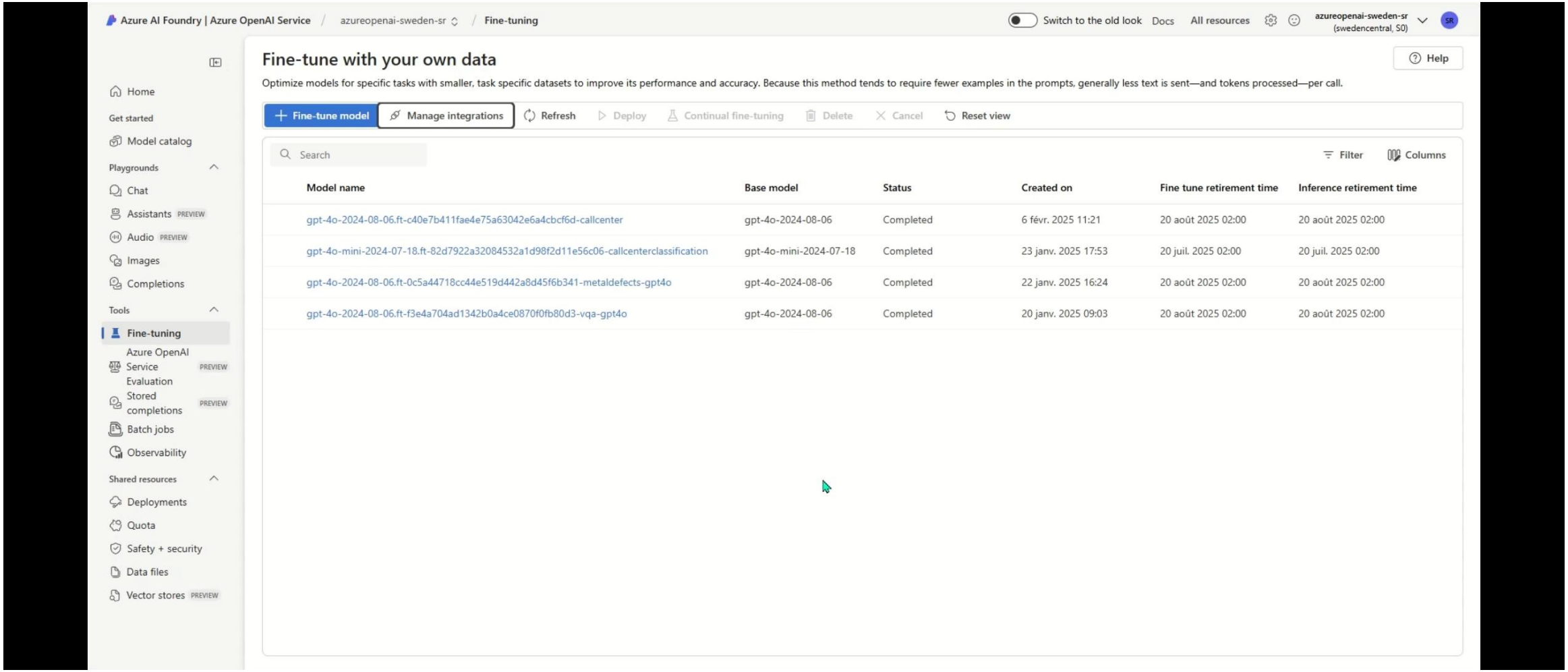


Fine-tuning using the Graphical User Interface

From Azure AI Foundry

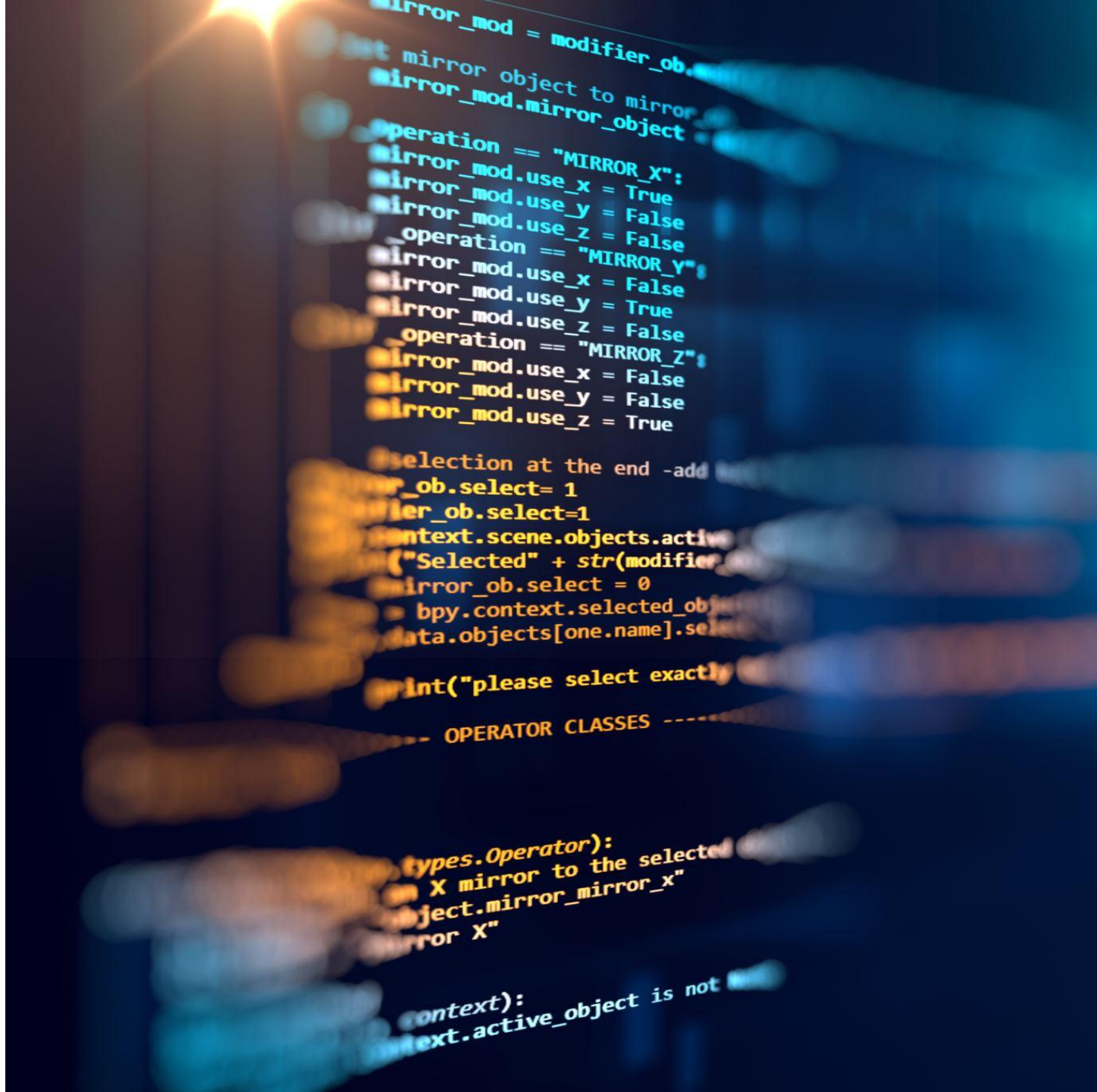


