Easy and Scalable Federated Learning in the Age of Large Language Models with NVIDIA FLARE

Holger Roth, Principal Federated Learning Scientist, NVIDIA

FL@FM-ICME, Niagara Falls, July 15th, 2024

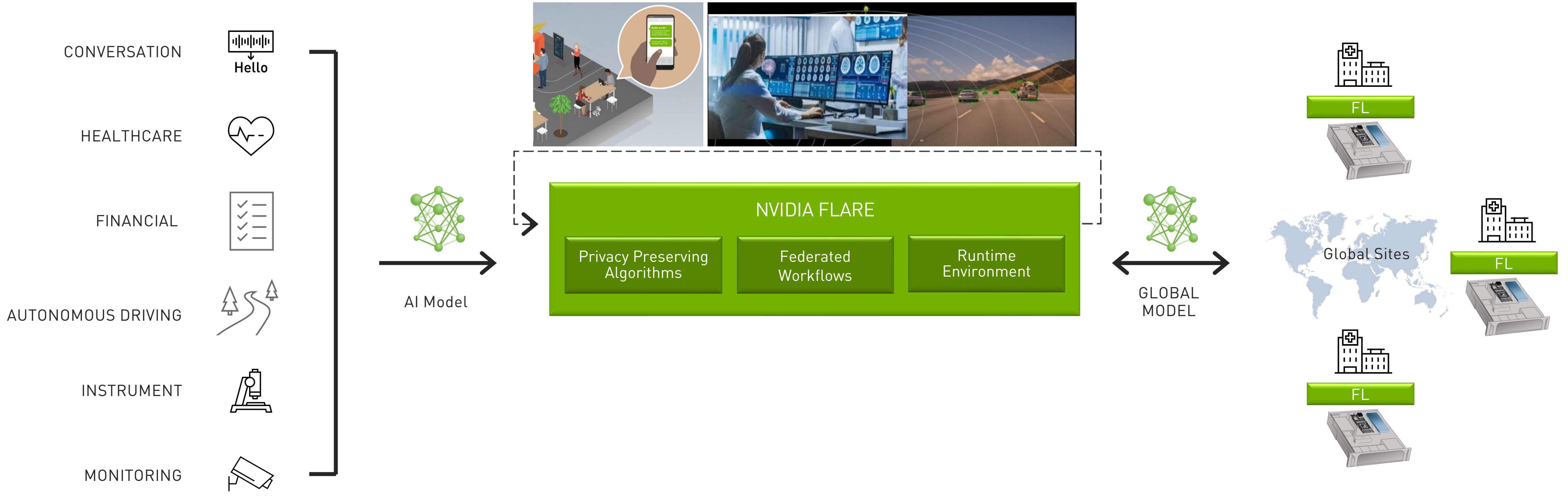
LLMs in NVIDIA FLARE

1. NVIDIA FLARE Overview
2. Parameter-efficient Fine-tuning (PEFT)
3. Supervised Fine-tuning (SFT)

NVIDIA FLARE Overview

NVIDIA Federated Learning

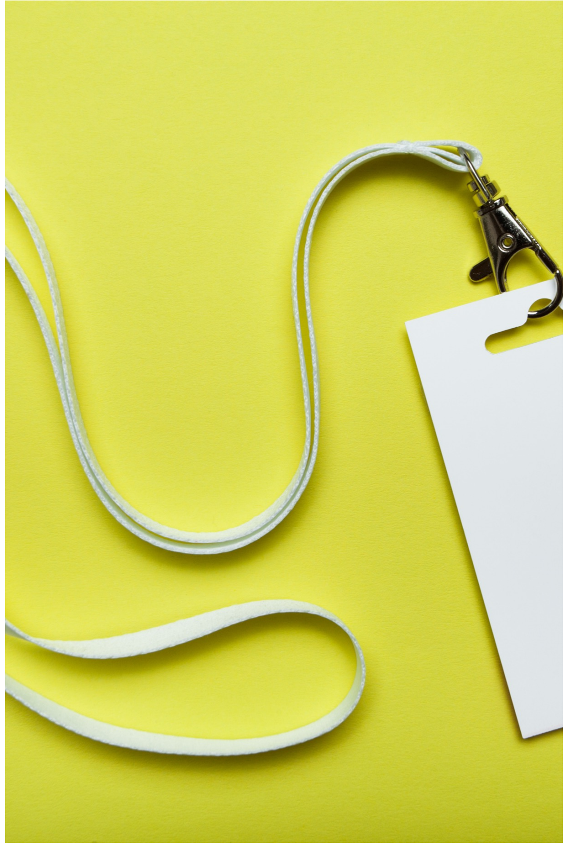
Applications across industries



Available on GitHub: https://github.com/NVIDIA/NVFlare

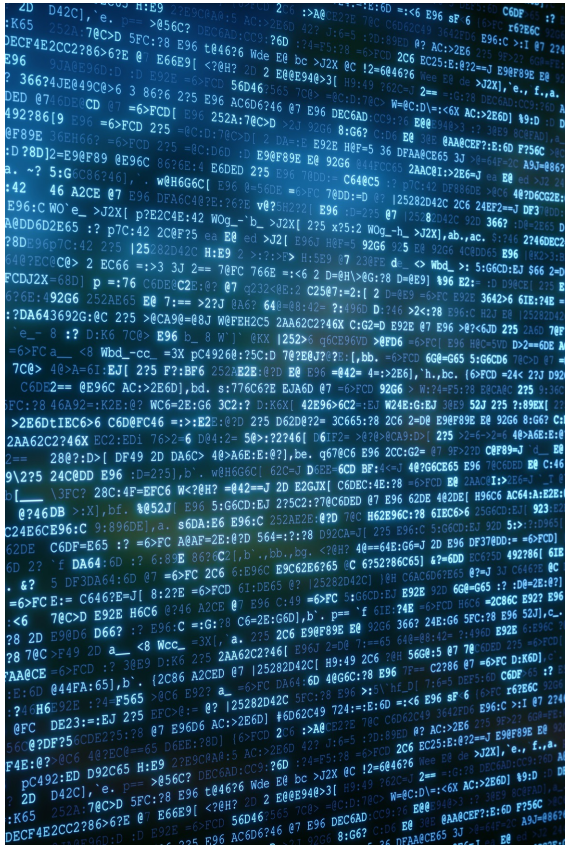
NVIDIA FLARE Security and Data Privacy

Defense in Depth approach to protecting data privacy and model IP



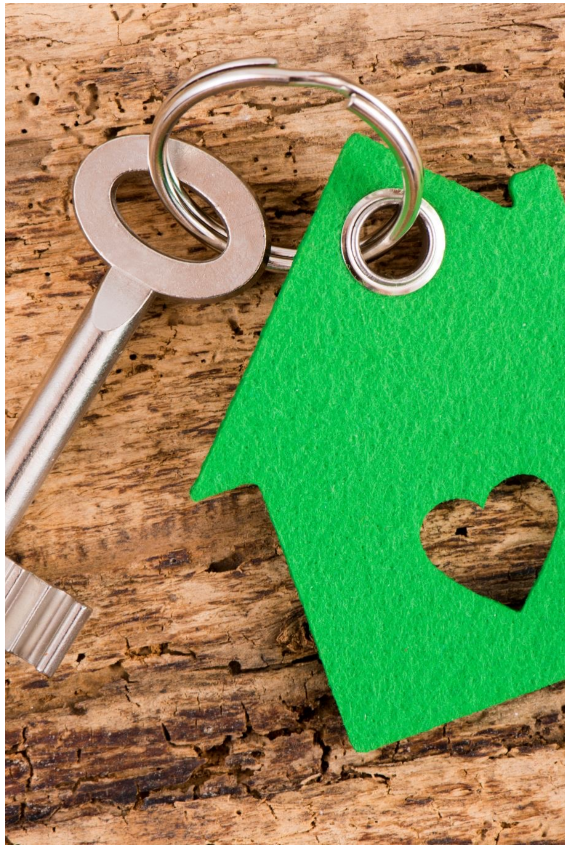
User Identity Verification

Certificate and derived token authentication



Data Encryption in Transit

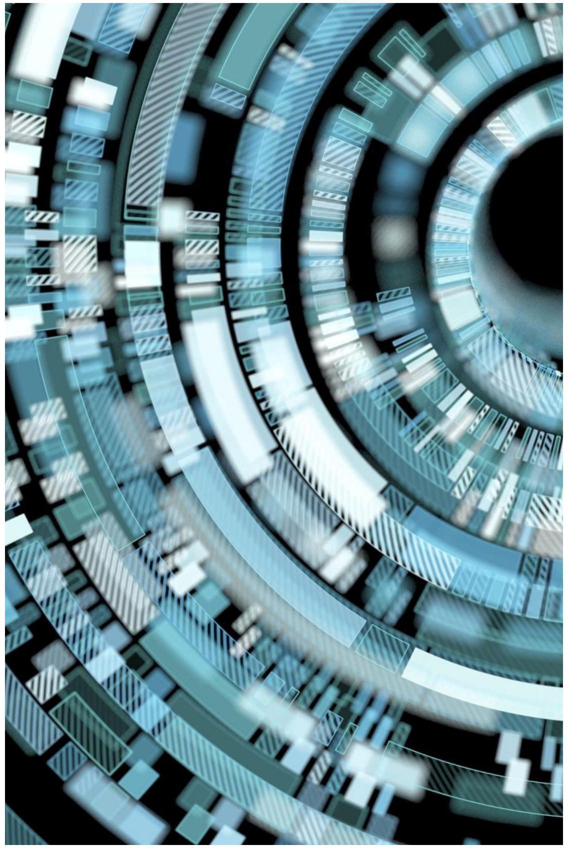
Server-Client communication encrypted



User Defined Security Policy

Site-Specific authentication

Job authorization



Privacy Preserving Algorithm

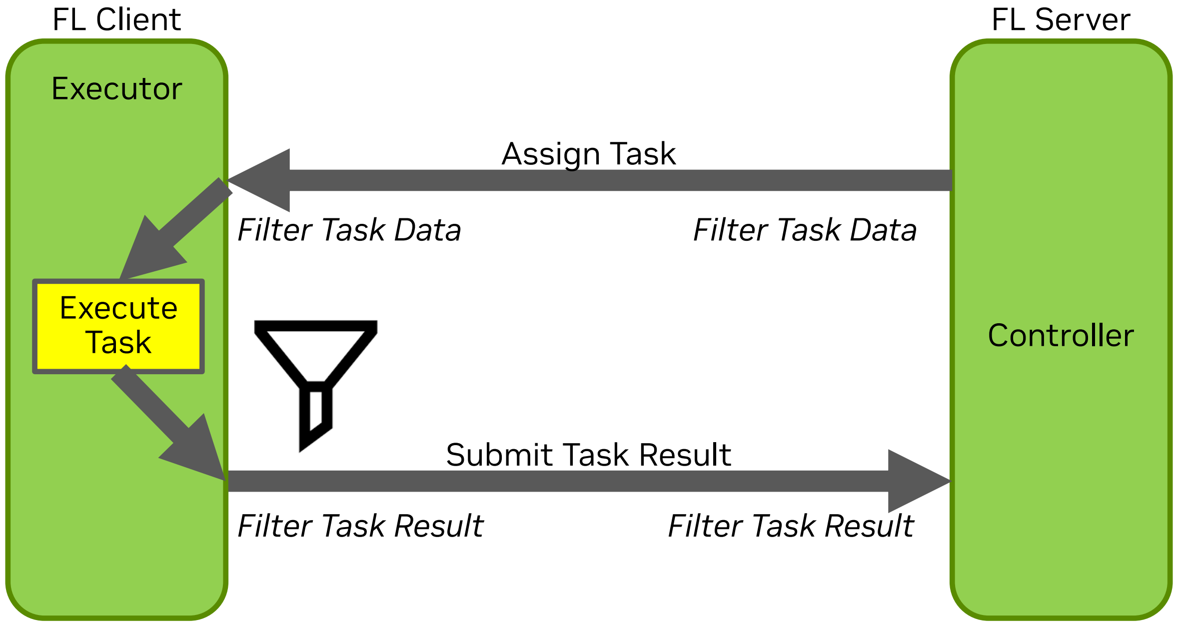
Differential Privacy

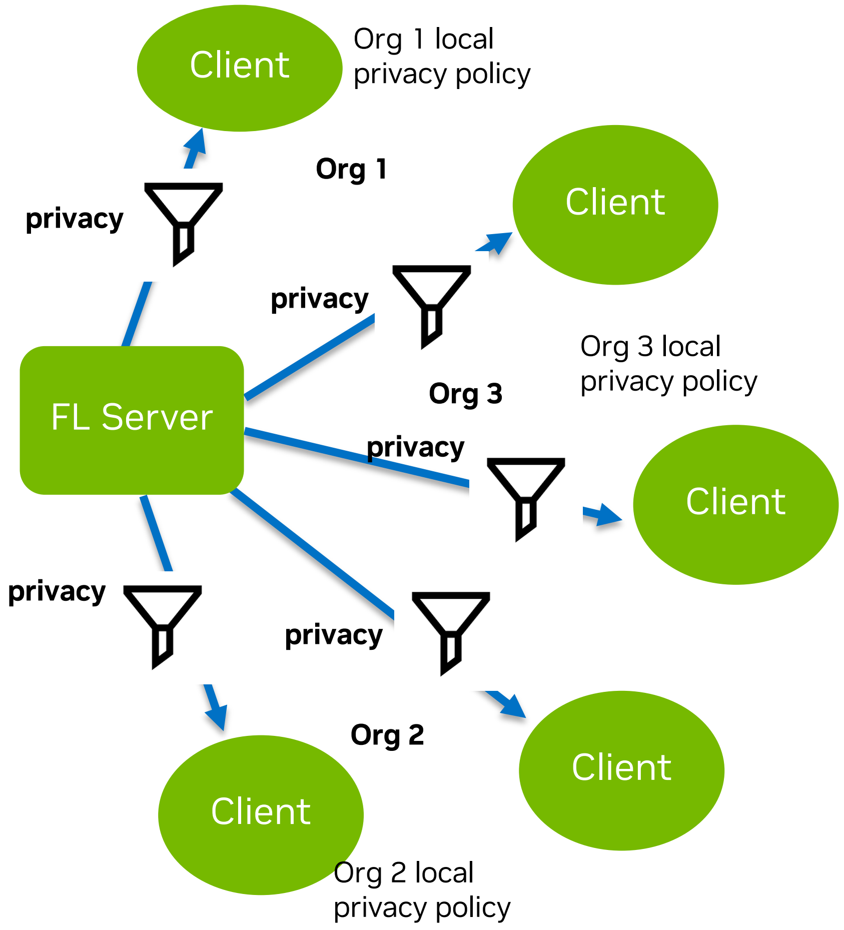
Homomorphic Encryption

Confidential Computing

High Level Architecture

Data privacy architecture





- **Privacy filter can depend on:**
  - Scope: any key-value pair such as datasets
  - Data kind: Weights, Weights Diff or Analytics data
  - Or any other data
- **Research develop privacy filter**
