



# Research and AI at Scale

## Part 2

Kaitlyn Wade, PhD Student  
Department of Computer Science  
Western University

# Land acknowledgement

---

*We acknowledge that Western University is located on the traditional lands of the Anishinaabek, Haudenosaunee, Lūnaapéewak and Attawandaron peoples, on lands connected with the London Township and Sombra Treaties of 1796 and the Dish with One Spoon Covenant Wampum.*

*With this, we respect the longstanding relationships that Indigenous Nations have to this land, as they are the original caretakers. We acknowledge historical and ongoing injustices that Indigenous Peoples endure in Canada, and we accept responsibility to contribute toward revealing and correcting miseducation as well as renewing respectful relationships with Indigenous communities.*



# Agenda



What is the Digital Research Alliance of Canada?



Introduction to High-Performance Computing



Introduction to AI and Data Science



Best Practices, Tips, and Common Mistakes



Hands-on Activities

# DRI EDIA Champions Program

---

- **What is it?**

- An initiative led by the **Digital Research Alliance of Canada** to empower equity-deserving researchers.

- **Aims**

- Increase access to digital research infrastructure (DRI)
- Foster equity, diversity, inclusion, and accessibility (EDIA) in research

- **Why it matters:**

- Reduces barriers to entry for underrepresented groups
- Supports a diverse research community by ensuring equitable access to tools and resources



# **Digital Research Alliance of Canada**





**Digital Research  
Alliance** of Canada

**Alliance de recherche  
numérique** du Canada



Government-funded  
non-profit organization



Supports all  
researchers in Canada



Provides **free** access  
to tools and resources.

**Goal:** To advance Canadian research excellence by offering equitable access to **digital research infrastructure (DRI)**

# The Alliance Supports Research in Canada



## **Funds the Digital Research Infrastructure**

- Builds & maintains Canada's DRI ecosystem
- Ensures equitable access for all researchers at Canadian institutions



## **Provides training resources**

- Offers workshops, webinars, and online guides to help researchers learn and adopt digital tools

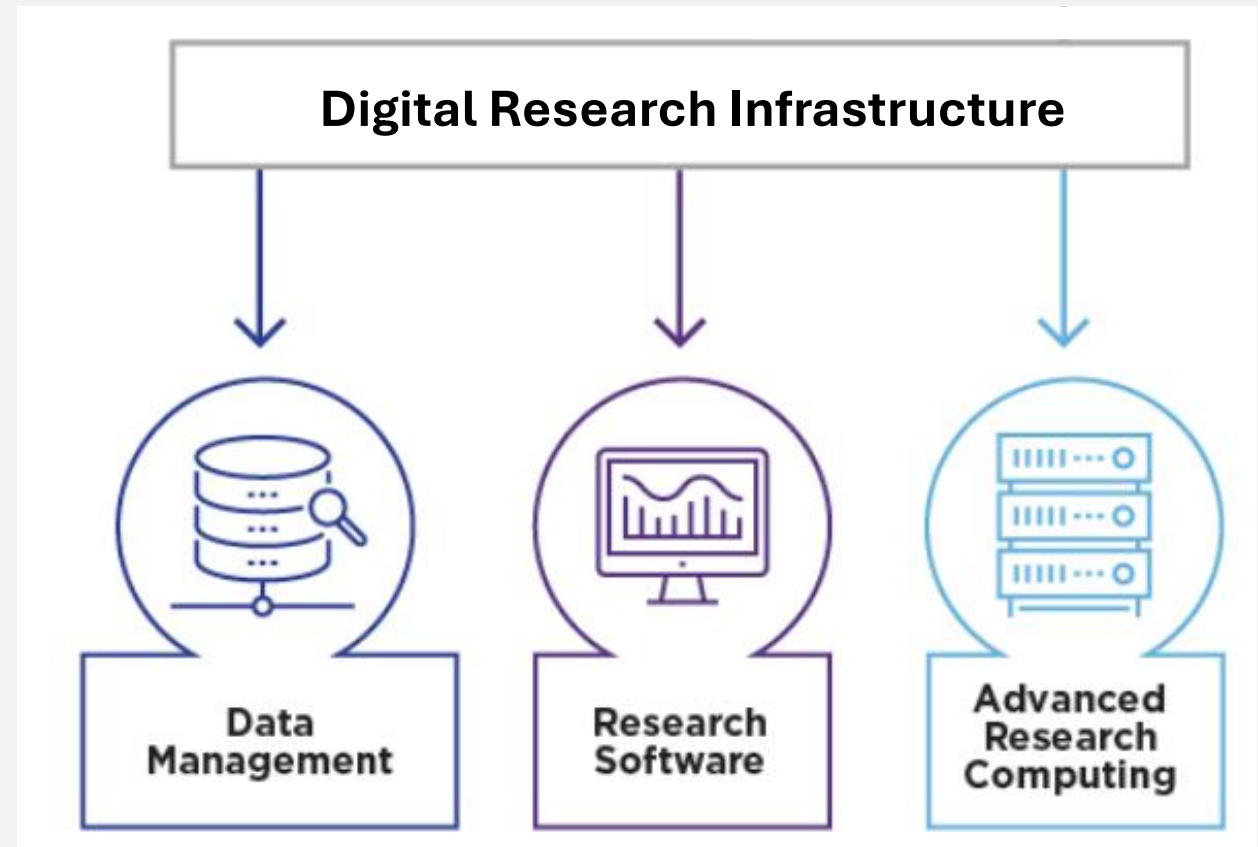


## **Supports research initiatives**

- Funds initiatives such as EDIA Champions
- Organizes Resource Allocation Competitions each year

# Digital Research Infrastructure (DRI)

- Comprehensive system of tools, platforms, and services that **supports research** across Canada.
- **3 components**
  1. Data Management
  2. Research Software
  3. Advanced Research Computing
- **Equitable access** for all researchers in Canada, regardless of institution size or geographic location





# Research Data Management Support

- **Federated Research Data Repository (FRDR)**
  - National platform for sharing and preserving Canadian research data
  - Allows storage, management, and sharing of large datasets
  - Indexed in Google Dataset Search, and more
- **Data Management Plan (DMP) Assistant**
  - **DMP**: outlines how research data will be handled through a project's lifecycle
  - Online, prompt-based tool that provides best-practice guidelines and examples
- Many more!



# DRI: Research Software



## Why Research Software Matters:

- **Enables Discoveries:** Helps researchers manipulate, analyze, and visualize data.
- **Supports Collaboration:** Facilitates teamwork and idea-sharing across borders.
- **Promotes Open Science:** Encourages transparency and accessibility in research.



## What the Alliance Offers:

- **Software Management Plan Template:** A tool to help manage research software from development to documentation.
- **National Cloud Strategy:** Supports access to scalable, secure cloud infrastructure for research projects.

