https://hao-ai-lab.github.io/dsc204a-f25/

DSC 204A: Scalable Data Systems
Fall 2025

Staff
Instructor: Hao Zhang
TAs: Mingjia Huo, Yuxuan Zhang

@haozhangml @haoailab
haozhang@ucsd.edu

Instructor

Hao Zhang (https://cseweb.ucsd.edu/~haozhang/)

- Ph.D. from CMU CS, 2020

- Projects: Parameter server, auto-parallelization

- Took 4-year leave to work for a "not-so-successful" startup (raised 100M+), 2016-2021

- Projects: Petuum, MLOps

- Then postdoc at UC Berkeley working on LLM+systems, 2021 – 2023

- Projects: vLLM, Vicuna, lmsys.org, Chatbot Arena

- Then co-founded a small startup and acquired by SNOW and started at UCSD

My Lab: https://hao-ai-lab.github.io/

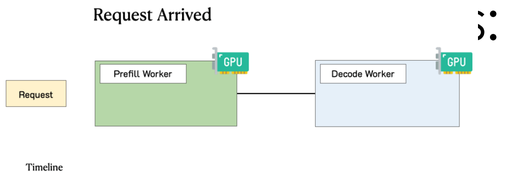
Research Area: Machine Learning + Systems

Recent topics (some will be covered in the final part of this course):

- Fast LLM Inference and Serving
- Large-scale distributed ML systems, Model parallelism, etc.
- Open source LLMs, data curation, evaluation

I also work for snowflake for 20% of my time (which is relevant to this course)

Sor :



DistServe

vLLM

vllm.ai

Starred 58.8k

FAST VIDEO

Starred 2.3k

Today

What is This Course and Why Study It

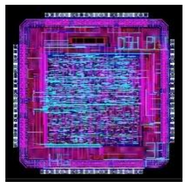
Course overview

Logistics

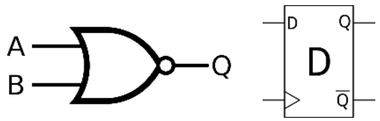
Warm up (If time permits)

What is this course about: **data-centric system** course

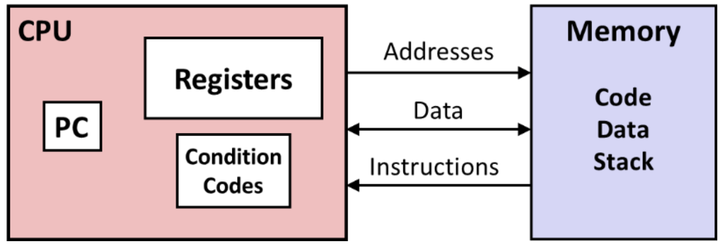
Computer Designer



Gates, clocks, circuit layout, …



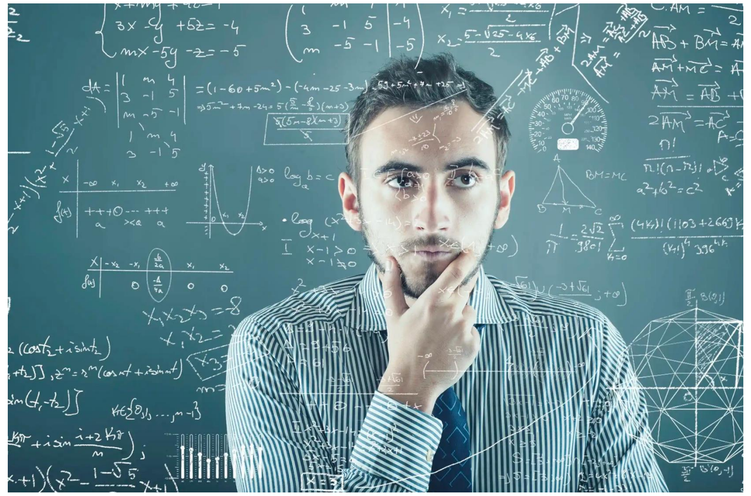
Assembly programmer



C programmer

```
#include <stdio.h>
int main(){
int i, n = 10, t1 = 0, t2 = 1, nxt;
for (i = 1; i <= n; ++i){
printf("%d, ", t1);
nxt = t1 + t2;
t1 = t2;
t2 = nxt; }
return 0; }
```

Data science



What is this course about: data

DATA

How to store and access the data?

- Computer Organizations
- OS
- Databases
- Data encoding

What is this course about: drawing values from data

BIG

DATA

How to store and access **big** data?

- Cloud
- Distributed storage
- Parallelisms, partitioning
- Networking

One classic example: Dataframe API

What is this course about: access and process big data

How to access and process big data?

- Distributed computing
- Batch and stream processors, dataflow systems, programming models
- Big data tools: Hadoop, Spark, Ray

One Modern example: LLMs

AI: new ways of drawing values from big data

LLMs: powerful AI that can scale with **data size**

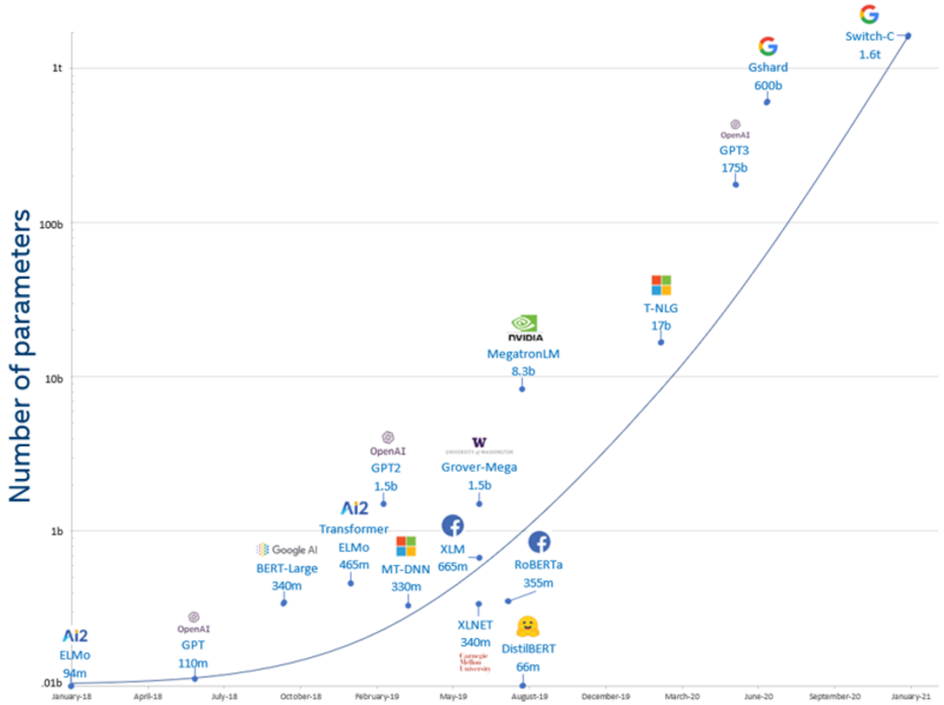


Figure 1: Exponential growth of number of parameters in DL models

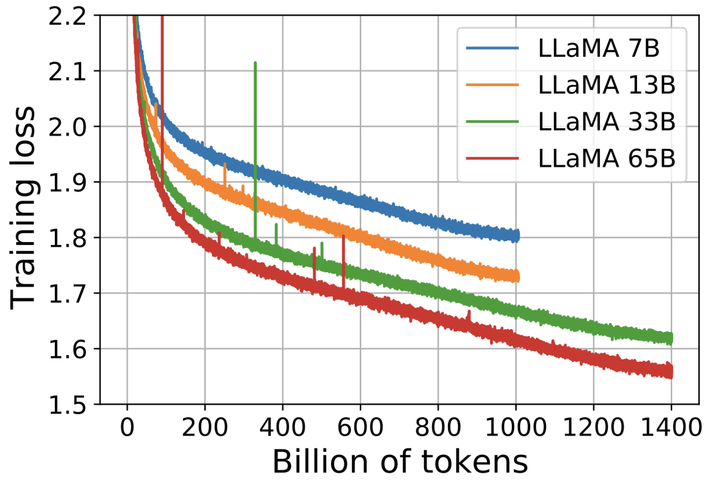


Figure 1: **Training loss over train tokens for the 7B, 13B, 33B, and 65 models.** LLaMA-33B and LLaMA-65B were trained on 1.4T tokens. The smaller models were trained on 1.0T tokens. All models are trained with a batch size of 4M tokens.

What is this course about: drawing values from data

BIG DATA

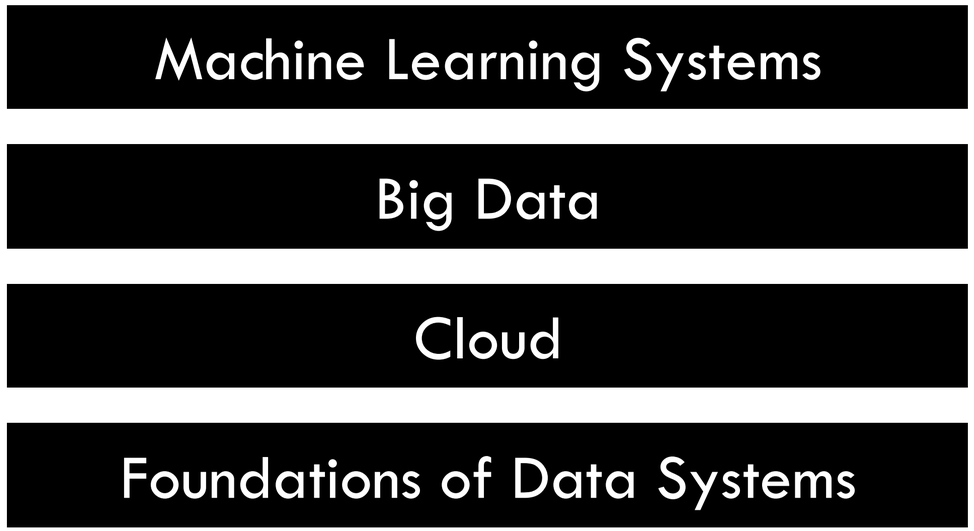
+AI

AI: New ways of drawing values from Big data

- ML frameworks, dataflow graphs
- Distributed ML systems, ML parallelisms
- Large language model systems

Hence the course is organized into four parts

- Foundations of data systems: OS, storage, compute
- Cloud: Cloud storage, network, parallelism, etc.
- Big Data: data processing and programming
- ML systems: ML frameworks, parallelism, LLM training and serving



What is this course about?

- Foundations of data systems
  - Data models, big data storage and retrieval, and how to encode information when you store data, etc.
  - ~~Transactions, synchronization, consistency, consensus~~

What is this course about?

- Cloud and Distributed Systems
  - Cluster, cloud, network, replication, partition, consistency, etc.
  - ~~RPC, Caching, Fault tolerance, Paxos, Concurrency~~

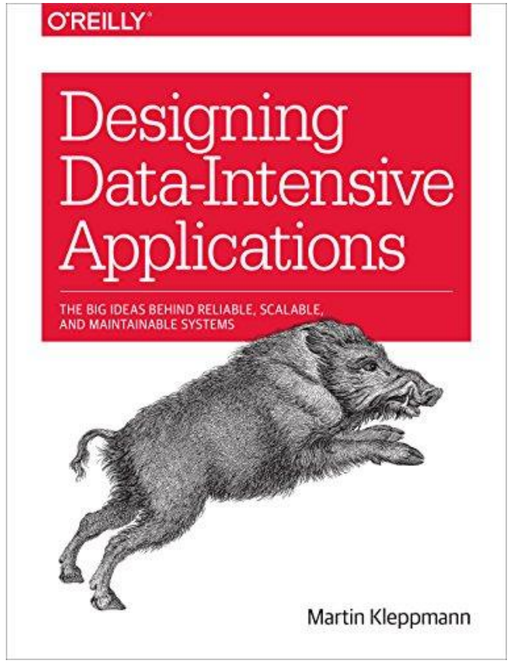
What is this course about?

