Teaching and Learning ML Ops/Systems on Chameleon

Chameleon Webinar - November 24, 2025

Fraida Fund

NYU TANDON SCHOOL OF ENGINEERING

ML in isolation:

Operational ML systems:

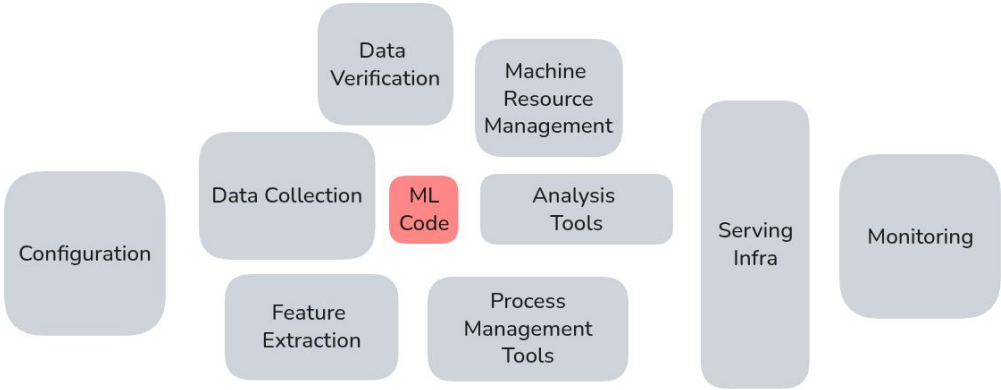


Image source: "Hidden technical debt in machine learning systems." NeurIPS 2015.

How it started: Intro ML

In the previous example, we trained each model for a fixed number of epochs. Now, we'll explore what happens when we vary the training hyperparameters, but train each model to the same validation **accuracy target**. We will consider:

- how much *time* it takes to achieve that accuracy target ("time to accuracy")
- how much *energy* it takes to achieve that accuracy target ("energy to accuracy")
- and the *test accuracy* for the model, given that it is trained to the specified validation accuracy target

Energy consumption

To do this, first we will need some way to measure the energy used to train the model. We will use Zeus, a Python package developed by researchers at the University of Michigan, to measure the GPU energy consump

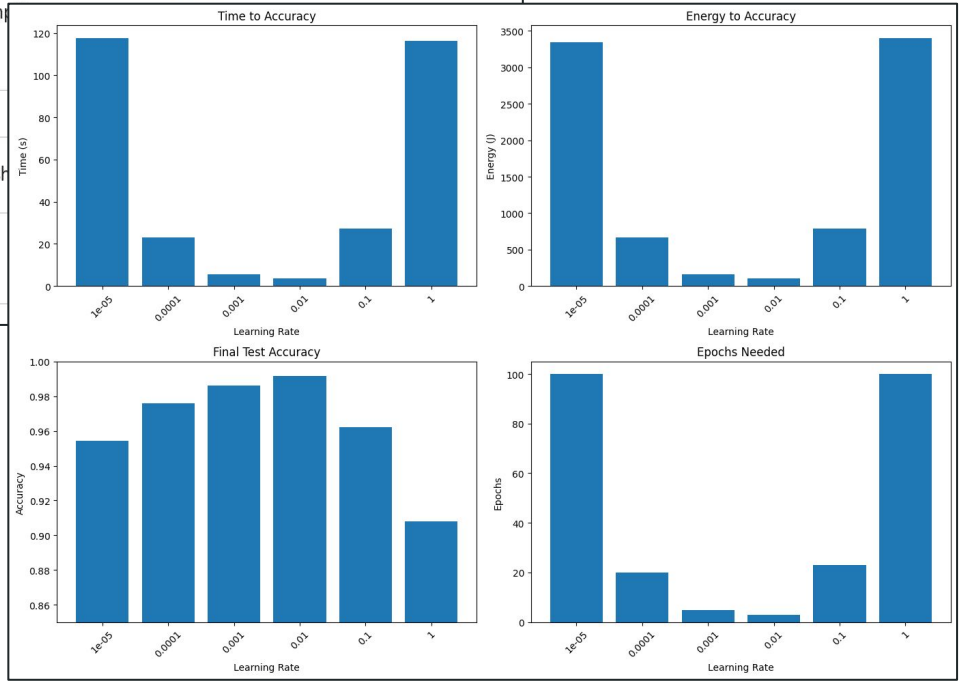
First, install the package:

```
!pip install zeus
```

