Logging in to Argo-

1. ssh [sdashora@hopper.orc.gmu.edu](mailto:sdashora@hopper.orc.gmu.edu)

Text

Description automatically generated

1. use winscp to transfer python file

Graphical user interface, text, application, email

Description automatically generated

1. module load gnu10

Text

Description automatically generated

4.load python modules and virtual environment and activate

Text

Description automatically generated

1. go to scratch folder in virtial environment upgrade pip and then install all the necessary libraries–

Text

Description automatically generated

6 Now go to ondemand.orc.gmu.edu- >>files>>home directory

Graphical user interface, text, application, email

Description automatically generated

New tab would open up with all the folders at head node

Graphical user interface, text, application

Description automatically generated

1. Create slurm script – to allocate gpu to run modelling

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. Test if gpu has been allotted-

Text

Description automatically generated

(Only for testing if python script file throws any error)



After testing exit virtual environment

Text

Description automatically generated with low confidence

1. Navigate to slurm file folder-

Run using sbatch command if error run dos2unix \* and ahain run slurm file

Next commands to check if the job has been submitted or not

Text

Description automatically generated

1. For version control- list of required packages- to install all the packages in other vitual environment in one go

Text

Description automatically generated with medium confidence

Text

Description automatically generated

1. To capture modules as well-

A picture containing text, indoor, screenshot

Description automatically generated

Another alternative to run slurm from ondemand—if everything else is done then slurm can be run from on demand website as well

Graphical user interface, application

Description automatically generated

Active jobs

Graphical user interface, text, application, email

Description automatically generated

Systems of systems are complex structures which require extensive testing and evaluation before release. Although individual subsystems may have been tested and approved for high levels of dependability, the system's emergent behaviors may allow it to migrate into a harmful operating state without any sub-system or agent failing.  The issue indicates the need for extensive testing of agent combinations that might result in scenarios with highly harmful results. Validation testing necessitates the identification of all emergent behavior, particularly that connected to dangerous settings. The paper addresses the complex behavior of components within systems, whose unexplored interactions can lead to precarious situations. To understand the uncertain conduct of these components, this paper uses an abstracted example of two object interactions on a 10 X 10 space to develop a digital twin of a real-world system. The developed twin was then used to test and evaluate all the possible corner cases of component interaction. The study describes how supervised deep learning can be utilized for addressing emergent behaviors of digital twin. It uses a series of experiments with incremental space complexity to understand hierarchical learning and predicting capabilities of algorithms for corner and missing cases. Initially, by comparing the accuracies of experiments, the paper concludes that learning and capability of algorithms decreases with increasing combinatorial complexity under similar parametric environment. Later paper concludes that combinatorial complexity can be handled using different set of parameters in an algorithm for varying space combinations.

This study supports the vision of developing digital twin and using deep learning algorithm to identify every unusual condition of any complex real-world system to avoid unanticipated scenarios.