Tailoring LLMs to Your Use Case

Christopher Pang, Senior Solutions Engineer | 17 November 2023

Agenda

- LLMs in Context
- Tuning Hosted API LLMs
- Data Collection and Prep for Tuning
- Tuning Self-Managed LLMs

Why Bother Creating A Custom Model?

Motivations for Fine-Tuning

1. You just want the best use-case/task-specific results.

Why Bother Creating A Custom Model?

Motivations for Fine-Tuning

1. You just want the best use-case/task-specific results.

2. You want to save money and reduce latency.

Why Bother Creating A Custom Model?

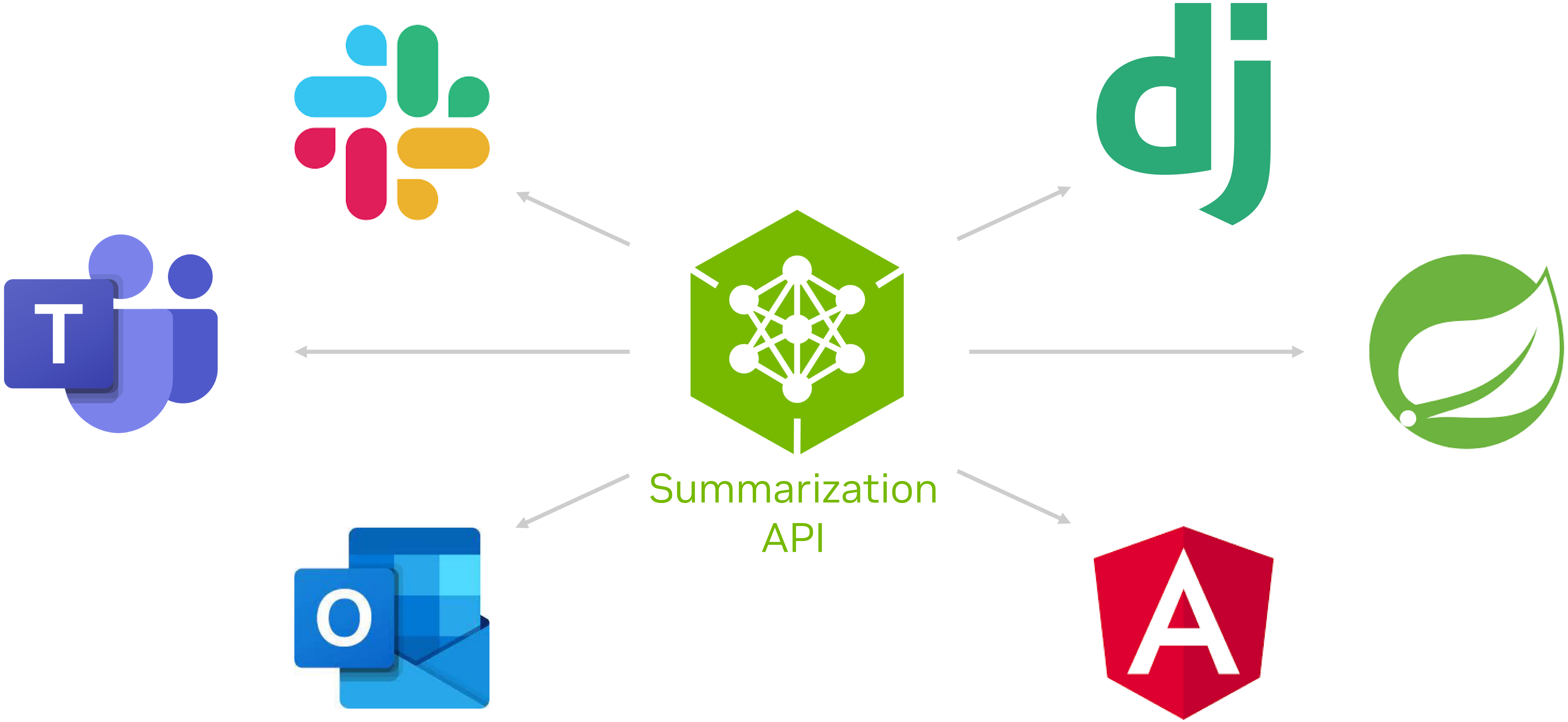
Motivations for Fine-Tuning

1. You just want the best use-case/task-specific results.
2. You want to save money and reduce latency.
3. You want a smaller model (fewer parameters) that was trained to imitate a larger one.

Example App 1: Domain-Tailored Summarization with Hosted LLM APIs

Domain-Tailored Summarization

Typical downstream clients only read from the API



Domain-Tailored Summarization

Our example application also writes new data back into the API

Domain-Tailored Summarization

Step 1: What do you want to summarize?

You can either choose to provide your own **Free Text**, extract text from a webpage by **Scraping a URL**, or select from a list of **NVIDIA TechBlog** articles to summarize.

Free Text | Scrape URL | NVIDIA TechBlog

Filter by Name...

Title	Categories	Date
Enabling Greater Patient-Specific Cardiovascular Care with AI Surrogates	Simulation / Modeling / Design	Thu, 11/9/2023, 7:16 PM
Accelerating Neurosymbolic AI with RAPIDS and Prometheux Vadalog Parallel	Data Science; Recommenders / Personalization	Thu, 11/9/2023, 3:17 PM
Whole Human Brain Neuro-Mapping at Cellular Resolution on NVIDIA DGX	Computer Vision / Video Analytics; Content Creation / Rendering; Data Center / Cloud	Wed, 11/8/2023, 12:55 PM
Setting New Records at Data Center Scale Using NVIDIA H100 GPUs and NVIDIA Quantum-2 InfiniBand	Data Center / Cloud; Generative AI / LLMs; Simulation / Modeling / Design	Wed, 11/8/2023, 12:00 PM

3531 result(s)

Selected Article:

Controls

Use HTML Headings to Chunk Text

Step 2: Which models do you want to use?

Model 1

NeMo GPT20B, Nev

Remove

Source: NVIDIA NeMo LLM Service

Description: 20 billion parameter NeMo GPT model, customized on the xsum dataset. One of the default customizations provided by NeMo.

