

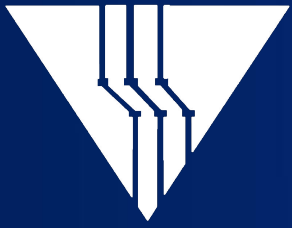


Bridges-2 Webinar

Utilizing Bridges-2 for State-of-the-Art Open Source Large Language Models

Mei-Yu Wang

Pittsburgh Supercomputing Center



Pittsburgh Supercomputing Center

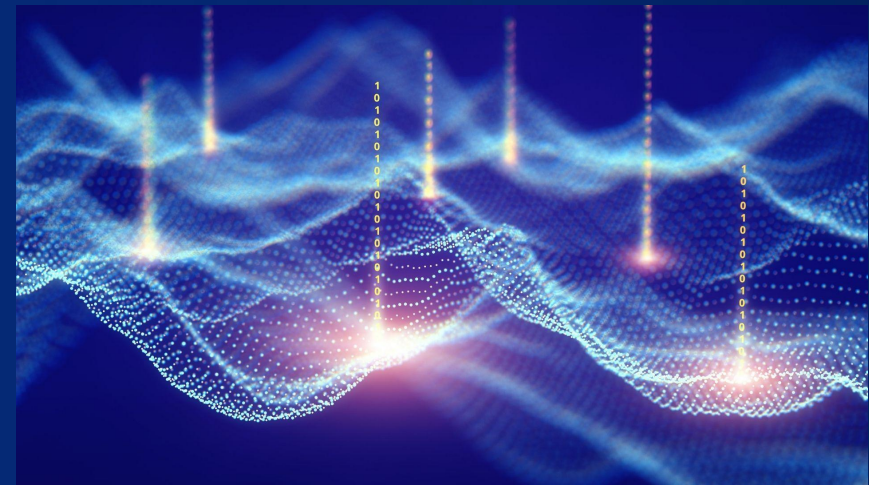
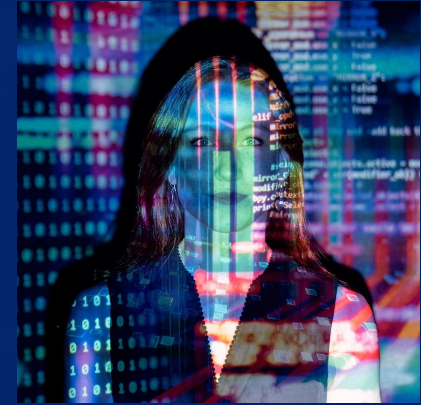
enabling discovery since 1986

The **Pittsburgh Supercomputing Center (PSC)** provides advanced research computing capability, education, and expertise to the national research community.

Since 1986, PSC has provided university, government, and industry researchers with access to some of the most powerful systems available for high-performance computing, enabling discovery across all fields of science.

OUR AREAS OF EXPERTISE

- high-performance and data-intensive computing
- data management technologies
- software architecture, implementation, and optimization
- enabling ground-breaking science, computer science, and engineering
- user support for all phases of research and education
- STEM outreach in data science, bioinformatics, and coding



Welcome!



PITTSBURGH SUPERCOMPUTING CENTER

Hewlett Packard Enterprise

NSF

Carnegie Mellon University

Bridges-2

Bridges-2 Overview

Bridges-2 is a unique computational platform for scientific and engineering research for national grid. It consists of custom artificial intelligence (AI) high performance computing (HPC), big data, and big data analytics.

Bridges-2 is available at no cost to researchers across the United States and their international counterparts, as well as to industry.

The Bridges-2 project is led by principal investigators Steven J. Benson, Patrick A. Rodriguez, Sergio G. Swales, and Robin W. Bustin, funded by NSF's Injury Prevention Area.

Bridges-2 is supported by National Science Foundation award 1008007. The Bridges-2 system was delivered by Hewlett-Packard Enterprise.

 **Hewlett Packard Enterprise** is delivering *Bridges-2*

Bridges-2 Leadership Team



Sergiu Sanielevici
PI & Dir. Support
for Sci. Apps.



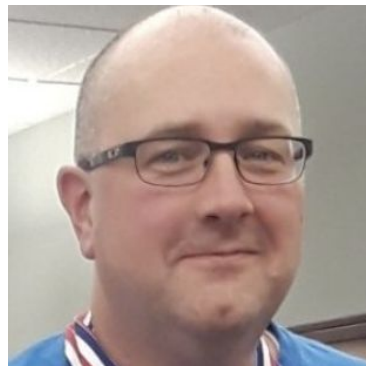
Robin Scibek
Dir. Comms.
co-PI



Paola Buitrago
Dir. AI & Big Data
co-PI



Edward Hanna
Dir. Systems & Ops.
co-PI



Tom Maiden
User Services Mgr.
co-PI



Stephen Deems
Project Manager



Andrew Adams
Information Security
Officer

Bridges-2 Webinars

- A forum for the Bridges-2 community to learn and share ideas and achievements: [Bridges-2 Webinar series | PSC](#)
- Topics and speakers of interest to work that is being done, or that may be done in future.
- Please suggest future speakers (including from your own team) and/or topics (including your own)!