- Big Data Processing and Programming model
  - Batch processing, stream processing, MapReduce, Hadoop, Spark, Ray, etc.

What is this course about?

- ML Systems
  - ML frameworks, dataflow graph representation of ML, ML parallelism, LLMs, LLM training and serving
  - ~~ML architecture details, learning algorithms/theory, optimizations, NLP~~

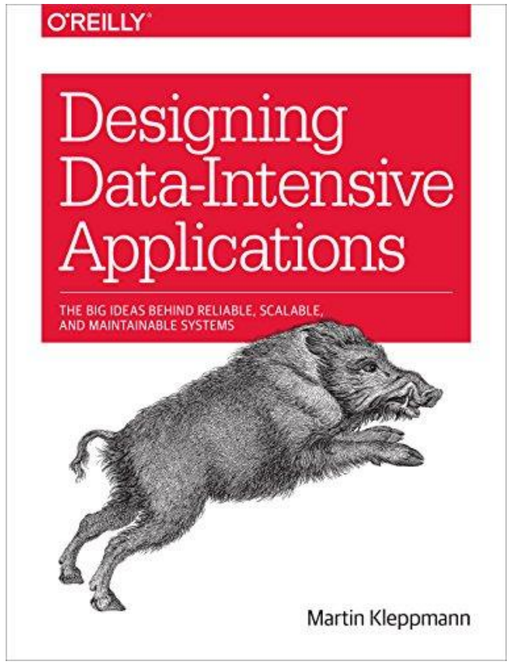
Suggested Textbooks

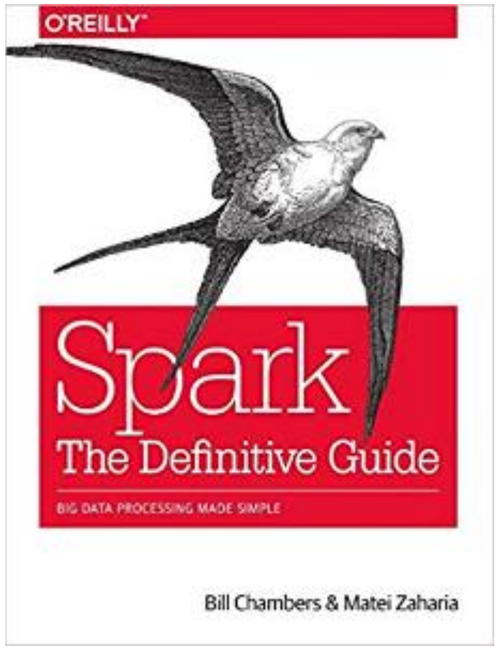


- Chapter 3. Storage and retrieval
- Chapter 4. Encoding and evolution
- Chapter 10. Batch processing
- Chapter 11. Stream processing
- Chapter 12. The future of data systems

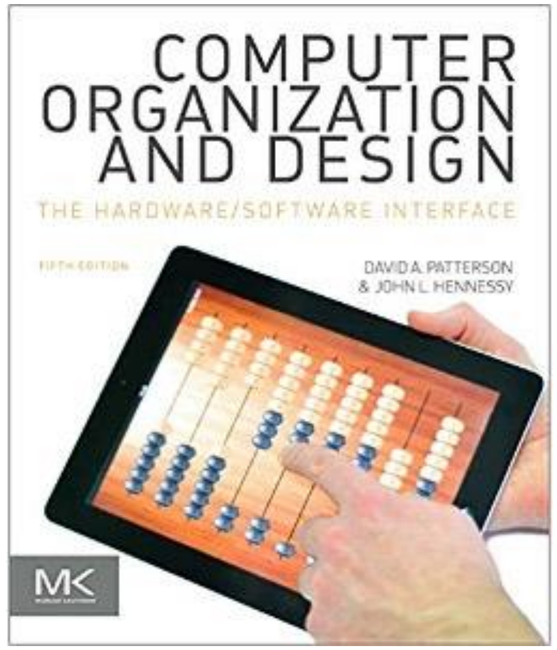
Suggested Textbooks

Computer systems are about carefully layering levels of abstraction.

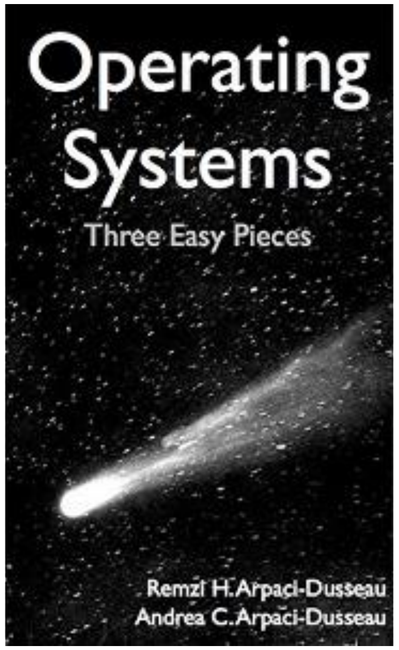




Scalable data flows



Low-level system software



Learning outcomes of this course

- Explain the basic principles of data systems, distributed systems, and data programming model.
- Identify the abstract data access patterns of, and opportunities for parallelism and efficiency gains in data processing at scale.
- Gain hands-on experience in creating end-to-end pipelines for data preparation, feature engineering, and distributed model training.
- Reason critically about practical tradeoffs between accuracy, runtimes, scalability, usability, and total cost.
- **Enter** the current trends of Big data + Big Models

What this course is NOT about

- Not a course on database, relational model, or SQL
  - Take DSC 202 instead (pre-requisite)
- Not a course on how to build scalable data systems
  - Take Distributed Systems, Operating Systems, Cloud Computing, …
- Not a training module for how to use Spark or PyTorch
  - We focus more on principles.
- Not a machine learning course
  - We focus more on system and data
- Not a machine learning system course
  - Take my CSE/DSC 291: deep learning systems in 26 Spring.
  - But could be a warm-up

Delta of this year's offering by Hao

- The pace will be faster: less basics, more advanced stuffs
  - Take DSC 202 or DSC102 instead if you expect more basics (pre-requisite)
- More new stuffs, less classic stuff: ~1/4 will be about new systems developed between 2016 – 2024
  - Data + ML systems: PyTorch, Ray
  - Machine learning parallelism
  - LLM systems
- Homework will be based on Ray and vLLM
- No mid-term, no in-class quiz
- More offline paper readings, scribe notes

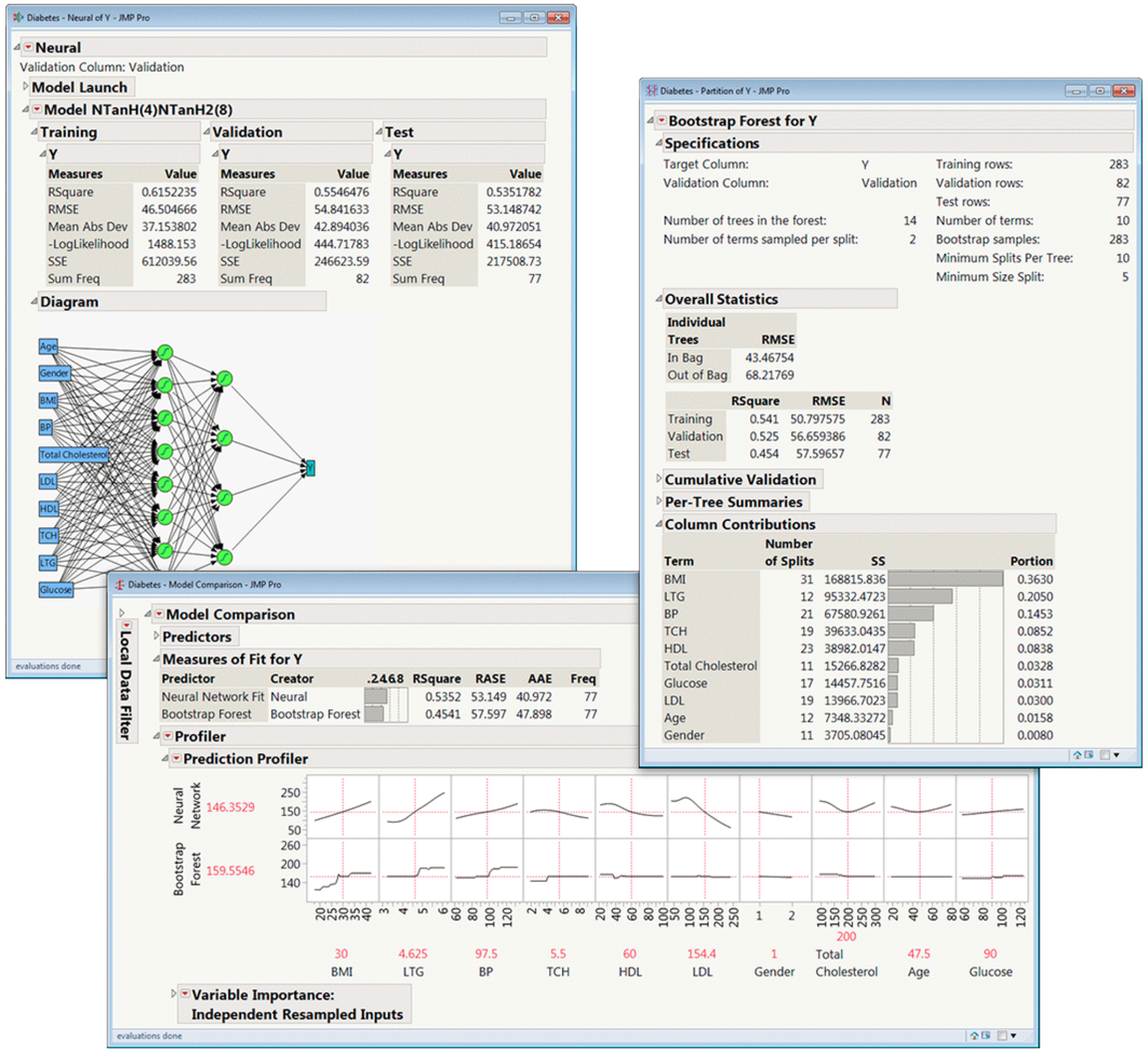
Why bother learning such low-level system-related stuff in Data Science?