Temperature	Top K	Top P
1	1	1

Repetition Penalty	Random Seed
1.1	0

How It Works

Domain-Tailored Summarization Under the Hood

Tune LLM on Your Data

Base and Custom LLM Output Predictions

Humans Comparing Outputs Side-by-Side

Generate New Data With Voting and Editing

Tuning Hosted API LLMs

Tune and Host an LLM Entirely Through an API

AI21 labs

Jurassic-2 Mid
Jurassic-2 Light

aws

Titan

cohere

Command

OpenAI

GPT 4
GPT 3.5
GPT 3

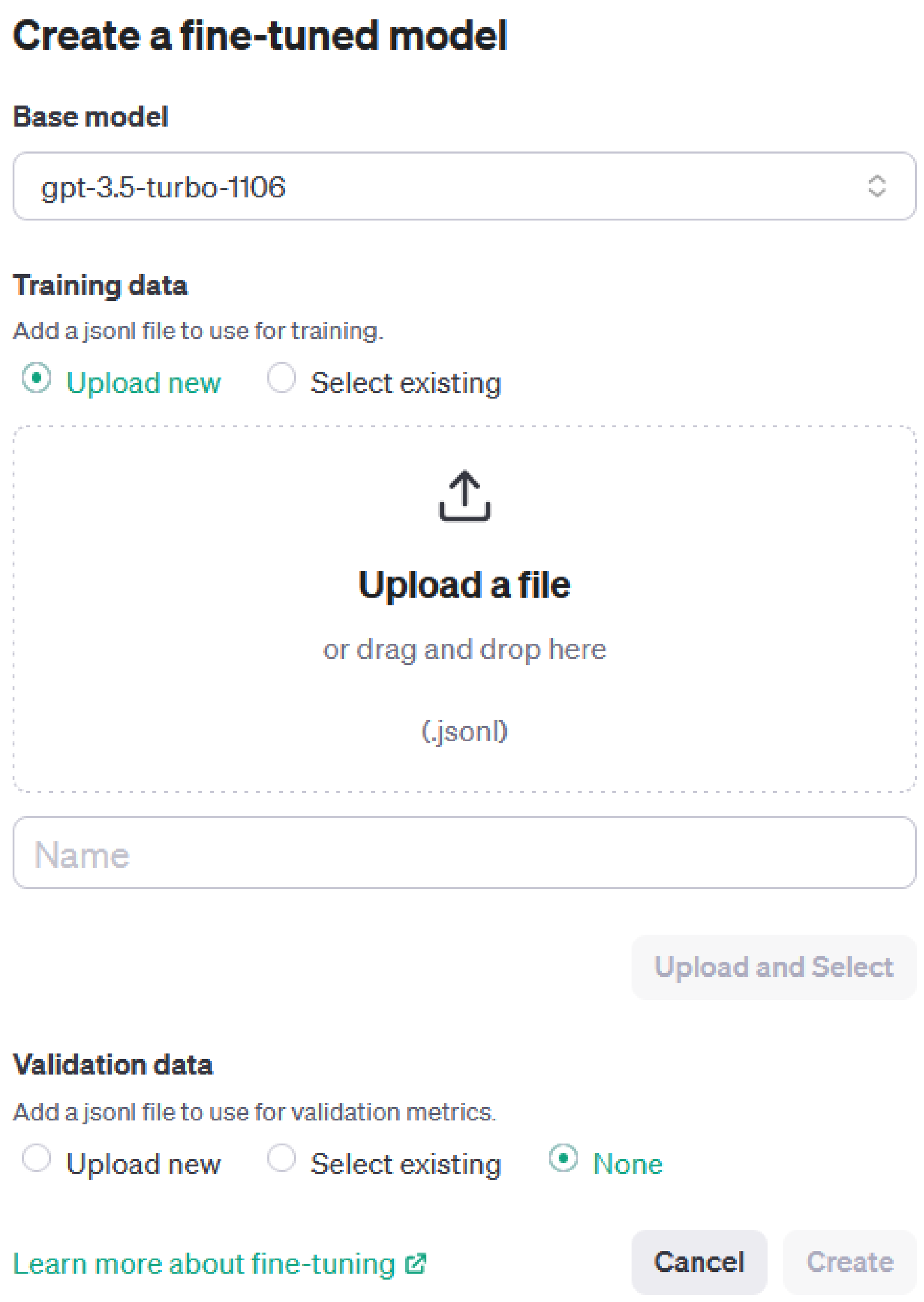
NVIDIA

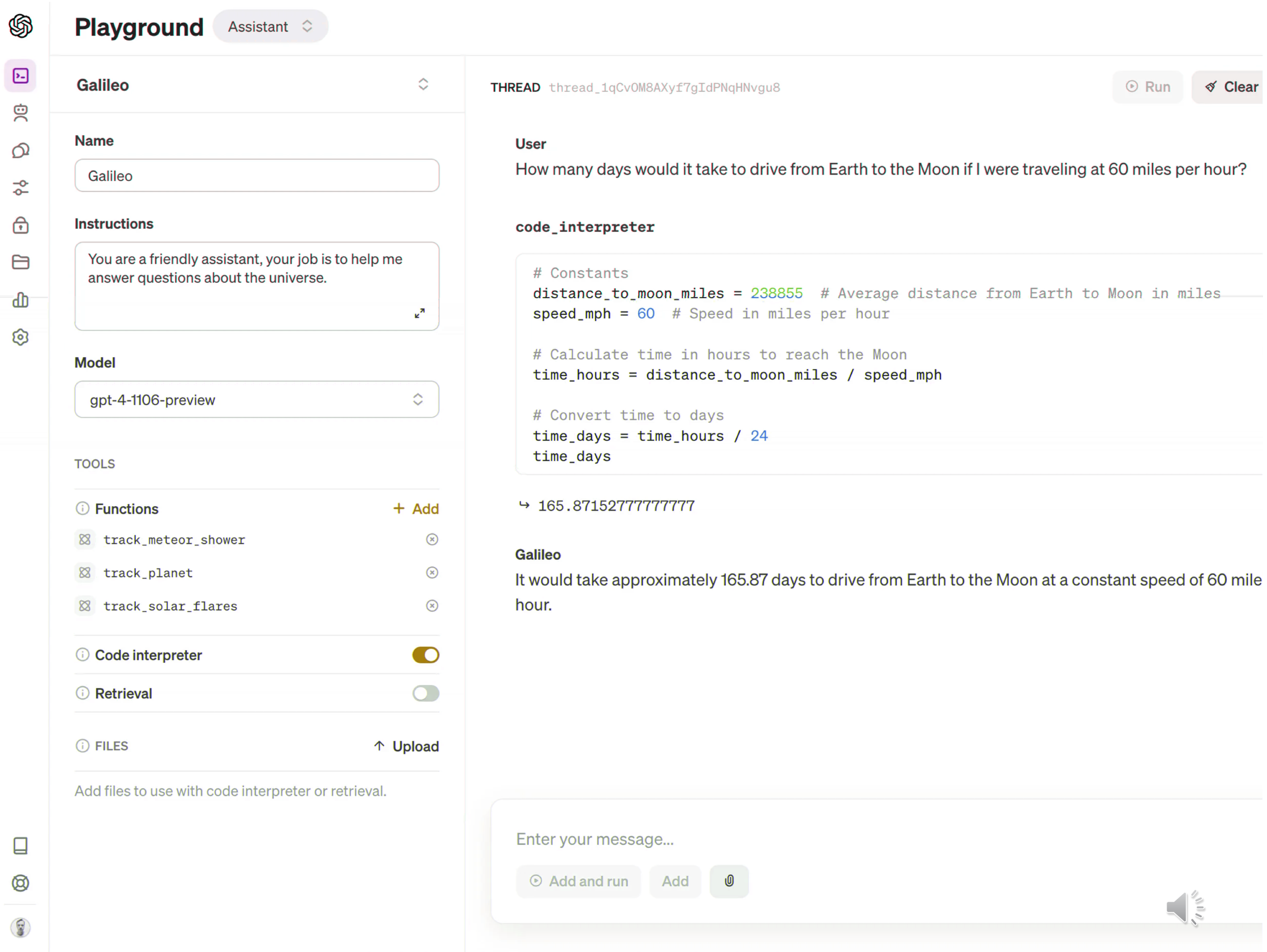
NeMo GPT
Llama 2

scale

Llama 2 7B and 70B
Mistral 7B
MPT 7B Instruct

OpenAI UI for Fine-Tuning and Assistants



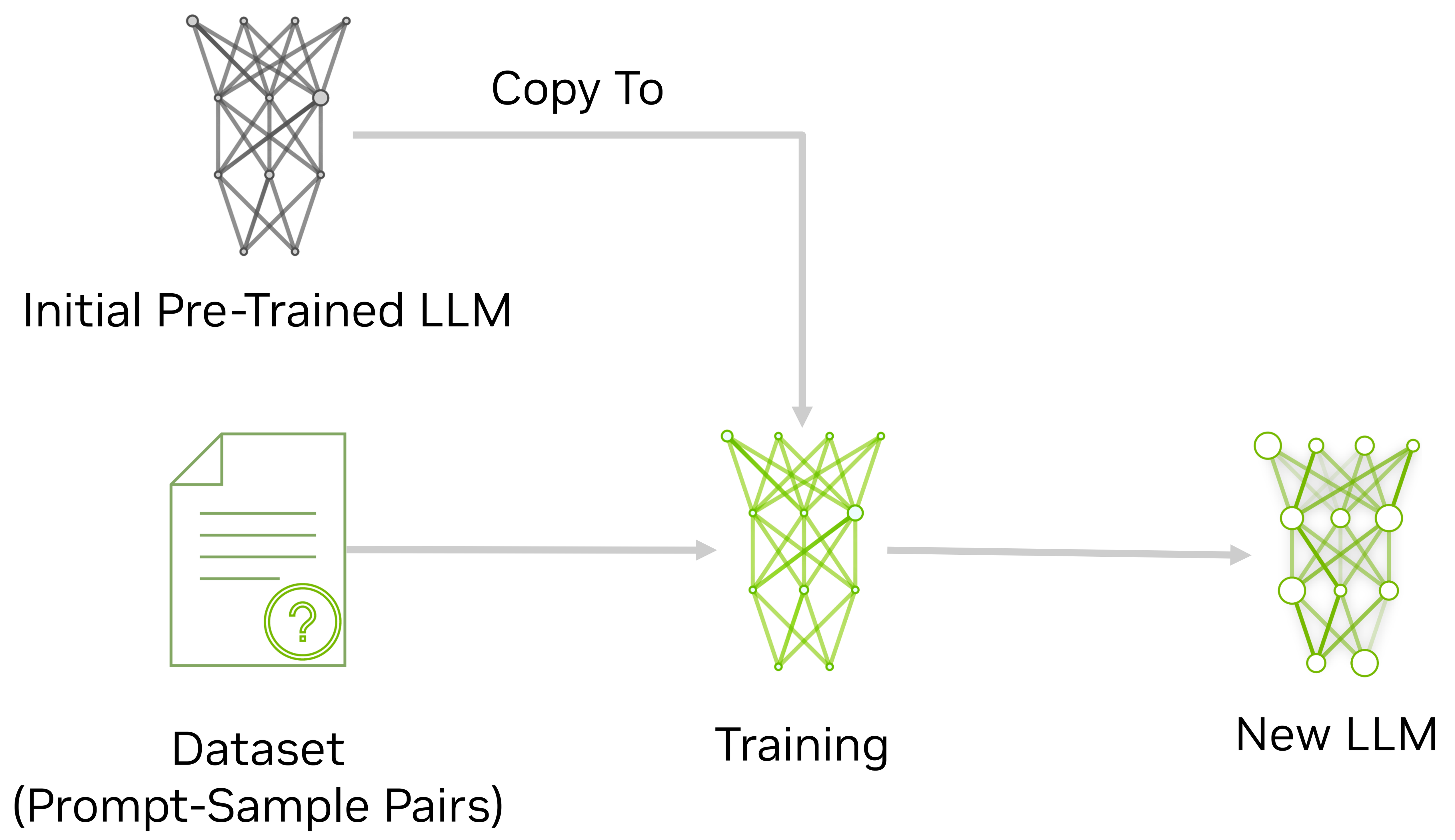


https://platform.openai.com/finetune

https://platform.openai.com/playground?mode=assistant

What is Fine-Tuning?

Traditionally means updating full parameters of the model with supervised learning

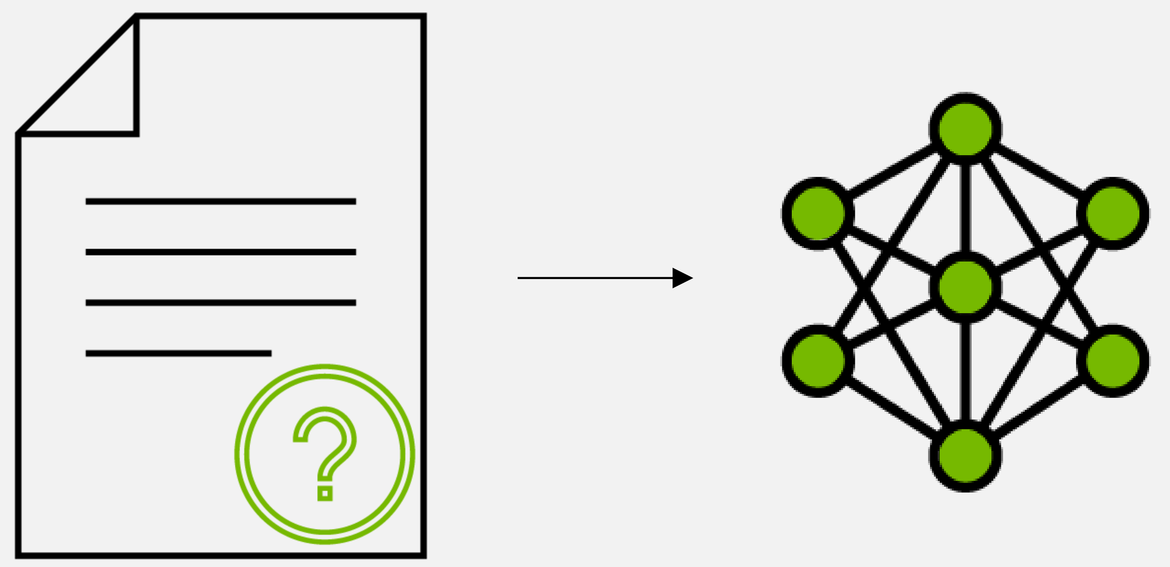


- Full-parameter fine-tuning re-trains a LLM with a prompt dataset in a supervised manner, requiring an update of <u>all model weights</u> per task
- Prompt dataset needs to be sufficiently large, on the order of thousands to hundreds of thousands of prompts

What is Fine-Tuning?

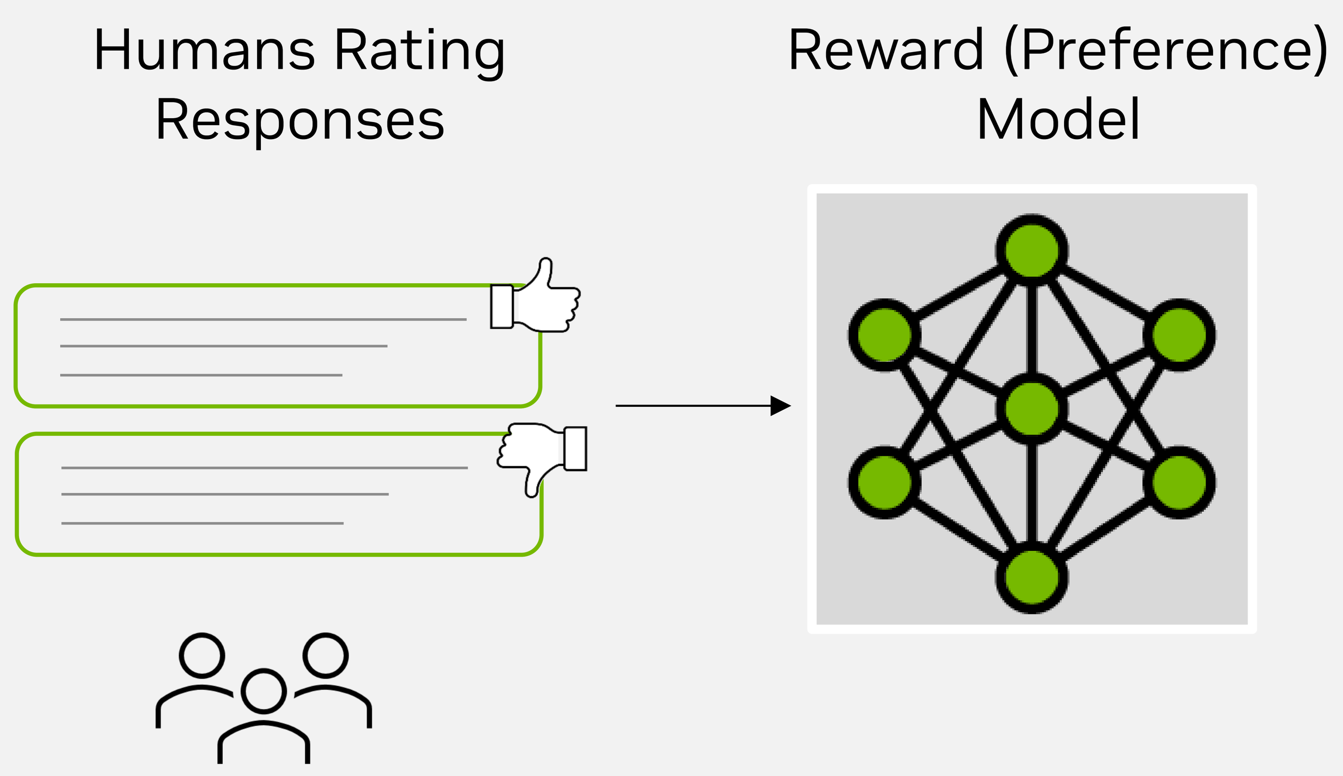
Also refers to alignment with human intent