- **Organization set privacy policy:**
  - privacy budget, noise level as data privacy policy

Who's Using NVIDIA FLARE?

...many more

https://developer.nvidia.com/flare
https://blogs.nvidia.com/blog/2021/11/29/federated-learning-ai-nvidia-flare

Parameter-efficient Fine-tuning (PEFT)

Adapt Foundational LLMs in FL

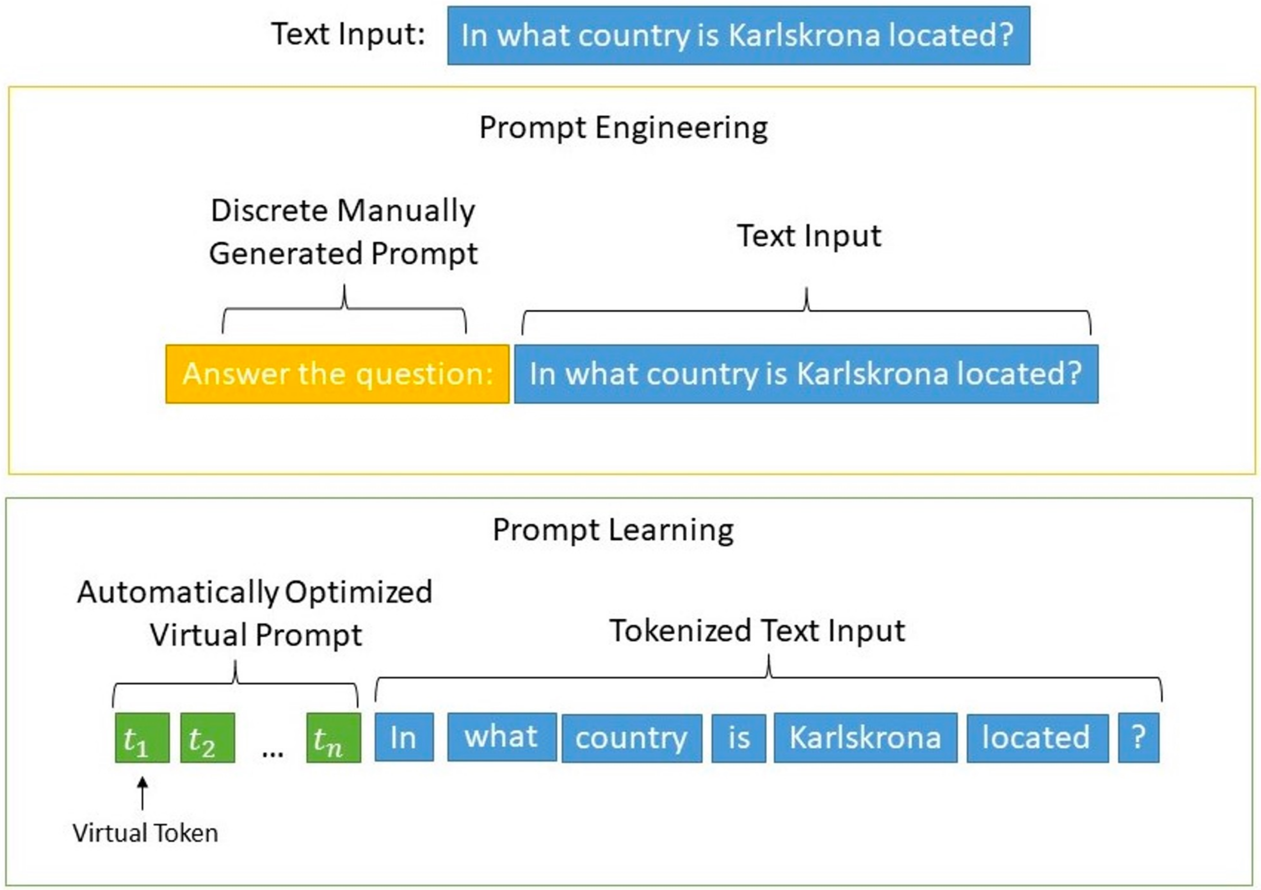
Parameter-efficient fine-tuning

Fine-tuning with a task-specific module

- **Most LLM layers fixed**; Only few dozen million params are being exchanged
- Tech: prompt-tuning/p-tuning/adapter/LoRA/others
- NVFlare example: sentiment analysis example with NeMo GPT model (345M/5B/**20B**)

Prompt Learning

Parameter-efficient adaptation of LLMs to downstream tasks



Tasks: brainstorming, classification, closed QA, generation, information extraction, open QA, summarization, etc.

Source: https://developer.nvidia.com/blog/adapting-p-tuning-to-solve-non-english-downstream-tasks

P-Tuning for Sentiment Analysis

Downstream task example:

- Financial PhraseBank dataset (Malo et al.) for sentiment analysis.
- The Financial PhraseBank dataset contains the sentiments for financial news headlines from a retail investor's perspective.

Example prompts and predictions:

The products have a low salt and fat content . ***sentiment: neutral***

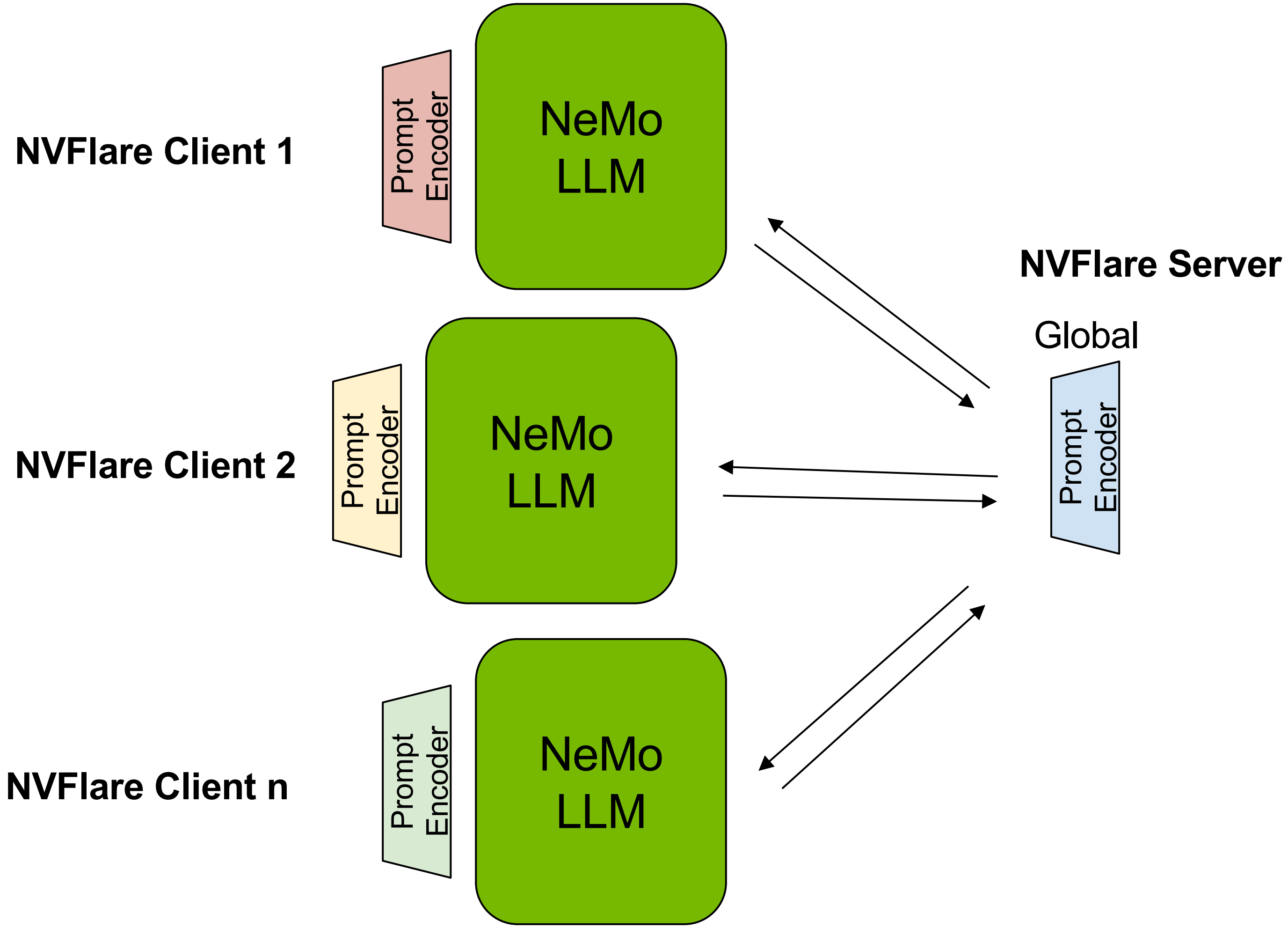
The agreement is valid for four years . ***sentiment: neutral***

