



Fine-Tuning des modèles fondationnels avec Azure AI Foundry

Jeudi 24 avril 2025



global AI community

Virtuelle Francophone

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.



Serge RETKOWSKY

France AI Global Black Belt
@Microsoft



Farid EL ATTAOUI

France AI Global Black Belt
@Microsoft



Nicolas ROBERT

AI Architect @Expertime
Microsoft MVP IA



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

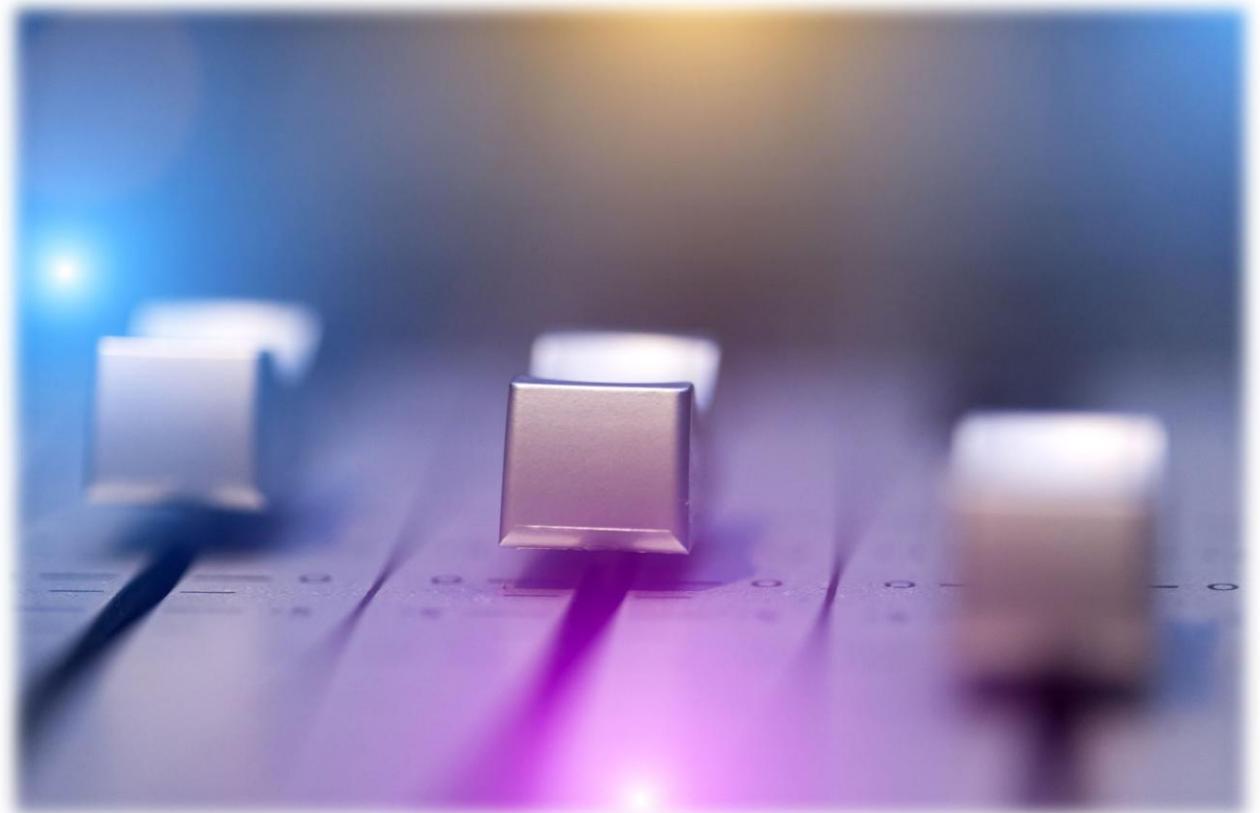
Fine-Tuning techniques

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

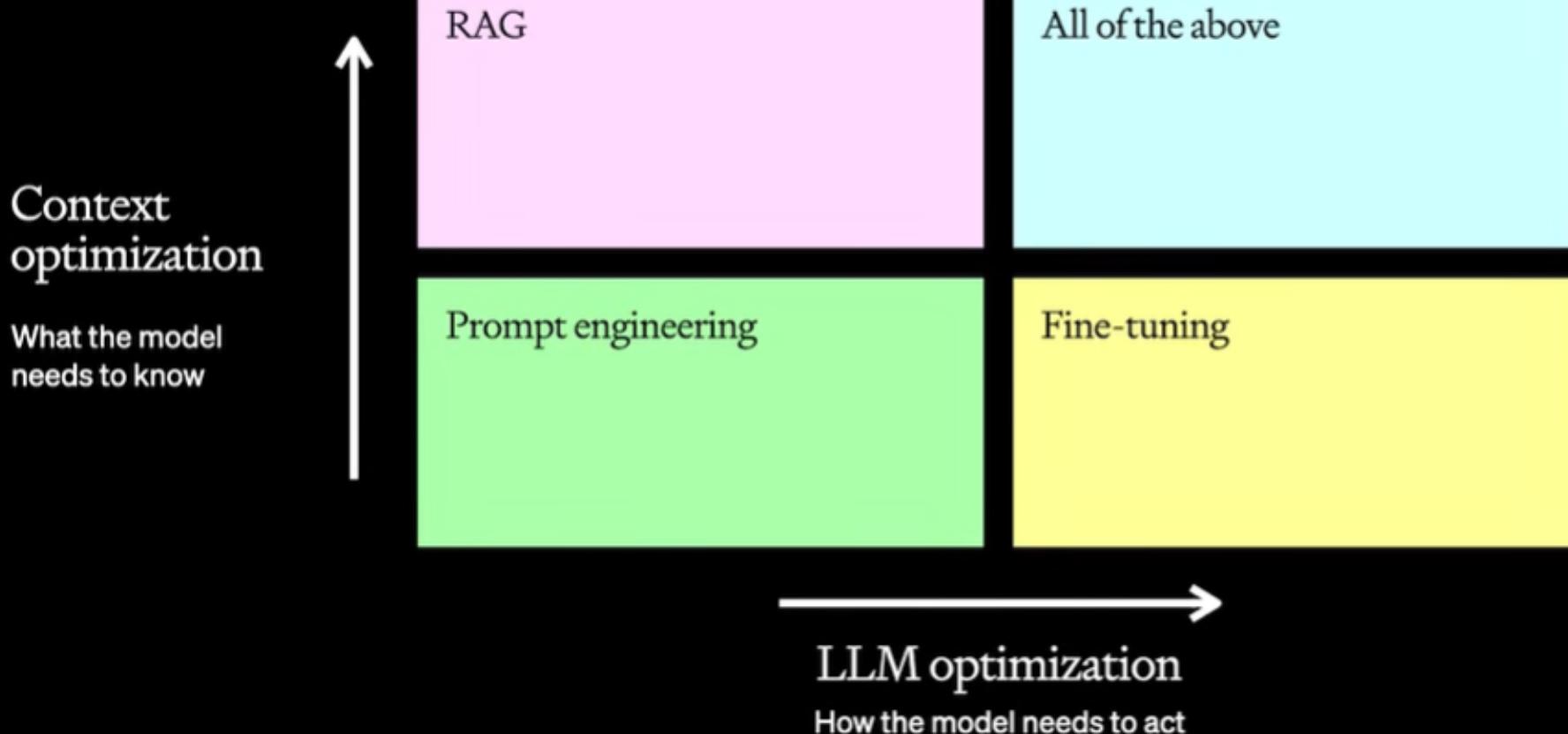
Demos



Gen AI journey

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

The optimization flow



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

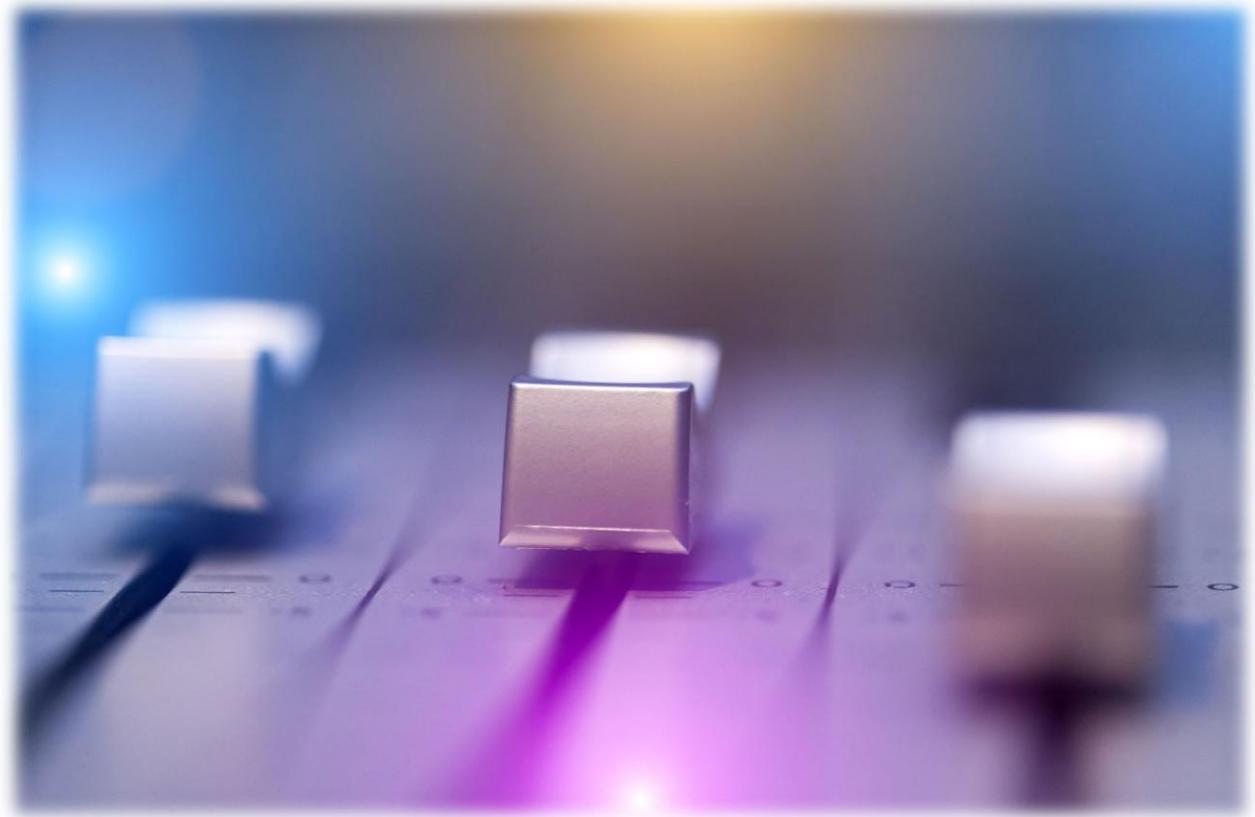
Fine-Tuning techniques

04

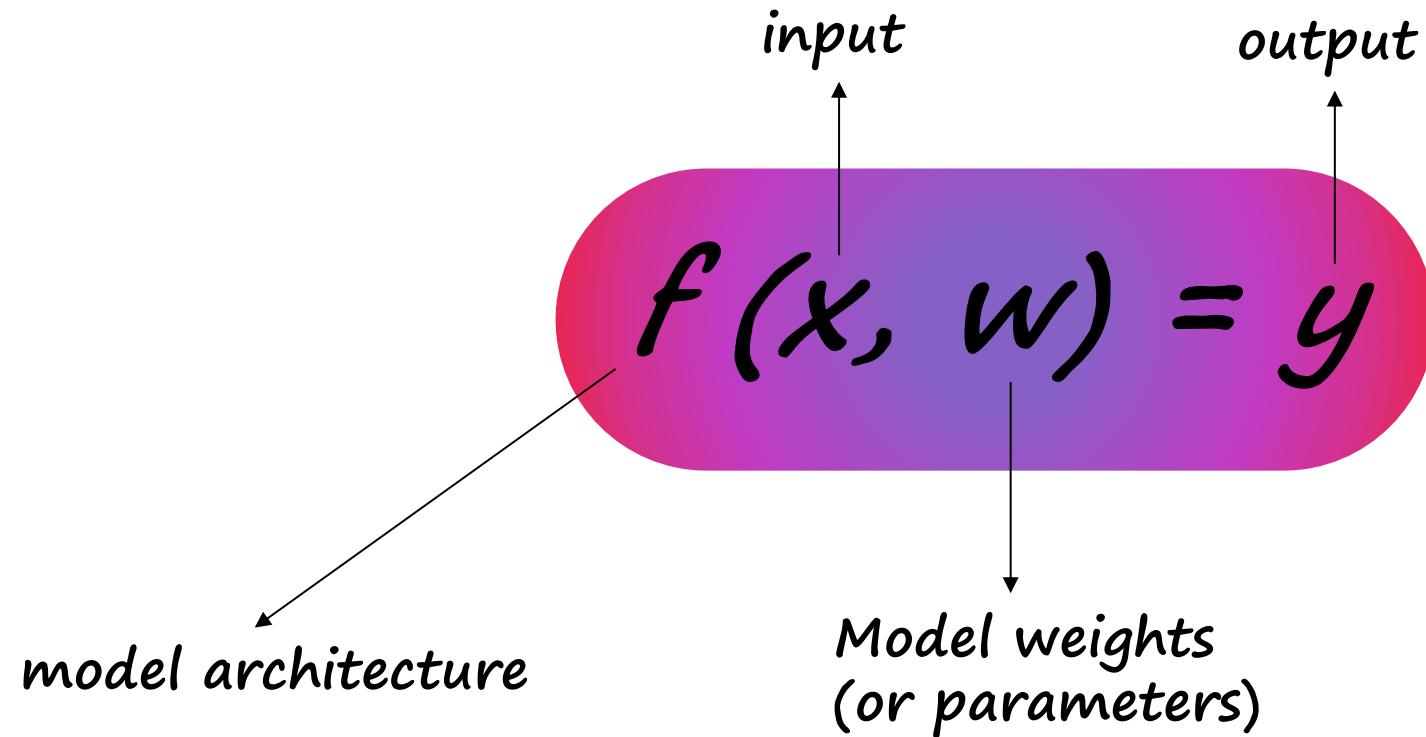
What is available on Azure?

