Please put away computers and just use pencil and paper for note taking

All slides are available online

Introduction

Arthur J. Redfern
arthur.redfern@utdallas.edu
Jan 14, 2019
Jan 16, 2019

Outline

- Welcome to the course
- Motivation
- Course preview
- Logistics
- Expectations
- Opportunities
- Questions

Welcome To The Course

Thank You

- You're taking an important step in your academic and professional careers
- I appreciate you taking it with me
- I'm looking forward to a fun semester

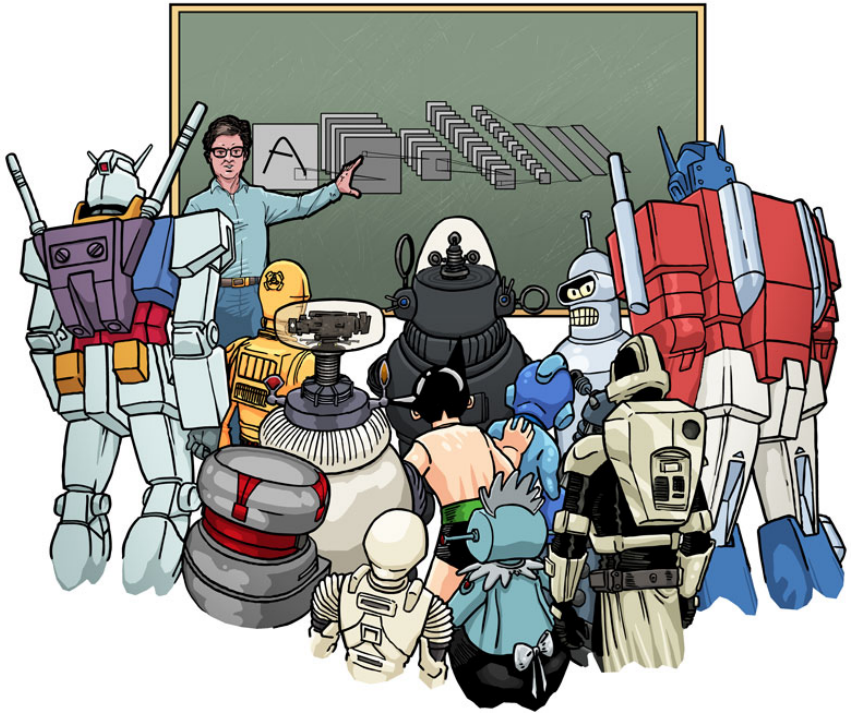


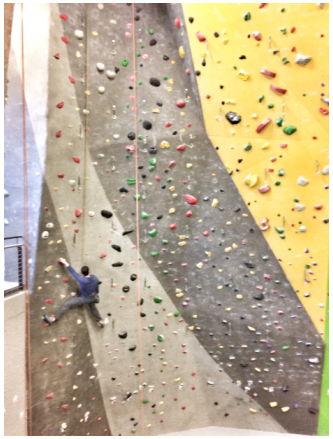
Figure from https://www.reddit.com/r/MachineLearning/comments/4tmpl9/tribute_an_art_made_by_andré_pinto_anthill_comics/

About Me

- Grew up a little south of Richmond, Virginia
- BS in EE from University of Virginia in 1995
  - Started in chemical engineering
  - Moved to electrical engineering
  - Specialized in signal processing
- PhD in ECE from Georgia Tech in 1999
  - My thesis was on Volterra systems (a nonlinear generalization of convolution)
- I'm ok with you calling me any of {Arthur, Dr. Redfern, Professor Redfern}

About Me

- Moved to Dallas to work at Texas Instruments in 2000
  - Physical layer communication system design
  - Signal processing for analog systems
  - Machine learning
- Currently I manage a machine learning lab in the TI Embedded Processors organization
  - Algorithms, software and hardware for different applications
  - This class will cover much of the same (that's not an accident)
- Live in Plano, Texas with my wife and son
- I like to add a new hobby every few years
  - Cars, guitar, poker, golf, running, biking, yoga, climbing, ...

