

HIGH PERFORMANCE RESEARCH COMPUTING

Programming Using the Jupyter AI Assistant

July 19, 2025
Keegan Smith



High Performance
Research Computing
DIVISION OF RESEARCH



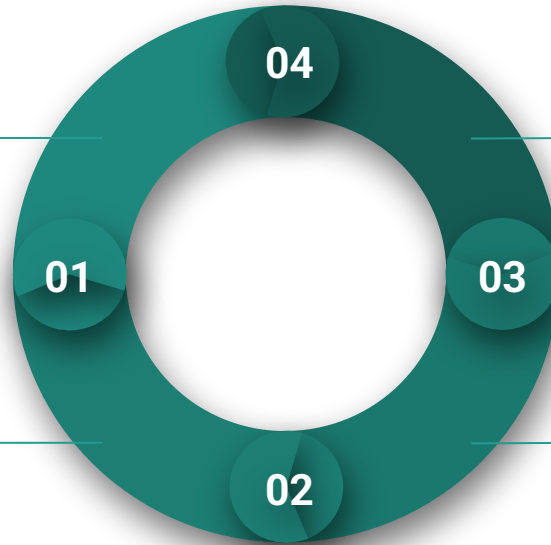
Schedule

Lab 1: Creating a session (15 minutes)

We will create a jupyter AI Assistant session from the ACES portal.

Lab 2: Debugging with Jupyter AI (15 mins)

We will go through some examples with two popular Python libraries: Pandas and Matplotlib for data exploration.



Lab 3: Creating a PyTorch Application (15 minutes)

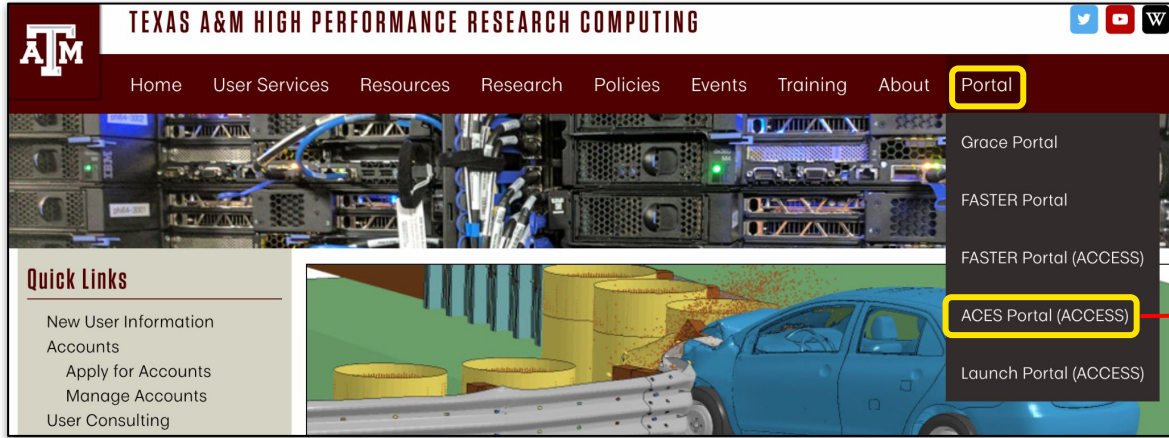
We will create and debug a Pytorch application from scratch with the assistance of Jupyter AI.

Lab 4: Creating a Tensorflow Application (15 minutes)

We will create and debug a Tensorflow application from scratch with the assistance of Jupyter AI.