Just email: sergiu@psc.edu

Introducing today's presenter: Dr. Mei-Yu Wang

Mei-Yu Wang acquired her Ph.D. in astrophysics from the University of Pittsburgh. Her doctoral research focused on developing novel probes for studying dark matter. She did postdoctoral research in studying dark matter and the Milky Way at the Texas A&M University and Carnegie Mellon University before she joined the HPC AI and Big Data Group group at PSC in 2022. Her primary roles now include addressing support requests and developing tests and benchmarking for the [Neocortex](#) system and the [Open Compass](#) project.

Q&A Logistics

- **We abide by <https://support.access-ci.org/code-of-conduct>**
- All of us except Mei-Yu will be muted during his presentation.
- Please type your questions into the Zoom chat.
- We may be able to address some questions in the chat while Mei-Yu is presenting.
- When Mei-Yu finishes her presentation, she will answer questions live during the final ~10 minutes of this webinar.
- For any remaining or follow-up questions, Mei-Yu may engage after the webinar: *mwang7@psc.edu*

Outline

- **Two examples of popular open LLMs: Llama & Gemma**
 - Llama (Meta)
 - Gemma (Google)
- **Examples of techniques for fine-tuning models with limited resources**
 - Parameter-efficient fine-tuning (PEFT)
 - Quantization
 - Fully Sharded Data Parallel
- **Brief Overview of the Bridges-2 GPU partition**
 - Type of GPU nodes/partitions
 - Batch job/interactive mode/OnDemand
 - How to set up the environment
- **Demo : performing model finetuning/inference with Llama-2 7B and Llama-3 8B (optionally Gemma 7B) using LoRA.**
- **Conclusion**

Outline

- **Two examples of popular open LLMs: Llama & Gemma**
 - Llama (Meta)
 - Gemma (Google)
- **Examples of techniques for doing Parameter-efficient fine-tuning**
 - LoRA
 - Quantization
 - Fully Sharded Data Parallel
- **Brief Overview of the Bridges-2 GPU partition**
 - Type of GPU nodes/partitions
 - Batch job/interactive mode/OnDemand
 - How to set up the environment
- **Demo : Finetuning/inferencing Gemma 7B & Llama-3 8B**
- **Conclusion**

Llama

- **Llama (Large Language Model Meta AI)** is a family of autoregressive large language models released by Meta AI starting in February 2023.

Name	Release Date	Number of Parameters	Context Length	Corpus size	Commercial viability
LLaMA-2	July 18, 2023	<ul style="list-style-type: none">● 7B/7B-chat● 13B/13B-chat● 70B/70B-chat	2049	2T	Yes
LLaMA-3	April 18, 2024	<ul style="list-style-type: none">● 8B/8B-Instruct● 70B/70B-Instruct	8912	15T	Yes

10

- **Other variants**

- **Code Llama:** a collection of code-specialized versions of Llama 2 in three flavors (base model, Python specialist, and instruct tuned).
- **Llama Guard:** a 7B Llama 2 safeguard model for classifying LLM inputs and responses.



Llama 3 Performance

Meta Llama 3 Instruct model performance

	Meta Llama 3 8B	Gemma 7B - It Measured	Mistral 7B Instruct Measured		Meta Llama 3 70B	Gemini Pro 1.5 Published	Claude 3 Sonnet Published
MMLU 5-shot	68.4	53.3	58.4	MMLU 5-shot	82.0	81.9	79.0
GPQA 0-shot	34.2	21.4	26.3	GPQA 0-shot	39.5	41.5 CoT	38.5 CoT
HumanEval 0-shot	62.2	30.5	36.6	HumanEval 0-shot	81.7	71.9	73.0
GSM-8K 8-shot, CoT	79.6	30.6	39.9	GSM-8K 8-shot, CoT	93.0	91.7 11-shot	92.3 0-shot
MATH 4-shot, CoT	30.0	12.2	11.0	MATH 4-shot, CoT	50.4	58.5 Minerva prompt	40.5

See https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md#benchmarks for benchmark results.

Source: <https://llama.meta.com/llama3>

Resources for getting started with Llama

- **Website**

- <https://llama.meta.com/>

- **Ways to download the model:**

- Meta: <https://llama.meta.com/llama-downloads>
- Hugging face: <https://huggingface.co/meta-llama>
- Kaggle: <https://www.kaggle.com/models/meta-research/llama-3>,
<https://www.kaggle.com/models/meta-research/llama-2>

- **Github**

- Meta-llama/llama-recipes
 - <https://github.com/meta-llama/llama-recipes>
 - examples to get started using Llama for fine-tuning, inference...etc.
- Meta-llama/llama
 - <https://github.com/meta-llama/llama>
 - Provide a script for downloading the model weights and a minimal example to load models and run inference.
- Torchtune
 - <https://github.com/pytorch/torchtune>



