CSCI 1390 Introduction

Thursday, January 23, 2025

Welcome to CSCI 1390!

- **Course staff**:
  - **Instructor**: Deepti Raghavan (deeptir@brown.edu)
  - **HTA**: Nathan Harbison (nathan_harbison@brown.edu)
  - **UTAs**: Alice Song, Siddharth Boppana (ziyun_song@brown.edu, siddharth_boppana@brown.edu

My Background

- New professor at brown, affiliated with Systems@Brown (PhD from Stanford, undergrad from MIT)

- Research interests: systems for ML, operating systems and networking in datacenters

  - I also teach CS2690, a seminar on datacenter operating systems, in the fall!

  - I generally enjoy thinking about problems related to performance and programmability

Resources

- **Course Website**: Contains course policies, homework and project schedule, lecture schedules and notes
- **Canvas**: Will contain lecture recordings, homework and project assignment PDFs; written homework due on Canvas
- **EdStem**: Class discussion forum
- **Submission system for projects**: still being set up (same as cs300 infra)

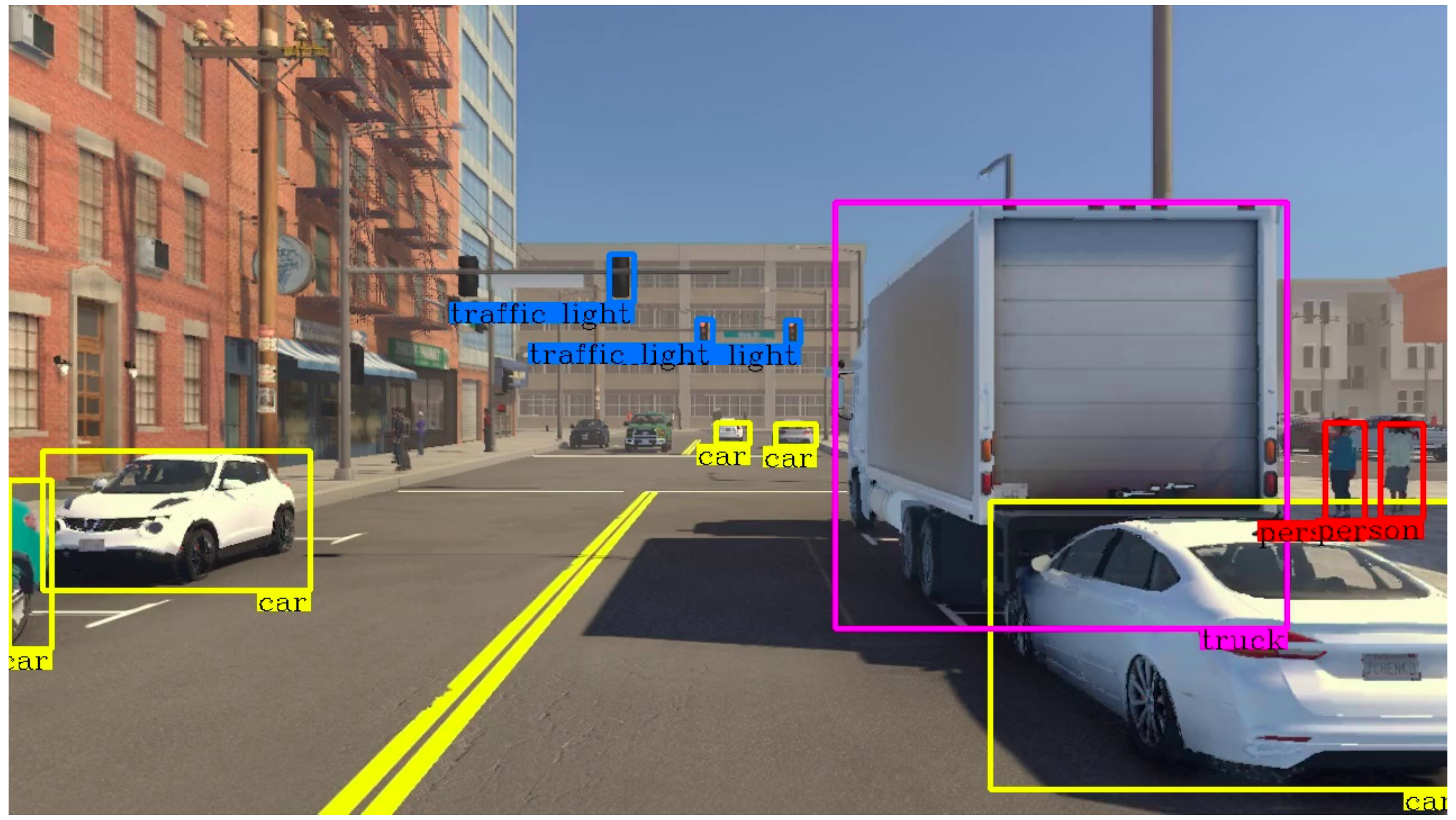
Info About the Waitlist

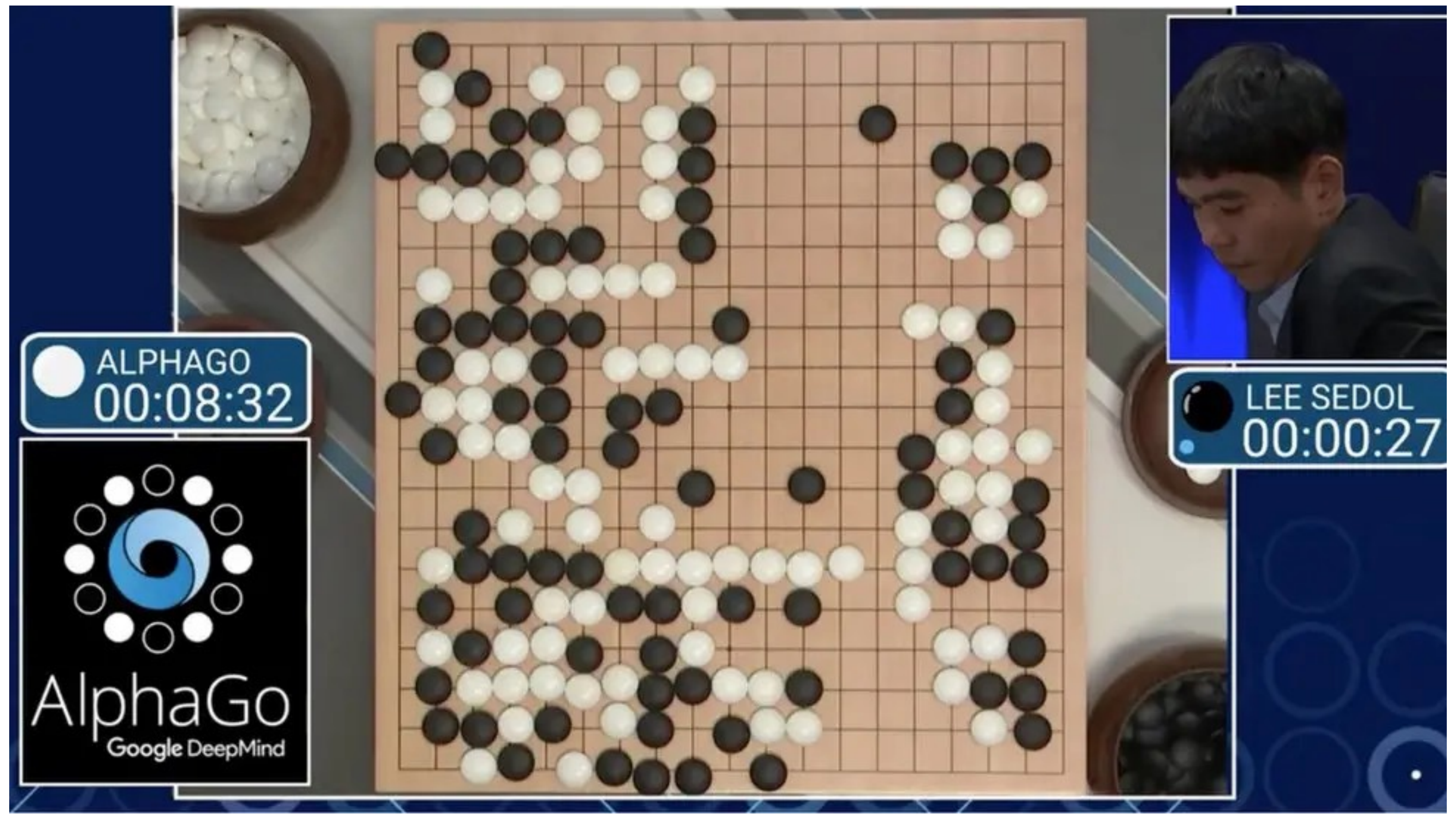
- I had emailed some of you previously (who requested overrides in the fall) to fill out a waitlist
- Class is still full, only if registered drop is there a chance for you to get off the waitlist
- We will be posting a *new form for you to fill out*; so we can keep track of who is still interested

Today's Agenda

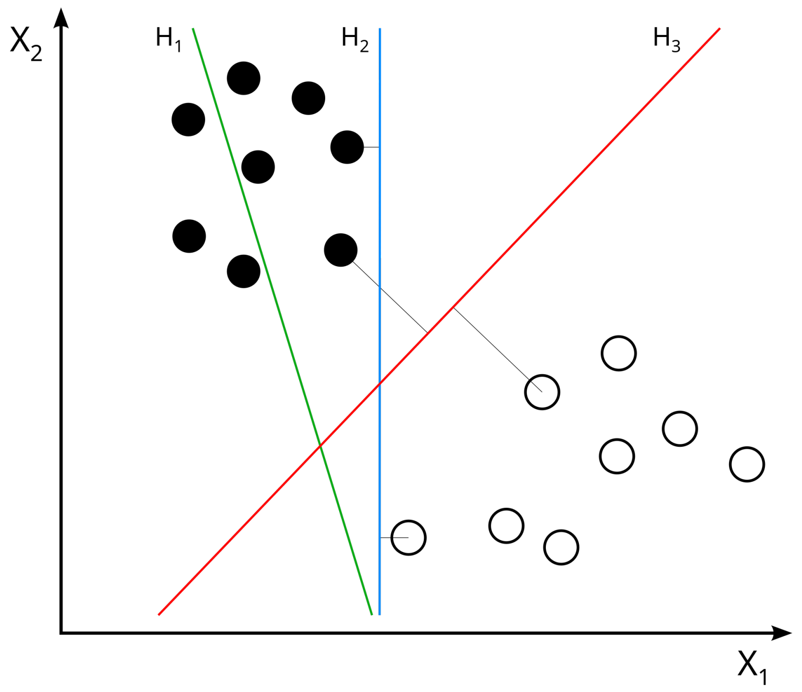
- What is this course about?
- Preview of topics covered in the semester
- Course logistics

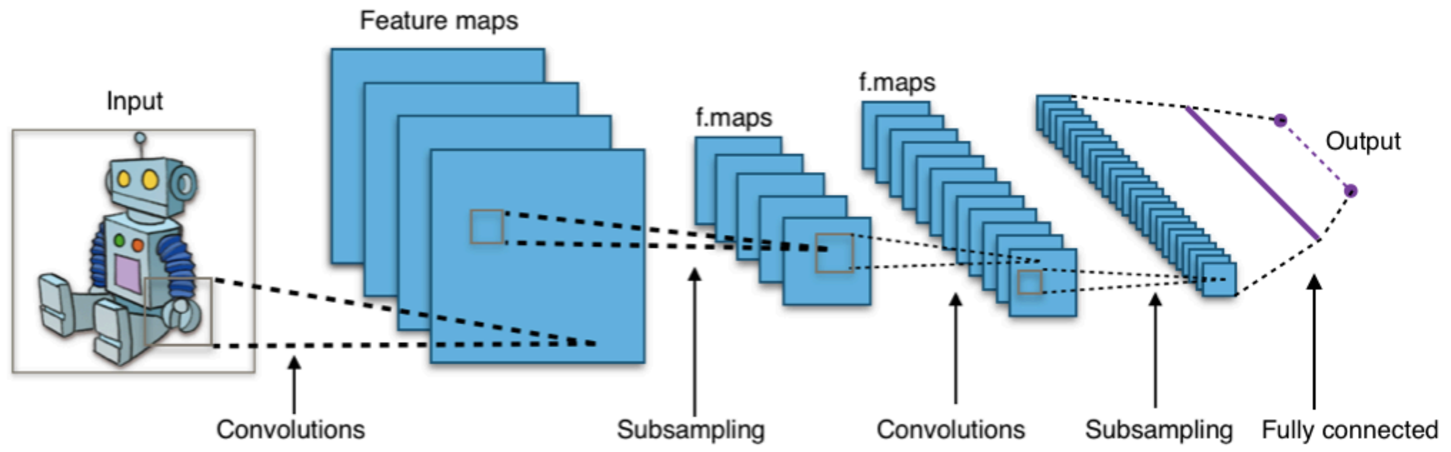
AI and ML is everywhere today!





