Final Project Ideas and Overview

10/26/2023

Final Project Goals

- Design and build something within systems that you are interested in!
  - What are existing solutions to a problem?
  - Why does that problem matter?
  - What is your idea to explore and why might it be a good idea?
  - What tools/datasets can you leverage to test your hypotheses?
  - How will you code up and evaluate your idea?

- Non-goals
  - Building a massive system from scratch
  - Deep application programming – there should be a systems component
  - Large-scale evaluation, e.g., training GPT-4

Logistics and Components

- Projects must be done in groups of 2 or 3

- Projects include two components to turn in:
    - Source code (must include hands-on implementation element)
    - 2-page write-up

- Everything due on Dean’s Date at 5pm ET
    - Grading will factor in completeness *and* originality of project

Ideas

- Anything related to systems is fair game
  - Often helpful to think about applications/workloads you like, and then determine how systems play a role there

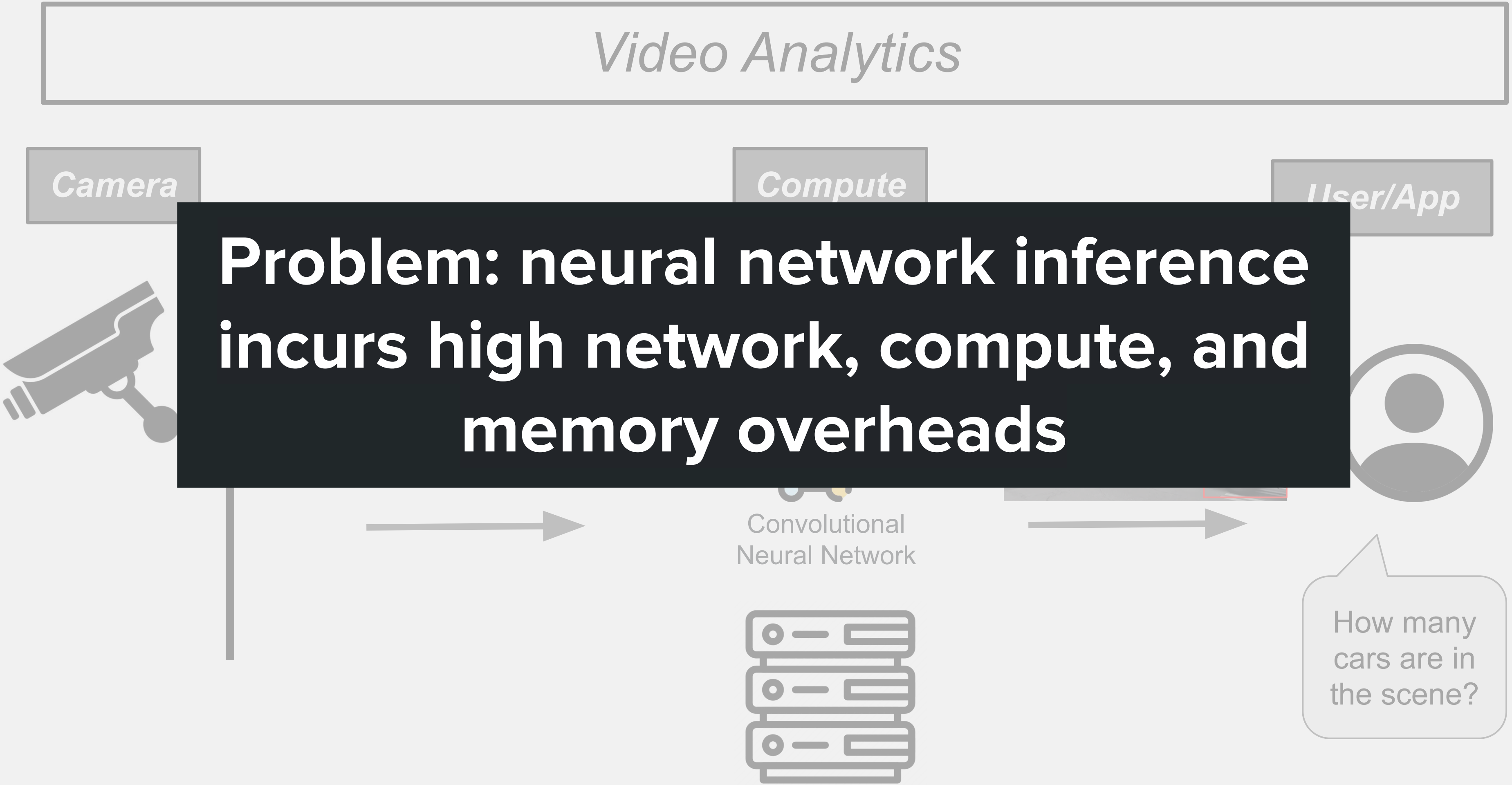
- Projects can extend existing programming assignments, though we encourage you to find something else that excites you

- Feasible projects that lack a bit of originality
  - Designing and implementing a new cache eviction algorithms for certain workloads and comparing with existing algorithms
  - New congestion control algorithms (i.e., when to send the next packet) for different networks or applications
  - Both projects can use existing frameworks, letting you focus on just the novel component (rather than infrastructure)

How to boost the efficacy and resource-efficiency of machine learning systems?