Gemma

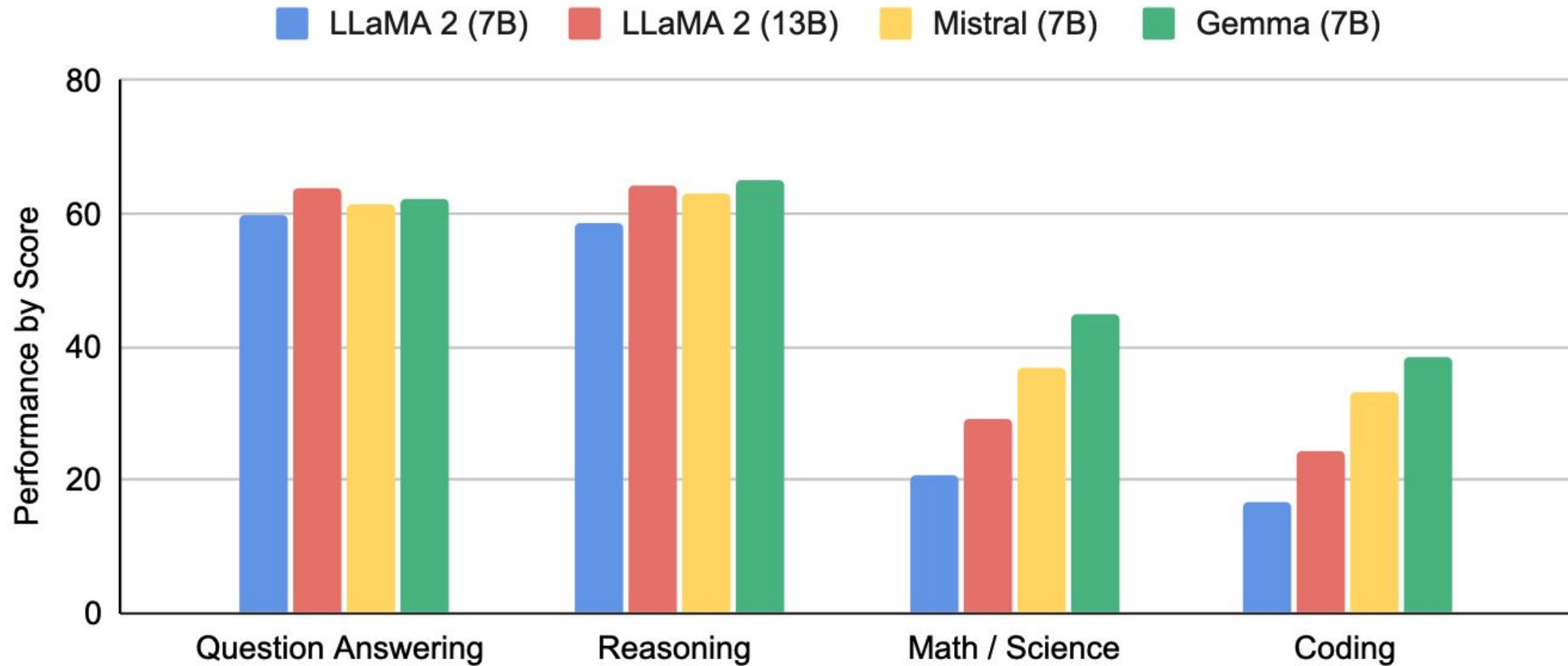
- Gemma is a family of lightweight, state-of-the-art open models built from the same research and technology used to create the Gemini models, developed by Google DeepMind and other teams across Google.



Name	Release Date	Number of Parameters	Context Length	Corpus size	Commercial viability
Gemma	February 21, 2024	<ul style="list-style-type: none">● 2B/2B-it (v1.0, v1.1)● 7B/7B-it (v1.0, v1.1)	8192	6T	Yes

- Other variants:
 - **CodeGemma**: a collection of code-specialized versions of Gemma.
 - **PaliGemma**: an open vision-language model built with open components such as the SigLIP vision model and the Gemma language model.
 - **RecurrentGemma**: an open model based on Griffin, a hybrid model that mixes gated linear recurrences with local sliding window attention.

Gemma Performance



Source: Gemma technical report

<https://storage.googleapis.com/deepmind-media/gemma/gemma-report.pdf>

Resources for getting started with Gemma

- **Website:** <https://ai.google.dev/gemma>
- **Ways to download the model:**
 - Kaggle: <https://www.kaggle.com/models/google/gemma>
 - Hugging Face: <https://huggingface.co/google>
- **Technical report:** <https://storage.googleapis.com/deepmind-media/gemma/gemma-report.pdf>
- **Github:**
 - google-deepmind/gemma:
 - <https://github.com/google-deepmind/gemma>
 - examples to get started using Gemma for fine-tuning, inference...etc.
 - For tutorials, reference implementations in various ML frameworks:
 - <https://github.com/google/generative-ai-docs/tree/main/site/en/gemma/docs>
 - TorchTune
 - <https://github.com/pytorch/torchTune>