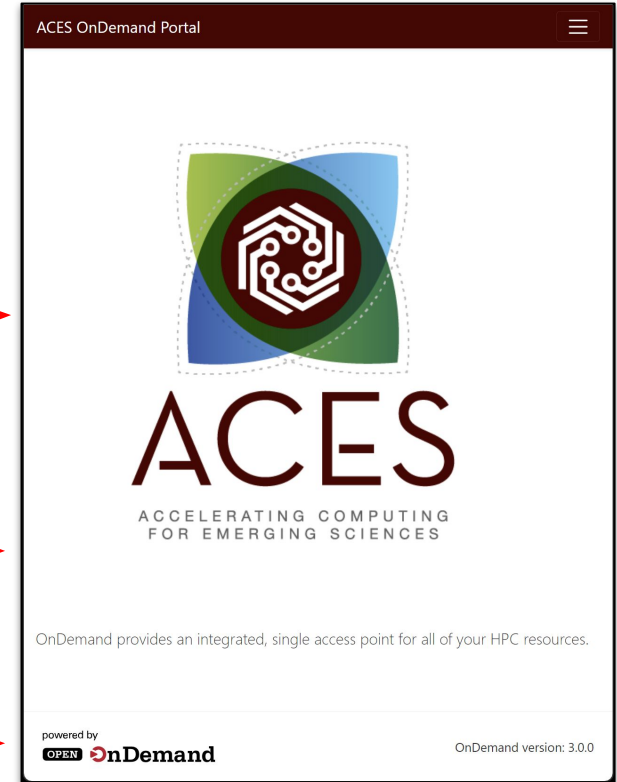


ACES Portal

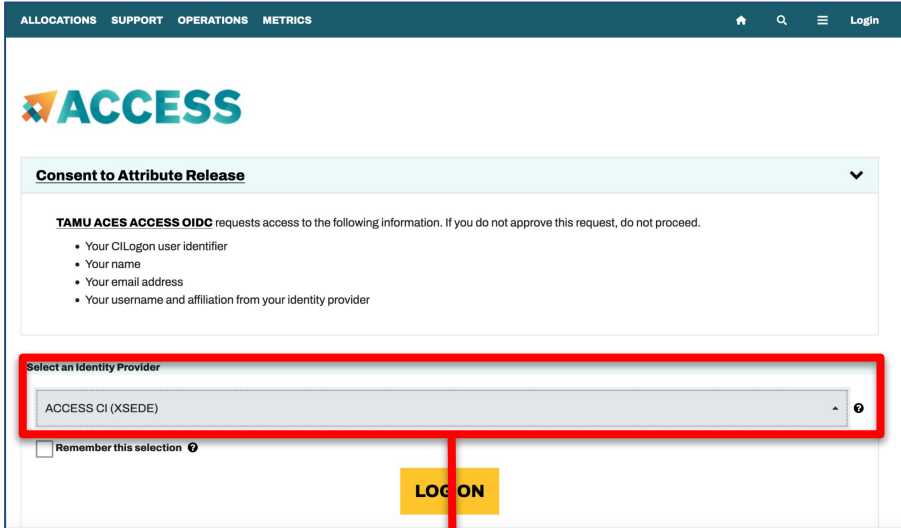


ACES Portal portal-aces.hprc.tamu.edu
is the web-based user interface for the ACES cluster

Open OnDemand (OOD) is an advanced web-based
graphical interface framework for HPC users

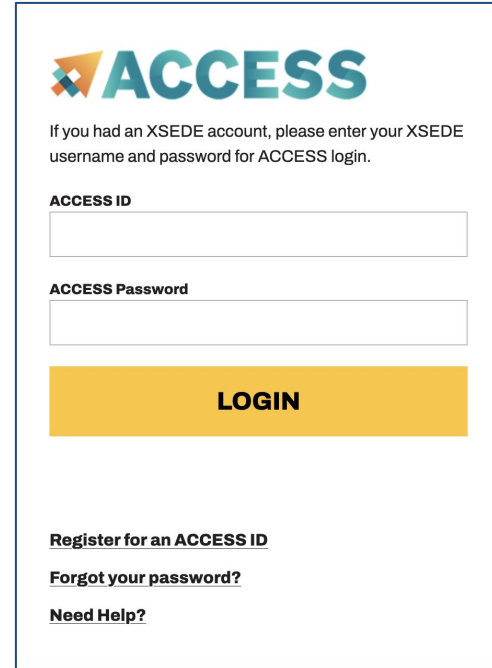


Accessing ACES via the Portal (ACCESS)



The screenshot shows the ACCESS portal interface. At the top is a navigation bar with links: ALLOCATIONS, SUPPORT, OPERATIONS, METRICS, and a Login link. Below the navigation bar is the ACCESS logo. A section titled "Consent to Attribute Release" contains a message from TAMU ACES ACCESS OIDC and a list of requested information: CILogon user identifier, name, email address, and username/affiliation. Below this is a "Select an Identity Provider" dropdown menu with "ACCESS CI (XSEDE)" selected. A red rectangle highlights this dropdown. Below the dropdown is a "Remember this selection" checkbox and a yellow "LOG ON" button. A red line points from the "LOG ON" button to the text below.