Then, import it, and start an instance of a monitor, specifying the GPU that it sh

```
from zeus.monitor import ZeusMonitor

monitor = ZeusMonitor(gpu_indices=[0])
```



Training cost + velocity

Husky vs wolf

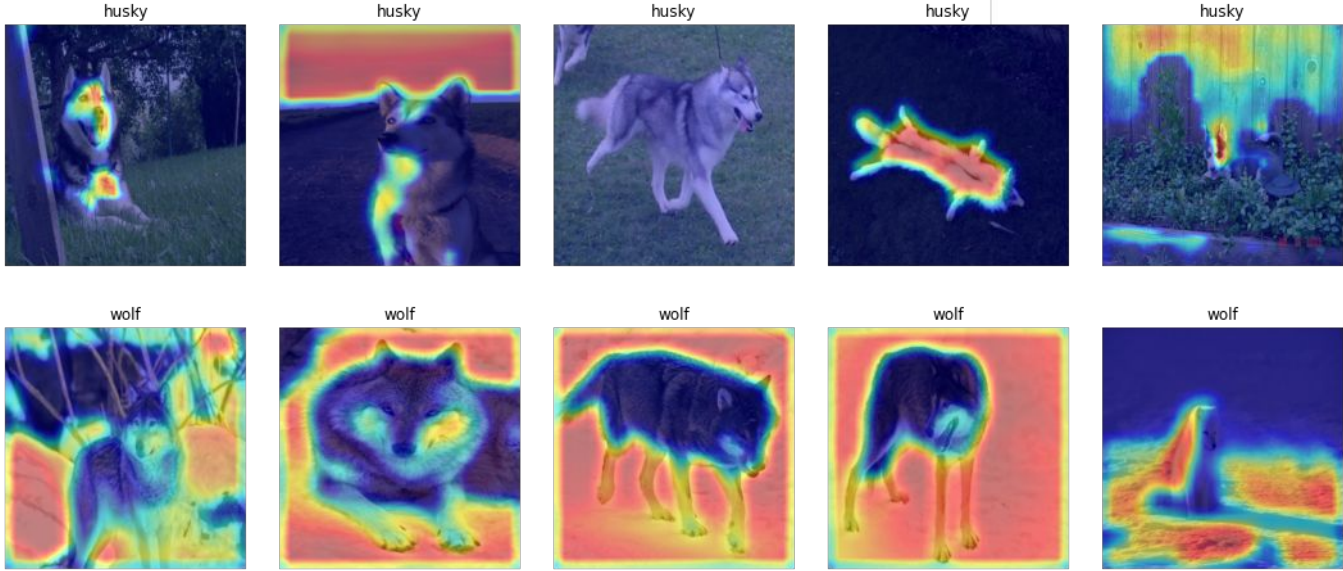
In this question, you will train a convolutional neural network to distinguish between images of a husky and images of a wolf.

You will also use GradCam, a technique that helps explain the prediction of a convolutional neural network, to better understand your model.

Eventually, your model will be deployed to identify huskies and wolves in the wild. However, when it is, you find that it does not do as well as you expected. You will use the GradCam insight as well as some examples of misclassified samples "in the wild" to explain why, and what you might do to fix your model.