05

Demos

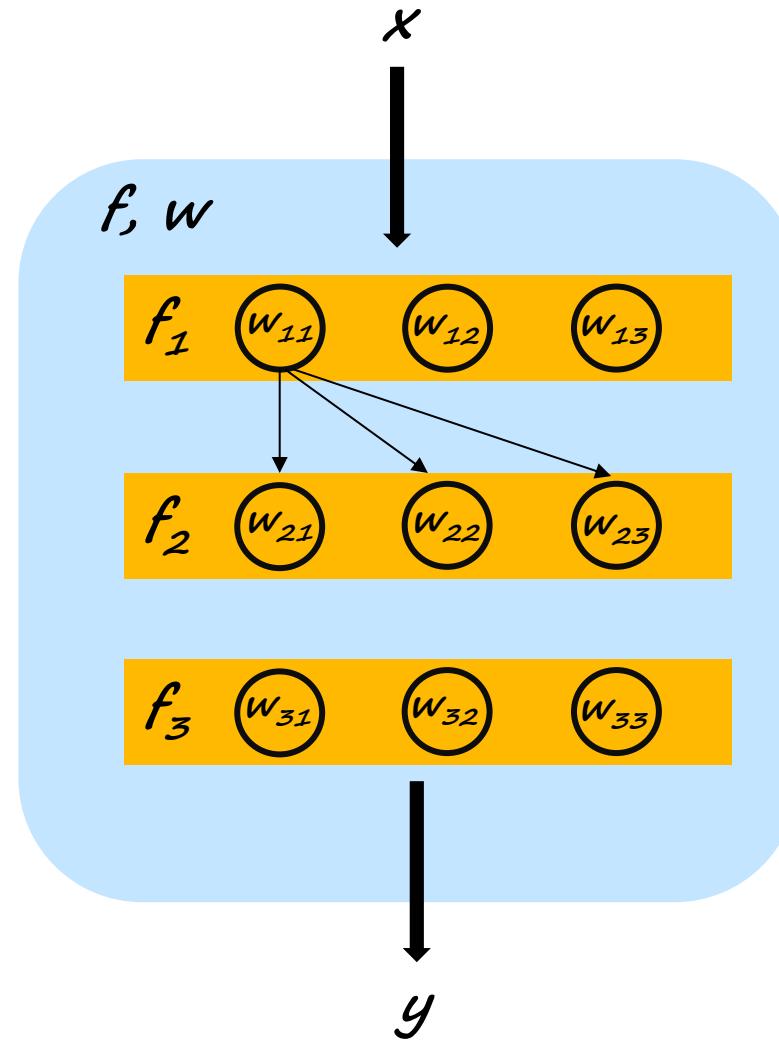


AI model as a function



AI model as a Network

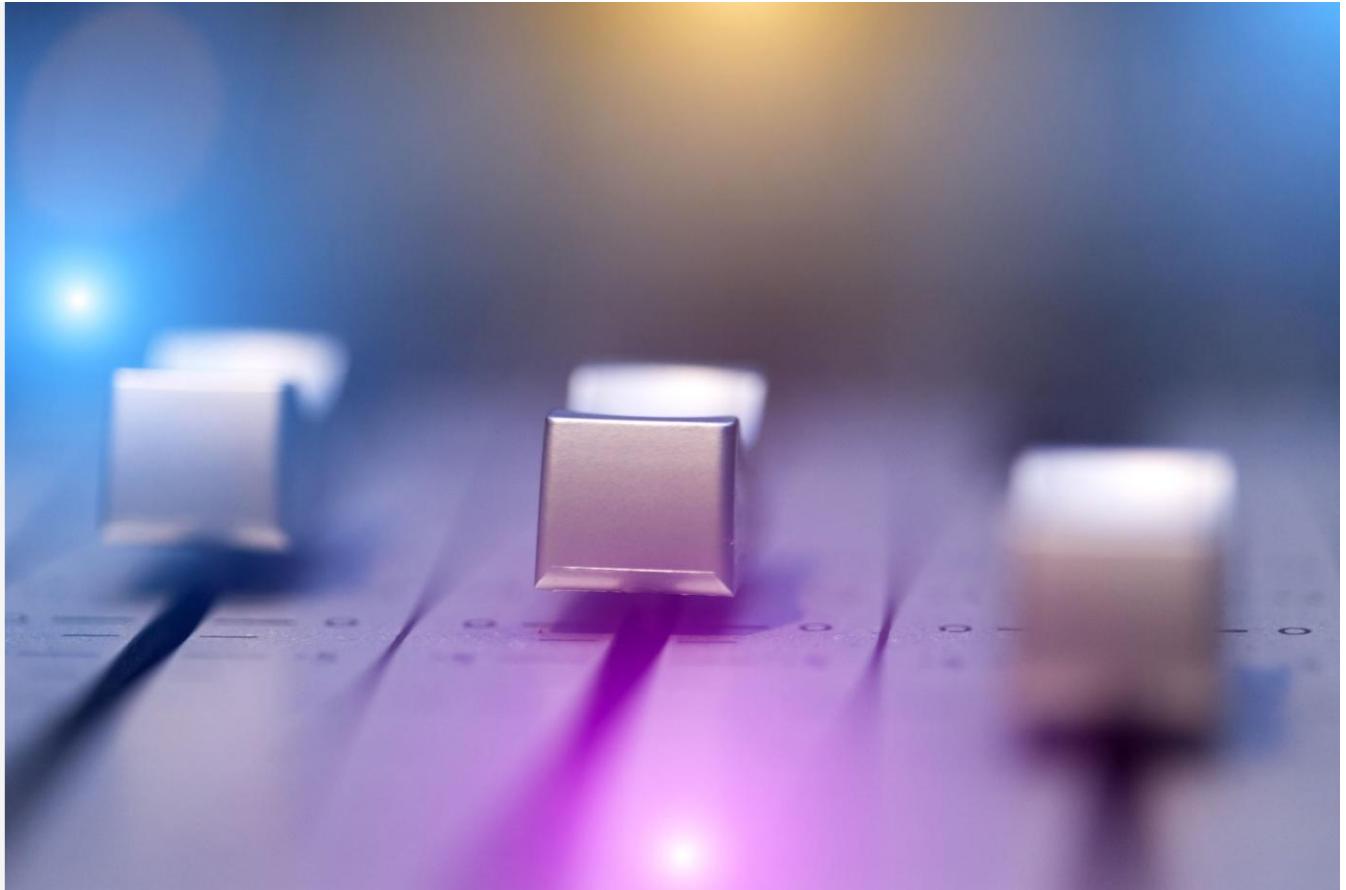
$$f(x, w) = y$$



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 ?

Stage 1:
1) Data preparation & sampling
2) Attention mechanism
3) LLM architecture

Building an LLM

Implements the data sampling and develop the foundational mechanism

4) pre-training

Foundation model

Pretrains the LLM on unlabeled data to obtain a foundation model for further fine-tuning

8) fine-tuning

9) fine-tuning

Classifier

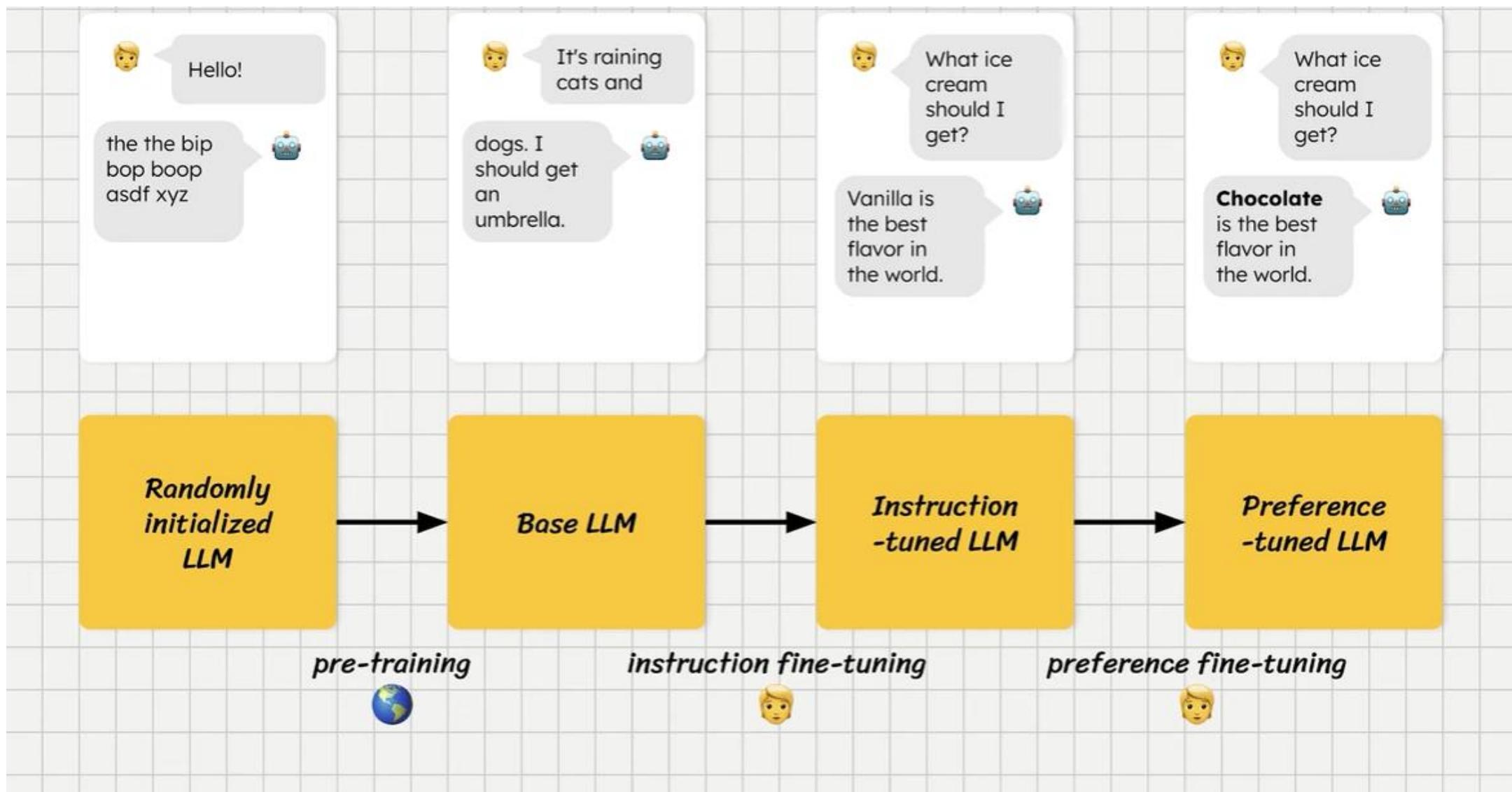
Translator

*Fine-tunes the pretrained LLM to create a classification / translator model or a personal chat model
(these are examples)*

Personal assistant

Instruction
fine-tuning

LLM training the different steps



[source](#)

Preferences on ChatGPT

You're giving feedback on a new version of ChatGPT.

Which response do you prefer? Responses may take a moment to load.



Response 1

Here are **50 more open-ended philosophical questions**, expanding on topics related to art, culture, new geniuses, chess prodigies, and the creation of masterpieces:

1. Will AI ever create art that is truly comparable to human masterpieces?

- **Optimistic Response:** AI will evolve to produce art that evokes deep emotional responses and resonates with human experience, becoming an integral part of artistic expression.
- **Pessimistic Response:** AI-created art will always lack the nuance, originality, and emotional depth that human artists bring to their work.



Response 2

Certainly! Here are **20 more open-ended philosophical questions** focusing on the creation of art masterpieces, the emergence of new geniuses, chess prodigies, and culture in general:

1. Can artificial intelligence create art that rivals human masterpieces?

- **Optimistic Response:** AI can produce innovative and emotionally resonant art, expanding the boundaries of creativity.
- **Pessimistic Response:** AI-generated art lacks the depth and soul of human-created masterpieces, remaining mere simulations.