Mike

Problem: neural network inference incurs high network, compute, and memory overheads



Video Analytics

Camera

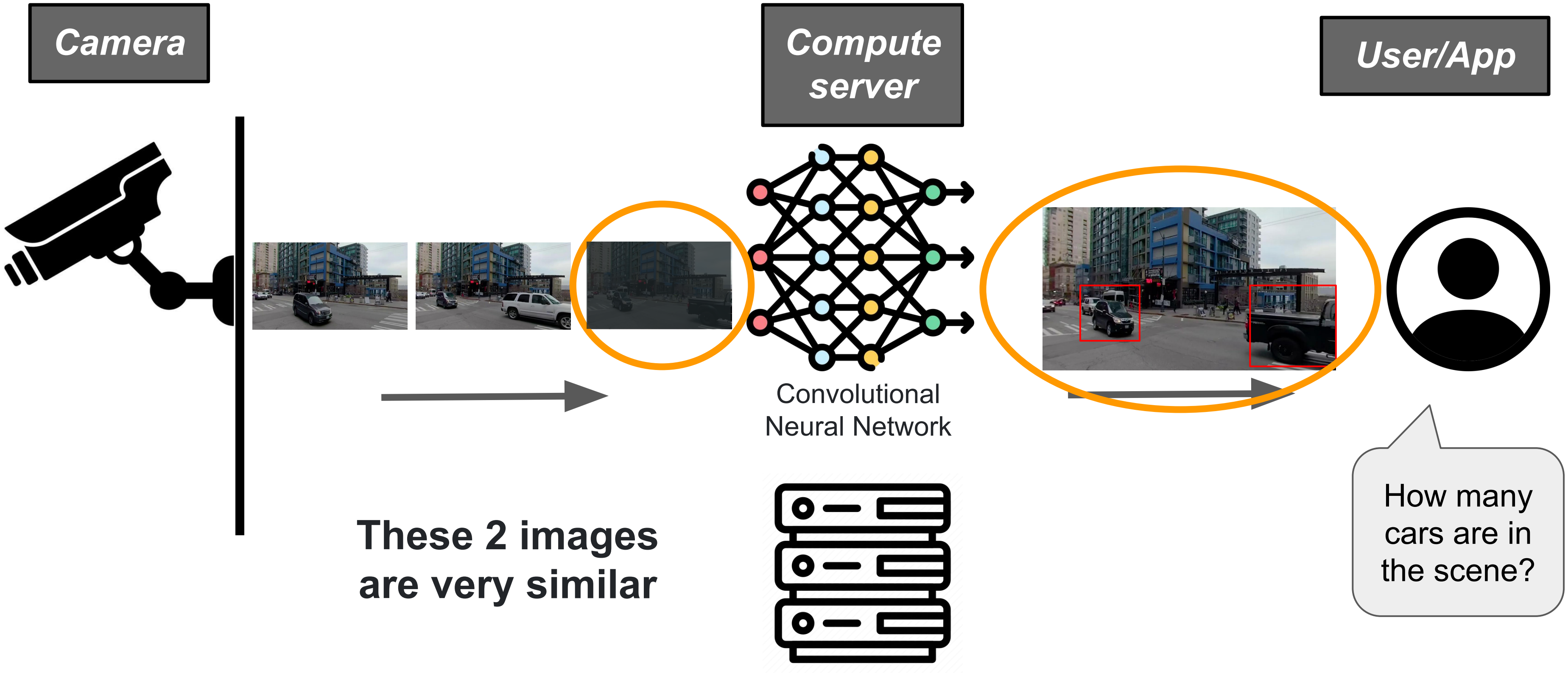
Compute

User/App

Convolutional Neural Network

How many cars are in the scene?