Most ML techniques were invented in the 80s and 90s



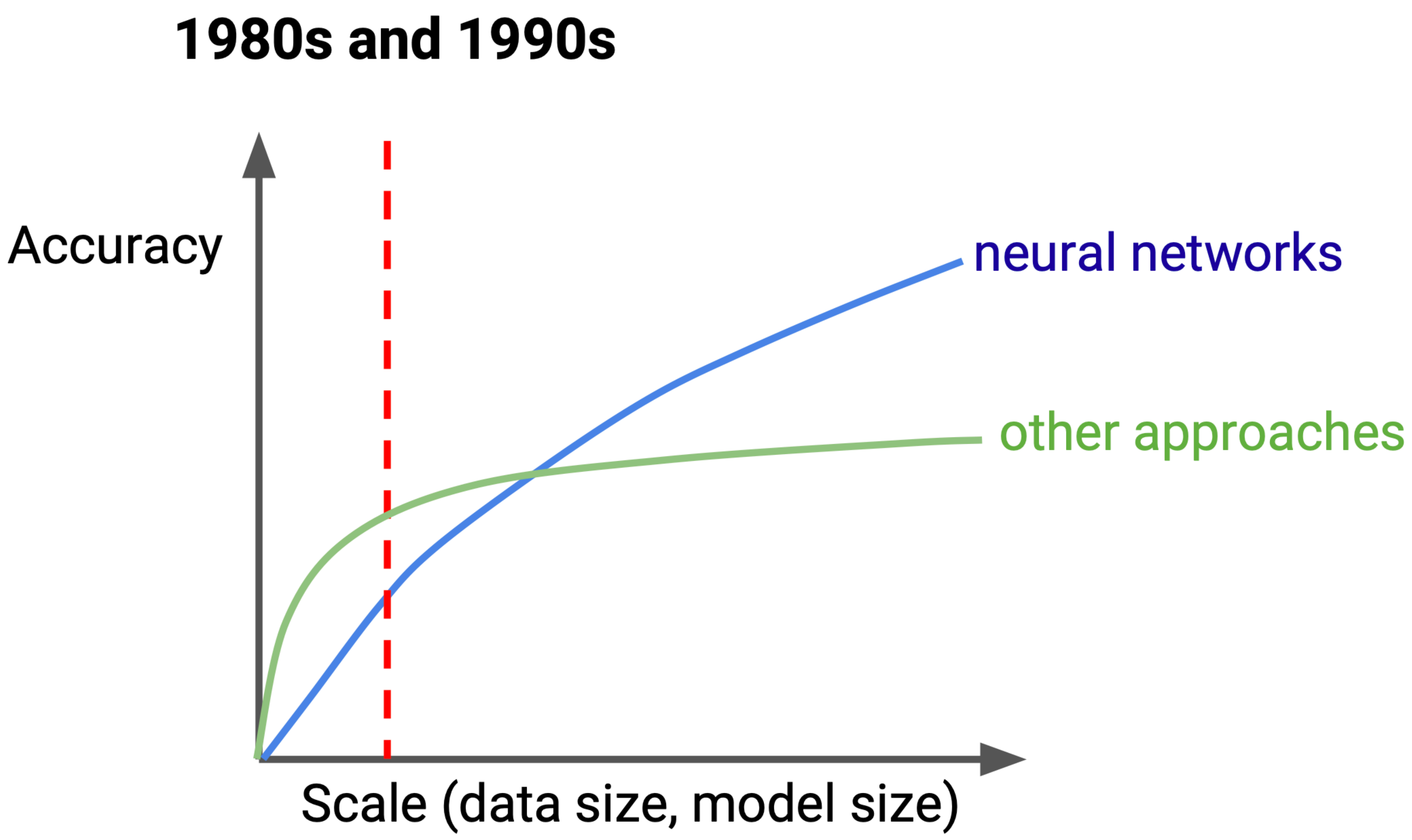


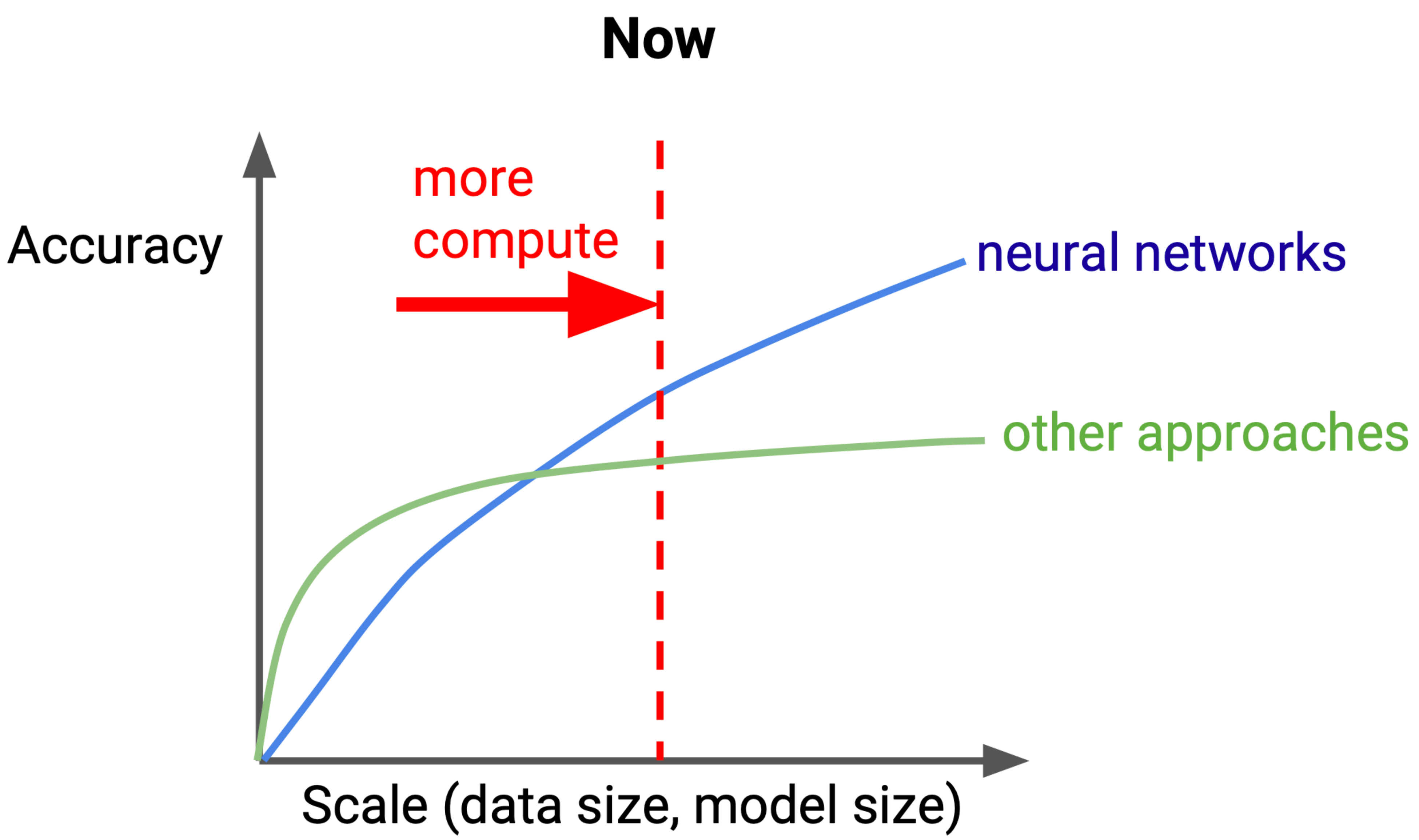
Perceptron Algorithm	Backpropagation	SVM (Support Vector Machine)	ConvNet	GBM (Gradient Boosting Machine)
1958	**1986**	**1992**	**1998**	**1999**

Why didn't ML's widespread success happen in the 90s?