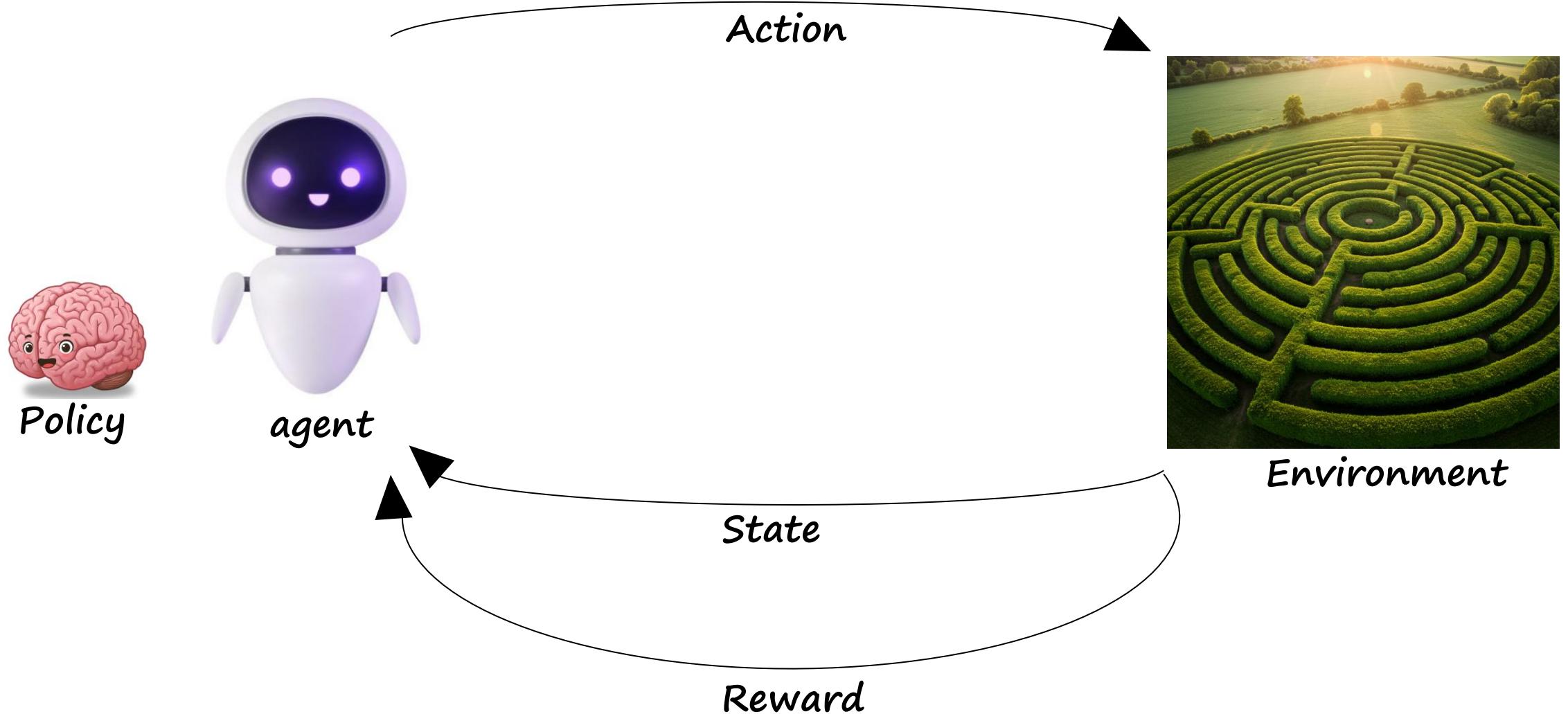


2. Is the concept of 'genius' a social construct, or does

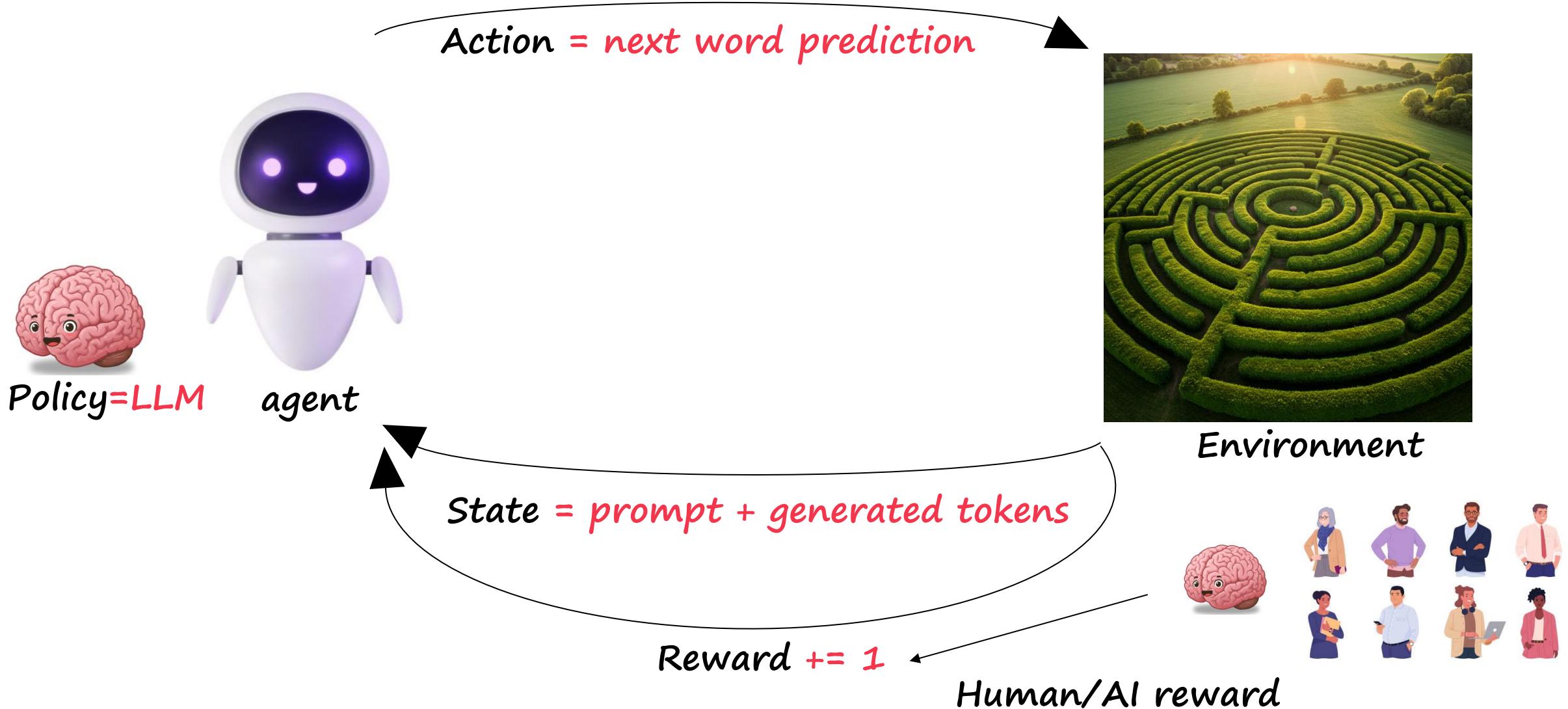
I prefer this response

I prefer this response

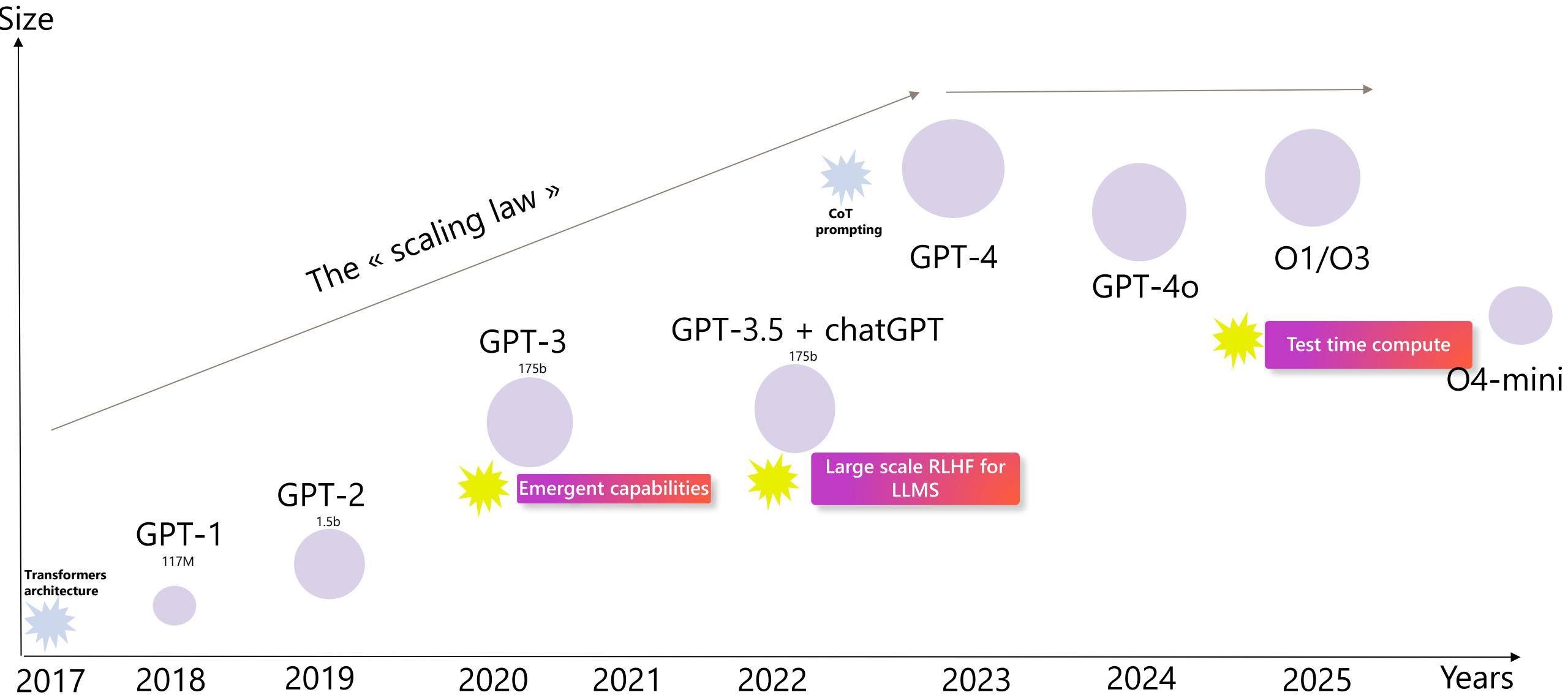
Reinforcement Learning in a Nutshell



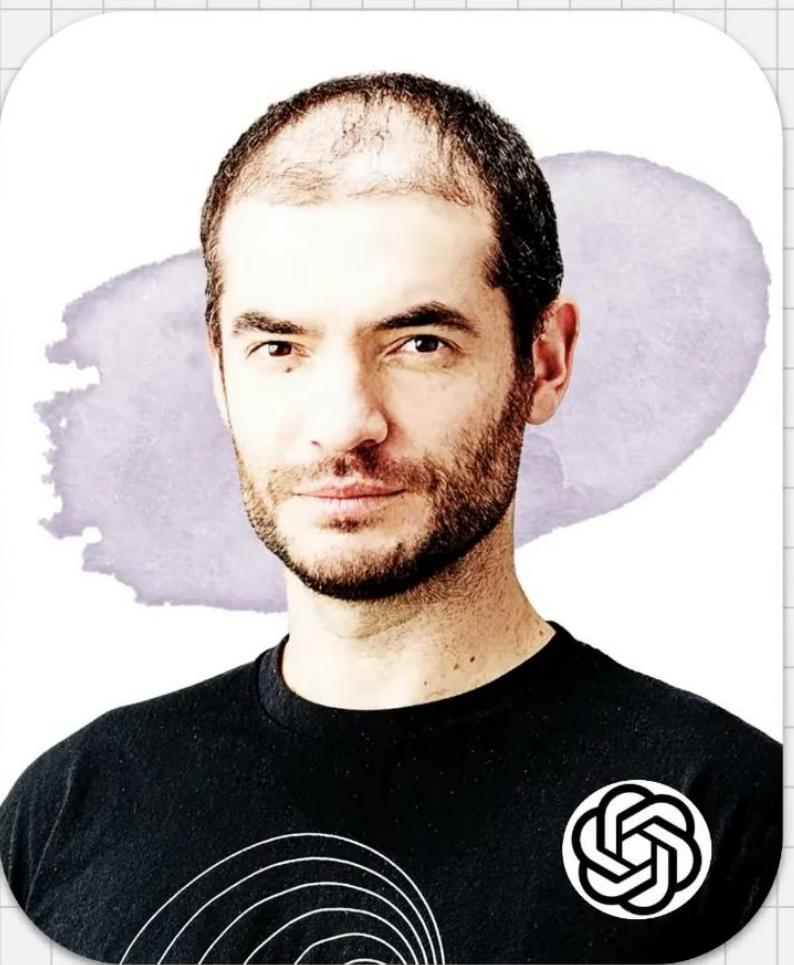
Reinforcement Learning for LLMs



Evolution of LLMs flagships models



Is it the end of the scaling law?