I will Provide 2 Arguments

1. Operating Large, distributed systems is an essential skill today
2. The tech world is scaling and accelerating…
3. You might be able to make more money if you know how to deal with distributed systems ☺

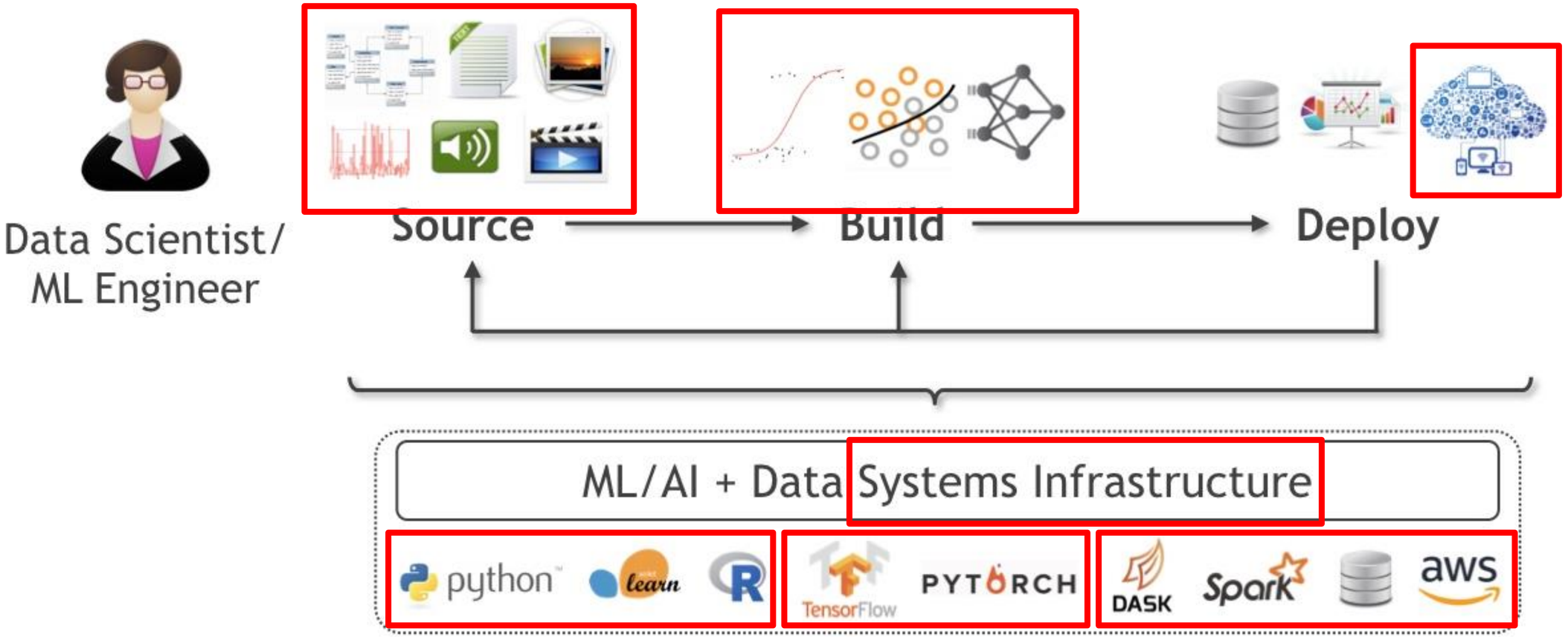
“Statisticians”/“Analysts” 20 years ago

- Methods: Sufficed to learn just math/stats, maybe some SQL
- Types: Mostly tabular (relational), maybe some time series
- Scale: Mostly small (KBs to few GBs)
- Tools: Simple GUIs for both analysis and deployment; maybe an R-like console



https://www.jmp.com/en_au/offers/jmp-pro-for-academic-research.html
https://www.technologymagazine.com/data-and-data-analytics/sas-tops-worldwide-advanced-and-predictive-analytics-market-share

In the era of 2020s:

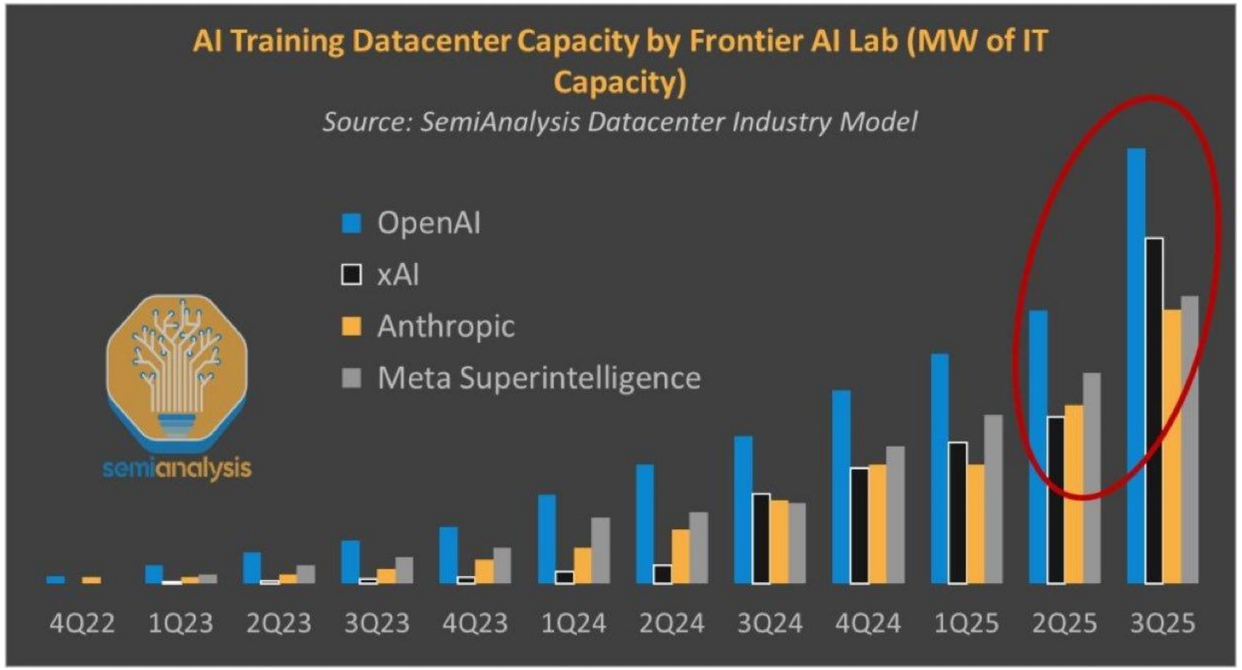


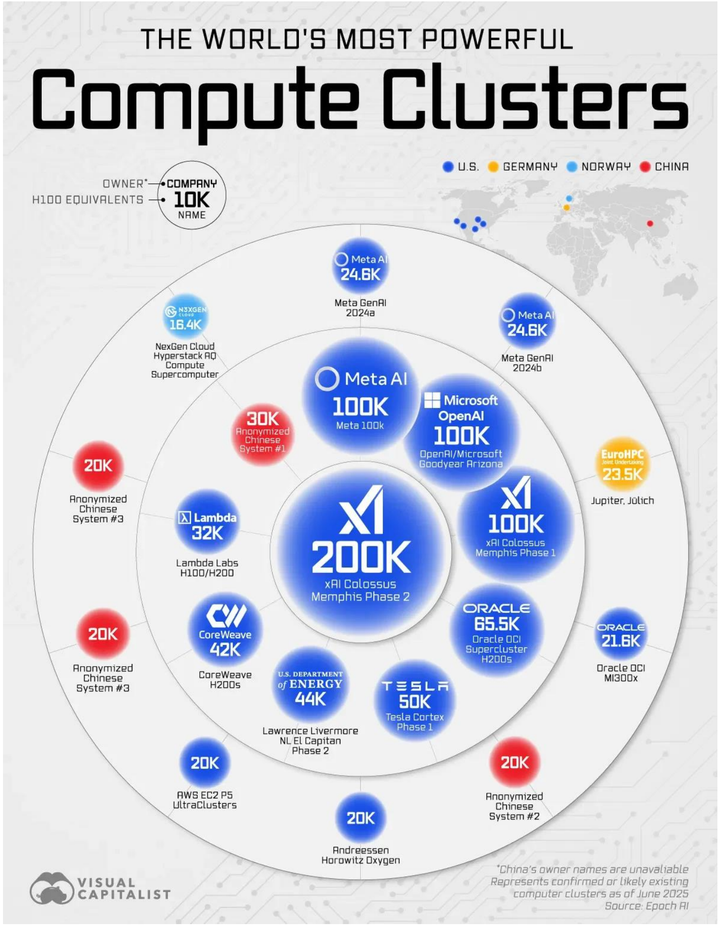
Data acquisition
Data preparation

Feature Engineering
Training & Inference
Model Selection

Serving
Monitoring

The Entire Tech World Now is About Scaling





Q: what skills are most needed to scale on the software side?

glassdoor

statistician | Location

Statistician Salaries United States

Overview | **Salaries** | Interviews | Insights | Career Path

How much does a Statistician make?

Updated Jan 4, 2022

Industry: All industries

Employer Size: All company sizes

Experience: All years of Experience

Very High Confidence

$88,989 /yr

Average Base Pay

2,398 salaries

$61K Low | $89K Average | $131K High

glassdoor

data scientist | Location

Data Scientist Salaries United States

Overview | **Salaries** | Interviews | Insights | Career Path

How much does a Data Scientist make?

Updated Jan 4, 2022

Industry: All industries

Employer Size: All company sizes

Experience: All years of Experience

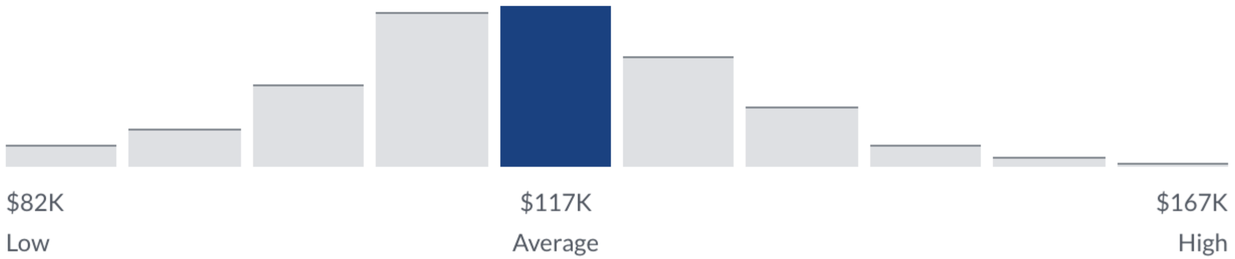
To filter salaries for Data Scientist, Sign In or Register.

Very High Confidence

$117,212 /yr

Average Base Pay

18,354 salaries



— $88,989

= $28,223!

GLASSDOOR

Community Jobs Companies Salaries Search

How much does an AI Engineer make?

Updated Dec 13, 2023

Experience: All years of Experience

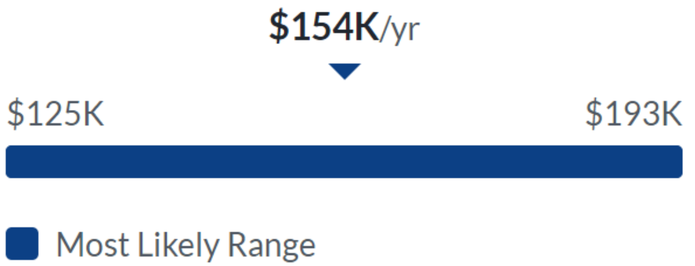
Industry: All industries

Confident

Total Pay Range **$125K - $193K**/yr

Base Pay **$104K - $156K**/yr

Additional Pay **$20K - $38K**/yr



Total Pay Trajectory

For Machine Learning Engineer

- **$152,007 /yr**
  Machine Learning Engineer
- $172,167 /yr
  Senior Machine Learning Engineer
- $165,994 /yr
  Lead Machine Learning Engineer

See Full Career Path >

Download as data table

— $88,989

= $65011!

OpenAI

Work Here? Claim Your Company

OpenAI Software Engineer Salaries

Software Engineer compensation at OpenAI ranges from $570K per year for L4 to $915K per year for L5. The median compensation package totals $925K. View the base salary, stock, and bonus breakdowns for OpenAI's total compensation packages. Last updated: 1/7/2024

Average Compensation By Level

+ Add Comp | Compare Levels

Level Name	Total	Base	Stock (/yr)	Bonus
L3 (Entry Level)	US$ --	US$ --	US$ --	US$ --
L4	US$570K	US$245K	US$325K	US$0
L5	US$914.5K	US$302K	US$612.5K	US$0
L6	US$ --	US$ --	US$ --	US$ --

Another Perspective

The fastest growing companies in SV is either data or AI companies: they operate either big data or big models.

Fastest-growing data companies

snowflake

databricks

Fastest-growing model companies

OpenAI

ANTHROP\C

Questions?

