



# Microsoft Ignite





# Unlocking NLP Potential: Fine-Tuning with Microsoft Olive

Workshop PRE016

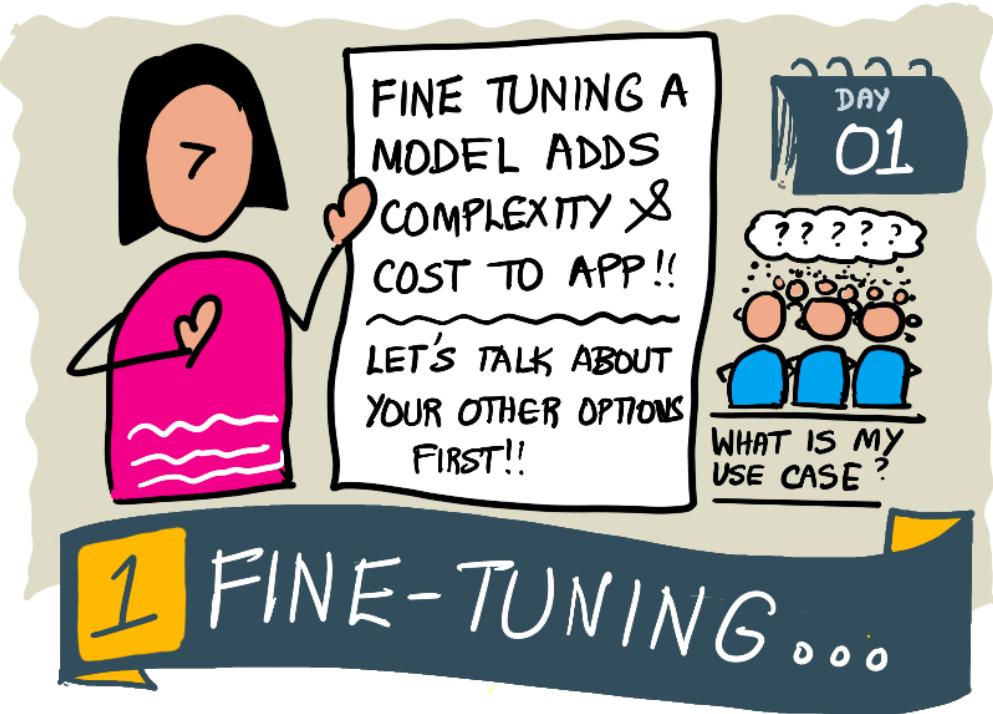
Sam Kemp Principal Program Manager  
Lee Stott Principal Cloud Advocate

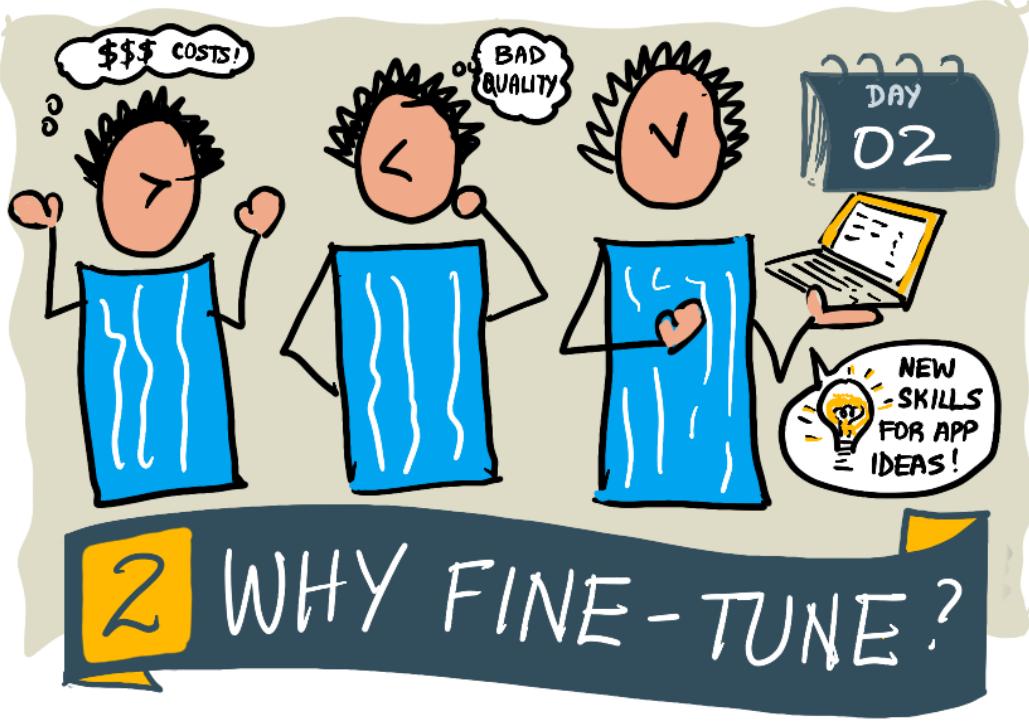


# What are we going to cover

Fine tuning a common practice in machine learning where we retrain an existing model **with new data** to improve performance on task.

This is an advanced technique that **requires some expertise** to get desired results. Incorrect usage may degrade performance.





## WHY SHOULD I FINE-TUNE?

Fine-Tuning may be appropriate if your desired **response quality** is not achievable with prompt engineering or RAG approaches.

- ✓ TOKENIZATION COSTS
- ✓ PROMPT ENGINEERING LIMITS
- ✓ APPLICATION DOMAIN NEEDS
- ✓ UPSKILLING CHEAPER MODELS

Another reason may be the **cost efficiency** achieved by reduced token usage or ability to upskill a cheaper model (within reason).



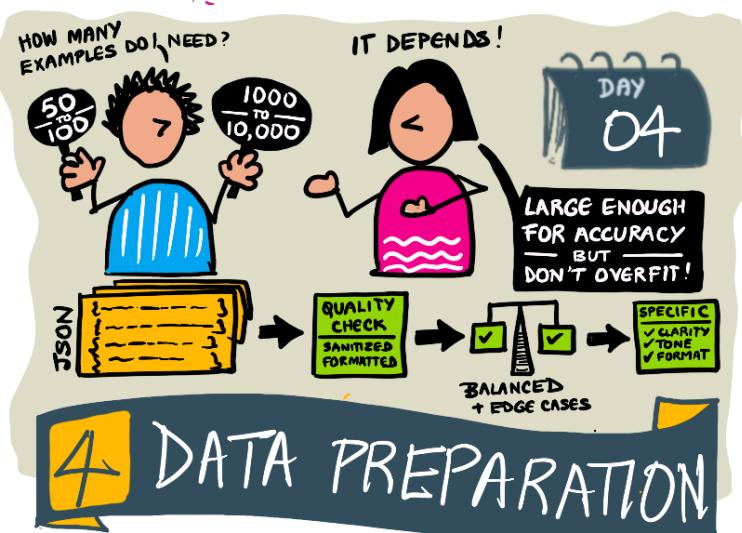
- DATA GATHERING & FORMATTING
- MODEL TRAINING & COMPLEXITY
- MODEL EVALUATION & ITERATION
- MODEL DEPLOYMENT & MAINTENANCE

## WHEN SHOULD I FINE-TUNE?

But the approach is valid only if the **benefits outweigh costs**.