A portrait of Illya Sutskever, a man with short dark hair and a beard, wearing a black t-shirt with a white circular logo. He is set against a background of a white circle with a purple watercolor wash behind it, all resting on a grid pattern.

"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*

Evolution of LLMs flagships models

Size

↑

2017

2018

2019

2020

2021

2022

2023

2024

2025

Years

The scaling law



CoT
prompting

GPT-4

GPT-4o

O1 / O3

CoT + RLHF:
From training focus to
inference focus

O4-mini

GPT-1

Transformers
architecture



GPT-2

1.5b

GPT-3

175b

GPT-3.5 + chatGPT

175b

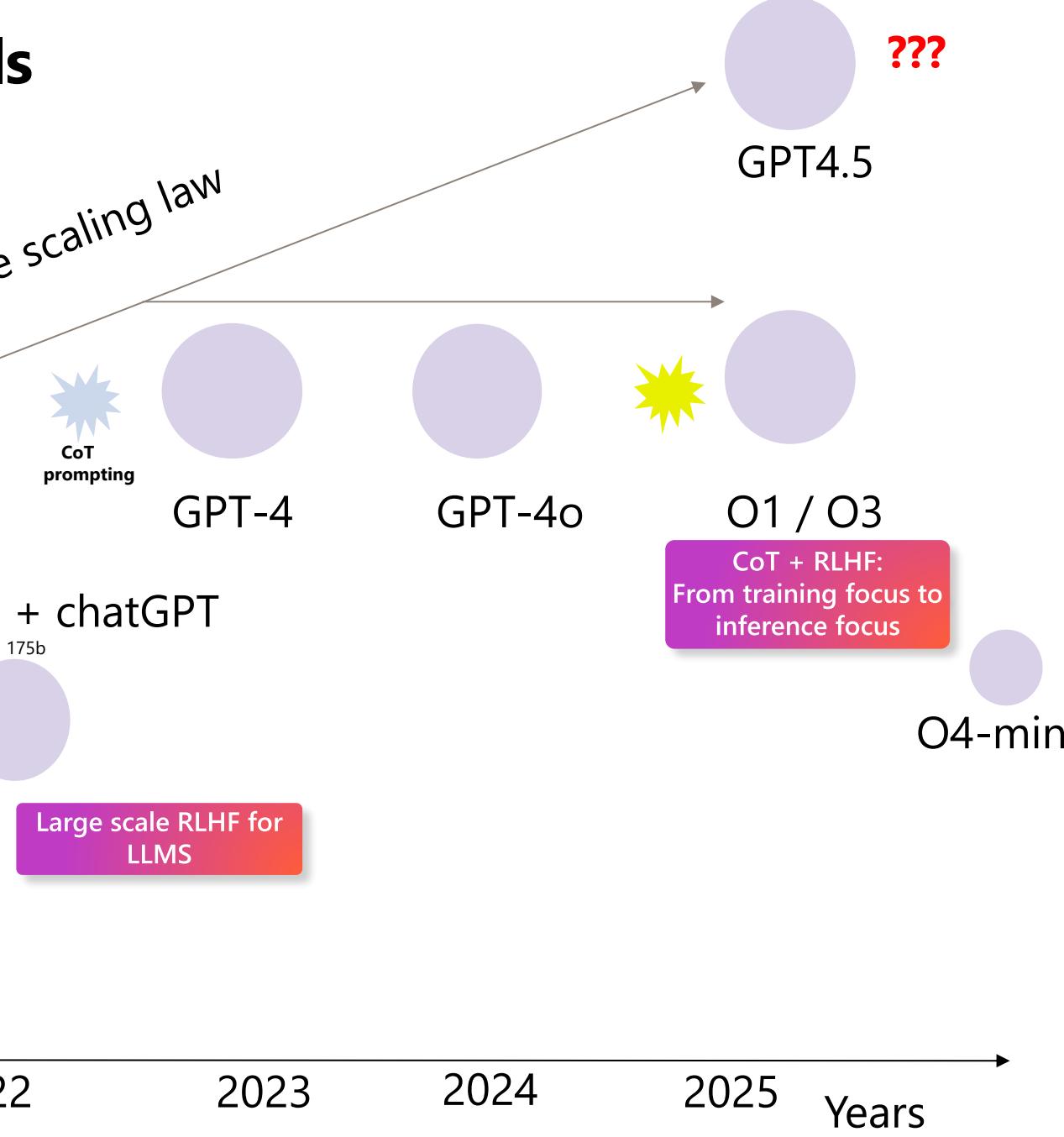


Emergent capabilities

Large scale RLHF for
LLMS

GPT4.5

???



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

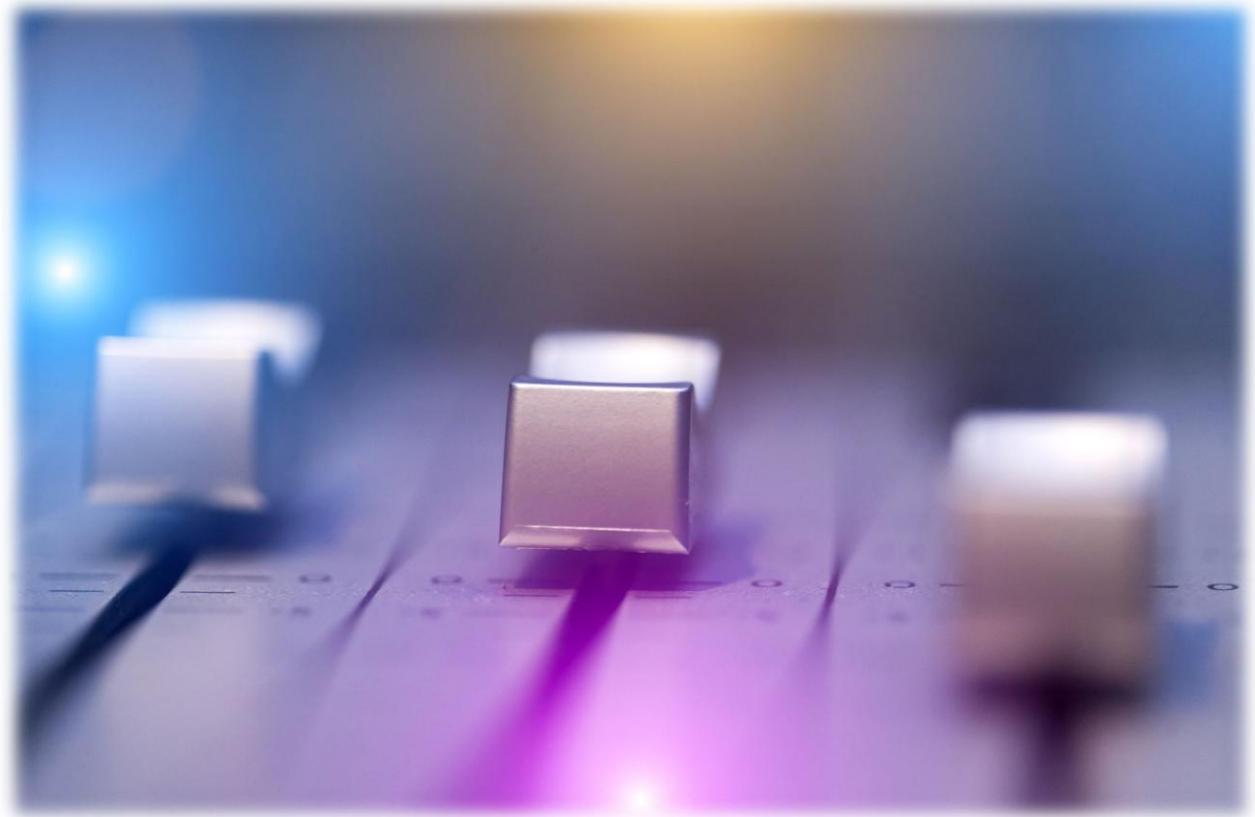
Fine-Tuning techniques

04

What is available on Azure?

05

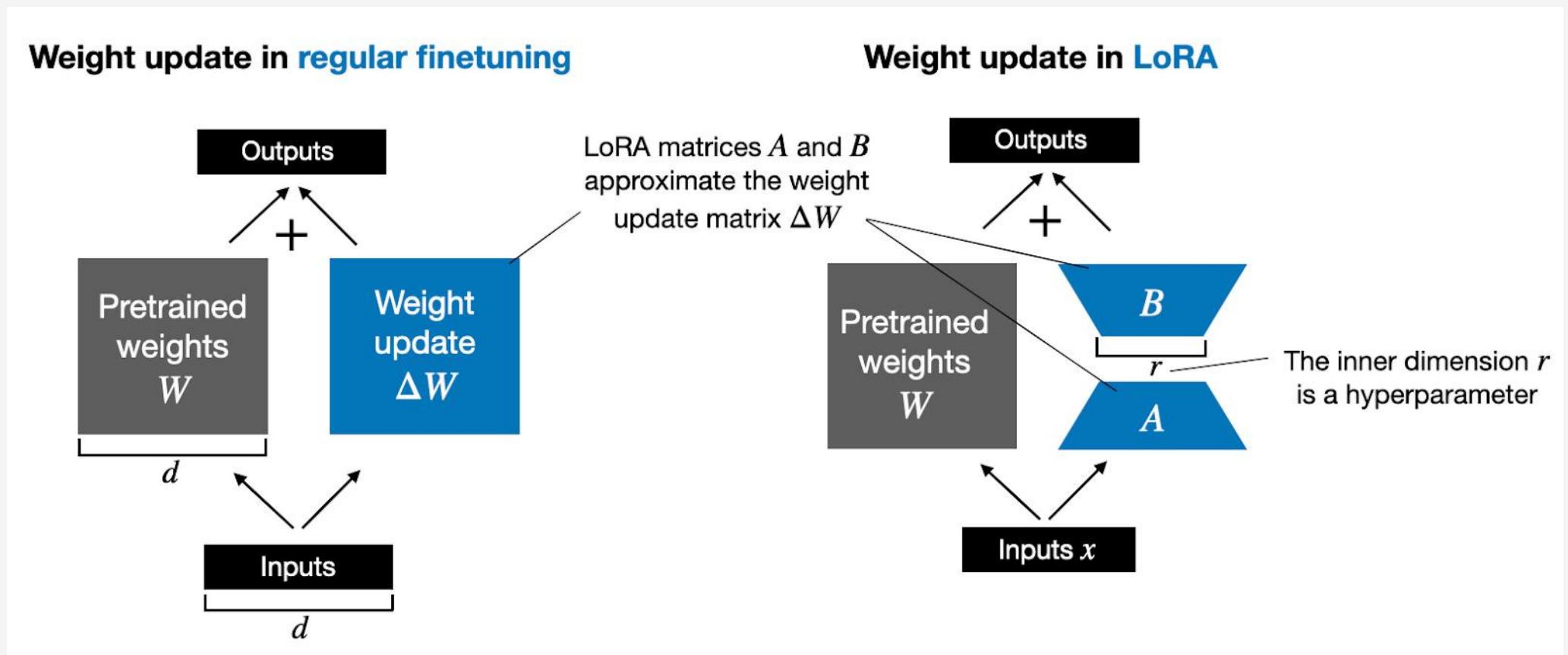
Demos



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)



LoRA Fine-Tuning: Key Trade-offs



Advantages

Parameter-Efficient:

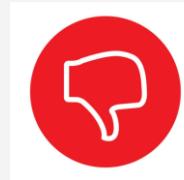
Drastically reduces trainable parameters (often <1%). Less memory, faster training, and lower compute costs. Enables fine-tuning on consumer hardware.