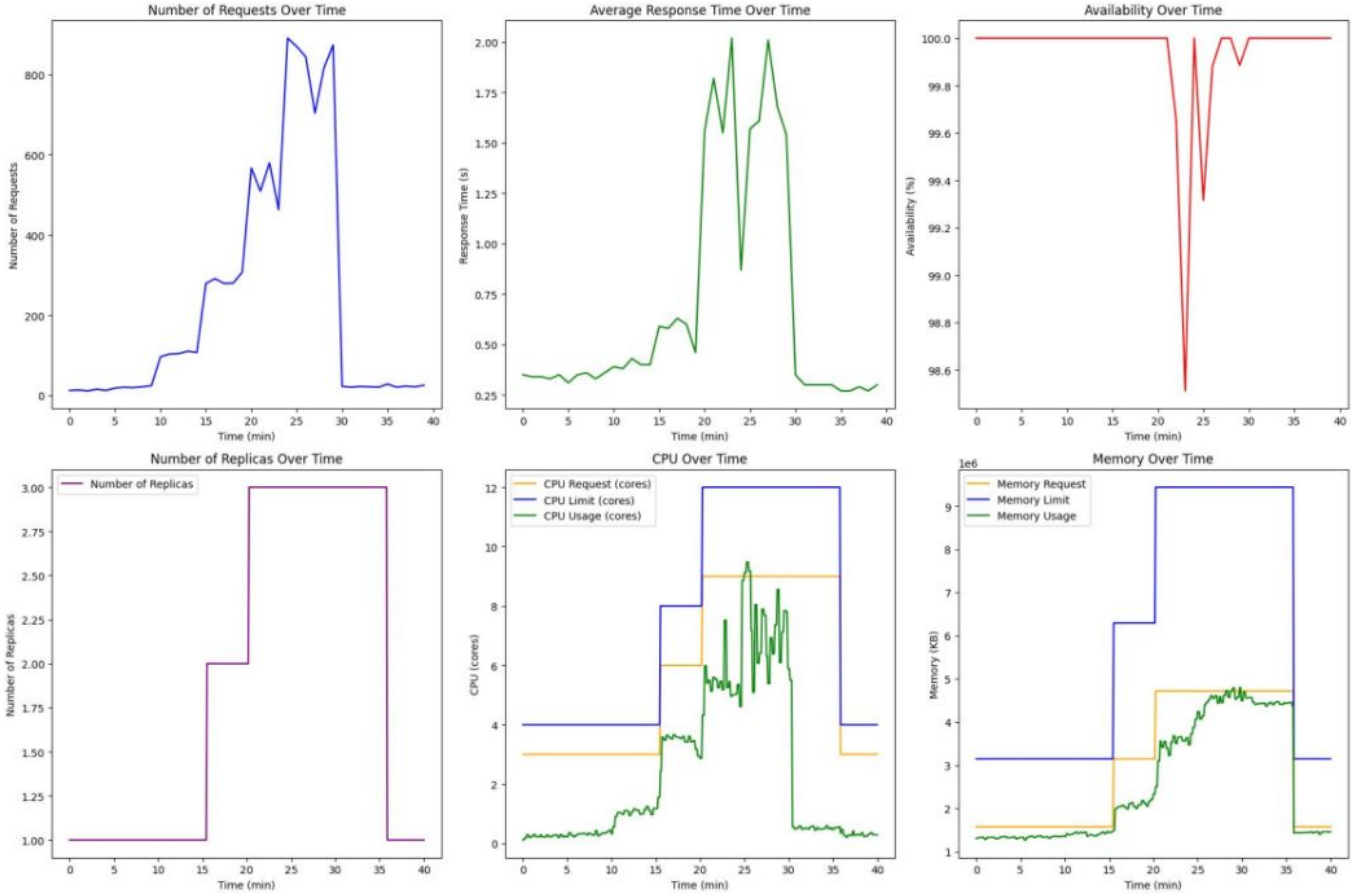
Open workspace

Attribution: This question is adapted from mater
mentorship of Fraida Fund and Mohamed Saeed.
https://github.com/shaivimalik/covid_illegitimat