Frame Filtering Idea 1



Camera

Compute server

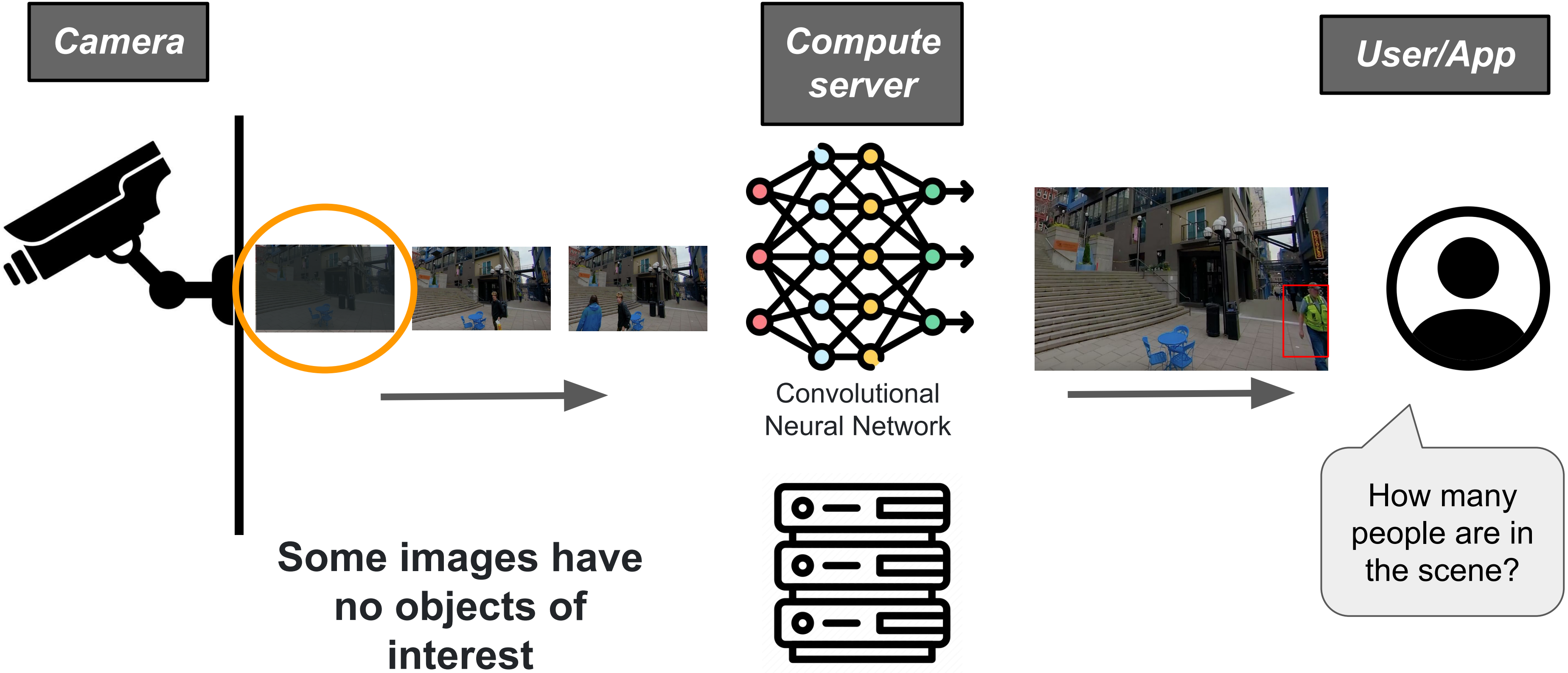
User/App

Convolutional Neural Network

These 2 images are very similar

How many cars are in the scene?

Frame Filtering Idea 2



Camera

Compute server

User/App

Convolutional Neural Network

Some images have no objects of interest

How many people are in the scene?

Video Analytics Project Ideas

Ideas

- Frame filtering to reduce redundant inference
- Identifying images without any interesting objects

Tooling

- Machine learning – PyTorch, TensorFlow, HuggingFace
- Image processing – OpenCV, Pillow
- Don’t need a physical camera, can simulate this setup on your laptop

Datasets

- VIRAT security surveillance videos https://viratdata.org
- YouTube videos

Multi-dispatch Strict Serializability

- Chris

Databases

- Key-value stores
    - Read and write single key-value pairs
    - Fine for some workloads but there are shortcomings
- Transactions are an extension of key-value semantics
    - Fate-sharing
    - Simultaneous externalization

Consistency Models

- Govern the behavior of databases as observed by clients
  - “If a transaction containing writes has committed, what values can be read by a subsequent transaction?”
- Strongest consistency model for transactional databases is strict serializability
- Guarantees:
  - Global order: All clients will observe the same order of transactions
  - Real time order: The real time order in which transactions occur also constrains the behavior of the database

Consistency Models Can Be Too Constraining

- Constrains behavior of clients in order to provide guarantees
- Strict Serializability requires single-dispatch
  - Each client may only have one outstanding transaction at a time

MDSS

- Strict serializability that allows multiple outstanding transactions per client
- Research is currently focused on examining the performance tradeoffs and building the first prototypes of a system that implements MDSS

Project Ideas

- Deploy systems that implement different consistency models and examine their performance for different workloads
  - Ex: FoundationDB (Strict Serializability)
  - Would read/skim papers to find workloads
- Examine the impact of swapping on workload performance
  - E.g. How does some inference workload perform when only half of its maximum resident set size is allowed to be in memory (and the rest is on the swap device)?

How to build a Content Delivery Network?