Taken from: Zhihao Jia's class slides at CMU

The Rise of Neural Networks





Taken from: Jeff Deans's Keynote at HotChips 2017

More “big data” arrived in the 2000s

flickr

MTurk

kaggle

IMAGENET

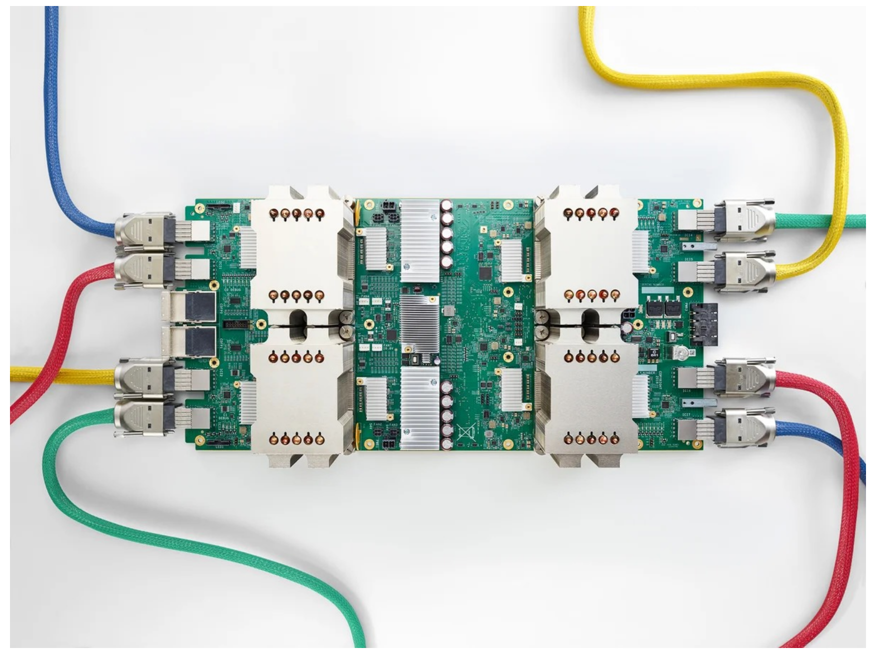
2001 2004 2005 2009 2010

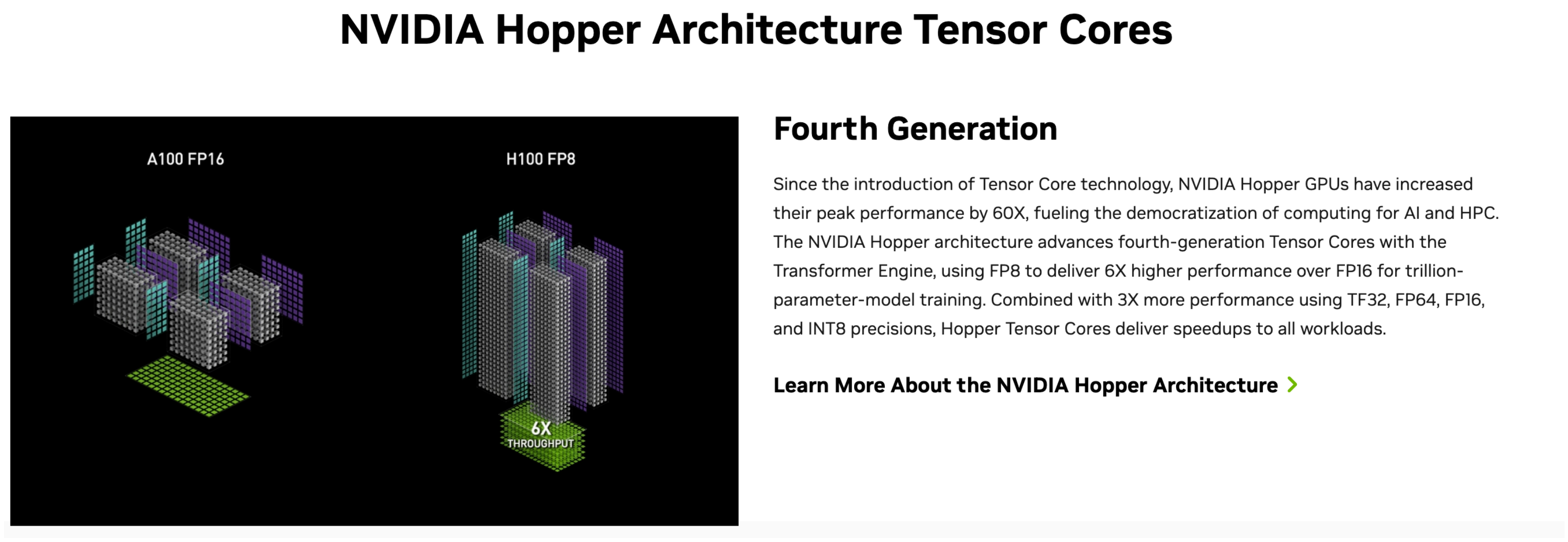
Large-scale training datasets become available