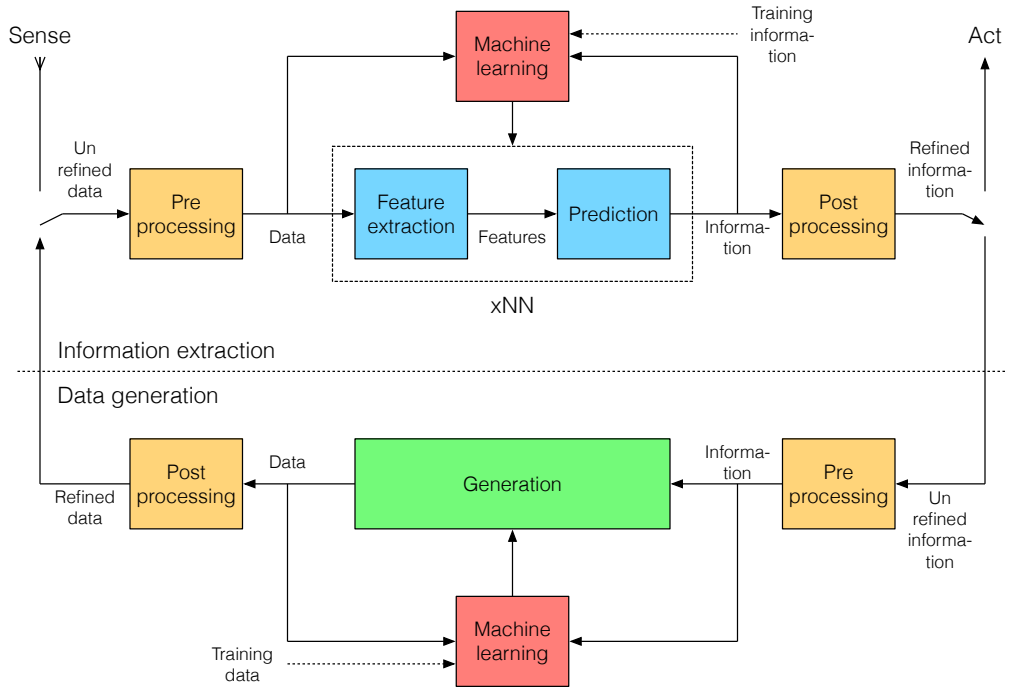


Motivation

Information Extraction

Data to information

- Sense → compute → act
- Transformation from un refined data space to data space to feature space to information space to refined information space



Definitions (not Webster quality)

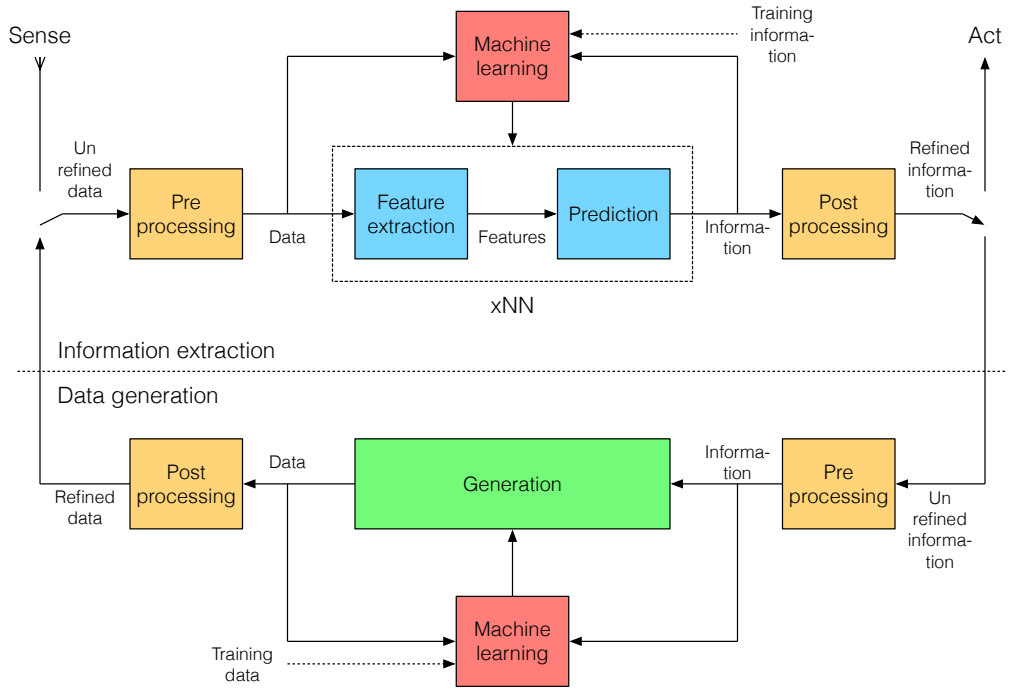
- Intelligence is the ability to acquire and apply knowledge
  - Artificial intelligence is intelligence exhibited by algorithms
- Learning is the acquisition of knowledge from experience
  - Machine learning is learning from data (experience) applied to an algorithm such that it exhibits artificial intelligence
  - Deep learning is machine learning applied to a deep structure

xNNs are deep structures trained using deep learning to exhibit artificial intelligence

Information Extraction

Data to information

- Pre processing
  - Make feature extraction easier
  - Data cleaning, dimensionality reduction, … frequently via application specific side information
- Feature extraction
  - Make prediction easier
  - Hand engineered or learned
- Prediction
  - Classification (discrete)
  - Regression (continuous)
- Post processing
  - Clean up predictions frequently via application specific side information



Trend: instrumentation and automation of everything

Why Use xNNs For Information Extraction?

Theory

- Many tasks can be cast as a classification problem
  - Probability of input x belonging to class y
  - Given an input x predict a pmf P(y|x)
- Neural networks are universal approximations
  - The view of xNNs as a function applicable to all sorts of different data inputs is very important to keep in mind
  - Typically use them to approximate the mapping from x to P(y|x)
- It's possible to design xNN to exploit structure in the data
  - Spatial with CNNs
  - Sequential with RNNs
  - Localized with attention

Figure from https://www.mapillary.com/dataset/vistas

Why Use xNNs For Information Extraction?