# DRI: Advanced Research Computing



High-performance computing (HPC) and data storage resources



Used in diverse fields including health sciences, engineering, humanities, and aerospace



Accelerates discoveries and innovation



Available to researchers nationwide, regardless of location or discipline



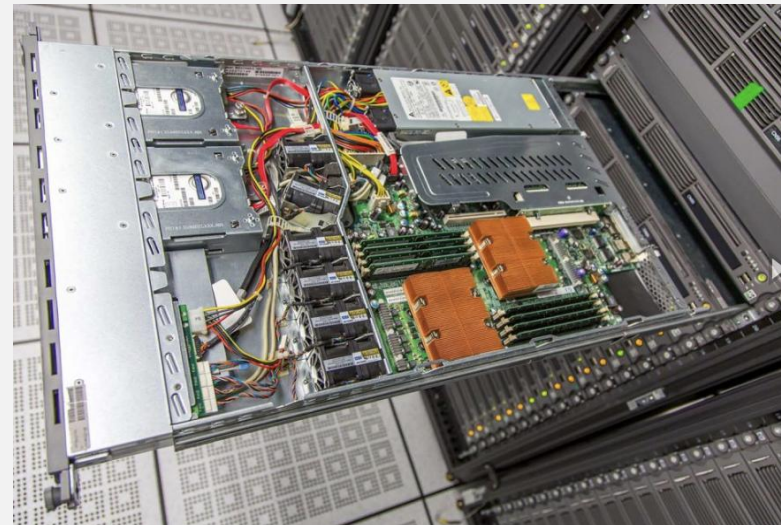
Supported by 38 partner universities and a team of 200+ experts

# ARC: Canada's Top 5 Supercomputers

- **Narval** (Calcul Québec)
  - #190 in the world
- **Niagara** (SciNet / U. Toronto)
  - # 282 in the world
  - **Mist** GPU cluster
- **Cedar** (SFU)
  - # 301 in the world
- **Béluga** (Calcul Québec)
- **Graham** (SHARCNET / U. Waterloo)



Graham Cluster

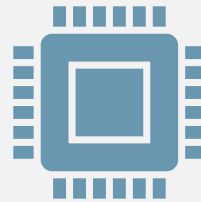


Close-up view of a server

# Upgrades to ARC



- Major upgrades ongoing
- Expected completion in spring 2025
- May be some outages or service reductions



- Faster processing
- More storage
- Improved reliability
- Upgraded GPUs



## New Systems

- Béluga → **Rorqual**
- Cedar → **Fir**
- Niagara & Mist → **Trillium**
- Graham → **Nibi**


# Summary

---

- **Digital Research Alliance of Canada**
  - Government-funded **non-profit** dedicated to supporting research in Canada
  - Coordinates Canada's **Digital Research Infrastructure (DRI)**
  - Makes DRI tools **available to all researchers** in Canada
- DRI Components
  - **Research Data Management (RDM)**: tools for organizing, sharing and preserving research data
  - **Research Software**: platforms enabling collaboration and open science
  - **Advanced Research Computing (ARC)**: high-performance computing and data storage for research
- **Upgrades** to ARC coming in 2025!




**Any Questions?**



Which of the 3 pillars  
of DRI do you see  
yourself using in your  
research?







# **Introduction to High-Performance Computing**



# Terminology

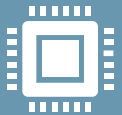
---



**Research computing:** research that depends on a computer



**Advanced research computing (ARC):** research that needs a more powerful device than a personal computer



**Supercomputer:** large, custom, shared system or cluster



**High-performance computing (HPC):** using advanced computing resources to perform complex tasks

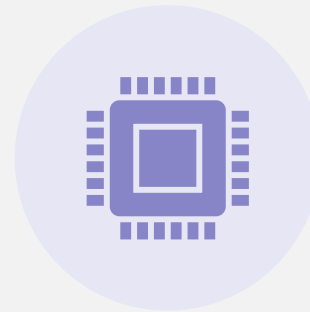
# High-Performance Computing

- 
- Uses powerful computers (clusters of **CPUs, GPUs**), **large amounts of memory**, and **high-speed storage**
  - Allows researchers to solve problems that are too complex for regular computers
  - HPC is needed when:
    - The problem takes **too long** → **faster computation**
    - The problem is **too big** → **more memory**
    - The data is **too big** → **more storage**