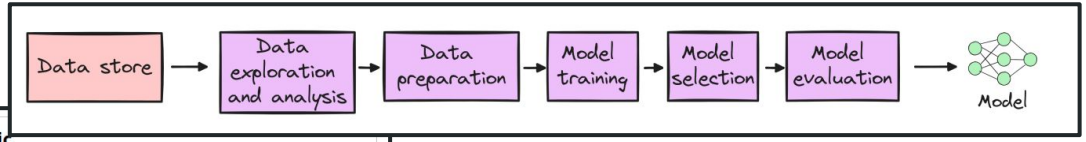


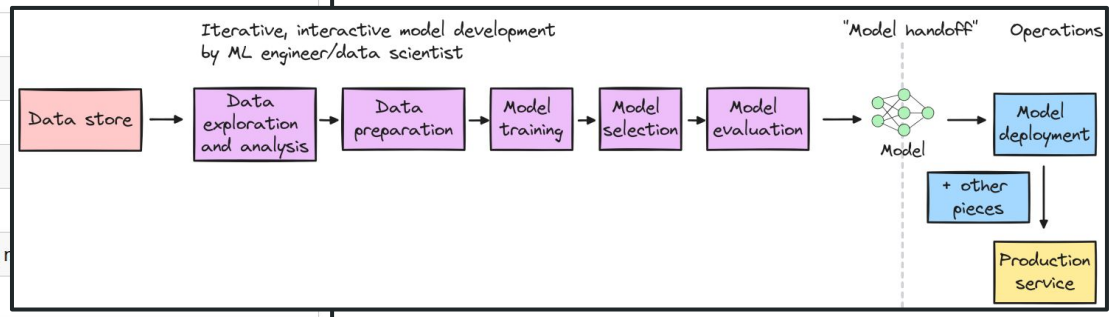
Training-serving skew

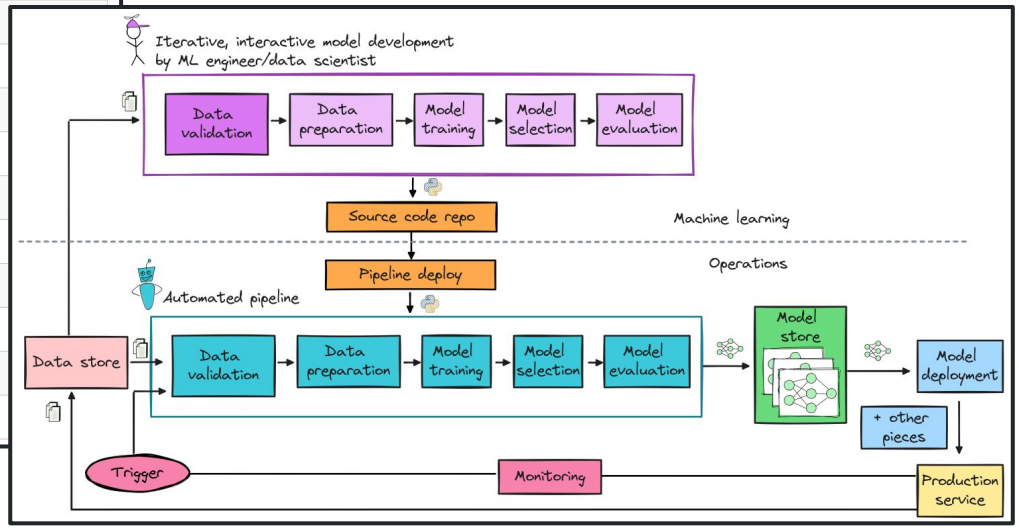
Inference time + K8S deployment



Lecture Date	Topic
	0: Prerequisite review, Python+numpy
9/4	1: Intro to ML, exploratory data analysis
9/11	2: Linear regression
9/18	3: Gradient descent, bias-variance tradeoff
9/25	4: Model selection, regularization
10/2	5: Logistic regression for classification (prerecorded, r
10/9	6: K nearest neighbor, feature selection
10/16	**Midterm exam**
10/23	7: Decision trees, ensembles
10/30	8: Support vector classifiers, kernels, hyperparameter optimization
11/6	9: Neural networks
11/13	10: Deep neural networks, convolutional neural networks
11/20	11: Unsupervised learning
11/27	*Thanksgiving break*
12/4	12: Reinforcement learning
12/11	13: Deploying machine learning systems
12/18	**Final exam**





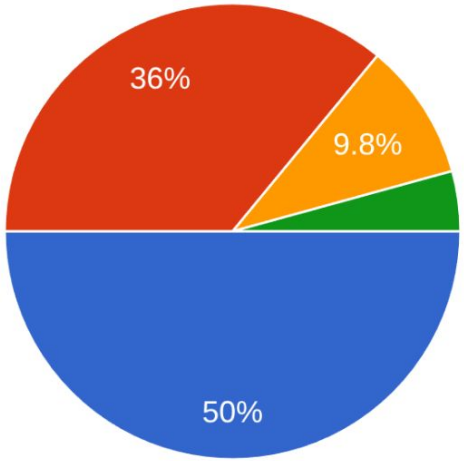


Planning an ML Systems/Ops course

Student survey

Please indicate your interest (note: your choice here is not a commitment to enroll in the course).

164 responses



- I would be very interested/almost certain to enroll in this course
- I would be interested/likely to enroll
- I am slightly interested/might enroll or might not
- I am not interested/would not enroll

→ 191 enrolled Spring 2025

Other courses in our curriculum

Machine learning-centric

Intro ML, Deep Learning, Computer Vision, AI for X

Infrastructure-centric

Cloud Computing, Internet Architecture and Protocols

Some ML Systems/MLOps

High Performance ML, Efficient AI + Hardware Accelerators, Big Data & ML Systems

Examples

Machine Learning Systems Design @ Stanford

Systems for Machine Learning @ UT Austin

Machine Learning Systems @ U of SC

Operationalizing Machine Learning at Chicago

Machine Learning in Production at CMU

Full Stack Deep Learning at UC Berkeley

Research-seminar type

Small scale, discussion, academic paper readings + guest lectures

"Notebook-style" hands-on work

Lab assignments and HW on e.g. Colab or managed services on commercial clouds

Much less on **infra + systems**

Teaching
ML in isolation:

Infrastructure requirements:

Run Python code/notebooks (maybe + GPU)

Works on: Jupyter, Google Colab, traditional HPC

Teaching
Operational ML systems:

Infrastructure requirements: ???