1 Supervised Fine-Tuning of LLM



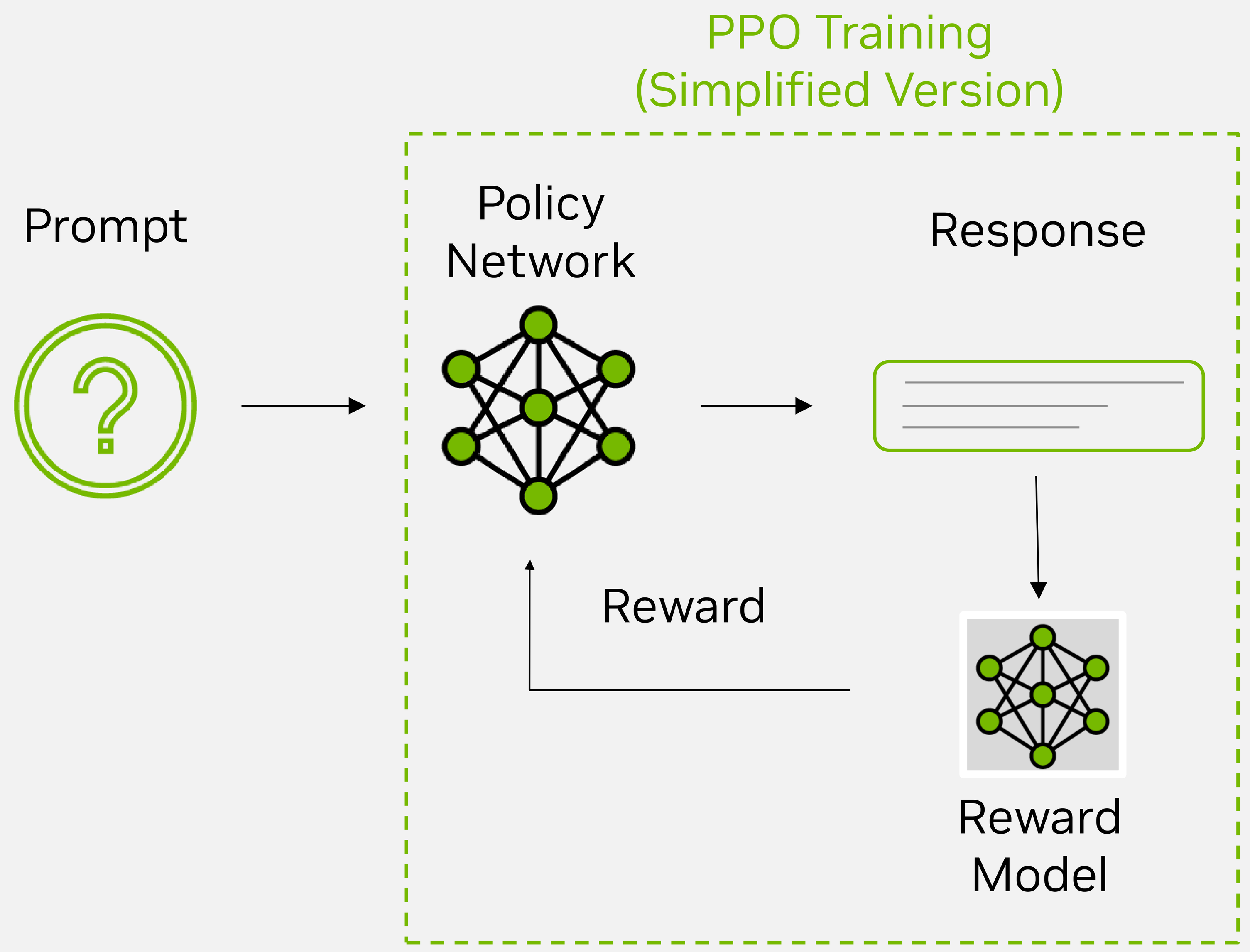
~10K-100K prompt-responses as input. Fine-tune LLM using prompt and responses.

2 Train Reward Model with Human Feedback



~100K-1M responses ranked and rated. Reward model: trained to mimic human feedback of model generated responses to prompts.

3 Reinforcement Learning Pipeline with Human Feedback

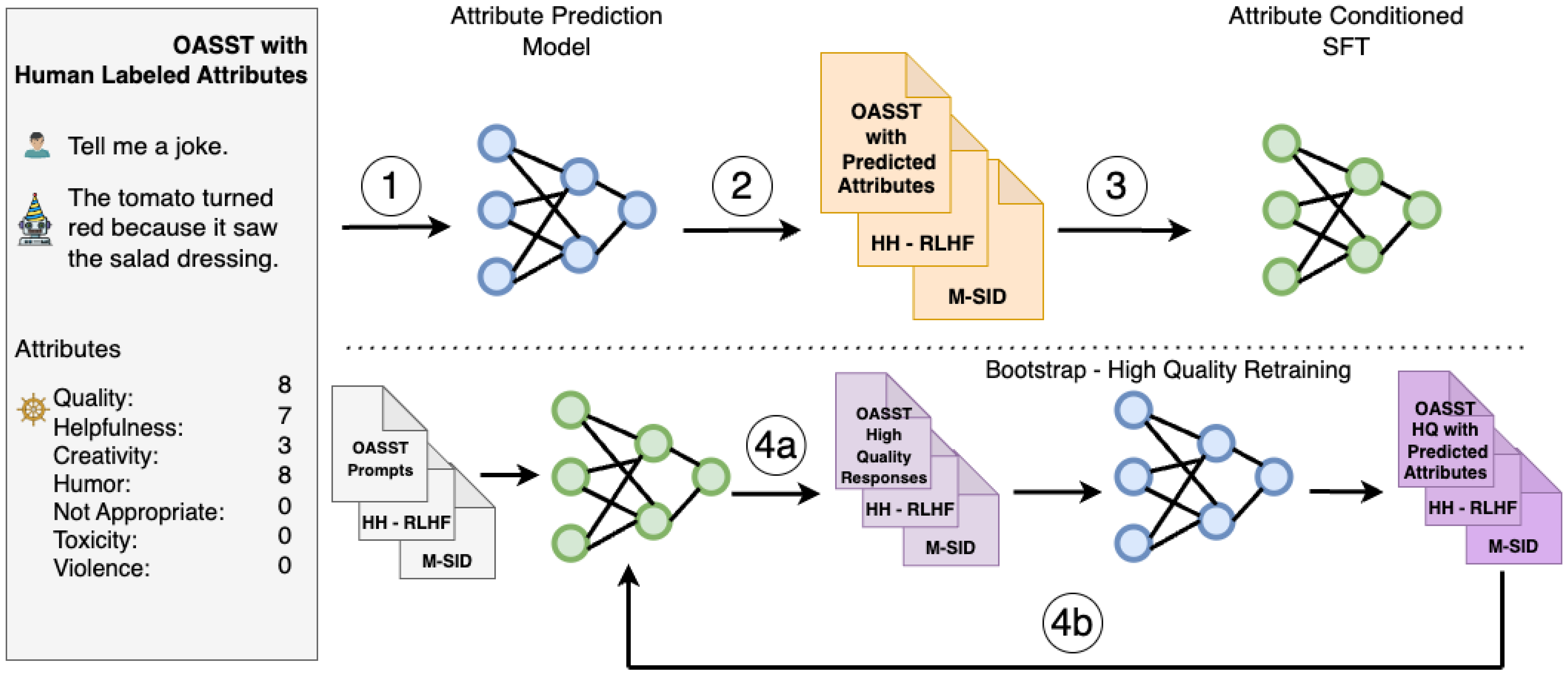


Build pipeline with RLHF to continuously improve model over time. Multiple neural networks interacting.

For more details (from 🤗): https://huggingface.co/blog/rlhf

Alignment with Human Intent: SteerLM

A technique to customize LLMs during inference

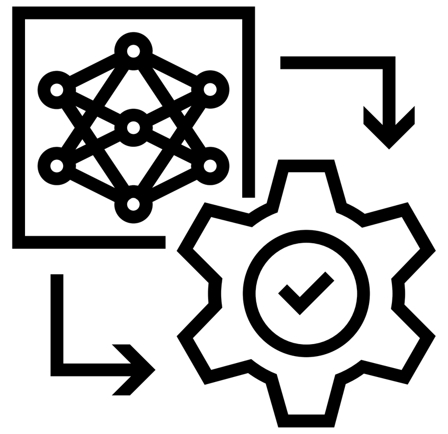


1. Train a prediction model on human-annotated datasets to evaluate response quality on any number of attributes like helpfulness, humor, and creativity.
2. Annotate diverse datasets by predicting their attribute scores to enrich the diversity of data available to the model.
3. Fine-tune by training the LLM to generate responses conditioned on specified combinations of attributes, like user-perceived quality and helpfulness.
4. Bootstrap training through model sampling by generating diverse responses conditioned on maximum quality, then fine-tuning on them to further improve alignment

https://huggingface.co/nvidia/SteerLM-llama2-13B
https://arxiv.org/abs/2310.05344

Latest Techniques for Customizing LLMs

Data, compute & time investment



FULL-PARAMETER FINE-TUNING

Accuracy for specific use-cases

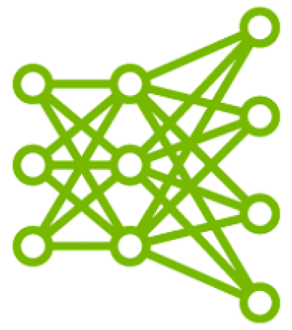
TECHNIQUES	• SFT • RLHF • SteerLM
HOW	**Tune LLM model weights**
TRAINING DATA	**Thousands** of examples & complex use cases
ADVANTAGE	**SFT is traditional method supported by all libraries. Robust for tuning to challenging domains (biomedical, coding, etc.)**

Prompt Engineering

Prompt design is crucial to obtaining good results from an LLM

Zero-Shot

Asking the foundation model to perform a task with no previous example

Q: “What is the capital of France?” →  → A: “Paris”

LLM

Lower token count
More space for context

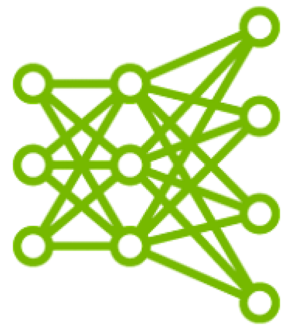
Few-Shot

Providing examples as context to the foundation model before giving it a task

Q: What is the capital of Spain?
A: {'answer': 'Madrid'}

Q: What is the capital of Italy?
A: {'answer': 'Rome'}

Q: What is the capital of France?