Supervised fine-tuning of GPT-4o for text classification



Demo 2

Supervised Fine-tuning for Chart Analysis



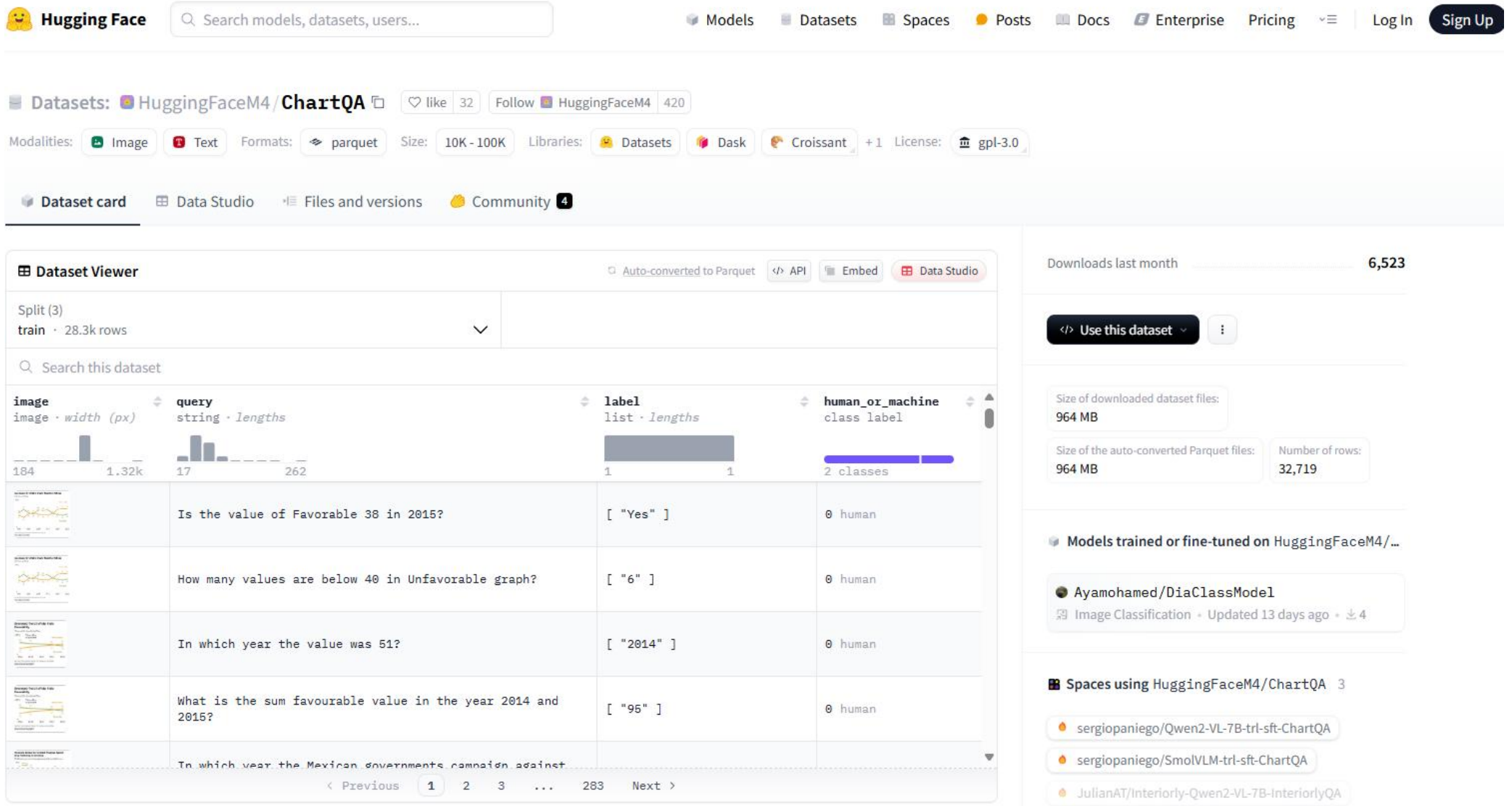
Supervised Fine-tuning for Chart Analysis