Notebook/batch not a natural interface for learning ML Systems + Operational ML!

"Cloud" is the natural interface - full control over: Compute + storage + network + software systems layers

What we did

By the end of the course, students should **understand the full lifecycle of machine learning systems and have experience designing, building, and maintaining** them. Specifically, students should be able to:

- implement scalable training workflows, including training very large models, multi-GPU training, and cluster management,
- manage experiment tracking and versioning,
- deploy performant inference systems, accounting for compute requirements and latency alongside ML metrics such as accuracy,
- evaluate and monitor deployed systems, including both prediction quality and compute resource usage,
- apply DevOps principles such as CI/CD, infrastructure as code, and cloud-native computing, to ML systems, and
- reason about system reliability, maintainability, and risk.

1. Introduction to ML systems
(prototype vs production; ML *pipeline* as central object vs ML model; business alignment)

2. Cloud Computing
(Building blocks of cloud; IaaS, PaaS, SaaS service models; containers; container orchestration)

3. DevOps for ML systems
(IaC; CI/CD, version control, as applied to ML systems; cloud native computing)

4. Model training at scale
(gradient accumulation; reduced/mixed-precision; PEFT; distributed training with DDP/FSDP)

5. Model training infrastructure, platforms
(experiment tracking, training on a cluster)

6. Model serving
(graph compilation; operator fusion; quantization; concurrent execution; dynamic batching)

7. Monitoring and evaluation
(ML, domain-specific, and operational metrics; evaluating fairness/bias; canary deployments; A/B testing; getting ground truth for production data.)

8. Data systems
(storage in the cloud; ETL pipelines; feature stores)

9. Safeguarding ML systems
(types of harm; mitigation strategies)

10. Commercial clouds

Enrollment: 191 students in first offering.

Audience: Mostly MS Computer Engineering students + some MS Electrical Engineering, Data Science, Computer Science, etc.

Prerequisites: Only Intro ML
(broad survey of machine learning, ends at "transfer learning with ConvNets")

Structure: Weekly in-person lecture
(+ readings, videos, case studies) followed by hands-on lab completed at home (4-5 hours).

Grading: 60% weekly labs, 40% group project

Project: Students worked in groups of 3-4 to design and implement a large-scale ML system.

Human support infrastructure:

- Weekly office hour with the instructor and separate office hours with two TAs.
- Online Q&A forum (by the end of semester, over 700 discussion threads, more than 3,000 unique posts)

The long-running example: GourmetGram

Imagine you have just joined a small startup called GourmetGram, whose entire business is built around sharing photos of food. Your first task is to develop a machine learning model that can look at a picture and automatically classify the food it contains—bread, soup, meat, dessert, vegetables, and so on—so the website can automatically tag each image with the correct category and use those tags to organize, search, and filter photos for users.

Project requirements

Students had to describe a **hypothetical business** they were designing for, and align their system (data, model, infrastructure) with the needs of that business.

Group structure in each 3-4 person group:

- Problem formulation, value proposition, data selection, and system integration were shared responsibilities among all group members.
- Each student was expected to take ownership of a specific subsystem: training, serving, monitoring, data pipeline development, or continuous integration and deployment.
- There were specific expectations for each role.

Opinionated choices

Doing it the "hard way" (minimal managed services)

Transition from self-managed to provider-managed services is easier

Step-by-step instructions for lab assignments

Students focus on doing + observing the system behavior

Intensive hands-on workload

Learn by doing

Project structure and requirements

Infrastructure

Platform tradeoffs

	User has control over compute + storage + network + systems?	Limits $ risk?	Like a "standard" cloud?
Conventional HPC	❌	✅	❌
Commercial clouds aws	✅	❌	✅
Other research testbeds FABRIC	✅	✅	❌
Chameleon Cloud	✅	✅	✅