Prerequisites

- DSC 200, 202 (or equivalent).
- Proficiency in Python programming & Unix Terminals
- Network and Operation System basics
- Deep learning basics: pytorch, tensorflow,
- For all other cases, email me with proper justification; a waiver can be considered (I normally approve all students)

Components and Grading

- 3 Programming Assignments: 44% (12% + 16% + 16%)
  - In total 5 late days! Plan your work well ahead.
- No Midterm (cheers!)
- Final Exam (06/14/2023 3pm-6pm): 36%
- Scribe Duties: 8%
- Reading summary: **12%**
- Extra Credit: 5%

Grading Scheme (grade is the better of the two)

Grade	Absolute Cutoff (>=)	Relative Bin (Use strictest)
A+	95	Highest 5%
A	90	Next 10% (5-15)
A-	85	Next 15% (15-30)
B+	80	Next 15% (30-45)
B	75	Next 15% (45-60)
B-	70	Next 15% (60-75)
C+	65	Next 5% (75-80)
C	60	Next 5% (80-85)
C-	55	Next 5% (85-90)
D	50	Next 5% (90-95)
F	< 50	Lowest 5%

Grading Scheme (grade is the better of the two)

Grade	Absolute Cutoff (>=)	Relative Bin (Use strictest)
A+	95	Highest 5%
A	90	Next 10% (5-15)
A-	85	Next 15% (15-30)
B+	80	Next 15% (30-45)
B	75	Next 15% (45-60)
B-	70	Next 15% (60-75)
C+	65	Next 5% (75-80)
C	60	Next 5% (80-85)
C-	55	Next 5% (85-90)
D	50	Next 5% (90-95)
F	< 50	Lowest 5%

Example, 82 and 33%,

Rel: B-; Abs: B+;

Final: B+

The structure of the course

	Topics	
Week 1-2	Foundations of Data Systems	Single Machine: CompOrg, OS, Storage
Week 3-5	Cloud	Cloud: Storage, network, parallelism, etc.
Week 6-8	Big Data	Big Data Processing, dataflow, Programming models
Week 8-10	Machine Learning Systems	MLSys: GPUs, ML libs, ML parallelism, LLM training/serving

🌍 https://hao-ai-lab.github.io/dsc204a-f25/

Programming Assignments

Three PAs

Will be based on Ray

- Good to study and try Ray from today if you have zero experience

Topics: exploring distributed data exploration, processing, and distributed ML

Most of the PAs should be doable using your laptop

- However, if you have trouble (due to hardware issue), please contact TAs

Expectations on the PAs

- Expectations on the PAs:
  - Individual projects; see webpage on academic integrity
- TAs will explain and demo the tools; handle all Q&A
- You are expected to put in the effort to learn the details of the tools' APIs using their documentation on your own!

- In short: if you want to learn something solid, do the PAs
- PAs will be the most challenging part of this course

Scribe Duties

Sign up your scribe duty here:

https://docs.google.com/spreadsheets/d/1NawbzzFapaUqaaldwgHx3CVxjRZyWxeq94F40N-pF-Y/edit?gid=0#gid=0

You should

- Scribe with as many details as possible
- Collaborate with other scribers
- Submit PRs to course website repo
- Reviewed and maybe iterated with the TA

Exams

- No Mid-term
- In-person Final exam (36%)
- All MCQs (select one and all that apply)
- You can bring as many books/cheat sheets/paper you want
- No phone/laptop/Internet/ChatGPT
- Data: TBD

Exams

Hao's lectures will feature some MCQs (that may appear in final exams) every week, so make sure to attend lectures or watch recordings.

TAs will give special recitations for preparing finals to help you navigate

MCQ Example: Who originally developed PyTorch?

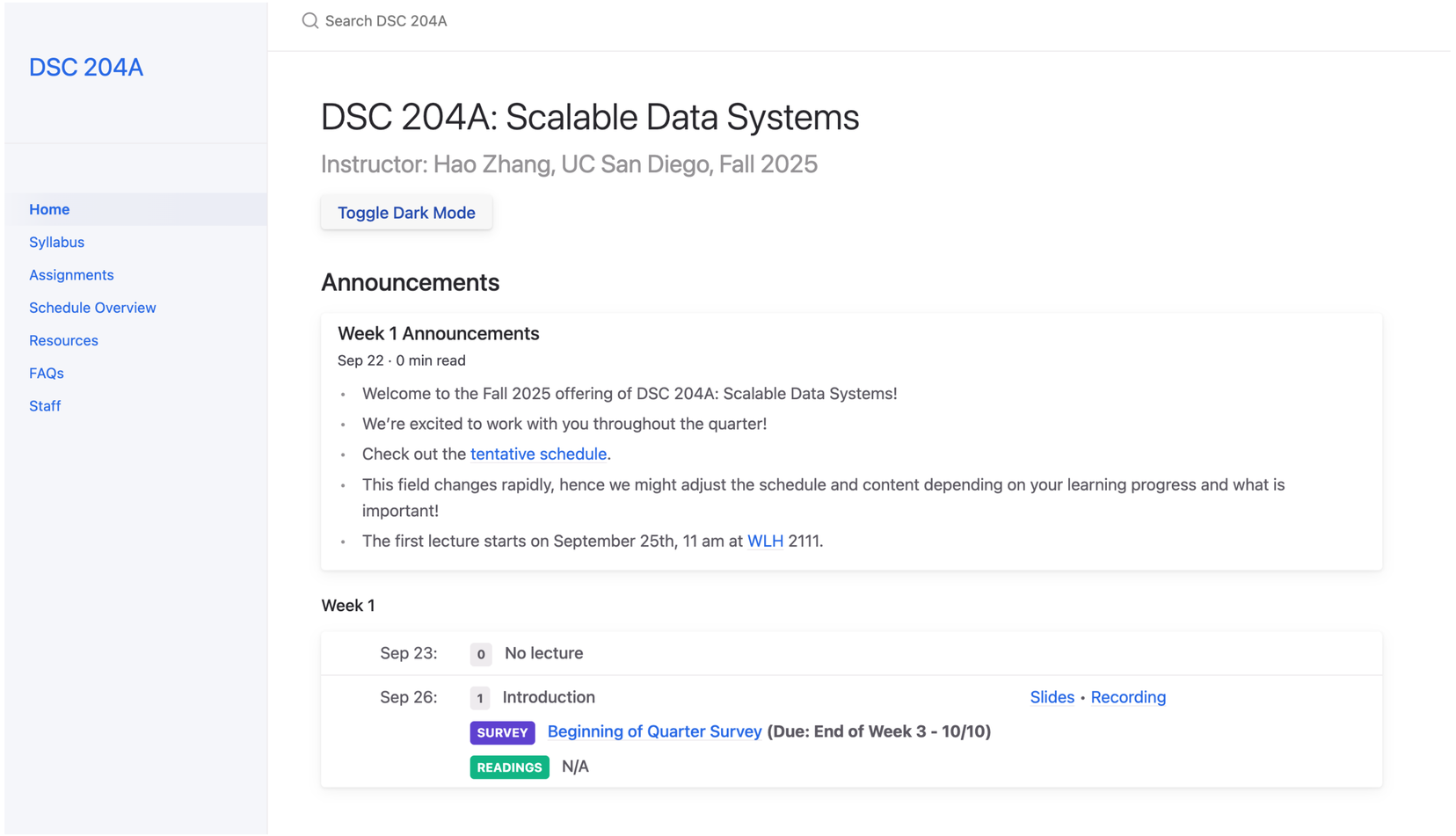
Karma Points

- Participation: lectures / piazza
- Guest lecture: ask hard questions to challenge our guests ☺
- Completing course surveys and evaluation: it helps me, helps TAs and help yourself

Respecting TAs' time

- Use piazza first, seeking helps from your peers
- Students answering questions on Piazza will be rewarded
- Office hours are for getting ideas on how to debug or better approach your homework.
- Write a description! Try to narrow down your problem area as much as possible.
- If you don't have a description, TA can reject your questions.
- Respect TA's working hours.
  - Respond in 24 hours.
  - Members may send msgs at night or on weekends, but only expect to receive a reply on weekday.

Course website



🌍 https://hao-ai-lab.github.io/dsc204a-f25/

General Dos and Do NOTs

- Do:
  - Follow all announcements on Piazza
  - Try to join the lectures/discussions live
  - Participate in discussions in class / on Piazza
  - Raise your hand before speaking
  - View/review podcast videos asynchronously by yourself
  - To contact me/TAs, use piazza first; if you really need to email, use "DSC 204A:" as subject prefix
  - Use LLMs to help your learning

General Dos and Do NOTs

- Do NOT:
  - Harass, intimidate, or intentionally talk over others
  - Violate academic integrity on the PAs, exams, or other components; I (and the school) am very strict on this matter!

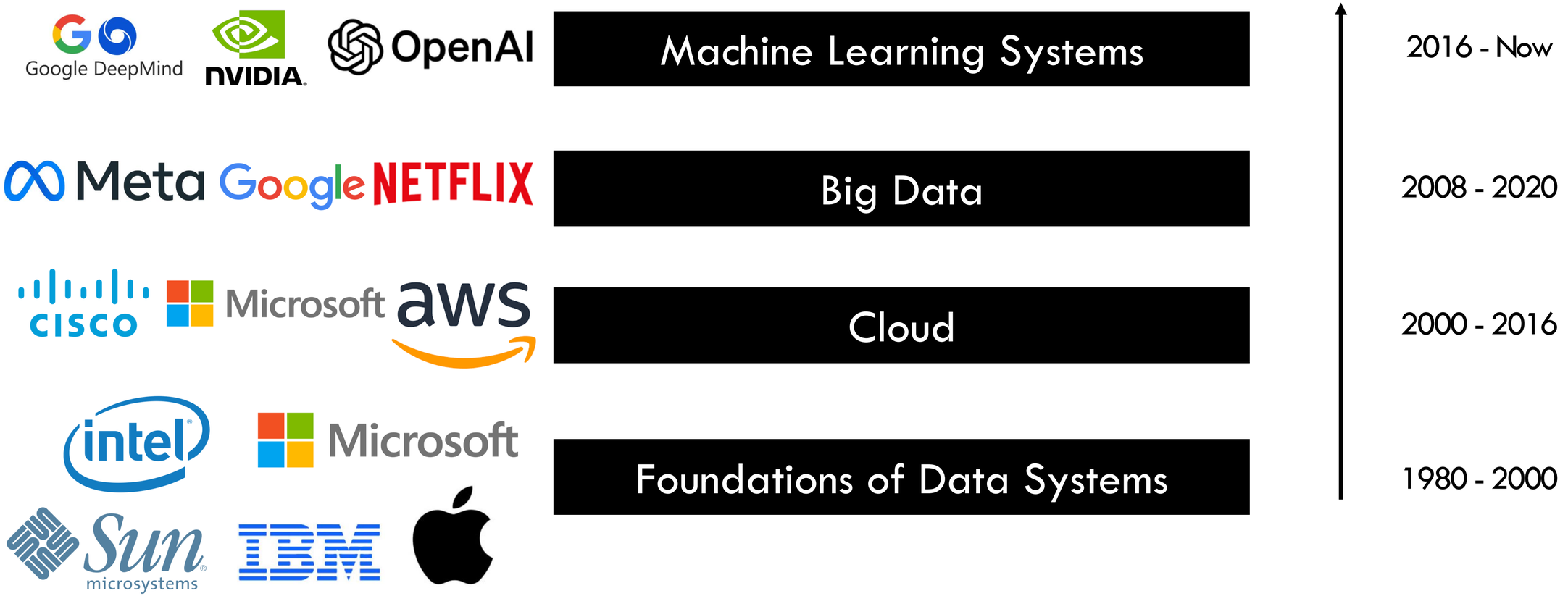
TODOs after Today's lecture

1. Make sure you are enrolled with Piazza, Canvas, Gradesope
2. Check all contents of course website (Schedule, Syllabus, Exam time)
3. Signup your scribe duty
4. Finish Start-of-quarter survey
5. Start the reading of week 2 (which is due on Wed of week 4)