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

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



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

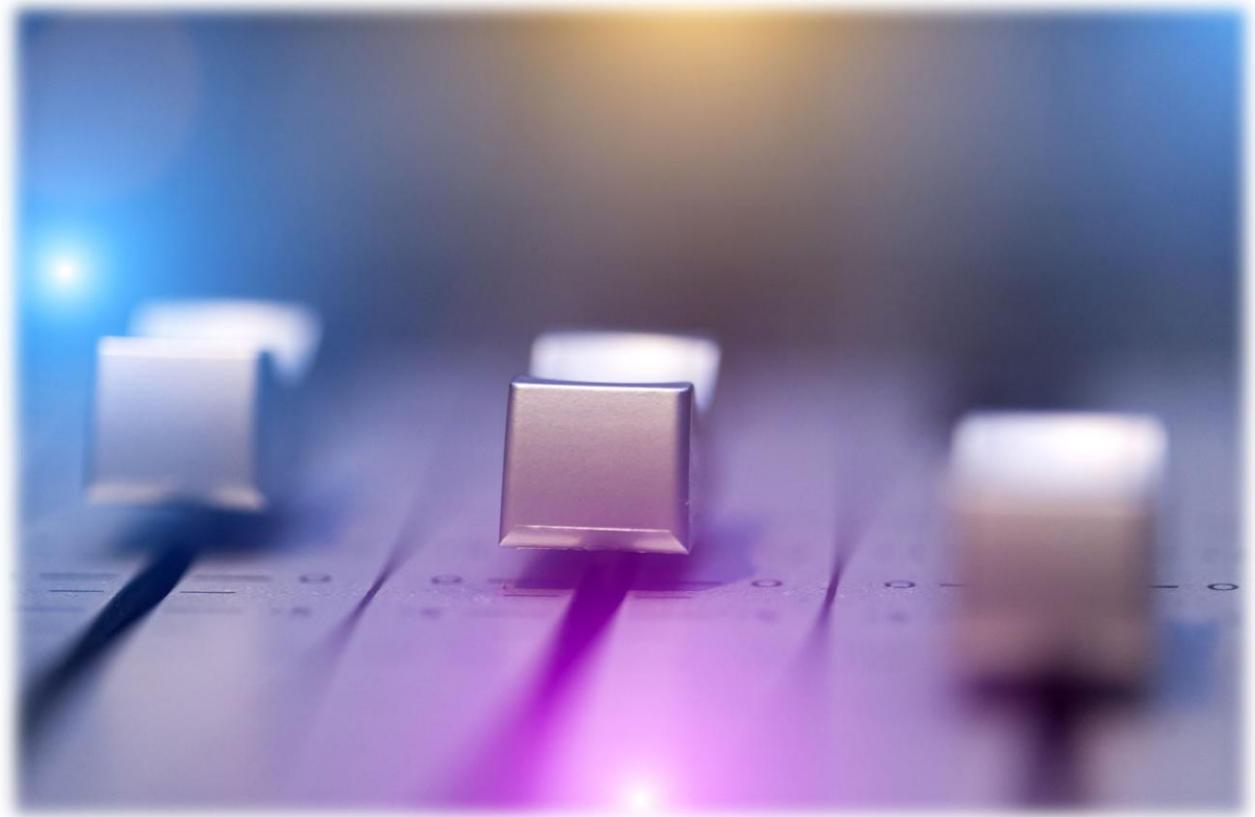
Fine-Tuning techniques

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

Complexity & more knowledge required →

What fine-tuning methods do we offer?

Supervised fine-tuning

- GPT-3.5-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

```
{  
  "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"  
    }  
  ]  
}
```

User question

Image data
URL or base64

True Answer

JSONL data file format for training and validation

Fine-tuning options

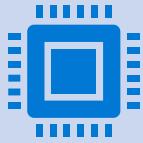
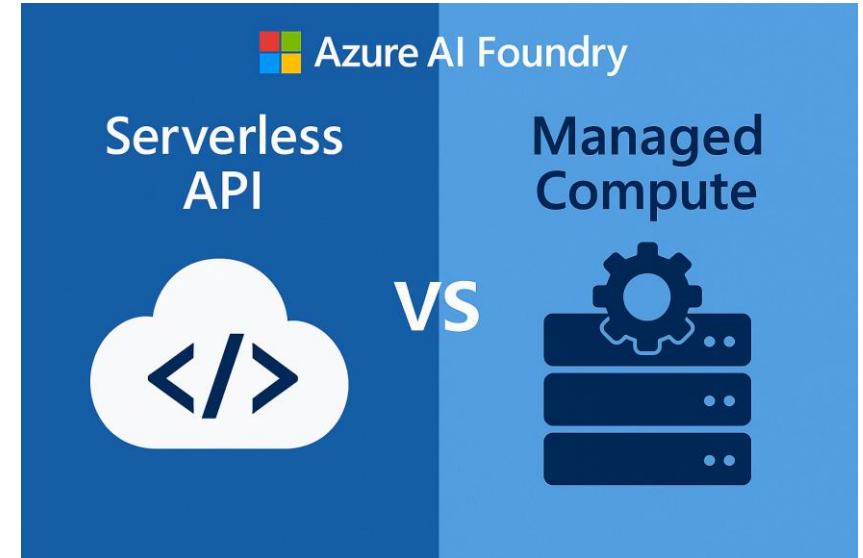
Azure's managed fine-tuning services streamline the customization process by taking care of the underlying infrastructure and optimization techniques. The compute is serverless, with charges based on hourly consumption. The managed option simplifies the process, allowing users to focus on their use case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

Find the right model to build your custom AI solution

Help

Model leaderboards

See what models are performing best in different criteria.

Browse leaderboards

Quality

- 1 o3
- 2 o4-mini
- 3 o1

Cost

- 1 Minstral-3B
- 2 Mistral-Nemo
- 3 mistral-small-2503

Throughput

- 1 Llama-3.2-1B-Instruct
- 2 o3-mini
- 3 o1-mini

Collections

Industry

Capabilities

Billing types

Inference tasks

Fine-tuning tasks (9)

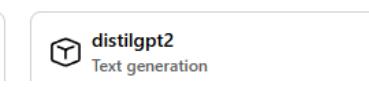
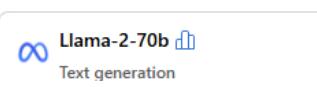
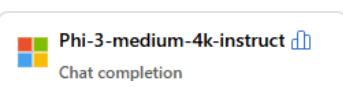
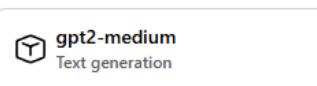
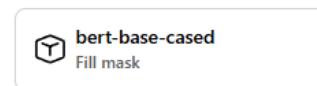
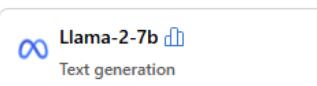
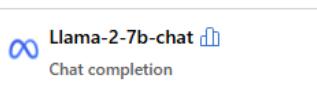
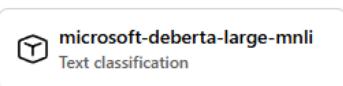
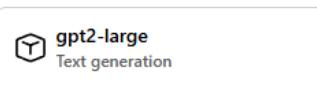
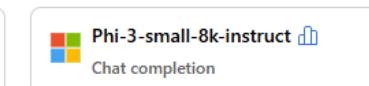
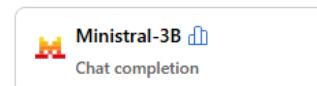
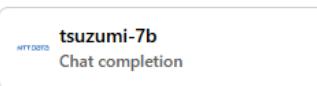
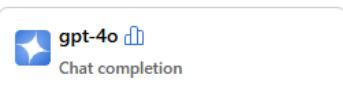
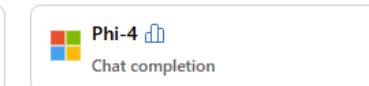
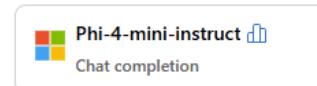
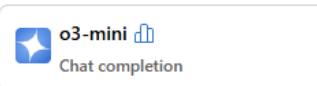
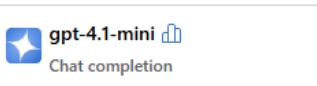
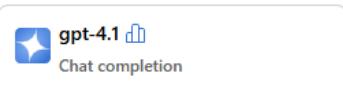
Licenses

Clear all

Compare models

Search

Models 73



Serverless / Managed compute

← Phi-4-mini-instruct PREVIEW

Deploy Fine-tune

Details Benchmarks Existing deployments License

Phi-4-mini-instruct is a lightweight open model that belongs to the Phi-4 model family and supports preference optimization to support precise instruction following.

Phi-4-mini-instruct is a dense decoder-only Transformer with multi-head attention, and shared embedding. It is designed to be fine-tuned on an offline dataset with a June 2024 data cutoff. The model is intended for broad multilingual support, including English, French, German, Spanish, Italian, Japanese, Korean, Norwegian, Polish, and more.

See more

Data, media and languages

Property	Description
Supported data types	Input: text
Supported languages	ar, zh,

Transparency

Release Notes

This release of Phi-4-Mini is based on valuable user feedback from the Phi-3 series. The Phi-4-Mini model employed new architecture for efficiency, larger vocabulary for multilingual multimodal support, and better post-training techniques were used for instruction following, function calling, as well as additional data leading to substantial gains on key capabilities. It is anticipated that most use cases will benefit from this release, but users are encouraged to test in their particular AI applications. The enthusiastic support for the Phi-4 series is greatly appreciated. Feedback on Phi-4-Mini is welcomed and crucial to the model's evolution and improvement.

Fine-tuning options PREVIEW

Choose a serving method for fine-tuning this model:

Serverless API 

This option lets you fine-tune the model on a fully-managed service that does not require you to host or manage infrastructure.

Managed compute 

This option lets you fine-tune the model on Azure infrastructure that you host and manage.

Cancel

Quick facts

Microsoft

Phi-4-mini-instruct

chat-completion

Last trained
June 2024

Pricing
See pricing

Benchmarks

0.44 Quality index AI quality	N/A USD per 1M tokens Estimated cost
----------------------------------	---

Model ID

Reference this model ID when deploying the model in code

azureml://registries/azureml/models/Phi-4-mini-instruct/versions/1

Managed compute

Basic settings ✓

2 Compute

3 Training data

4 Validation data optional

5 Task parameters optional

6 Review

Compute
Select and configure the compute resource for executing your training job.

Select Virtual Machine size * ⓘ

Name ↑	Category	Available q... ⓘ	Cost ⓘ
<input type="radio"/> Standard_NC24ads_A100_v4 24 cores, 220GB RAM, 64GB storage	--	60 cores	\$3.67/hr
<input type="radio"/> Standard_NC48ads_A100_v4 48 cores, 440GB RAM, 128GB storage	--	60 cores	\$7.35/hr
You do not have enough quota for the following VM sizes. Click here to view and request quota. ↗			
<input type="radio"/> Standard_NC96ads_A100_v4 96 cores, 880GB RAM, 256GB storage	--	60 cores	\$14.69/hr
<input type="radio"/> Standard_ND96amsr_A100_v4 96 cores, 1800GB RAM, 2900GB storage	--	0 cores	\$32.77/hr

Back

Next

Submit

Cancel

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

Pricing

[Azure OpenAI Pricing page](#)

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

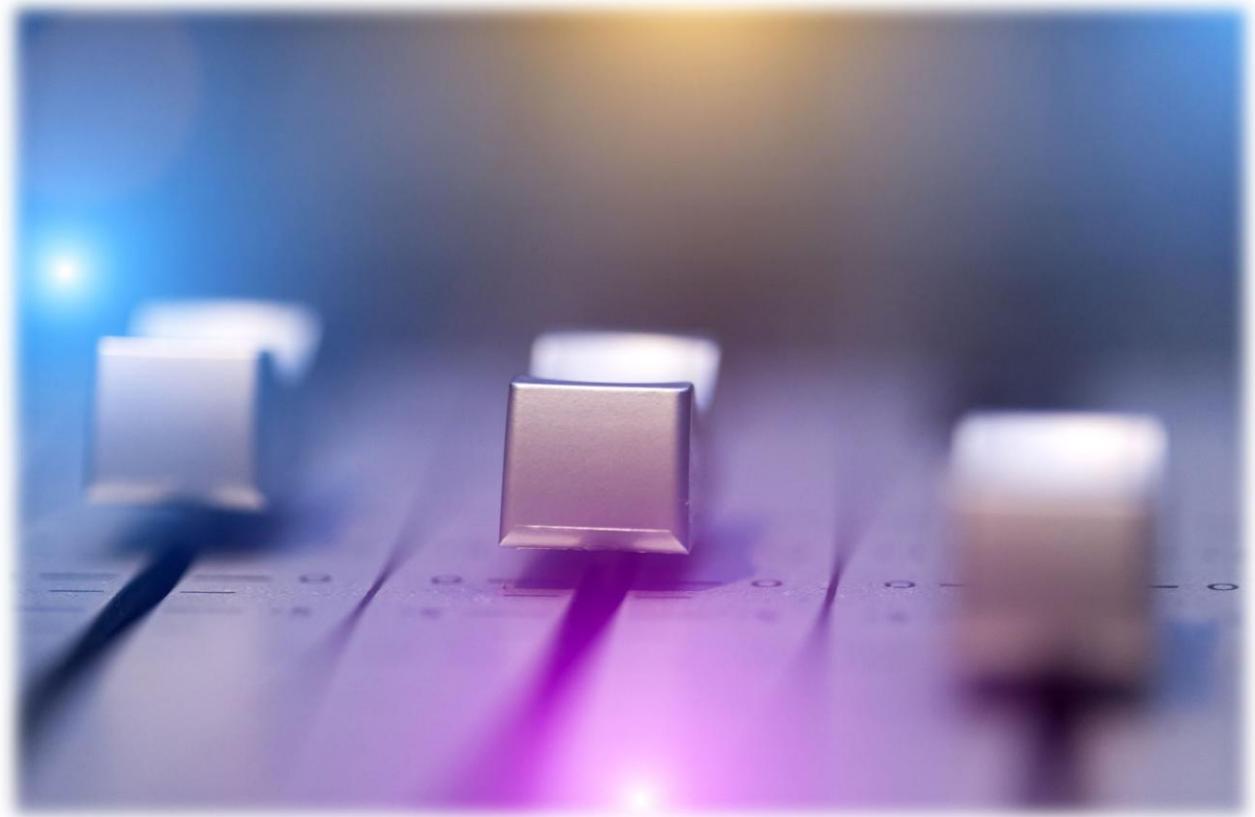
Fine-Tuning techniques

04

What is available on Azure?

