<u>9/27: Quantum Computing</u>

- Hardware improving, i.e. Google in Santa Barbara
    - Google, Intel, IBM: largest players
        - Google nearby, 72 qubits in Santa Barbara computer
            - qubit: <alpha, beta>
            - **Quantum computing computes the qubits of complex numbers by harnessing the laws of quantum mechanics**
                - 1 qubit: 2 complex numbers, 2 qubits: 4 complex numbers, 3 qubits: 8 complex numbers, etc.
                    - Now computes at 2^72 complex numbers, outputs 72 bits
                    - Has 2 states: |0> and |1>
                    - |psi> = alpha |0> + beta |1>
                    - |alpha|^2 + |beta|^2 = 1 ← complex probabilities
                    - Computation based on estimation of probability of state based on energy levels
        - Chips stored in fridges @ 0.1K
            - Can only handle ~100 steps of computation before heat failure
            - Improving at a doubly exponential rate every ~10 days - Neven's law, quantum computing gets twice as good (reliability and size)
    - Reliability at ~99%: not good, test by running many times (Google: 10 million times) and checking results against each other, caused by particles interfering, cannot control electrons, small size makes it harder to correct errors, need 100x more qubits to correct errors
    - Double slit experiment: Photons don't reach through 2 slits, but may through 1 slit
        - Quantum mechanics uses complex numbers to create complex probabilities
- Algorithms
    - Breaking encryptions that classic computers cannot w/ Shor's algorithm, which can factor very large #s: creation of new encryptions that quantum computers also cannot
    - Deutsch-Jozsa algorithm, takes 2 qubits, applies a matrix, outputs a single bit of information, makes a conclusion from that bit
- Interdisciplinary field
    - Chemistry, computer science, physics, electrical engineering, math
- Misc.
    - John von Neumann, Richard Feynman, Einstein
    - Merging of quantum computing and machine learning
    - Superposition: Idea that a computer can be in more than one state at the same time, both |1> and |0>, Ex) can put many different inputs and compute all of it at the same time to one output bitstring

- Entanglement: If we know something about one qubit, then we know something about every other qubit (gloves), doesn't break relativity: no transfer of information
- Motivations to simulate quantum chemistry, solve optimization, requires large enough problems to overwhelm classic computers (~3000 perfect qubits required for quantum advantage), factor large numbers, solve NP problems
- Languages of pyQuil, Q# (Microsoft)
- 788 qualified people in quantum computing, needed in the industry

10/4: Network Algorithmics

- Algorithmics to speed abstractions in all systems
    - Ex. 1) Virtual Memory
        - Abstraction: Illusion of infinite memory
        - Algorithmics: Paging algorithms
    - Ex. 2) Relational Databases
        - Abstraction: Operations on logical tables
        - Algorithmics: Query planning
- How did algorithmics start?
    - Context: Web exploding, traffic doubling, address doubling in 1990s
    - Problem: TCP (connected queues) and IP (datagram) slow, as were routers and servers
    - Network Algorithmics: techniques to restore speed of abstractions to that of fiber
    - This lecture: trying to give you a sense of what this field is/was about
- Outline
    - Confluences as a way of thinking of interdisciplinary research
    - Network Algorithmics viewed as confluences from lookups to logging
- What is a confluence?
    - Where 2 rivers meet → Main Stream + Impacting Stream = New Stream
        - Inflection Point = something that causes this merge
        - Milieu Change = something that changes the values of the old idea to create a completely new idea
        - Ex. 1) Realistic Painting + Psychology = Impressionism
            - Inflection Point = Photography, less point in painting when pictures give exact copy
            - Milieu Point = Ideas to canvas, visualization of psychology
        - Ex. 2) Algorithms + Probability = Randomized Algorithms
            - Inflection Point = Crypto, Miller-Rabin algorithm for testing primes
            - Milieu Point = Always to sometimes
        - More CS examples:
            - Algorithms + Networks = Distributed Algorithms
                - Inflection Point = Popularity of Internet
                - Milieu Change = Asynchrony, partial failure
            - Economics + Computer Science = Computational Economics
                - Inflection Point = Internet Auctions
                - Milieu Point = Large scale, small latency
- Why confluences?
    - Separate trends from fads
    - Provide a research theme
    - Balance desire for beauty and impact
    - Suggest a new field
- Interdisciplinary work does not equal confluence
    - Ex) Network + Learning Theory =/= Network Learning

