AI & Machine Learning

# When to use supervised fine-tuning for Gemini

October 4, 2024

**Erwin Huizenga**
Machine Learning Engineer, Google Cloud

**May Hu**
Product Manager, Vertex AI

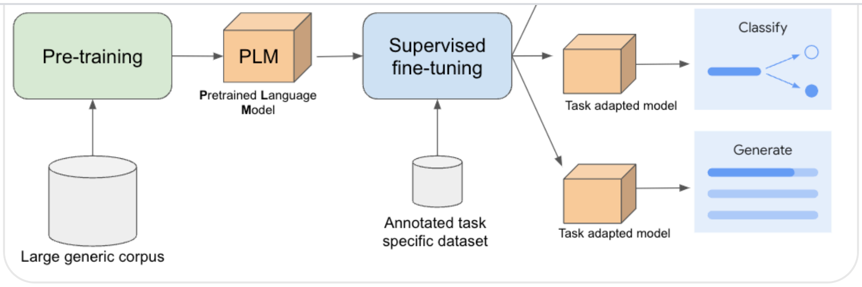
Have you ever wished you could get a foundation model to respond in a particular style, exhibit domain-specific expertise, or excel at a specific task? While foundation models like Gemini demonstrate remarkable capabilities out-of-the-box, there can be a gap between their general knowledge and the nuanced understanding required for specific applications.

tailor these powerful models for specialized tasks, domains, and even stylistic nuances. Developers often ask when to use SFT and how it compares against options like prompt engineering, in-context learning, and RAG.

This article delves into the what is SFT, when to embrace SFT and how it compares to other methods for optimizing your models output.

## What is supervised fine-tuning?

Developing a large language model (LLM) often begins with pre-training. During this stage, the model learns general language understanding from massive amounts of unlabeled text. The primary objective of pre-training is to equip the model with broad language comprehension. These pre-trained LLMs have shown remarkable performance across many tasks. This pre-trained model can then be adapted to enhance performance for downstream use cases that may require more specific understanding, such as summarizing financial documents.



In order to adapt the model for these use cases we fine-tune the pre-trained model using a task-specific annotated dataset. This dataset includes input examples (e.g., an earnings report) and their corresponding desired outputs (e.g., a summary). By associating inputs with their correct outputs, the model learns to perform the specific task. When we use an annotated dataset for fine-tuning, we call it supervised fine-tuning (SFT).

Model parameters are the primary mechanism for adjusting how the model behaves, parameters are the numerical values learned during the training process. How many of the model parameters we update during fine-tuning can differ, two common supervised fine-tuning approaches are:

- **Full fine-tuning:** updates all model parameters. However, full fine-tuning demands higher computational resources for tuning and serving, leading to higher overall costs.
- **Parameter-Efficient Fine-Tuning (PEFT):** refers to a class of methods which freeze the original model and only updates a small set of newly added extra parameters, enabling faster and more resource-efficient fine-tuning. PEFT is

constrained.

While both full fine-tuning and PEFT are considered supervised learning methods, they differ in the extent of parameter updates they perform and one may be more appropriate for your use case. As an example, Supervised fine-tuning for Gemini models on Vertex AI uses LoRA (Low-Rank Adaptation) which is a PEFT method.

## When to use supervised fine-tuning?

Consider using SFT when your goal is to enhance the model's performance on a specific, well-defined task, and you have access to a dataset of high-quality annotated examples. Supervised fine-tuning is particularly effective for efficiently activating and refining the relevant knowledge already present within the pre-trained model, especially when the task aligns with the original pre-training data. Here are some scenarios where SFT shines:

- **Domain expertise:** Infuse your model with specialized knowledge, transforming it into a subject matter expert in law, medicine, or finance.
- **Format customization**: Tailor your model's output to adhere to specific structures or formats.

- **Edge cases:** Improve the model's ability to handle specific edge cases or uncommon scenarios.
- **Behavior Control:** Guide the model's behavior, such as when to provide concise or detailed responses.

One of SFT's strengths is its ability to yield improvements even with limited amounts of high-quality training data, making it a more cost-effective alternative to full fine-tuning in many cases. Furthermore, fine-tuned models tend to be easier to interact with. Through SFT, the model learns to perform the task effectively, reducing the need for lengthy and complex prompts during inference. This translates to lower costs and reduced inference latency.

While SFT is great for reinforcing existing knowledge, it won't solve all your challenges. It may not be the ideal solution for scenarios involving dynamic or evolving information, such as incorporating real-time data. In these cases, other techniques might be more suitable, so let’s walk through these other options.

## Additional options for LLM customization

techniques offer effective ways to adapt the model's behavior, each with its strengths and trade-offs.

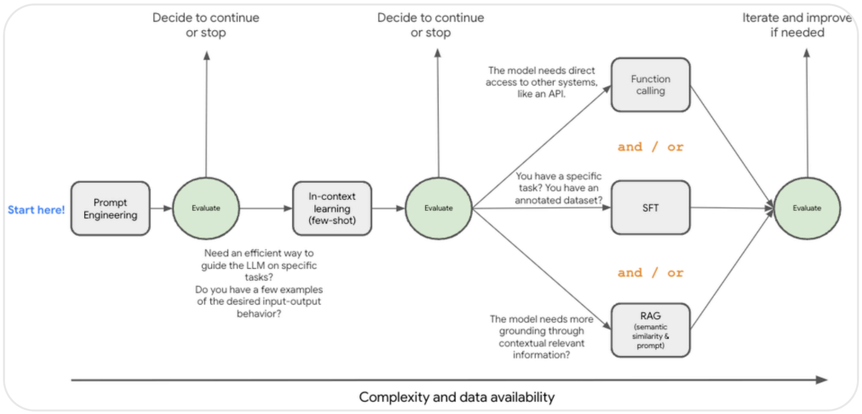
**Prompt Engineering** is easy to get started and for controlling outputs and It’s low-cost and accessible. It might be less reliable for handling complex or nuanced tasks and it requires expertise and experimentation.