- This notebook demonstrates the process of **vision fine-tuning of GPT-4o** leveraging a chart analysis benchmark dataset for visual and logical reasoning. It covers data preparation, fine-tuning, deployment, and evaluation against GPT-4o's baseline performance.

- We are using the **ChartQA** dataset, a benchmark designed for question answering tasks involving chart images. Each entry in the dataset comprises a chart image, an associated question, and the corresponding answer, facilitating the development and evaluation of models that integrate visual and logical reasoning to interpret and analyze information presented in graphical formats.

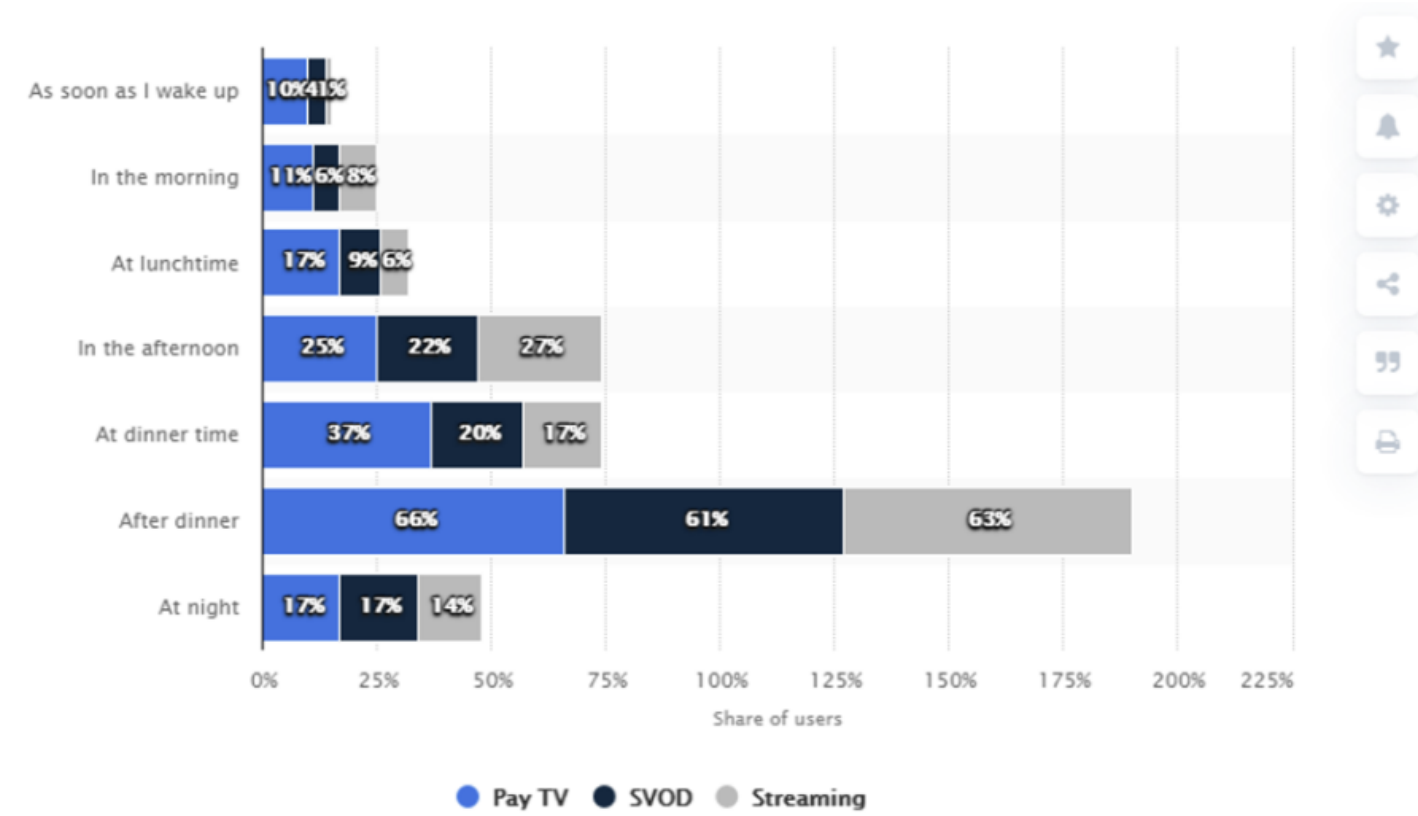
https://github.com/retkowsky/Azure-OpenAI-demos/blob/main/Gpt-4o-Text-FineTuning/Text classification with gpt-4o fine tuned model.ipynb