- Network Algorithmics
  - Definition: a set of algorithms to make network services (servers/ routers) faster
  - Streams: Computational architecture + algorithms + networking
  - Inflection Points: Cheap clusters (servers), explosion of Web, IPv6
  - Ex. 1) RDMA (servers)
    - Servers getting slow as they used remote disks, READs from disk to memory slow
    - Inspiration from computer architecture: DMA from disk to memory, bypass CPU
    - Do same thing across network (RDMA)
      - Networking + Architecture = Algorithmics
        - Inflection Point: Cheap clusters
        - Milieu Change: Machine bus → Network bus
        - DMA → RDMA
  - From RDMA to Fast Servers: Other ideas
    - Inflection Point: Internet holding up
      - Fast Buffers: Avoid packet copies without changing protocol → 0 copy interfaces
  - Ex. 2) Prefix Lookup (Routers)
    - Facebook feed is a set of Internet messages carrying a 32 bit IP address
    - Routers send your message onward by looking at a forwarding table
    - To avoid a table of a billion addresses in each router, routers store prefixes
    - Reduces router memory, but routers now have to solve longest prefix match for every packet billions of times/second, IPv6 even harder with 128 bits, how to cope?
      - Prefix Lookup Details
        - Prefix is the start bits common to lots of computers in an organization like UCLA
        - Core routers don't store all addresses in UCLA, just a few prefixes
        - One UCLA prefix: 128.97.0.0/16
        - Write each number between dots as binary, take first 16 bits → something idk :(
        - Networking + Algorithms = Network Algorithmics
          - Inflection Point = Traffic, IPv6
          - Milieu Change = Milliseconds to microseconds
  - Fast Routers common by 2000s
    - Cisco Cat6K, GSR, Juniper M40
    - All problems (switching, lookups, ACLs, scheduling) had reasonable hardware
    - New algorithmics problems after 2000 in measurement and security and flexible routers

10/11: Natural Language Processing

- Information extraction - unstructured text to database entries
  - Processing social media, archives, medical records, reviews, stocks, medicine, etc.
- Machine translation - Google Translate, Facebook, etc.
- Sentiment/Opinion Analysis - used in elections/product reviews for polling
- Examples of applications: pseudocode → code, summarization, generating writing
- Why is NLP hard?
  - Lexical ambiguity: “I saw her duck with a telescope”, ambiguity comes from words
  - Syntactic ambiguity: “San Jose cops kill man with knife”, “Include your children when baking cookies”, “Local High School Dropouts Cut in Half”, “Hospitals are Sued by 7 Foot Doctors”, etc., comes from multiple interpretations of prepositional phrases
  - Semantic ambiguity: 2 ways of reading the sentence
  - Anaphoric ambiguity: Pronoun reference ambiguity
  - Language is not static
  - Models that work with some datasets can vary greatly on others
  - Word Embedding can be trained from the news to be sexist: surgeon → nurse, architect → interior designer, beer → cocktail, professor → associate professor
  - Human bias in structured prediction models: more likely to identify someone cooking as a woman → model biased, only 16% of cases identify males cooking
- Machine Learning - Input → Label space, identify the association between labels
- Prerequisites - Linear algebra, calculus, probability, programming skills, algorithm/data structure, Intro to ML, NLP

10/18: Computer Architecture for Machine Learning

- What is architecture?
    - You can have a computer without having an architecture
- Computers pre-1964
    - Each computer was new
        - Implemented machine (has mass) → hardware
        - Instructions for hardware (no mass) → software
    - Software lagged Hardware
        - Each new machine design was different
        - Software needed to be rewritten in assembly/machine language state
    - Unimaginable today
        - Going forward: need to separate HW interface from implementation
- What should be in an architecture?
    - Ingredients:
        - Memory: a place to put values (state, variables, etc.)
        - Instructions: moves from one state to the next
        - Program: a set of instructions (let's put it in memory)
        - Execution model: When do we execute each instruction?
    - Von Neumann Execution:
        - Most common model today
        - Instructions are executed sequentially, defined by a program counter
- How does hardware use the ISA?
    - Steps to executing any instruction:
        - Fetch: Grab instruction from memory
        - Decode: Interpret instruction
        - Execute: Perform computation
        - Writeback: Update state
- Summary:
    - ISA abstracts hardware to make software simpler
    - Von Neumann ISA
        - Used by every CPU you own
        - Program: Set of instructions
        - Execution model: Sequential execution
- What is deep learning?
    - Machine Learning:
        - Problem: want to write a function, but it's too complicated
        - Goal: Use data to train a function
        - Approach 1: Define the form of a function that is easy to train
        - Approach 2: Gently nudge the parameters towards providing the correct answer (backpropagation)
    - Linear algebra (matrix multiplication)
- Neural net:
    - Layer: one of the volumes above