- Do you have a good use case?  
(format, edge cases, new skills)
- Have you tried other options?
- Did you factor in other costs?  
(compute, data, maintenance)
- Did you confirm the benefits?  
(evaluation, region availability)

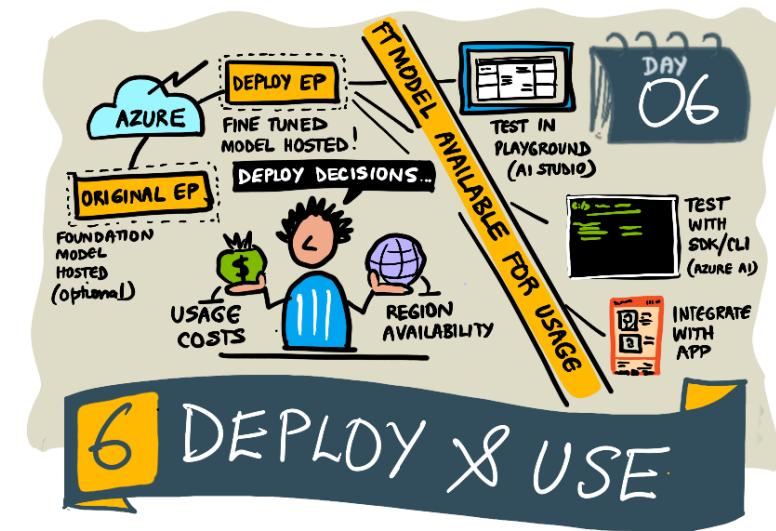
# PREP - Train - Evaluate - deploy



- ✓ SIZE OF TRAINING DATA (#examples)
- ✓ FORMAT OF EXAMPLE (model based)
- ✓ DATA REPRESENTATION (fairness)
- ✓ REQUIREMENTS COVERAGE (quality)

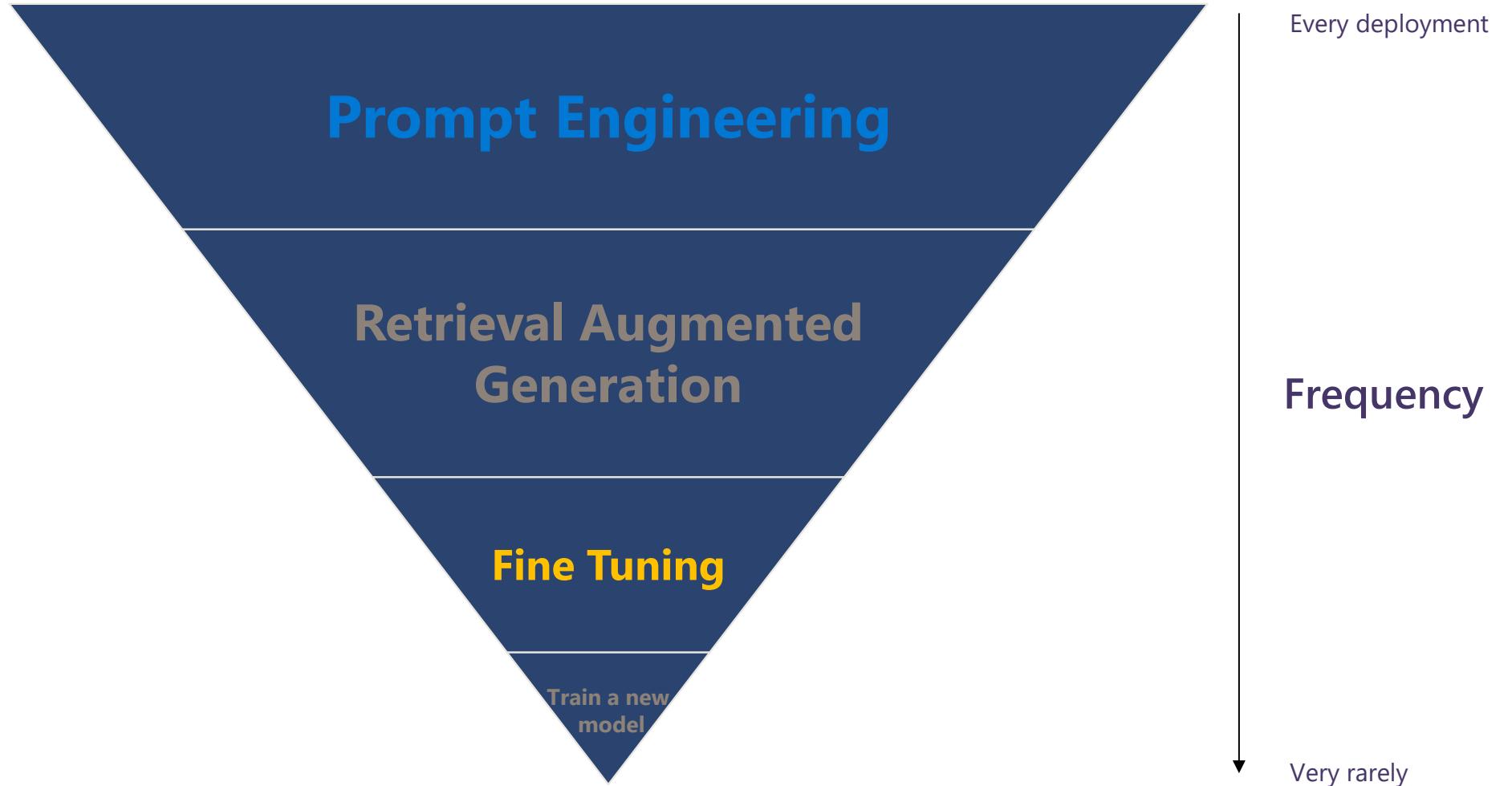


- ✓ UPLOAD DATA FOR TRAINING
- ✓ CREATE FINE TUNING JOB
- ✓ MONITOR JOB STATUS
- ✓ TEST FINE TUNED MODEL



- ✓ REGION AVAILABILITY (constraint)
- ✓ MODEL RATE LIMITS (shared)
- ✓ USE IN PLAYGROUND (validate)
- ✓ USE IN SDK / APP (integrate)