Supervised Fine-tuning for Chart Analysis



[HuggingFaceM4/ChartQA · Datasets at Hugging Face]

Supervised Fine-tuning for Chart Analysis



Question

Between watching videos at dinner time and after dinner, which time had the bigger difference between Pay TV and SVOD?

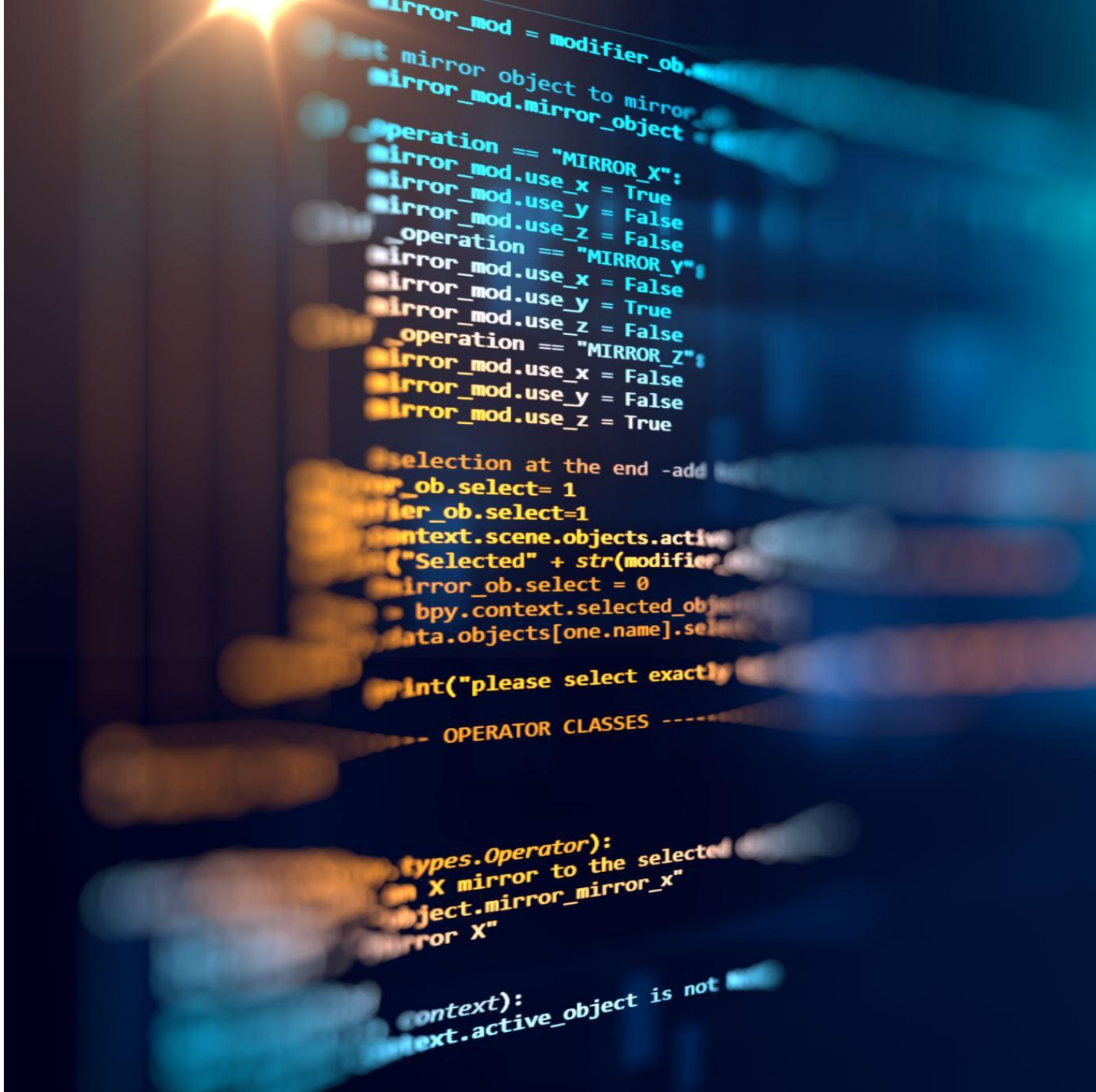
Answer

At dinner time

gen-cv/vision-fine-tuning/01-AOAI-vision-fine-tuning-starter/fine-tune-aoai-gpt4o-for-chart-analysis.ipynb at main · Azure/gen-cv