kaggle™



Examples of scientific applications with Llama

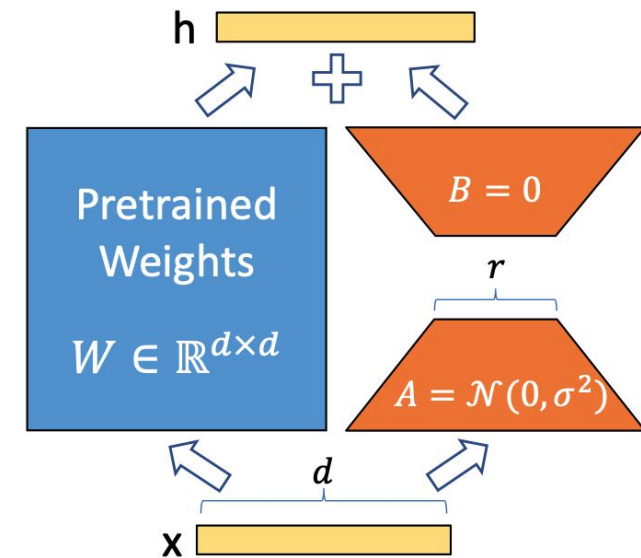
- **Fine Tuned with domain specific knowledge:**
 - **Medical specific LLMs:** finetuned with medical paper (arXiv:2304.14454), medical conversational model (arXiv:2304.08247) , clinical data: (arxiv:2307.03042), Medical application: (arxiv:2402.12749)
 - **Biochemistry:** (arxiv:2306.08018)
 - **Finance:** FinLlama (arxiv:2403.12285)
- **Retrieval Augmented Generation (RAG)**
 - **Medical:** Disease prediction system (arxiv:2402.00746), PMC-LLaMA (arxiv:2304.14454)

Outline

- Two examples of popular open-source LLMs: Gemma & Llama
 - Gemma
 - Llama-3/Llama-2
- **Examples of techniques for fine-tuning models with limited resources**
 - Parameter-efficient fine-tuning (PEFT)
 - Quantization
 - Fully Sharded Data Parallel
- **Brief Overview of the Bridges-2 GPU partition**
 - Type of GPU nodes/partitions
 - Batch job/interactive mode/OnDemand
 - How to set up the environment
- **Demo : Finetuning/inferencing Gemma 7B & Llama-3 8B**
- **Conclusion**

Parameter-efficient fine-tuning (PEFT)

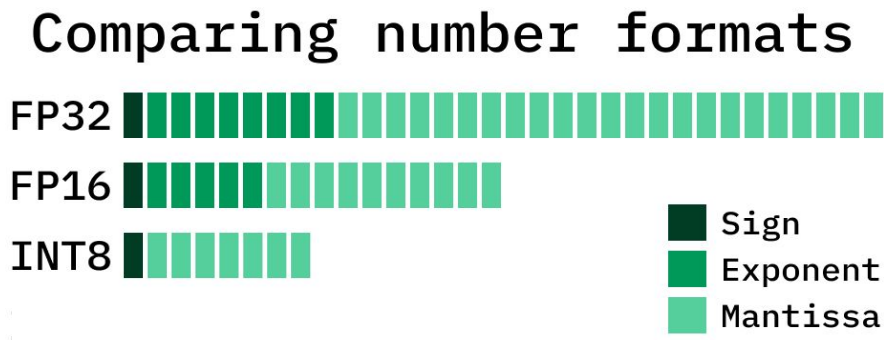
- In traditional fine-tuning, all model parameters are updated to tailor the outputs to the specific task. It is also possible to freeze some layers and leave the rests trainable.
- In contrast, when fine-tuning with PEFT (Parameter-Efficient Fine-Tuning), the base model weights remain frozen, and only the adapter modules are trained. Consequently, the number of trainable parameters could be drastically reduced to less than 1%.
- Examples:
 - **LoRA**
 - **P-tuning**
 - **Prefix tuning**



Hu et al., (2021) "LoRA: Low-Rank Adaptation of Large Language Models" <https://arxiv.org/abs/2106.09685>