Practice

- It's possible to train xNNs from input output data
  - End to end supervised learning allows feature learning vs feature engr (though implicit in network design and training hyper params)
  - Automatic differentiation with reverse mode accumulation
  - Gradient descent variants
  - Keys are convergence and regularization for generalization
- Software and hardware exists for efficient implementations
  - High level graph specification and transformations
  - Low level graph software runtime
  - Primitives for compute and data movement
  - Lowering of tensor ops to matrix matrix mult where possible
- They provide state of the art results in many applications
  - Vision, speech and language are the biggest successes
  - Games, art, control, genomics, ...
  - A general tool with logical extensions to new apps going forward

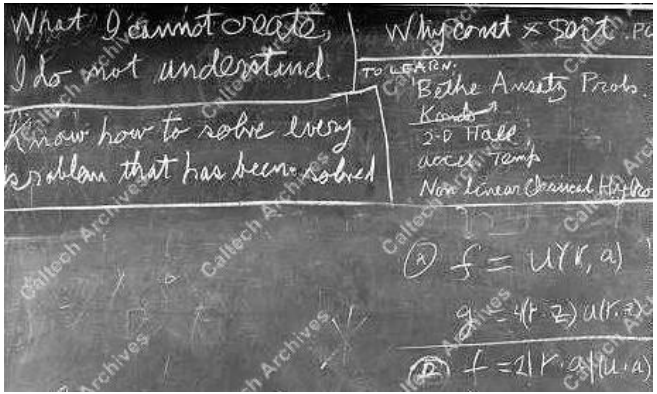
You can't discuss top performing vision, speech or language algorithms without xNNs

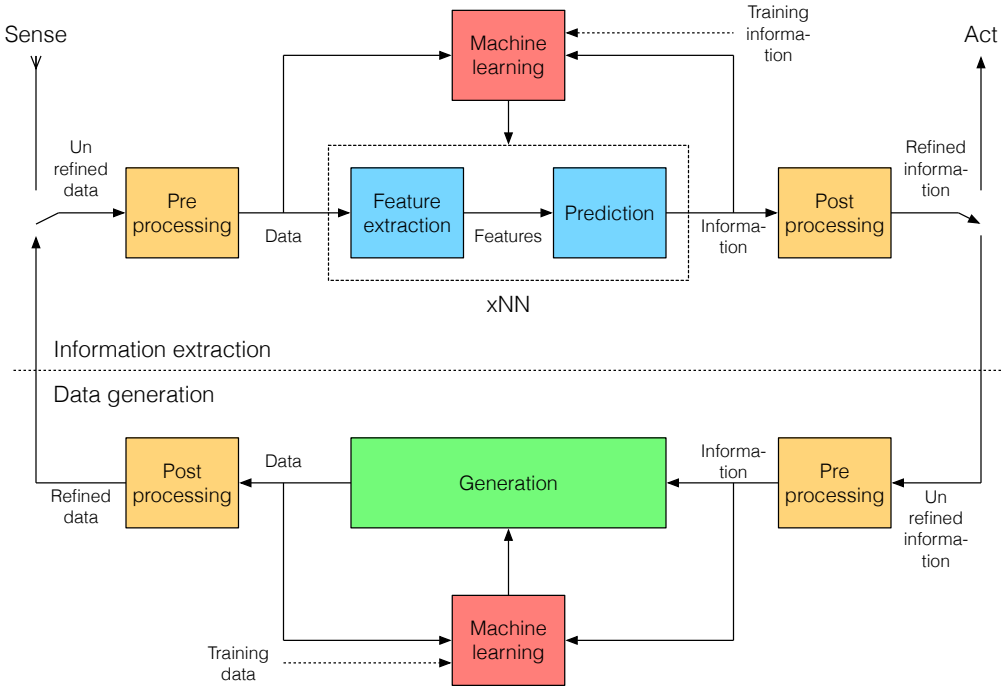
Figure from https://www.tensorflow.org/tfrc/

Data Generation

Information to data

- Richard Feynman: “What I cannot create, I do not understand.”
- Generative models are learning to create better data from information
  - The complement to info extraction
  - A 1 to many mapping
  - But currently less mature





Picture from http://archives.caltech.edu/pictures/1.10-29.jpg

Data Generation

While less mature that information extraction, data generation via GANs and other methods is getting better all the time; this is from Jun **2014**

Figure from https://arxiv.org/abs/1406.2661

Data Generation

While less mature that information extraction, data generation via GANs and other methods is getting better all the time; this is from Oct **2017**

Figure from https://arxiv.org/abs/1710.10196

Why xNNs Now?

Success

- A confluence of developments
  - More data
  - More compute
  - Better network designs
  - Better training algorithms
  - More efficient information distribution
  - Applicability to problems of interest
  - More people, companies and funding
- Successes leading to positive feedback

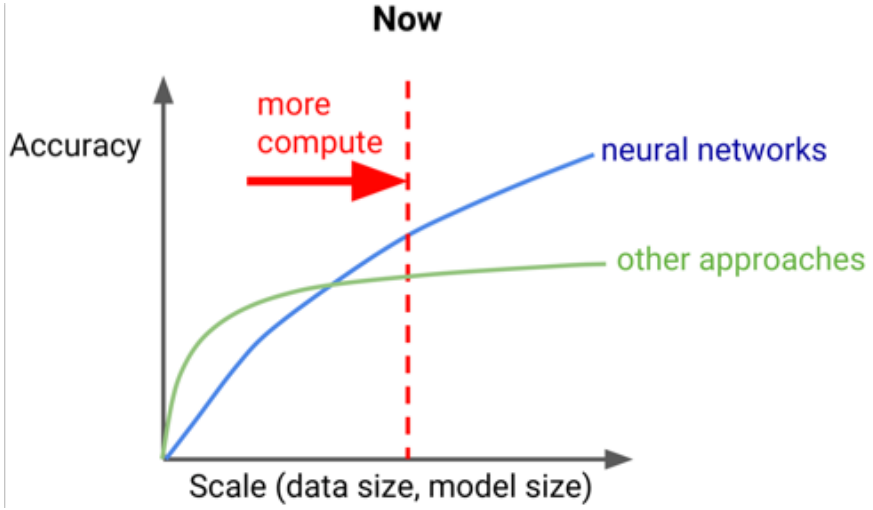


Figure from https://www.hotchips.org/wp-content/uploads/hc_archives/hc29/HC29.22-Tuesday-Pub/HC29.22.69-Key2-AI-ML-Pub/HotChips%20keynote%20Jeff%20Dean%20-%20August%202017.pdf