    - Neuron: one element of the volume, a linear function of inputs/other neurons
    - Synapse: a “connection” between neurons in different layers, the slope of the line
- Google TPU Processor
    - Developed around 2014
    - Made for speech recognition
    - Has a matrix multiplication unit to prevent overwhelming calculations
- Paradox
    - General purpose processors stagnating (+20% in 5 yrs)
    - Machine learning processors thriving
    - 2 reasons:
        - No longer technology free ride
        - Scaling general purpose architectures is much harder than scaling linear algebra architectures
- Conclusion
    - Instruction abstractions make it hard to build scale general purpose architectures
    - Traditional Von Neumann ISA and general purpose computers hard to improve
    - Further co-design across devices, architectures, and algorithms

10/25: Challenges in Deep Learning

- Prediction error by 2015 outperforms humans in ideal conditions
- Input → decision rule → output
- Human learning: observation → learning → decision rule, biological process
- Machine learning: training data → learning → decision rule, must be much more structured
- Decision function (model)
    - Machine learning: find the best function to map input to output according to training data
    - Assuming we use a linear function: sum of wi times xi
        - The goal is to find the best w, linear support vector machine
- Given training data, we pick the one with the smallest loss on training data
- How good is training data?
    - loss(f(x), y) is 0 if f(x) = y and 1 if not
    - Training error on all the data is the sum of the losses
    - Machine learning is optimization, find the minimum training error
- More data = better performance/better estimation/more representative
- How to find the best linear function?
    - Optimization
        - Design a way to find the minimizer of a function
        - Make it efficient on large datasets
            - ImageNet: millions of data points
            - Google search: trillions of data points
- Software
    - LIBLINEAR
        - Used in all linear classification tasks (e.g., scikit-learn)
- Machine learning
    - Find the function to map X (images, video, sentence, document, etc.) to Y (Label, real number, image, sentence, etc.)
- Regression
    - Y is real number instead of 0/1
    - Ex: stock price prediction, predict height by geometric figures
    - Switch to square loss: loss (f(x), y) = (y - f(x))^2
- Google Search
    - Query → Machine Learning Model → website 1, website 2, … website n
- Neural network
    - Defines a nonlinear function, can represent more complex mapping, needs more data than linear model
    - Too simple = underfit, not able to improve upon adding data
    - Too complex = overfit, can fit all data, but may not perform well in future
- History
    - Pre-1998: main stream of machine learning research
    - 1998-2012: dominated by linear/kernel SVM

    - 2012-now: back due to Big Data and Efficient Hardware, leads to exciting performance on many tasks
- Learning neural network
    - Find the weights W = [W1, W2, … , WL] to minimize loss
        - Same optimization as linear
    - Multiple layers lead to high computational cost
- More complex output space
    - Object detection: output bounding box + label
    - Machine translation (or any sentence generation)
- Challenges/Research Topics
    - Model design
    - Efficiency (training and testing)
    - Robustness
    - Explainability
    - …
- Model Design
    - Design network architecture for each task
        - NLP, computer vision, graph mining, etc.
    - Automatic model design
        - Neural architecture search
- Training efficiency
    - Training by SGD
        - Sample data point → check current prediction → update
    - 3 days for training ImageNet on 1 GPU
    - Speed up using multiple GPUs/TPUs
        - Efficient algorithm for large batch size
        - Learning rate scheduling
    - ImageNet can be solved in minutes
        - BERT training in 76 minutes
- Model size and inference speed
    - Limited storage on mobile devices
    - Inference needs to be made in milliseconds
- Models are sensitive to small perturbation
    - Although models outperform humans, a small perturbation to a natural image can make accuracy 0
        - Current solution: consider perturbation in the training phase
    - Similar problem: domain shift, Ex) self-driving cars w/ weather and environment, medical data w/ different race/sensors
- Model interpretability