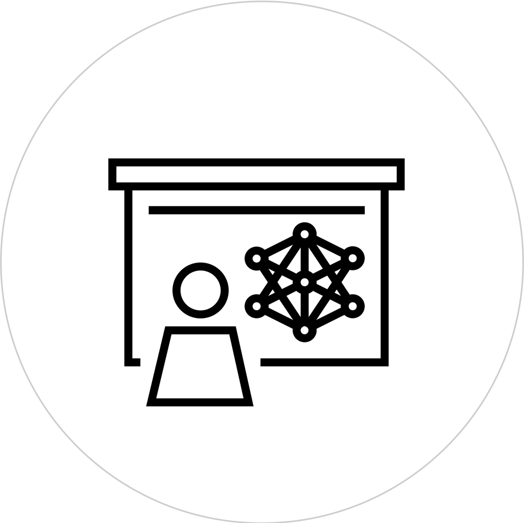
→  → A: {'answer': 'Paris'}

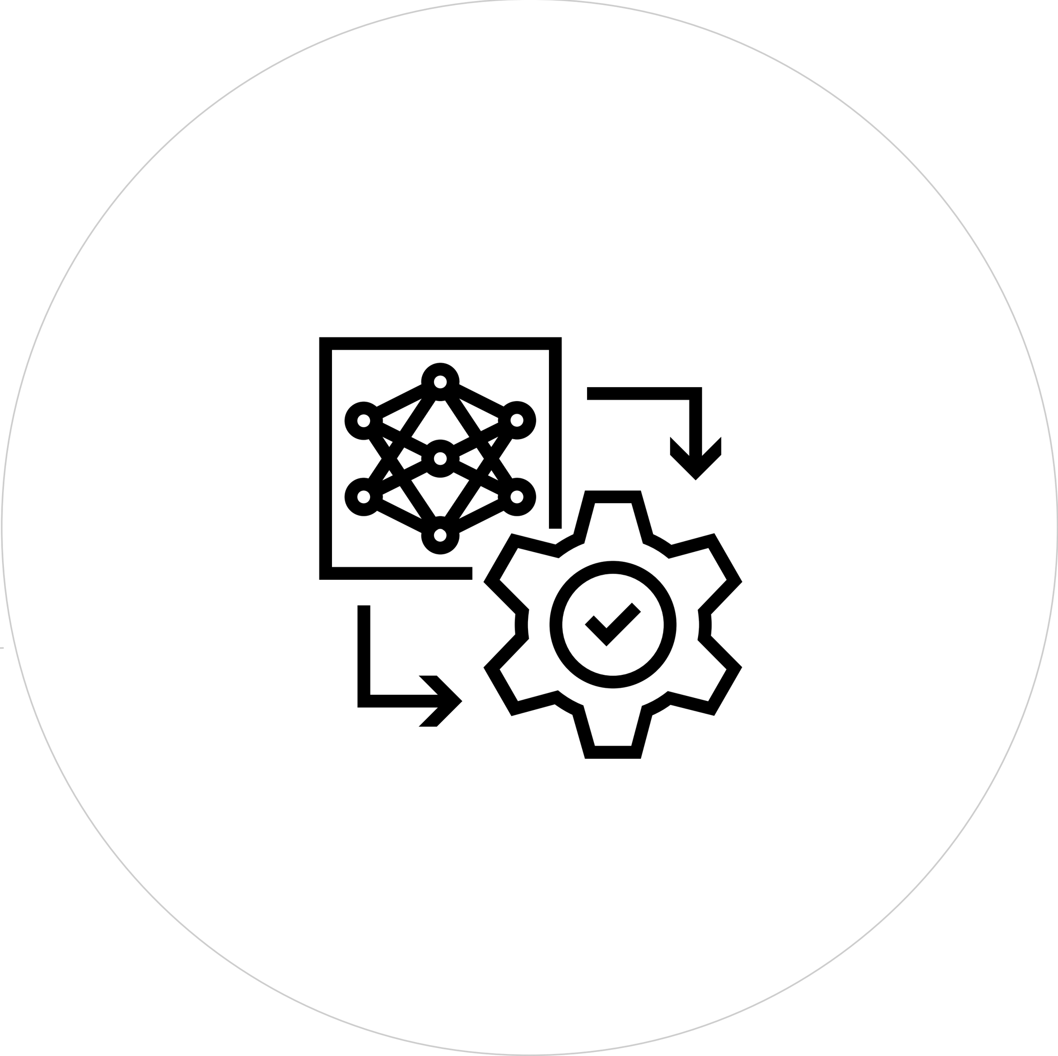
LLM

Better aligned responses
Higher accuracy on complex questions

Latest Techniques for Customizing LLMs

Data, compute & time investment





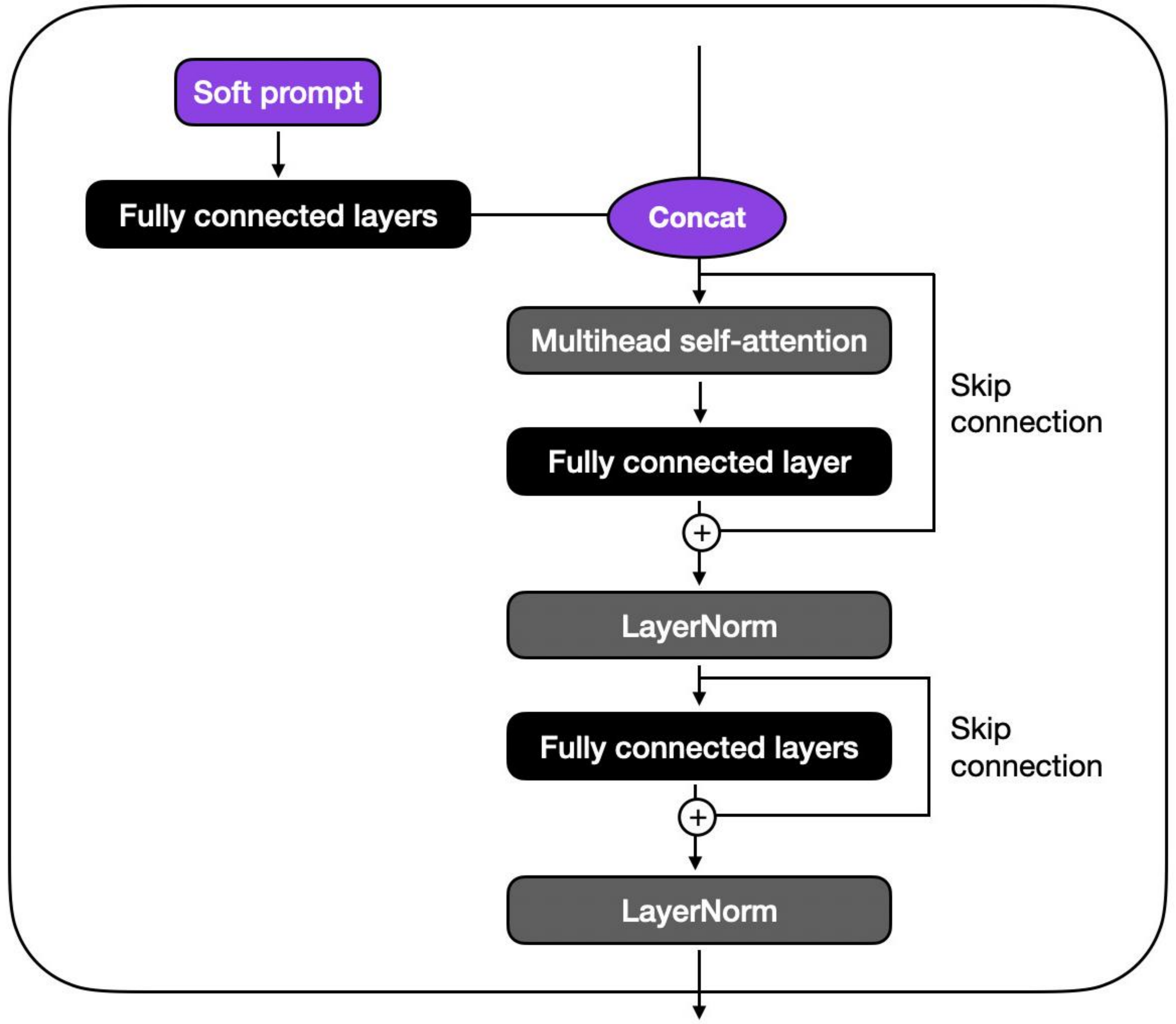
Accuracy for specific use-cases

	PROMPT ENGINEERING	**FULL-PARAMETER FINE-TUNING**
TECHNIQUES	• Few-shot / In-context learning • Chain-of-thought reasoning • System prompting	• SFT • RLHF • SteerLM
HOW	**Prompt templates**	**Tune LLM model weights**
TRAINING DATA	**Single-digit** number of prompt-completion examples & simple use cases	**Thousands** of examples & complex use cases
ADVANTAGE	**Minimal input of sample prompts – can be tuned online by end users**	**SFT is traditional method supported by all libraries. Robust for tuning to challenging domains (biomedical, coding, etc.)**

Prompt Learning

Comparison to Full-Parameter Fine-Tuning

TRANSFORMER BLOCK WITH PREFIX



Source: Lightning AI (Creators of PyTorch Lightning)

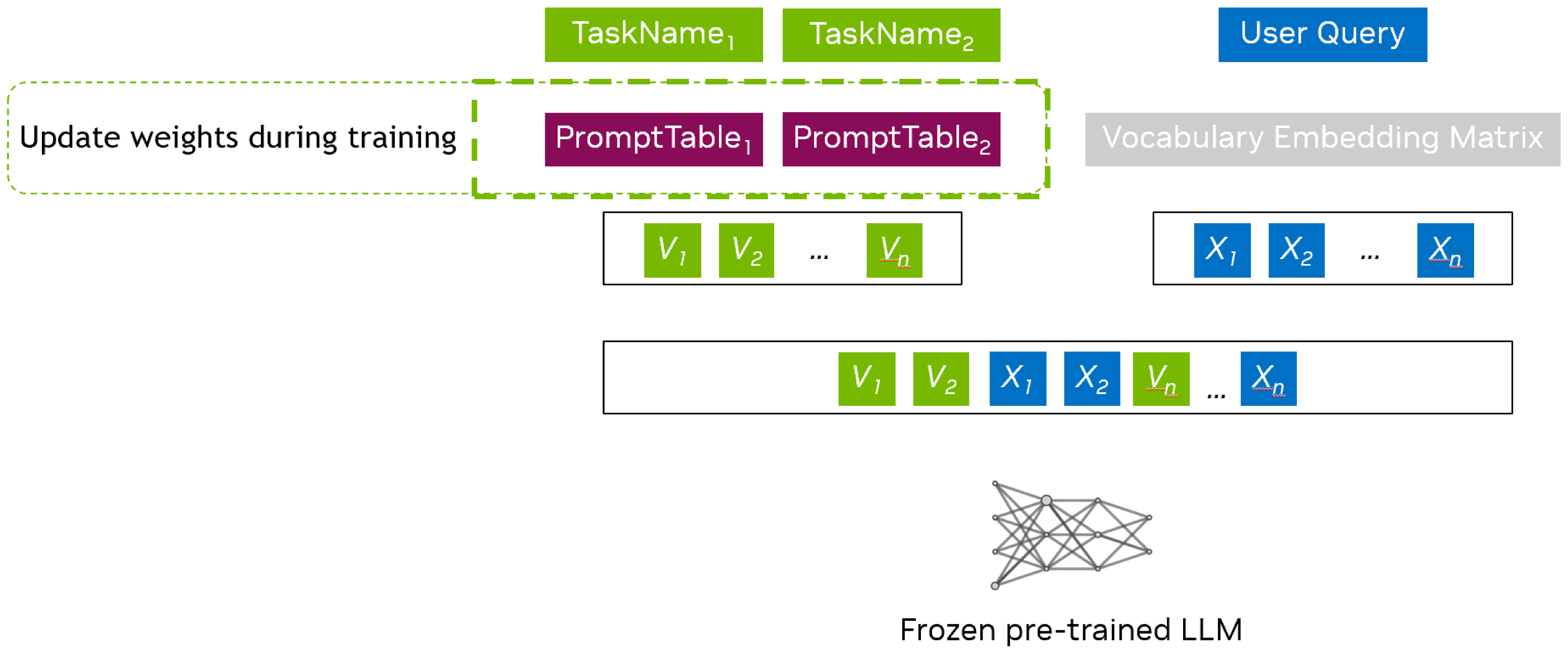
- Prompt learning adds a small number of trainable virtual tokens upstream of the LLM
- More efficient: for each new custom task, all we do is train those tokens
- The downstream foundation model is unchanged
- Often outperforms full-parameter fine-tuning when training data is small

Prompt Learning (Continued)

Prompt Tuning vs P-Tuning

Prompt Tuning

Fixed prompt of special tokens, where only embeddings can be updated.

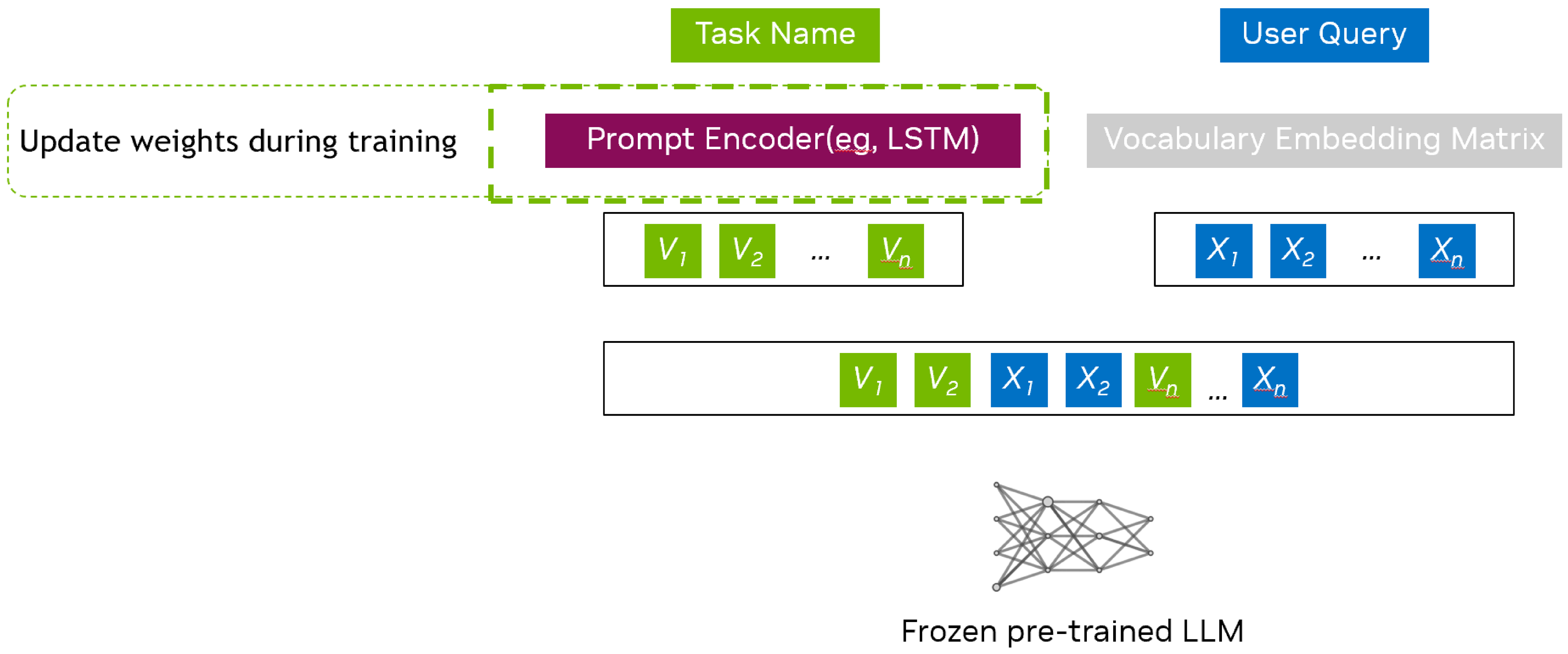


Fewer parameters to fine-tune.

Limited capacity to adapt to target task, but lower HW resource cost.

P-Tuning

A small LSTM (Long Short-Term Memory) model is used to predict embeddings of a fixed prompt of tokens.