# Hierarchy of language model customization



# What about on-device AI?

## Why now?

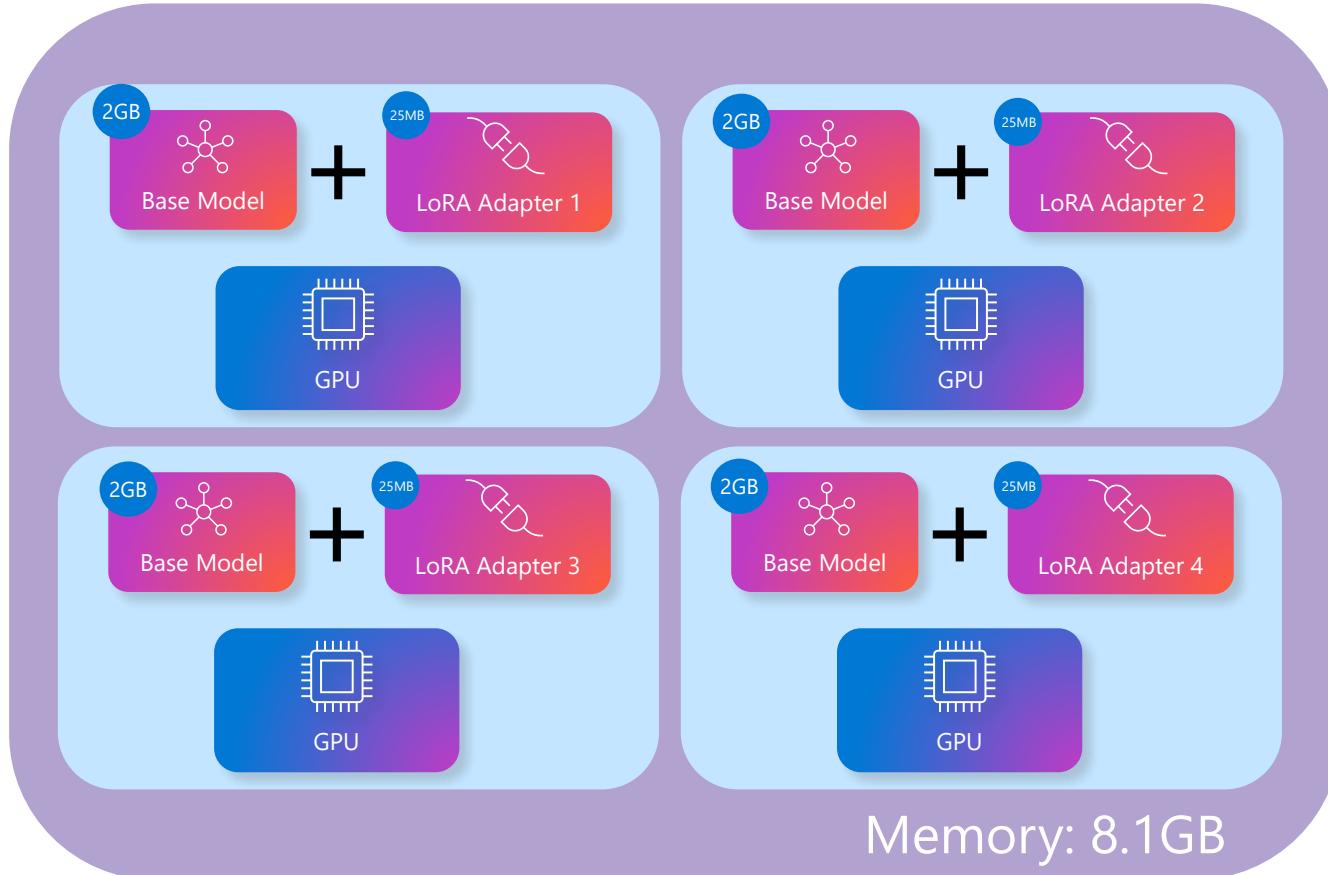
- Fine-tuning boosts model efficacy for SLMs that can be inferenced on-device.
- NPUs boost performance and ease power consumption.
- Multi-LoRA serving paradigm means device can *efficiently* host many fine-tuned adapters that give high model quality.
- It is not *all on device* or *all on cloud* BUT... **Hybrid**

## Motivation

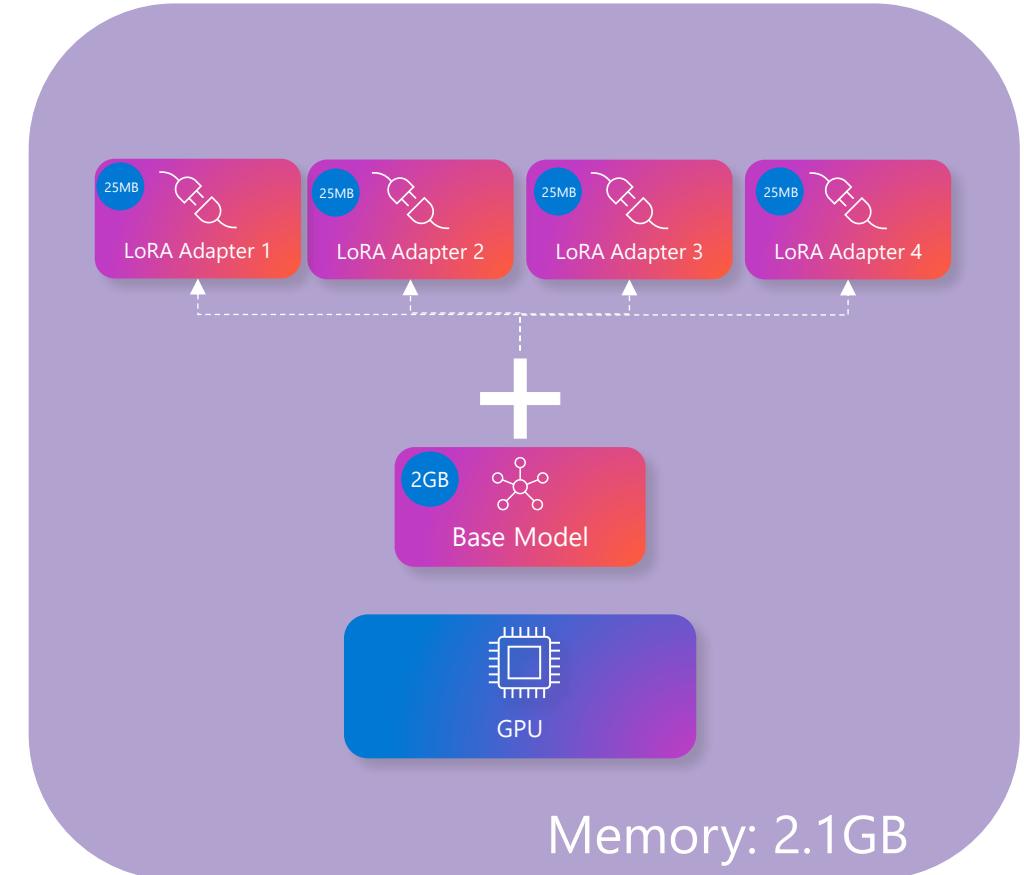
- COGs
- Privacy
- Improve latency
- Disconnected scenarios

# Multi-LoRA Serving

Smaller resource footprint



Serving LoRA models separately



Multi-LoRA Serving

# Hybrid AI Patterns

Fallback	Offload	Gating	Federated
<ul style="list-style-type: none"><li>• Prefer client and fallback to cloud when client is not capable, or</li><li>• Prefer cloud and fallback to client when no network</li></ul>	<ul style="list-style-type: none"><li>• Run part of model pipeline locally and rest of models on cloud, or</li><li>• Run part of model pipeline in cloud and rest of models locally</li></ul>	<ul style="list-style-type: none"><li>• Run lightweight model locally and use results to decide whether to call cloud</li></ul>	<ul style="list-style-type: none"><li>• Use local SLM to determine which cloud models to use, or</li><li>• Use cloud LLM to determine client models to use</li></ul>

# What you will build in this lab...

You will develop a travel companion application that leverages the **Hybrid AI pattern**:

- **Large Language Models (LLMs) hosted in the cloud**, and
- **Small Language Models (SLMs) hosted on the end-user device**

The application provides personalized travel recommendations, itinerary planning, and real-time assistance to travellers.