05

Demos



Demo 1

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

The image shows a close-up of a person's hand pointing their index finger towards a computer monitor. The monitor displays a dark-themed Python script. The script appears to be a Blender operator for mirroring objects. It includes logic for selecting objects, setting mirror modifiers, and handling errors if no objects are selected. The code uses standard Python syntax with some specific Blender API calls.

```
mirror_mod = modifier_obj.modifiers.new("Mirror", type='MIRROR')
# Set mirror object to mirror
mirror_mod.mirror_object = mirror_object
if operation == "MIRROR_X":
    mirror_mod.use_x = True
    mirror_mod.use_y = False
    mirror_mod.use_z = False
elif operation == "MIRROR_Y":
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
elif operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

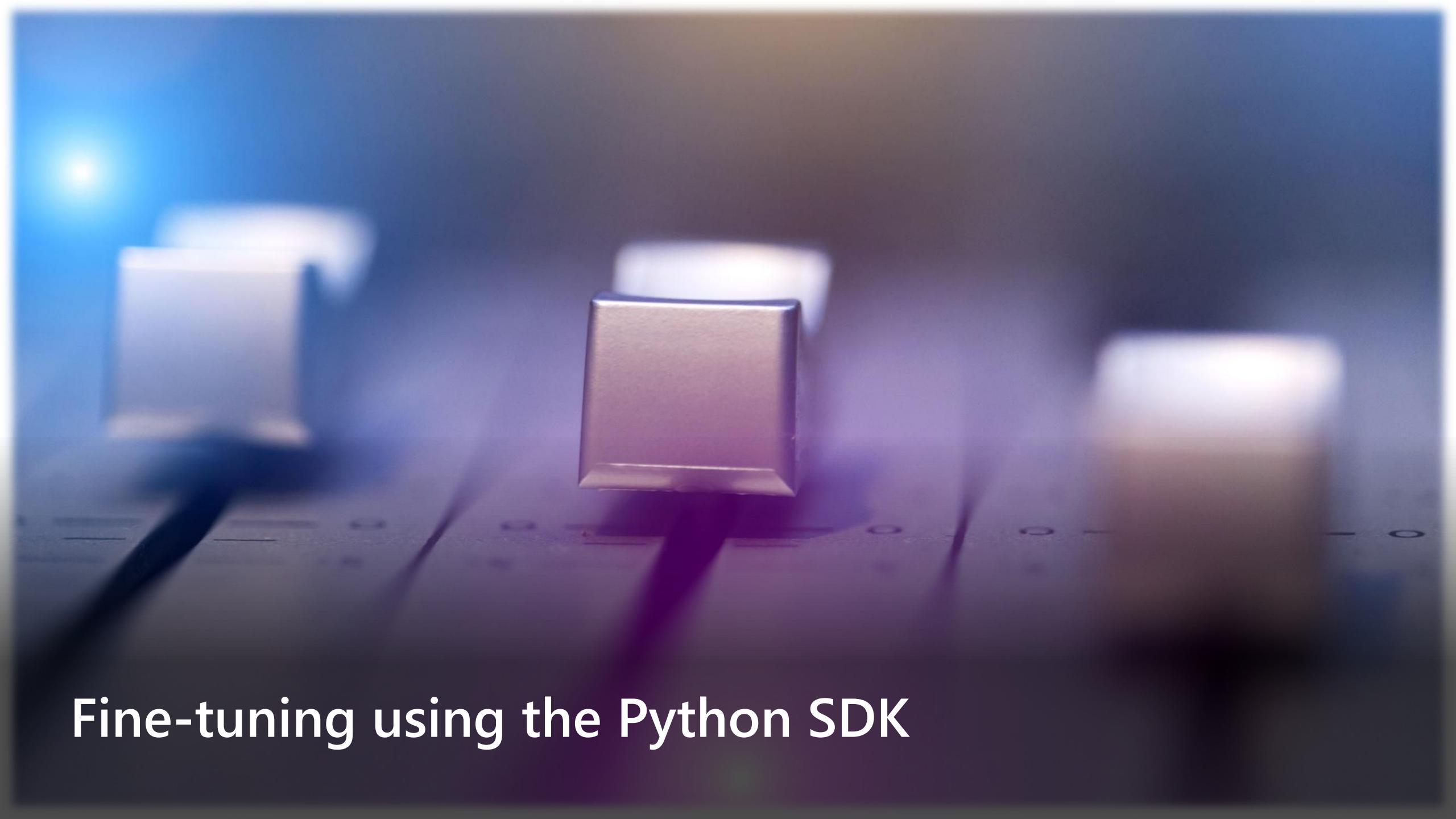
#selection at the end -add
if ob.select==1:
    mirror_ob.select=1
    context.scene.objects.active = eval("Selected" + str(modifier))
    mirror_ob.select = 0
    bpy.context.selected_objects.append(data.objects[one.name])
    data.objects[one.name].select = 1
    print("please select exactly one object")
else:
    print("please select exactly one object")

----- OPERATOR CLASSES -----

@operator
def X_mirror_to_the_selected(self, context):
    # X mirror to the selected object.mirror_mirr
    mirror_X = context.scene.objects.active
    if context:
        if context.active_object is not None:
```

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

Confusion Matrix of classification with gpt-4o baseline model



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.
    })
```

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
)
```

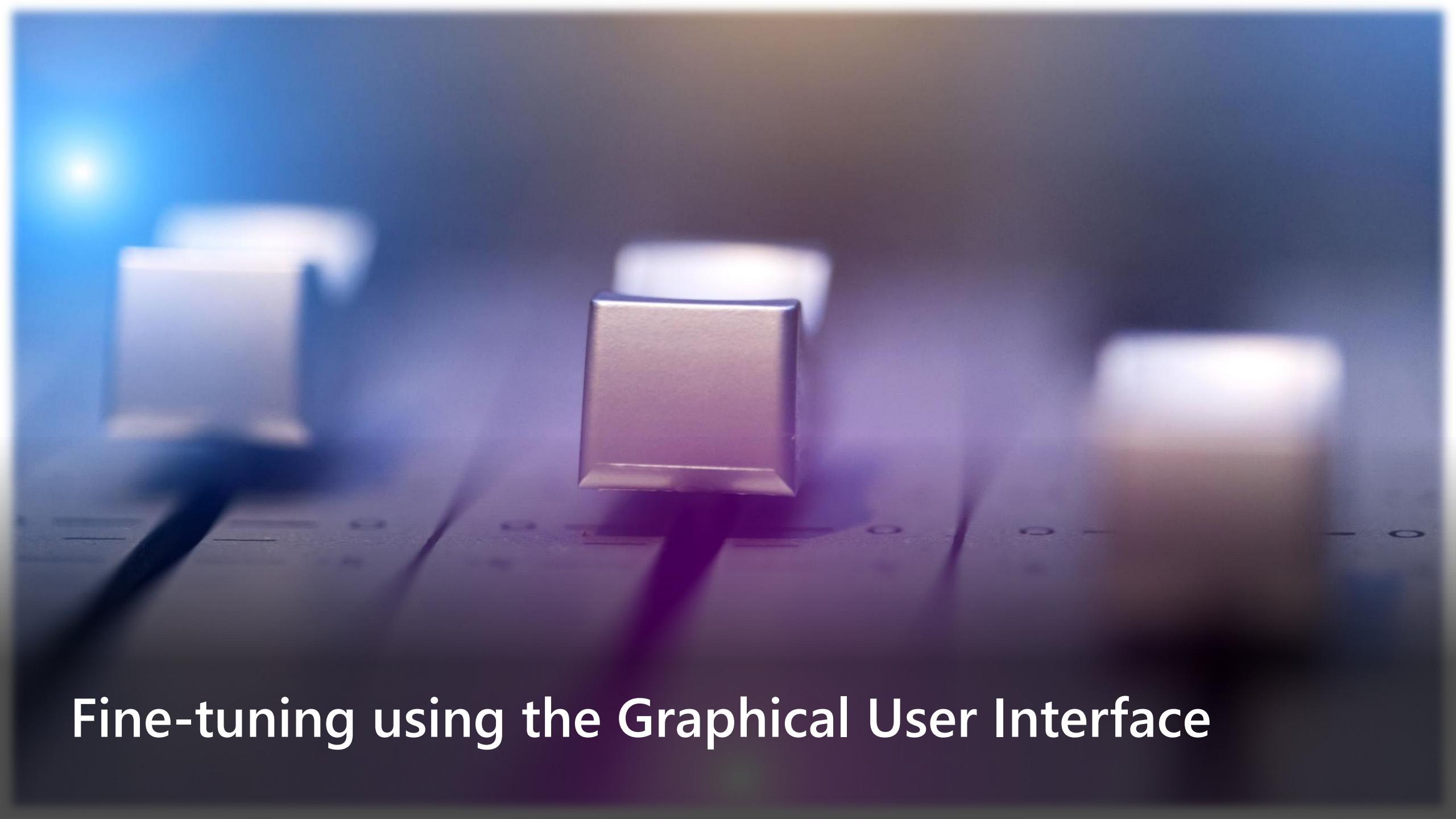
Fine-tuned model: callcenter

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

Confusion Matrix of classification with gpt-4o fine-tuned model



Fine-tuning using the Graphical User Interface

From Azure AI Foundry

The screenshot shows the Azure AI Foundry interface with a red box highlighting the 'Fine-tuning' section in the left sidebar. A modal window titled 'Select a model to fine-tune' is open, displaying a list of 42 models under the 'Base models' tab. The 'Phi-4-mini-instruct' model is selected and detailed in the right panel.

Select a model to fine-tune

Choose a model to fine-tune by filtering through our catalog. Note that specific models may only be available for fine-tuning in specific regions. [Learn more about regional constraints for fine-tuning](#)

Note: Fine-tuning of models other than Azure OpenAI Service models is in preview.

Models 42 Collections Fine-tuning tasks Customization method Show description

Phi-4-mini-instruct

Task: Chat completion

Phi-4-mini-instruct is a lightweight open model built upon synthetic data and filtered publicly available websites - with a focus on high-quality, reasoning dense data. The model belongs to the Phi-4 model family and supports 128K token context length. The model underwent an enhancement process, incorporating both supervised fine-tuning and direct preference optimization to support precise instruction adherence and robust safety measures.

Phi-4-mini-instruct is a dense decoder-only Transformer model with 3.8B parameters, offering key improvements over Phi-3.5-Mini, including a 200K vocabulary, grouped-query attention, and shared embedding. It is designed for chat-completion prompts, generating text based on user input, with a context length of 128K tokens. This static model was trained on an offline dataset with a June 2024 data cutoff. It supports many languages, including Arabic, Chinese, Czech, Danish, Dutch, English, Finnish, French, German, Hebrew, Hungarian, Italian, Japanese, Korean, Norwegian, Polish, Portuguese, Russian, Spanish, Swedish, Thai, Turkish, Ukrainian.