- Leon

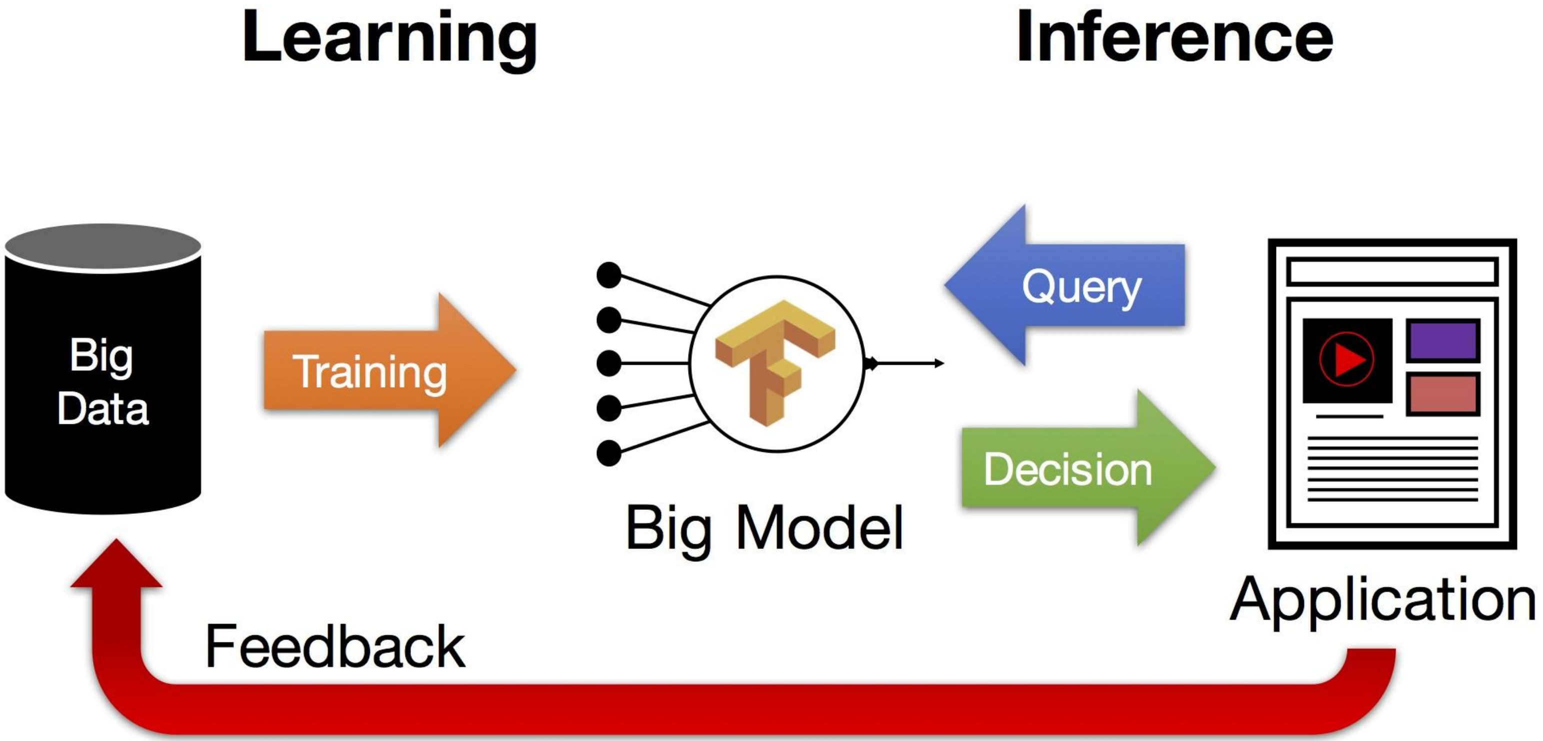
Build your own Content Delivery System

- Build software for a self-hosted CDN!
- In essence: a caching HTTP server
- Considerations:
  - How do you push data out to your CDN nodes?
  - Which caching strategies should you use?
  - Can you distribute data according to demand / storage price / available bandwidth?
    Should you / can you cascade caches?
- Think of a particular application and its demands.
  Can you think of a special feature the CDN should have for that?

System and Networks for Machine Learning!