Adapted from: Zhihao Jia's class slides at CMU

AI hardware has become widely available in the 2000s

NVIDIA CUDA

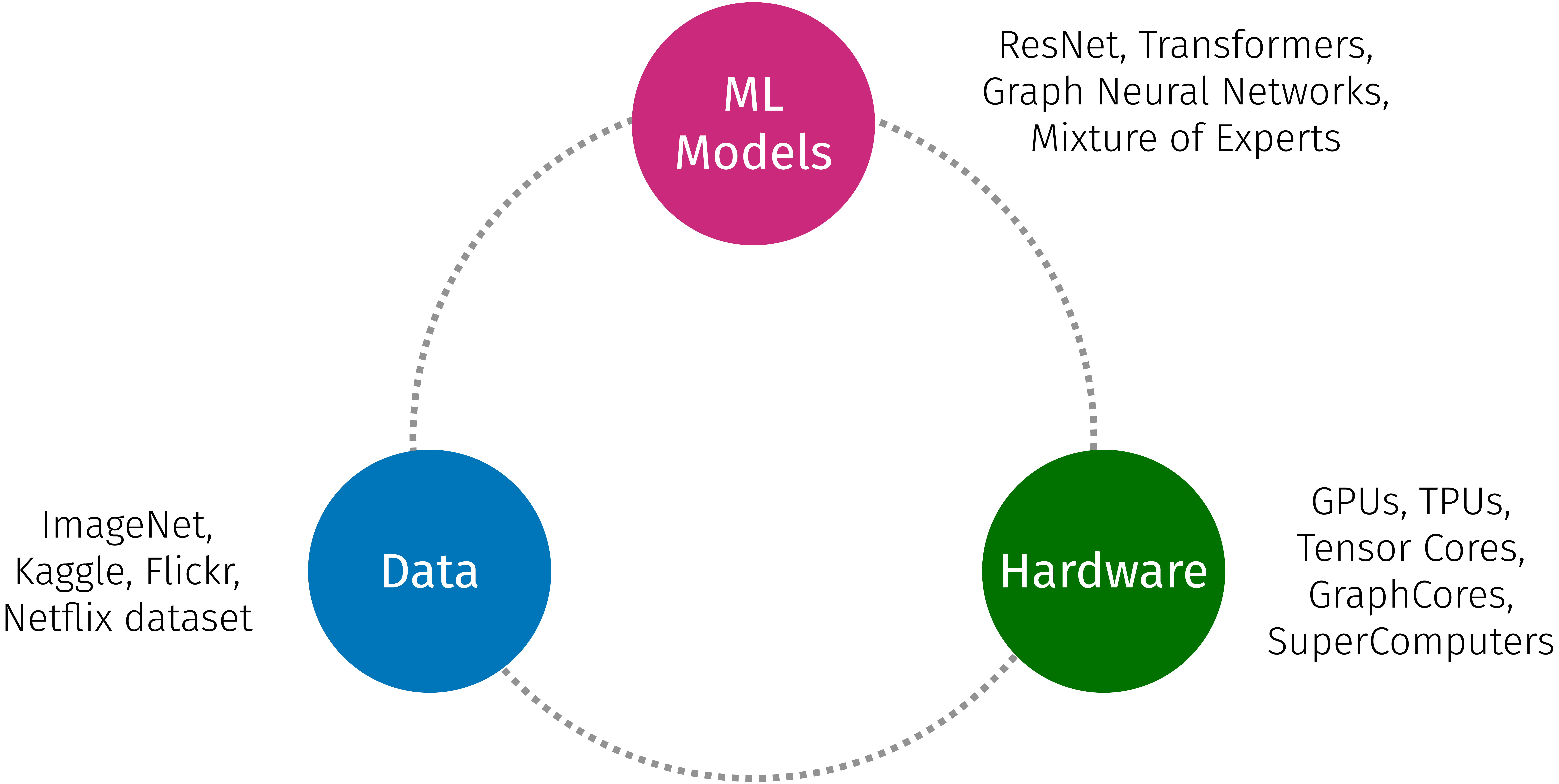




2006 **2007** **2016** **2017**

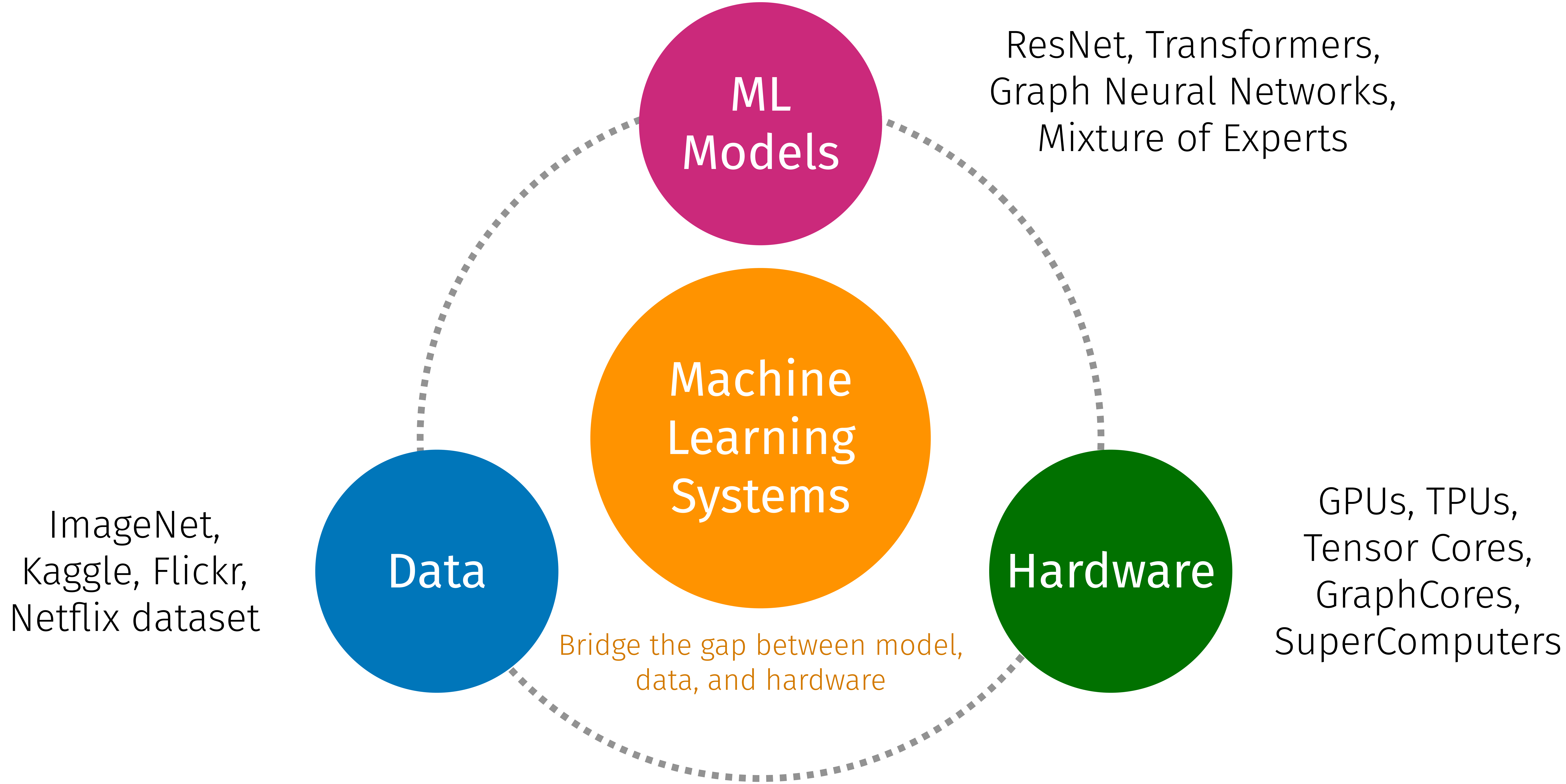
Adapted from: Zhihao Jia's class slides at CMU