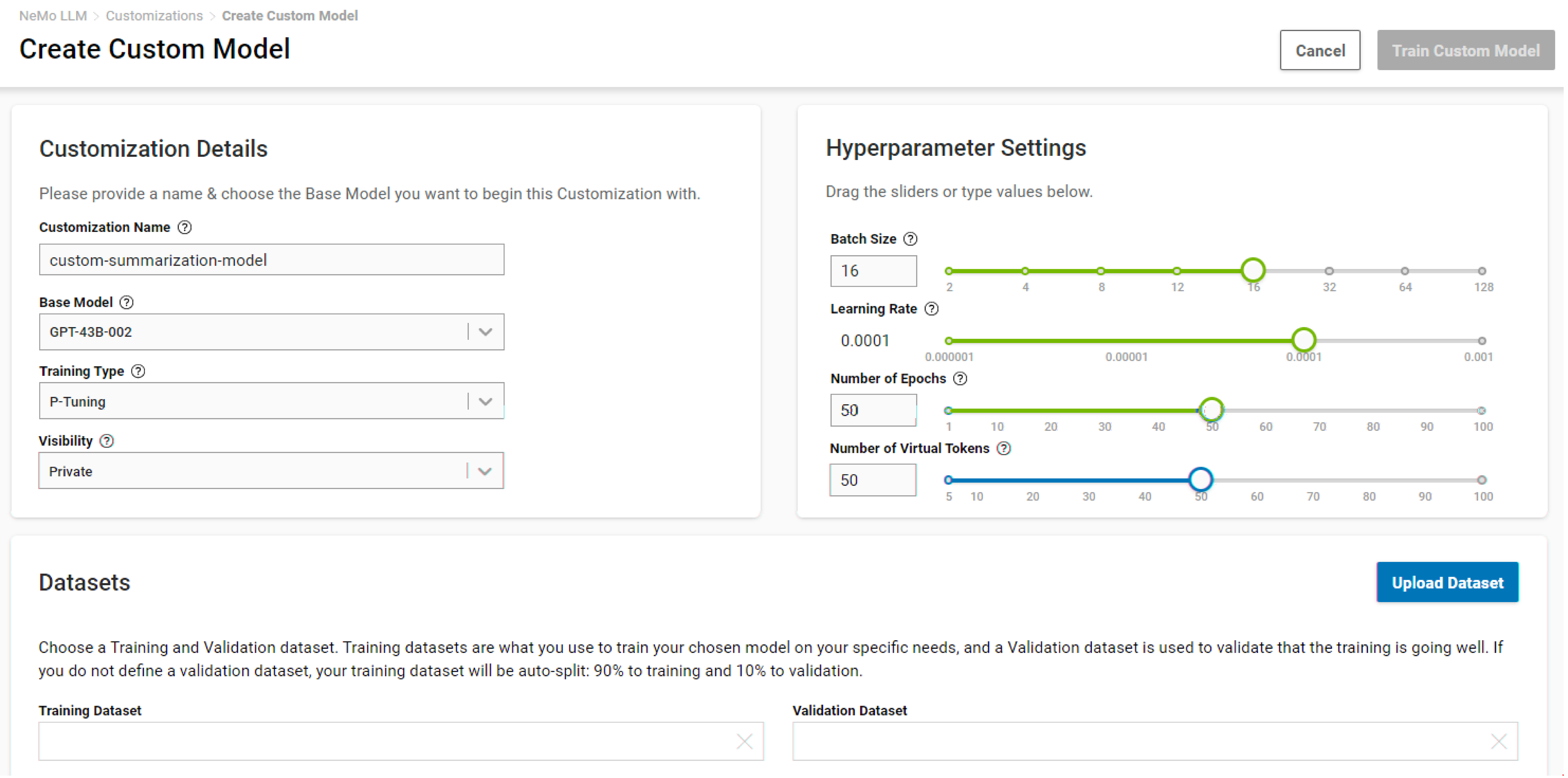


Requires more parameters to be tuned.

Higher accuracy at the cost of increased HW resources.

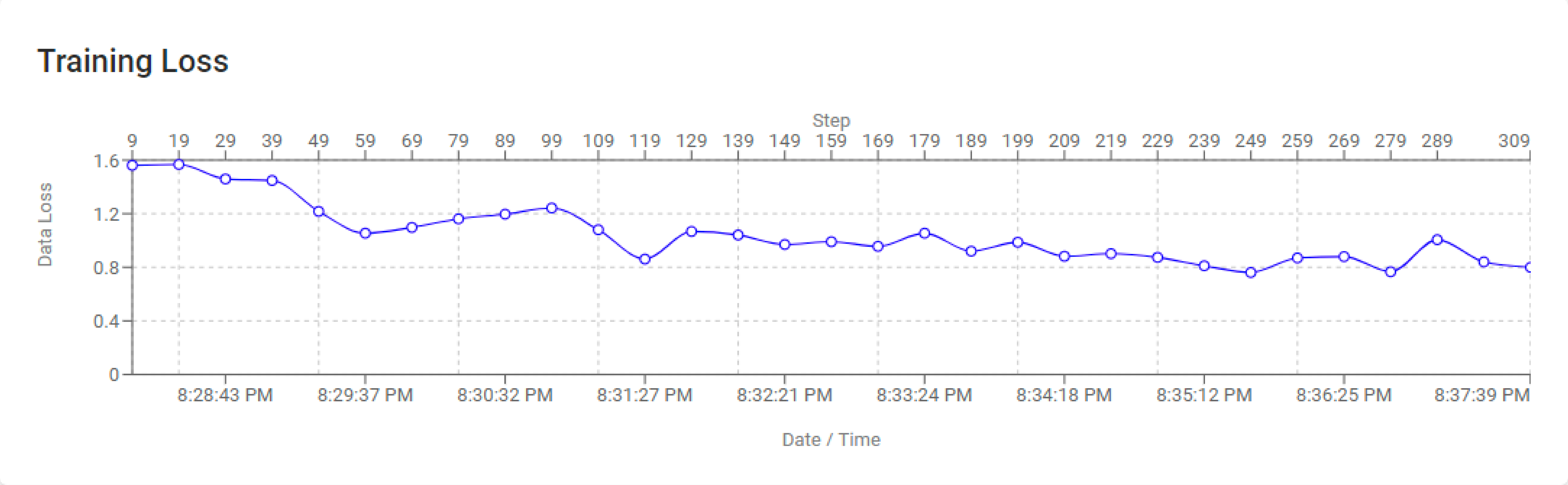
Example App 1: P-Tuning through NeMo LLM Service