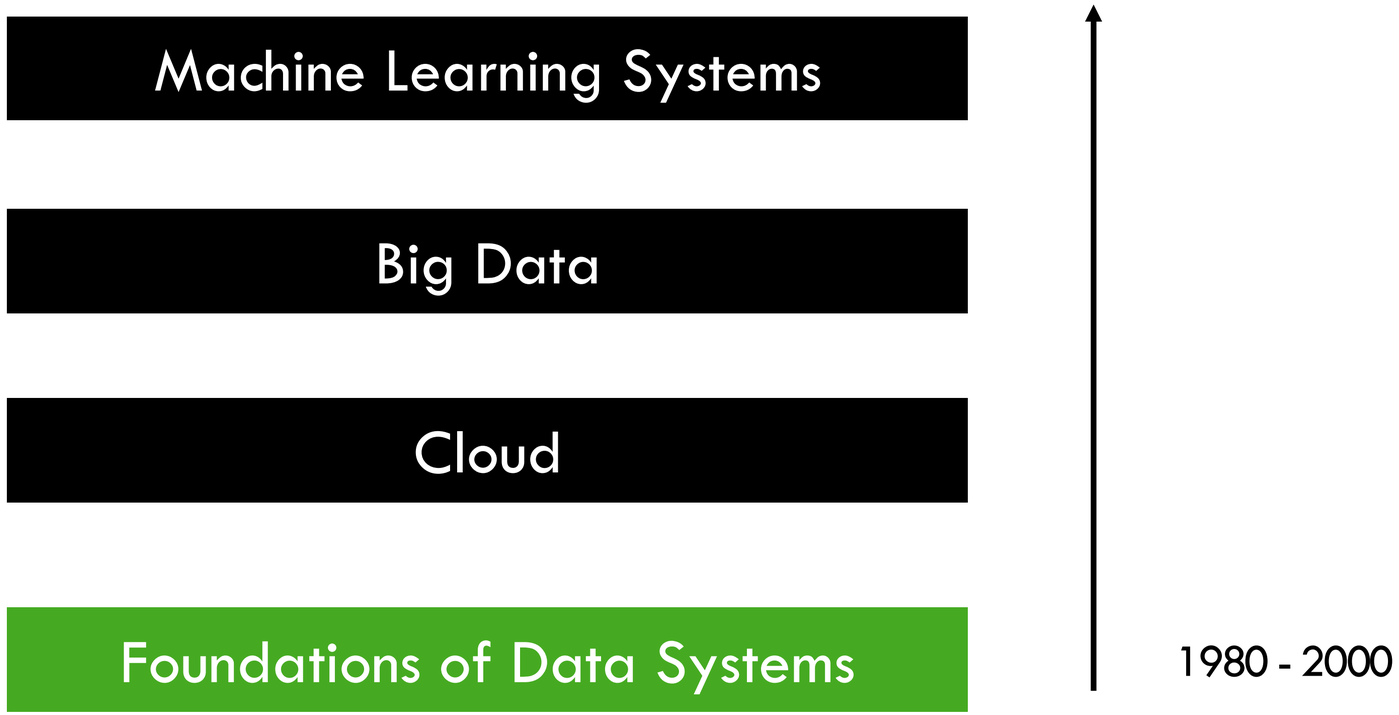
Questions?

Warmup: History of Compute and Data

- ~= History of "which is the most valuable company in tech"



Where We Are


Machine Learning Systems
Big Data
Cloud
Foundations of Data Systems
1980 - 2000

Foundation of Data Systems

- Computer Organization
  - Representation of Data
  - Processors, memory, storages
- Operating System Basics Review
  - Processes: scheduling,
  - File systems
  - Memory management

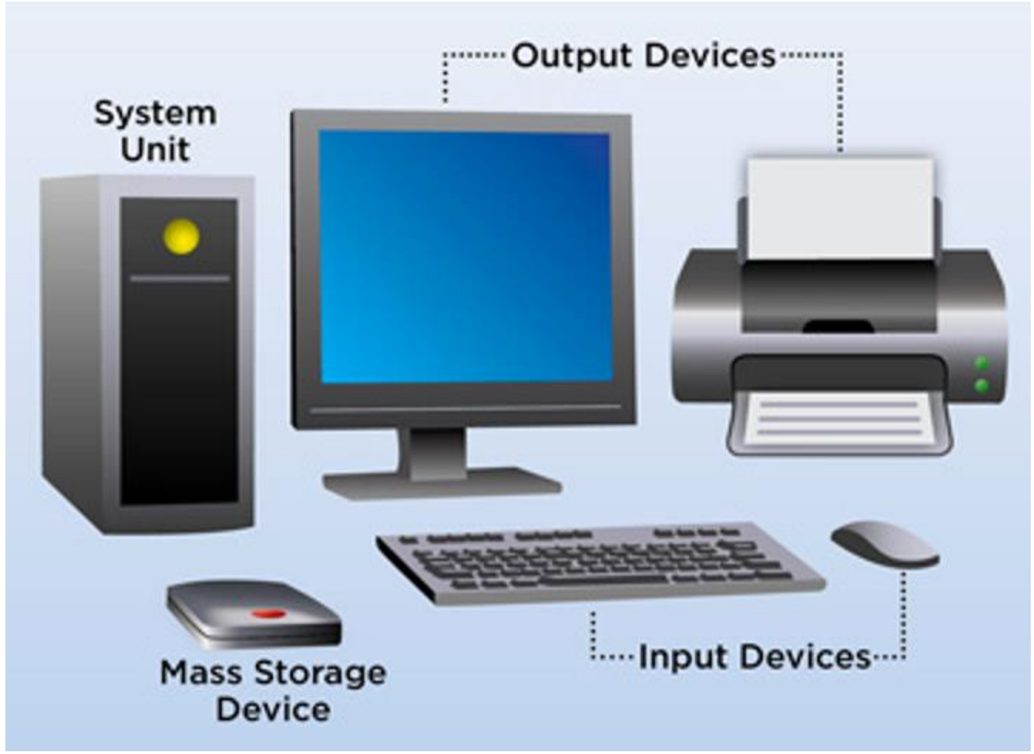
Q: What is a computer?

What is a computer?

Peter Naur

A programmable electronic device that
can store, retrieve, and process digital data.

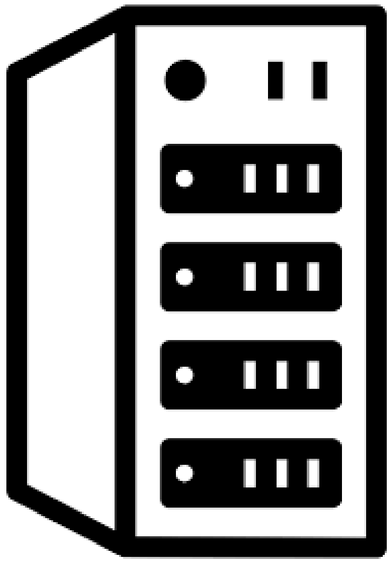
Basics of Computer Organization



- Hardware: The electronic machinery (wires, circuits, transistors, capacitors, devices, etc.)
- Software: Programs (instructions) and data

Ch. 1, 2.1-2.3, 2.12, 4.1, and 5.1-5.5 of CompOrg Book

Basics of Computer Organization



To store and retrieve data, we need:

- Disks
- Memory
- Why we need both? (we'll come back in near future)

To process data:

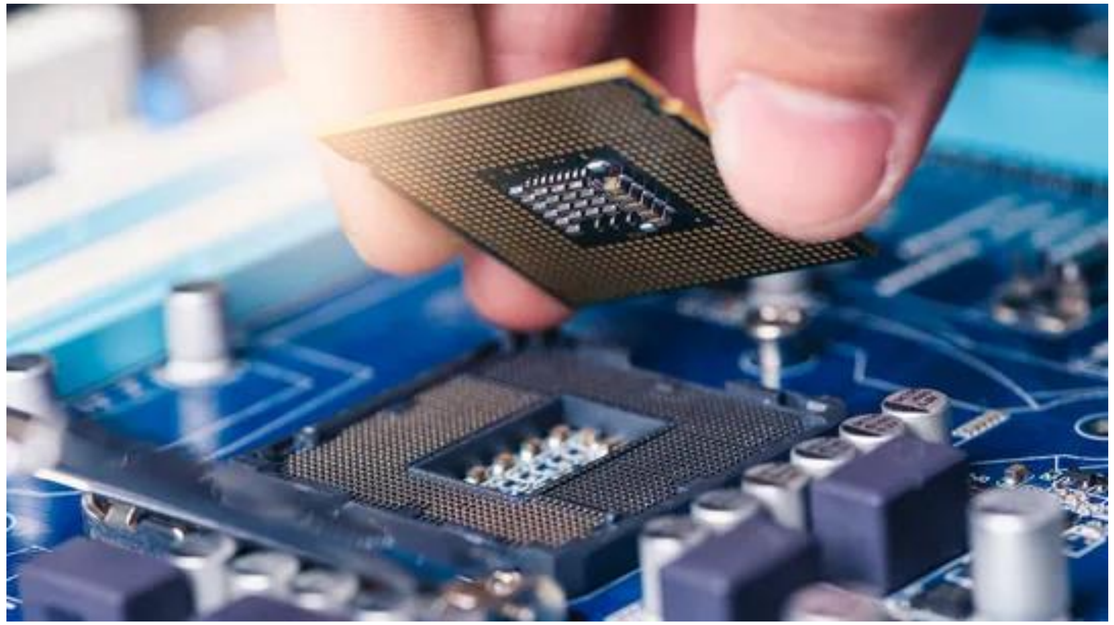
- Processors: CPU and GPU

To retrieve data from remote

- Networks

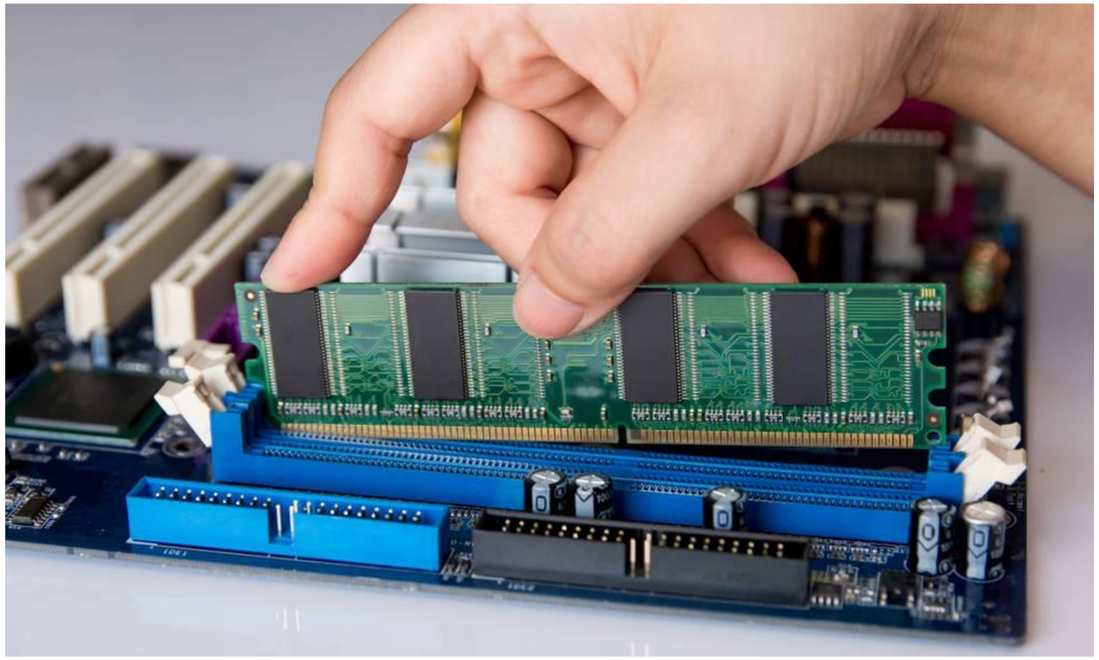
Key Parts of Computer Hardware

- Processor (CPU, GPU, etc.)
  - Hardware to orchestrate and execute instructions to manipulate data as specified by a program