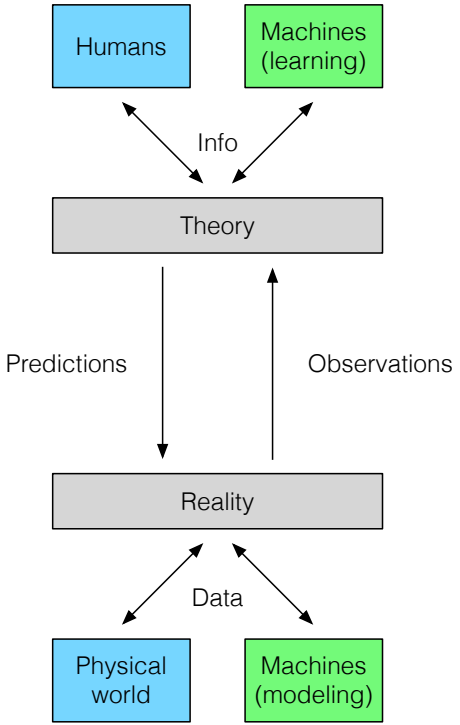
Why xNNs Now?

Spreading from vision, speech and language to all other fields; previously theory / brains was the domain of humans, now so much data can be generated in experiments and relationships in the data are so complex that algorithms are needed for both uncovering and extracting information

- Loop
  - Theory allows the prediction of reality
  - Reality provides observations that lead to theory

- Reality (the brawn)
  - For most of history was provided by physical systems
  - Computers / algorithms are commonly used now for modeling physical systems and generating observations
    - At 1 point in time computational science was semi controversial
    - Now it's common place and not given a 2nd thought

- Theory (the brain)
  - For most of history was provided by humans
  - Computers / algorithms are starting to become more common for the extraction of information from observations and have the potential to lead to new theory



Why xNNs Now?

A growing number of internal uses at companies to complement external applications

- Typically think about external applications
  - Ex: build a vision system to put in a product and sell to consumers
- But there's a less visible (no pun intended) trend of more and more internal uses for companies
  - Basically anywhere there's a heuristic being used can this be replaced with an xNN trained from input output data
  - Ex: Google Datacenter cooling
  - Ex: compiler heuristics

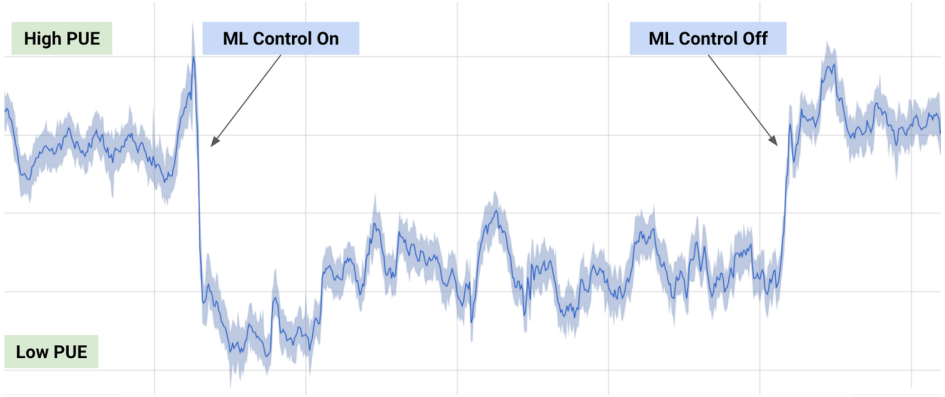


Figure from https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/

Why xNNs Now?

They're used in applications that capture the imagination; Soccer On Your Tabletop shown here

Figure from https://grail.cs.washington.edu/projects/soccer/

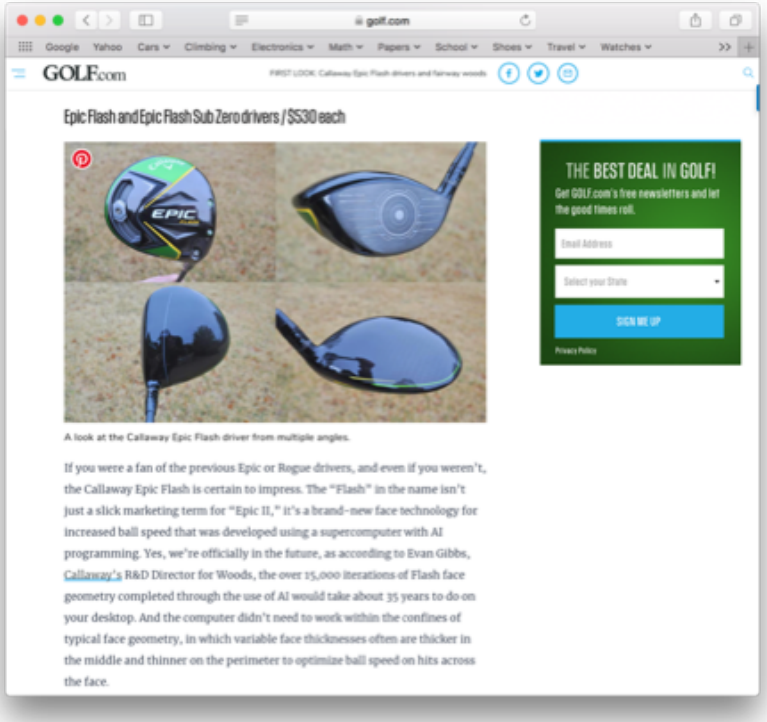
Why xNNs Now?