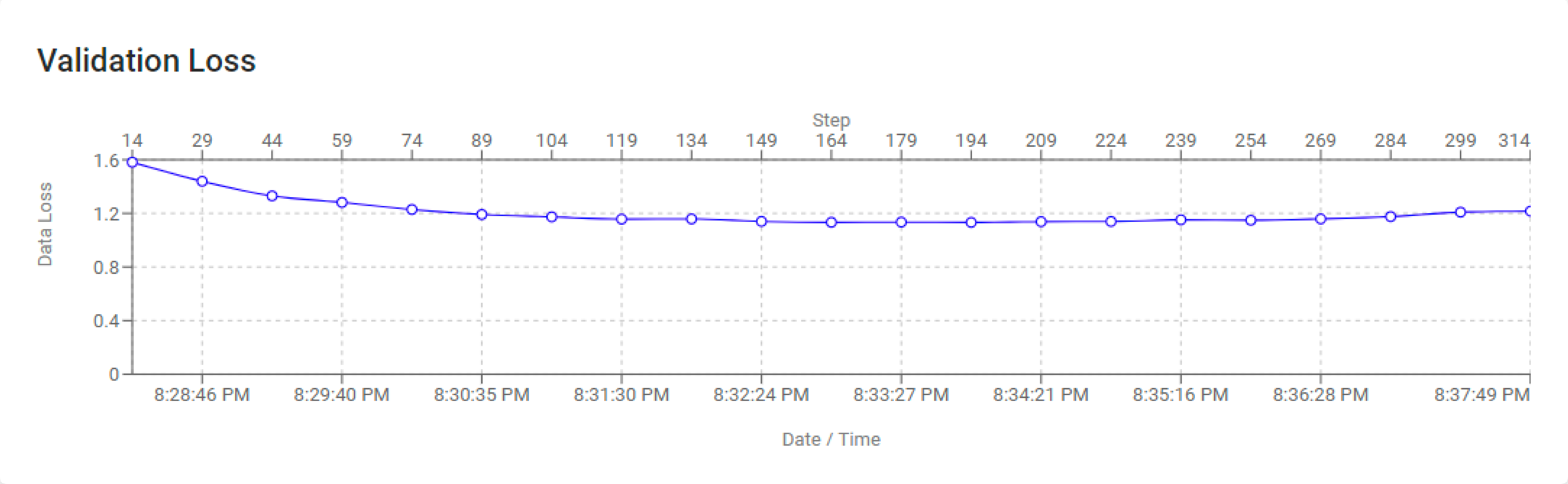
Web UI for Easy Model Customization



Example App 1: P-Tuning through NeMo LLM Service

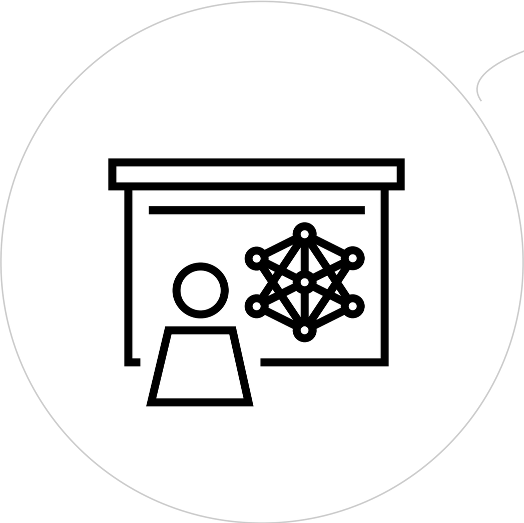
Loss Curves

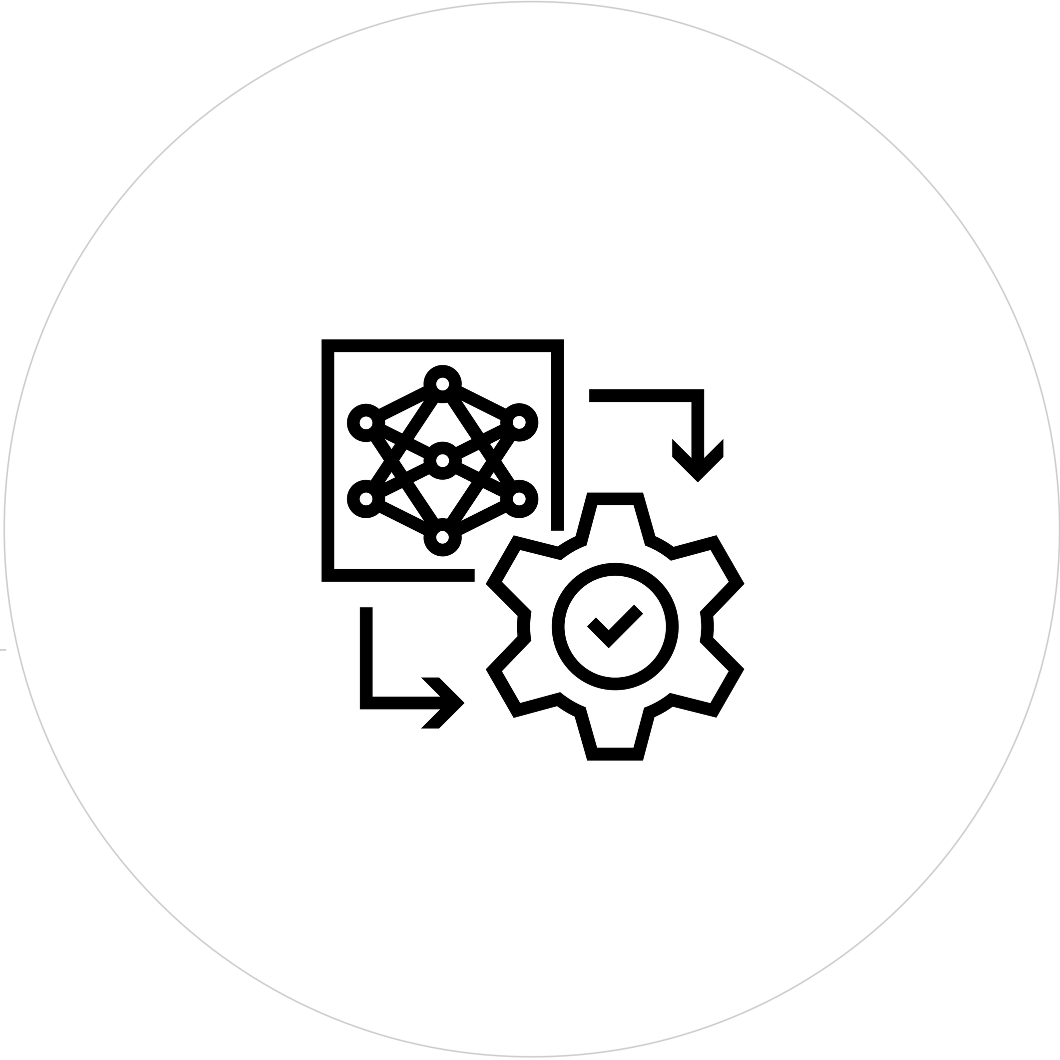




Latest Techniques for Customizing LLMs

Data, compute & time investment





PROMPT ENGINEERING | **PROMPT LEARNING** | **FULL-PARAMETER FINE-TUNING**

Accuracy for specific use-cases

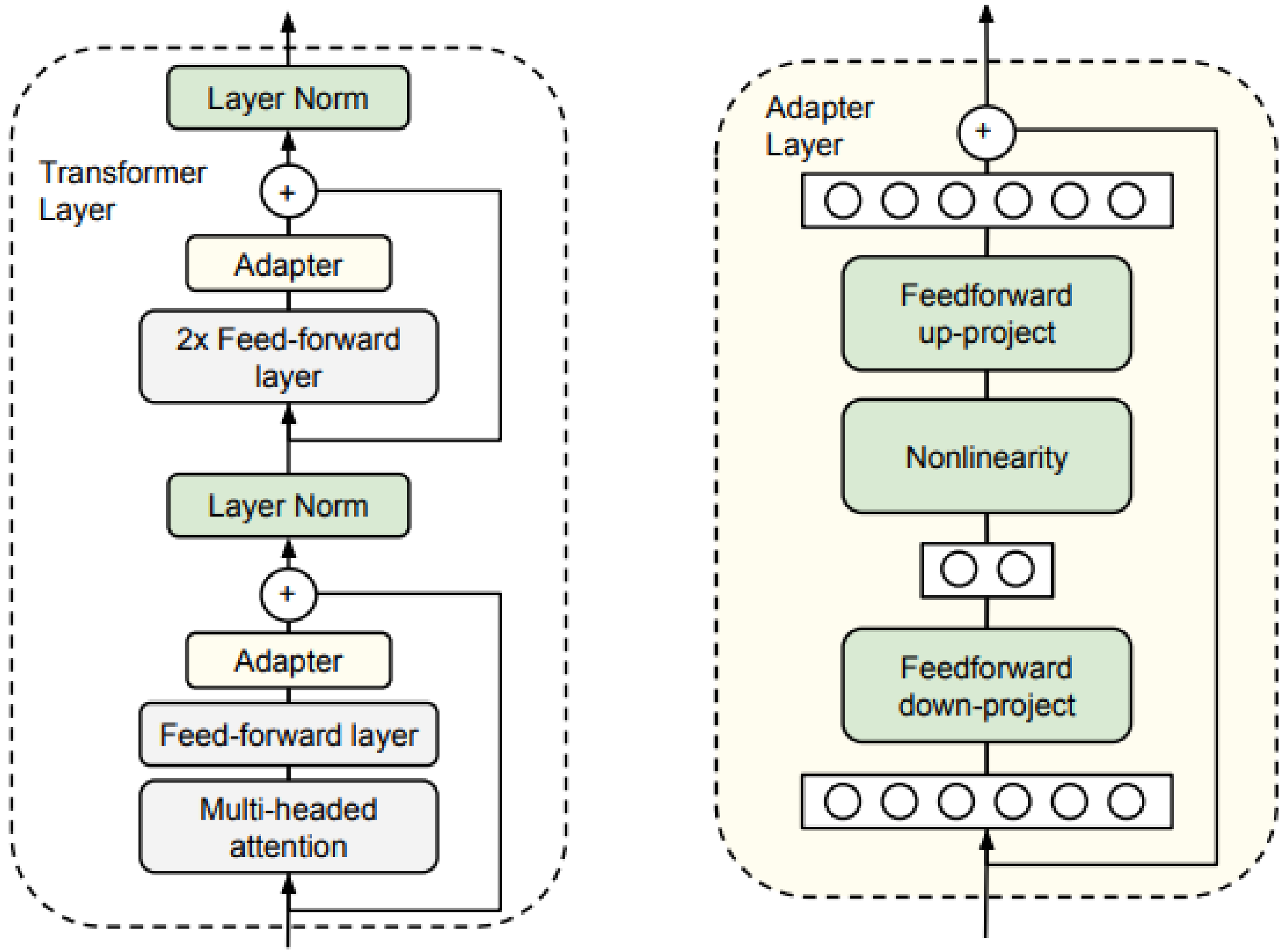
	PROMPT ENGINEERING	PROMPT LEARNING	FULL-PARAMETER FINE-TUNING
TECHNIQUES	• Few-shot / In-context learning • Chain-of-thought reasoning • System prompting	• Prompt tuning • P-tuning	• SFT • RLHF • SteerLM
HOW	**Prompt templates**	**Tune companion model**	**Tune LLM model weights**
TRAINING DATA	**Single-digit** number of prompt-completion examples & simple use cases	**A few hundred examples** & use cases where prompt engineering is not sufficient	**Thousands** of examples & complex use cases
ADVANTAGE	**Minimal input of sample prompts – can be tuned online by end users**	**Fast customization – only tuning a small model per task – downstream foundation model is unchanged**	**SFT is traditional method supported by all libraries. Robust for tuning to challenging domains (biomedical, coding, etc.)**

Adapter-Based Techniques

Adapters, LoRA, IA3

Adapters

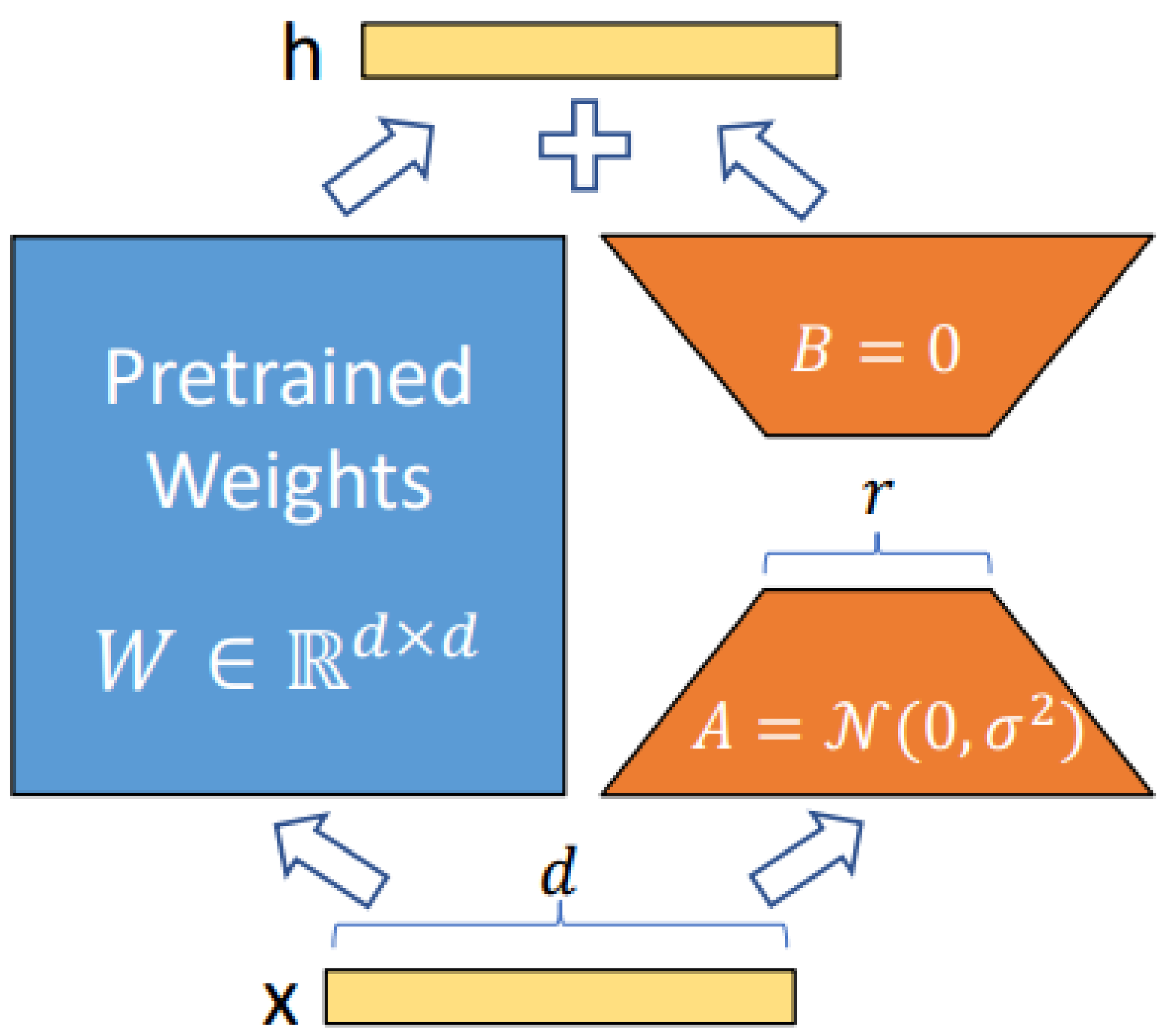
Insert into each transformer layer, only update weights of adapters



https://arxiv.org/abs/1902.00751

Low-Rank Adaptation (LoRA)

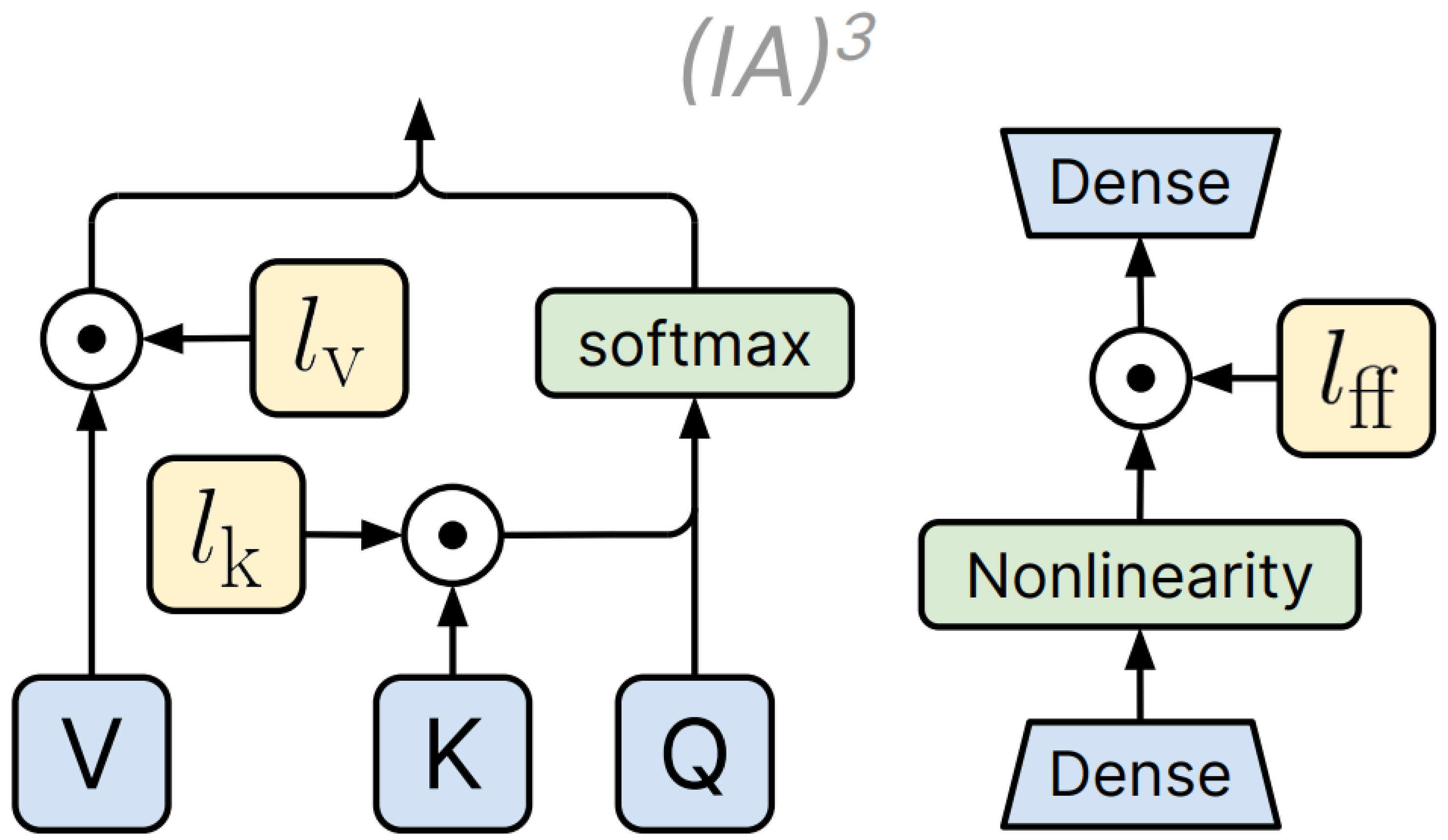
Optimize rank decomposition matrices of dense layers