We will focus on the specific areas

1. Setting up **Azure AI Compute**
2. Prepare **Training Data**
3. One-Button **Fine-Tuning of a LLM** (GPT3.5-Turbo) with Azure AI Studio
4. **Deploy** the Fine-Tuned **LLM** (GPT3.5-Turbo) to a **cloud endpoint**
5. **Fine-tune and optimize** an **SLM** (Phi-3.5-Mini-Instruct) for **on-device inference** using Olive and the ONNX Runtime.
6. **Evaluate** the model's performance and compare it with the Azure-deployed model.
7. Create a **.NET/Python application** that inferences *both* the LLM in the cloud (from #4) and SLM on-device (from #5)
8. Clean up and delete resources.

# Microsoft AI tooling and services

## Best-in-class AI foundation models



### Azure AI Services

Pre-trained, turnkey solutions for intelligent applications



### Azure Machine Learning

Full-lifecycle tools for designing and managing AI models



### Responsible AI Tooling

Build and manage apps that are trustworthy by design



### Azure AI Studio

A comprehensive platform to develop and deploy custom copilots



### AI Toolkit for VS Code

End-to-end AI Developer toolkit download, finetune, deploy



# Lab 1. Environment Setup





00 : 00 : 00

Change Clock Type

Digital

Duration: 00 ▾ 15 ▾ 00 ▾

TimeUp Reminder (Optional): -- ▾ -- ▾ -- ▾

Choose Sound Effect

Tick ▾

Choose TimeUp Sound

Alarm ▾

Enable Count Up  Combine With Bar Clock

Start

Pause

Stop

Reset

# Lab 2 Data Preparation



# Synthetic Dataset Generation Using GPT

Prompt

Generate me some sample travel agent interactions which could happen in the commercial travel industry with realise content of a maximum of 80 words be accurate and realistic.

Extract conversations from each discussion in JSON Format

here are some examples

```
{"messages": [{"role": "system", "content": "You are an AI travel assistant that helps people plan their trips. Your objective is to offer support for travel-related inquiries, such as visa requirements, weather forecasts, local attractions, and cultural norms. You're helpful, friendly, and engaging. You do not provide answers not related to travel."}, {"role": "user", "content": "What's a must-see in Paris?"}, {"role": "assistant", "content": "Oh la la! You simply must twirl around the Eiffel Tower and snap a chic selfie! Want to feel like royalty? How about a visit to the grand Palace of Versailles next?"}]}
```

The final output should be in this format

# Considerations for Dataset Preparation

Remove

- Personal/sensitive information

Ensure

- Diverse and high-quality examples

Use

- Weights for specific assistant messages



00 : 15 : 00

Change Clock Type

Digital

Duration:

00

15

00

TimeUp Reminder (Optional):

--

--

--

Choose Sound Effect

Tick

Choose TimeUp Sound

Alarm

Enable Count Up  Combine With Bar Clock

Start

Pause

Stop

Reset



# Lab 3. Simple Fine Tuning using Azure AI Studio



# Fine Tuning with Azure AI



## Data

Curate 100s+ examples  
Format as chat or completion  
Define training & validation sets  
Upload data to the service



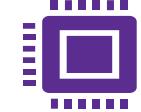
## Training

Select base model  
Set hyperparameters  
Specify data  
Executes on Shared Compute



## Evaluation

Training & validation loss  
Training & validation accuracy



## Deployment

Model endpoint for inferencing with AOAI

**Create a custom model**

Base model

Training data

Validation data

Advanced options

Review

**Base model**

Every fine-tuned model starts from a base model which influences both the performance of the model and the cost of running your custom model.

[Learn more about each base model](#)

**Base model type**

gpt-3.5-turbo

**Model suffix**

amazing\_finetuned\_model

**Create a custom model**

Base model

Training data

Validation data

Advanced options

Review

**Training data**

Select a training dataset to use when customizing your model. Training data must be in a JSON file and should consist of several hundred prompt/completion pairs.

[Learn more about preparing your data for Azure OpenAI](#)

Selected training file: validation.json

Choose dataset Local file Azure blob or other shared web locations

Drag and drop. Browse for a file

**Create a custom model**

Base model

Training data

Validation data

Advanced options

Review

**Validation data**

Select up to one validation dataset to use when iteratively assessing your customized model's performance during training. Validation data must be in a JSON file and should be representative of the training data without repeating any of it.

[Learn more about preparing your data for Azure OpenAI](#)

Selected validation file: validation.json

Choose dataset Local file Azure blob or other shared web locations

Validation file FT\_samsum\_val(2).jsonl  
wiz\_azure.validation.jsonl  
FT\_samsum\_val(1).jsonl  
validation.jsonl

Back Next Cancel

**Create a custom model**

Base model

Training data

Validation data

Advanced options

Review

**Training data**

Select a training dataset to use when customizing your model. Training data must be in a JSON file and should consist of several hundred prompt/completion pairs.

[Learn more about preparing your data for Azure OpenAI](#)

Selected training file: validation.json

Choose dataset Local file Azure blob or other shared web locations

Drag and drop. Browse for a file

**Create a custom model**

Base model

Training data

Validation data

Advanced options

Review

**Validation data**

Select up to one validation dataset to use when iteratively assessing your customized model's performance during training. Validation data must be in a JSON file and should be representative of the training data without repeating any of it.

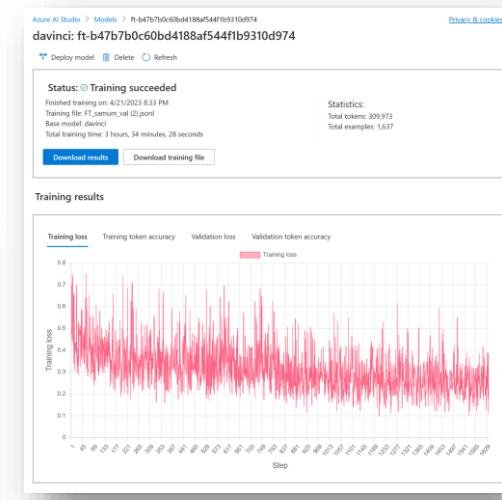
[Learn more about preparing your data for Azure OpenAI](#)

Selected validation file: validation.json

Choose dataset Local file Azure blob or other shared web locations

Validation file FT\_samsum\_val(2).jsonl  
wiz\_azure.validation.jsonl  
FT\_samsum\_val(1).jsonl  
validation.jsonl

Back Next Cancel



**Deployments**

Deployments provide endpoints to the Azure OpenAI base models, or your fine-tuned models, configured with settings to content moderation model, version handling, and deployment size. From this page, you can view your deployments, edit them, or create new ones.