Quantization

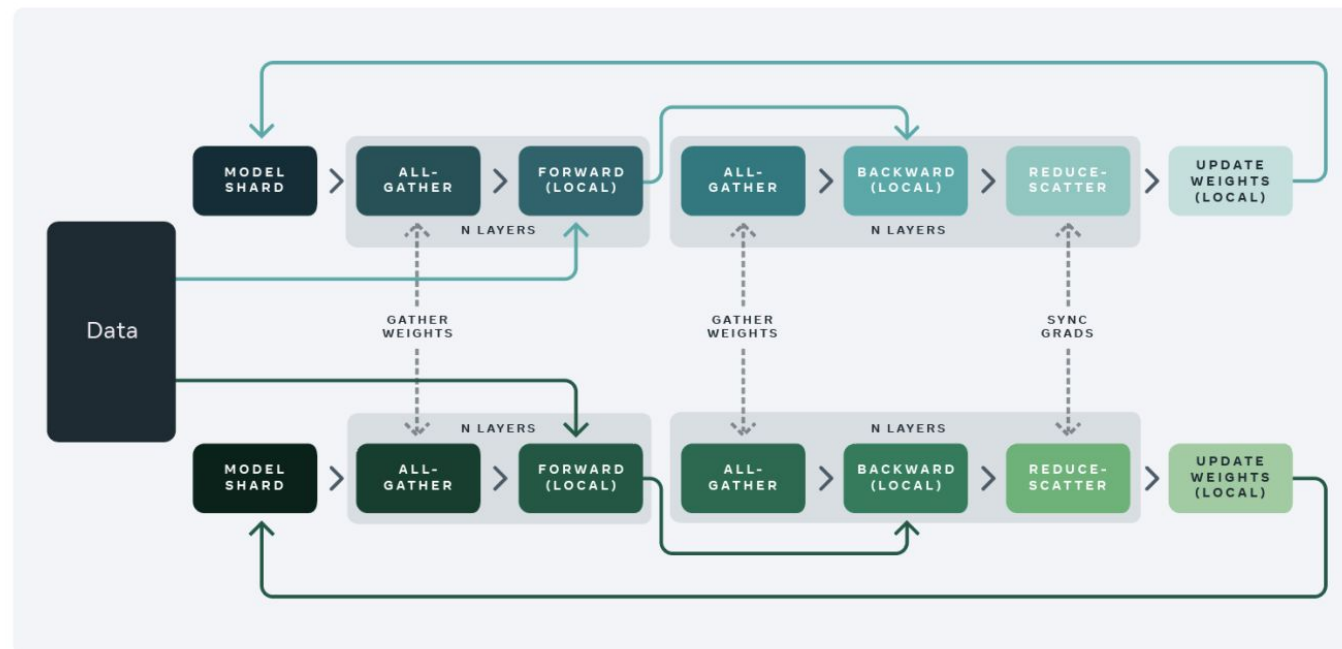
- Quantization involves representing model weights and activations, typically 32-bit floating numbers, with lower precision data such as 8-bit int or 4-bit int.
- The benefits of quantization include smaller model sizes, faster fine-tuning, and faster inference—particularly beneficial in resource-constrained environments.
- However, the tradeoff is a reduction in model quality due to the loss of precision.
- Example library:
 - [BitsAndBytes](#)
 - [Quanto](#)
 - [TorchAO](#)



Fully Sharded Data Parallel (FSDP)

- Unlike traditional data-parallel, which maintains a per-GPU copy of a model's parameters, gradients and optimizer states, FSDP shards all of these states across data-parallel workers and can optionally offload the sharded model parameters to CPUs.
- It is available in PyTorch and is Integrated with Hugging Face Accelerate
- Paper: <https://arxiv.org/pdf/2304.11277>

Fully sharded data parallel training



source:
<https://engineering.fb.com/2021/07/15/open-source/fsdp/>

Outline

- Two examples of popular open-source LLMs: Gemma & Llama
 - Gemma
 - Llama-3/Llama-2
- Examples of techniques for fine-tuning with limited resources
 - Parameter-efficient fine-tuning (PEFT)
 - Quantization
 - Fully Sharded Data Parallel
- **Brief Overview of the Bridges-2 GPU partition**
 - Type of GPU nodes/partitions
 - Batch job/interactive mode/OnDemand
 - How to set up the environment
- Demo : Finetuning/inferencing Gemma 7B & Llama-3 8B
- Conclusion

Bridges-2 GPU and GPU-shared partitions

- See Bridges-2 user guide for details:

<https://www.psc.edu/resources/bridges-2/user-guide/#gpu-partitions>

- **Partition:**

- The **GPU-shared** partition

The GPU-shared partition is for jobs that will use **part of one GPU node** (up to **4** GPUs, maximum runtime: **48 hours**).

- The **GPU** partition

The GPU partition is for jobs that will use **one or more entire GPU nodes** (up to **64** GPUs, maximum runtime: **48 hours**)

- **GPU type:**

Node Type	Total # of nodes	# GPUs per node	Memory per GPU	RAM per node
V100-32	24 Tesla V100-32GB SXM2	8	32 GB	512 GB
	1 DGX-2	16	32 GB	1.5 TB
V100-16	8 V100-16GB	8	16 GB	192 GB

How to run jobs on Bridges-2

- See Bridges-2 user guide for details:

<https://www.psc.edu/resources/bridges-2/user-guide/#running-jobs>

- **Batch Mode** (<https://www.psc.edu/resources/bridges-2/user-guide/#batch-jobs>)

Using slurm scripts to submit jobs to the queue so that they will run as soon as resources are available.

- **Interactive Sessions** (<https://www.psc.edu/resources/bridges-2/user-guide/#interactive-sessions>)

Where you type commands and receive output back to your screen as the commands complete. Best for debugging and short test jobs (maximum requested time is up to **8 hours**).