11/1: Computer Graphics and Vision

- Graphics and Vision: A Unified View
    - This major field is about computers and images
    - Computer Graphics (CG)
        - Computational models → images and videos
        - Forward mathematical problem
        - Synthesis
    - Computer Vision (CV)
        - Images and videos → computational models
        - Inverse mathematical problem
        - Analysis
- History of Computer Graphics and Computer Vision
    - 2 PhD projects at MIT in the early 1960s
        - Ivan E. Sutherland, 1963
            - "Sketchpad, a man-machine graphical communication system"
        - Lawrence G. Roberts, 1963
            - "Machine perception of 3-dimensional solids"
        - CG and CV have developed as independent fields
        - In recent years, CG and CV are synergizing
        - Cross-fertilization with other fields:
            - Physics, biology, cognitive science, artificial intelligence, art, …
- What is an image/video?
    - Array of pixels (1+ numbers)
    - A video is a time sequence of images
    - How they are formed:
        - Objects in the world (static or dynamic)
        - Illumination (light sources)
        - Imaging device (eye, camera)
    - We want to synthesize and analyze images and videos by computer
- Why? Images and movies are everywhere!
    - Entertainment: Motion pictures and games, virtual worlds, industrial design, scientific and medical visualization, human-computer interaction, fine arts, robotics, automotive, visual surveillance/biometrics, industrial inspection, medical imaging, remote sensing, image and video retrieval, etc.
- Computer Graphics
    - The art and science of creating imagery by computer
    - 3 main research themes
        - Modeling
            - How do we model (mathematically represent) objects?
            - How do we construct models of specific objects?
        - Animation
            - How do we represent the motions of objects?
            - How do we give animators control of this motion?

        - Rendering
            - How do we simulate the real-world behavior of light?
            - How do we simulate the formation of images?
- Standard Display Devices
    - CRT (Cathode Ray Tube), Plasma, LCD (Light Crystal Display), OLED (Organic Light-Emitting Diode)
- Exotic Display Devices
    - Immersive, Head-Mounted, Autostereoscopic, Holographic, Volumetric
- Computer Aided Design
    - Precision modeling, engineering visualization/simulation
- Elements of CG
    - The graphics pipeline: Modeling → Animation → Rendering
- Modeling
    - Primitives
        - 3D points
        - 3D lines and curves
        - Surfaces (BREPs): polygons, patches,
        - Volumetric representations
        - image-based representations
    - Attributes
        - Color, texture maps
        - Lighting properties
    - Geometric transformations
    - Representing objects geometrically on a computer
    - Altering geometric models
- Rendering
    - Visibility
    - Simulating light propagation
        - Reflection, Absorption, Scattering, Emission, Interference
    - Draw visible surfaces onto display
- Animation
    - Keyframe animation
    - Motion capture
    - Procedural animation
        - Physics-based animation
        - Behavioral animation
- Subsurface Scattering
    - Translucency and varied levels of light penetration can be created using subsurface scattering effects
- Many Open Research Problems
    - How to model/render/animate a complex scene?
- Computer Vision
    - Related fields

        - Image processing
        - Pattern recognition
        - Visual perception
    - Image understanding
        - True IU seems to involve a great deal of human intelligence
        - Automated systems are still far from human performance
        - Some good solutions in constrained special cases (IC manufacturing, circuit boards, etc.)
    - Inverse problems are generally tougher to solve
- Vision Research Themes
    - Edge and region extraction
    - Image segmentation
    - Visual reconstruction: from images to surface
    - Shape from X
        - Contours, shading, stereo, motion
    - Object tracking
    - Object recognition (including faces)
    - Event and activity recognition
- Core Knowledge Needed to Specialize in CG/CV
    - Mathematics
        - Especially geometry, linear algebra, applied math, numerical methods
    - Programming and software development
        - Especially C/C++, OpenGL, Javascript
    - Creativity and an appreciation of Art + Science + Engineering
    - Core computer graphics subjects
        - Rendering - synthesizing images from mathematical representations
        - Modeling - geometry-based, physics-based, biology-based
        - Animation - kinematics, dynamics, motion control
        - Interactive techniques - human-computer interaction, GUIs, games, etc.