Deployment name	Model name	M...	Deployment...	Capacity	Status
chatset	text-chat-davinci-002	1	Standard	-	Suk
CD002	code-davinci-002	1	Standard	120K TPM	Suk
SMGPT4-32k	gpt-4-32k	0613	Standard	5K TPM	Suk
Text-ada-001BGQ	test-ada-001	1	Standard	1K TPM	Suk

**Deploy model**

Set up a deployment to make APIs calls against a provided base model or a custom model. Finished deployments are available for use. Your deployment status will move to succeeded when the deployment is complete and ready for use.

Select a model

Select a model Fine tuned

Deployment name

Deployment name ada-ft-29ba3837da1491d82a5dfb237dc173-alicia-tries-fine-tu...

davinci:ft-b47b7b0c60bd4188af544fb9310d974

Advanced options

Create Cancel

# Selecting a model to fine tune

Model catalog

Search

Llama-2-7b-chat	Llama-2-7b	CodeLlama-7b-hf	Llama-2-70b-chat	Llama-2-70b	Llama-2-13b-chat
Chat completion	Text generation	Text generation	Chat completion	Text generation	Chat completion
CodeLlama-7b-Instruct-hf	CodeLlama-34b-Instruct-hf	CodeLlama-13b-Instruct-hf	gpt-4	gpt-4-32k	babbage-002
Text generation	Text generation	Text generation	Chat completions	Chat completions	Completions
davinci-002	gpt-35-turbo	tiiuae-falcon-7b-instruct	tiiuae-falcon-7b	tiiuae-falcon-40b	tiiuae-falcon-40b-instruct
Completions	Chat completions	Text generation	Text generation	Text generation	Text generation
mistralai-Mistral-7B-Instruct-v01	tiiuae-falcon-40b	mistralai-Mistral-7B-v01	tiiuae-falcon-7b	mistralai-Mixtral-8x7B-Instruct...	openai-whisper-large
Chat completion	Text generation	Text generation	Text generation	Chat completion	Speech recognition

- **Considerations:**
  - **Capabilities:** Which models are fine tunable? What can the base model be fine tuned to do?
  - **Cost:** What's the pricing model for fine tuning
  - **Customizability:** How much can I modify the base model – and in what ways?
  - **Convenience:** How does fine tuning actually happen – do I need to write custom code? Do I need to bring my own compute?
  - **Safety:** Fine tuned models are known to have safety risks – are there any guardrails in place to protect against unintended harm?

# Fine Tuning Best Practices



# Fine Tuning: Built in Safety

## Automatic safety assessments:

- Training data is sampled for potentially harmful content
- Simulated adversarial conversations with fine tuned models
- No charge for rejected jobs – and pointers to how to fix!

## Private & Secure

- Evaluated in dedicated, customer specific, private workspaces
- Evaluation endpoints in same geography as AOA resource
- Assessments are not stored or visible – only final status is persisted

The screenshot shows the Azure OpenAI Studio interface. On the left, there's a sidebar with various options like Home, Model catalog, Backgrounds, Chat, Assistants, Images, Completions, and Fine-tuning. The main area has a title "Welcome to the newly updated Azure OpenAI Studio" and a sub-section "Fine-tuning". It displays a list of fine-tuning jobs, one of which is highlighted: "fjob-40613: fjob-7885a78fcf7440d6b1fa7c6bb4ae737e-safety-toxic". Below this, a detailed view shows the job status as "Training failed". The status message indicates that safety evaluations identified scores above the acceptable threshold for [Violence, Self Harm], and it suggests retraining with a safe dataset. The event log at the bottom shows the process flow from postprocessing to completion.

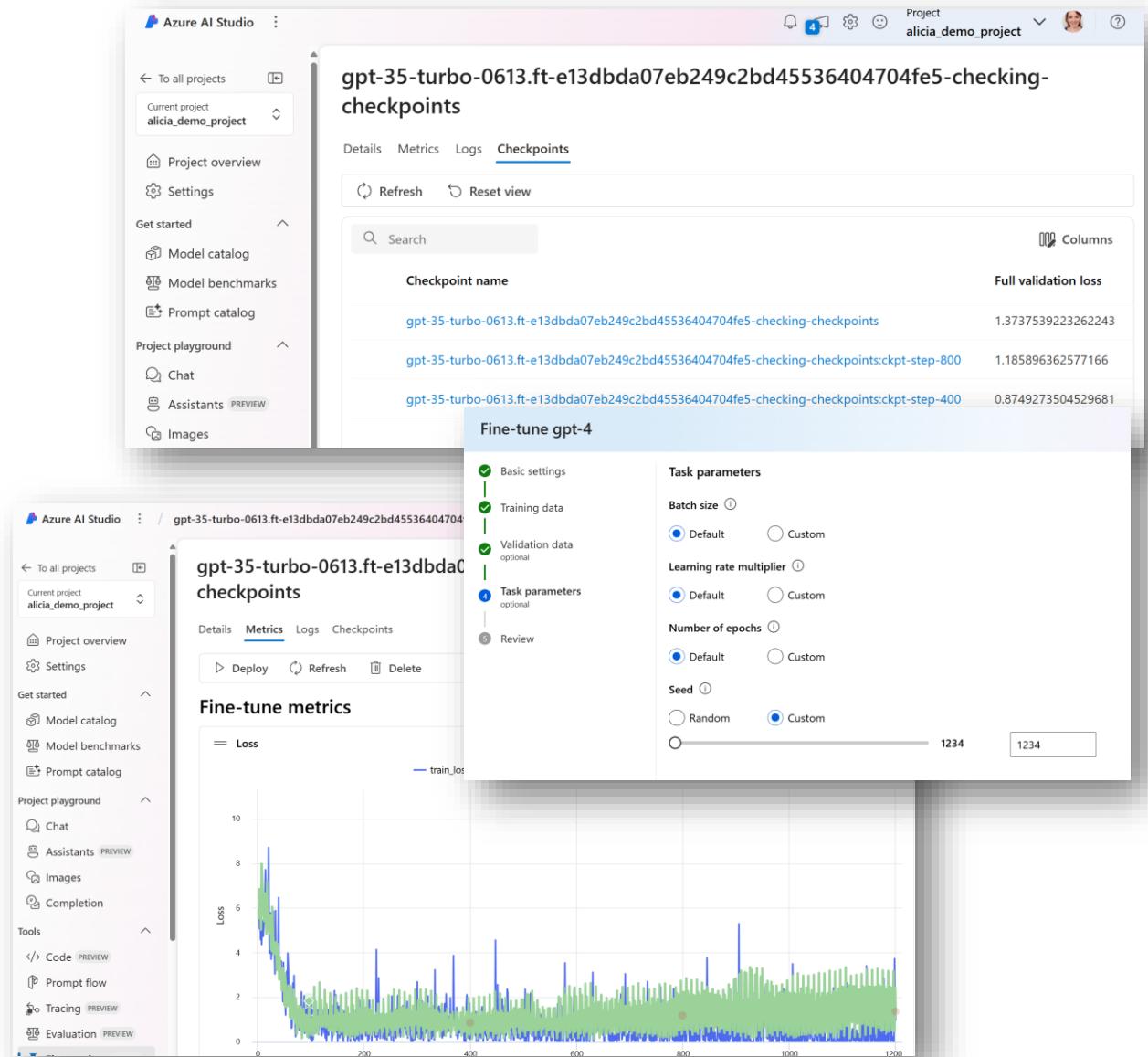
# Checkpoints, full validation metrics, and more!