The model is intended for broad multilingual commercial and research use. The model provides uses for general purpose AI systems and applications which require 1) memory/compute constrained environments; 2) latency bound scenarios; 3) strong

Next Cancel

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

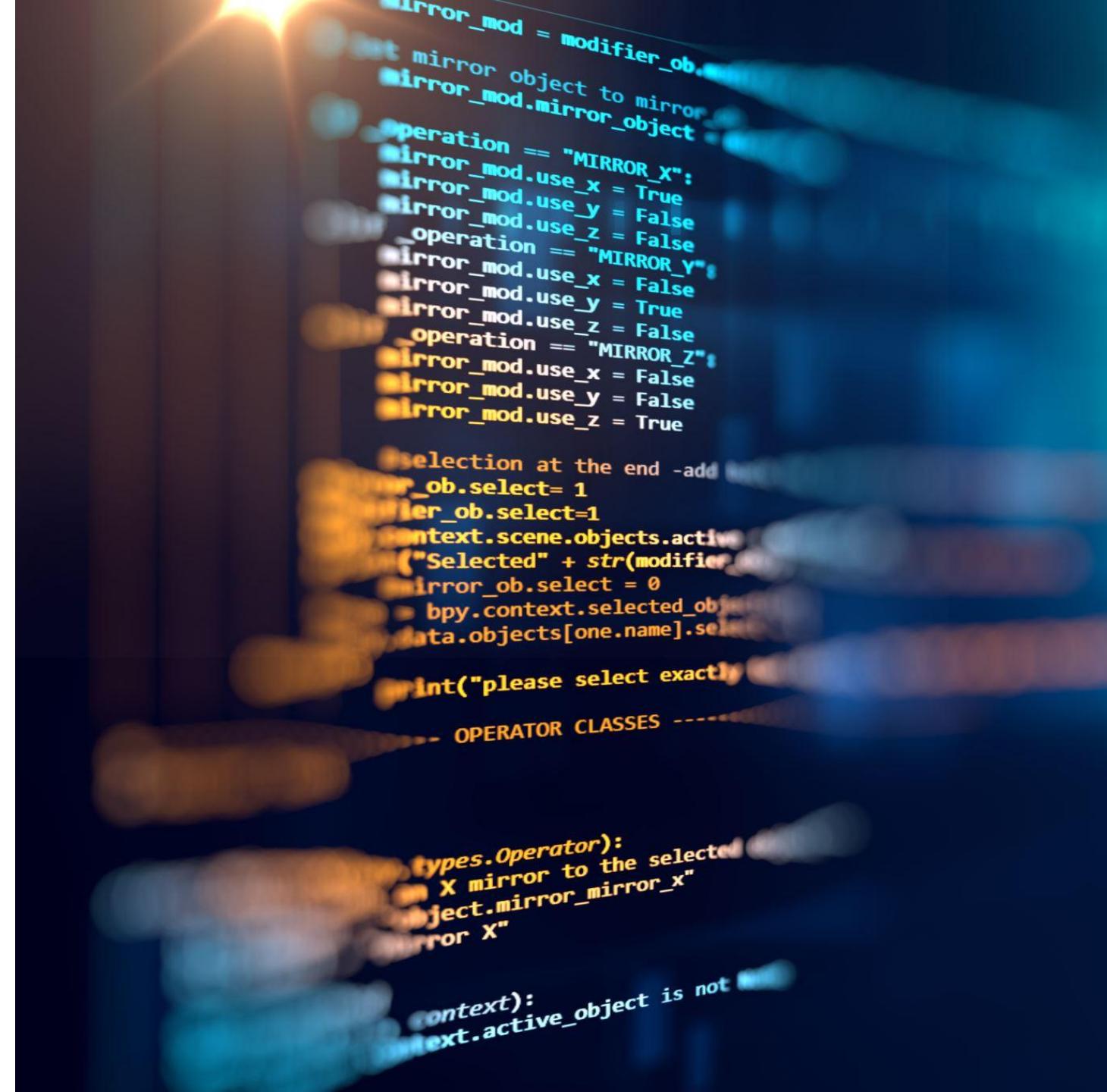
The screenshot shows the Azure AI Foundry | Azure OpenAI Service interface. The left sidebar has a dark theme with white icons and text. It includes sections for Home, Get started, Model catalog, Playgrounds, Chat, Assistants (PREVIEW), Audio (PREVIEW), Images, Completions, Tools (with a dropdown menu for Fine-tuning, Azure OpenAI, Service Evaluation, Stored completions, PREVIEW, Batch jobs, Observability, Shared resources, Deployments, Quota, Safety + security, Data files, and Vector stores PREVIEW), and a preview section for Continual fine-tuning.

The main content area is titled "Fine-tune with your own data" and contains a sub-header: "Optimize models for specific tasks with smaller, task specific datasets to improve its performance and accuracy. Because this method tends to require fewer examples in the prompts, generally less text is sent—and tokens processed—per call." Below this is a table with the following data:

Model name	Base model	Status	Created on	Fine tune retirement time	Inference retirement time
gpt-4o-2024-08-06.ft-c40e7b411fae4e75a63042e6a4cbcf6d-callcenter	gpt-4o-2024-08-06	Completed	6 févr. 2025 11:21	20 août 2025 02:00	20 août 2025 02:00
gpt-4o-mini-2024-07-18.ft-82d7922a32084532a1d98f2d11e56c06-callcenterclassification	gpt-4o-mini-2024-07-18	Completed	23 janv. 2025 17:53	20 juil. 2025 02:00	20 juil. 2025 02:00
gpt-4o-2024-08-06.ft-0c5a44718cc44e519d442a8d45f6b341-metaldefects-gpt4o	gpt-4o-2024-08-06	Completed	22 janv. 2025 16:24	20 août 2025 02:00	20 août 2025 02:00
gpt-4o-2024-08-06.ft-f3e4a704ad1342b0a4ce0870f0fb80d3-vqa-gpt4o	gpt-4o-2024-08-06	Completed	20 janv. 2025 09:03	20 août 2025 02:00	20 août 2025 02:00

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.

Supervised Fine-tuning for Chart Analysis

Hugging Face Search models, datasets, users... Models Datasets Spaces Posts Docs Enterprise Pricing Log In Sign Up

Datasets: HuggingFaceM4/ChartQA like 32 Follow HuggingFaceM4 420

Modalities: Image Text Formats: parquet Size: 10K - 100K Libraries: Datasets Dask Croissant +1 License: gpl-3.0

Dataset card Data Studio Files and versions Community 4

Dataset Viewer Auto-converted to Parquet API Embed Data Studio

Split (3)
train • 28.3k rows

Search this dataset

image	query	label	human_or_machine
image · width (px)	string · lengths	list · lengths	class label
184	1.32k	17	262
184	Is the value of Favorable 38 in 2015?	["Yes"]	0 human
184	How many values are below 40 in Unfavorable graph?	["6"]	0 human
184	In which year the value was 51?	["2014"]	0 human
184	What is the sum favourable value in the year 2014 and 2015?	["95"]	0 human
184	To which year the Mexican governments campaign against		

Downloads last month 6,523

Use this dataset

Size of downloaded dataset files: 964 MB

Size of the auto-converted Parquet files: 964 MB Number of rows: 32,719

Models trained or fine-tuned on HuggingFaceM4/...

Ayamohamed/DiaClassModel Image Classification Updated 13 days ago ↓ 4

Spaces using HuggingFaceM4/ChartQA 3

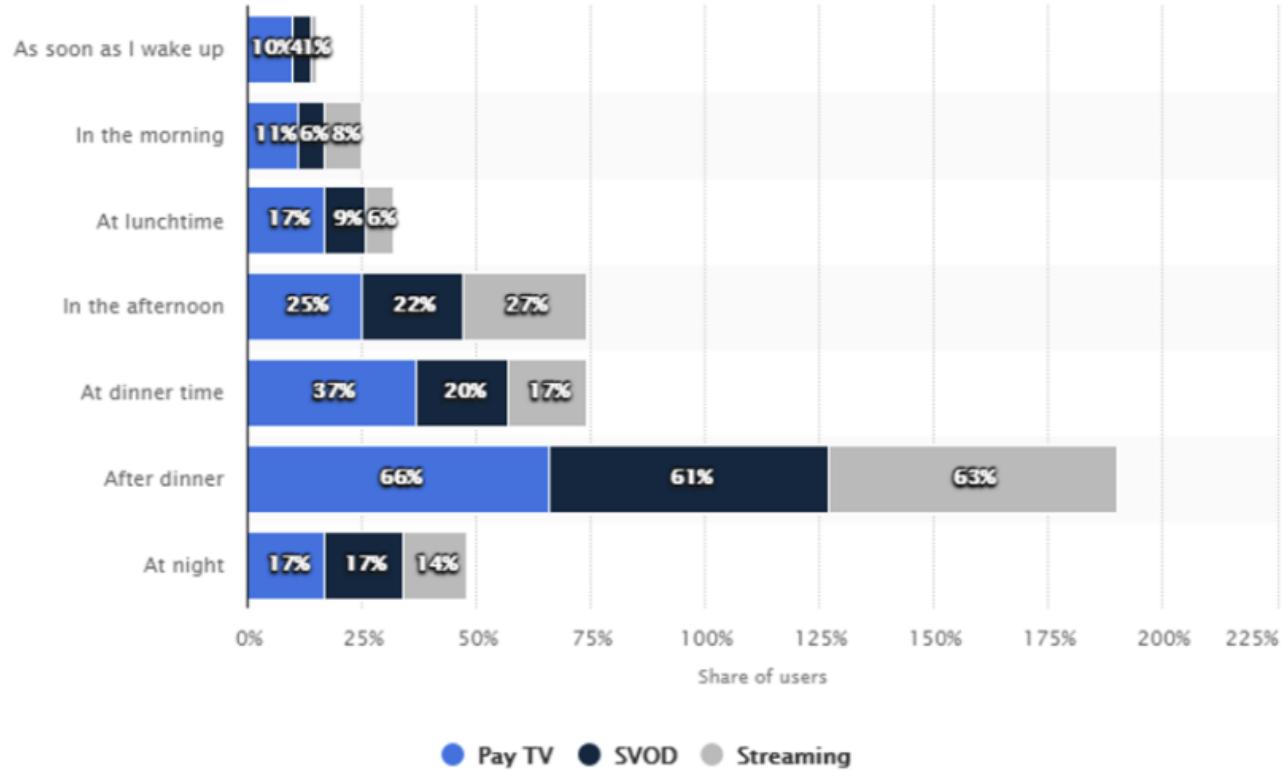
sergiopaniego/Qwen2-VL-7B-trl-sft-ChartQA

sergiopaniego/SmolVLM-trl-sft-ChartQA

JulianAT/Interiorly-Qwen2-VL-7B-InteriorlyQA

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

© Statista 2021

Show source

Additional Information

Demo 3

Florence-2 fine tuning for
object detection

The image shows a close-up of a person's hand pointing their index finger towards a computer monitor. The monitor displays a dark-themed Python script. The hand is illuminated by a warm, orange-yellow light, creating a strong contrast against the cool blue and black tones of the code and the background.

```
mirror_mod = modifier_ob
# Set mirror object to mirror
mirror_mod.mirror_object = None
if operation == "MIRROR_X":
    mirror_mod.use_x = True
    mirror_mod.use_y = False
    mirror_mod.use_z = False
elif operation == "MIRROR_Y":
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
elif operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

#selection at the end -add
if ob.select==1:
    mirror_ob.select=1
    context.scene.objects.active = eval("Selected" + str(modifier))
    mirror_ob.select = 0
    bpy.context.selected_objects.append(data.objects[one.name])
    data.objects[one.name].select = 1
    print("please select exactly one object")
else:
    print("please select exactly one object")

-- OPERATOR CLASSES ---

@operator
class MIRROR_OT_Mirror(bpy.types.Operator):
    bl_idname = "object.mirror"
    bl_label = "X mirror to the selected object.mirror_mirror_x"
    bl_options = {'REGISTER', 'UNDO'}
    bl_description = "X mirror to the selected object.mirror_mirror_x"

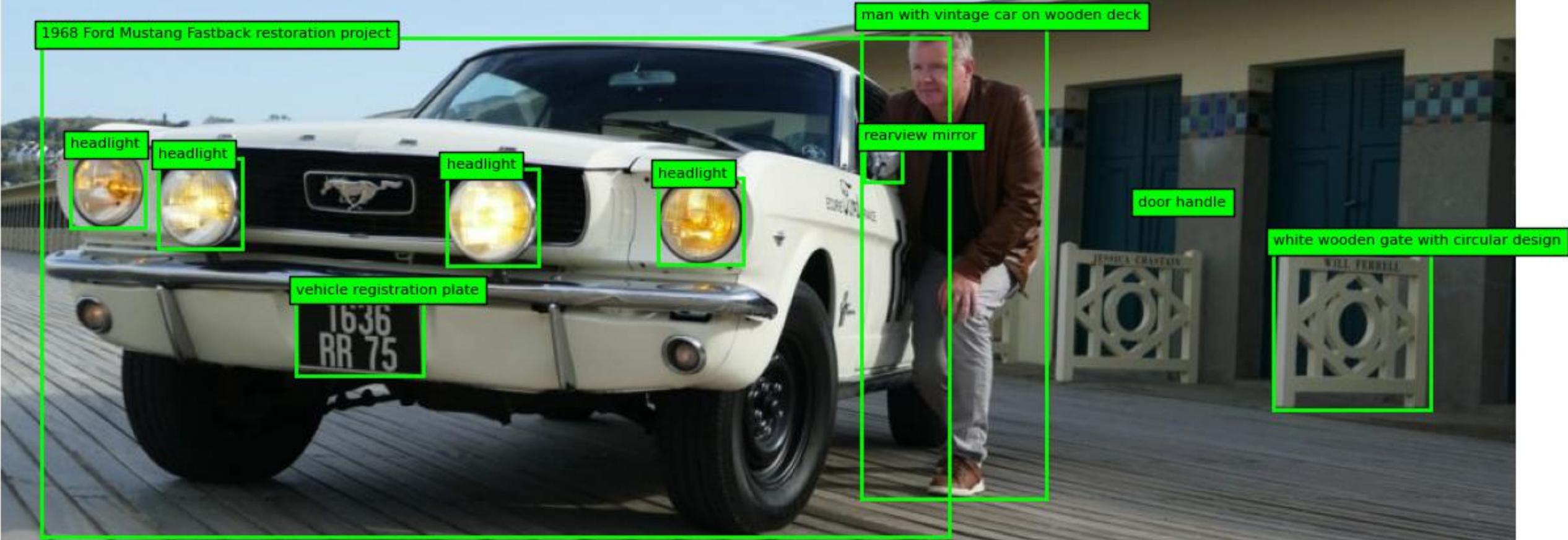
    @classmethod
    def poll(cls, context):
        if context.object is None or context.object.type != 'MESH':
            return False
        if context.active_object is not None and context.active_object.type != 'MESH':
            return False
        return True

    def execute(self, context):
        # Set mirror object to mirror
        mirror_mod = modifier_ob
        # Set mirror object to mirror
        mirror_mod.mirror_object = None
        if operation == "MIRROR_X":
            mirror_mod.use_x = True
            mirror_mod.use_y = False
            mirror_mod.use_z = False
        elif operation == "MIRROR_Y":
            mirror_mod.use_x = False
            mirror_mod.use_y = True
            mirror_mod.use_z = False
        elif operation == "MIRROR_Z":
            mirror_mod.use_x = False
            mirror_mod.use_y = False
            mirror_mod.use_z = True

        #selection at the end -add
        if ob.select==1:
            mirror_ob.select=1
            context.scene.objects.active = eval("Selected" + str(modifier))
            mirror_ob.select = 0
            bpy.context.selected_objects.append(data.objects[one.name])
            data.objects[one.name].select = 1
            print("please select exactly one object")
        else:
            print("please select exactly one object")
```

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]



Agenda

01

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

02

What is a foundation model?