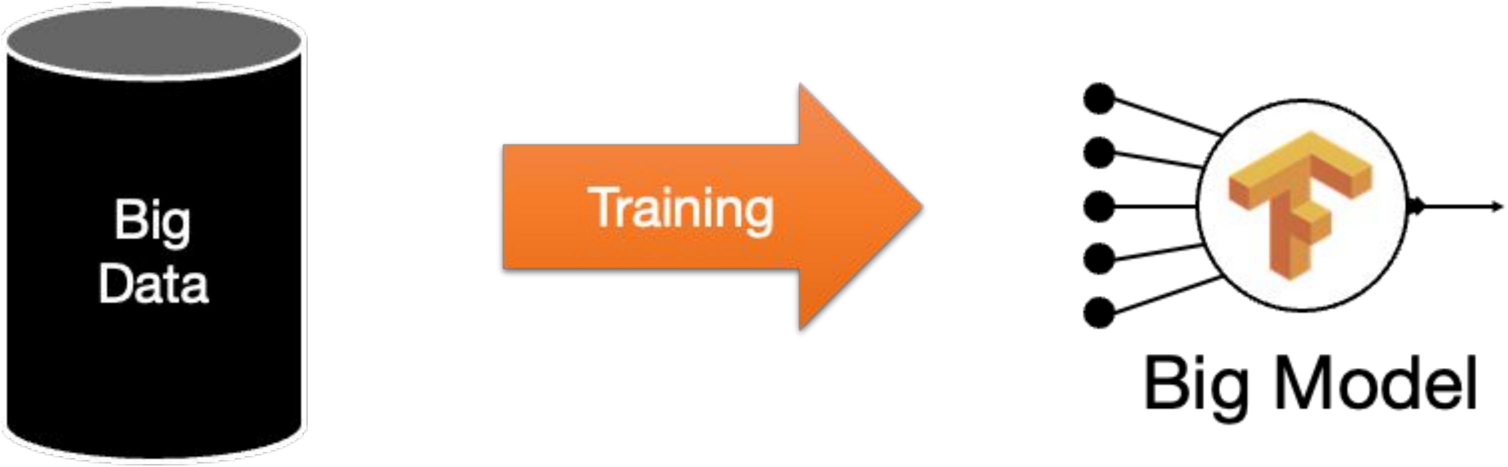
Rui Pan

Background: Machine Learning Lifecycle



Source: https://ucbrise.github.io/cs294-rise-fa16/prediction_serving.html

Background: Training

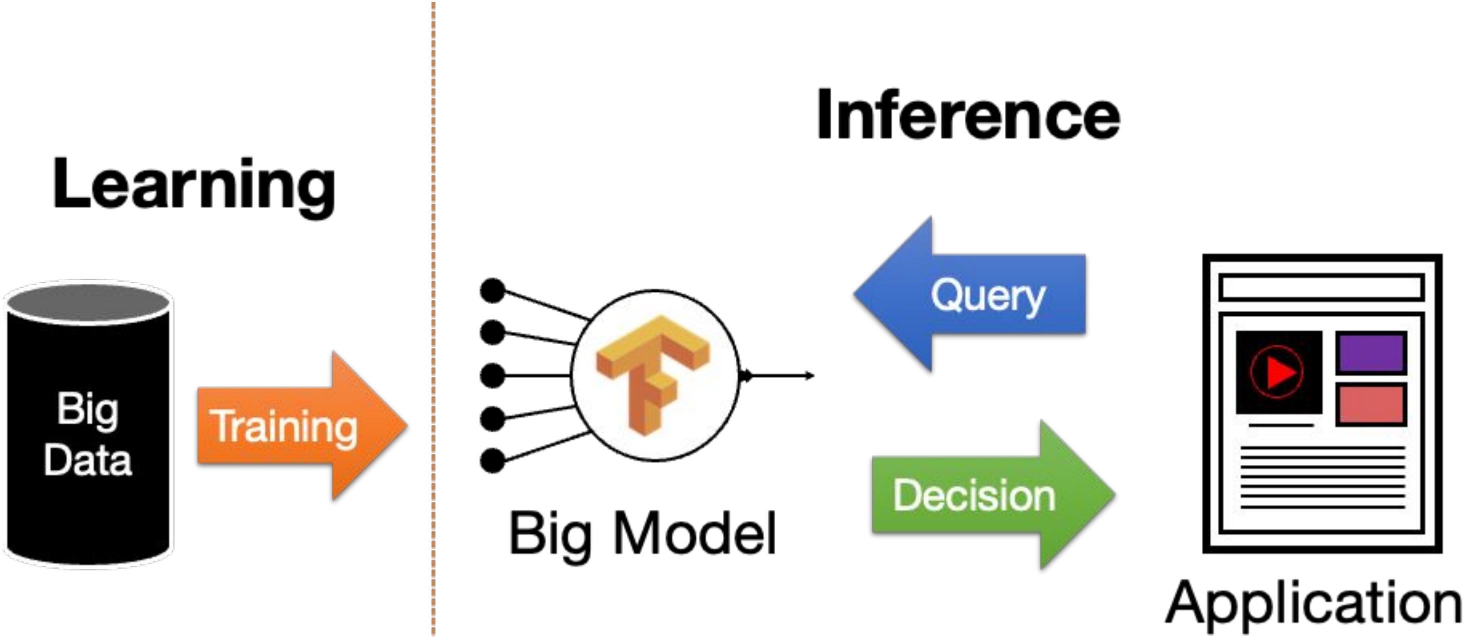


Timescale: minutes to days

Systems: offline and batch optimized

*Heavily studied ... primary focus of the **ML research***

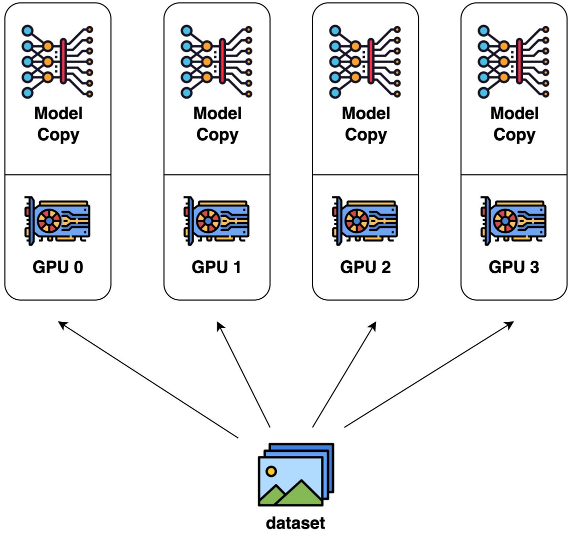
Background: Inference/Serving

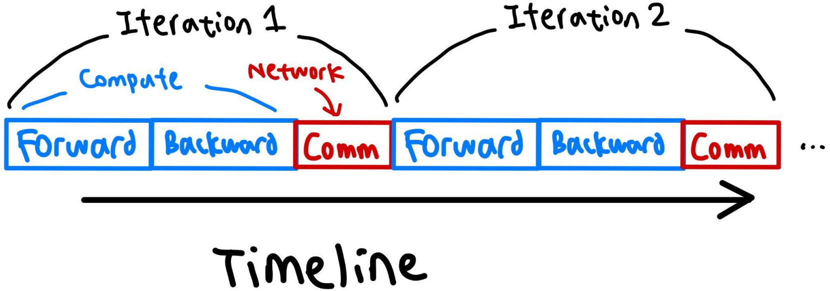


Timescale: ~10 milliseconds
Systems: *online* and *latency* optimized
Less Studied …

Background: Distributed/Parallel Training

- Increasing demand since models and datasets are getting larger and single-GPU training is just infeasible
    - GPT3 175B took 34 days on 1024 expensive GPUs
- Data Parallel: most common form of parallelism due to its simplicity
- Split dataset into shards and place one on each device
- Aggregate and average results across GPUs after every training iteration