**Checkpointing support in API, AI:** view and deploy training checkpoints

**Full Validation Statistics:** Detailed information on accuracy & loss

**Seed Property** for reproducible training runs

**Epochs, Learning Rate, Batch Size** parameters to optimize for your data





00 : 35 : 00

Change Clock Type

Digital

Duration:

00

35

00

TimeUp Reminder (Optional):

--

--

--

Choose Sound Effect

Tick

Choose TimeUp Sound

Alarm

Enable Count Up  Combine With Bar Clock

Start

Pause

Stop

Reset

# **Break**

20 Mins



# Lab 4. Deployment





00 : 35 : 00

Change Clock Type

Digital

Duration:

00

35

00

TimeUp Reminder (Optional):

--

--

--

Choose Sound Effect

Tick

Choose TimeUp Sound

Alarm

Enable Count Up  Combine With Bar Clock

Start

Pause

Stop

Reset

# **Move on Lab 5**

Whilst your Fine Tuned Model is deploying

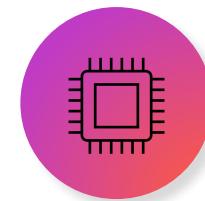
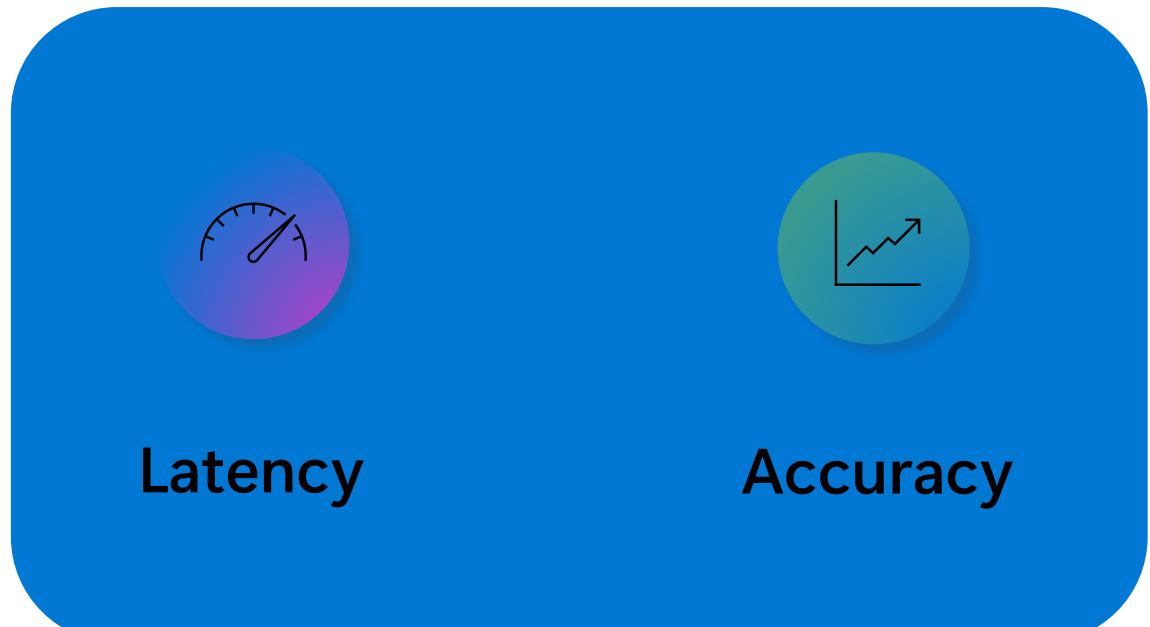


# Lab 5. Optimizing an SLM for On-device inference with Olive



# The key challenge with on-device AI

How can we ship an accurate model with low latency  
within given hardware constraints?



Hardware

# Techniques

What can an AI Engineer do to deliver high accuracy with low-latency on a hardware constrained device?

- Choose a high-quality small model (e.g. <3B parameters)
- Finetune a model
- Quantize the model
- Optimize the model graph
- Choose an efficient model runtime
- Exploit rapidly evolving hardware (e.g. NPU)
- Optimize the serving (e.g. single-LoRA vs multi-LoRA)
- Compress the model

# Solution Landscape

What are the current solution offerings for on-device



Finetuning

LLaMA<sup>GC</sup>

vLLM



ONNX  
RUNTIME



GGUF



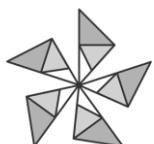
ONNX

PyTorch



Pruna AI

vLLM



OLIVE

Model Format

Optimization

Qualcomm

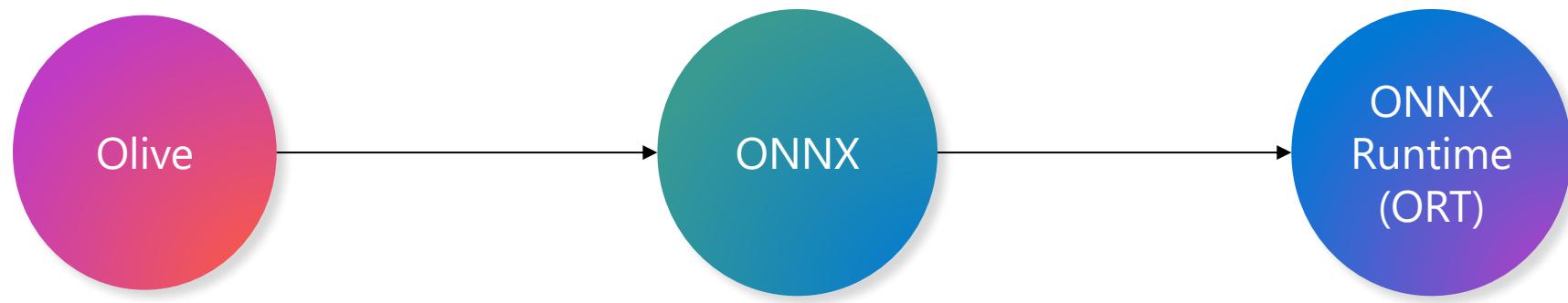
intel.

