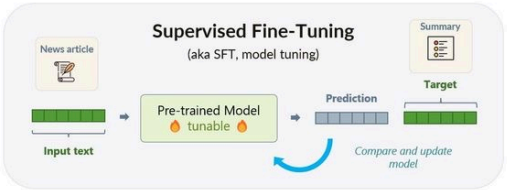
Azure OpenAI Fine Tuning

Specific Data.



Babbage-002 & Davinci-002:

- GPT3 based: smaller, lower latency
- Understand & generate natural language or code
- Completion support

GPT-35-Turbo:

- Most capable & cost effective GPT-3.5 model
- More sophisticated capabilities
- Chat support

Fine-tuning models

Models	Training per compute hour	Hosting per hour	Input Usage per 1,000 tokens	Output Usage per 1,000 tokens
Babbage-002	N/A	N/A	N/A	N/A
Davinci-002	N/A	N/A	N/A	N/A
GPT-3.5-Turbo (4K)	$45	$3	$0.0005	$0.0015
GPT-3.5-Turbo (16K)	$68	$3	$0.0005	$0.0015

Customization of PreTrained Models: Tailor models like GPT to specific data needs.

Improving Model Performance: Aligns model outputs more closely with userspecific styles and terminologies.

Dataset Requirements: Users provide their own data for training to finetune the models.

Training Process: Finetuning continues training a pretrained model on new datasets, requiring fewer resources than training from scratch.

Cost : Fine-tuning on Azure OpenAI can be expensive, especially for large models and extensive datasets.
Compute Costs: 10 hours * $10/hour = $100
Storage Costs: 5 GB * $0.02/GB = $0.10
API Usage: 500,000 tokens * $0.06/1,000 tokens = $30
Total Estimated Cost: $130.10

Accessibility through Azure: Integrated into Azure, finetuning benefits from its cloud infrastructure and services.

Custom Deployments: Models can be easily deployed within Azure, integrating with other Azure services.

Fine Tuning

Customization of already existing LLM, so it can act as

- Task specific
- Domain specific.

- **Example:**
  - Pre-trained GPT → fine-tuned on legal documents → becomes better at answering legal questions.
  - LLaMA 2 → fine-tuned on customer support chats → becomes a specialized support assistant.

✅ **TL;DR:** Fine-tuning = adapting a general LLM to a specific domain or task without training from scratch.

Imp. Fine Tuning can be really costly, so we are encouraged to use prompting rather.

Cost : Fine-tuning on Azure OpenAI can be expensive, especially for large models and extensive datasets.
Compute Costs: 10 hours * $10/hour = $100
Storage Costs: 5 GB * $0.02/GB = $0.10
API Usage: 500,000 tokens * $0.06/1,000 tokens = $30
Total Estimated Cost: $130.10

Imp. Not all model can be fine tuned and not all models are available in all regions.

Azure AI

✅ **Azure OpenAI Models Available for Fine-Tuning**

Model	Fine-Tuning Availability	Notes
GPT-3.5-Turbo (0613)	✅ Yes	Widely used for general-purpose tasks.
GPT-4	✅ Yes (Preview)	Advanced capabilities; fine-tuning is in preview.
GPT-4o	✅ Yes	Optimized for efficiency; fine-tuning is in preview.
GPT-4o-mini	✅ Yes	Lightweight version; fine-tuning is in preview.
GPT-4.1	✅ Yes	Latest model; fine-tuning is available in all regions with Global Training.
GPT-4.1-mini	✅ Yes	Mini version of GPT-4.1; fine-tuning is available in all regions with Global Training.
GPT-4.1-nano	✅ Yes	Nano version of GPT-4.1; fine-tuning is available in all regions with Global Training.

Mostly used; GPT 3.5

parameters; 175b

Open Source

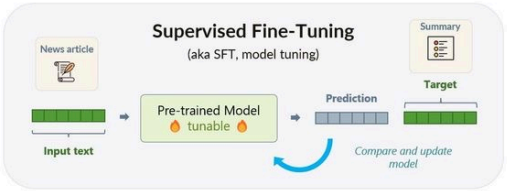
Here's a **shortlist of the most important open-source models that can be fine-tuned:**