Key Parts of Computer Hardware

- Main Memory (aka Dynamic Random Access Memory)
  - Hardware to store data and programs that allows very fast location/retrieval; byte-level addressing scheme

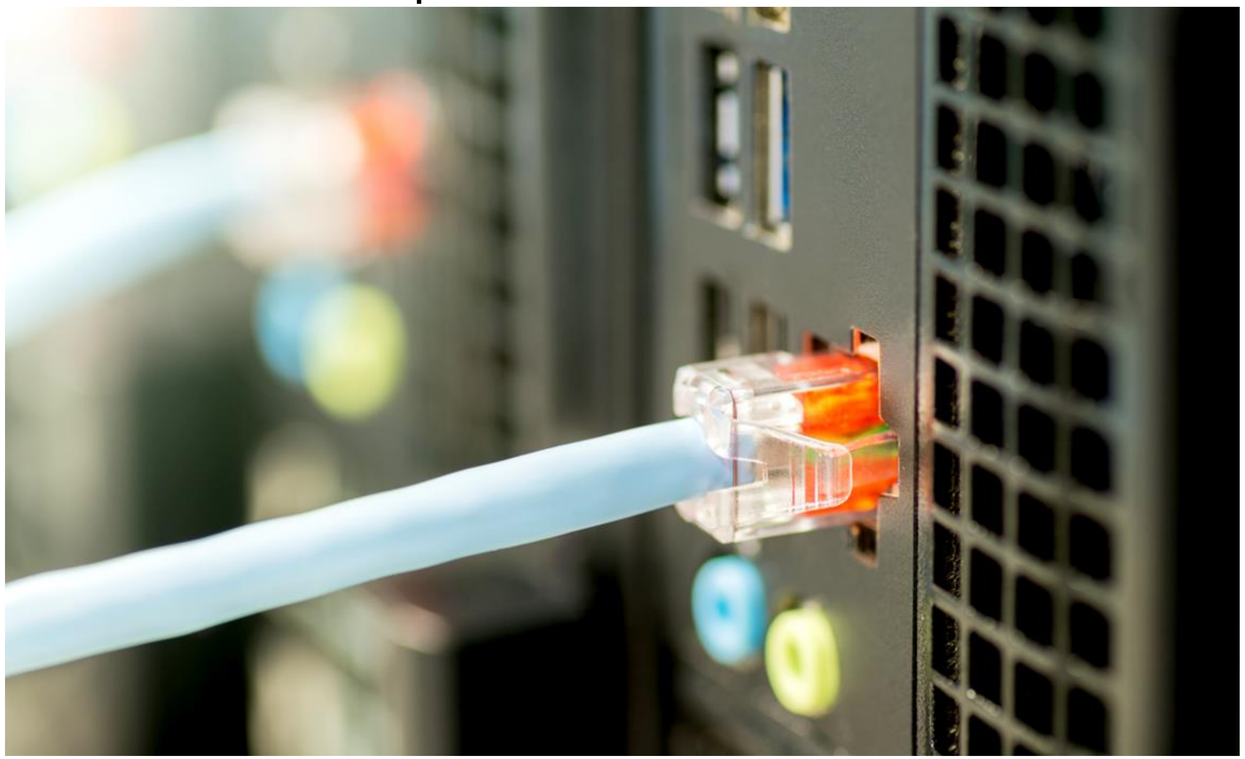


Key Parts of Computer Hardware

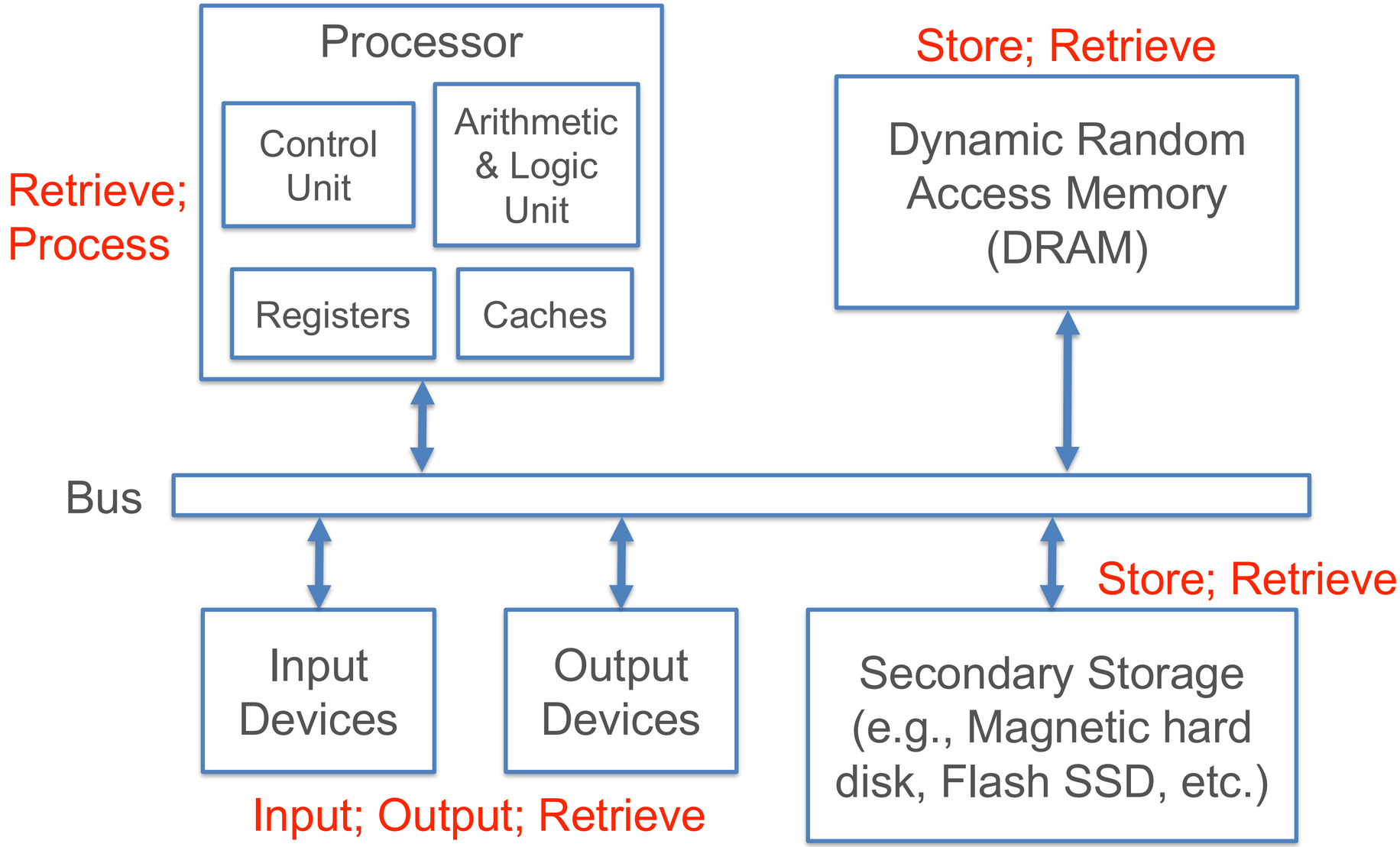
- Disk (aka secondary/persistent storage)
  - Similar to memory but persistent, slower, and higher capacity / cost ratio; various addressing schemes

Key Parts of Computer Hardware

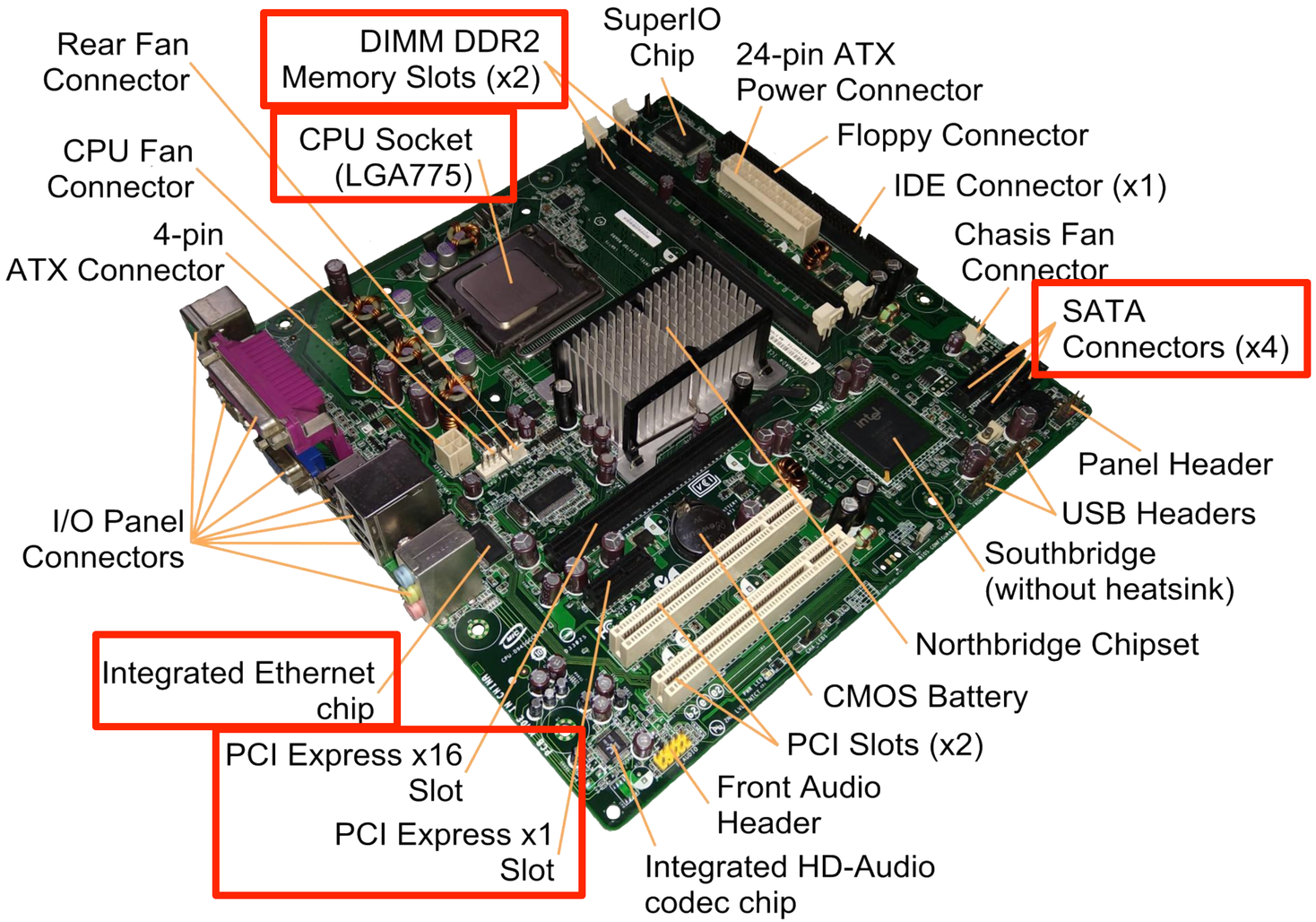
- Network interface controller (NIC)
  - Hardware to send data to / retrieve data over network of interconnected computers/devices