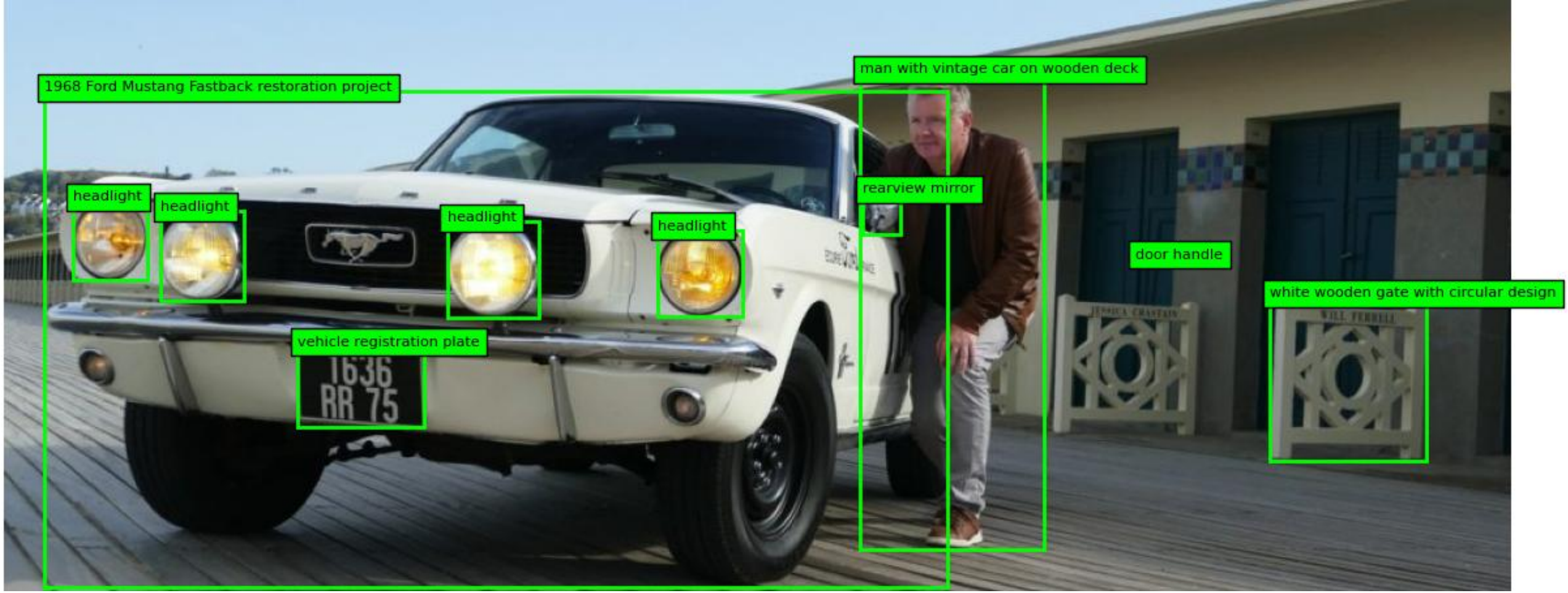
Demo 3

Florence-2 fine tuning for object detection



Florence-2

- **Florence-2** is a lightweight vision-language model open-sourced by Microsoft under the MIT license.
- It offers strong **zero-shot** and **fine-tuning** capabilities for tasks such as **image captioning, object detection, visual grounding, and segmentation.**
- Despite its compact size, training on the extensive **FLD-5B dataset (126 million images and 5.4 billion annotations)** enables Florence-2 to perform on par with much larger models like Kosmos-2.
- microsoft/Florence-2-large · Hugging Face



Florence-2 (Zero shot example)

[florence-2/Florence-2 - Advancing a Unified Representation for a Variety of Vision Tasks.ipynb at main · retkowsky/florence-2]

Florence-2 (Zero shot example)

[florence-2/Florence-2 - Advancing a Unified Representation for a Variety of Vision Tasks.ipynb at main · retkowsky/florence-2]

Florence-2 (Zero shot example)

[florence-2/Florence-2 - Advancing a Unified Representation for a Variety of Vision Tasks.ipynb at main · retkowsky/florence-2]

Florence-2 (Zero shot example)

[florence-2/Florence-2 - Advancing a Unified Representation for a Variety of Vision Tasks.ipynb at main · retkowsky/florence-2]

Florence-2 fine-tuning for object detection

```
Number of detected labels = 1

 1. Detected label: POTHOLE with Bounding-Box: [164.1999969482422, 93.1500015258789, 335.0, 247.0500030517578]
```

florence-2/finetuning-objectdetection/Fine-tuning Florence-2 for Object Detection wih Maestro.ipynb at main · retkowsky/florence-2