# Key Features of HPC



**Parallel Processing:** running multiple computations at once



**Massive storage & memory:** handles large datasets and computational tasks

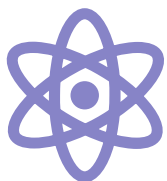


**Scalability:** scale computational power as the size of the task grows



**Performance & speed:** solves problems in hours or days that would take personal computers months

# HPC Use Cases



Engineering



AI & Machine  
Learning



Health  
Sciences



Environmental  
Science



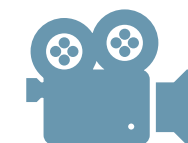
Astronomy



Humanities &  
History



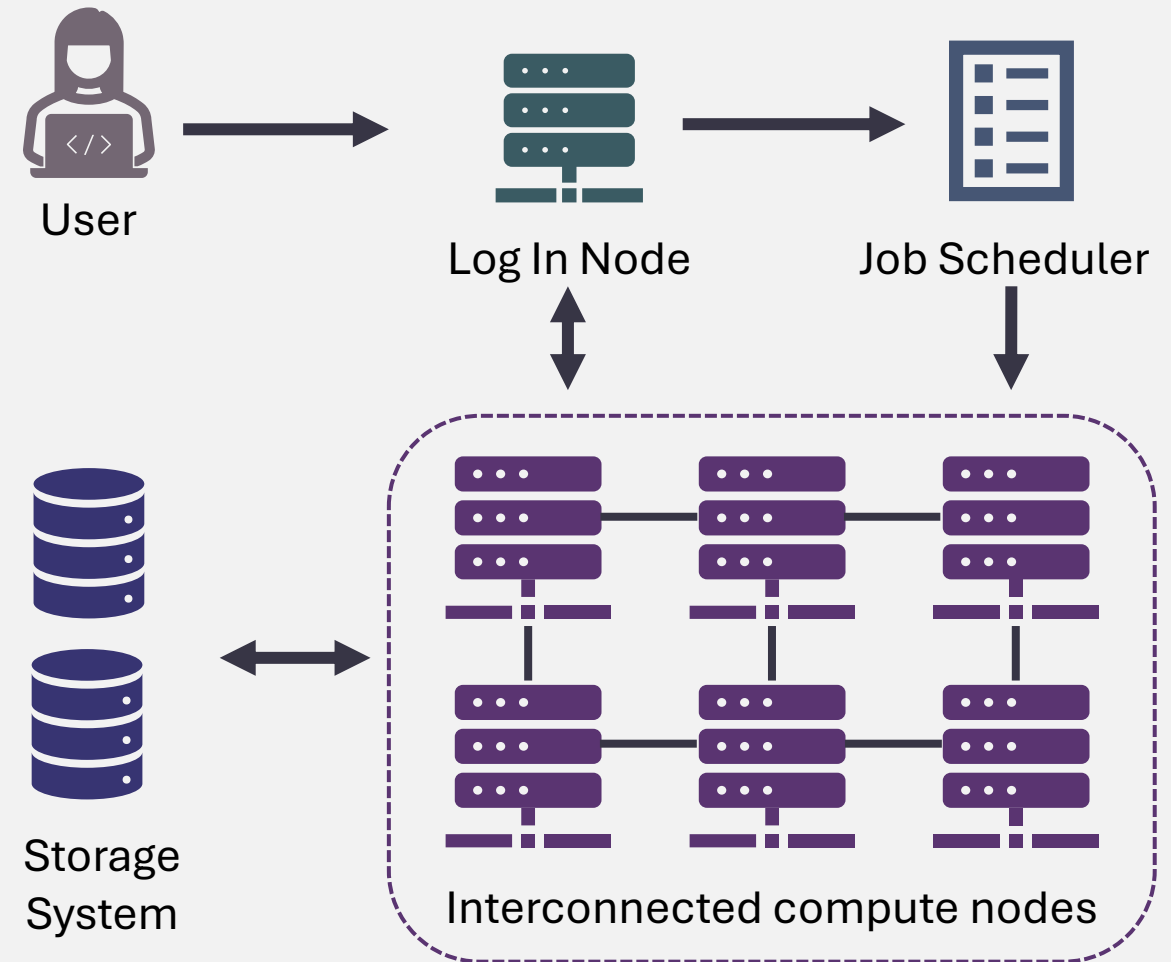
Finance



Media

# Components of an HPC System

- **Nodes:** computers that make up an HPC system
  - **Login Nodes:** where users log in and upload/ download files
  - **Compute Nodes:** execute computational tasks
- **Job scheduler:** manage and allocates computer resources
- **Networking:** fast interconnects for transferring data
  - High-bandwidth, low latency
- **Storage:** fast and scalable storage
  - Parallel filesystems



# Job Scheduling

- 
- Ensures resources (CPUs, GPUs, memory) are used in an **efficient** manner
  - Importance of job scheduling
    - Ensures **fair** resource allocation
    - **Manages** server load
    - **Reduces** wait time
    - Improves overall **throughput**
  - How it works
    - Jobs (tasks) are submitted to a queue with a **priority level**
    - Scheduler **allocates resources based on priorities**
    - Jobs **executed based on resource availability**

# Scheduling Factors in ARC systems

- 
- **Priority allocations**
    - Priority allocations given to certain groups after annual competition
  - **Past usage**
    - If a group recently used a lot of resources, their priority decreases
  - **Time**
    - The longer the job waits, the more priority it accrues
  - **Available resources and job size**
    - Requests for many resources often have longer wait time
    - Requests for moderate resources may have shorter wait time




# Summary

- 
- HPC is used for problems that are too big for personal computers.
  - HPC offers parallel processing power, shared resources, specialized architectures.
  - Unlike personal computers, HPC requires **job scheduling and resource management**.
  - Job scheduling is essential for accessing and managing HPC resources.
    - We will learn more about this later SLURM!
  - **HPC enables researchers to tackle complex computational tasks far beyond the capacity of personal computers.**



**Any Questions?**



Share your  
experiences using  
HPC resources!





# **Getting Started with Advanced Research Computing**



# Getting Access

---

1. Register with the Alliance CCDB: [https://ccdb.alliancecan.ca/account\\_application](https://ccdb.alliancecan.ca/account_application)
  - Pls must get an account first, so they can sponsor your account at no cost.
  - The approval process typically takes 1-2 business days.
2. Setup Multi-factor Authentication  
[https://docs.alliancecan.ca/wiki/Multifactor\\_authentication](https://docs.alliancecan.ca/wiki/Multifactor_authentication)
3. Recommended: Setup SSH keys

For Niagara (and other clusters in the future)

- Go to [https://ccdb.alliancecan.ca/services/opt\\_in](https://ccdb.alliancecan.ca/services/opt_in)
- Click on the “Join” button next to Niagara and Mist.
- After a business day, you will get an email confirming your access to Niagara and Mist

# How to Log In

- 
- ARC systems are remote
  - Connect remotely to the supercomputer using **secure shell (ssh)**
  - Interact with the supercomputer using the command line

1. Open your terminal
2. Connect to the **cedar** supercomputer by typing:  
**ssh USERNAME@cedar.alliancecan.ca**
3. Enter your password (not needed if you use key authentication)
4. Follow in multi-factor authentication prompt (e.g. Duo)

# Example: Logging in to Cedar

```
PS C:\Users\Kaitlyn> ssh kwade4@cedar.alliancecan.ca
(kwade4@cedar.alliancecan.ca) Duo two-factor login for kwade4

Enter a passcode or select one of the following options:

1. Duo Push to phone (iOS)

Passcode or option (1-1): 1
Success. Logging you in...
Success. Logging you in...
Last login: Tue Jan  7 14:39:49 2025 from 23-248-4-188.tpia.execulink.com
-----
=====
Welcome to Cedar! / Bienvenue sur Cedar!

For information see: https://docs.alliancecan.ca/wiki/Cedar
Email support@tech.alliancecan.ca for assistance and/or to report problems.
```

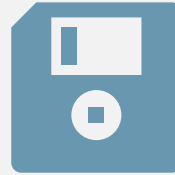
When you login, you will be in your HOME directory

# Storage Spaces on ARC Clusters



## HOME

- Stored on the login node
- Small space
- Only for source code, small parameter files, and job scripts
- Slow to read and write files



## PROJECT

- Large storage space
- Linked to a professor's account
- Data should be static – backed up using tapes



## SCRATCH

- Intensive read/ write operations
- Transient storage – use only for temporary files
- Files are purged if not accessed after 60 days



# How to Transfer Data

---

- To download files when logged in to an ARC cluster, there are many options:
  - `wget URL`
  - `curl URL`
  - `git clone REPO_URL` (to clone a GitHub repository)
- To copy files **from your computer to an ARC cluster** (**cedar** in our example)
  - `scp filename USERNAME@cedar.alliancecan.ca:path/filename`
- To copy files **from an ARC cluster to your computer** (**cedar** in our example)
  - `scp USERNAME@cedar.alliancecan.ca:path/filename filename`

# How to Set Up a Python Virtual Environment

- 
1. Check which versions of Python are available  
**module avail python**
  2. Load version of Python  
**module load python/3.10**
  3. Create a new environment called ENV  
**virtualenv -no-download ENV**
  4. Activate the environment  
**source ENV/bin/activate**
  5. Upgrade pip  
**pip install -no-index -upgrade pip**
  6. Exit the virtual environment  
**deactivate**

# How to Install Packages

- **Recommended way:** using **pip** in your job script

```
pip install --no-index <package name>
```

- **--no-index** means to use “wheels” provided on cluster
- This is faster and doesn’t use up space in your **HOME** folder

- **Last resort:** use **pip** on log in node

- Can install from the centralized web-based **pip** or from GitHub
- In job script: activate the environment you create in **HOME**

```
source $HOME/env/bin/activate # Activate “env” from HOME
```

**DO NOT use  
Anaconda!**  
It is not meant to be  
used on ARC  
resources!

# Pre-installed packages

—  
There are many pre-built Python “**wheels**” are optimized for ARC clusters

- Pytorch
- Biopython
- Sklearn
- Many more!