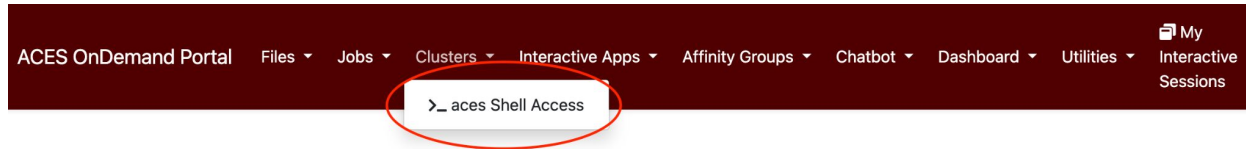
Select the Identity Provider appropriate for your account.



The screenshot shows the ACCESS portal login page. At the top is the ACCESS logo. Below it is a message: "If you had an XSEDE account, please enter your XSEDE username and password for ACCESS login." There are two input fields: "ACCESS ID" and "ACCESS Password". Below these fields is a yellow "LOGIN" button. At the bottom, there are links: "Register for an ACCESS ID", "Forgot your password?", and "Need Help?".

Log-in using your ACCESS or institutional credentials.

Setting up



OnDemand provides an integrated, single access point for all of your HPC resources.

Message of the Day

IMPORTANT POLICY INFORMATION

- Unauthorized use of HPRC resources is prohibited and subject to criminal prosecution.
- Use of HPRC resources in violation of United States export control laws and regulations is prohibited.
- Sharing HPRC account and password information is in violation of State Law. Any shared accounts will be DISABLED.
- Authorized users must also adhere to ALL policies at: <https://hprc.tamu.edu/policies>

!! WARNING: THERE ARE ONLY NIGHTLY BACKUPS OF USER HOME DIRECTORIES. !!



Setting up

In your terminal, type:

```
bash /scratch/training/jupyter_ai_intro/setup.sh
```

This will put the notebook `jupyter_ai_intro.ipynb` in your `$SCRATCH` directory.



Starting Jupyter AI Assistant

The screenshot shows the ACES OnDemand Portal interface. The top navigation bar includes links for Files, Jobs, Clusters, Interactive Apps, Affinity Groups, Chatbot, Dashboard, Utilities, My Interactive Sessions, Develop, Help, and a user profile icon. The 'Interactive Apps' dropdown menu is open, displaying a list of applications categorized into GUI, Imaging, Servers, and TEST_HPRC_ONLY. The 'Jupyter AI Assistant' option is highlighted with a red circle. Other options in the 'Servers' category include Jupyter Notebook, JupyterLab, RStudio, TensorBoard, and Tutorials OnDemand. The background of the portal shows the ACES logo, a message of the day, and important policy information.

ACES OnDemand Portal

Files Jobs Clusters Interactive Apps Affinity Groups Chatbot Dashboard Utilities My Interactive Sessions Develop Help

GUI

- MATLAB
- VNC
- NextSilicon VNC

Imaging

- CryoSPARC
- ImageJ
- Jmol
- Paraview
- cisTEM

Servers

- Jupyter AI Assistant**
- Jupyter Notebook
- JupyterLab
- RStudio
- TensorBoard
- Tutorials OnDemand

TEST_HPRC_ONLY

- Hugging (Inter)Face

ACES

OnDemand provides an integrated, secure environment for your HPC resources.

Message of the Day

IMPORTANT POLICY INFORMATION

- Unauthorized use of HPRC resources
- Use of HPRC resources in violation of State Law. Any shared accounts will be DISABLED.
- Sharing HPRC account and password
- Authorized users must also adhere to criminal prosecution.

!! WARNING: THERE ARE ONLY NIGHTLY BUILD RELEASES !!

powered by OPEN OnDemand

OnDemand version: 3.1.10

Starting Jupyter AI Assistant

Home / My Interactive Sessions / Jupyter AI Assistant

Interactive Apps

GUI

MATLAB

NextSilicon VNC

VNC

Imaging

CryoSPARC

CryoSPARC 4.2.1

ImageJ

Jmol

Paraview

cisTEM

Servers

Jupyter AI Assistant

Jupyter Notebook

JupyterLab

RStudio

Jupyter AI Assistant version: 5c10a8f

This app will launch a [JupyterLab](#) server on the [ACES cluster](#) with the HPRC custom jupyter ai extension.

Module

Python/3.11.3 (foss/2023a)

Optional python environment to be activated

Select Path

Node type

CPU only

Tutorials OnDemand

TEST_HPRC_ONLY

Hugging (Inter)Face

Interactive Apps [Sandbox]

Servers

Jupyter AI Assistant

Number of cores (max 96)

1

Total GB memory (max 488)

5

Account

This field is optional.

Email

This field is optional.