Diluted EPS rose to EUR3 .68 from EUR0 .50 . ***sentiment: positive***

The company is well positioned in Brazil and Uruguay . ***sentiment: positive***

Profit before taxes decreased by 9 % to EUR 187.8 mn in the first nine months of 2008 , compared to EUR 207.1 mn a year earlier . ***sentiment: negative***

NVFlare for P-Tuning With NeMo



LLM parameters stay fixed; Prompt encoder parameters are trained/updated

Lightning Client API

Example with NeMo PEFT script

Transform your script to FL with a few lines of code changes:

1. Import nvflare lightning api
2. Patch your lightning trainer
3. (Optionally) validate the current global model
4. Train as usually

Directly use all the PEFT methods implemented in NeMo script:

- adapter
- ia3
- p-tuning
- adapter + p-tuning
- LoRa

```
from nemo.core.config import hydra_runner
from nemo.utils import AppState, logging
from nemo.utils.exp_manager import exp_manager
from nemo.utils.model_utils import inject_model_parallel_rank

# (0): import nvflare lightning api
import nvflare.client.lightning as flare

mp.set_start_method("spawn", force=True)

...

# (1): flare patch
flare.patch(trainer)

while flare.is_running():

    # (2) evaluate the current global model to allow server-side model selection
    print("--- validate global model ---")
    trainer.validate(model)

    # (3) Perform local training starting with the received global model
    print("--- train new model ---")
    trainer.fit(model)
```

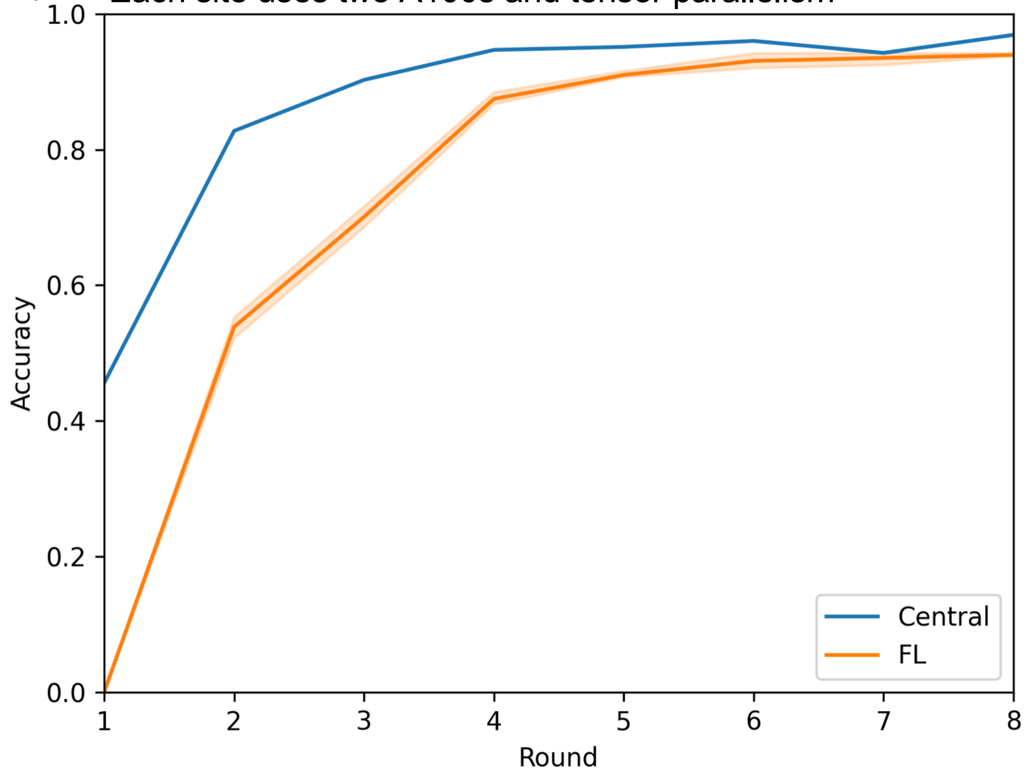
Based on https://github.com/NVIDIA/NeMo/blob/main/examples/nlp/language_modeling/tuning/megatron_gpt_peft_tuning.py

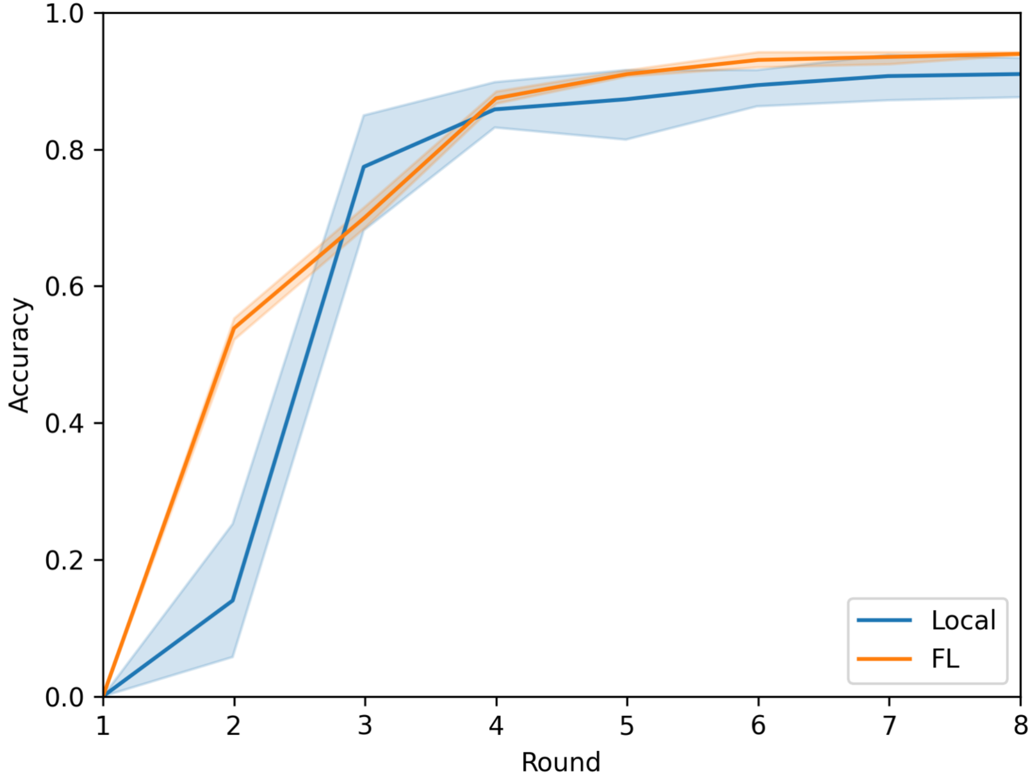
P-Tuning for Sentiment Analysis

FL can achieve performance comparable to centralized training

Federated p-tuning experiment:

- Using ***20B NeMo Megatron-GPT*** model hosted on HuggingFace
- **50M** parameters are updated (0.25%)
- 1800 pairs of statement and sentiment
- 600 for each site; shared validation set for direct comparison
- Each site uses two A100s and tensor parallelism

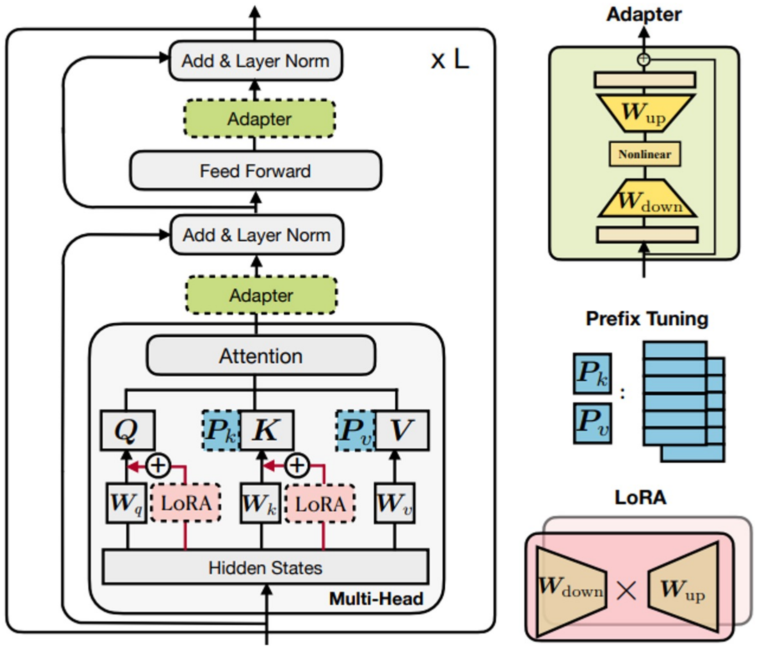




Compare PEFT Methods With NeMo

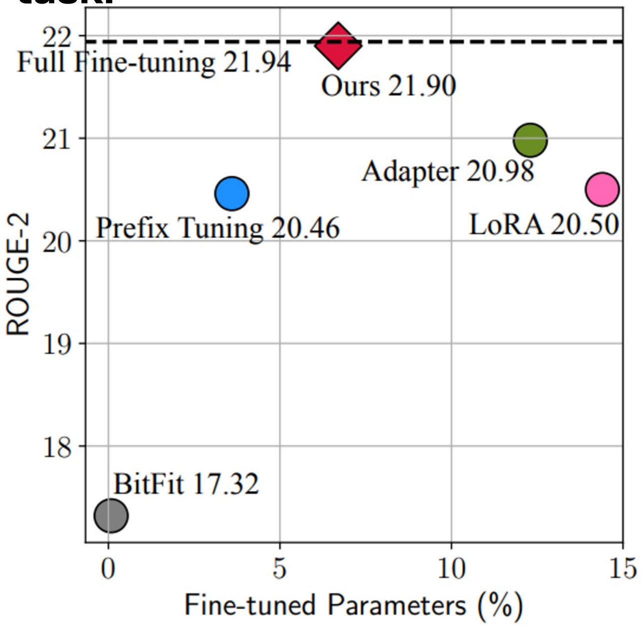
Only 1 line configuration change

Transformer and PEFT methods:



Source: https://arxiv.org/abs/2110.04366

Different PEFT methods on the XSum summarization task:

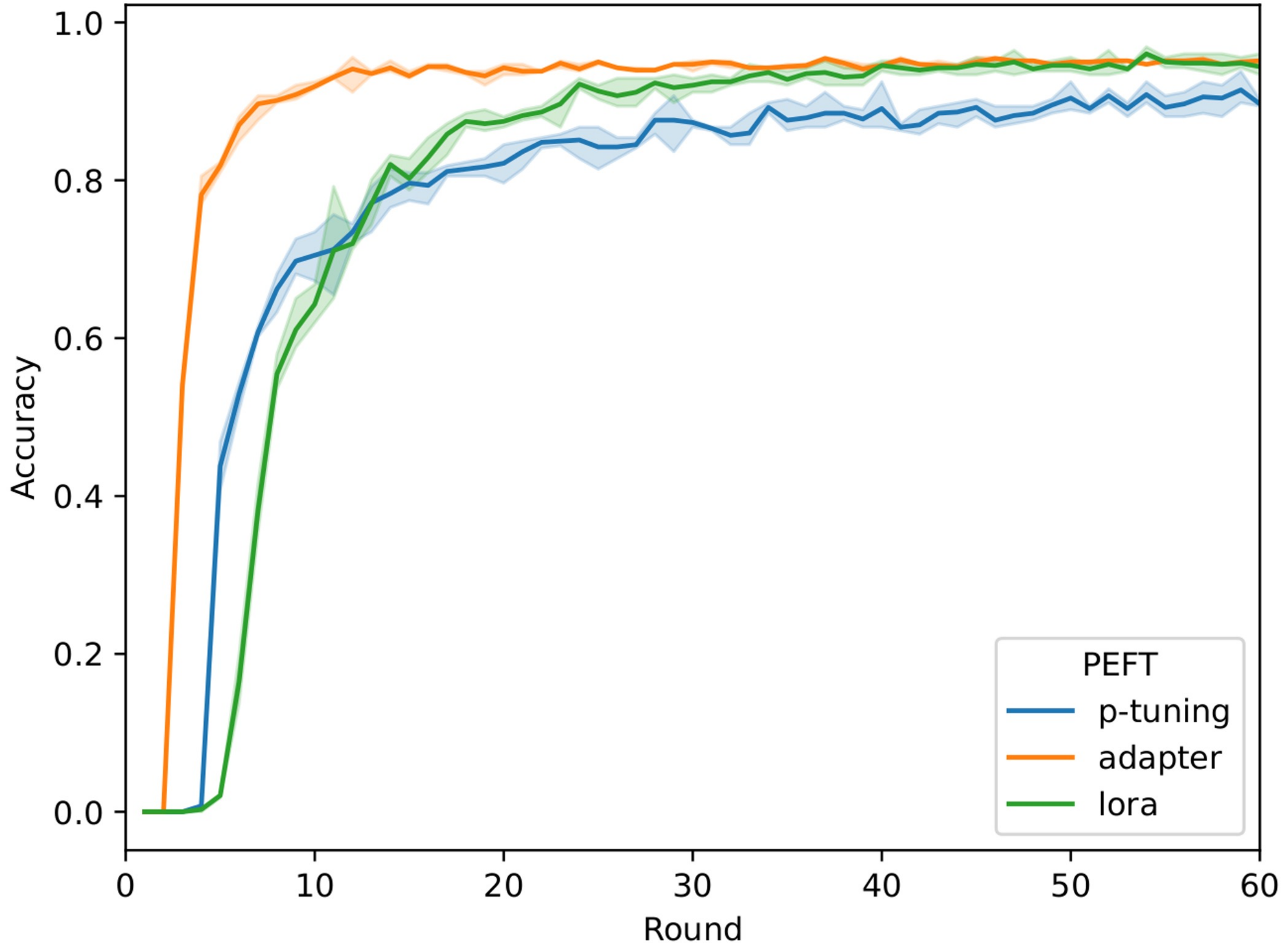



```
peft:
  peft_scheme: "adapter"  # can be either adapter,ia3, or ptuning
  restore_from_path: null

  # Used for adapter peft training
  adapter_tuning:
    type: 'parallel_adapter' # this should be either 'parallel_adap
    adapter_dim: 32
    adapter_dropout: 0.0
    norm_position: 'pre' # This can be set to 'pre', 'post' or null
    column_init_method: 'xavier' # IGNORED if linear_adapter is use
    row_init_method: 'zero' # IGNORED if linear_adapter is used, op
    norm_type: 'mixedfusedlayernorm' # IGNORED if layer_adapter is
    layer_selection: null  # selects in which layers to add adapter
    weight_tying: False
    position_embedding_strategy: null # used only when weight_tying

  lora_tuning:
    adapter_dim: 32
    adapter_dropout: 0.0
    column_init_method: 'xavier' # IGNORED if linear_adapter is use
    row_init_method: 'zero' # IGNORED if linear_adapter is used, op
    layer_selection:  null  # selects in which layers to add lora a
    weight_tying: False
    position_embedding_strategy: null # used only when weight_tying

  # Used for p-tuning peft training
  p_tuning:
    virtual_tokens: 10  # The number of virtual tokens the prompt e
    bottleneck_dim: 1024  # the size of the prompt encoder mlp bott
    embedding_dim: 1024  # the size of the prompt encoder embedding
    init_std: 0.023

  ia3_tuning:
    layer_selection:  null  # selects in which layers to add ia3 ad
```

NeMo YAML configuration

Compare PEFT Methods With NeMo

P-tuning vs. Adapter vs. LoRa



Tensor parallel with **2 GPUs** per client

345M Param NeMo GPT Megatron model

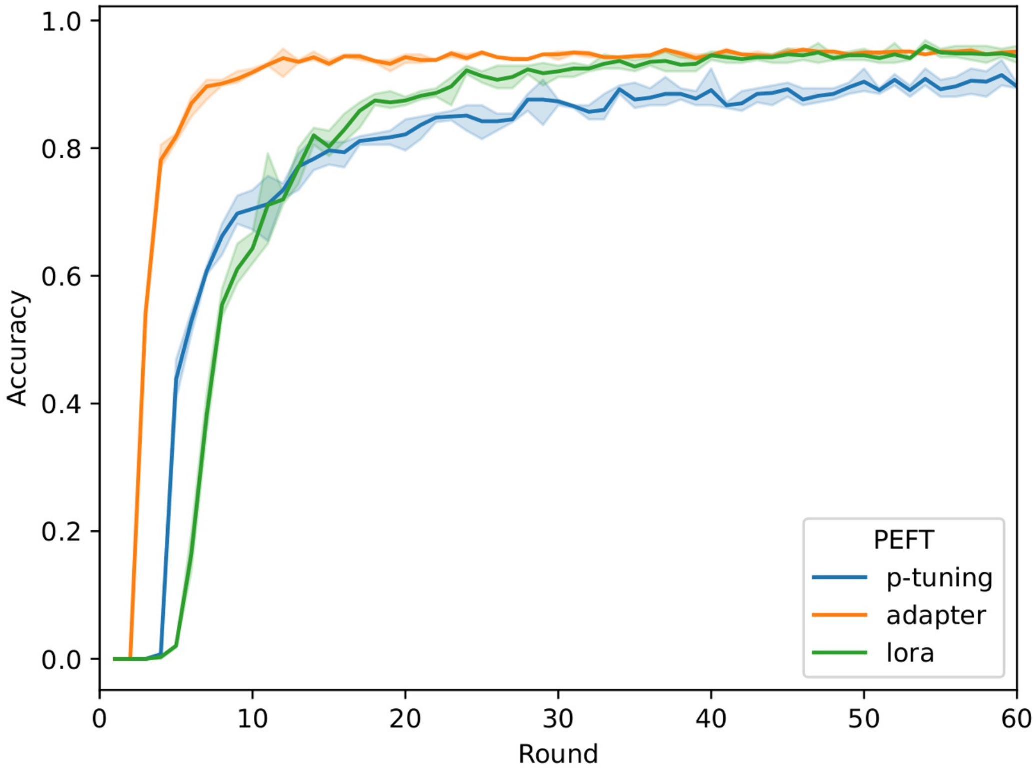
PEFT Method	Execution time
P-tuning	4h 59m
Adapter	11h 25m
LoRA	7h 27m