Model	Type	Notes / Example Use
LLaMA 2 (Meta)	Text / Chat	7B–70B parameters; widely used for conversational and general NLP fine-tuning
Mistral	Text	Efficient models (7B–8x7B); general NLP tasks, Apache 2.0 license
T5 (Google)	Text	Text-to-text transformer; translation, summarization, classification
BERT (Google)	Text	QA, classification, sentiment analysis; standard NLP downstream tasks
CodeLlama	Code / Text	Optimized for code generation, understanding, and code-related tasks

LLAMA 2

7 - 70b

Need for Fine Tuning



Babbage-002 & Davinci-002:

- GPT3 based: smaller, lower latency
- Understand & generate natural language or code
- Completion support

GPT-35-Turbo:

- Most capable & cost effective GPT-3.5 model
- More sophisticated capabilities
- Chat support

- **DomainSpecific Knowledge:** When the model needs to generate text with specialized vocabulary and concepts.
- **Custom Instructions and Formatting**: When output needs to follow specific templates or formats.
- **Company or Brand Voice**: When text needs to reflect a particular style or tone.
- **Proprietary or Confidential Information**: When generating or processing sensitive information specific to an organization.
- **Enhanced Context Understanding**: When maintaining context over long interactions is crucial.
- **Unique Customer Interactions**: When handling specific customer queries unique to a product or service.
- **Regulatory Compliance**: When generating text that adheres to specific legal or regulatory standards.
- **Interactive Applications**: When creating engaging and interactive user experiences.

Imp Q1 when to choose Fine Tuning vs prompts.

Practical Rule of Thumb

Use Fine-Tuning when:

- Data is **relatively static.**
- You want **highly consistent answers** in a specific domain.
- You want to **teach the model company-specific terminology.**

Use Prompt Engineering when:

- Data changes frequently (like Power BI dashboards).
- You need **flexible, dynamic responses.**
- Data is **confidential** — RLS or filters can be applied before sending to the LLM.
- You want to **avoid training costs.**

Most Preffered in Corporates

Hybrid Approach

Fine-tune for domain knowledge + Prompt Engineering for dynamic context.

Example:

- Fine-tune on your **company style, tone, terminology.**
- Use **prompt engineering** to feed **real-time data** (e.g., Power BI rows filtered by RLS).

This is the **best of both worlds**, especially for corporate chatbots.

TL;DR:

- **Fine-tune = teach the model once** (good for static, domain-specific knowledge).
- **Prompt engineering = guide the model dynamically** (good for live, changing, confidential data).

Q1 How to Fine Tune ?

GPT 35 Turbo

1. Training Data

example

```
{"messages": [{"role": "system", "content": "You are a chatbot that always responds in a humorous way."},
              {"role": "user", "content": "What did you do over the weekend?"},
              {"role": "assistant", "content": "I became a professional couch potato and mastered the art of napping."}]}
```

What you are trying to do

1. **Model Behavior Customization**
   - You want the model to **always respond in a humorous way.**
   - Each example in your dataset shows the **desired style of response** (funny, witty, playful).
2. **Structure**
   - Each entry is a JSON object with a `"messages"` array:

```
json                                                    Copy code

{"role": "system", "content": "You are a chatbot that always responds in a humorou
{"role": "user", "content": "User question"},
{"role": "assistant", "content": "Desired humorous response"}
```

   - `"system"` : sets the **behavior/tone of the model.**
   - `"user"` : the **prompt/question** from the user.
   - `"assistant"` : the **response you want the model to generate.**
3. **Goal**
   - After fine-tuning on this dataset, the model **learns to consistently respond humorously**, even to new questions it hasn't seen before.
   - The model is **adapting its weights** based on the examples you provided.

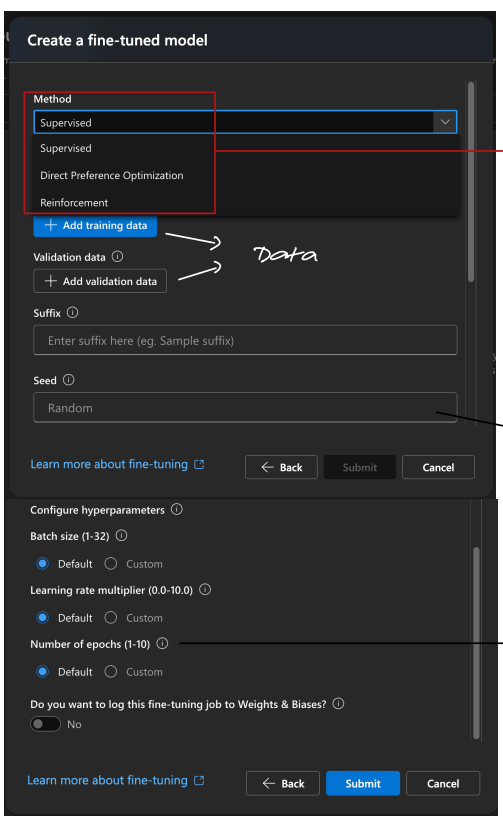
- System — Instruction
- User — Query
- Assistant — Answer