Secret Ingredients to ML's Success



Taken from: Zhihao Jia's class slides at CMU

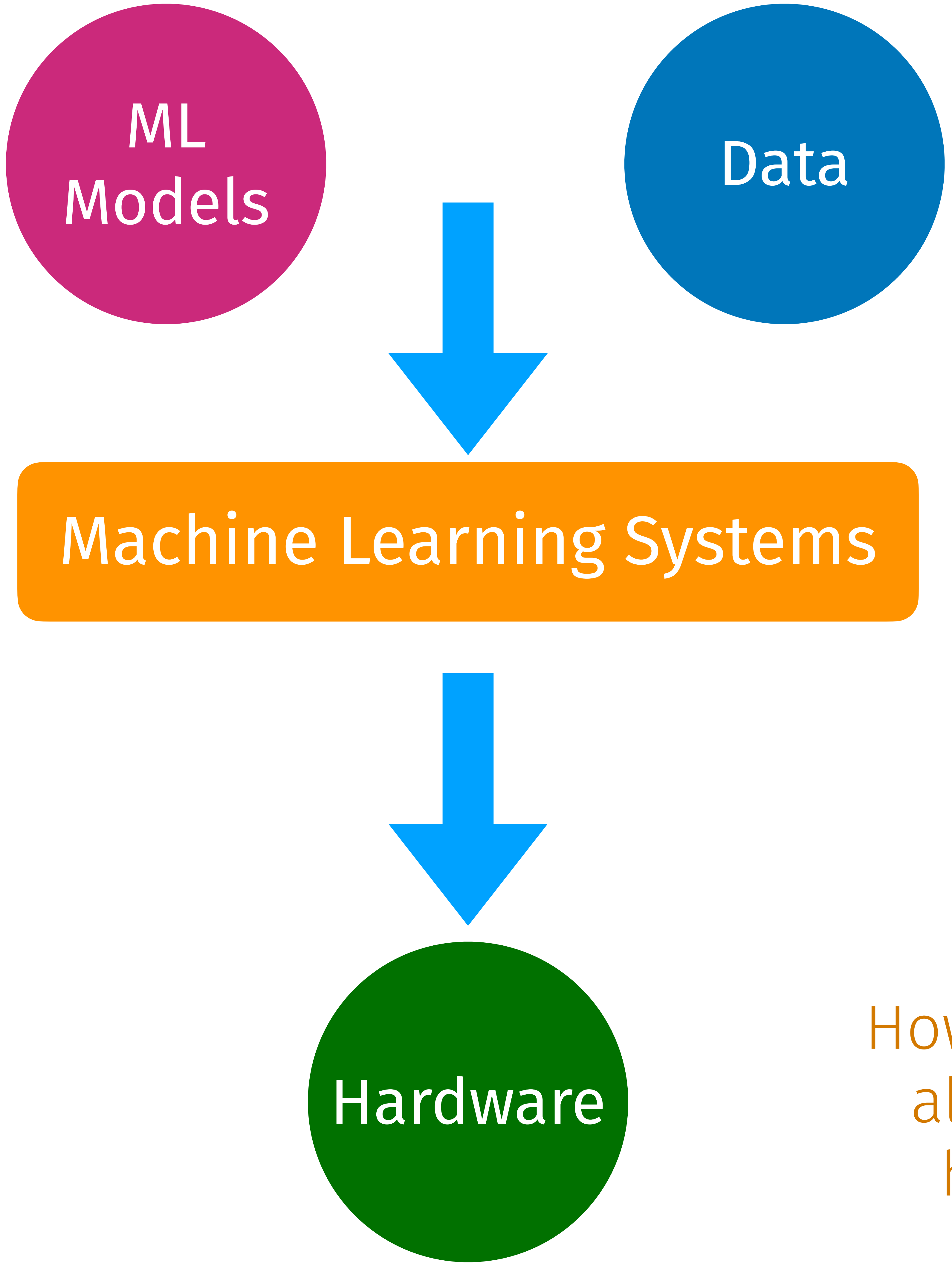
What is machine learning systems?



Taken from: Zhihao Jia's class slides at CMU

ML Systems Improve Efficiency, Programmability

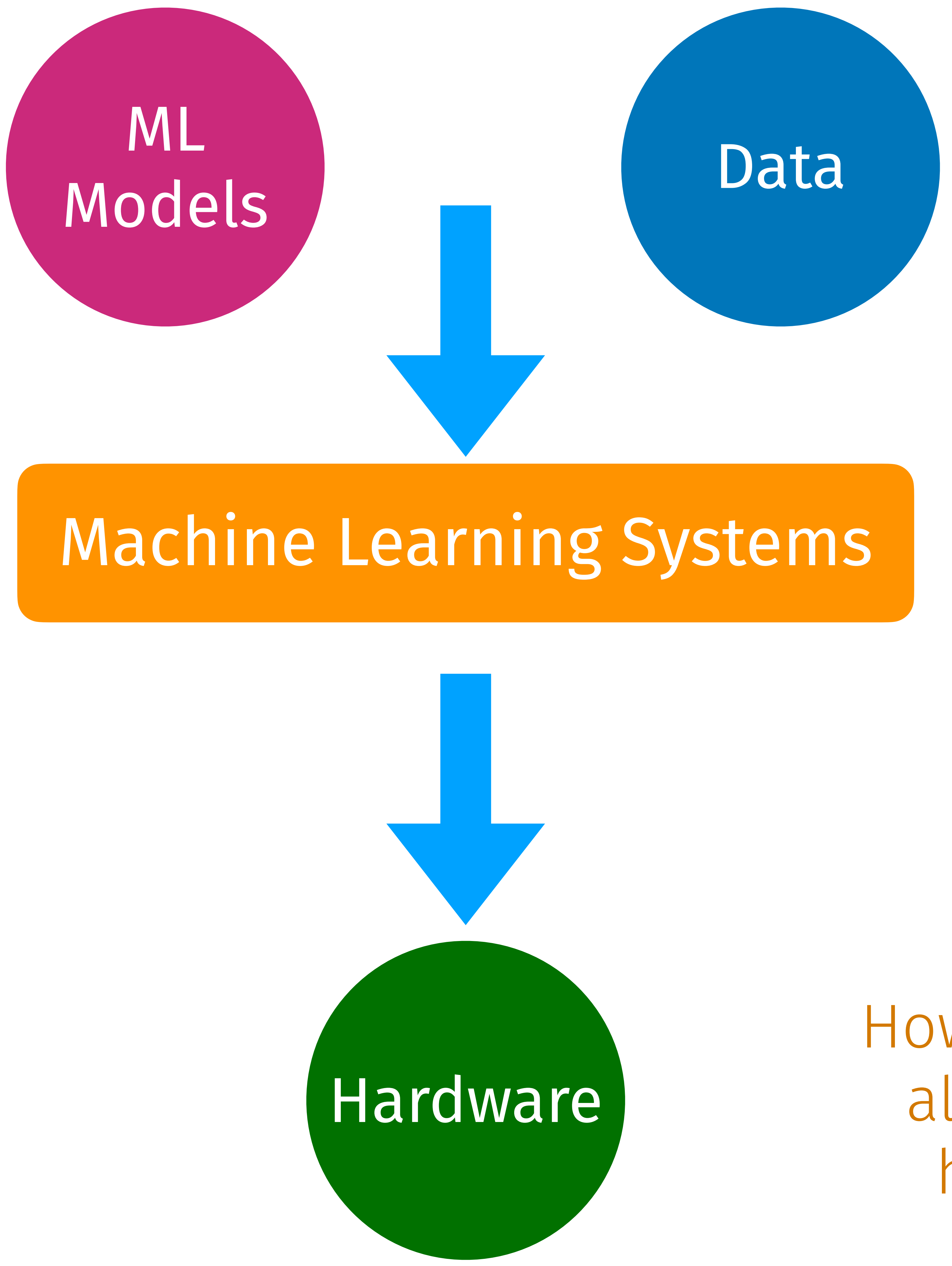
What interfaces should we provide to prototype different ML applications (training or inference)?