Agenda

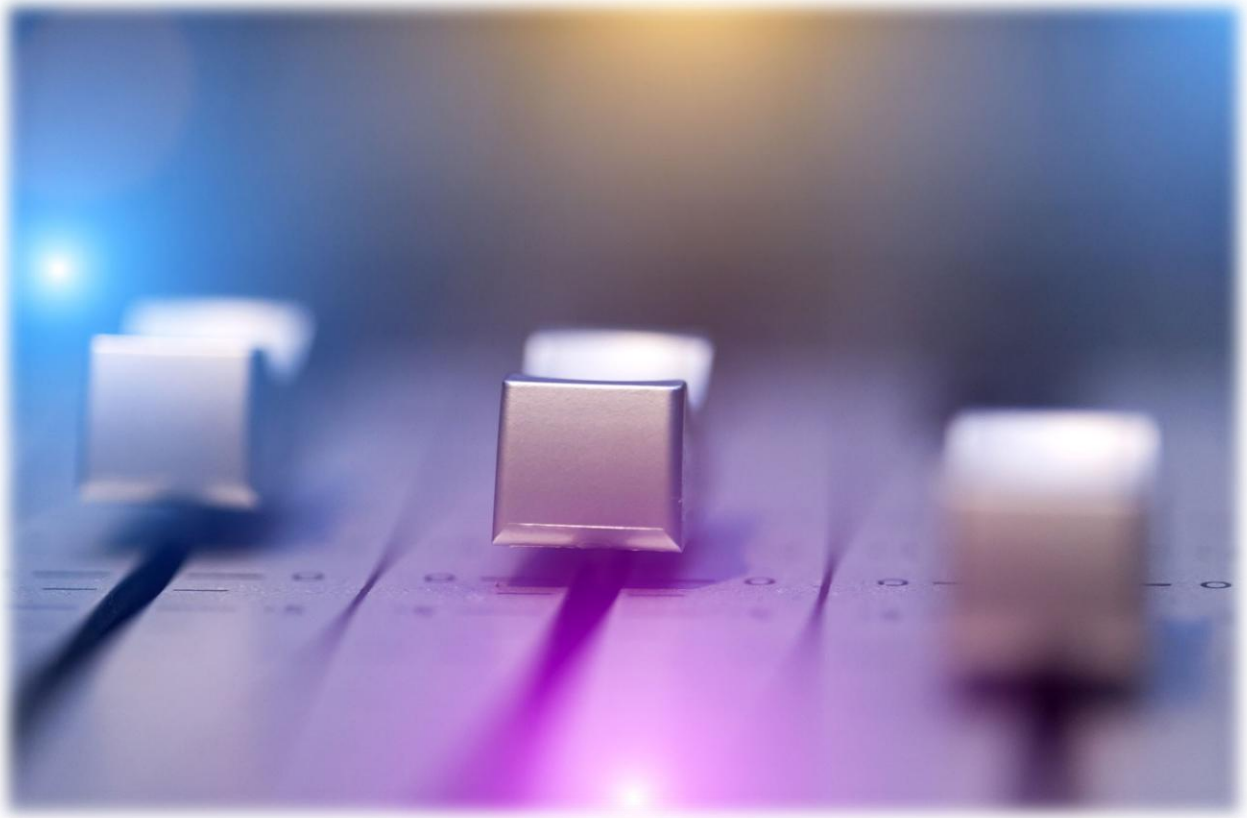
01
Why / When should we fine-tune a foundation model?