☐ I would like to receive an email when the session starts

Launch

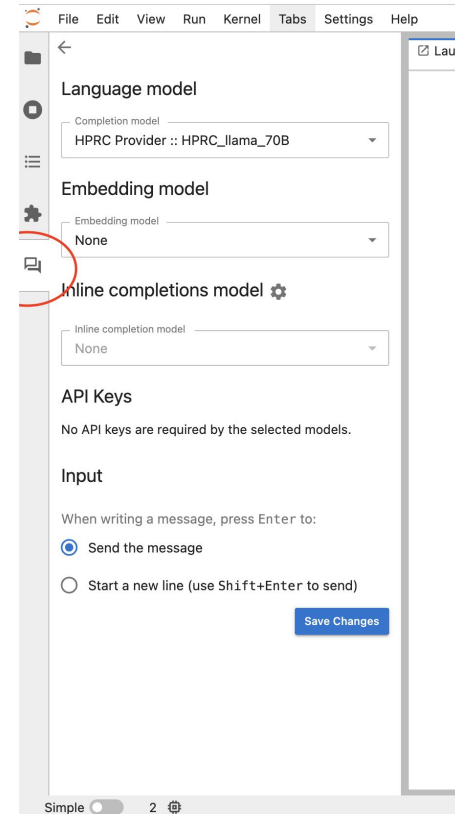
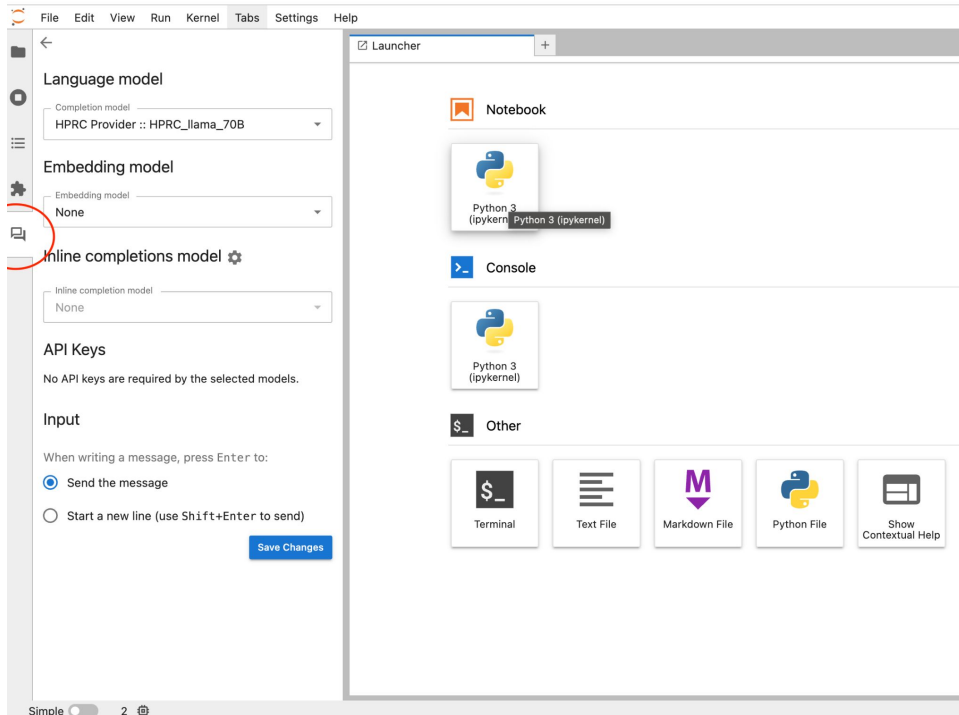
* The Jupyter AI Assistant session data for this session can be accessed under the [data root directory](#).

Python/3.11.3 (foss/2023a)