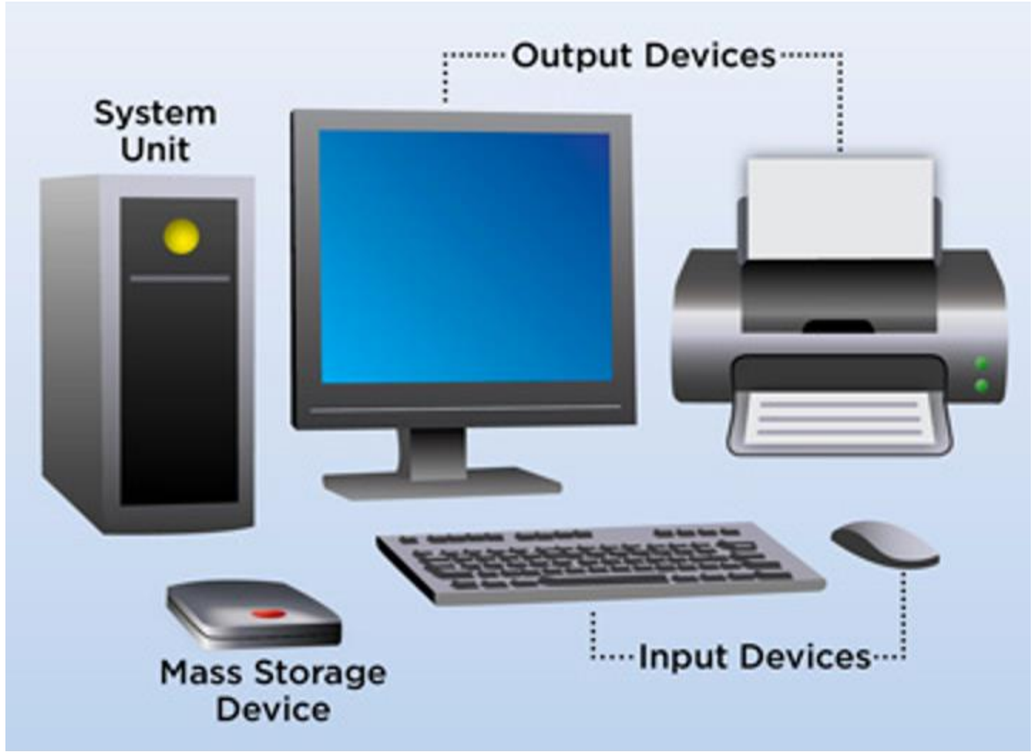
Abstract Computer Parts and Data



In Reality


Rear Fan Connector
DIMM DDR2 Memory Slots (x2)
SuperIO Chip
24-pin ATX Power Connector
CPU Socket (LGA775)
Floppy Connector
CPU Fan Connector
IDE Connector (x1)
4-pin ATX Connector
Chasis Fan Connector
SATA Connectors (x4)
Panel Header
USB Headers
I/O Panel Connectors
Southbridge (without heatsink)
Integrated Ethernet chip
Northbridge Chipset
CMOS Battery
PCI Express x16 Slot
PCI Slots (x2)
Front Audio Header
PCI Express x1 Slot
Integrated HD-Audio codec chip

Parts of a Computer



- Hardware: The electronic machinery (wires, circuits, transistors, capacitors, devices, etc.)
- Software: Programs (instructions) and data

https://www.webopedia.com/TERM/C/computer.html

Key Aspects of Software

- Instruction
  - A command understood by hardware; finite vocabulary for a processor: Instruction Set Architecture (ISA); bridge between hardware and software
- Program (aka code)
  - A collection of instructions for hardware to execute

Key Aspects of Software

- Programming Language (PL)
  - A human-readable formal language to write programs; at a much higher level of abstraction than ISA
- Application Programming Interface (API)
  - A set of functions ("interface") exposed by a program/set of programs for use by humans/other programs
- Data
  - Digital representation of information that is stored, processed, displayed, retrieved, or sent by a program

Main kinds of Software

- Firmware
  - Read-only programs “baked into” a device to offer basic hardware control functionalities
- Operating System (OS)
  - Collection of interrelated programs that work as an intermediary platform/service to enable application software to use hardware more effectively/easily
  - Examples: Linux, Windows, MacOS, etc.

Main kinds of Software

- Application Software
  - A program or a collection of interrelated programs to manipulate data, typically designed for human use
  - Examples: Excel, Chrome, PostgreSQL, etc.

Foundation of Data Systems

- Computer Organization
  - **Representation of Data**
  - Processors, memory, storages
- Operating System Basics
  - Processes: scheduling,
  - File systems
  - Memory management

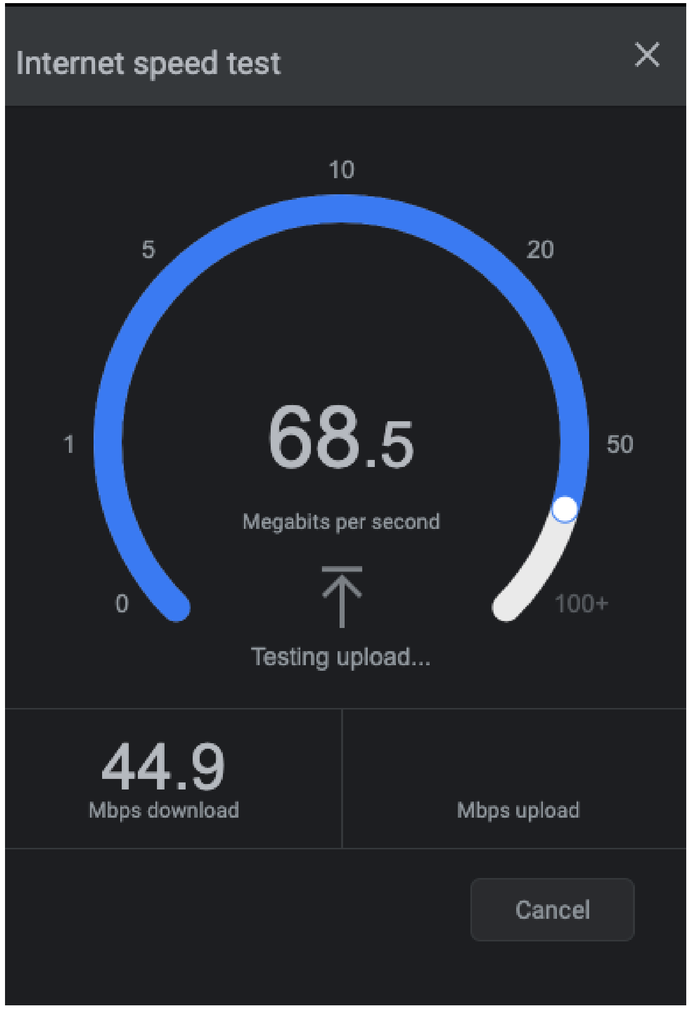
Q: How is data represented in computers?

Digital Representation of Data

- Bits: All digital data are sequences of 0 & 1 (binary digits)
  - high-low/off-on electromagnetism on disk.
- Data type: First layer of abstraction to interpret a bit sequence with a human-understandable category of information; interpretation fixed by the PL
  - Example common datatypes: Boolean, Byte, Integer, “floating point” number (Float), Character, and String
- Data structure: A second layer of abstraction to *organize* multiple instances of same or varied data types as a more complex object with specified properties
  - Examples: Array, Linked list, Tuple, Graph, etc.

Count everything in binary

- Use Base 2 to represent Number
  - 0, 1, 10, 11, 100, 101, …
  - Represent 15213_{10} as 0011 1011 0110 1101_2
  - Represent 1.20_{10} as 1.0011 0011 0011 0011 [0011]…$_2$
- Represent negative numbers as …?
  - (we'll come back to this)



Name	Size	Kind
HB50 cupcakes.JPG	2 MB	JPEG image
Roller Skating.JPG	1.3 MB	JPEG image
50HBJukebox2.jpg	720 KB	JPEG image
Facebook.tiff	399 KB	TIFF image
7_days_to_enrol.png	173 KB	PNG image
JoggingShoes.jpg	71 KB	JPEG image

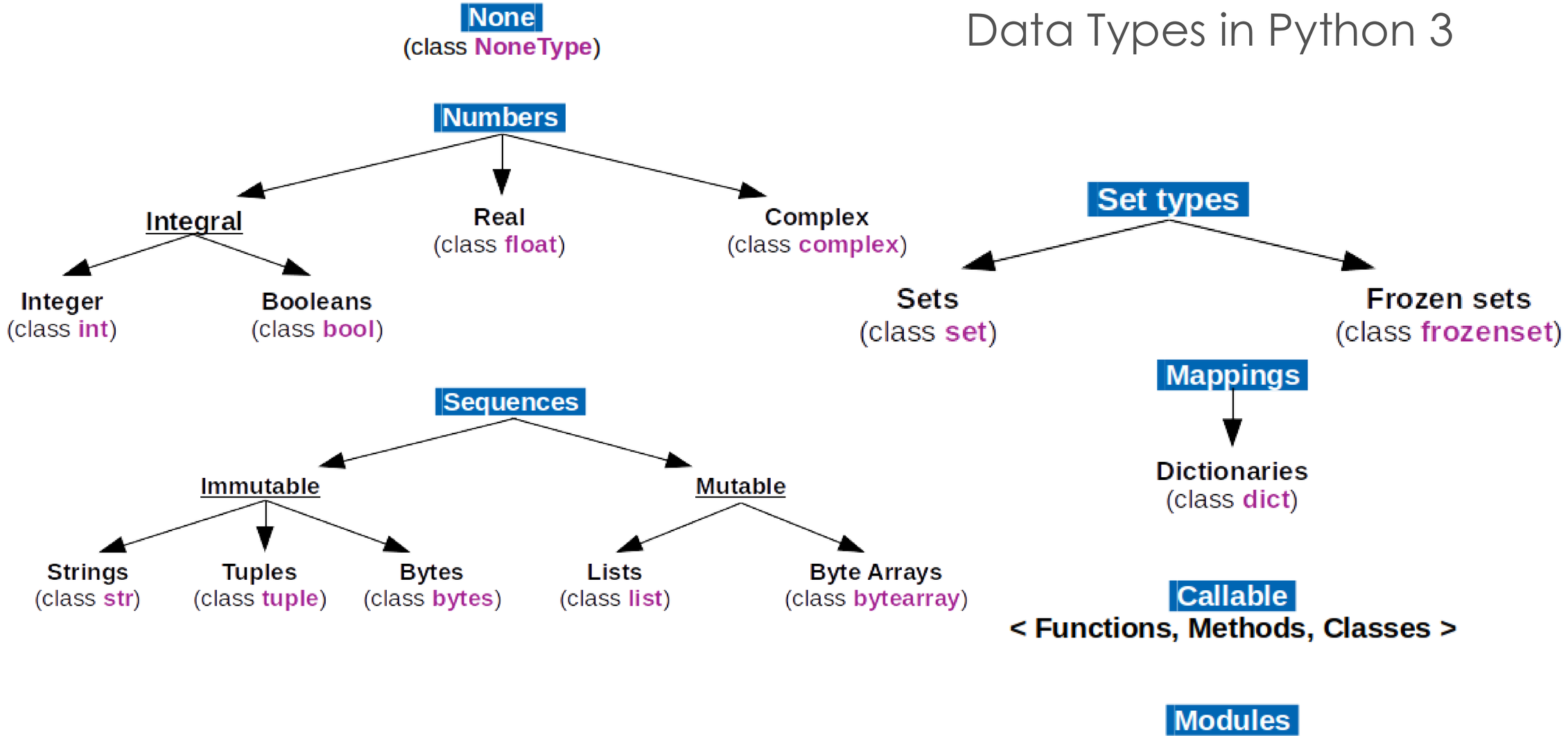
Encoding Byte Values

- Byte = 8 bits
- Why?
  - Historical Development
  - Practicality and Standardization
- A Byte (B; 8 bits) is typically the basic unit of data types
  - CPU can't address anything smaller than a byte.