ARC clusters support other programming languages, including R, C++, MATLAB, Julia, Java, etc.

For convenience, the SciPy stack is available as a module

- NumPy
- SciPy
- Matplotlib
- IPython
- pandas
- SymPy
- nose

```
module load scipy-stack
```

# Other scientific software

---

## Chemistry

- RDKit
- CP2K
- LAMMPS
- GROMACS
- ABINIT
- AMBER
- Atomicrex

## Bioinformatics

- Plink
- REGENIE
- Rosetta
- IQ-Tree
- Bowtie-2
- Samtools
- MUSCLE

## Physics / Engineering

- Quantum ESPRESSO
- Abaqus
- OpenFOAM
- MESA
- STAR-CCM

## Earth Sciences

- CCSM
- CDO
- GRASS GIS
- PROJ
- QGIS

And many more!

[https://docs.alliancecan.ca/wiki/Available\\_software](https://docs.alliancecan.ca/wiki/Available_software)

# How to a Submit Job

- 
- When we first log in, we are on a **login node**
    - Login nodes are **NOT for computations!**
  - To run on a **compute node**, we need to create a **job script** that has requests for specific resources for a specific amount of time.
  - The script is passed to the **scheduler**
    - On ARC clusters, the scheduler is called **SLURM**
    - To submit a job, use the **sbatch** command
  - The scheduler **allocates compute resources** to your job and runs it in due time

# Simple Linux Utility for Resource Management

## 1. Job submission

- User submits job using the **sbatch** SLURM command
- Job script specifies resource requirements (CPU, memory, runtime)

## 2. Job scheduling

- SLURM checks available resources and priority queues
- Schedules jobs based on requested resources, priority, fairness policies

## 3. Job execution

- SLURM allocates requested resources
- Job runs on allocated nodes

## 4. Completion and monitoring

- Users can monitor jobs using **squeue** or view log files.



# Example Job Script: submit\_job.sh (1)

```
#!/bin/bash
#SBATCH --account=my_account      # Account name
#SBATCH --job-name=my_job        # Name of job
#SBATCH --ntasks=1               # Number of tasks
#SBATCH --mem=4G                  # Amount of CPU RAM
#SBATCH --time=00:01:00           # Max runtime (HH:MM:SS)
#SBATCH --gpus=1                  # Number of GPUs

module load python/3.10 cuda cudnn # Load required module
...
```



# Example Job Script: submit\_job.sh (2)

```
module load python/3.10 cuda cudnn          # Load python version

virtualenv --no-download $SLURM_TMPDIR/env    # Create environment "env"
source $SLURM_TMPDIR/env/bin/activate        # Activate "env"
pip install --no-index --upgrade pip         # Upgrade pip

pip install --no-index tensorflow            # Install package called tensorflow

python main.py                              # Run source-code for run main.py
```

**\$SLURM\_TMPDIR** is a temporary directory that SLURM creates for each job

# Helpful SLURM Commands

- 
- Submit job
    - **sbatch** **<job\_script.sh>**
  - Check on status of a job
    - **sq**
  - Cancel a job
    - **scancel** **<job ID>**
  - View output
    - **cat** **<slurm-job ID>.out**
  - Check job efficiency
    - **sacct** **<job\_ID>**

# Email Notifications from SLURM

#SBATCH --mail-user=<user@email.com>

#SBATCH --mail-type=ALL

Slurm Job\_id=52457383 Name=submit\_job.sh Began, Queued time 00:02:36

Notification when job starts

SM

SLURM workload manager<slurm-noreply@cedar.computecanada.ca>

To: ☒ Kaitlyn Wade

Slurm Job\_id=52457383 Name=submit\_job.sh Ended, Run time 00:03:36, COMPLETED, ExitCode 0