- **OnDemand** (<https://www.psc.edu/resources/bridges-2/user-guide/#ondemand>)

A web browser interface that allows you to run interactively, or create, edit and submit batch jobs and also provides a graphical interface to tools like RStudio, Jupyter notebooks, and IJulia

How to set up the environments for AI/ML applications

- **See Bridges-2 user guide for details:**
 - **PSC Pre-Built AI module** (<https://www.psc.edu/resources/bridges-2/user-guide/#ai-environments>)
Pre-built AI environment (Anaconda-based) including several popular AI/ML/BD packages.
 - **NVIDIA NGC containers** (<https://www.psc.edu/resources/software/singularity/>)
Containers developed by NVIDIA that are performance-optimized and ready to deploy for AI/ML applications on GPU-powered systems.
 - **Create your own Conda environment/custom AI environment**
<https://www.psc.edu/resources/bridges-2/user-guide/#using-a-conda-module-environment>
<https://www.psc.edu/resources/bridges-2/user-guide/#ai-environments>
 - **Create your own Singularity container**
<https://www.psc.edu/resources/bridges-2/user-guide/#using-singularity-containers>

Ways to run deep learning jobs

Interactive Sessions

- **Commands to start interactive sessions:**

- For GPU-shared partition:

```
interact --partition GPU-shared --gres=gpu:type:n -t time
```

- example: `interact -p GPU-shared --gres=gpu:v100:2 -t 2:00:00`

- For GPU partition:

```
interact --partition GPU --gres=gpu:type:n -N x -t time
```

- example: `interact -p GPU --gres=gpu:v100-32:8 -N 1 -t 1:00:00`

- **Rules:**

- **--partition:** GPU-shared Or GPU

- **--gres=gpu:type:n**

- **type:** v100-32 or v100-16. Use v100 if node type is not specified.

- **n: number of GPUs per node.** For GPU partition, n must be either 8 or 16 for DGX-2. For GPU-shared partition, n should be less than 4.

- **-t:** requested walltime, in the format HH:MM:SS

- **-N/--nodes:** number of nodes

- See <https://www.psc.edu/resources/bridges-2/user-guide/#gpu-partitions> for more details

Batch mode

Example slurm script (NGC container)

```
#!/bin/bash
#SBATCH -p GPU-shared
#SBATCH -t 2:00:00
#SBATCH --gpus=v100:4
#SBATCH --account=xxxxxxx      # Please change it to your allocation ID

#type 'man sbatch' for more information and options
#this job will ask for 4 V100 GPUs for 2 hours (node type not specified)

#echo commands to stdout
set -x

# move to working directory
cd /ocean/projects/groupname/username/path-to-directory

#run the program which is already in your project space
singularity exec --nv /ocean/containers/ngc/pytorch/pytorch_latest.sif python3 pytorch_test.py
```

Rules:

Similar to interactive sessions, but use `--gpus=type:n` instead to specify **total number of GPUs** for `n` and node types.

To submit slurm script, type `sbatch name_of_your_script`

See

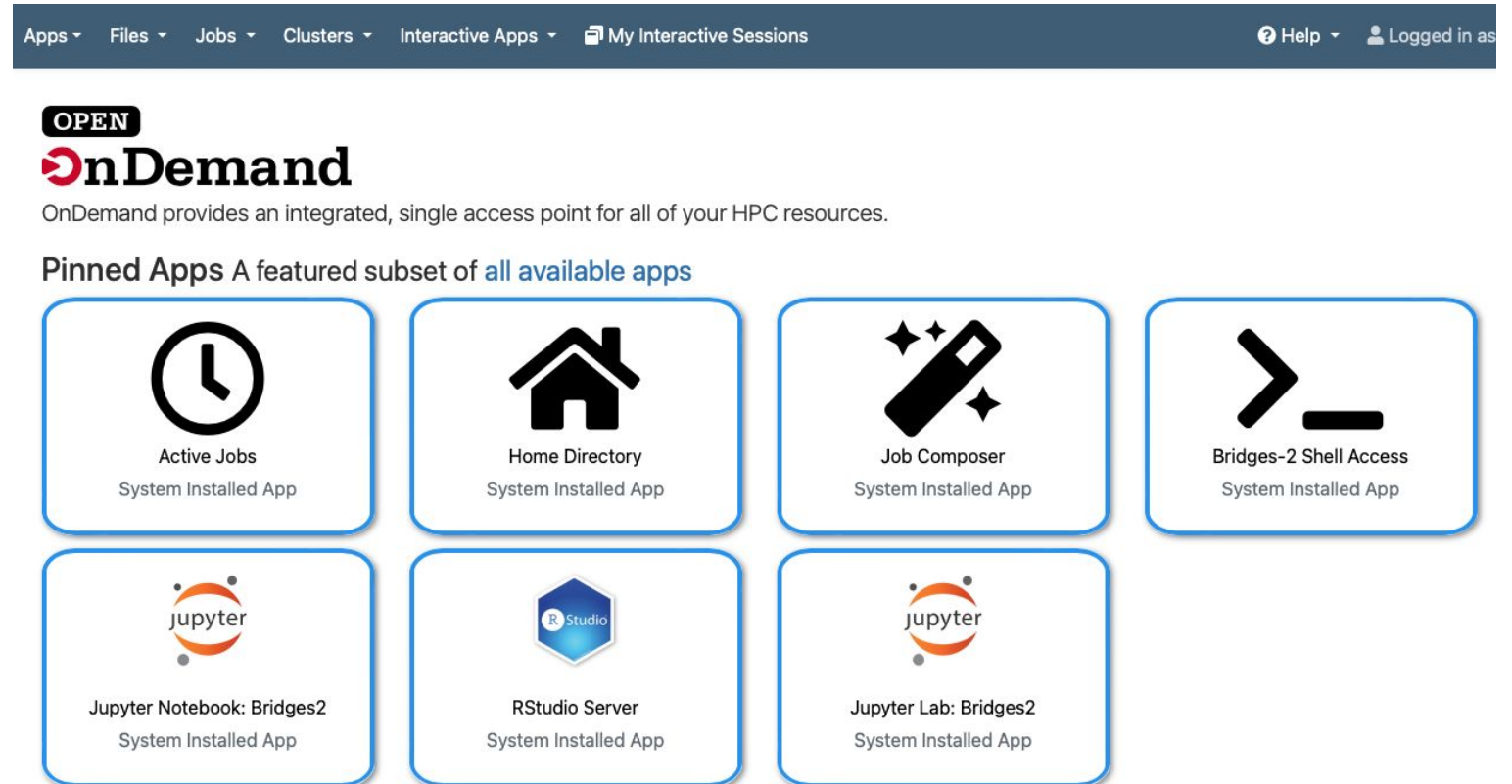
<https://www.psc.edu/resources/bridges-2/user-guide/#batch-jobs> about sbatch commands and options