https://arxiv.org/abs/2106.09685

IA3

Like adapters, but where each adapter is a vector that scales key, value or ffn



https://arxiv.org/abs/2205.05638

Latest Techniques for Customizing LLMs

	PROMPT ENGINEERING	PROMPT LEARNING	ADAPTERS	FULL-PARAMETER FINE-TUNING
TECHNIQUES	• Few-shot / In-context learning • Chain-of-thought reasoning • System prompting	• Prompt tuning • P-tuning	• Adapters • LoRA • IA3	• SFT • RLHF • SteerLM
HOW	**Prompt templates**	**Tune companion model**	**Add custom layers to LLM**	**Tune LLM model weights**
TRAINING DATA	**Single-digit** number of prompt-completion examples & simple use cases	**A few hundred examples** & use cases where prompt engineering is not sufficient	**Hundreds** of examples for a multitude of downstream tasks	**Thousands** of examples & complex use cases
ADVANTAGE	**Minimal input of sample prompts – can be tuned online by end users**	**Fast customization – only tuning a small model per task – downstream foundation model is unchanged**	**Achieve higher accuracy while requiring less samples than traditional fine-tuning**	**SFT is traditional method supported by all libraries. Robust for tuning to challenging domains (biomedical, coding, etc.)**

Data Collection and Preparation For Tuning

Obtaining Datasets for Tuning

- Don't over-rely on public datasets. Datasets are everywhere. Need to curate input/output pairs.
- "Less is More for Alignment." High-quality, low-quantity training data vs. low-quality, high-quantity.
  - https://arxiv.org/abs/2305.11206
- Synthetic data generation: Use high-end model and complex prompt template to induce correct behavior/outputs from smaller model ("context distillation").

```
{“prompt”: “Summarize the following text:\nNVIDIA announced the release of…”,
 “completion”: “NVIDIA’s new product Guardrails…”}

{“prompt”: “Summarize the following text:\nWhen Jensen Huang first founded…”,
 “completion”: “NVIDIA’s CEO outlines vision for…”}

{“prompt”: “Summarize the following text:\nHi team,\n\nI noticed that Omni…”,
 “completion”: “Omniverse Replicator feature request…”}
…
```

Example App 2: Tailored RAG

Retrieval Augmented Generation (RAG)

Motivation

- Decouples an LLM from only being able to act on original training data
- Obviates the need to retrain the LLM with the latest data
- LLMs limited by context window sizes

Concept

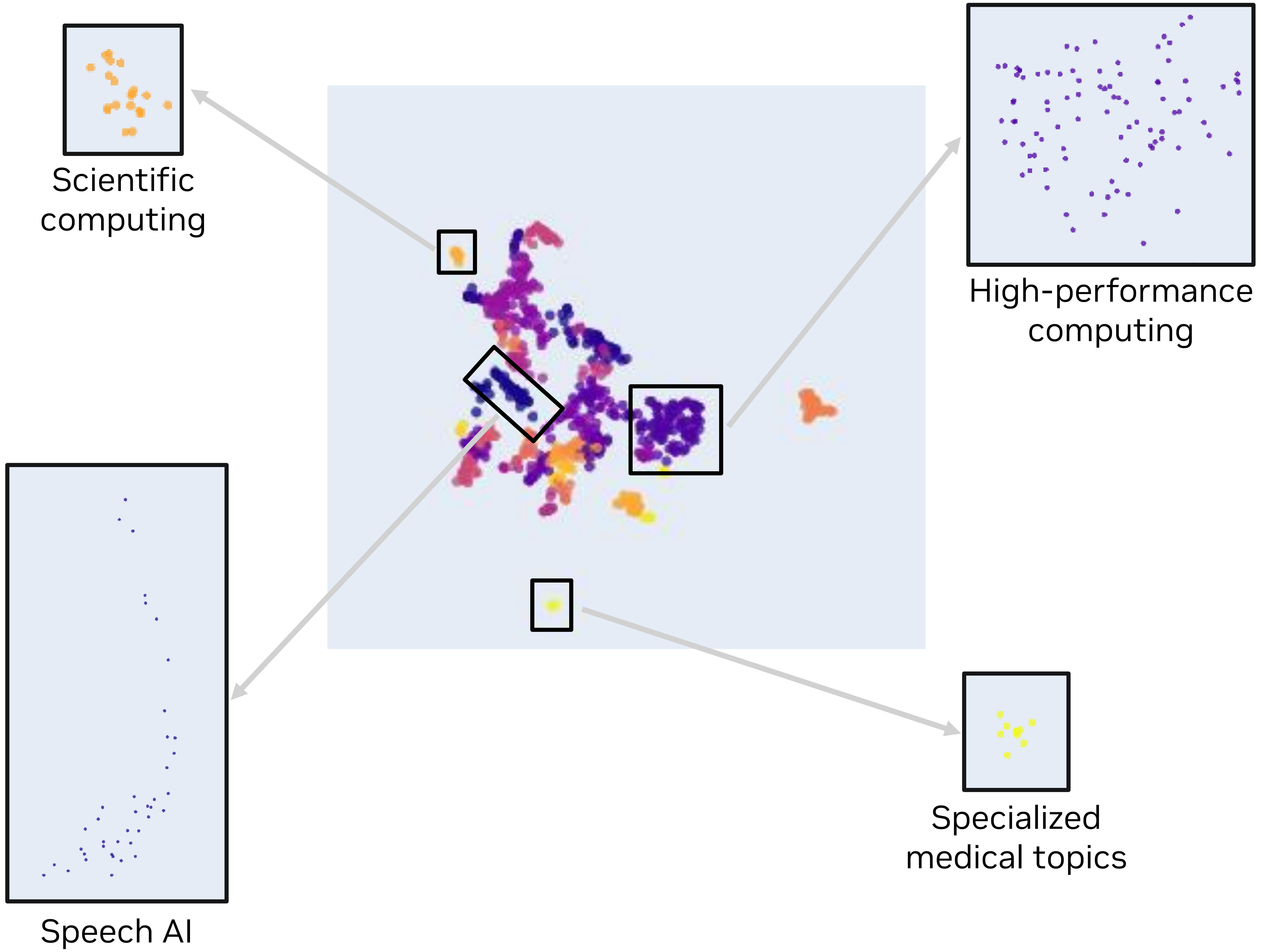
- Connect LLM to data sources at inference time
  - e.g., databases, web, documents, 3rd party APIs, etc.
- Find relevant data
- Inject relevant data into the prompt

Components of the Application

1. Human input (prompt)
2. Vectorization (embedding)
3. Retrieve vectors and calculate distance
4. Extract closest matching docs
5. Inject relevant docs into the prompt
6. Output becomes up-to-date, more accurate, with ability to cite source

Embeddings and the Vector Database

Searching via semantic similarity



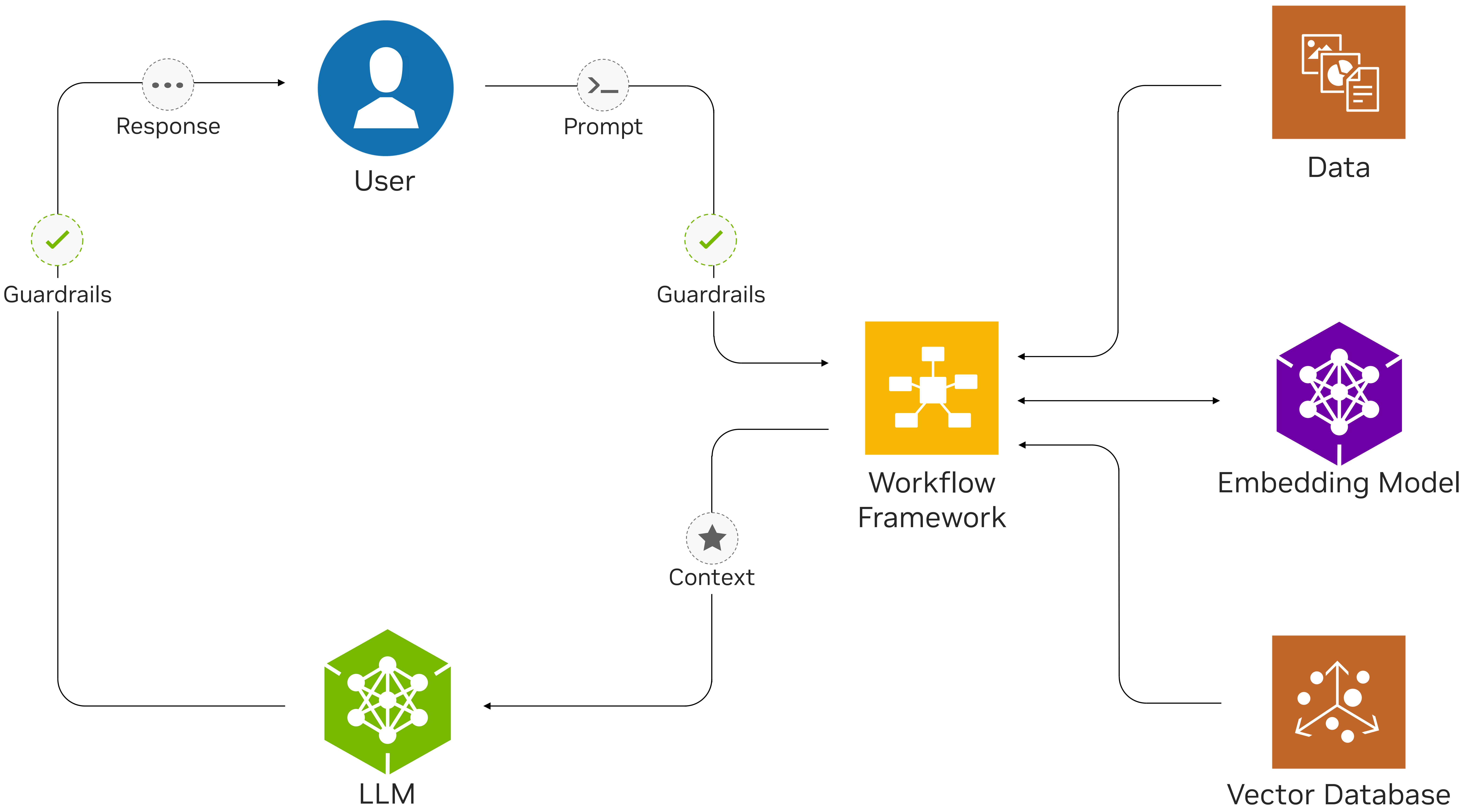
2D representation of a 768-dimension embedding space

