The City College of New York

CSC 36000:
Modern Distributed Computing *with AI Agents*

By Saptarashmi Bandyopadhyay
Email: sbandyopadhyay@ccny.cuny.edu
Assistant Professor of Computer Science
City College of New York and Graduate Center at City University of New York

October 29, 2025 CSC 36000

Today's Lecture

Optimizing Performance and Efficiency in Distributed AI

- **The Energy Footprint**
- **Quantization**
- **Pruning**

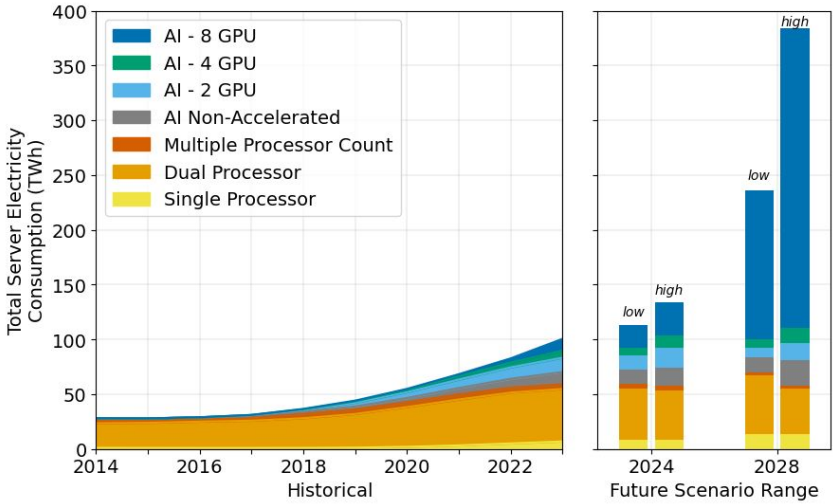
Live Coding Demonstration

Optimizing Performance and Efficiency In Distributed AI

The Energy Footprint

AI presents significant environmental and economic challenges for energy consumption.

When we design distributed AI systems, efficiency should be a top priority.

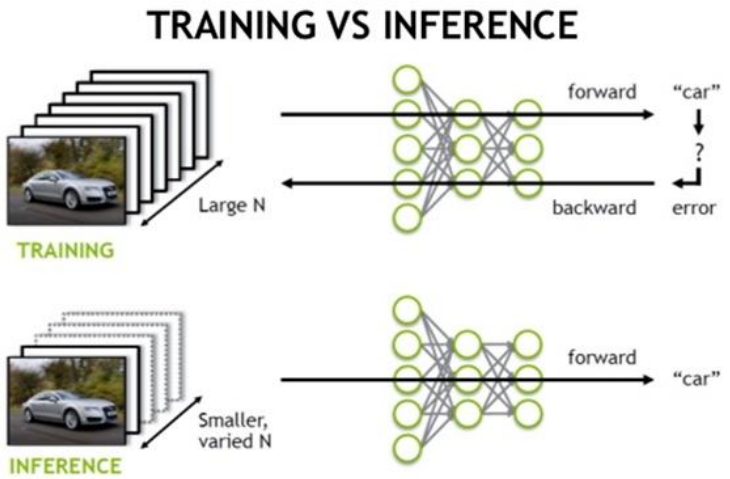


Source: https://iee.psu.edu/news/blog/why-ai-uses-so-much-energy-and-what-we-can-do-about-it

Training vs Inference

Two phases define how AI models use energy:

- **Training:** Models with billions or trillions of parameters take enormous amounts of energy and compute to train.
- **Inference:** The amount of energy for a single query is tiny. Not so much when this is scaled to billions of queries per day.



Strategies for Efficient AI

Often, making AI efficient goes hand-in-hand with adapting it to run on devices with few computational resources (e.g. smart glasses, phones, smart watches)

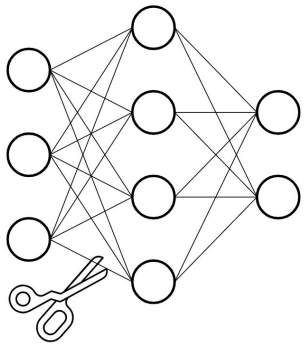
There are two main ways we can reduce the computational resources used by our AI:

- We can reduce the number of parameters (*pruning*)
- We can reduce the size of each parameter (*quantization*)

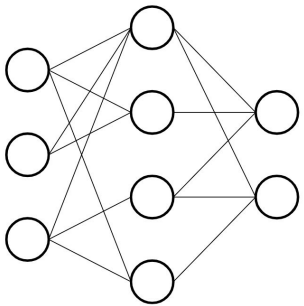
Pruning

When we use *pruning*, we remove parameters that are unimportant or redundant to the overall output of the model

- **Structured Pruning:** Remove whole blocks (e.g. channel in a ConvNet)
  - Creates a dense, but smaller, model
- **Unstructured Pruning:** Remove individual weights
  - Creates a sparse model that requires specialized hardware/software to realize inference speedup



Before pruning



After pruning

Pruning techniques often assume large networks rely heavily on particular small subnetworks for the bulk of their performance

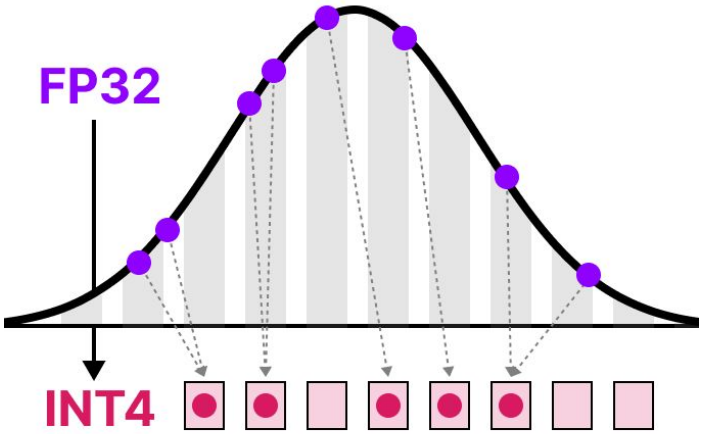
Quantization

By default, most machine learning model weights are represent with 32-bit floating points.

This allows for precise, rich calculations but is memory and energy intensive.

Quantization techniques convert these larger numbers into smaller numbers (e.g. four-bit integers) while attempting to preserve the original performance as much as possible (some performance usually lost)

An FP32 model quantized to INT4 can reduce the memory footprint by a factor of eight!



Live Coding Demonstration:

https://colab.research.google.com/drive/1cX1rsgyCZBJ8qmEReC4VNJAQUbFyDaQu?usp=sharing

Questions?