Notification when job ends

SM

SLURM workload manager<slurm-noreply@cedar.computecanada.ca>

To: ☒ Kaitlyn Wade

**Slurm** Job\_id=37957542 Name=submit\_job\_chr6.sh Failed, Run time 07:00:06, TIMEOUT, ExitCode 0

Notification if job fails

S

slurm@calculquebec.ca

To: ☒ Kaitlyn Wade

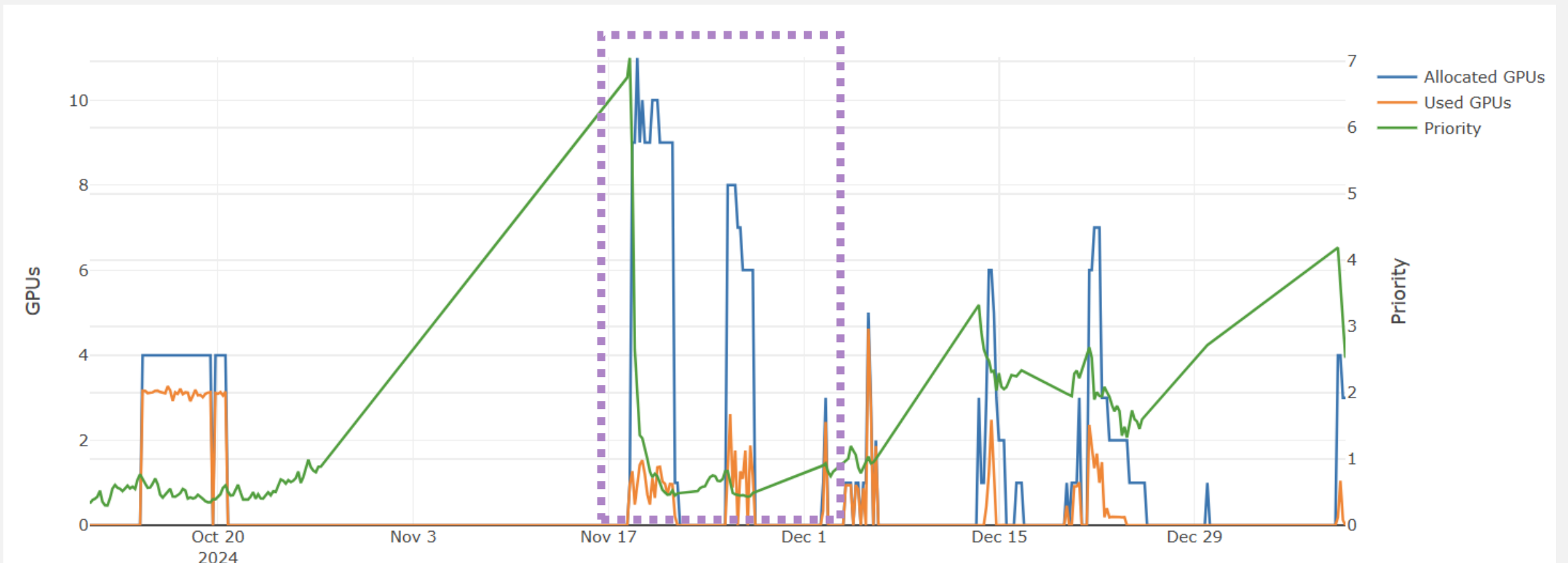
# Monitoring Usage

- Some ARC clusters have graphical user portals
  - <https://portail.narval.calculquebec.ca/>
  - **Account > Job Stats**
- You can monitor:
  - Compute usage
  - Energy consumption
  - Carbon emissions

Type	Allocated	Used
Time	8.0h	6.8h
Nodes	1	
CPU cores	1	0.97
CPU cores by node	ng20202: 1	
Memory	96.0 GB	81.6 GB
GPUs per node	1x NVIDIA A100-SXM4-40GB	Cycles: 88.37%, Memory: 94.79%, Power: 58.82%
Energy		2.90 kWh
Electric car range equivalent		19.20 km
CO2 emissions		1.45 g

# Monitoring Usage

- Allocating a large amount of resources reduces a job's priority and the group's priority





# Live Demo



# **Introduction to Data Science & Artificial Intelligence**



# What is Data Science?



**Exploratory:** understanding data, identifying patterns and trends

- Correlations between genetic mutations and diseases
- Trends in the composition of exoplanet atmospheres



**Confirmatory:** testing hypotheses or validating assumptions

- Associations between rainfall patterns and bird migration
- Economic indicators that predict stock market trends



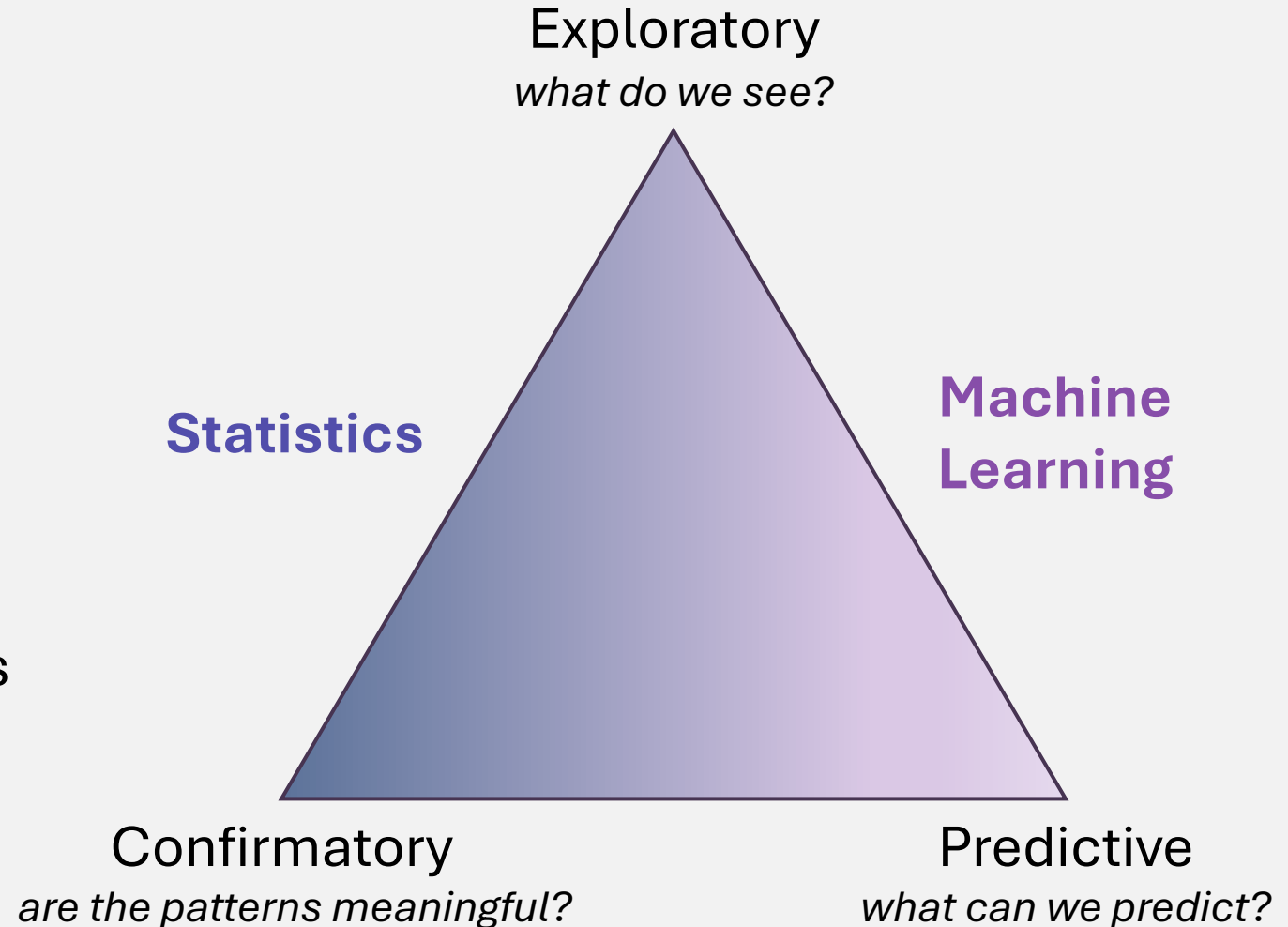
**Predictive:** making predictions about the future based on current data

- Predicting likelihood of disease based on health records
- Predicting future storm patterns