03

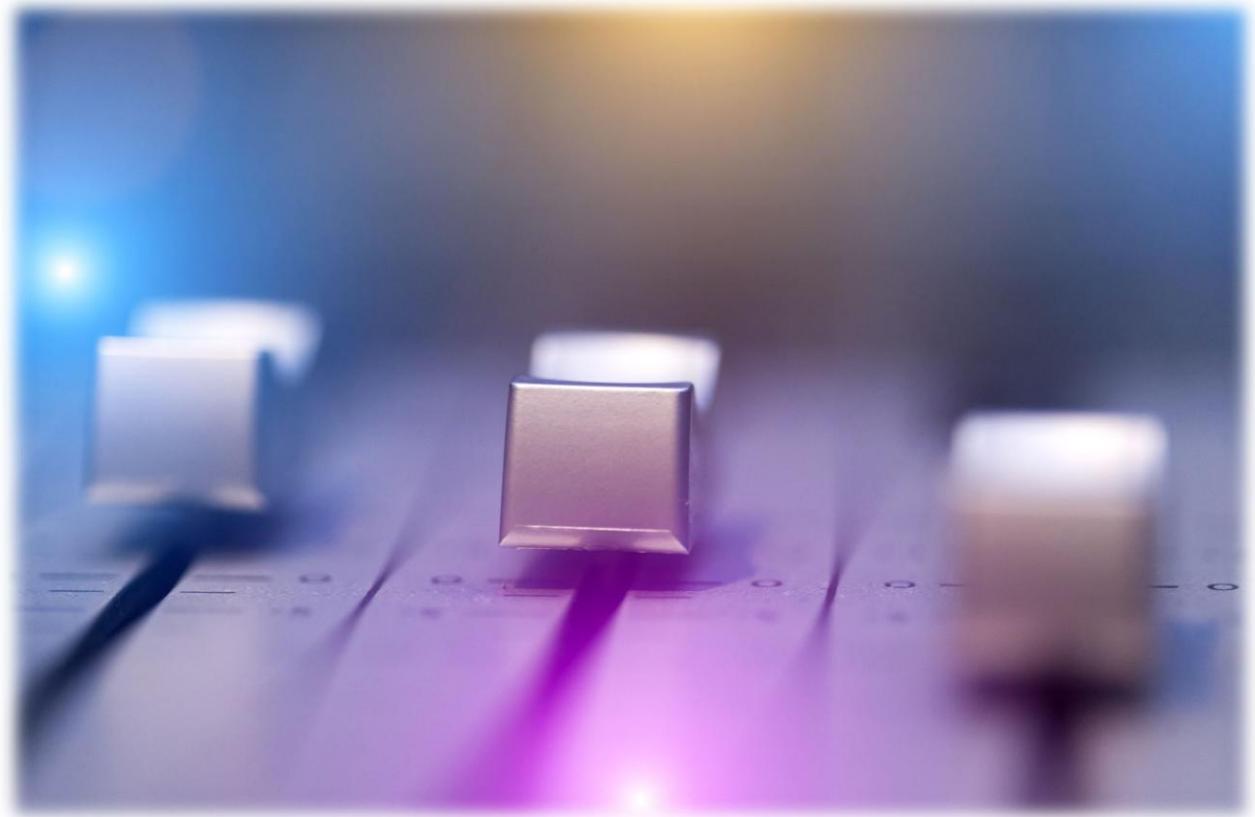
Fine-Tuning techniques

04

What is available on Azure?

05

Demos



Microsoft a 50 ans



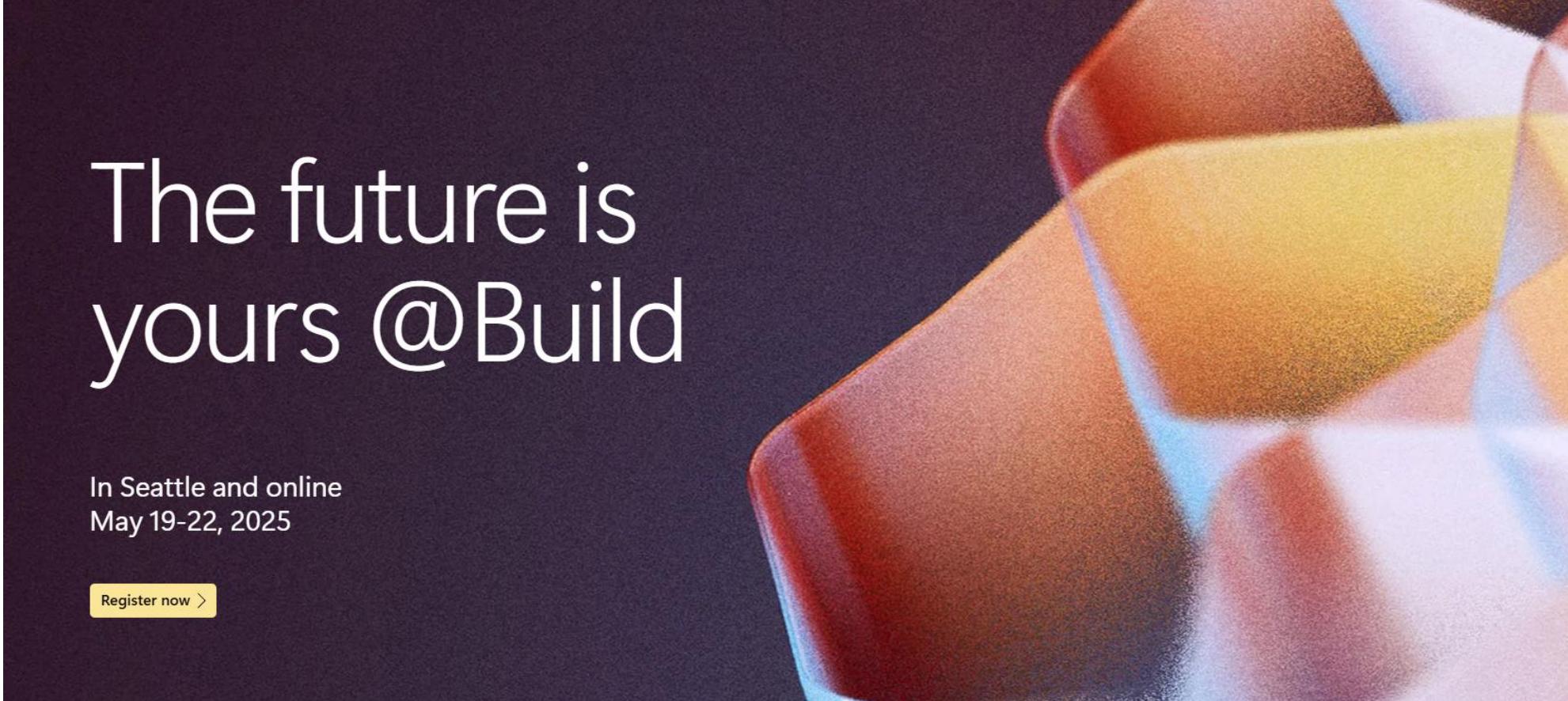
Celebrating Microsoft's 50 years
- The Official Microsoft Blog

programmez!
Le magazine des dévs - CTO & Tech Lead

1975-2025

SPÉCIAL PRINTEMPS 2025
HORS-SÉRIE #18





The future is yours @Build

In Seattle and online
May 19-22, 2025

[Register now >](#)

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 Agents 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.



Announcing GPT-4.1, 4.1-mini, and 4.1-nano models in Azure OpenAI Service

[Announcements](#) • Apr 14, 2025 • 2 min read

Announcing the GPT-4.1 model series for Azure AI Foundry and Microsoft Azure Blog >

We are excited to share the launch of the next generation of the GPT-4.0 model series with GPT-4.1, 4.1-mini, and 4.1-nano to Microsoft Azure OpenAI Service.

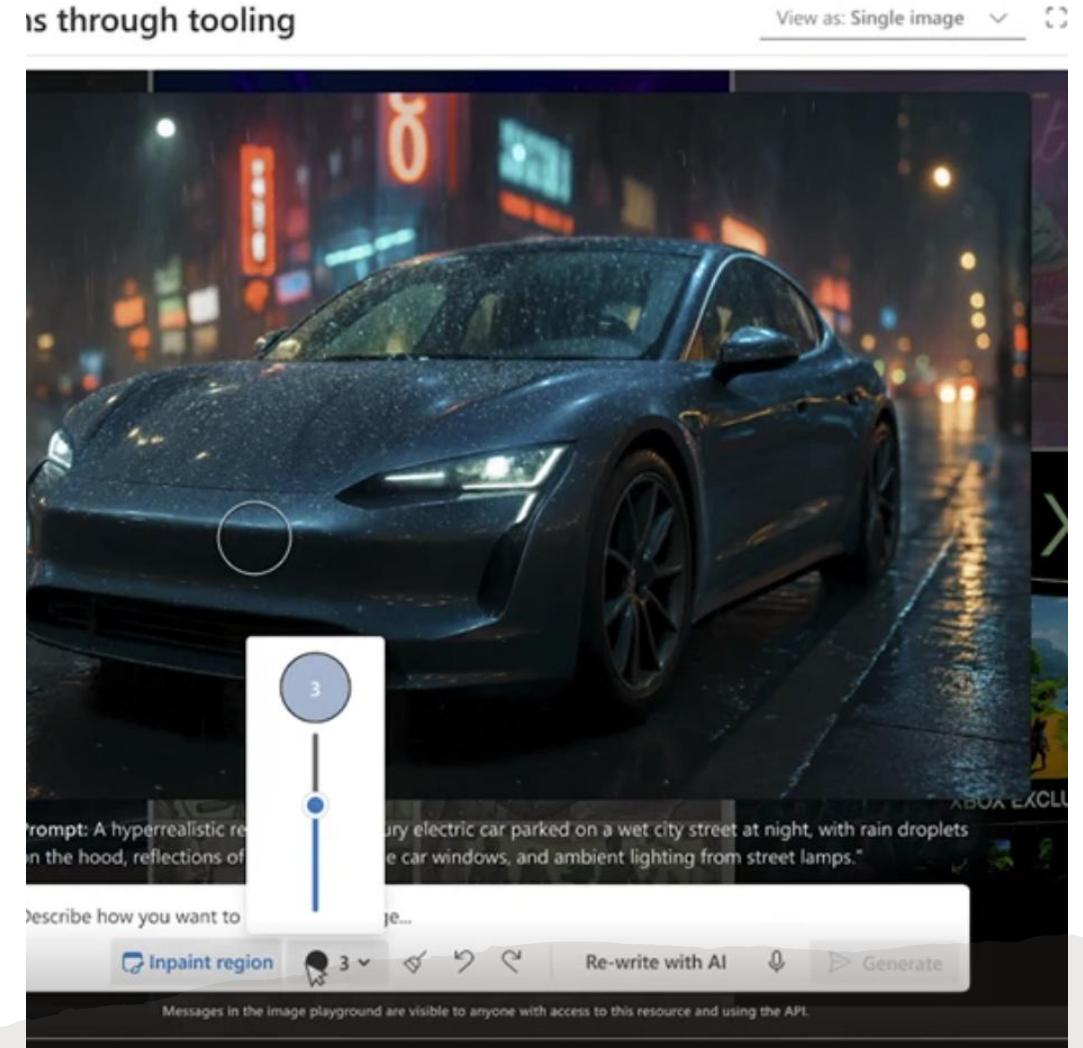


Global AI community France

<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





MERCI

TACK

ขอบคุณครับ

KIITOS

MULȚUMESC

DANKE

SALMAT PO

GRACIAS

감사합니다

شکرًا

DZIĘKUJĘ

OBRIGADO

MAHALO

TEŞEKKÜRLER

ευχαριστώ

ধন্যবাদ

СПАСИБО

ഡാൻറി

ଧତିଲ

Thank you!

多謝晒

PALDIES

ДЯКУЮ

ĎAKUJEM

متشكرم

TERIMA KASIH

ТАК

HVALA

شكريه

DĚKUJI

谢谢

CÁM ƠN

DANKON

GRAZIE

KÖSZÖNÖM

БЛАГОДАРЯ

ありがとうございます