They're used in applications that capture the imagination; Soccer On Your Tabletop shown here

Figure from https://grail.cs.washington.edu/projects/soccer/

Hard To Escape

- It feels like AI / ML / xNNs are everywhere …
- Ex: Callaway Epic Flash Driver
  - "It's a brand-new face technology for increased ball speed that was developed using a supercomputer with AI programming."
  - https://www.golf.com/gear/2019/01/04/callaway-epic-flash-drivers-fairway-woods/
- Side note: unfortunately for me the weak link in my golf game is the human and not the club :(



What's In A Name?

- Some people are attracted to neural networks because they sound cool
  - Woo-hoo! We're building robot brains!
- Some people are turned off by neural networks because they sound like fiction
  - I'm not a sucker, I've seen this movie before
- How would you think about neural networks if they were instead called a composition of matrix multiplication, convolution, piecewise linear and transcendental functions?
- A suggestion: don't get hung up on names too much either way as you don't want someone else's words having control over you, instead look at what a technology can do

Mostly useless fact: I'm a big fan of the name Aquaman's parents gave him in the movie (Arthur)

Figure from https://www.vanityfair.com/hollywood/2019/01/aquaman-box-office-biggest-dc-movie

Course Preview

Cooking Vs Baking

- Cooking is ~ an art
  - "In general, cooking is considered an “art” because you are free to change the measurements or ingredients based on your preference”
- Baking is ~ a science
  - "Considered a “science” because it generally calls for accurate and precise measurements of the ingredients"
- Deep learning has room for both
  - But when we’re cooking we’ll still be guided by theory

Quotes from https://difference.guru/difference-between-cooking-and-baking/ and figure from http://whiteapronblog.com/2017/06/02/north-carolina-thirteen-layer-cake/