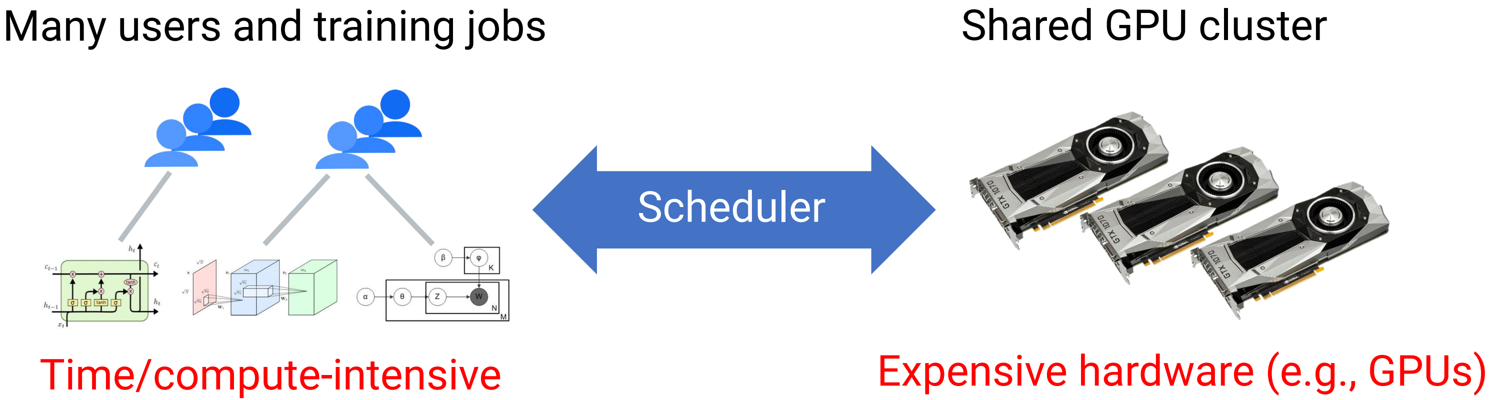


Source: https://arxiv.org/abs/2104.04473, https://colossalai.org/docs/concepts/paradigms_of_parallelism/

Idea 1: Benchmarking Distributed Training on your Laptop

- Given: documentation, tutorials, and scripts for running PyTorch distributed via Python (1, 2, 3), many out-of-the-box CV models, and PyTorch profiler
- Insert logging (e.g. timers) into training scripts to measure the time taken for computation and communication and the size of data transfer in each iteration
- Change a few things in your script: use different models, different batch sizes, and different parallelism degrees
- Try to reason about how they affect the system performance
    - E.g. if we increase the batch size, how will this affect the size of inter-GPU data transmission? What about the computation time in each iteration?
- Optional: implement your own DataParallel module with communication modules in PyTorch and compare with PyTorch DaraPatallel performance, and infer what optimizations PyTorch did

Background: ML Training in Shared Clusters



Many users and training jobs

Shared GPU cluster

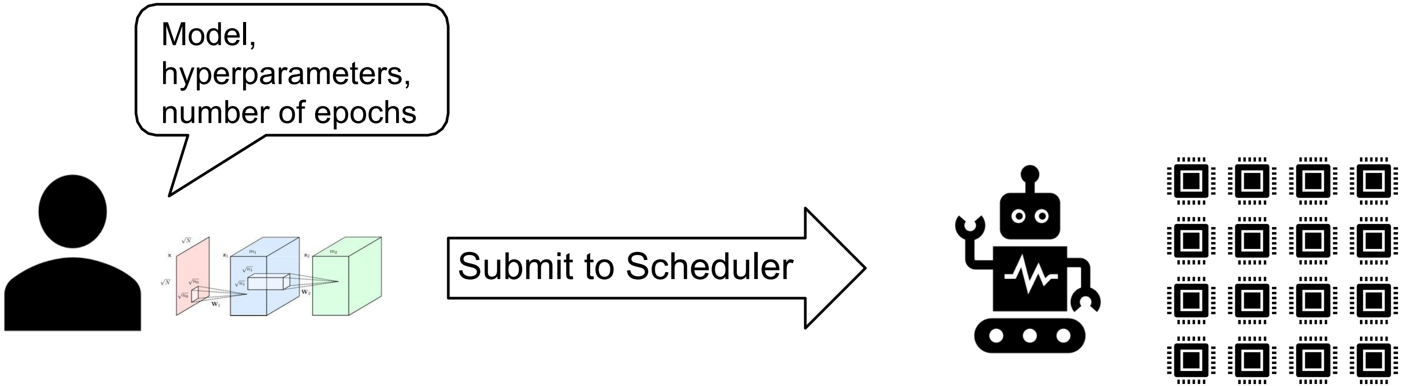
Scheduler

Time/compute-intensive

Expensive hardware (e.g., GPUs)

Cluster scheduler decides how to allocate resources to jobs and place jobs on GPUs in order to minimize training time, maximize cluster utilization, and ensure fairness.

Background: Workflow of ML Training in Shared Clusters



Model, hyperparameters, number of epochs

Submit to Scheduler

Users submit:

- Training script (model, dataset)
- Hyperparameters (batch size, number of GPUs)
- Training duration (number of epochs/iterations)
- Job submission timestamp
- ...

Sample workload (train ResNet-18 on CIFAR-10 using batch size 32 on 2 GPUs for 10780 iterations, job arrived at timestamp 2000s):