Chameleon resources used for lab assignments

Compute	Basic VM compute instances (`m1.small`, `m1.medium`, `m1.large`) Single-GPU instances (`compute_liqid`, `compute_gigaio`, `gpu_rtx6000`) Multi-GPU instances (`gpu_a100`, `gpu_v100`, `gpu_p100`, `gpu_mi100`) Edge devices (`raspberrypi5`) (BYOD)
Network	Floating IP addresses Virtual private networks + routers Security groups to permit ports on which services are running
Storage	Block storage volumes Object storage
Interface	Browser-based GUI (OpenStack Horizon GUI) CLI (`openstack` CLI) via Chameleon-hosted Jupyter environment Python API (`python-chi`, OpenStack Python API) via hosted Jupyter Terraform via OpenStack provider

Systems and frameworks used for lab assignments

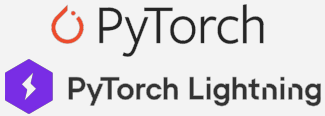
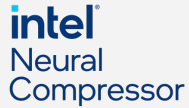
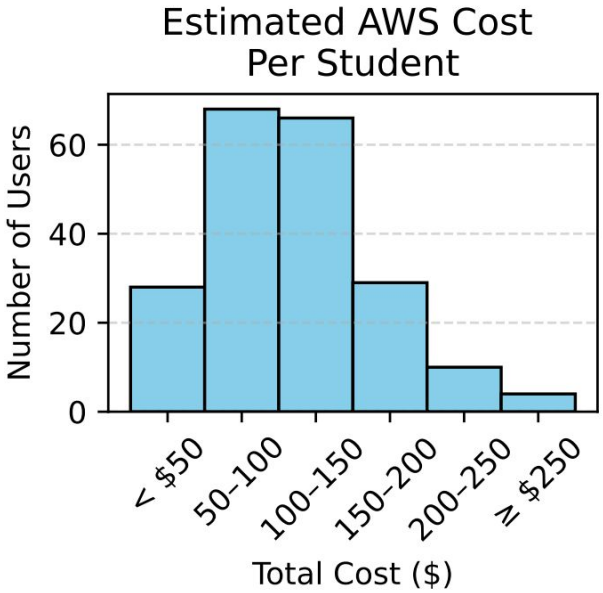
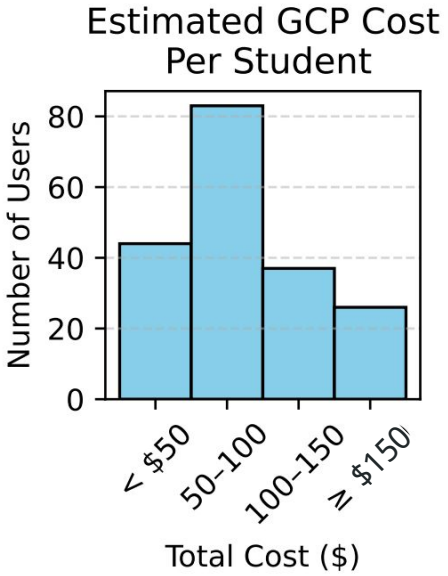
Cloud				
DevOps				
Model training				
Model serving				
Evaluation and monitoring				

Table 1: Usage and estimated cost overall (and per student) by lab assignment and Chameleon node type or VM flavor.

Assignment	Instance Type	Instance Hours	Floating IP Hours	AWS Cost	GCP Cost
1. Hello, Chameleon	`m1.small`	2,620	2,620	$40 ($0.21)	$57 ($0.3)
2. Cloud Computing	`m1.medium` (x3)	52,332	17,444	$2,264 ($12)	$5,347 ($28)
3. MLOps	`m1.medium` (x3)	32,344	10,781	$1,399 ($7.3)	$3,305 ($17)
4. Train at Scale (Multi GPU)	`gpu_a100_pcie`	167	167	$2,993 ($16)	$2,456 ($13)
	`gpu_v100`	210	210	$3,764 ($20)	$3,088 ($16)
4. Train at Scale (One GPU)	`compute_gigaio`	218	218	$722 ($3.8)	$1,106 ($5.8)
5. Training in a Cluster (Multi GPU)	`compute_liqid_2`	330	330	$1,524 ($8)	$662 ($3.5)
	`gpu_mi100`	1,002	1,002	$4,627 ($24)	$2,009 ($11)
5. Experiment Tracking (One GPU)	`compute_gigaio`	28	28	$41 ($0.21)	$32 ($0.17)
	`compute_liqid`	130	130	$190 ($0.99)	$150 ($0.78)
6. Model Serving Optimizations	`compute_gigaio`	215	215	$191 ($1)	$154 ($0.81)
	`compute_liqid`	460	460	$410 ($2.1)	$329 ($1.7)
6. Serving from the Edge	`raspberrypi5`	492	492	NA	NA
6. System Serving Optimizations	`gpu_p100`	707	707	$3,582 ($19)	$1,417 ($7.4)
7. Monitoring and Evaluation	`m1.medium`	9,889	9,889	$461 ($2.4)	$381 ($2)
8. Persistent Data	`m1.large`	8,693	8,693	$1,490 ($7.8)	$626 ($3.3)
Total		**109,837**	**53,387**	**$23,698 ($124)**	**$21,119 ($111)**