Using Jupyter AI Assistant

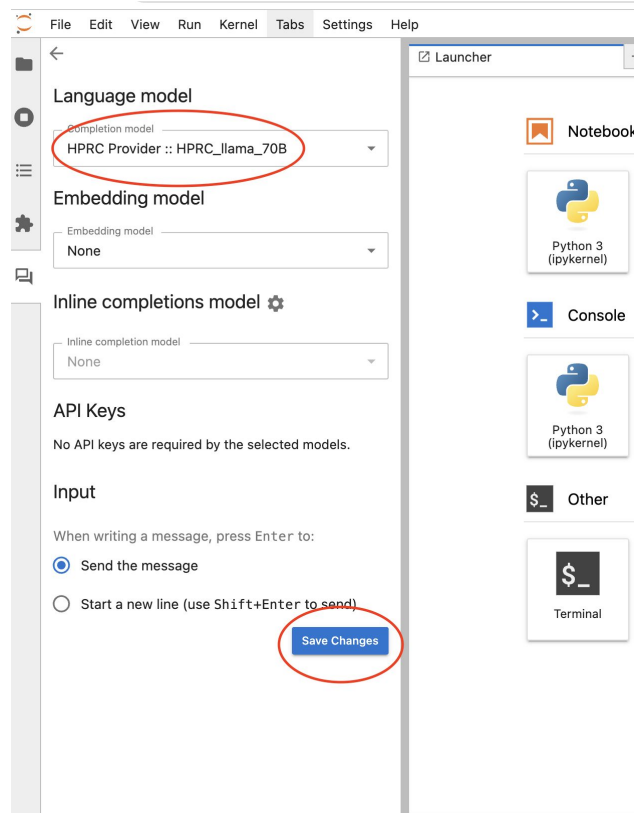
Click on the message icon on the left side of the screen



Using Jupyter AI Assistant

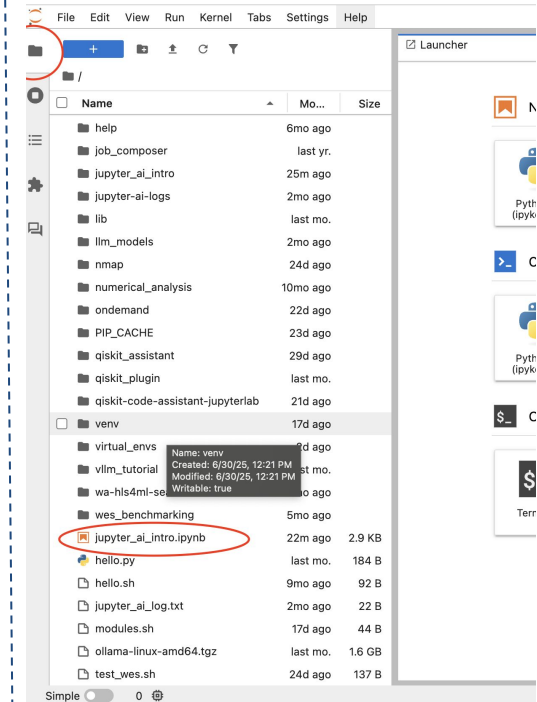
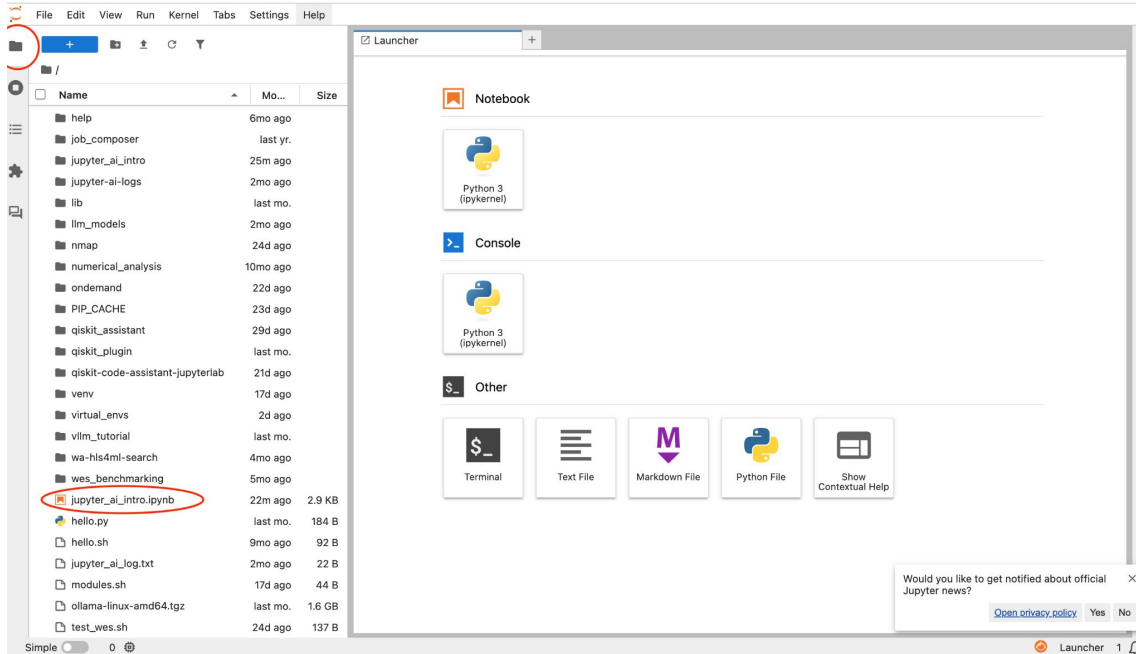
Select the “HPRC Provider
:: HPRC_llama_70B”
Completion model from
the dropdown