02
What is a foundation model?

03
Fine-Tuning techniques

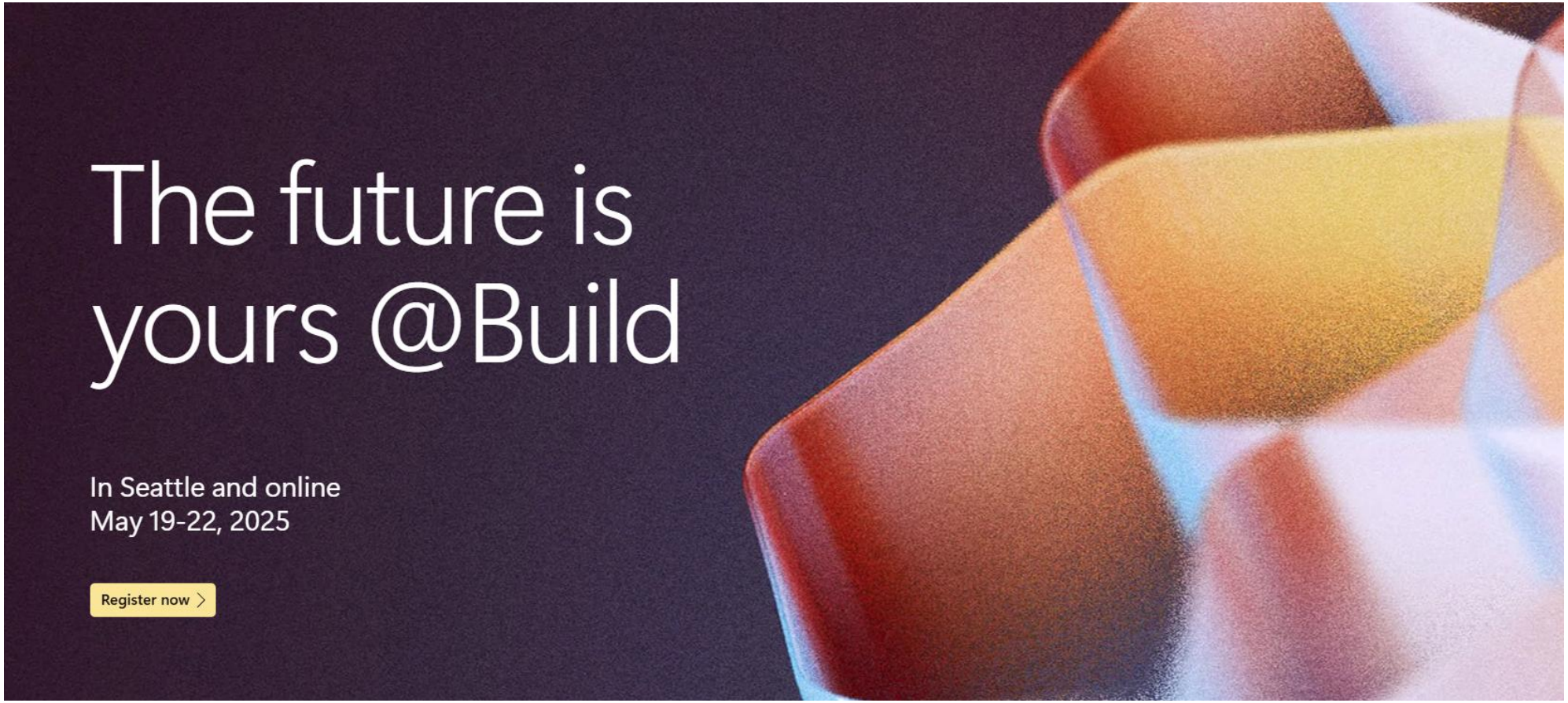
04
What is available on Azure?

05
Demos



Microsoft a 50 ans

Celebrating Microsoft's 50 years - The Official Microsoft Blog



Microsoft Build | May 19-22, 2025

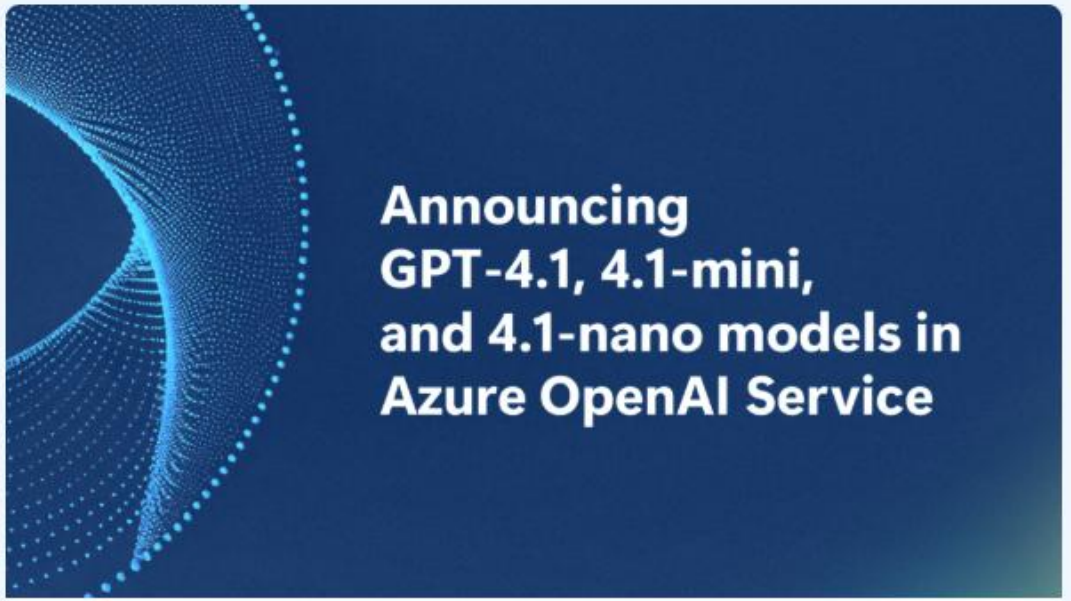
https://build.microsoft.com/en-US/home

Navigate blog by: Content type Product Audience Search the blog

Announcements · Apr 16, 2025 · 2 min read

o3 and o4-mini: Unlock enterprise agent workflows with next-level reasoning AI with Azure AI Foundry and GitHub

We are thrilled to announce the availability of the latest iterations in the o* reasoning series: o3 and o4-mini models on the Microsoft Azure OpenAI Service.



Announcements · Apr 14, 2025 · 2 min read

Announcing the GPT-4.1 model series for Azure AI Foundry and GitHub developers

We are excited to share the launch of the next generation of the GPT-4o model series with GPT-4.1, 4.1-mini, and 4.1-nano to Microsoft Azure OpenAI Service.