How do we effectively deploy ML computations on modern, heterogeneous hardware?

How do we create new algorithms that are hardware-aware?

ML Systems Improve Efficiency, Programmability

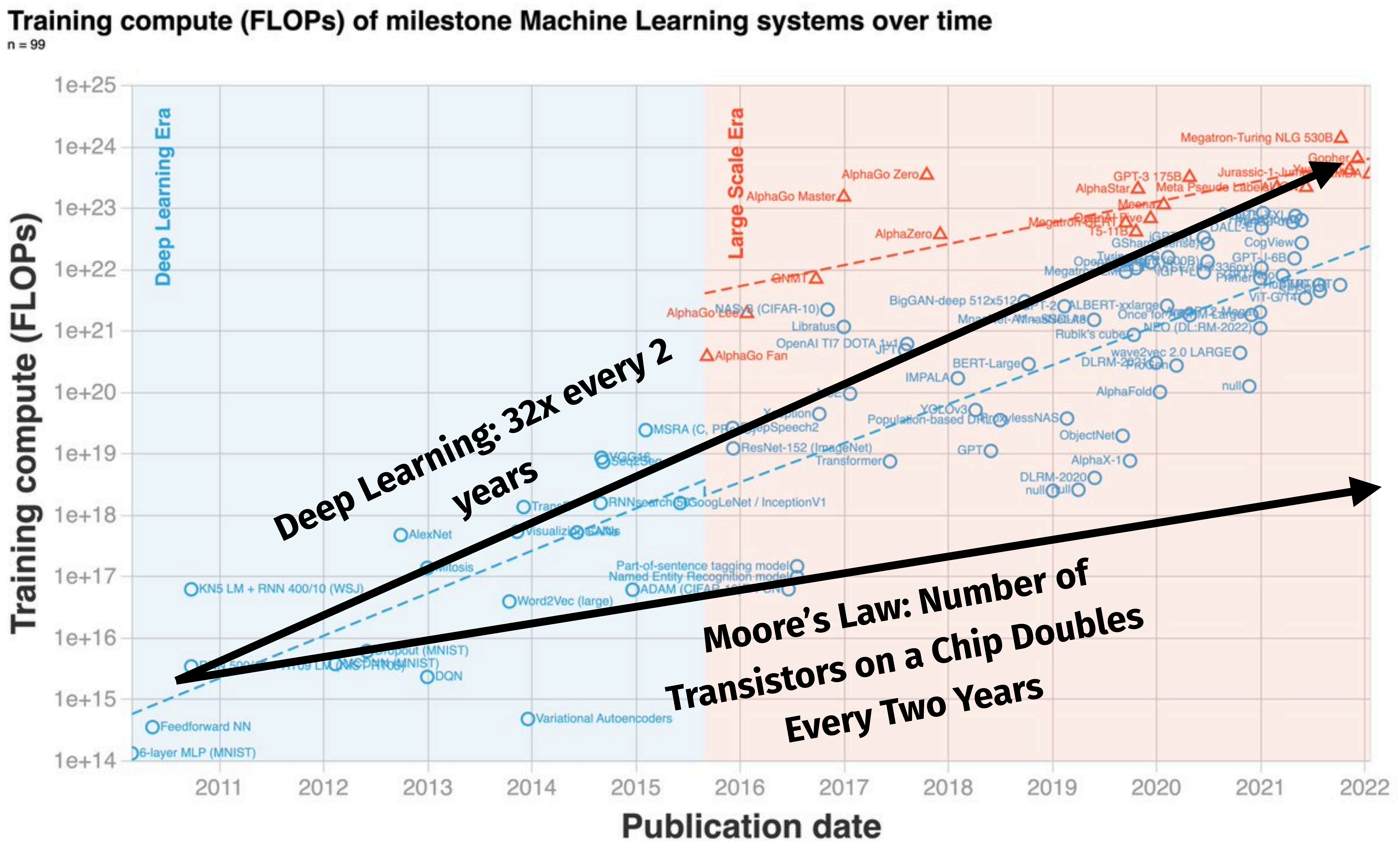


What interfaces should we provide to prototype different ML applications (training or inference)?

How do we effectively deploy ML computations on modern, heterogeneous hardware?

How do we create new algorithms that are hardware-aware?

Training Costs are Going Up



Moore's Law has also ended, leading to wide scale efforts in specialized hardware

Original Graph: Sevilla et al., "Compute Trends Across Three Eras of Machine Learning", 2022