A substantial minority of students used more

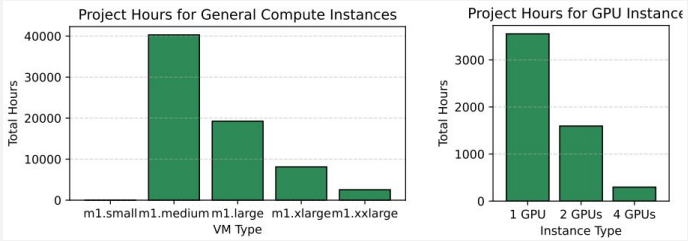




"Most expensive" student's cost 😮: $665 on AWS, $590 on GCP

Chameleon resources used for projects

Compute	70,259 hours general compute 5,446 GPU instance 975 hours other bare metal 175 hours edge *76,855 hours total*
Storage	9 TB block storage volumes 1.5 TB object storage
Estimated cost on commercial cloud	$25,889 ($136/student) on AWS $26,218 ($137/student) on GCP



Note: cost estimates are not as precise for projects as for lab assignments, because we do not know what "lowest equivalent-cost commercial cloud instance" is in each case.

How it went: Spring 2025 +

What went well

Students were able to complete labs

Some on-the-fly fixes due to scale, changes in underlying systems

Students had access to resources with zero $ risk

Obviously, the last days before the deadline were wild...

Students got a nice "portfolio" project out of the deal

Many students went on to engage in research + projects on the topic

Students feel more comfortable/confident and aware of research in this area

What could have gone better

Timelines were difficult to get right

First time offering + moving target = things break at the last minute

Some students did not engage meaningfully with labs

Focus on "doing the steps to get the grade" and not what it does/how it works

Some students did not engage meaningfully with project

Group problems, deadline problems

What we'll do differently

Labs - more scaffolding

Clean up some confusing points, add video, more supporting material

Labs - more on data systems

This was minimal the first time around, but would have made projects better

Project - more structure and intermediate deadlines

Add deadlines + demos for: individual contribution, integration, operation

More on commercial clouds

Extra topics: LLMOps, RAG, Agents

How you can use it

Instructor checklist

- ❑ Plan your sequence
- ❑ Trial your sequence
- ❑ Reserve resources in advance
- ❑ Talk to me (optional but recommended!)

Open educational resources

Course website has all materials**! You can use them!

(Check out the "Instructor Guide" in the menu on left)

Contact: ffund@nyu.edu

** Currently being updated in preparation for the next course offering this Spring.

Other education efforts @ Chameleon

Teaching Cloud Computing with Chameleon: Making Complex Concepts Accessible

How Chameleon Cloud Transforms Computer Science Education Across Europe

May 27, 2025 by Massimo Canonico

User Experiments, Education

Note

Chameleon Cloud serves a vibrant community of educators across US universities who are transforming how computer science is taught through hands-on cloud computing experiences. The international collaborations highlighted in this story represent the kind of academic knowledge sharing that strengthens educational practices globally. Through our user community meetings focused on educational applications—including our 2023 User Meeting's education mini-symposium with its theme of "teaching with testbeds"—we've seen how best practices developed on Chameleon inspire educators worldwide. Researchers like Massimo Canonico, who has shared educational slide decks and teaching resources with the Chameleon community, exemplify how knowledge flows bidirectionally: international partnerships enhance educational methodologies that benefit classrooms both abroad and here in the United States.

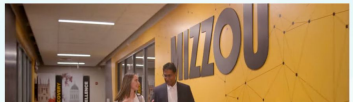
Teaching from Edge to Cloud at the University of Missouri

This month, we're featuring an interview with Professor Prasad Calyam, a distinguished educator and researcher at the University of Missouri-Columbia.