Click, “Save Changes”



Main Content

Open the “jupyter_ai_intro.ipynb” notebook in your jupyter ai assistant session:



Debugging a simple program

- Click on the highlighted code cell
- Run the code cell with the highlighted button at the top of the screen
- Use the `/fix` command in the chat window to help debug the issue

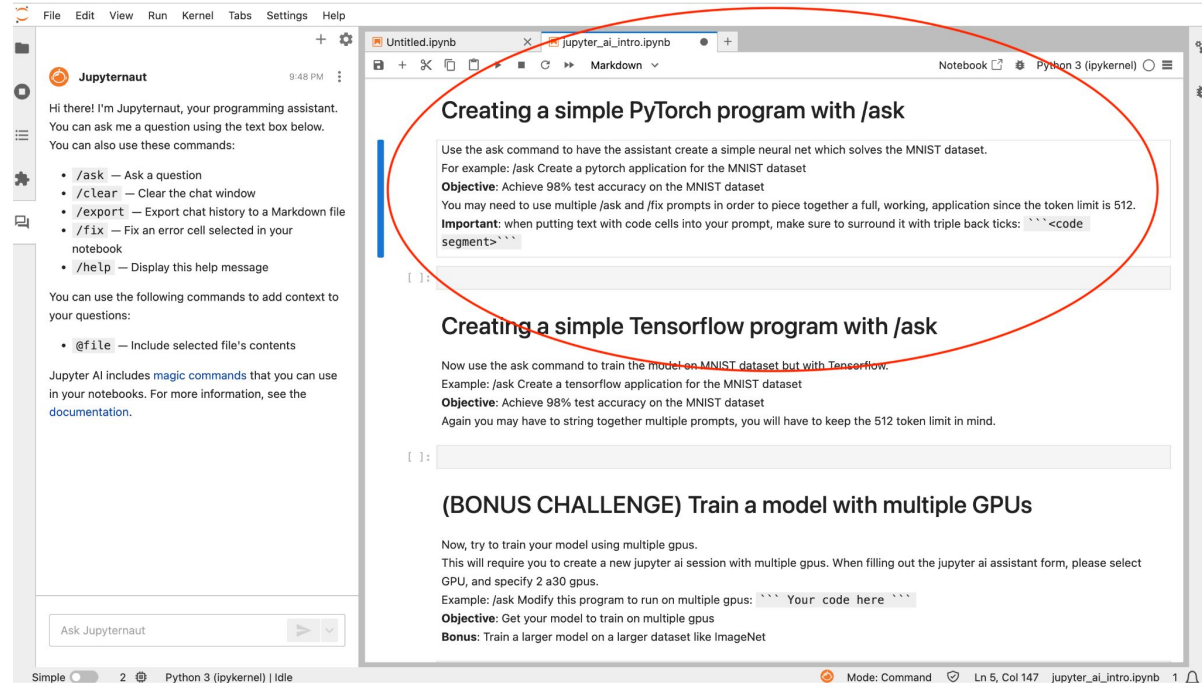
The screenshot displays the Jupyter AI interface. On the left is a chat window with Jupyter AI instructions and a list of commands: `/ask` (Ask a question), `/clear` (Clear the chat window), `/export` (Export chat history to a Markdown file), `/fix` (Fix an error cell selected in your notebook), and `/help` (Display this help message). The main area shows a Jupyter notebook with a code cell containing a Fibonacci function. The code is:

```
def fibonacci(n):  
    if n <= 1:  
        return 1  
    return fibonacci(n - 1) + fibonacci(n - 2)  
result = fibonacci(10000)  
print(result)
```

 The code cell is highlighted with a red oval, and the run button (a play icon) at the top of the cell is also circled in red. Below the code cell, there are sections titled "Creating a simple PyTorch program with /ask" and "Creating a simple Tensorflow program with /ask", each with instructions and examples of how to use the `/ask` command.

Creating a simple PyTorch program

- Use the `/ask` command to help develop code for solving the MNIST dataset with pytorch



The screenshot displays the JupyterLab environment. On the left, the Jupyter AI assistant chat window is open, showing a welcome message and a list of commands: `/ask` (Ask a question), `/clear` (Clear the chat window), `/export` (Export chat history to a Markdown file), `/fix` (Fix an error cell selected in your notebook), and `/help` (Display this help message). It also mentions that users can use commands to add context to questions, such as `@file` to include file contents.