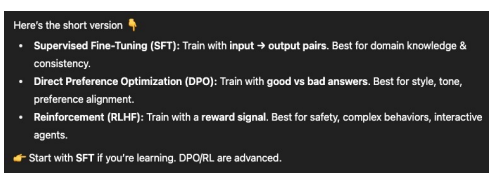
2. Validation Data

Similar to Train Data but not trained, this data is rather used for validation.

Imp. GPT 3.5 is best for experiment and is available East US 2.

FINE TUNING PRACTICALS





SFT = Supervised F.T

mostly used.

Seed

setting it will give same output multiple times.

Epoch

one complete cycle of train data

Imp. Epochs = 6 ; Training prompts = 15

Total steps = 6 x 15 = 90

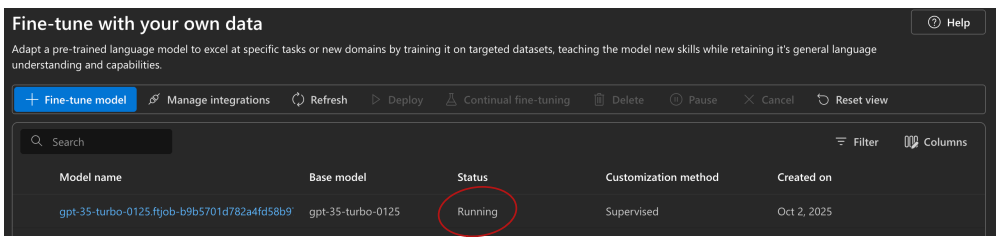
→ Loss should be very low

↳ How well model is learning overtime

→ Validation Loss → Training Data.

Deploy Fine Tune Model

So, we will be deploying our F.T model as Base Model in Deployments

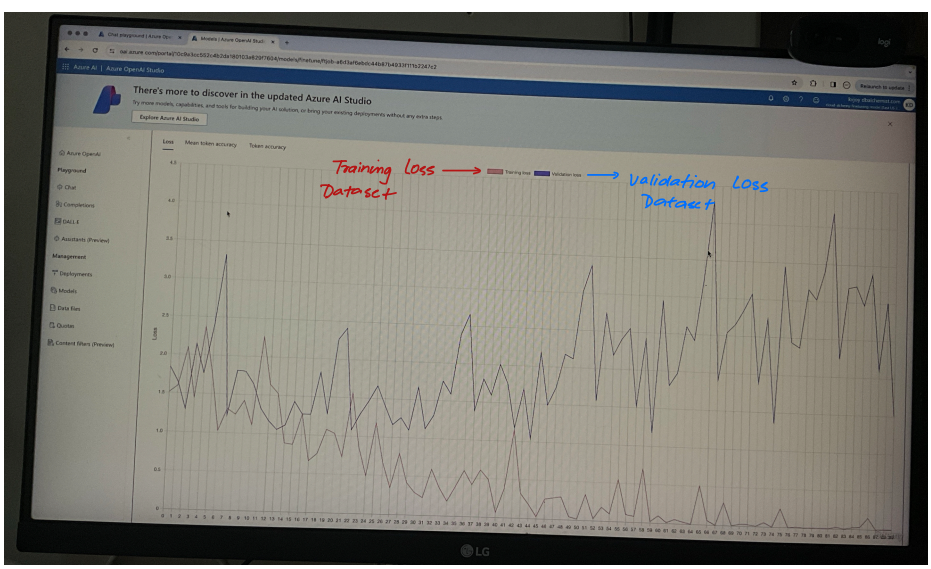


Once it is completed

↓

we will deploy F.T model as Base Model

Fine Tuned Model Evaluation;



Imp.! Loss should always be minimum.

Also, 90 Steps = 15 Datapoints x 6 epochs

Model = GPT 35 Turbo.

Azure OpenAI Fine Tuning