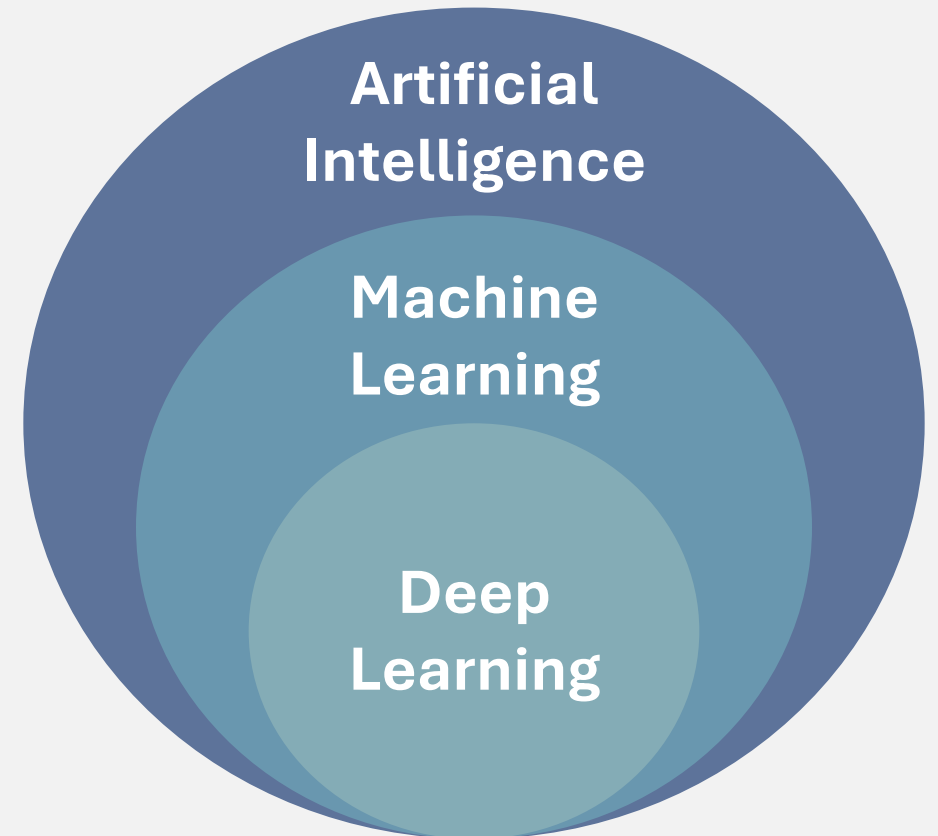
# What is Data Science?

- 
- Data science is not linear
  - Often start with **exploratory analysis**
  - Use insights to inform:
    - **Confirmatory analysis**
    - Models we build for **predictive analysis**
  - Machine learning and statistics are important tools



# Some Terminology

- 
- **Artificial Intelligence (AI):** simulating human intelligence in computers to perform tasks
  - **Machine Learning (ML):** uses algorithms to learn patterns from data and make predictions
  - **Deep Learning (DL):** specialized form of ML for very complex tasks
    - DL models to mimic the structure and reasoning of the human brain



# Types of Data in Machine Learning (1)

- 
- **Labelled Data:** data with input features and corresponding output labels
    - Photos labelled as “cat” or “dog”
    - Patient data with a diagnosis
  - **Unlabelled Data:** data with input features, but no corresponding output labels
    - Raw images
    - Patient data with no diagnosis information

Labelled data



cat



dog



cat



dog

Unlabelled data



# Types of Data in Machine Learning (2)

- **Structured Data:** organized data with a clear schema (rows and columns)
  - Examples: spreadsheets
- **Unstructured Data:** data with no predefined schema
  - May need advanced methods like Natural Language Processing (NLP) or Computer Vision
  - Examples:
    - Audio recordings, videos
    - Free-text documents
    - Emails, messages

Structured Data

Pet Name	Species	Age
Louie	Cat	8
Gus	Cat	3
Akira	Dog	6

Unstructured Data



# Supervised Learning

- 
- **Supervised Learning:** learn from labelled data to make predictions
    - Maps input data (features) to known outputs (labels)

## Classification Problems

- Assign data to categories or classes
- **Examples:** classify species of birds; email spam detection
- **Models:** logistic regression, k-nearest neighbours, support vector machines

## Regression Problems

- Predict continuous numeric values
- **Examples:** forecast temperatures; predict house prices
- **Models:** linear regression, decision trees, random forests, etc.

# Unsupervised Learning

- **Unsupervised Learning:** discover patterns in data without labeled outputs
  - Analyze data to organize it based on similarities or differences