Aug. 21, 2023 by Alicia Esquivel

User Experiments, Education

Profile



Educating with Chameleon at Vanderbilt

July 17, 2023 by Aniruddha Gokhale

User Experiments, Education

This month we present an interview with Aniruddha Gokhale, Full Professor of Computer Science and Engineering at Vanderbilt University who is using Chameleon to teach classes on cloud computing, computer networks, and distributed systems. Professor Gokhale discusses the challenges of teaching with testbeds, explains how he managed to overcome it in his own teaching, and shares recommendations on how to best leverage the power of testbeds in the classroom.

What areas of knowledge are you teaching using Chameleon?

have been teaching three different but related systems courses, one course a semester. These courses are CS4287/5287: Principles of Cloud Computing, CS/ECE 4383/5383: Computer Networks and CS 6381: Distributed Systems Principles. Of these, the course on distributed systems is primarily a graduate-level course though undergrads in senior standing can take instructor's permission to register for the course. This summer I intend to get my class material on Chameleon Trovi.

I teach these courses since they are directly aligned with my research interests. All of these courses could be taught as just a theory course. However, without a practical, hands-on and immersive experience, there is simply no way for a student to appreciate and understand the fundamental challenges, the solutions to these challenges and the impact of these solutions on the performance, reliability, scalability and security of the applications that rely on these systems. Consequently, my approach to teaching these courses has always relied on a mix of theory and empirical approach, where the theory behind one or more specific concepts is taught followed by a programming assignment that reifies those concepts.

Chameleon for Education: IIT's Intro to Parallel Programming

Aug. 23, 2021 by Melanie Cornelius

Tips and Tricks, Education

Interested in using Chameleon for education? Illinois Institute of Technology's TA and PhD candidate Melanie Cornelius and Dr. Zhiling Lan use Chameleon for undergraduate and graduate students in their Intro to Parallel Programming and Parallel and Distributed Processing classes. Learn all about how the course is structured, incorporating Chameleon into assignments, and tips for using Chameleon for education.

Other resources the Chameleon team has created for those interested in using Chameleon for educational purposes or just beginning to explore Chameleon can be found here:

- A blog post to help you or your students get started using Chameleon: how to launch artifacts on Trovi (perfect for classes!), where to find pre-packaged experiments by the Chameleon team on basic topics - orchestration, machine learning, setting up Chameleon resources, networking - and ~5 min YouTube videos for select packaged experiments.
- An older blog post with similar information as the previous link, enumerating each of the pre-packaged experiments and getting started tutorials available on Trovi.
- An Introduction to Chameleon YouTube video: For students just learning how to use Chameleon, this video walks through making a reservation, setting up keys and logging in.

Teaching computer networks on Chameleon

EDIT

8 5 2 1 2024-03-21T13:17UTC education

About

This repository collects network topologies + configurations for teaching computer networks on Chameleon.

A typical sequence for computer networks using these materials, following Kurose & Ross 8th edition, might be:

- Hello, Chameleon
- Hello, Linux (optional)
- TCP/IP protocol stack (aligned with Chapter 1 in Kurose & Ross)
- Socket programming in Python (Chapter 2-3 in K&R)
- TCP congestion control (Chapter 3 in K&R)
- Static routing (Chapter 4 in K&R)
- Designing subnets (Chapter 4 in K&R)
- ARP (Chapter 6 in K&R)
- Secure networked applications (Chapter 8 in K&R)
- Network layer security (Chapter 8 in K&R)

You can also find these materials on Github at: https://github.com/teaching-on-testbeds/chameleon-education/

This material is based upon work supported by the National Science Foundation under Grant No. 2231984.

Mini-symposium on education using Chameleon

This mini-symposium will bring together participants who teach or are interested in teaching using Chameleon and associated testbeds. It will contain presentations of digital teaching materials using Chameleon and a discussion of teaching techniques.

Session I — Open Educational Resources

- Gokhale, Vanderbilt University (Github, Github, Github, assignments)
- Alicia Esquivel Morel, University of Missouri - Columbia (website)
- Chandra Shekhar Pandey, NYU Tandon School of Engineering (Github)
- Tyler Estro, Stony Brook University (Trovi artifact)
- Manvitha Kuncham, Northern Illinois University (slides, Github)
- Massimo Canonico, University of Piemonte Orientale, Italy (website, video)

Session II — Experiences (pre-recorded)

- John Rieffel, Union College (video)
- Massimo Canonico, University of Piemonte Orientale, Italy (video)
- Violet Syrotiuk, Arizona State University (video)
- David Koop, Northern Illinois University (video)
- Mike Papka, Argonne National Laboratory (video)

I would love to hear from you!

Contact: ffund@nyu.edu