11/8: Data Mining and Social Science

- Existing Work
  - 1-D ideal point model
  - High-dimensional ideal point model
  - Issue-adjusted ideal point model
- Motivation
  - Voters have different positions on different topics
  - Traditional matrix factorization method cannot give the meanings for each dimension
  - Topics of bills can influence politician's voting, and the voting behavior can better guide the topics of bills as well
- Text Part
  - We model the probability of each word in each document as a mixture of categorical distributions, as in PLSA and LDA
- Combining Two Parts Together
  - The final objective function is a linear combination of the 2 average log-likelihood functions over the word links and voting links
  - We also add an l_2 regularization term to Q and X to reduce overfitting
- Challenges
  - How to model behavior links using ideology?
    - A link generative model with ideology as latent variable
    - Another social choice problem: to follow or not?
  - How to combine different types of behaviors?
    - Heterogeneous network mining
- Ideology Detection via Single Type of Links
  - Assumptions
    - People tend to follow, mention, or retweet others who share similar ideology score
    - One's true idology score is different from one's public image ideology score
- Objective Function
  - Maximize the likelihood of observed positive links and sample negative links
  - Maximize the weighted sum of the likelihood function for each link type
- Data Collection
  - Step 1: Seed users collections
  - Step 2: Expand from seeds
  - Step 3: Filtering
- Intuitions to the Solution
  - With different stances, words are used differently even for the same issue
  - User interaction will contribute to learning the right contrasting viewpoints for each issue
- Opinion Propagation Model
  - Moving directions determine the position

    - Moving directions are influenced by social neighbors
- Intervention
    - How to alleviate opinion divergence?
        - Increase number of friends (diversity)
        - Strong opinion
- Summary
    - Online behaviors are ubiquitous
        - Content, interaction with other entities, dynamic
    - User stance prediction solution
        - Content rich information network mining
    - Three case studies in this line
        - Issue-based ideology detection for Congress people
        - Ideology detection for Twitter users via heterogeneous network mining
        - The co-evolution model that can explain and predict the dynamics

11/15: Robotic Space Exploration

- JPL aims to:
  - Understand how the Earth works and how its changing
  - Pave the way for space exploration
  - Understand the solar system
  - Understand how human life developed
  - Is there life beyond Earth
  - Understand the diversity of planetary systems in our galaxy
  - Understand how the universe began and how it's evolving
- Mars Exploration Program:
  - Some satellites launched in 2001
  - First robots on the surface in 2011
  - InSight launched in 2018
  - Cruise → Land → Retrieve → Launch → Rendezvous → Return
- In-Space Applications:
  - Climbing robot, telescope assembly
- Cassini: Exploration of Saturn with multiple satellites
- Small satellites are a growing component of space exploration
- MarCO - first interplanetary CubeSat mission

11/22: Distributed Web and Mobile Systems

- What is a Distributed System?
    - Multiple interconnected computers that cooperate to provide some service
    - Google, Yelp, Amazon, BofA, Instagram, Dropbox, etc.
- Data Centers
    - Spread services and data storage/processing across hundreds of thousands of machines
- Other Distributed Systems Examples
    - Internet and data center network, DNS, Operating System for multi-core processors, Multiplayer games
- Why make a system distributed?
    - Handle geographic separation: Facebook, Google, and Amazon users span the planet
    - Build reliable systems with unreliable components
    - Aggregate systems for higher capacity
        - CPU cycles, disks, memory, network bandwidth
        - Cost grows non-linearly
    - Customize computers for specific tasks (microservices)
        - Cache server, load balancer, ads, speech-to-text
- Challenge 1: Partial Failures
    - A failure in one part of the system can result in failures in many other parts
    - Design distributed systems assuming failures will occur
- Challenge 2: Ambiguous Failures
    - If a server doesn't reply, how can you distinguish between:
        - Server failure
        - Network failure
        - Neither: both too slow
    - Cannot make assumptions about incurred delays
- Challenge 3: Concurrency
    - Why not partition users across machines?
    - Goal: ensure consistency of distributed state in the face of concurrent operations
    - Challenge: cross-server synchronization must operate with slow/unreliable machines and network
- Other Challenges
    - Performance at scale
        - Ex) Amazon redesigns software services for every order of magnitude change in scale
    - Comprehensive testing
        - Impossible to test/reproduce all possible scenarios
        - Even if enumerated, cannot test at scale
    - Security
        - Ex) malicious modification of network traffic
- Distributed Systems Objectives