Example notebook

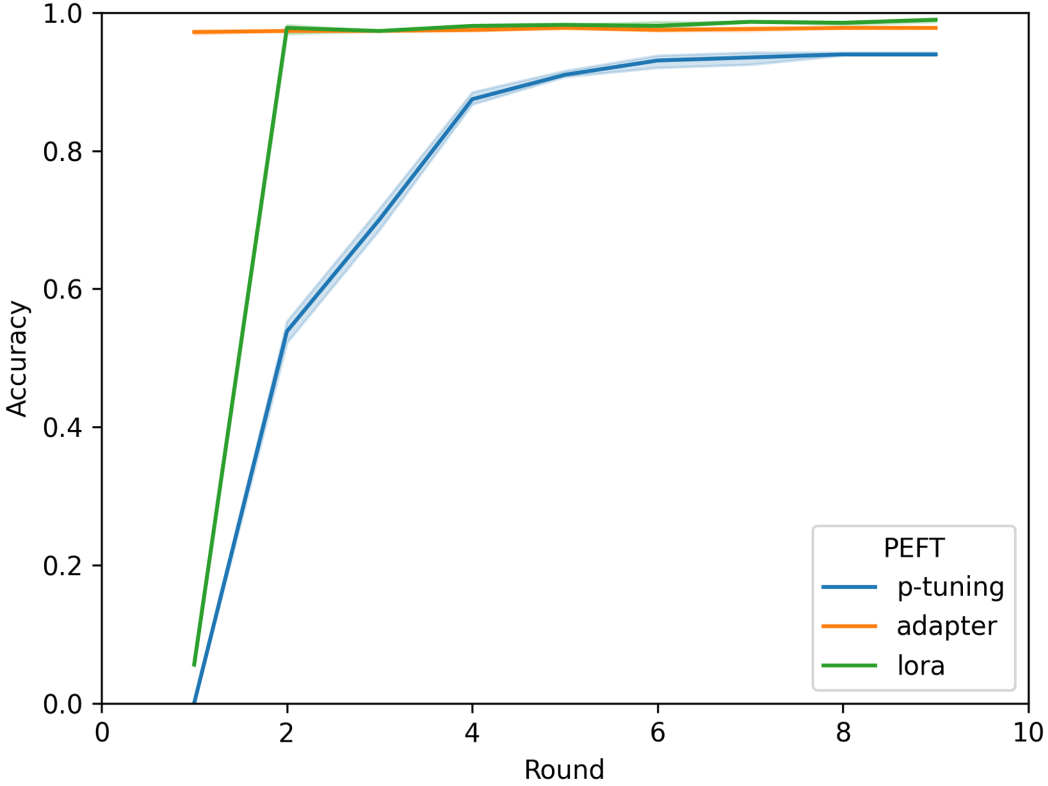
Compare PEFT Methods With NeMo

P-tuning vs. Adapter vs. LoRa

345m



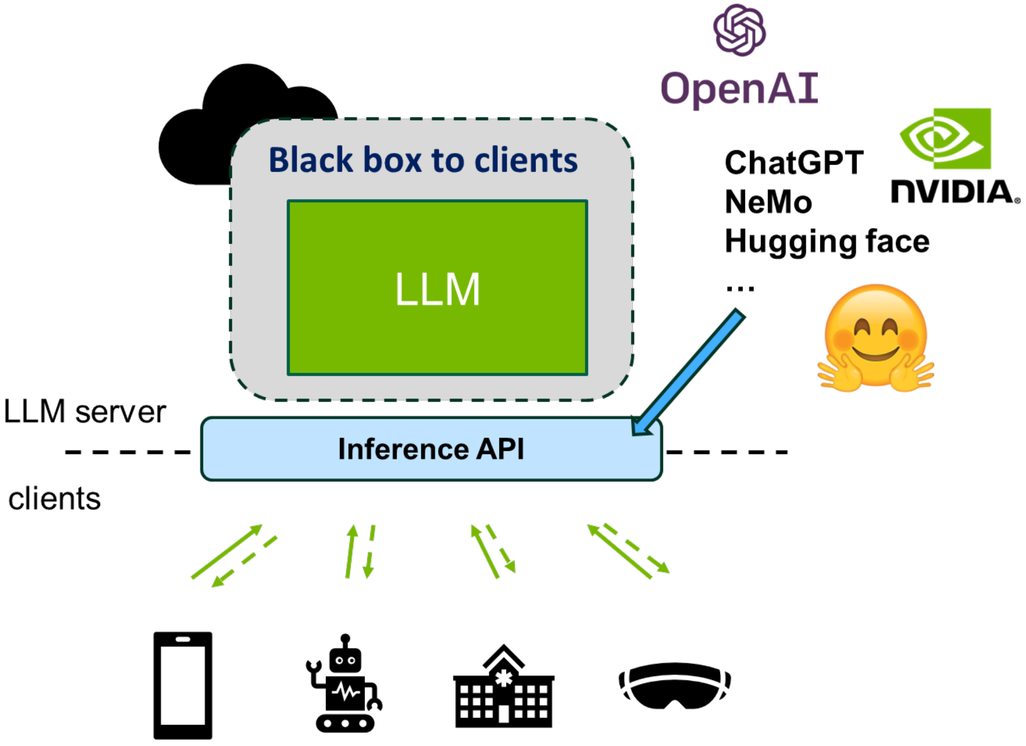
20B



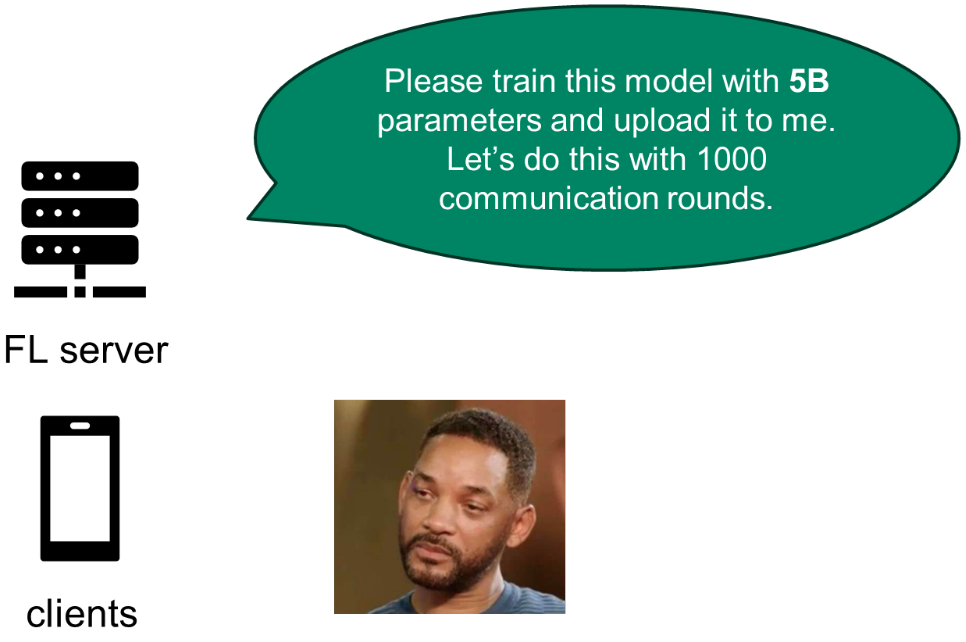
Example: https://github.com/NVIDIA/NVFlare/tree/main/integration/nemo/examples

FedBPT: Efficient Federated Black-box Prompt Tuning for Large Language Models

ICML 2024



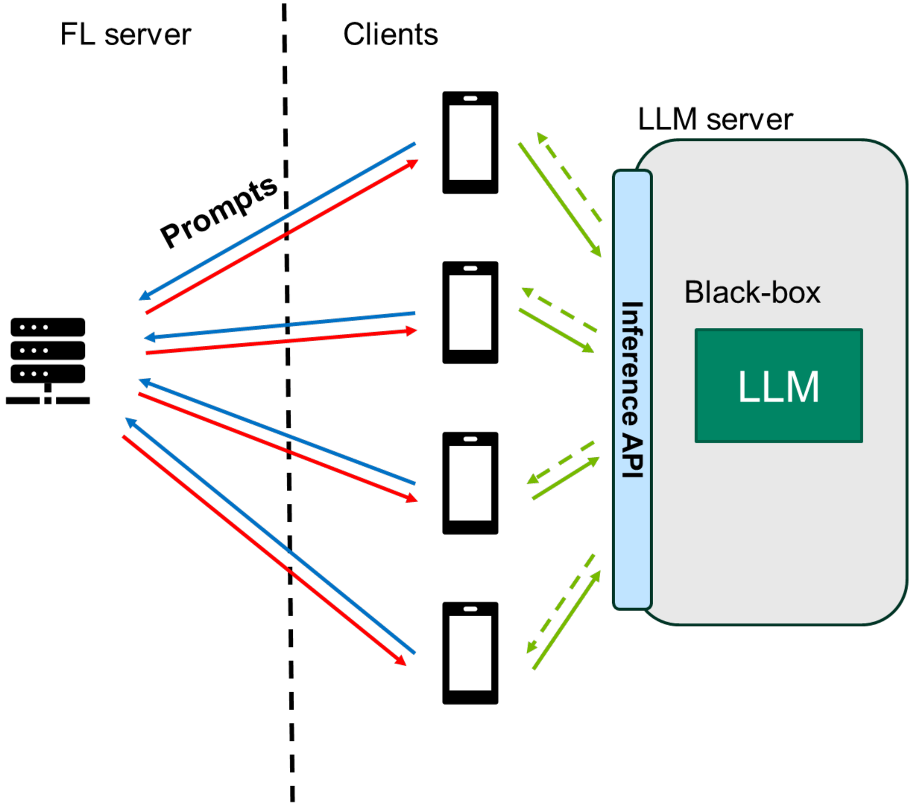
Model access



Computation/memory limitation
Communication cost

[Jingwei Sun et al. ICML 2024 (arxiv 2310.01467)]

FedBPT: Efficient Federated Black-box Prompt Tuning for Large Language Models



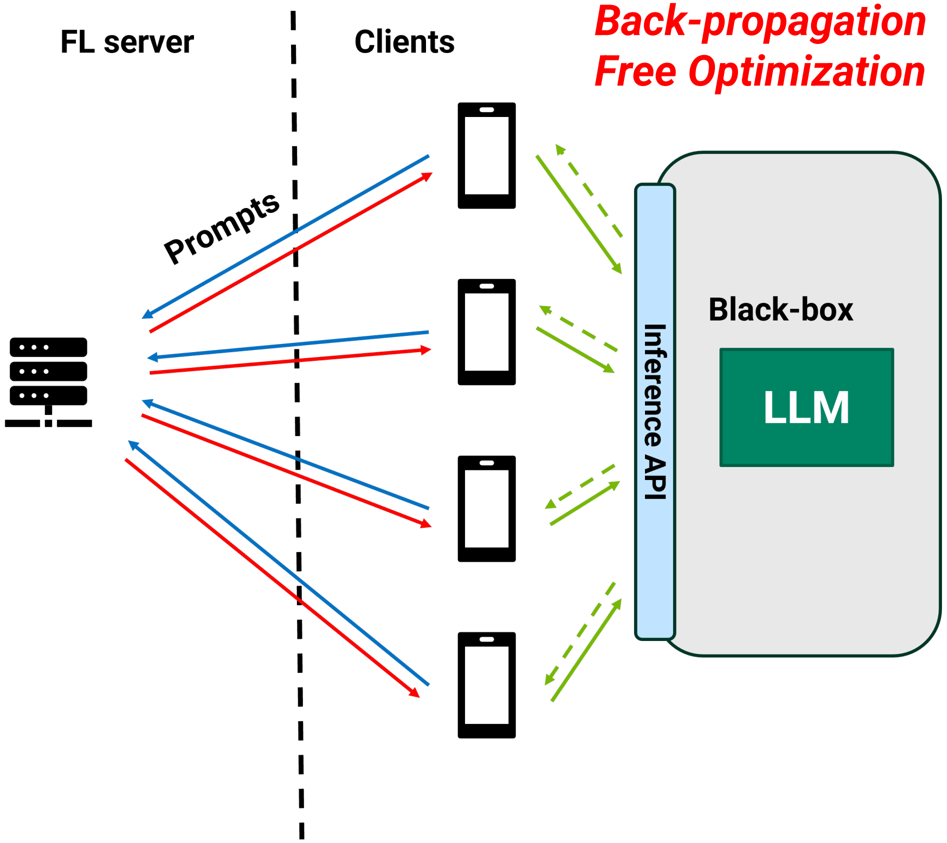
- The clients train prompts while treating the LLM as a black-box model.
- The clients only conduct inference without back-propagation.
- The clients upload and download prompts rather than the whole model.

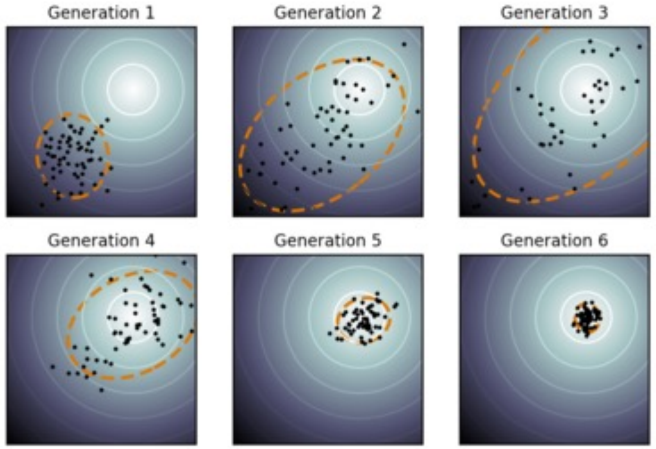
Jingwei Sun et al. ICML 2024 (arxiv 2310.01467)

FedBPT: Efficient Federated Black-box Prompt Tuning for Large Language Models

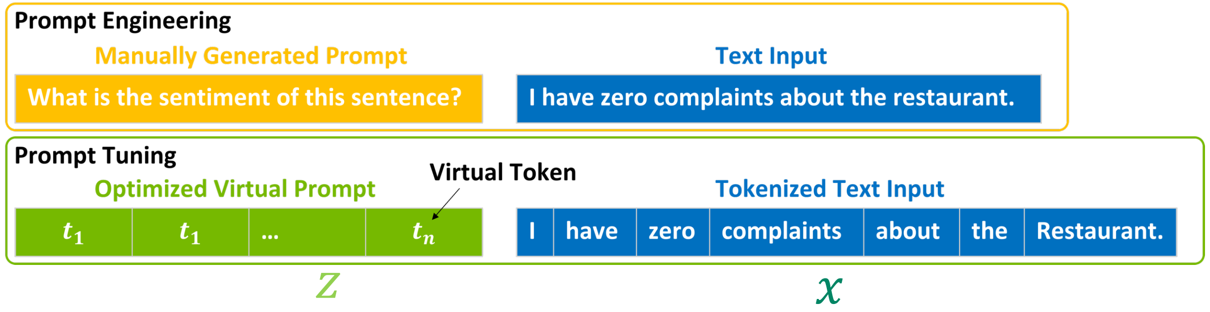
Inference is all you need!!!