Taken From: CS229S at Stanford 2023, Lecture 1, Given by Azalia Mirhoseini

Models are Getting Larger

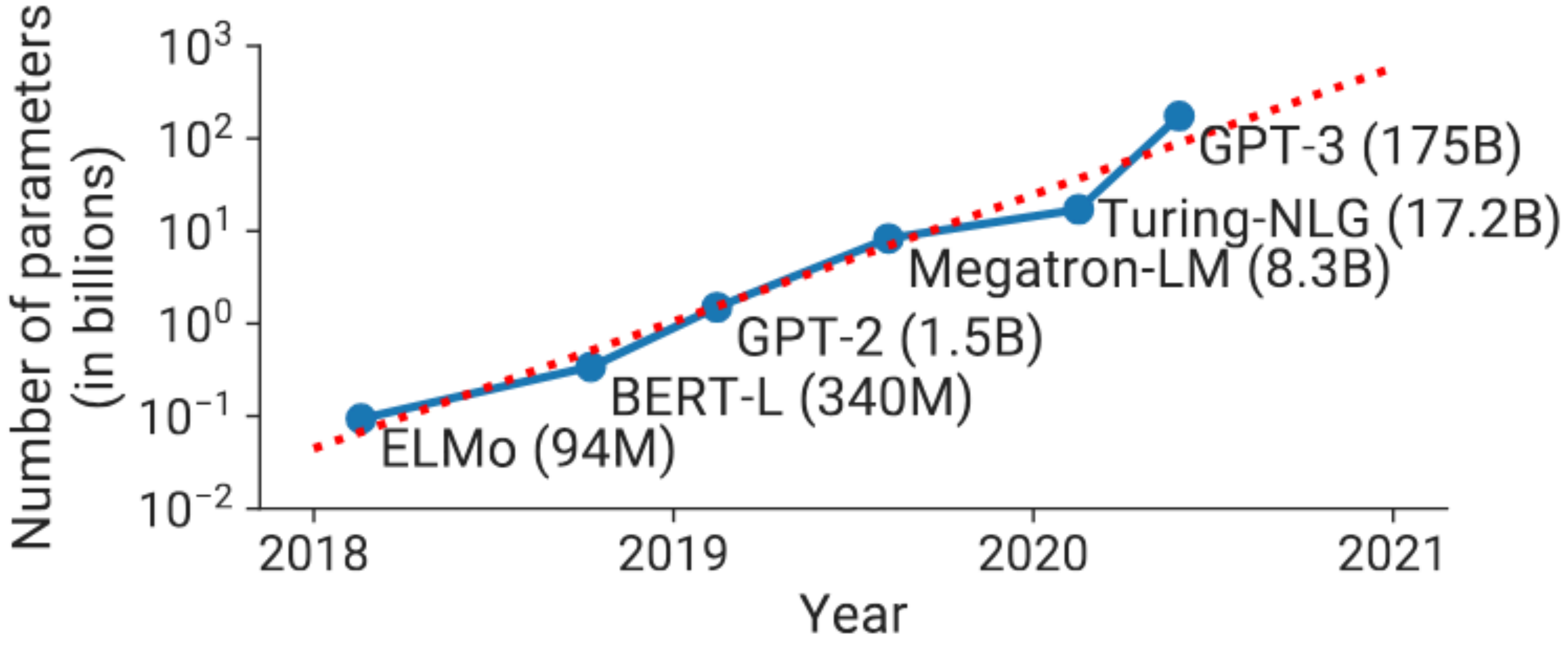


Figure 1: Trend of sizes of state-of-the-art Natural Language Processing (NLP) models with time. The number of floating-point operations to train these models is increasing at an exponential rate.

Device	Memory Size
V100 GPU (2018)	32 GB
TPU v3 (2019)	32 GB
A100 GPU (2020)	40 GB
A100 GPU (2022)	80 GB

Original Graph: Narayanan Et. Al, Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM

What will you learn about in CSCI 1390?

- Challenges in distributing ML workloads: memory vs. communication tradeoffs
- How ML algorithms use hardware: memory bandwidth vs. arithmetic intensity
- Custom hardware used for deep learning (and programming GPUs), ML compilers and frameworks
- Methods that reduce the memory overhead of deep learning: quantization, distillation, sparsity
- Compound AI systems: using AI and non-AI tools to achieve complex tasks
- Survey of selected topics in systems ML research: state space modeling, video analytics, debugging, security, applying ML techniques to systems

What background do you need for the class?

- Basic background in systems: processes, threads, networking, OS-paging (cs300/330!)
  - Example: we will study pipeline parallelism, a method to train models that overlaps computation with communication
- Basic background in ML:
  - Example: we will study attention, and how to design efficient attention algorithms (but we will have a recap on the transformers architecture)

What is this class NOT?

- How to use LLMs effectively (e.g., prompt engineering)
- How to use PyTorch classes for distributed training (we will study how these work)
- How to use public-facing ML APIs (in the cloud or otherwise)
- New ML architectures (we will study systems approaches that make existing architectures more efficient)
- A way to get access to cloud resources

Class Structure

Class Components

- Programming-Heavy Projects: 65 %
- Written Homeworks: 20 %
- PLQs: 10 %
- Class Participation: 5 %

Class Projects (4): 65 %

- Project 1: Exploring parallelism strategies for training
- Project 2: Attention and KV Caching (taken from Stanford cs229s)
- Project 3: Optimizing Cuda Kernels
- Project 4: Vector Databases
- Shoutout to our TAs for creating projects 1, 3, and 4 from scratch!
- Schedule posted on website
  - Project 1 will be posted **by Monday morning**, we are trying to post it earlier :)

Class Projects (4): 65 %

- The deliverable consists of both the **code itself** and a **writeup containing**:
  - Graphs that analyze tradeoffs of your implementation and explanations of the trends in these graphs (these are prompted in the assignment clearly)
  - Back-of-the-envelope math questions related to reasoning about performance tradeoffs (e.g., FLOPs required to compute attention)

Class Projects (4): 65 %

- How each project is graded:
  - Code:
    - Manually inspect the code to check for completeness
    - Run the code to check code works on our infrastructure
    - Grade the implementation on its achieved performance
  - Writeup: Clarity of your writeup and answers to conceptual and math questions
  - Some projects will have an additional live grading meeting where you talk to the TA through your code and writeup

Class Projects (4): 65 %: Infrastructure

- Projects 1 and 4 are designed to be run locally; we will provide an environment with which you can work
  - We will try to set up a way for you to submit your homework to the grading server before the due date to check if it works there, and performs as expected
- Projects 2 and 3 are designed to be run on Brown CS's Hydra Compute Cluster
  - If you are enrolled in the CS department, you should have access

Class Projects (4): 65 %: Late Policy

- TLDR: 144 late hours total, can be divided across the projects in any way, no more than 72 per assignment; hours between midnight and 7 do not count
- Detailed policy on website (taken from cs300)
- Can only be used on projects, not on written homework assignments or PLQs

Written Homeworks (x2)

- For each written homework, we will assign technical readings (e.g., 1-2 research papers or tech reports) and ask conceptual questions
  - First homework will focus more on making sure you understood the key ideas in the paper
  - Second homework will add an additional layer of asking you to think critically about the reading and review it
  - Homeworks will be graded on answers, clarity, and writing
- If you enjoy this, you will also enjoy many of our 2000-level classes in systems!

PLQs (10 %)

- After most lectures, course staff will post a PLQ to EdStem (there are about 22 lectures in total)

- PLQ is due before the start of the next lecture

- Late PLQs are not accepted, but you can miss up to 5 PLQs, no questions asked

  - However, if you miss more than 7 PLQs, you will not get any credit for the PLQ portion of the grade

Class Participation

- Ask questions in class, hours or EdStem
  - When you speak in class or hours, please say your name so we can take note!
- Answer each other’s questions in hours or EdStem

Collaboration and GenAI policy

- We encourage working together to think about assignments, questions, and class content
- All code, writeup, and answers to math questions must be your own work!
- Please credit all collaborators and external sources in your writeup
- GenAI: do not use genAI to fill in code snippets, and do not generate answers to the writeup questions or math questions.
- You agree to NOT post the solutions for any projects publicly

Talk to us!

- If you find yourself struggling, please talk to us!
- If you have an accommodation that affects your learning or classroom experience, please let us know about it
- We will be posting an anonymous feedback form where you can bring up any concerns about the class or give any suggestions to improve your learning experience

Final Thoughts for Today

- We live in exciting times: many of the techniques and algorithms you will be learning about *were introduced within the last decade, some even within the past five years*

- Please be cognizant that this is a brand new class, so:

  - There will be technical difficulties with the projects.

- Credits (full list on the website):

  - cs229s at Stanford

  - CS 15-849 at CMU