Future Directions

Lecture 26, Computer Networks (198:552)
A 10,000-foot recap of the class

• Foundational: layering, congestion control, scheduling
• Inter and intra-domain networks
• Software-defined networking
• Data center networking
• Co-designing networks with applications
The times they are a-changin’…

• Technological constraints
  • Demise of Moore’s law, memory wall

• Tools
  • Machine learning

• Industrial trends
  • Consolidation, disaggregation, net neutrality

• Unconventional networks
  • Blockchain networks, edge networks

Today: some examples of recent work along these directions
Machine learning

• Algorithmic techniques to learn **functions** from inputs to outputs

• Chief distinction among approaches: kinds of functions that can be learned

• Examples: mapping images to labels, predicting next match score from historical scores, etc.
Neural networks

- Neuron (perceptron): tries to approximate a function $f^*$
  - ... internally using a (parameterized) function $f$
  - i.e.: $y = f(x; p)$
  - Learning: Choose parameter $p$ to “best approximate” $f^*$

- Example representational functions:
  - Linear function on inputs
  - Sigmoid over linear combination of inputs

- **Layers**: can compose these functions with each other
  - Structure of composition forms the “network” of neural networks
Expressiveness of neural networks

• Universal approximation theorem
  • Any “reasonable” function can be represented to within arbitrary error over all inputs

• Doesn’t mean any function can be learned, however

• The size of the network isn’t bounded
Applications in networking

- "ML for systems": good to apply ML when
  - Optimal solutions are hard to compute
  - State of the art mostly constitutes heuristics
  - Existing solutions use a “model” of the system

- Some examples…
Pensieve: Video bit-rate adaptation

[Neural adaptive video streaming with Pensieve
Hongzi Mao et al, SIGCOMM’17]
Pensieve: Video bit-rate adaptation

- State:
  - Past chunk throughput
  - Past chunk download time
  - Next chunk sizes (at varying bitrates)
  - Current buffer size
  - Number of chunks left
  - Last chunk bit-rate

- Use raw observation signals; no network “model”

- Function learned: map state to requested bit-rate
Remy: Congestion control

• **State:**
  - Interarrival time between ACKs (EWMA)
  - Time between TCP sender timestamps (EWMA)
  - Ratio between most recent RTT and minimum RTT

• **Action:**
  - A multiple to the current congestion window
  - Increment or decrement to the congestion window
  - A lower bound on time between successive packet transmissions

• Simulate multiple senders & choosing best rule given the state
  - Subdivide the state space used in the most used rule
Remy: Congestion control

Figure 4: Results for each of the schemes over a 15 Mbps dumbbell topology with \( n = 8 \) senders, each alternating between flows of exponentially-distributed byte length (mean 100 kilobytes) and exponentially-distributed off time (mean 0.5 s). Medians and 1-\( \sigma \) ellipses are shown. The blue line represents the efficient frontier, which here is defined entirely by the RemyCCs.

[Computer-generated congestion control, Winstein and Balakrishnan, SIGCOMM’13]
Building Systems for ML

• Data center architectures for ML
  • How to train really large networks?
  • Can we make apps using ML really fast?
  • What programmatic frameworks can make developing ML apps really easy?

• “Edge” video analytics
Living on the edge…

• “IoTs”
  • mobile phones
  • Cameras: street intersections, shopfront, car dashcam
  • Sensors

• Constraint: significantly less powerful than compute clusters
  • Power, compute, bandwidth

• But want to implement sensor data processing
  • Ex: Video analytics using neural networks
Partitioning edge video analytics
Flexible auto-encoding of features

“Neural networks meet physical networks”, Chinchali et al., HotNets 2018
Example: Drone-video-based tracking
Technological constraints: Moore’s law

- Processors aren’t clocked faster any more (Dennard scaling)
- Soon, can no longer pack more transistors in the same area (feature size limits)

- Implication (1): Application code won’t automatically get faster
- Implication (2): Need to re-design applications or the hardware from the ground up
Trend: compute offloads to accelerators

- Example: smartNICs (e.g., Azure NIC)
  - Hardware runs (part of) the network stack’s processing

- Other accelerators:
  - GPUs
  - TPUs
  - Matrix computation accelerators in the research realm
Disaggregation of resources

- Typical server: compute cores, memory, storage. Problems:
  - Memory wall: not enough bandwidth between compute & mem
  - Provisioning for evolution in storage and mem technologies
  - Inefficient usage of per-server statically allocated resources
The need for disaggregation

*Figure 1*: Distribution of relative disk/memory capacity demand to CPU usage for tasks in Google’s datacenter.
Disaggregation

[Network requirements for resource disaggregation, OSDI’16]
Research questions

• Want to build **resource blades**: separate compute, mem, storage
• Can we provide a high bandwidth low latency fabric to interconnect the different components?
• Should communication be reliable? Packet or circuit switched?
• Resource allocation for different applications?
• Application abstractions: move away from VMs?
• How should the new OS look like? Failure models? Abstractions of memory? Storage?
The times they are a-changin’…

- Technological constraints
  - Demise of Moore’s law, memory wall
- Tools
  - Machine learning
- Industrial trends
  - Consolidation, disaggregation, net neutrality
- Unconventional networks
  - Blockchain networks, edge networks
Your thoughts?

What are you excited about? What would you like to work on?
Course presentation

• 10 mins per group + 2 mins for questions
• Describe the problem
• Your solution approach
• Existing approaches and if/how your approach is different

• **Report on your progress:**
  • What have you built?
  • Can you show a demonstration?
  • Can you show preliminary evaluation results?
  • What lessons have you learned from this exercise?
Course feedback