AMD

Hardware

# The ONNX Trilogy

an E2E solution for On-Device AI



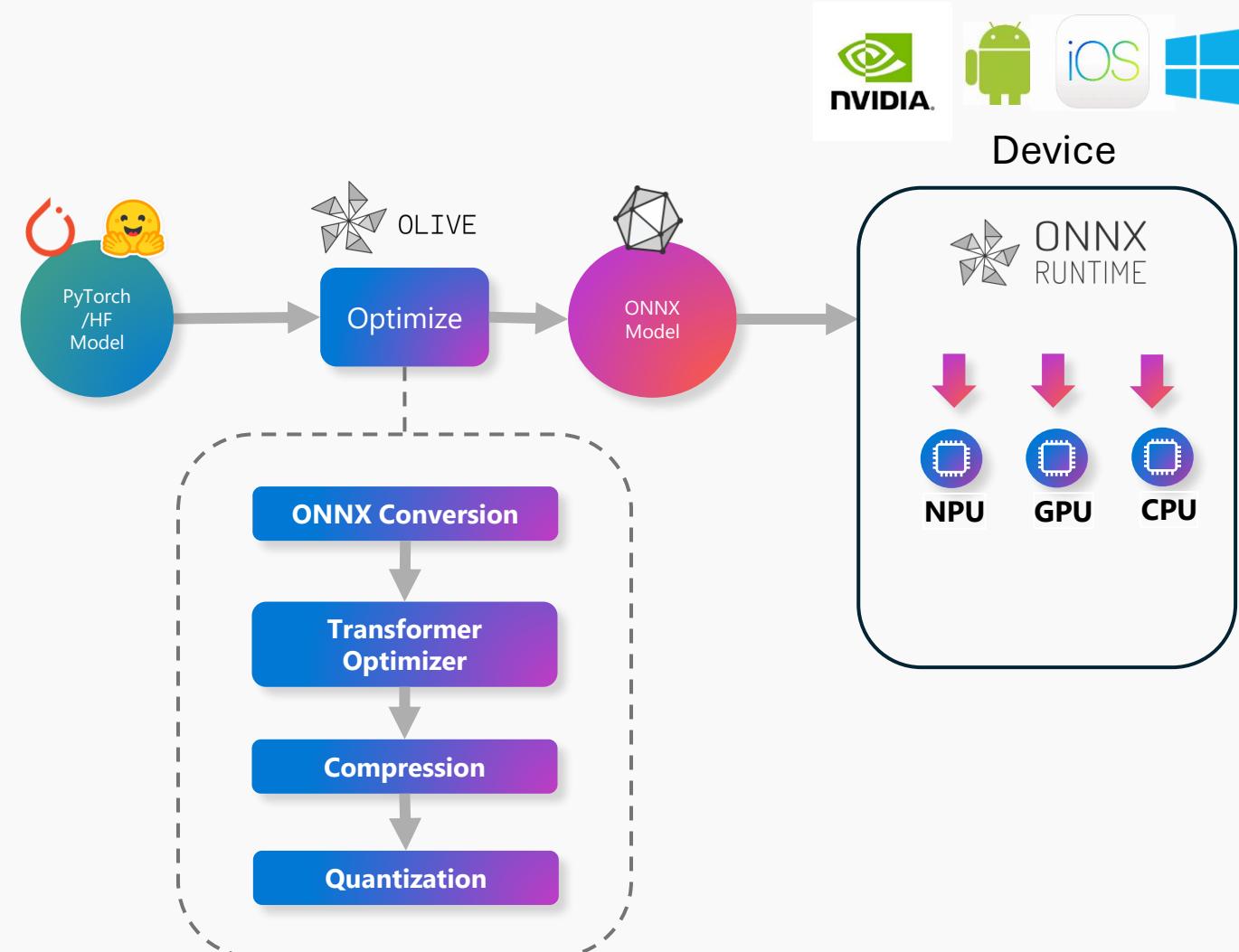
A toolchain creates  
optimized ONNX models

Open and interoperable file  
format for ML and DNN  
models.

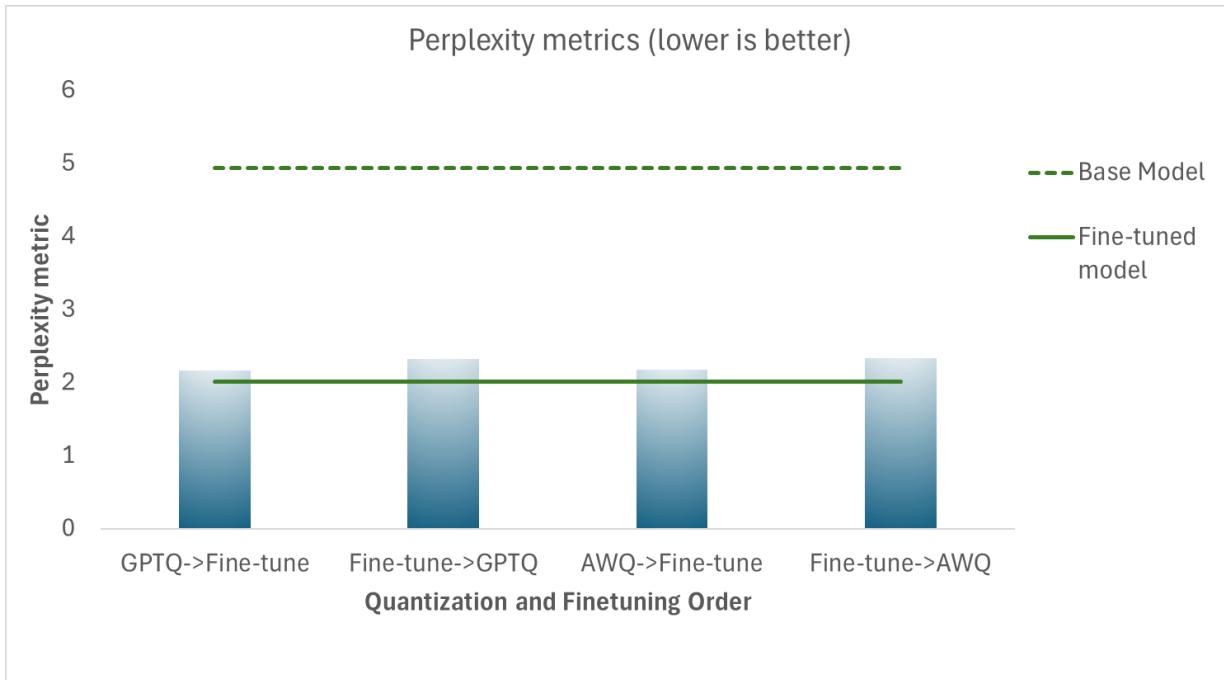
Fast and efficient *model  
inference engine* that works  
across a diverse range of  
hardware accelerators and  
platforms.