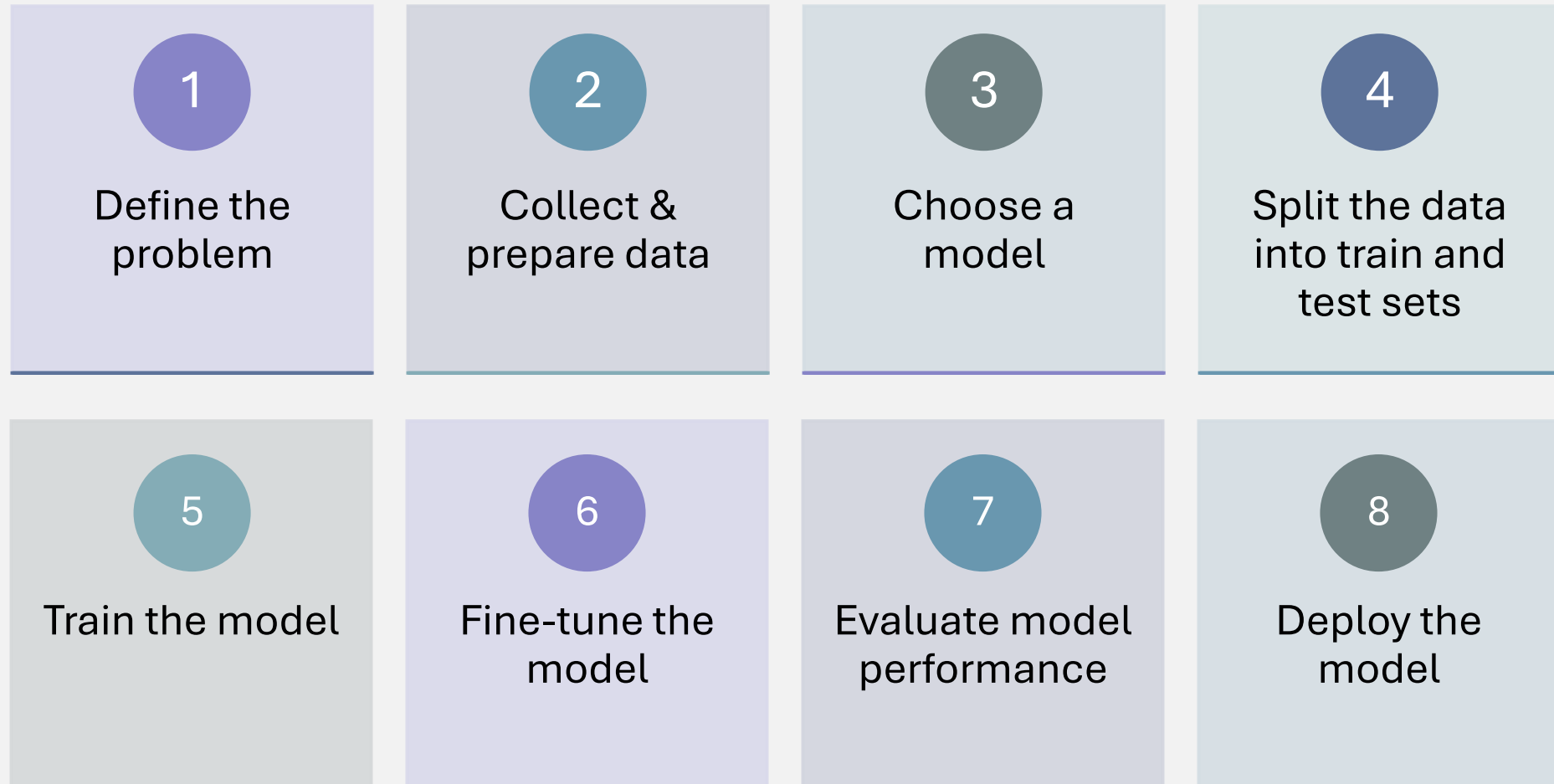
## Clustering Problems

- Grouping similar datapoints together without predefined labels
- **Examples:** Grouping news articles by topic; grouping customers with similar profiles
- **Models:** k-means clustering, hierarchical clustering

## Dimensionality Reduction

- Reduce the # of features in a dataset, while retaining essential info.
- **Examples:** reducing # of variables in gene expression data
- **Models:** principal component analysis (PCA), t-distributed Stochastic Neighbour Embeddings (t-SNE)

# How We Approach Machine Learning Problems



# Prepare Your Data

---

## Data exploration

- Basic statistics
- Grouping / filtering as needed
- Visualizations

## Data cleaning

- Remove or correct invalid entries
- Handle missing data

## Data augmentation

- Increase the size of your data set
- Correct for imbalance



# Choose a Model

## Try simple models first

- Simple models may be faster!
- Depending on your problem, they might be good enough

## Consider your problem and data

- Is your data labelled or unlabelled?
- Is your problem a classification or regression problem?

## Consider the complexity of your problem

- More complex models have higher chance of overfitting
- Simpler models may underfit the data

# Split the Data

---

Splitting data is important for evaluating how well your model generalizes (how it will perform on new unseen data)

## Training set

- 70-80% of the data
- Model learns patterns and relationships from training data

## Testing set

- 20-30% of the data
- Assesses how well model performs on new data
- Reduces risk of memorizing training data (overfitting)

## Validation set

- Can be used to tune the model before testing
- Often 80-10-10 or 70-15-15 train-test-validation split

# Train and Evaluate the Model

---

## Training the model

- Python libraries: **scikit-learn**, **tensorflow**, **PyTorch**
- Train model on training data

## Evaluating the model

- Test model using testing data
- Metrics to consider
  - **Accuracy**: how often is the model correct (for classification)
  - **Mean squared error (MSE)**: how far off predictions are from true values (for regression)
  - Many more!

# Bias-Variance Tradeoff

- 
- **Bias:** errors caused by overly simplistic models
  - **Variance:** errors caused by overly complex models
  - **Underfitting (high bias)**
    - Model is too simple to capture data's complexity
    - Poor performance training and test data
  - **Overfitting (high variance)**
    - Model is too sensitive to noise in the data
    - Good performance on training data, poor performance on test data

**The goal is to capture true patterns in the data without overacting to noise.**

# Neural Networks & Deep Learning

---

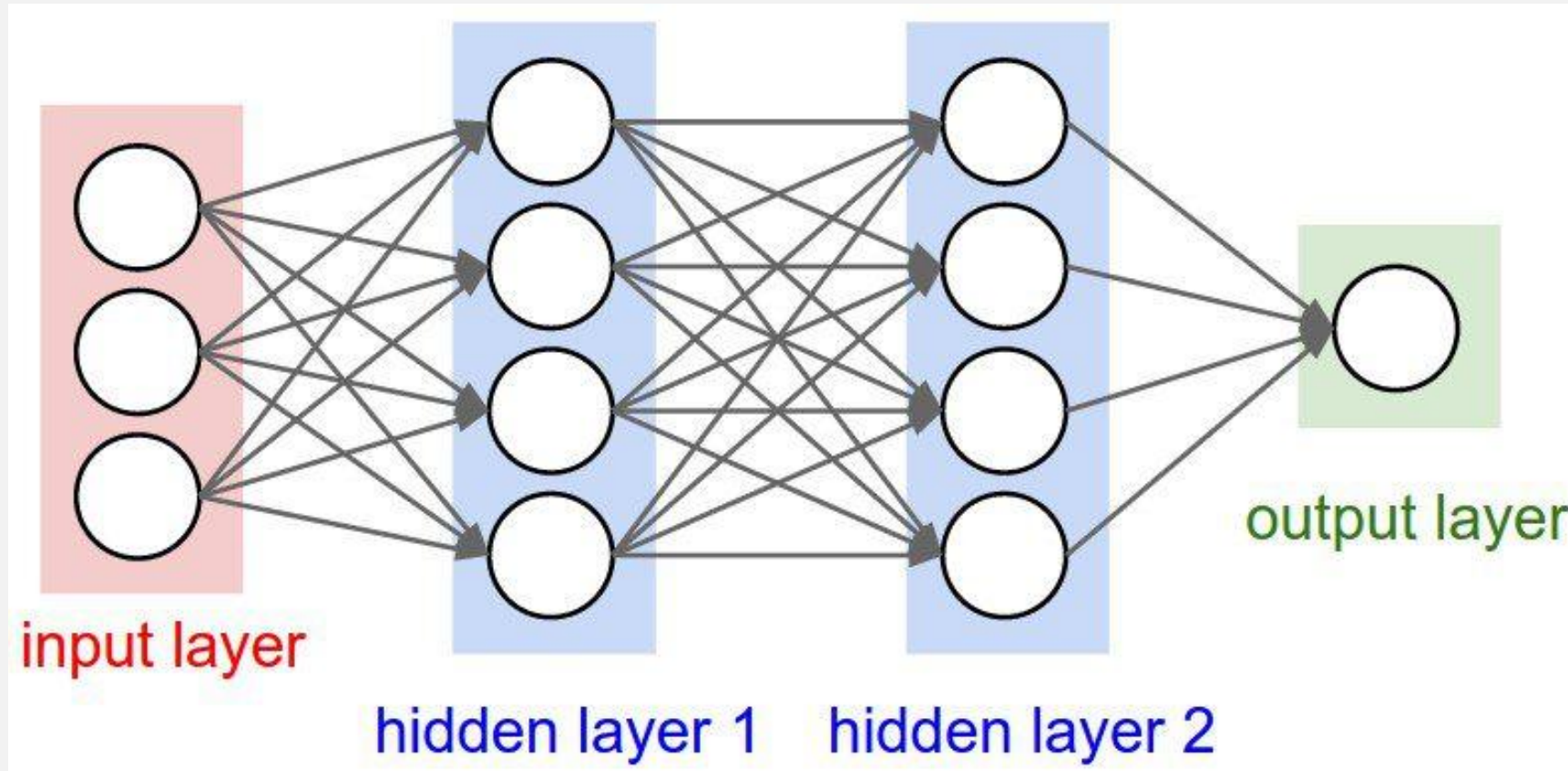
## Neural Network

- ML algorithm **inspired by the brain**
- Layers of interconnected nodes (**neurons**) process data and learn from it
- Each layer transforms input data and passes it to the next for further processing