See

<https://www.psc.edu/resources/bridges-2/user-guide/#gpu-partitions> for more details about GPU batch jobs.

Using OnDemand to run Jupyter notebooks

- Open <https://ondemand.bridges2.psc.edu> via a web browser. Enter your PSC username and password.
- Once logged in, click on “Jupyter Notebook: Bridges2” or go to “Interactive Apps -> Jupyter Notebook”



The screenshot shows the OnDemand web interface. At the top, there is a navigation bar with links for Apps, Files, Jobs, Clusters, Interactive Apps, and My Interactive Sessions. On the right side of the navigation bar, there are links for Help and Logged in as. Below the navigation bar, the OnDemand logo is displayed with the word "OPEN" in a black box above it. The text "OnDemand" is in a large, bold font. Below the logo, a description states: "OnDemand provides an integrated, single access point for all of your HPC resources." Underneath this, the text "Pinned Apps A featured subset of all available apps" is shown. There are seven app tiles arranged in two rows. The first row contains four tiles: "Active Jobs" (clock icon), "Home Directory" (house icon), "Job Composer" (wand icon), and "Bridges-2 Shell Access" (shell icon). The second row contains three tiles: "Jupyter Notebook: Bridges2" (Jupyter logo), "RStudio Server" (RStudio logo), and "Jupyter Lab: Bridges2" (Jupyter logo). Each tile includes the app name and "System Installed App" below it.

OnDemand

- Enter the information about your job, such as requested time, number of nodes, partition, and specify the number of GPUs using the “Extra Slurm Args” column (similar to typical batch job).
- Click “Launch” to submit the job

The screenshot shows the Bridges2 OnDemand web interface. At the top, there is a navigation bar with 'Bridges2 OnDemand', 'Files', 'Jobs', 'Clusters', 'Interactive Apps', 'My Interactive Sessions', and 'Help'. Below this is a breadcrumb trail: 'Home / My Interactive Sessions / Jupyter Notebook'. On the left, a sidebar menu shows 'Interactive Apps' with sub-items 'Servers', 'Jupyter Lab', 'Jupyter Notebook' (highlighted), and 'RStudio Server'. The main content area is titled 'Jupyter Notebook version: v1.0.1-3-g94d29b4'. It contains several form fields: 'Number of hours' (set to 1), 'Number of nodes' (set to 1), 'Account' (set to pscstaff), 'Partition' (set to GPU-shared), and 'Extra Slurm Args' (set to --gpus=v100-32:2). There is also an 'Extra Jupyter Args' field and a checkbox for 'I would like to receive an email when the session starts'. A blue 'Launch' button is at the bottom. A footer note states: '* The Jupyter Notebook session data for this session can be accessed under the data root directory.'

- Once the job starts, click the “Connect to Jupyter” to launch the Jupyter notebook interface

The screenshot displays the Bridges2 OnDemand web interface. At the top, a dark blue navigation bar contains the text "Bridges2 OnDemand" and several menu items: "Files", "Jobs", "Clusters", "Interactive Apps", and "My Interactive Sessions". On the right side of this bar, there is a "Help" icon and the text "Logged in as mwang7".