3 Parts To The Course

Math

- Linear algebra
- Calculus
- Probability
- Algorithms

Networks

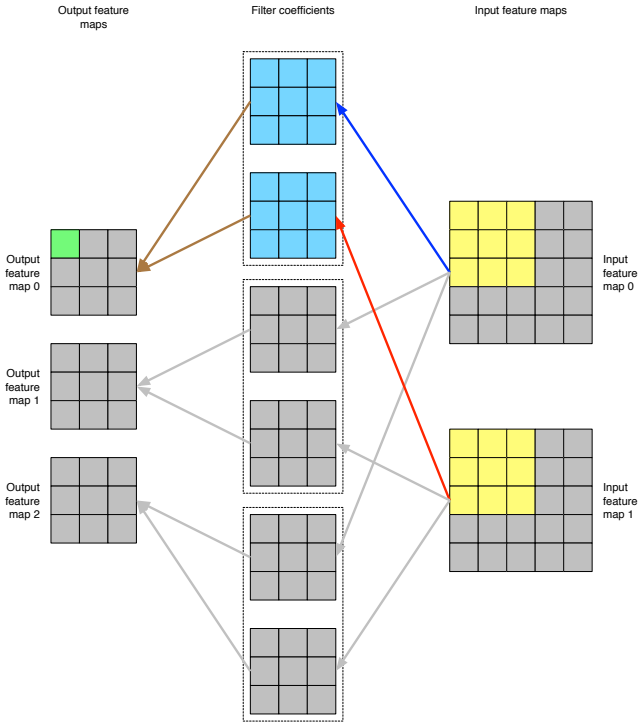
- Design
- Training
- Implementation

Applications

- Vision
- Speech
- Language
- Games
- Art

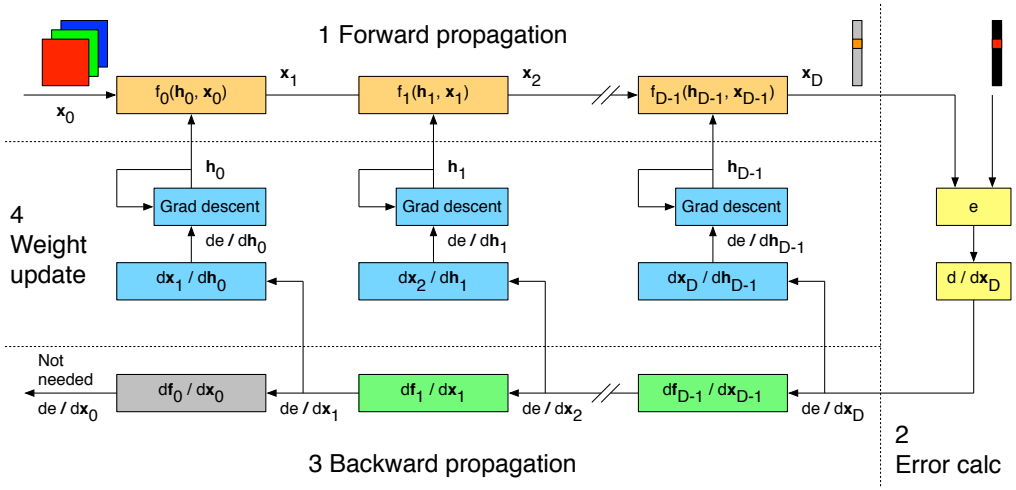
Math: Linear Algebra

- Dimensionality reduction
  - Frequently part of pre processing
  - DFTs and PCA for dimensionality reduction
  - Matrix embeddings
- Transformations from data to features to classes
  - xNNs are compositions of nonlinear functions (layers)
  - Linear transformations are key components of the nonlinear functions with learnable parameters that control the network mapping
  - Examples: densely connected, CNN style 2D convolution, RNN and attention based layers
- A laundry list of topics
  - Sets, fields, vectors, matrices, tensors, functions, vector spaces, normed vector spaces, inner product spaces, matrix vector multiplication, matrix matrix multiplication, CNN style 2D convolution, RNNs, attention, average pooling, DFTs and PCA



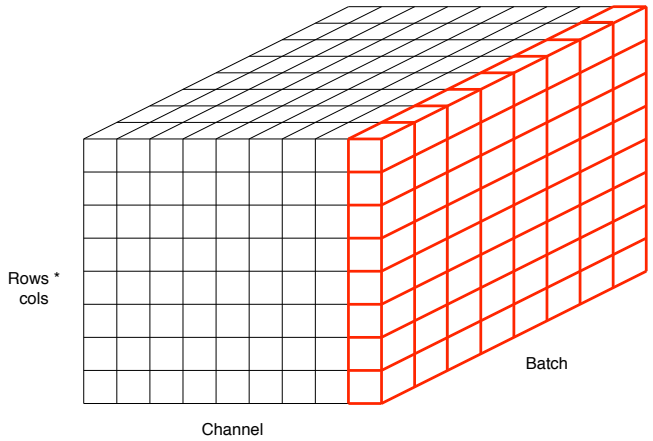
Math: Calculus

- Universal approximation
  - Proof
- Training
  - Automatic differentiation with reverse mode accumulation to back propagate error gradients
  - Gradient descent variants to update learnable parameters
- A laundry list of topics
  - Derivatives, sub derivatives, partial derivatives, gradients, Jacobians, chain rule, critical points, gradient descent, automatic differentiation with reverse more accumulation and universal approximation



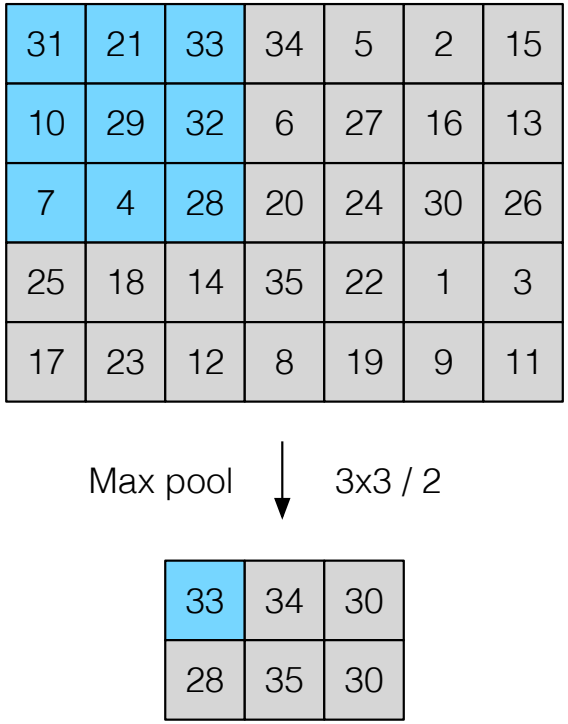
Math: Probability

- Training
  - Training vs testing data
  - Parameter initialization
  - Information flow
  - Normalization of feature maps between layers
  - Stochastic width / depth based regularization to improve generalization
  - Loss functions based on distances between pmfs
- Modeling
  - Next element prediction
  - Conditioned on inputs
- A laundry list of topics
  - Probability spaces, events, random variables, expected value, normalization, law of large numbers, central limit theorem, random processes, stationarity, time averages, ergodicity, entropy, mutual information, Kullback Leibler divergence, data processing inequality, compression, Huffman and arithmetic coding



Math: Algorithms

- Pooling
  - Common down sampling strategies include max operations
  - Max, spatial pyramid and RoI pooling
- Non maximal suppression
  - A common post processing for multiple object detection and object based image segmentation
  - Dependent on sorting
- A laundry list of topics
  - Comparison sorts, sequential merge sort, parallel merge sort, pooling, median and rank order filtering



Networks: Design

- How to design a network to achieve a goal
  - How to select and combine the layers described in the linear algebra and algorithm lectures
  - Encoder decoder style architectures
  - Lots of examples of different backbone networks

- A laundry list of topics
  - Goals, size considerations of the network, feature maps and filter coefficients, problem complexity, graph specification, layer types, tail body head decomposition, tail designs, head designs, body designs including chain, parallel, dense and residual structures, optimized architecture search and visualization

- The start of the 2nd spiral presentation of material
  - The math lectures are a 1st look at everything needed to build and train a network
  - The network design, training and implementation lectures look at everything in more detail