On the right, a Jupyter Notebook titled 'jupyter_al_intro.ipynb' is open. The first cell contains the title 'Creating a simple PyTorch program with /ask' and instructions on how to use the `/ask` command to create a simple neural network for the MNIST dataset. It provides an example prompt: `/ask Create a pytorch application for the MNIST dataset` and states the objective: 'Achieve 98% test accuracy on the MNIST dataset'. It also notes that multiple `/ask` and `/fix` prompts may be needed and that code should be wrapped in triple backticks. The second cell is titled 'Creating a simple Tensorflow program with /ask' and provides similar instructions for creating a TensorFlow application. The third cell is titled '(BONUS CHALLENGE) Train a model with multiple GPUs' and provides instructions for training a model on multiple GPUs.

Creating a simple Tensorflow program

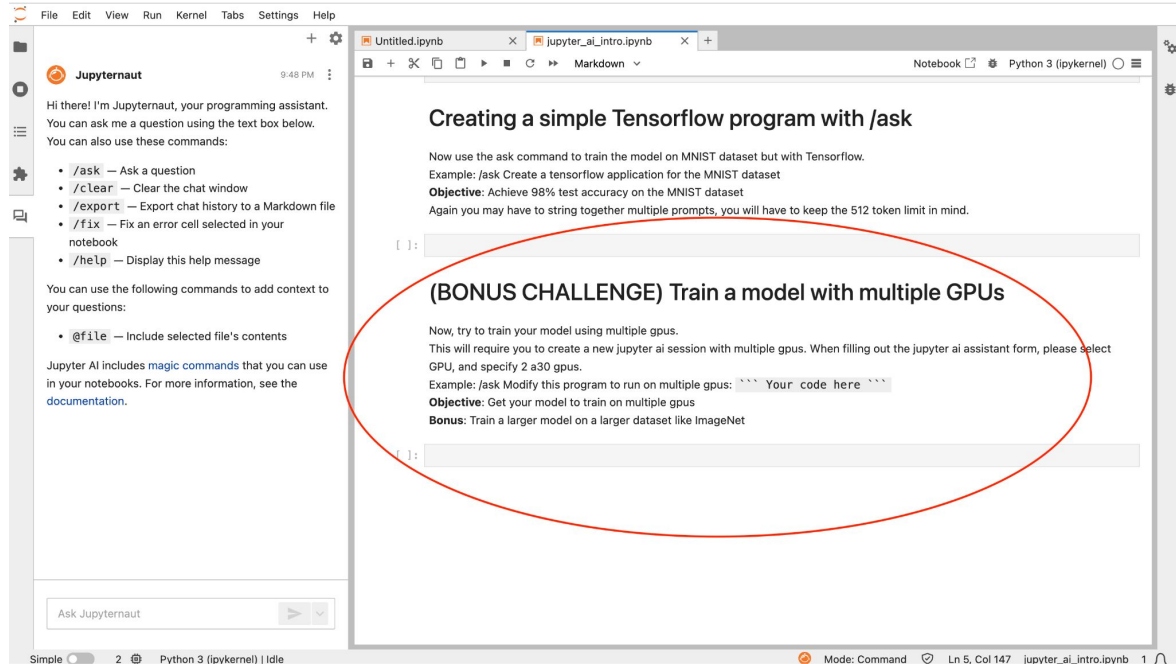
- Use the `/ask` command to help develop code for solving the MNIST dataset with Tensorflow

The screenshot displays the Jupyter AI interface. On the left is a chat window titled 'Jupyter AI' with a list of commands: `/ask` (Ask a question), `/clear` (Clear the chat window), `/export` (Export chat history to a Markdown file), `/fix` (Fix an error cell selected in your notebook), and `/help` (Display this help message). Below this, it states 'You can use the following commands to add context to your questions:' and lists `@file` (Include selected file's contents). At the bottom of the chat is an input field labeled 'Ask Jupyter AI'.