Back-propagation Free Optimization





Evolution Strategy (CMA-ES)



$$z^* = \arg\max_{z \in Z} f(z; X, Y, \Theta)$$

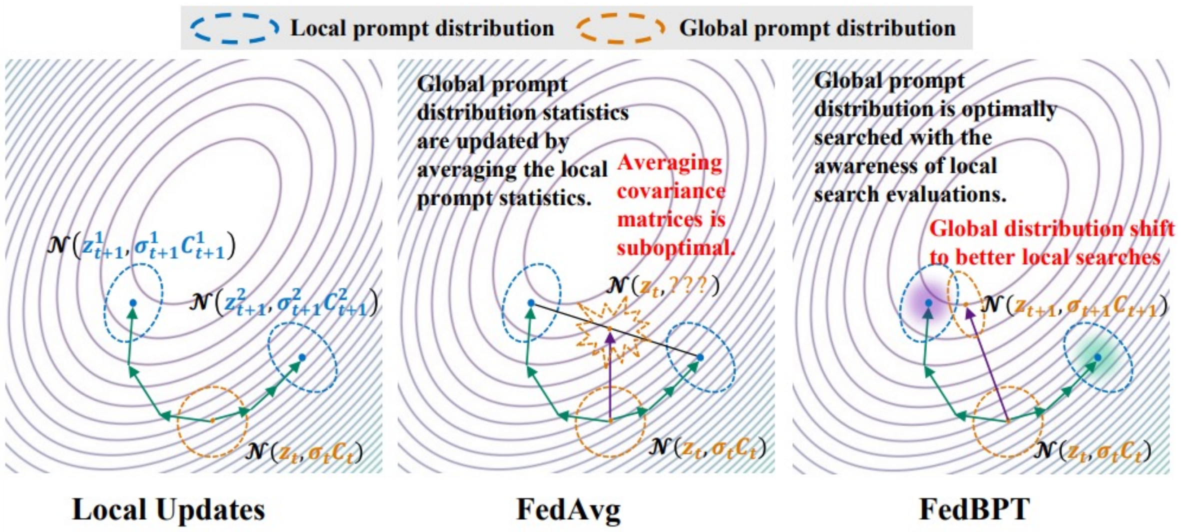
Dimension of z is much lower than Θ

[Jingwei Sun et al. ICML 2024 (arxiv 2310.01467)]

NVIDIA

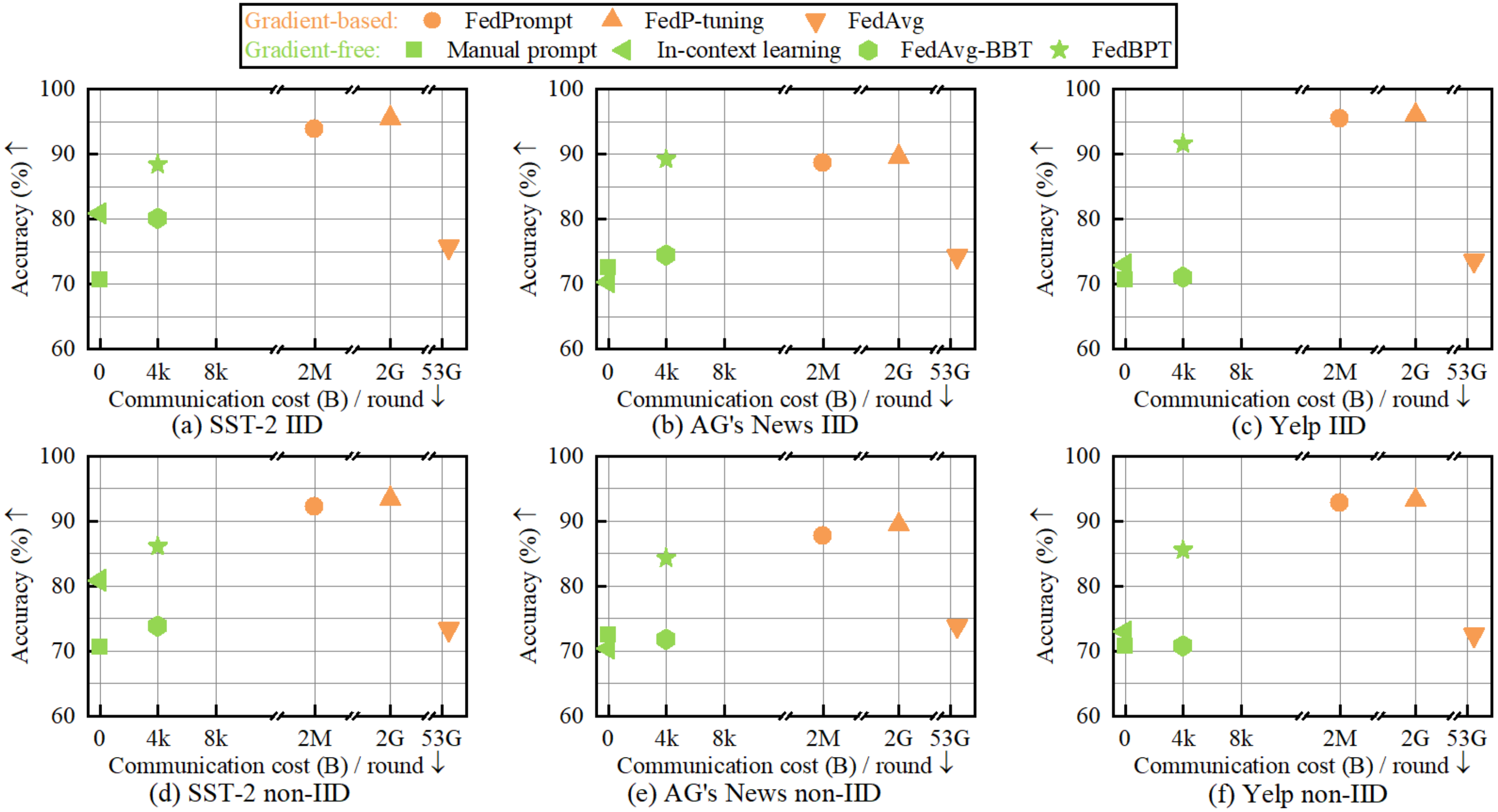
Results on RoBERTa-350M

		# trainable parameters	SST-2		AG's News	
	Method		IID	Non-IID	IID	Non-IID
Gradient-based	FedPrompt	51K	90.25	85.55	87.72	85.62
	FedAvg + P-tuning	15M	**90.6**	**87.16**	**88.17**	**86.11**
	FedAvg + model tuning	355M	83.6	83.6	75.75	75.75
Gradient-free	Manual prompt	0	83.6		75.75	
	FedAvg + BBT	500	84.45	84.17	76.54	76.46
	FedBPT	*500*	***87.52***	***86.70***	***82.36***	***81.03***



Jingwei Sun et al. ICML 2024 (arxiv 2310.01467)