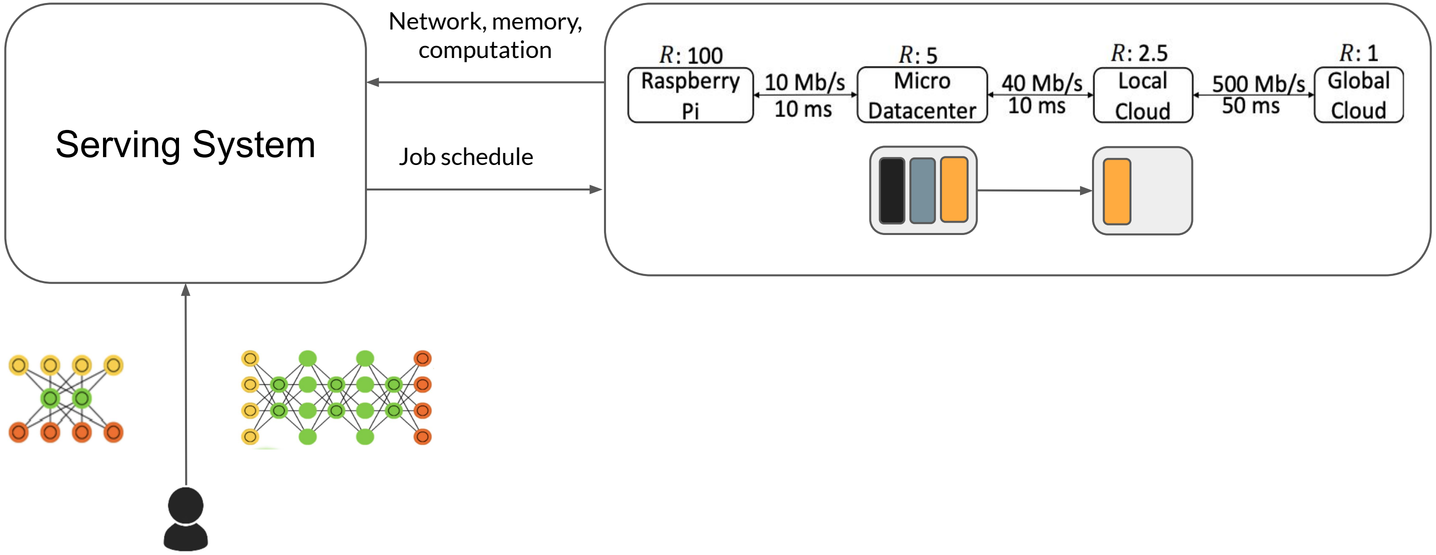
```
ResNet-18 (batch size 32)        python3 main.py
--data_dir=/cifar10 --batch_size 32
--num_steps 10780 --num_gpus 2 --timestamp
2000
```

Idea 2: Develop & Evaluate Simple Scheduling Policies

- Given: Simulator for cluster experiments, workload trace, cluster spec
- Run and evaluate simple and already-implemented scheduling policies
  - E.g. FIFO: schedule jobs in the order they arrive
- Evaluate performance using these metrics
  - Average job completion time (JCT)
  - Makespan: total time to run all jobs in the workload trace
  - Cluster utilization: GPU shouldn't idle, otherwise waste compute and $
  - Fairness: long-running jobs shouldn't cause small jobs to starve
- Implement and evaluate some other simple scheduling policies/algorithms
  - E.g. SJF: Shortest Job First
  - E.g. SRTF: Shortest Remaining Time First