## Deep Learning

- Subset of ML that uses deep neural networks with **many hidden layers**
- Can learn **complex patterns** from large datasets
- Examples: used in self-driving cars, large-language models (LLMs)

# Example of a Neural Network



<https://pyimagesearch.com/2016/09/26/a-simple-neural-network-with-python-and-keras/>

# How a Neural Network “Learns” (1)

- 
- Neural networks **process input** data and **produce an output** (prediction)
  - They learn by **repeatedly improving guesses**
    - Adjust internal “rules” (weights and biases) based on how wrong their predictions are
  - **Loss function**: how far the prediction is from the actual answer
    - A “score” for how well the network is doing

# How a Neural Network “Learns” (2)

- 
- The network uses the process of **backpropagation** to adjust its “rules”
    - Gradients point the way to reduce loss
  - **Learning rate**: controls how much the network changes at each step
    - Too high: may miss details
    - Too low: may take a long time
  - An **epoch** is one complete pass of showing all the training data to the network.
    - Learns a little better with each epoch
    - Multiple epoch = more learning



# AI and Ethics

---

- **Data Privacy:** models trained on private data can reveal sensitive details
  - **Best practices:** anonymize and encrypt data; differential privacy
- AI models are often called “black boxes”
  - **Interpretability:** understanding how a model processes data to make decisions (transparency)
    - What features influence prediction
  - **Explainability:** extent to which internal workings can be explained in human terms (verification)
    - Why a specific output was produced
- **Fairness and bias:** models can reproduce societal and/or dataset biases

# Foundational Models, LLMs, and Gen AI

- 
- **Foundational Models:** large, versatile AI models, basis for specialized tasks
    - Pre-trained on diverse datasets; can be fine-tuned for specific tasks
    - **Examples:** GPT-4, BERT, CLIP
  - **Large Language Models (LLMs):** subset of foundational models specializing understanding and generating natural language.
    - **Examples:** ChatGPT, Bard, LLaMA, Claude
    - **Applications:** language translation; healthcare; drug discovery
  - **Generative AI (Gen AI):** using AI models to create new content
    - **Healthcare:** can be used to improve image quality of medical images
    - **Assistive technology:** can generate audio descriptions for visual content

# Summary

- 
- Artificial Intelligence > Machine Learning > Deep Learning
  - Machine learning is data-driven
  - Divide data into training/ testing/ validations sets from the start to avoid bias
  - Model selection matters
  - Deep learning is a very powerful tool, but is not transparent
  - LLMs and GenAI are very promising, but use with caution

# Interesting Reading

---

- Gentle Introduction to Machine Learning Models:  
[https://wandb.ai/wandb\\_fc/gentle-intros/reports/A-Gentle-Introduction-to-Machine-Learning-Models--VmlldzoyOTUxNjQw](https://wandb.ai/wandb_fc/gentle-intros/reports/A-Gentle-Introduction-to-Machine-Learning-Models--VmlldzoyOTUxNjQw)
- An Overview of Deep Learning for Curious People  
<https://lilianweng.github.io/posts/2017-06-21-overview/>
- Jason Brownlee's Tutorials:  
<https://machinelearningmastery.com/python-for-machine-learning-7-day-mini-course/>



**Any Questions?**



# Final Thoughts

# Working with “Big Data”

---

- “**Big Data**”: datasets that are too large or complex to be managed, processed, or analyzed using traditional data processing techniques
- **Efficient AI Tools and Techniques**
  - **Accelerate**: simplifies training on multiple GPUs
    - Handles parallelism and synchronization
  - **Mixed Precision Training**: reduce memory usage and increase computational speed using FP16 precision
- **Making the most out of your hardware**
  - **Dask**: parallel computing library that can process large datasets quickly
    - Scalable from laptop to server
    - Uses same syntax as Pandas

# Coding Best Practices



**Automate data processing scripts**



**Use version control (e.g. GitHub)**



**Document everything!**



**Make your code portable**



**Give variables, functions, and files meaningful names**



**Test your code as you write it – not at the end!**



# AI Best Practices



**Explore your dataset first**



**Keep your data and files organized (meaningful file names)**



**Double-check your data processing and data splitting**



**Use appropriate evaluation metrics**



**Watch out for imbalanced datasets**



**Explore models of varying complexity**

# Tips and Tricks for Using ARC

## It's all about balance

- Asking for too many resources can lead to long wait time
- Asking for too few resources can reduce parallelization

## Add checkpoints in your code

- Allows your job to restart where it left off
- Helpful for debugging

## Optimize I/O operations

- Avoid reading and writing lots of small files to the disk (it's slow!)
- Having lots of small files wastes space

## Compress your data (i.e.: .zip, .tar, etc.)

- Fast to copy
- Takes up less space

# Common Mistakes when Using ARC

## Running scripts directly in the command-line



- This will run on the login node (very slow!)
- **Use SLURM instead!**

## Data storage issues



- Storing too much data in **home**
- Not having backups of **scratch**
- **Pay attention to your usage!**

## Requesting the wrong resources



- Requesting an entire node, when it is not needed
- **Monitor efficiency using the SLURM command seff!**

## Package versions



- Always use Python “wheels” that are available
- **These are optimized for the ARC infrastructure!**

## Using Anaconda



- Anaconda is great for PCs, but not large, shared systems
- **Use virtual environments instead!**



# Hands-on Activity

# Hands-On Activities

---

## 1 ) Handwriting Classification Case Study

Go to: [github.com/kwade4/RAISE-DRI/tree/main/workshop\\_examples/mnist](https://github.com/kwade4/RAISE-DRI/tree/main/workshop_examples/mnist)

**Beginner Users** (or if you do not already have a CCDB account)

- Follow the instructions in **mnist\_colab.pdf** to train an AI model on Google Colab.

**Experienced Users** (or if you already have a CCDB account)

- Follow the instructions in **mnist\_cluster.pdf** to train an AI model on one of the Alliance's computing clusters.

## 2 ) Canadian Temperature Regression Case Study

Go to: [github.com/kwade4/RAISE-DRI/tree/main/workshop\\_examples/canada-temperatures](https://github.com/kwade4/RAISE-DRI/tree/main/workshop_examples/canada-temperatures)

- Follow the instructions in **canada-temperature.pdf**
- You will learn how to visualize data and train a regression model to predict average temperature in Canadian cities.



# **Q&A Session**





**Thank you!**