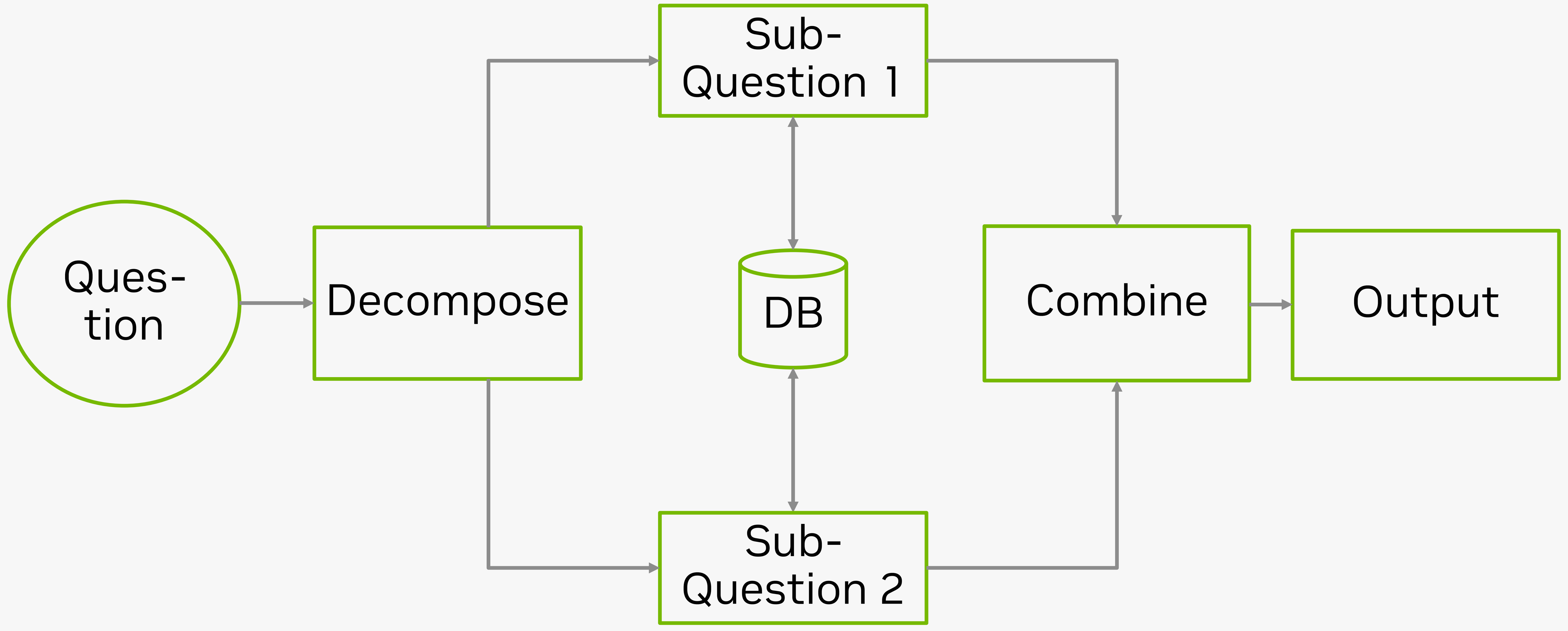
- Embeddings are data (text, image, or other data) represented as numerical vectors
  - Input text -> embedding model -> output vector
- Part of **semantic search**
  - Model trained to embed similar inputs close together
- Useful for: classification, clustering, topic discovery
- Many pretrained and trainable embedding model sources
  - Modern ones are often deep neural networks

Query: Who will lead the construction team?
Chunk 1: The construction team found lead in the paint.
Chunk 2: Ozzy has been picked to lead the group.

Chunk 1 shares more keywords with the query, but semantic search can differentiate the meanings of "lead" and understand that "team" and "group" are similar, so Chunk 2 may be more helpful for the query.

Canonical RAG Workflow





Question Decomposition

Making hard questions easier

- Retrieval augmented generation (RAG) can struggle out-of-the-box with complex prompts when retrieval fails to find the right documents.
- Solution: Tune a small question decomposition model.
- Decompose complex questions into easier sub-question with a single topic—makes retrieval more likely to succeed.
- Anthropic: “Question Decomposition Improves the Faithfulness of Model-Generated Reasoning”

Example App 2: Naive RAG

Retrieval Augmented Generation Question Answering

Try out the following questions in both Naive and Question Decomposition mode.

- Who are the founders of BonVoyage AI? (Answer: Alex Sanders and Julia Hopper)
- What is the current revenue of BonVoyage AI? (Answer: $185 million USD)
- What is the latest internal evaluation of DriveNouveau? (Answer: $900 million USD)
- What is the latest internal evaluation of BonVoyage AI's closest competitor? (Answer: $900 million USD)
- What are the latest internal evaluations of Voyage AI as well as its closest competitor DriveNouveau? (Answer: $250 million USD, $900 million USD)

Ask me a question that can only be answered from our internal documents.

Number of Recs

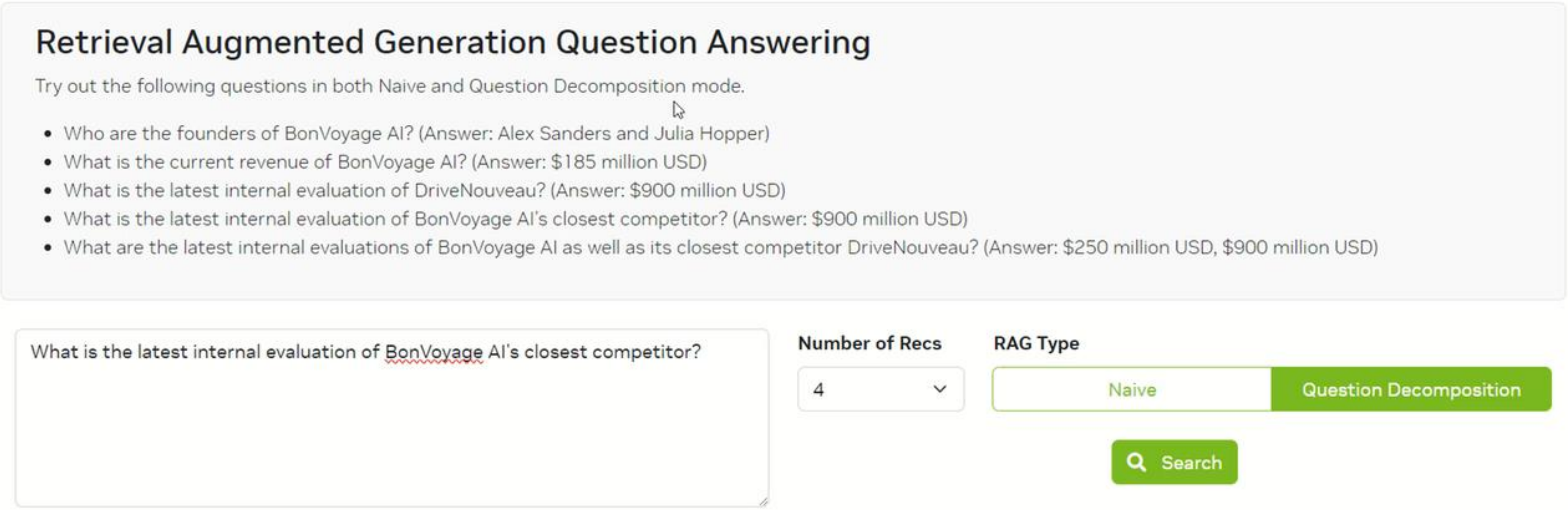
3

RAG Type

Naive | Question Decomposition

Search

Example App 2: Question Decomposition RAG



How It Works

Synthetic Data Generation and Context Distillation

OctoML

Acquire dataset of complex questions → Prompt Llama 2 70B with few-shot prompts → Manually verify results → Train Llama 2 7B with PEFT (LoRA) and FP4 to mimic Llama 2 70B → Prompt Llama 2 7B with zero-shot prompts. Hosted on OctoAI platform.

More Robust RAG

- Tuning a question decomposition model
- Tuning a QA model that explicitly only answers from context

More Robust RAG

- Tuning a question decomposition model
- Tuning a QA model that explicitly only answers from context
- Retrieval enhancements:
  - Tuning a custom embedding model
  - Re-ranker
  - Decoupling retrieval and generation chunks
  - Using document metadata/hierarchy

More Robust RAG

- Tuning a question decomposition model
- Tuning a QA model that explicitly only answers from context
- Retrieval enhancements:
  - Tuning a custom embedding model
  - Re-ranker
  - Decoupling retrieval and generation embeddings
  - Using document metadata/hierarchy
- Agents: using external tools (e.g., to answer a math question)

More Robust RAG

- Tuning a question decomposition model
- Tuning a QA model that explicitly only answers from context
- Retrieval enhancements:
  - Tuning a custom embedding model
  - Re-ranker
  - Decoupling retrieval and generation embeddings
  - Using document metadata/hierarchy
- Agents: using external tools (e.g., to answer a math question)
- Serving for inference

Tuning Self-Managed LLMs

Self-Managed vs. Hosted API

Self-Managed LLMs

Own & manage underlying model weights

Motivations:

- Privacy/Ownership
- Portability/Flexibility
- Cost: Run on own infrastructure
- Choice of customization

Examples for Getting Started: NeMo Framework, HuggingFace Hub + PEFT

Hosted API LLMs

Access only available through hosted APIs

Motivations:

- Easy to use: Push-button experiences
- Easy deployment: Don't have to worry about managing hardware and keeping your API healthy

Examples for Getting Started: OpenAI, Cohere, AWS Bedrock, NeMo LLM Service

NVIDIA NeMo Framework

From foundation model to application

Fine-Tuning in NeMo

An All-in-One Implementation

1. Set Parameter-Efficient Fine-Tuning Type
2. Set Hyperparameters

```
peft:
    peft_scheme: "adapter"  # can be either adapter, ia3, ptuning,
adapter_and_ptuning, or lora
    restore_from_path: null
```

Fine-Tuning in NeMo

An All-in-One Implementation

1. Set Parameter-Efficient Fine-Tuning Type
2. Set Hyperparameters
   a) Adapter

```
peft:
    peft_scheme: "adapter"  # can be either adapter, ia3, ptuning,
adapter_and_ptuning, or lora
    restore_from_path: null

    adapter_tuning:
      type: 'parallel_adapter' # this should be either 'parallel_adapter' or
'linear_adapter'
      adapter_dim: 32
      adapter_dropout: 0.0
      norm_position: 'pre' # This can be set to 'pre' or 'post', 'pre' is
normally what is used.
      column_init_method: 'xavier' # options: xavier, zero or normal
      row_init_method: 'zero' # options: xavier, zero or normal
      norm_type: 'mixedfusedlayernorm' # options are ['layernorm',
'mixedfusedlayernorm']
```

Fine-Tuning in NeMo

An All-in-One Implementation

1. Set Parameter-Efficient Fine-Tuning Type
2. Set Hyperparameters
   a) Adapter
   b) LoRA

```
peft:
    peft_scheme: "adapter"  # can be either adapter, ia3, ptuning,
adapter_and_ptuning, or lora
    restore_from_path: null
```

```
    lora_tuning:
      adapter_dim: 32
      adapter_dropout: 0.0
      column_init_method: 'xavier' # options: xavier, zero or normal
      row_init_method: 'zero' # IGNORED if linear_adapter is used, options:
xavier, zero or normal
```

Fine-Tuning in NeMo

An All-in-One Implementation

1. Set Parameter-Efficient Fine-Tuning Type
2. Set Hyperparameters
   a) Adapter
   b) LoRA
   c) P-Tuning

```
peft:
    peft_scheme: "adapter"  # can be either adapter, ia3, ptuning,
adapter_and_ptuning, or lora
    restore_from_path: null
```

```
    p_tuning:
      virtual_tokens: 10  # The number of virtual tokens the prompt encoder
should add at the start of the sequence
      bottleneck_dim: 1024  # the size of the prompt encoder mlp bottleneck
      embedding_dim: 1024  # the size of the prompt encoder embeddings
      init_std: 0.023
```

Recap

What did we learn today?

1. **Why** might you want to you customize your own LLM?
   a. Better performance, save money, reduce latency, smaller models.
2. **How** should you customize your own LLM?
   a. For most use cases, parameter-efficient fine-tuning (PEFT). Choose what's easiest for you.
3. **What data** is needed to customize your own LLM?
   a. You're already generating your own data. Start recording it! Also try synthetic data generation.
4. Do you use a **hosted API or self-manage** to customize your own LLM?
   a. Choice is up to the developer. Consider cost, convenience, privacy, portability.

Live Q&A

Q&A

Apply to NVIDIA Inception for startups:
NVIDIA.com/startups

Join the NVIDIA Developer community to get access to technical training, technology, AI models and 600+ SDKs:

Developer.nvidia.com/join

Explore More Gen AI/LLM Training:

25% off Workshops for LLM Day registrants*

USE CODE: **TRAIN-LLM**

Instructor-Led Workshops

> Generative AI With Diffusion Models

> Rapid Application Development Using LLMs

> Efficient Large Language Model (LLM) Customizations

Self-Paced Courses

> Generative AI Explained (Free)

> Generative AI With Diffusion Models

View our comprehensive Gen AI/LLM learning path, covering fundamental to advanced topics
> Gen AI/LLM Learning Path

**Offer valid for any of the DLI public workshops scheduled through March 01, 2024.*

Supplementary Materials

Comparison of Approaches

	Prompt Engineering	Prompt Turning	P-Tuning	Adapter	LoRA	IA3	Full-Param Fine-Tuning
Frozen model weights	Yes	Yes	Yes	Yes	Yes	Yes	No
Same model architecture	Yes	Yes	Yes	No	No in training Yes in Inference	No	Yes
New added parameters	Zero	Limited	Limited	Moderate	Moderate	Limited	Large
Extra inference latency	High	Moderate	Moderate	Limited	Zero	Limited	Zero
Extra inference computation cost	High	Moderate	Moderate	Limited	Zero	Limited	Zero
Multi-task in one inference batch	Yes	Yes	Yes	No	No	No	No
Accuracy	Fair	Good	Good	Better	Better	Better	Best
Training data requested	Minimum	Limited	Limited	Moderate	Moderate	Moderate	High
Training computation cost	Zero	Limited	Limited	Moderate	Moderate	Moderate	High