Bytes -> Data types: bool, int, float, string, …

- The *size* and *interpretation* of a data type depends on PL
- Boolean:
  - Examples in data sci.: Y/N or T/F responses
  - Just 1 bit needed but actual size is almost always 1B, i.e., 7 bits are wasted!
- Integer:
  - Examples in data science: #friends, age, #likes
  - Typically 4 bytes; many variants (short, unsigned, etc.)
  - Java *int* can represent -2^{31} to $(2^{31} - 1)$; *C unsigned int* can represent 0 to $(2^{32} - 1)$;

Digital Representation of Data



Digital Representation of Data

Q: *How many unique data items can be represented by 3 bytes?*

- Given k bits, we can represent 2^k unique data items
- 3 bytes = 24 bits => 2^{24} items, i.e., 16,777,216 items
- Common approximation: 2^{10} (i.e., 1024) ~ 10^3 (i.e., 1000); recall kibibyte (KiB = 1024 B) vs kilobyte (KB = 1000 B) and so on

Q: *How many bits are needed to distinguish 97 data items?*

- For k unique items, invert the exponent to get $\log_2(k)$
- But #bits is an integer! So, we only need $\lceil \log_2(k) \rceil$
- So, we only need the next higher power of 2
- 97 ->128 = 2^7; so, 7 bits

Digital Representation of Data

Q: *How to convert from decimal to binary representation?*

- Given decimal n, if power of 2 (say, 2^k), put 1 at bit position k; if k=0, stop; else pad with trailing 0s till position 0
- If n is not power of 2, identify the power of 2 just below n (say, 2^k); #bits is then k; put 1 at position k
- Reset n as n - 2^k; return to Steps 1-2
- Fill remaining positions in between with 0s

	7	6	5	4	3	2	1	0	Position/Exponent of 2
Decimal	128	64	32	16	8	4	2	1	Power of 2
5_{10}						1	0	1	
47_{10}			1	0	1	1	1	1	
163_{10}	1	0	1	0	0	0	1	1	
16_{10}				1	0	0	0	0	

***Q:** Binary to decimal?*

Digital Representation of Data

```
void show_squares()
{
    int x;
    for (x = 5; x <= 5000000; x*=10)
            printf("x = %d x^2 = %d\n", x, x*x);
}
```

x = 5 x^2 = 25
x = 50 x^2 = 2500
x = 500 x^2 = 250000
x = 5000 x^2 = 25000000
x = 50000 x^2 = -1794967296
x = 500000 x^2 = 891896832
x = 5000000 x^2 = -1004630016

Two-complement: Simple Example

	-16	8	4	2	1	
10 =	0	1	0	1	0	8+2 = 10

	-16	8	4	2	1	
-10 =	1	0	1	1	0	-16+4+2 = -10

Encoding Integers

Unsigned

$$B2U(X) = \sum_{i=0}^{w-1} x_i \cdot 2^i$$

Two's Complement

$$B2T(X) = -x_{w-1} \cdot 2^{w-1} + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

Sign Bit

```
short int x =  15213;
short int y = -15213;
```

Two-complement Encoding Example (Cont.)

```
x =      15213: 00111011 01101101
y =     -15213: 11000100 10010011
```

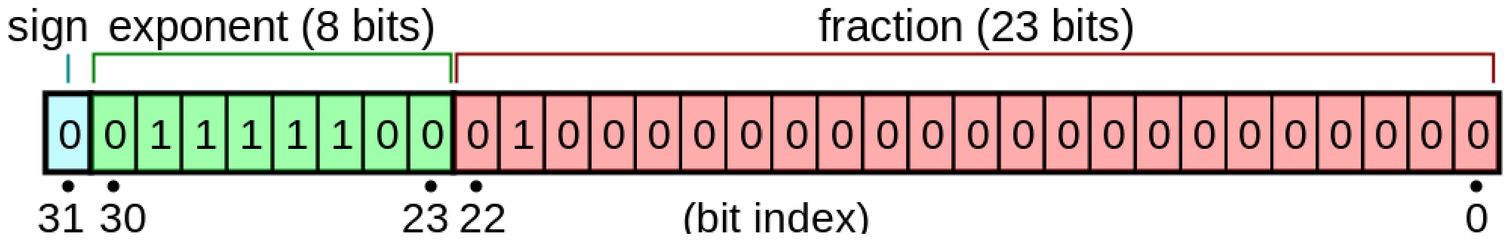
Weight	15213		-15213	
1	1	1	1	1
2	0	0	1	2
4	1	4	0	0
8	1	8	0	0
16	0	0	1	16
32	1	32	0	0
64	1	64	0	0
128	0	0	1	128
256	1	256	0	0
512	1	512	0	0
1024	0	0	1	1024
2048	1	2048	0	0
4096	1	4096	0	0
8192	1	8192	0	0
16384	0	0	1	16384
-32768	0	0	1	-32768
Sum		**15213**		**-15213**

Digital Representation of Data

- **Float**:
  - Examples in data sci.: salary, scores, model weights
  - IEEE-754 single-precision format is 4B long; double-precision format is 8B long
  - Java and C *float* is single; Python *float* is double!

Digital Representation of Data

- Float:
  - Standard IEEE format for single (aka binary32):



$$(-1)^{sign} \times 2^{exponent-127} \times (1 + \sum_{i=1}^{23} b_{23-i} 2^{-i})$$

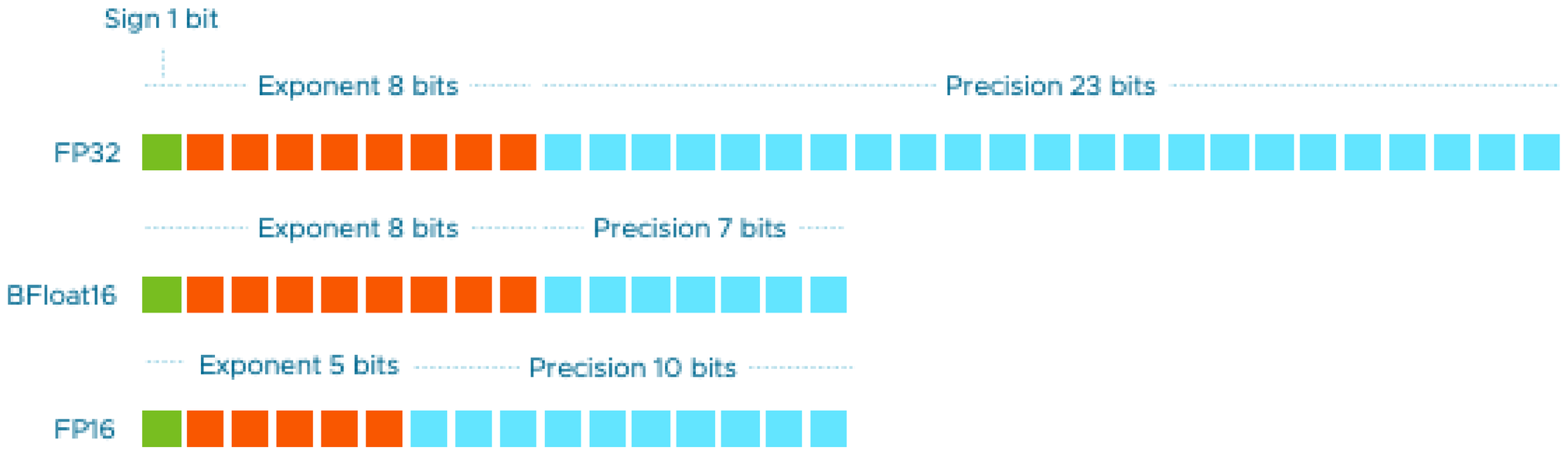
$$(-1)^0 \times 2^{124-127} \times (1 + 1 \cdot 2^{-2}) = (1/8) \times (1 + (1/4)) = 0.15625$$

Digital Representation of Data

- More float standards: double-precision (float64; 8B) and half-precision (float16; 2B); different #bits for exponent, fraction
- Float16 is now common for ***deep learning*** parameters:
  - Native support in PyTorch, TensorFlow, etc.; APIs also exist for weight quantization/rounding post training

https://docs.nvidia.com/deeplearning/sdk/mixed-precision-training/index.html

New magical float standards

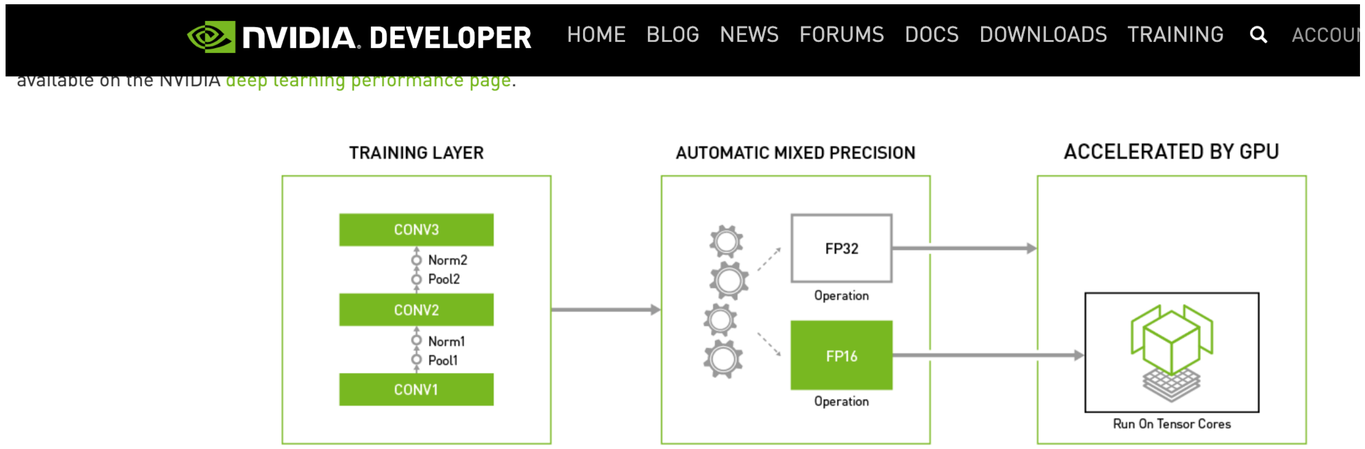


What's the difference between bf16 and fp16?

Fp16 vs. Fp32

NVIDIA Deep Learning SDK support mixed-precision training; 2-3x speedup with similar accuracy!

Form Factor	H100 SXM
FP64	34 teraFLOPS
FP64 Tensor Core	67 teraFLOPS
FP32	67 teraFLOPS
TF32 Tensor Core	989 teraFLOPS[2]
BFLOAT16 Tensor Core	1,979 teraFLOPS[2]
FP16 Tensor Core	1,979 teraFLOPS[2]
FP8 Tensor Core	3,958 teraFLOPS[2]



Using Automatic Mixed Precision for Major Deep Learning Frameworks

Digital Representation of Data

- Representing **Character (char)** and **String**:
  - Letters, numerals, punctuations, etc.
  - A string is typically just a variable-sized array of char
  - C *char* is 1B; Java char is 2B; Python does not have a char type (use *str* or *bytes*)
  - American Standard Code for Information Interchange (*ASCII*) for encoding characters; initially 7-bit; later extended to 8-bit
    - Examples: 'A' is 61, 'a' is 97, '@' is 64, '!' is 33, etc.
  - *Unicode UTF-8* is now common, subsumes ASCII; 4B for ~1.1 million "code points" incl. many other language scripts, math symbols, 🤗, etc. 💻

Digital Representation of Data

- All digital objects are *collections* of basic data types (bytes, integers, floats, and characters)
  - SQL dates/timestamp: string (w/ known format)
  - ML feature vector: *array* of floats (w/ known length)
  - Neural network weights: *set* of multi-dimensional *arrays* (matrices or tensors) of floats (w/ known dimensions)
  - Graph: an *abstract data type* (ADT) with *set* of vertices (say, integers) and *set* of edges (*pair* of integers)
  - Program in PL, SQL query: string (w/ grammar)
  - Other data structures or digital objects?

Practice Qs (review next class)

Q1: How many space do I need to store GPT-3 ?

Q2: What does **exponent** and **fraction** control in float point representation?

Q3: What is the difference between BF16 and FP16?