    - Goal: unify several machines to provide shared service
    - Enable concurrent use of state/resources and hide failures
    - Automate heavy lifting so developers don't have to
- Contrast with Operating Systems
    - Distributed Systems: give one app the illusion that it is running on a single machine (though it is using many)
    - Operating Systems: give every application the illusion that it is running on its own machine (though they share a machine)
- Usefulness in Industry
    - As developers of distributed systems: be able to reason about uncertainty
    - As users of distributed systems: be able to identify guarantees and properties offered by a system
        - When will it work well? When will it break?
- MapReduce Scenario
    - Genomics/biotech company that generates personalized DNA report (for ancestry analyses
    - Collect genome data from roughly one million users
        - 125 MB of data per user
    - Goal: analyze data to identify genes that show susceptibility to Parkinson's disease
- Other MapReduce scenarios
    - Ranking web pages for search
        - Requires analyzing 100 billion web pages
    - Selecting ads for different users
        - Requires analyzing clicks/actions for 1 billion users
    - Real-time system diagnostics
        - Requires analysis of data transactions and system logs, 7TB/month
- Lots of Data!
    - Petabytes and soon exabytes of data
    - Cannot simply use one server
        - Impossible to store data on one server
        - Processing will take forever on one server
    - Need distributed storage and processing
- Desirable Properties for Distributed Processing
    - Scalable
        - Performance grows with the number of machines
    - Fault tolerant
        - Can make progress despite machine failures
    - Simple
        - Low programmer expertise required
    - Widely applicable and flexible
        - Impose minimal restrictions on the type of processing feasible
- Distributed Data Processing

- Strawman solution
    - Partition data across many servers
    - Each server processes local data
    - Issue: may need to analyze entire dataset due to inter-data dependencies
      - PageRank: depends on ranking of all pages that link to it
      - 23andme: need data from all users with a certain gene to evaluate susceptibility to a disease
- MapReduce
  - Distributed data processing paradigm
    - Introduced by Google in 2004
    - Popularized by open-source Hadoop framework
  - MapReduce represents two things
    - Programming interface for data processing jobs: Map and Reduce functions
    - Distributed execution framework: Scalable and fault tolerant
- Fault Tolerance via Master Node
  - Master responsibilities for fault tolerance
    - Assign map/reduce jobs to workers
    - Keep track of the status of tasks
    - Keep track of machine failures (rerun tasks as necessary)
- Handling Master Failures
  - If the master node fails, can we start a new master and resume execution
    - No, you lose all state about task execution status (which tasks are finished, currently running (and by whom), and need to be run)
    - Need to start from scratch → wasted resources and time
  - Better solution: replicate master state to tolerate failures
- Synchronizing Replicas
  - How can we ensure that replicas are in sync?
    - Apply updates in same order at all replicas
  - Model every replica as a state machine
    - Given identical initial states, applying updates in same order results in same final state
  - Replicated state machine: run copies of same state machine across many servers
    - Challenge: doing this efficiently
- Video
  - Youtube:
    - Goal: high quality, no rebuffering, smoothness
    - Challenges: variable/unpredictable network, conflicting QoE goals
    - Solution: dynamically pick chunk qualities based on QoE metric and network prediction
  - Skype:
    - Similar to streaming, but new, real-time video, so no buffer

        - Solution: need better predictions of network, or tighter coupling between network and video encoder
    - Analytics:
        - Run ML pipelines (e.g., object recognition) to answer queries on live video
        - Challenges: upload bandwidth, edge compute, cluster scheduling
- Mobile Systems
    - Phones <<<< Servers in terms of storage/computation → run intense tasks on servers
        - Need to send state/context to server for execution
    - Challenge: latency to servers affects applications
    - Solution: hide latency with other parts of application, subsample