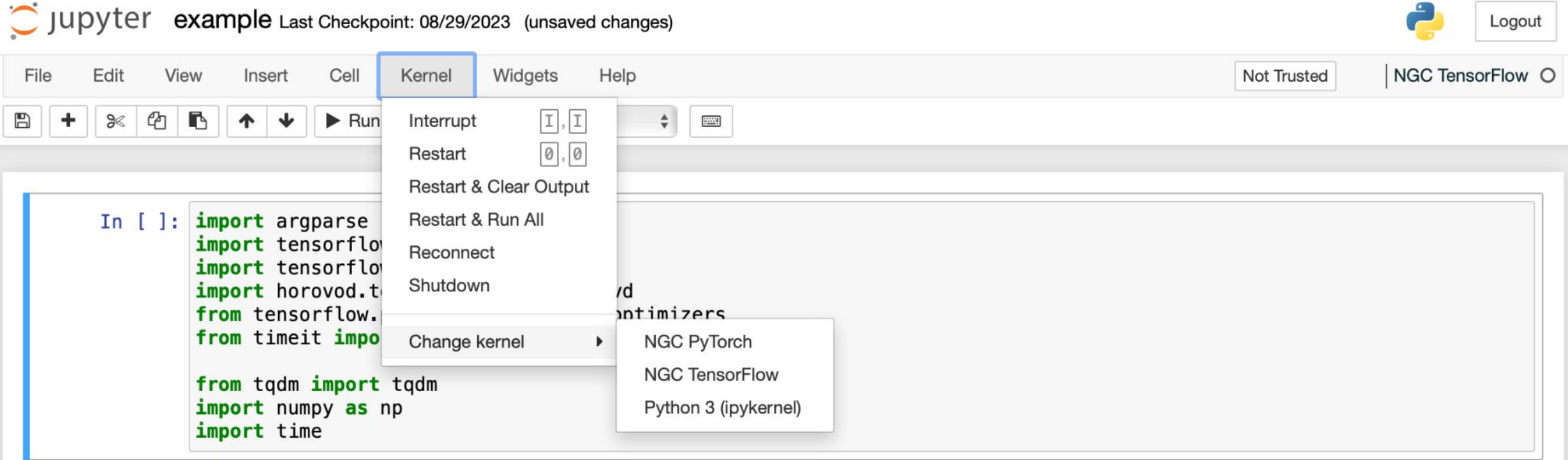
Below the navigation bar, a light green notification banner states "Session was successfully created." with a close button (X) on the right. Underneath this is a breadcrumb trail: "Home / My Interactive Sessions".

The main content area is divided into two sections. On the left is a sidebar menu with the following items: "Interactive Apps", "Servers", "Jupyter Lab", "Jupyter Notebook", and "RStudio Server". The "Jupyter Notebook" item is currently selected.

The right section displays the details for a "Jupyter Notebook (19032560)" session. At the top right of this section, it shows "1 node | 5 cores | Running". The session details include: "Host: >_v005.ib.bridges2.psc.edu" with a red "Delete" button to its right; "Created at: 2023-09-12 14:23:21 EDT"; "Time Remaining: 59 minutes"; and "Session ID: 91d342ef-fb4b-4955-a4c3-d827db832eb1". At the bottom of this section, a blue button labeled "Connect to Jupyter" is highlighted with a red rectangular box.

OnDemand

- You can use NGC containers for Pytorch and Tensorflow (latest) by selecting them from the “Kernel -> Change kernel -> NGC PyTorch/NGC TensorFlow”
- To use custom conda environment/containers, please check the Bridges-2 User Guide: <https://www.psc.edu/resources/bridges-2/user-guide/#custom-env>



The screenshot shows the JupyterLab interface. At the top left, the Jupyter logo is followed by the text "example Last Checkpoint: 08/29/2023 (unsaved changes)". On the top right, there is a Python logo and a "Logout" button. Below this is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". To the right of the menu bar, there is a "Not Trusted" indicator and a "Kernel" dropdown menu currently showing "NGC TensorFlow". The "Kernel" menu is open, showing options: "Interrupt", "Restart", "Restart & Clear Output", "Restart & Run All", "Reconnect", "Shutdown", and "Change kernel". The "Change kernel" option is selected, opening a sub-menu with three options: "NGC PyTorch", "NGC TensorFlow", and "Python 3 (ipykernel)". In the background, a code cell is visible with the following code:

```
In [ ]: import argparse
import tensorflow
import tensorflow
import horovod.t
from tensorflow.
from timeit impo

from tqdm import tqdm
import numpy as np
import time
```

Demo:

Performing model finetuning/inference for Llama 2-7B, Llama 3-8B and Gemma 7B with LoRA

See

<https://github.com/pscedu/bridges2-examples/tree/main/bridges2-llm-examples>

for detailed instructions and scripts/Jupyter notebooks

Summary

- The recent release of open LLMs such as Llama and Gemma provides a big step towards democratizing LLM usage.
- There are various techniques for fine-tuning LLMs with limited computational resources, such as various Parameter-efficient fine-tuning (PEFT) techniques, quantization, and fully shared data parallel methods.
- For Bridges-2 GPU partition, it is best to utilize the V100 32GB GPUs to work with LLMs. Bridges-2 also provides various ways to run jobs, such as batch/interactive mode and OnDemand web interface to easily set up and run Jupyter notebooks.
- We provide examples and instructions for doing model finetuning/inference with Llama/Gemma on Bridges-2: <https://github.com/pscedu/bridges2-examples/tree/main/bridges2-llm-examples>
- Please email **help@psc.edu** with any general questions regarding Bridges-2. You can also reach me by email **mwang7@psc.edu**.