On the right is a code editor window titled 'Untitled.ipynb'. It shows a notebook with a red circle highlighting the text 'Creating a simple Tensorflow program with /ask'. The notebook content includes a prompt: 'You may need to use multiple /ask and /fix prompts in order to piece together a full, working, application since the token limit is 512. Important: when putting text with code cells into your prompt, make sure to surround it with triple back ticks: ```<code segment>```'. Below this, it says 'Now use the ask command to train the model on MNIST dataset but with Tensorflow. Example: /ask Create a tensorflow application for the MNIST dataset Objective: Achieve 98% test accuracy on the MNIST dataset Again you may have to string together multiple prompts, you will have to keep the 512 token limit in mind.' Below the highlighted text is another prompt: '(BONUS CHALLENGE) Train a model with multiple GPUs. Now, try to train your model using multiple gpus. This will require you to create a new jupyter ai session with multiple gpus. When filling out the jupyter ai assistant form, please select GPU, and specify 2 a30 gpus. Example: /ask Modify this program to run on multiple gpus: ``` Your code here ``` Objective: Get your model to train on multiple gpus Bonus: Train a larger model on a larger dataset like ImageNet'.

Bonus Challenge

- Create a new Jupyter AI Assistant session with 2 A30 GPUs
- Use `/ask` and `/fix` to help develop code which uses multiple gpus to train a large model on the ImageNet dataset



The screenshot shows the Jupyter AI Assistant interface. On the left is a sidebar with the Jupyter logo and a list of commands: `/ask` (Ask a question), `/clear` (Clear the chat window), `/export` (Export chat history to a Markdown file), `/fix` (Fix an error cell selected in your notebook), and `/help` (Display this help message). Below this is a text input field labeled "Ask Jupyter AI Assistant".

The main area displays a notebook titled "jupyter_ai_intro.ipynb". The first cell contains the text: "Creating a simple Tensorflow program with `/ask`". It explains how to use the `/ask` command to train a model on the MNIST dataset, with an example: `/ask Create a tensorflow application for the MNIST dataset`. The objective is to achieve 98% test accuracy on the MNIST dataset, and a note mentions keeping the 512 token limit in mind.

The second cell is titled "(BONUS CHALLENGE) Train a model with multiple GPUs". It instructs the user to train a model using multiple GPUs, requiring a new Jupyter AI session with multiple GPUs and specifying 2 A30 GPUs. An example prompt is shown: `/ask Modify this program to run on multiple gpus: '``` Your code here ```'`. The objective is to get the model to train on multiple GPUs, and the bonus is to train a larger model on a larger dataset like ImageNet.

A red oval highlights the bonus challenge section in the second cell.

Technical Challenges

- Running LLM inference server on Intel PVC requires a full exclusive node
 - OneCCL does not play well with slurm, see <https://community.intel.com/t5/oneAPI-Registration-Download/ccl-issue-with-Multi-GPU-AI-Training-Data-Parallel-with-Intel/td-p/1609982>
- vLLM on xpu device would sometimes hang indefinitely when queuing prompts
 - Workaround: design an in-take web server to distribute prompts onto the backend LLM inference servers



Acknowledgements

This work was supported by

- the National Science Foundation (NSF), award numbers:
 - 2112356 - ACES - Accelerating Computing for Emerging Sciences
 - 1925764 - SWEETER - SouthWest Expertise in Expanding, Training, Education and Research
 - 2019129 - FASTER - Fostering Accelerated Scientific Transformations, Education, and Research
 - 2411377 - GOODLUCK - Growing Open OnDemand: Leveraging Unified Community Knowledge
- Staff and students at Texas A&M High-Performance Research Computing.
- ACCESS CCEP pilot program, Tier-II



Need Help?

First check the [FAQ](#)

- [Knowledge Base](#)
- Send us a ticket using the dashboard tab on our [web portal](#)
- Email further questions to help@hprc.tamu.edu

Help us help you -- when you contact us, tell us:

- Which cluster you're using
- Your username
- Job id(s) if any
- Location of your jobfile, input/output files
- Application used, if any
- Module(s) loaded, if any
- Error messages
- Steps you have taken, so we can reproduce the problem





High Performance
Research Computing
DIVISION OF RESEARCH

<https://hprc.tamu.edu>

HPRC Helpdesk:

help@hprc.tamu.edu

Phone: 979-845-0219

Take our short course survey!



HPRC Survey

https://u.tamu.edu/hprc_shortcourse_survey

Thank you! Any Questions?





High Performance Research Computing

DIVISION OF RESEARCH

<https://hprc.tamu.edu>

HPRC Helpdesk:

help@hprc.tamu.edu

Phone: 979-845-0219

Help us help you. Please include details in your request for support, such as, Cluster (ACES, Faster, Grace, Launch), NetID ACCESS ID or Username, Job information (Job ID(s), Location of your jobfile, input/output files, Application, Module(s) loaded, Error messages, etc), and Steps you have taken, so we can reproduce the problem.