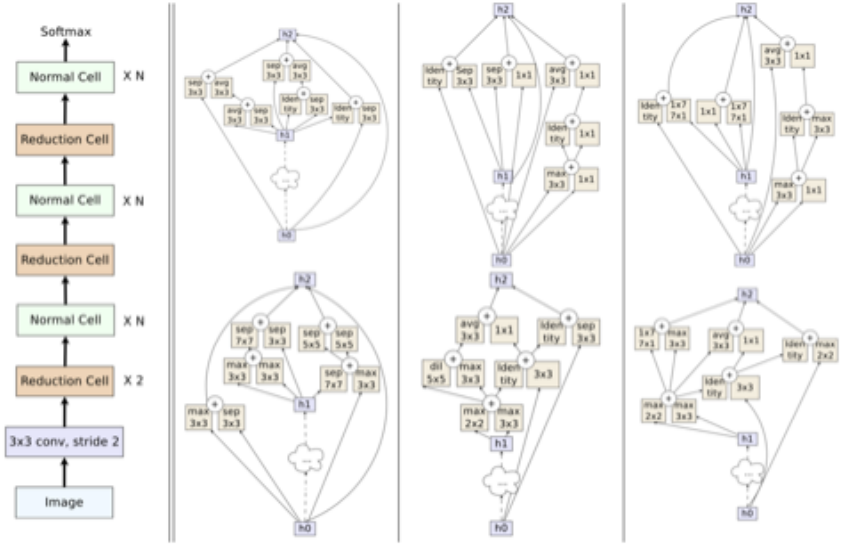
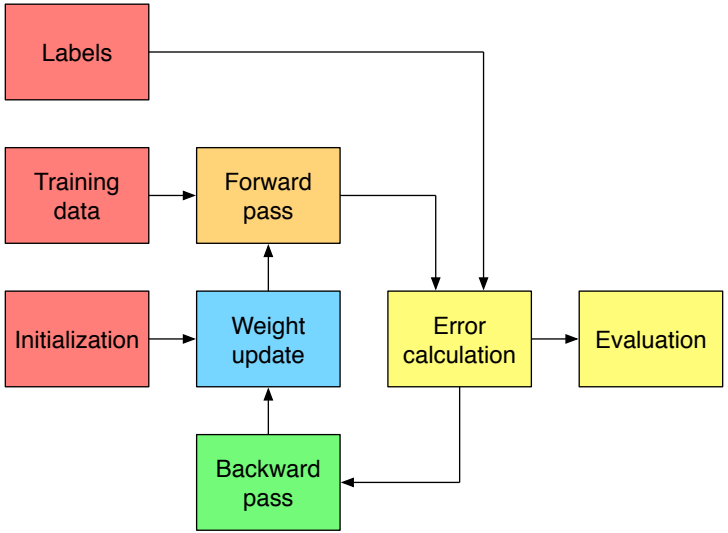


Figure from https://arxiv.org/abs/1802.01548

Networks: Training

- Estimating the weights that control the xNN mapping
  - Convergence speed and accuracy of result
  - Regularization to improve generalization
  - Builds on the material in the calculus and probability lectures

- A laundry list of topics
  - Supervised learning, differences with function optimization, convergence, overfitting, regularization and generalization, the curse of dimensionality, training validation testing data splits, natural data, labeling, cleaning, synthetic data, hand engineered generation, learned generation, data augmentation, random initialization, transfer learning, curriculum learning, batch normalization, group normalization, stochastic width and depth, classification and regression losses, loss surface shapes, unequal class weighting, aux network heads, multiple network heads, check pointing, reversible architectures, batch size, weight update methods including SGD, momentum, AdaGrad, RMSProb and Adam, learning a solver, weight decay, gradient noise, gradient clipping, synchronous stochastic gradient descent and hyper parameter selection



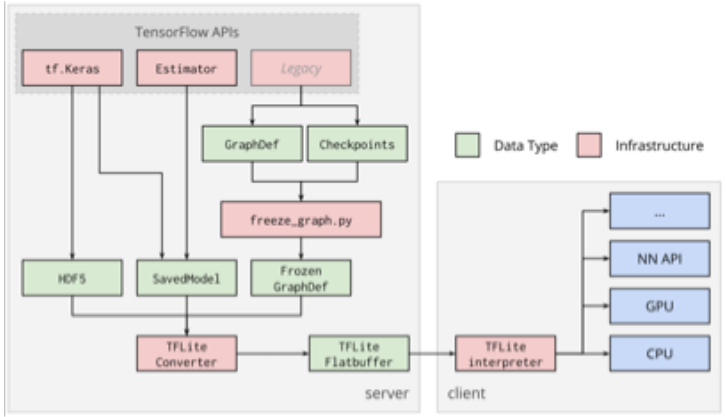
Networks: Training

- But I'm only 1 person and only have a computer with 1 GPU, how can I compete with companies and researchers with access to significantly more compute?

- A quote from Jeremy Howard from Fast.ai from the post: Now anyone can train ImageNet in 18 minutes (https://www.fast.ai/2018/08/10/fastai-diu-imagenet/)
  - "I've seen variants of the "big results need big compute" claim continuously over the last 25 years. It's never been true, and there's no reason that will change. Very few of the interesting ideas we use today were created thanks to people with the biggest computers. Ideas like batchnorm, ReLU, dropout, adam/adamw, and LSTM were all created without any need for large compute infrastructure. And today, anyone can access massive compute infrastructure on demand, and pay for just what they need. Making deep learning more accessible has a far higher impact than focusing on enabling the largest organizations - because then we can use the combined smarts of millions of people all over the world, rather than being limited to a small homogeneous group clustered in a couple of geographic centers."