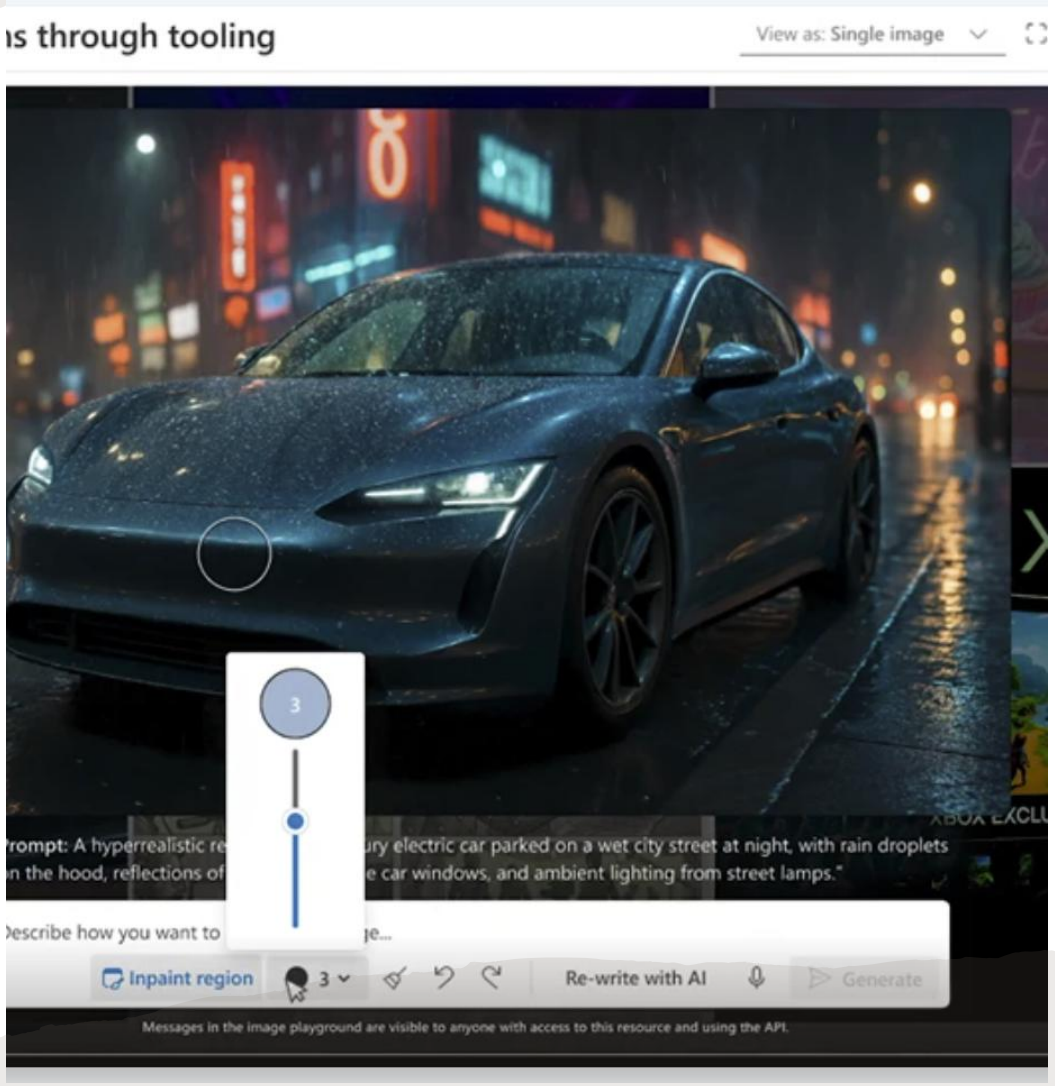
Microsoft Azure Blog

Global AI community France

https://www.linkedin.com/company/global-ai-community-france/

Unveiling GPT-image-1: Rising to new heights with image generation in Azure AI Foundry

- **Text-to-image**: Generate images from text prompts, similar to text2im in ChatGPT DALL-E.
- **Image-to-image**: Create new images from user-uploaded images and text prompts, a feature not available in ChatGPT DALL-E.
- **Text transformation**: Edit images using text prompts, akin to the transform feature in ChatGPT DALL-E.
- **Inpainting**: Edit images with text prompts and user-drawn bounding boxes, similar to inpainting with DALL-E.
- [Unveiling GPT-image-1: Rising to new heights with image generation in Azure AI Foundry | Microsoft Azure Blog]



Q&A

Microsoft

MERCI TACK ขอบคุณครับ KIITOS MULŢUMESC DANKE

SALMAT PO GRACIAS 감사합니다 شكراً DZIĘKUJĘ

OBRIGADO MAHALO TEŞEKKÜRLER ευχαριστώ धन्यवाद

СПАСИБО நன்றி

Thank you!

תודה 多謝晒

PALDIES ДЯКУЮ ĎAKUJEM متشكرم TERIMA KASIH TAK

HVALA شكريه DĚKUJI 谢谢 CÁM ƠN DANKON

GRAZIE KÖSZÖNÖM БЛАГОДАРЯ ありがとうございます