How to make DNN serving efficient across Edge Hierarchy? –Yinwei

DNN Inference Serving across the Edge Hierarchy



Serving System

Network, memory, computation

Job schedule

R: 100 Raspberry Pi

10 Mb/s 10 ms

R: 5 Micro Datacenter

40 Mb/s 10 ms

R: 2.5 Local Cloud

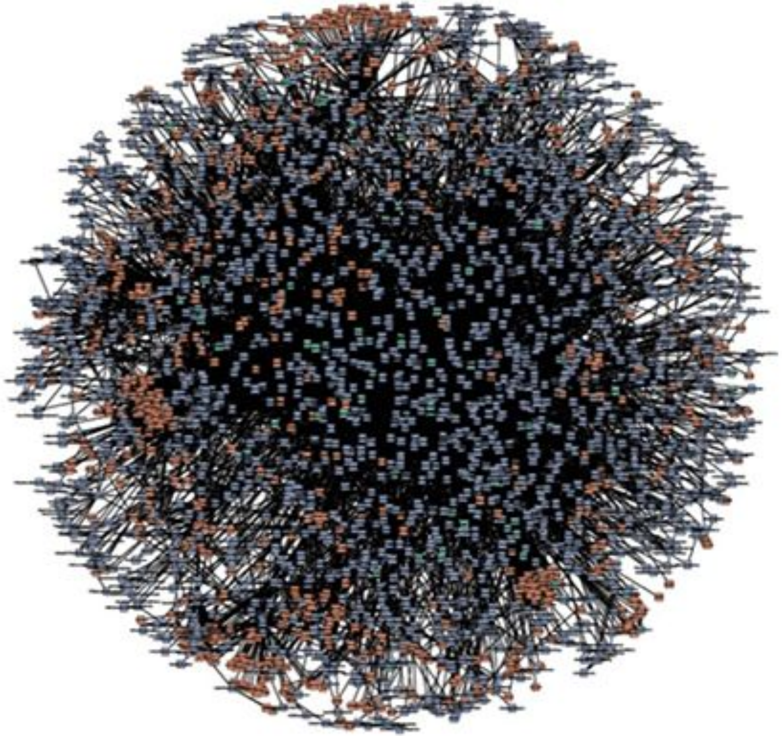
500 Mb/s 50 ms

R: 1 Global Cloud

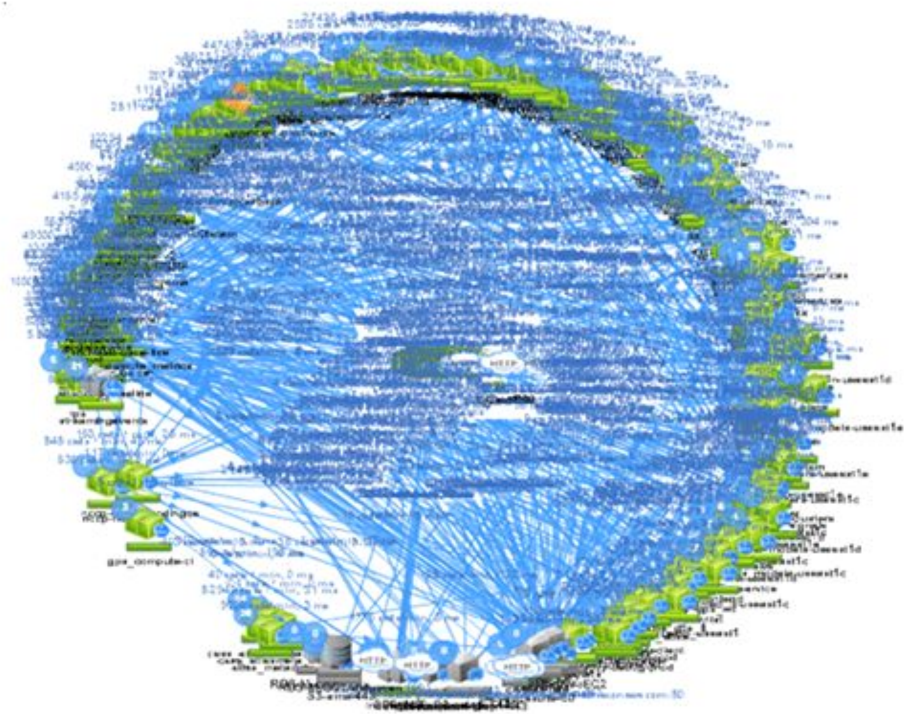
Build your Scheduling System！

- Problem formulation
  - Input: a graph representing edge hierarchy, a list of workloads (model & request rate)
  - Target: place the models (can be splitted) to minimize the serving latency
  - Constraint: memory and compute limit at each node, bandwidth limit between nodes.
- Naive solution: put everything in the most powerful server
- Tooling
  - Machine learning – PyTorch, TensorFlow
  - Profiling tools - DeepSpeed
    - Don’t need run the actual models if models are profiled.

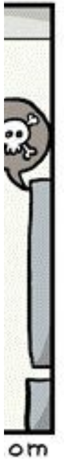
How to debug cloud applications with *ease*?
–Leo



amazon.com

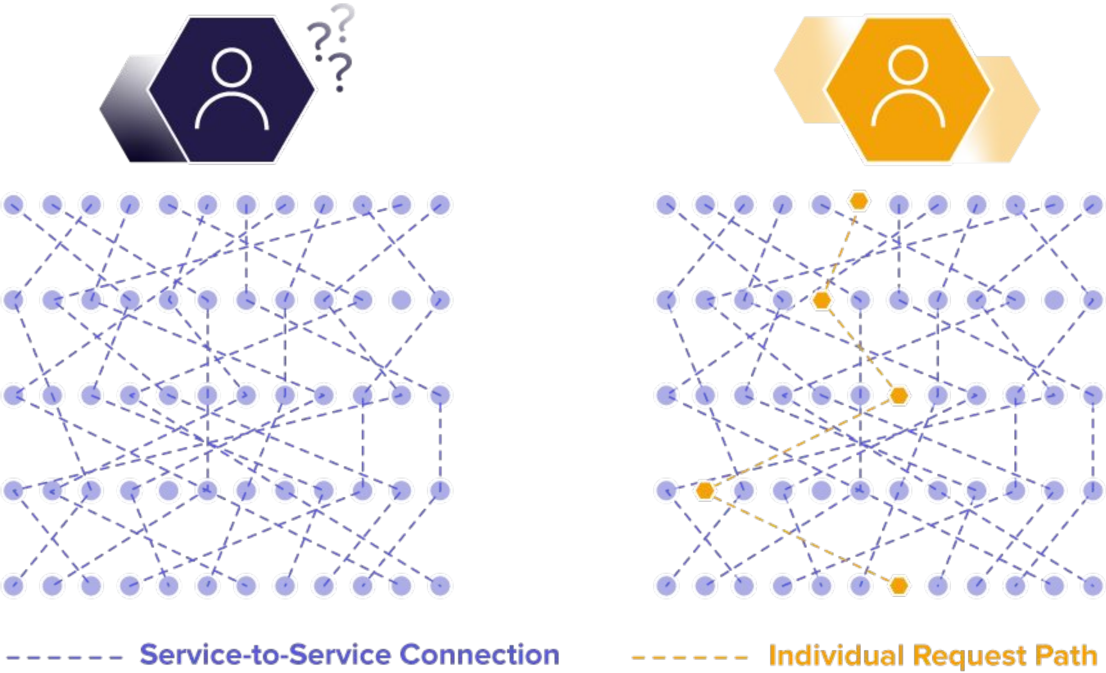


NETFLIX



Automatic capturing!

- Program analysis
    - Expose runtime info leading to a failure!
- Distributed Tracing
    - Linking events!
    - Across services
    - Across executions



Project ideas

- Tracking request paths at runtime
  - Microservice?
  - RPC-based apps?
- Use ML-based methods to automatically detect incidents
  - Where the bottleneck is?

Sources

https://medium.com/javarevisited/distributed-tracing-in-microservices-spring-boot-125272b58ad8

https://www.divante.com/blog/10-companies-that-implemented-the-microservice-architecture-and-paved-the-way-for-others