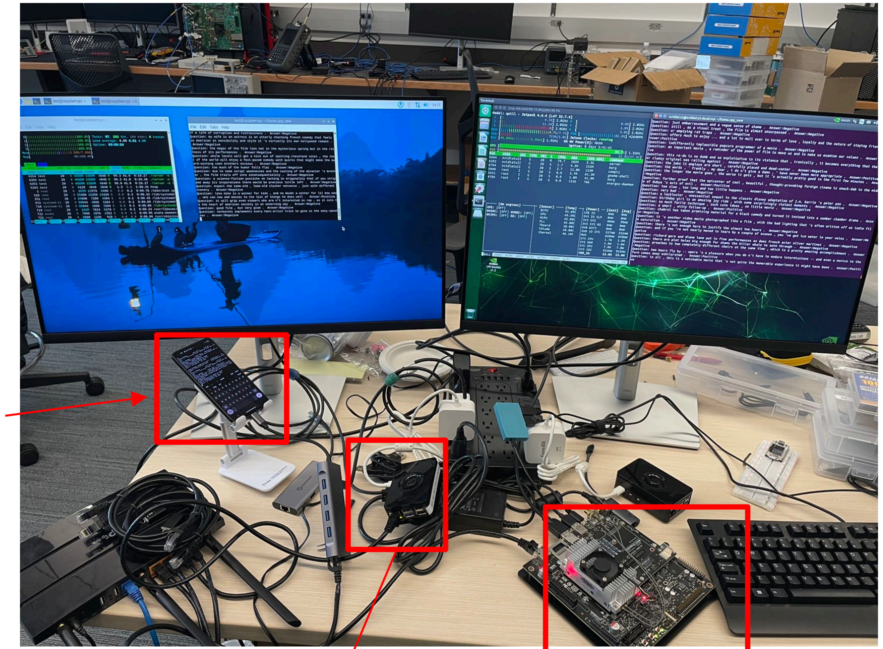
Results on Llama2-7B



Jingwei Sun et al. arxiv 2310.01467

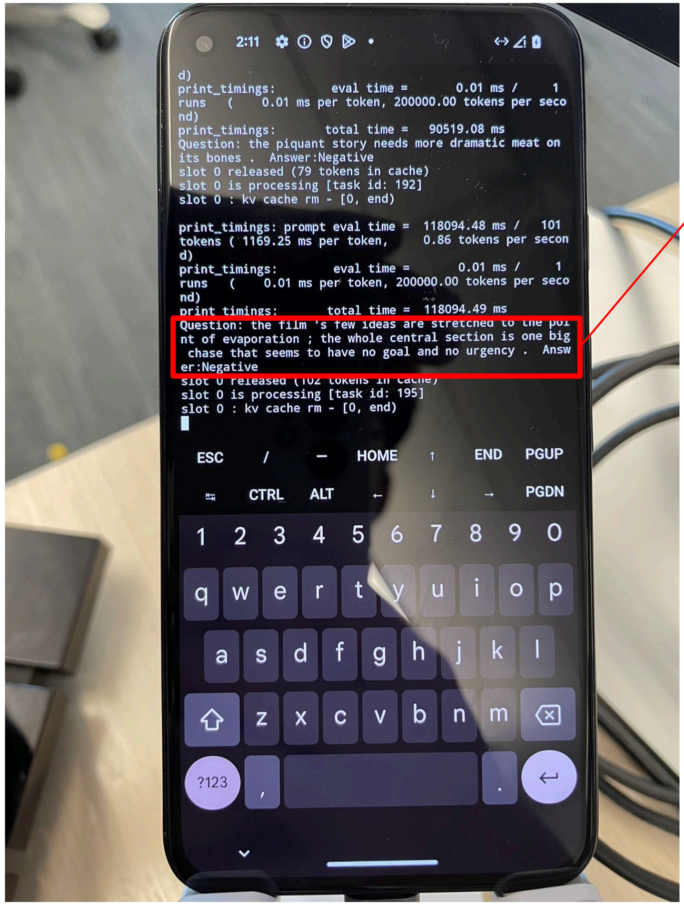
Demo with Llama2-7B

Smart Phone



Raspberry Pi

Jetson TX2



User data

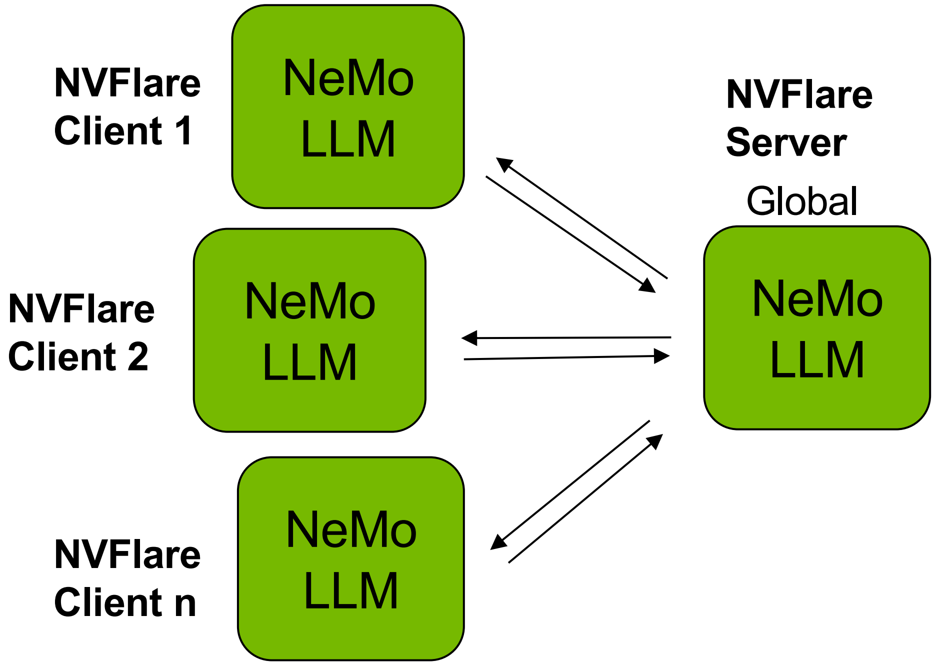
More details (paper and code in NVIDIA FLARE)...

Supervised Fine-tuning

Supervised Fine-tuning (SFT)

Towards “instruction-following” LLM

Unlike PEFT, SFT finetunes the entire network

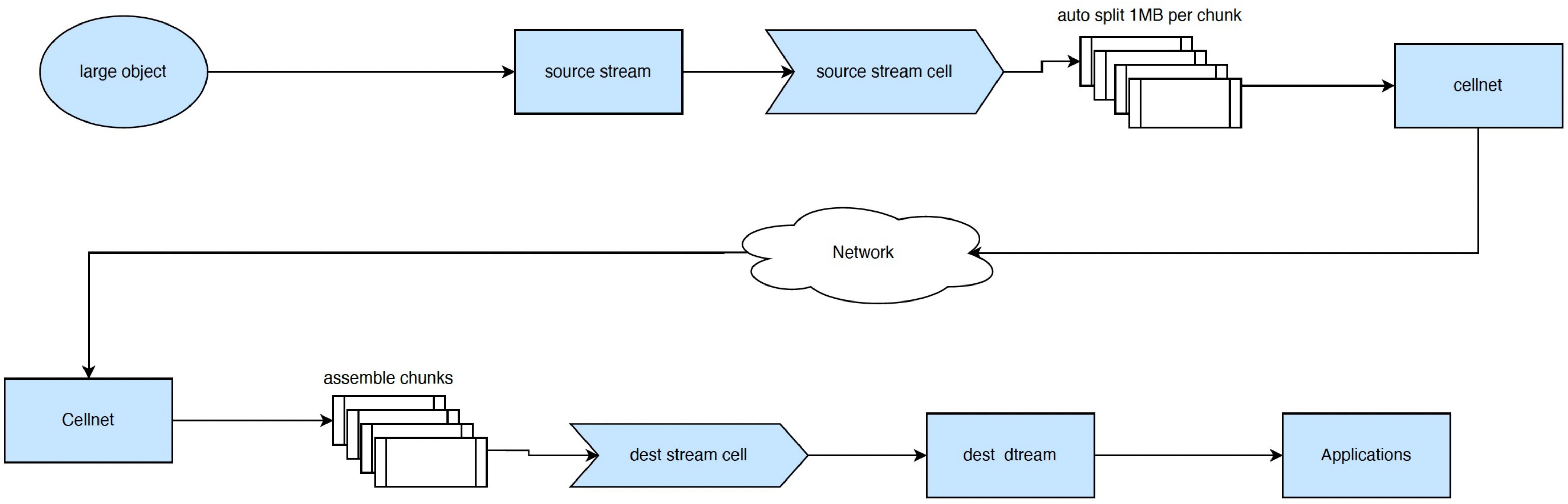


The first step of “Chat-GPT training scheme”.

NVFlare Streaming

Support Large Model Transmission

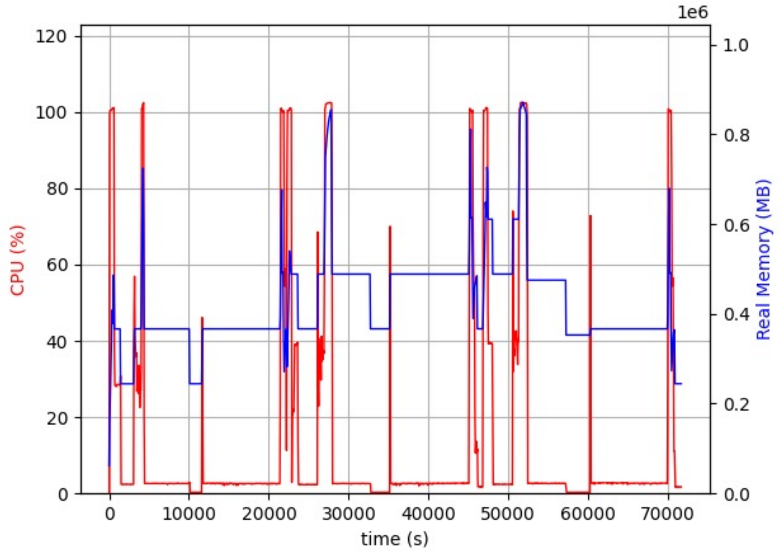
- Model size of mainstream LLM can be huge: 7B -> 26 GB (beyond the 2 GB GRPC limit)
- In order to transmit LLMs in SFT, NVFlare can now support **large object** streaming



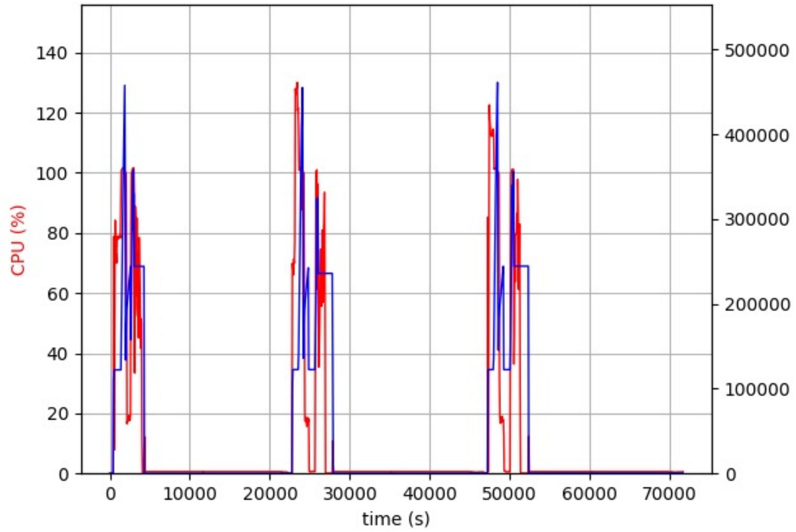
Memory Usage During Streaming

128GB model (compare Llama-2-70B 129GB ckpt)

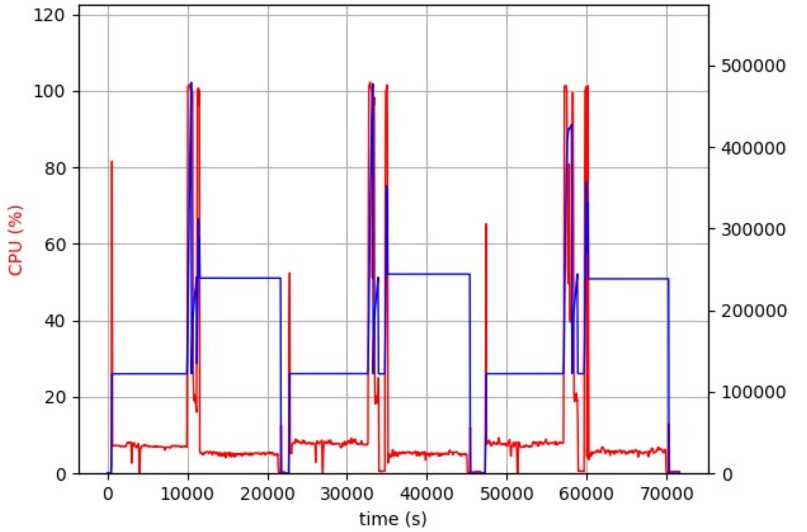
Server



Site-1



Site-2



- Model streaming across regions and cloud providers, including AWS and Azure.
- Clients received and sent the models in about 100 minutes.