# What is Olive?

- **O(NNX)Live** – an AI model optimization toolkit for the ONNX Runtime.
- An **easy-to-use** and **integrated** toolchain for AI Engineers where the following tasks can be composed into a workflow (pipeline):
  - Finetuning (QLoRA/LoRA/LoftQ)
  - Graph Capture (Dynamo Exporter/Model Builder)
  - Model Optimization (CPU/GPU/NPU/DirectML)
  - Quantization (GPTQ, AWQ, WOQ)
  - Runtime tuning
  - Creates models for Multi-LoRA serving
  - Deployment
- **Automatically** finds the optimized model for a given hardware target without requiring specialized knowledge and manual work.



# Lab 5 – Finetune and Optimize a model for on-device Inference



## Lab 7 – Consumption of Model in App





00 : 35 : 00

Change Clock Type

Digital

Duration: 00 ▾ 35 ▾ 00 ▾

TimeUp Reminder (Optional): -- ▾ -- ▾ -- ▾

Choose Sound Effect

Tick

Choose TimeUp Sound

Alarm

Enable Count Up  Combine With Bar Clock

Start

Pause

Stop

Reset

# **Break**

20 mins

# **Finish Lab 4**

Test your deployed GPT3.5-Turbo Fine tuned Model from Lab 4

# Lab 6. Evaluate your Model





00 : 45 : 00

Change Clock Type  
Digital

second changed successfully!  
Duration: 00 ▾ 45 ▾ 00 ▾  
Second changed successfully!

TimeUp Reminder (Optional): -- ▾ -- ▾ -- ▾

>

Choose Sound Effect Tick

Choose TimeUp Sound Alarm

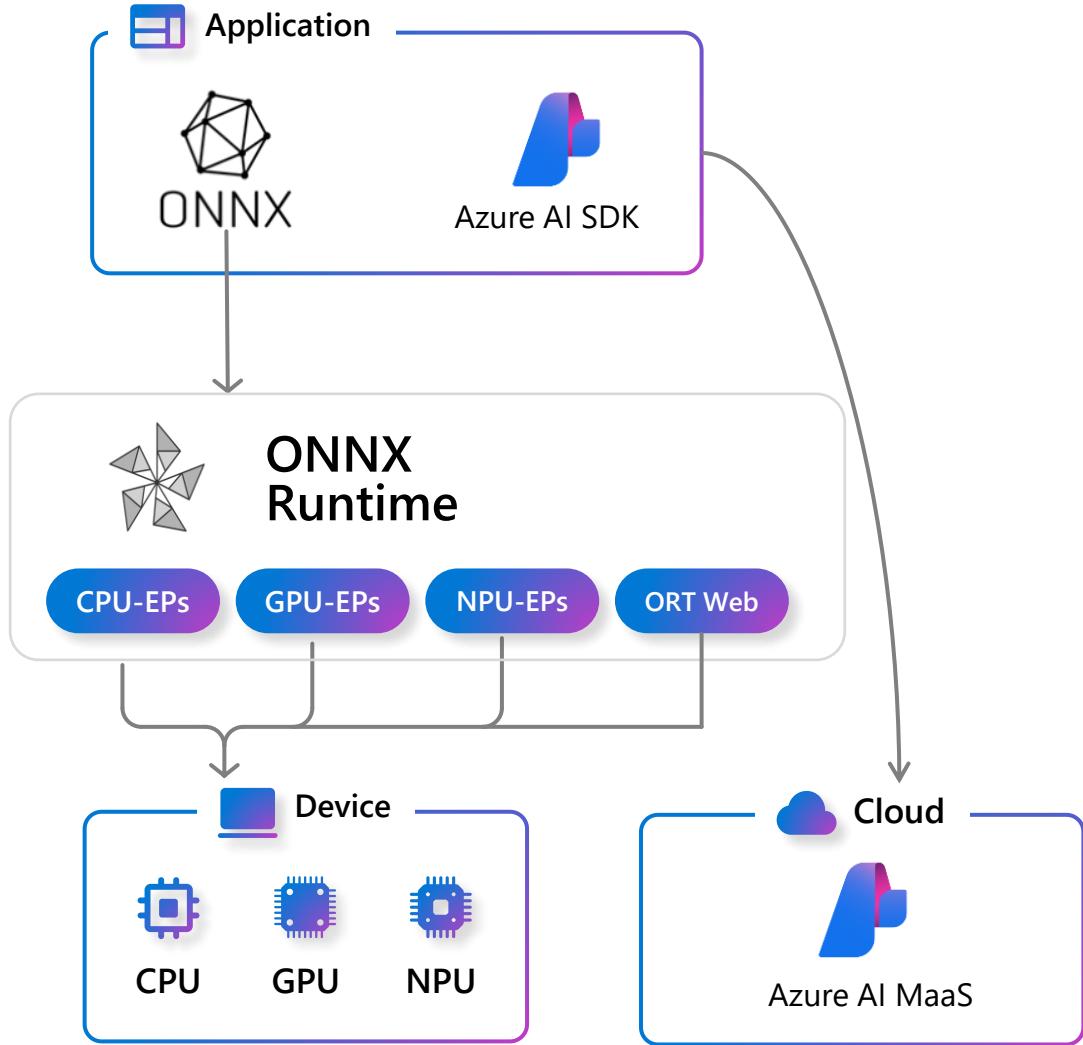
Enable Count Up  Combine With Bar Clock

Start Pause Stop Reset

# Lab 7. Consumption of Model in App



# Optimization and performance ONNX runtime



Execution provider	Company
Azure	Microsoft
QNN	Qualcomm
OpenVino	Intel
Vitis AI	AMD
DirectML	Microsoft
WebNN	Microsoft, Intel



00 : 15 : 00

Change Clock Type

Digital

Duration:

00

15

00

TimeUp Reminder (Optional):

--

--

--

Choose Sound Effect

Tick

Choose TimeUp Sound

Alarm

Enable Count Up  Combine With Bar Clock

Start

Pause

Stop

Reset



Please  
Take the survey!

ONNX Ecosystem Feedback





# Lab 8. Clean Up





EasyTimer

# Q&A

Continue to try this at  
<https://aka.ms/ignite/pre016>



# Your feedback is important!

Visit [aka.ms/MicrosoftIgniteEvals](https://aka.ms/MicrosoftIgniteEvals)  
to complete your session evaluation.



# Microsoft Learn Spark possibility

Thanks for joining us!

## Take the Microsoft Learn Challenge

Build the skills you need to thrive in the era of AI. Take the challenge before it ends on January 10, 2025 at [aka.ms/MSIgniteChallenge](https://aka.ms/MSIgniteChallenge).

## Verify your skills with Credentials

Take a free certification exam from Microsoft and/or GitHub, earn a Microsoft Applied Skills credential, or take a Practice Assessment onsite. Visit the Credentials desk in front of Room W179a.

## Dive deeper with Microsoft Learn Collections and Plans

Access the latest learning content on the topics you find interesting, all from trusted sources with Microsoft Learn Collections and Plans. Find out more at [aka.ms/LearnAtIgnite](https://aka.ms/LearnAtIgnite).