**In-Context Learning (ICL)** leverages examples within the prompt to guide the LLM's behavior and just like prompt engineering, it is easy to get started with. ICL, which is also referred to as few-shot prompting, can be sensitive to the examples provided in the prompt, and the order in which they are provided. Additionally, it may not generalize well.

**Retrieval Augmented Generation (RAG)** retrieves relevant information, from Google search or other sources, and provides it to the LLM to improve quality and accuracy.  This requires a good knowledge base and the additional step adds complexity and latency.

**Function Calling** is the capability of a language model to identify the need for external systems to fulfill a user request and generate structured function calls to interact with these tools, enhancing its functionality. When used, it can  add additional latency and complexity.

Now you might be wondering what is the right path? It's important to understand that the optimal path depends on your unique needs, resources, and objectives for your use case. These techniques are not mutually exclusive and are often combined. Let's explore a framework to that can help guide your decision:


Decide to continue or stop
Decide to continue or stop
Iterate and improve if needed
Start here!
Prompt Engineering
Evaluate
In-context learning (few-shot)
Evaluate
Need an efficient way to guide the LLM on specific tasks? Do you have a few examples of the desired input-output behavior?
The model needs direct access to other systems, like an API.
Function calling
and / or
You have a specific task? You have an annotated dataset?
SFT
and / or
The model needs more grounding through contextual relevant information?
RAG (semantic similarity & prompt)
Evaluate
Complexity and data availability

You can start by exploring prompt engineering and few-shot in-context learning to see if the model can grasp the nuances of your specific domain. Gemini's large context window offers all kinds of possibilities here. Once you've honed your prompting strategy, you can venture into Retrieval Augmented Generation (RAG) and/or Supervised Fine-Tuning (SFT) for further refinement. This diagram represents some of the latest techniques, but the field of generative AI is changing quickly and is likely to evolve.

with Gemini on Vertex AI

Supervised Fine-Tuning (SFT) is ideal when you have a specific task in mind and possess labeled data to guide the model. SFT can also lead to more efficient models, potentially lowering costs and speeding up response times, and it can be used in combination with other techniques you may already be trying.

Ready to get started? Dive into our Generative AI repository and explore notebooks like our how to use supervised fine tuning. Experience the transformative potential of SFT on Vertex AI, and tailor your AI applications for peak performance and customization.

Want to fine-tune a Gemini model? Head over to the Vertex AI documentation to see which ones you can customize

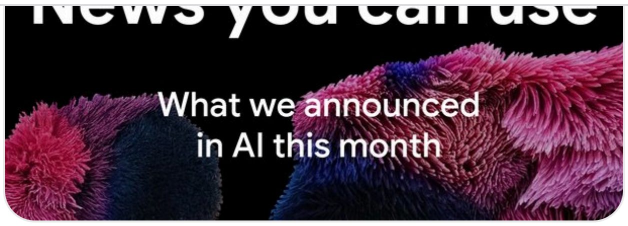
---

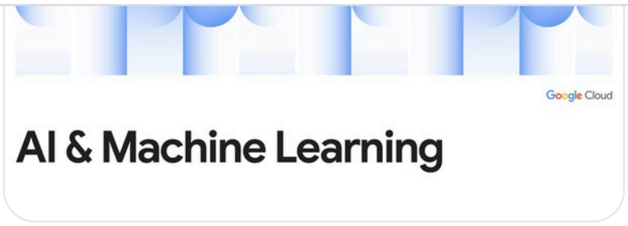
*A special thanks to Bethany Wang, Mikhail Chrestkha, Christos Aniftos and Elia Secchi from Google Cloud for their contributions.*

---

Posted in AI & Machine Learning

## Related articles





AI & Machine Learning

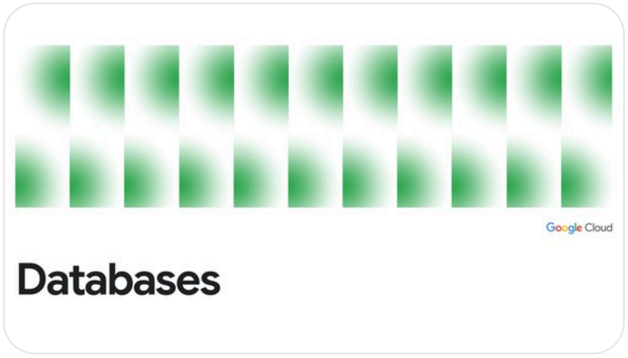
News you can use: What we announced in AI this month

By Google Cloud Content & Editorial • 7-minute read

AI & Machine Learning

Anyscale powers AI compute for any workload using Google Compute Engine

By Matthew Connor • 8-minute read



AI & Machine Learning

Formula E's AI equation: New Driver Agent boosts next generation of racers

By Olly Grundy • 5-minute read

Databases

Nuro drives autonomous innovation with AlloyDB for PostgreSQL

By Fei Meng • 5-minute read

Follow us