Source: https://nvflare.readthedocs.io/en/main/real_world_fl/notes_on_large_models.html#notes-on-large-models

SFT for Instruction Following

3 open datasets

We use three datasets:

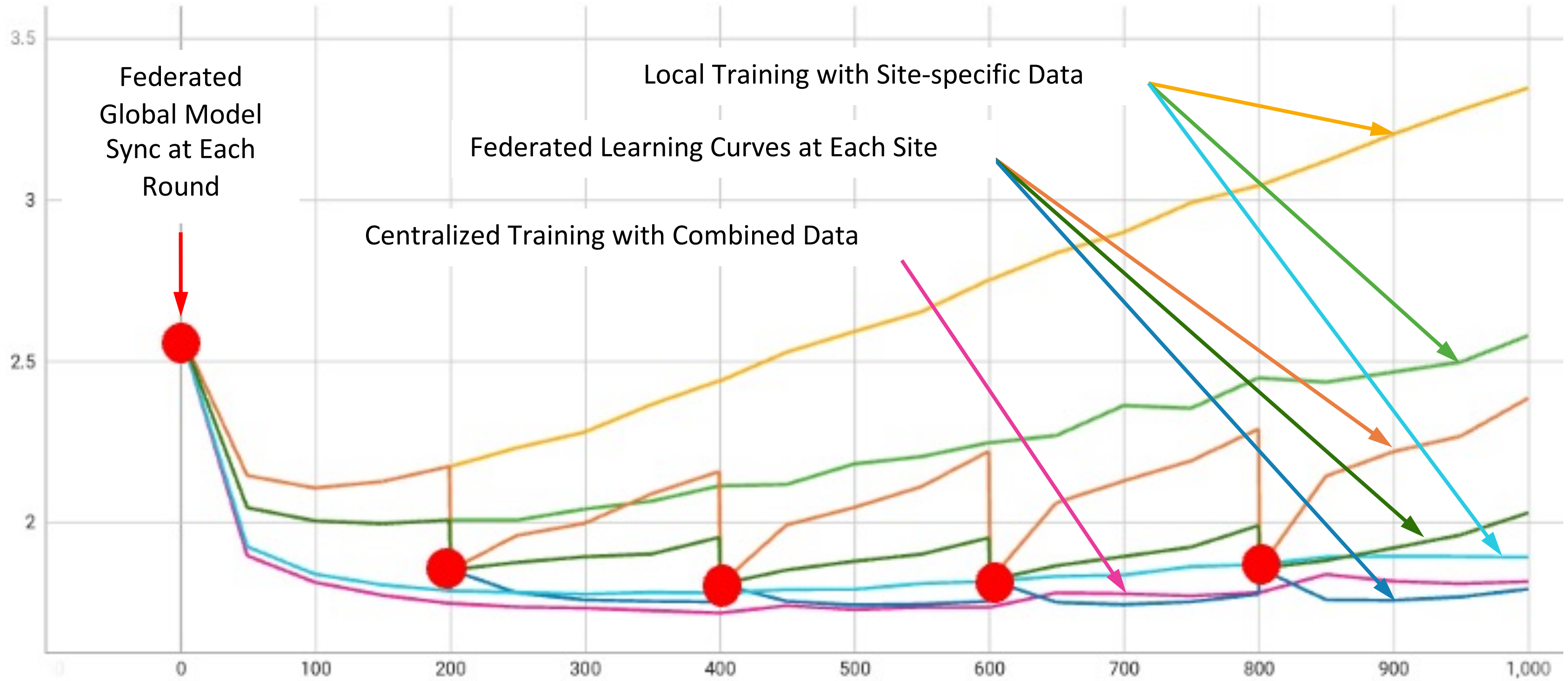
- Alpaca
- databricks-dolly-15k
- OpenAssistant

Containing instruction-following data in different formats under different settings:

- oasst1 features a tree struture for full conversations
- other two are instruction(w/ or w/o context)-response pairs

SFT With FL

Achieving better performance



NeMo 1.3B model, SFT for 5 rounds

5 experiments in total: training on each client's own dataset, combined dataset, and all three clients using FedAvg in NVFlare.

- Local models tend to overfit
- Steps in FL because of global model sync and update

SFT Model Evaluation

LLM Performance

Non-trivial task compared with “fixed downstream tasks” where we usually have metrics like accuracy, F-1 scores, etc.

Common practice is to test the resulting LLMs on **benchmark tasks**, and/or human evaluations

We perform standard language modeling tasks under zero-shot setting, including HellaSwag(H), PIQA(P), and WinoGrande(W)

BaseModel - Before SFT

	H_acc	**H_acc _norm**	**P_acc**	**P_acc _norm**	**W_acc**	**Mean**
BaseModel	0.357	0.439	0.683	0.689	0.537	0.541
Alpaca	0.372	0.451	0.675	0.687	0.550	0.547
Dolly	0.376	**0.474**	0.671	0.667	0.529	0.543
Oasst1	0.370	0.452	0.657	0.655	0.506	0.528
Combined	0.370	0.453	0.685	**0.690**	0.548	0.549
FedAvg	**0.377**	0.469	**0.688**	0.687	**0.560**	**0.556**

Table 1. Model performance on three benchmark tasks: HellaSwag (H), PIQA (P), and WinoGrande (W)

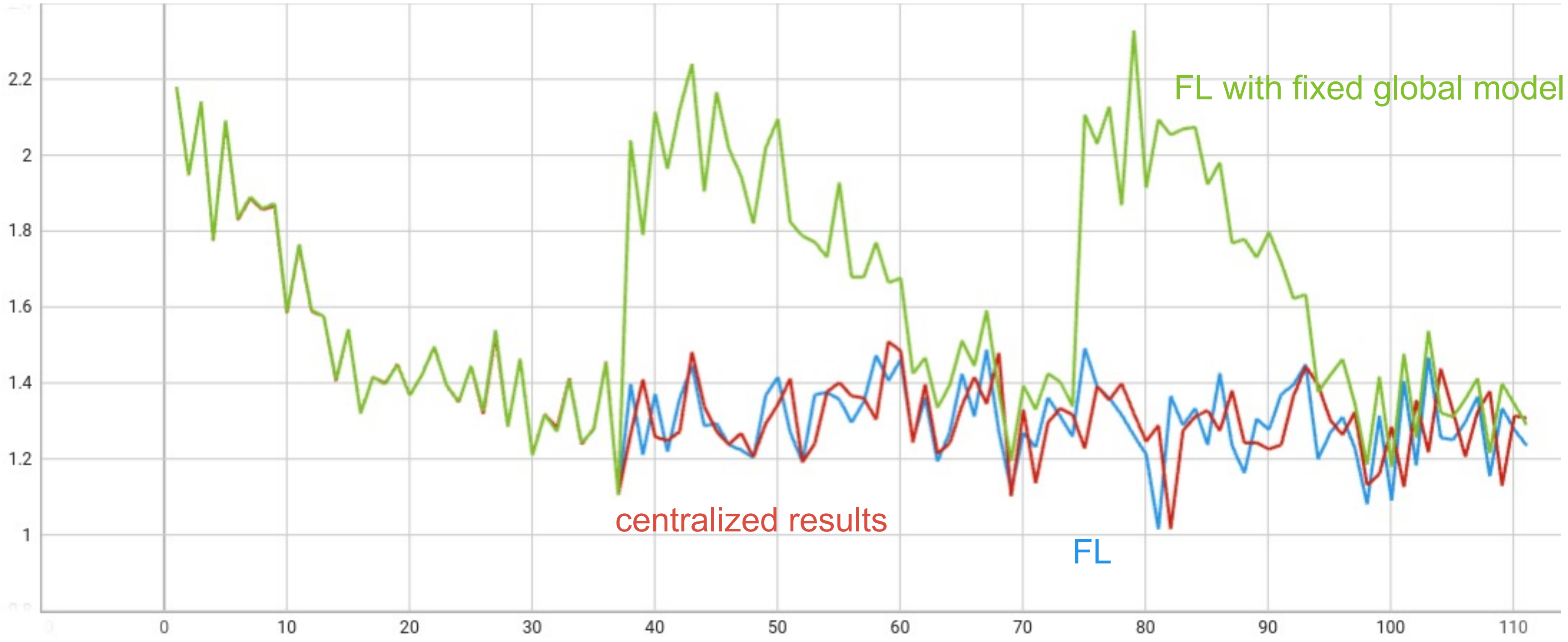
Blog: https://developer.nvidia.com/blog/scalable-federated-learning-with-nvidia-flare-for-enhanced-llm-performance

SFT and PEFT With HuggingFace

LLaMA-2

- Showcasing the functionality of federated SFT and PEFT with [Llama-2-7b-hf model]
- **Model transmission size** over the FLARE network
- PEFT: ~134 MB
- SFT: ~27 GB

PEFT curves for three-epoch centralized training and three-round (one epoch/round) federated learning with one client.



Example: https://github.com/NVIDIA/NVFlare/tree/main/examples/advanced/llm_hf

Conclusions

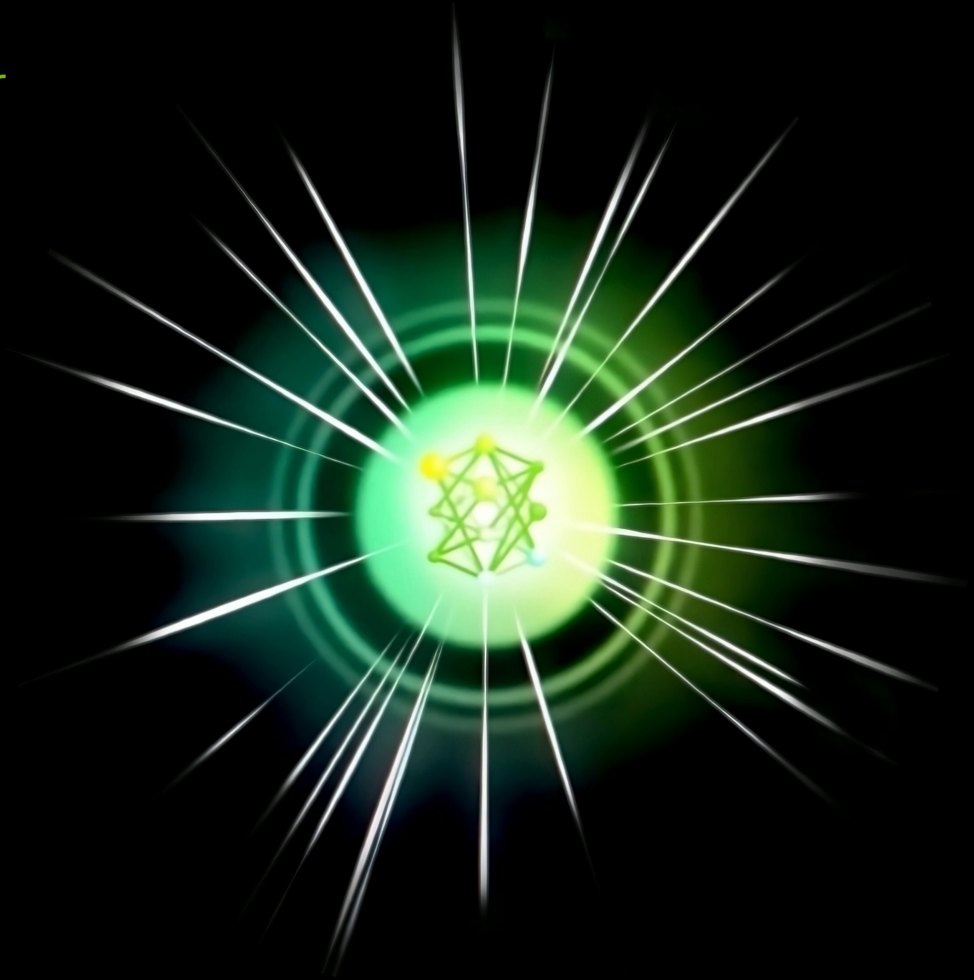
- FL enables adapting LLMs with privacy in mind.
- Fine-tuning LLMs with FL can utilize diverse distributed datasets.
- NVIDIA FLARE enables real-world collaborative LLM training with massive models (100s GB).

JOIN US AT NVIDIA FLARE DAY 2024

September 18th

Webinar

Check the news!



https://nvidia.github.io/NVFlare

Try it out at

https://github.com/NVIDIA/NVFlare

Thank you!

Holger Roth <hroth@nvidia.com>