Networks: Implementation

- Bigger networks enable higher accuracy but require more data movement and compute
  - Network modifications to improve implementations, hardware design and software to connect algorithms to hardware
- A laundry list of topics
  - Networks: theoretical complexity, precision, hardware size vs model size, training vs testing, data formats, quantization, network sizing and network simplification
  - Hardware: Moore's law, Dennard scaling, dark silicon and dark memory, power, roofline models, SoC architectures, domain specific architectures, control, memory, data movement, compression, Amdahl's law, computational basis, matrix multiplication primitive, inner, outer, Strassen style, input power of 2, sparse and analog matrix multiplication, Winograd style convolution, sort primitive, tree configurations, torus configurations and hardware design examples
  - Software: high level graph specification, high level graph transformations, static vs dynamic graphs, sessions, graph compilers, low level graphs, runtime initialization, runtime execution, software design examples, predicting performance and benchmarking

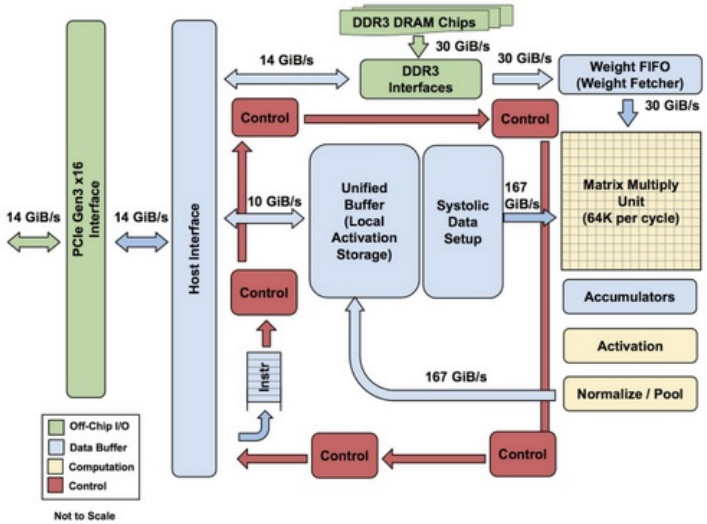


Software

Figure from https://www.tensorflow.org/lite/convert/

Networks: Implementation

- Bigger networks enable higher accuracy but require more data movement and compute
  - Network modifications to improve implementations, hardware design and software to connect algorithms to hardware
- A laundry list of topics
  - Networks: theoretical complexity, precision, hardware size vs model size, training vs testing, data formats, quantization, network sizing and network simplification
  - Hardware: Moore's law, Dennard scaling, dark silicon and dark memory, power, roofline models, SoC architectures, domain specific architectures, control, memory, data movement, compression, Amdahl's law, computational basis, matrix multiplication primitive, inner, outer, Strassen style, input power of 2, sparse and analog matrix multiplication, Winograd style convolution, sort primitive, tree configurations, torus configurations and hardware design examples
  - Software: high level graph specification, high level graph transformations, static vs dynamic graphs, sessions, graph compilers, low level graphs, runtime initialization, runtime execution, software design examples, predicting performance and benchmarking

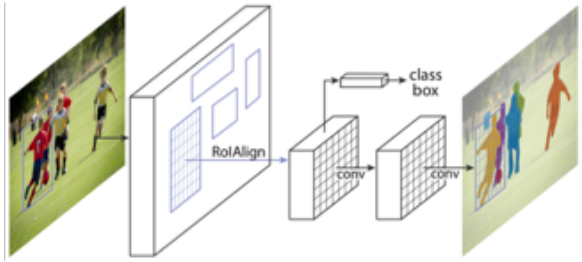


Hardware

Figure from https://www.nextplatform.com/2017/04/05/first-depth-look-googles-tpu-architecture/

Applications: Vision

- The vision lectures focus on CNNs
  - Exploit spatial structure
  - Transform all sorts of vision applications into classification problems
  - Use CNNs as a function that maps from images to information
  - Encoder decoder architectures with special attention paid to the spatial localization of information
- A laundry list of topics
  - Image capture and processing, hardware design examples, classification, pixel segmentation, up sampling, encoder decoder with skip connections, Atrous convolution, spatial pyramid pooling, 1 and 2 stage approaches to multiple object detection, feature pyramids, anchor boxes, spatial pyramid pooling, RoI pooling, region proposal networks, iterative methods, non maximal suppression, confidence threshold, intersection over union, precision, recall, precision recall curve, 2 and 3 stage approaches to object based segmentation, RoI align, depth estimation, stereo fundamentals, motion estimation and motion fundamentals
- The start of the 3rd spiral presentation of material



Figures from https://arxiv.org/abs/1703.06870

Applications: Speech

- The speech lectures focus on RNNs
  - Exploit sequential structure with RNNs (but also include CNNs and attention)
  - Use RNNs as a func that maps from sound to info
  - Encoder decoder architectures with special attention paid to the monotonic alignment of sequential information

- A laundry list of topics
  - Speech and audio signal chain, sampling, pre emphasis, windowing and spectrograms, MFCC, RNNs, GRUs, LSTMs, bi directional, pyramidal, sources of variability, speaker verification, speaker recognition, wake up, limited vocabulary speech recognition, confusion matrix, speech to text, sequence to sequence models, CTC, beam search, language model, auto segmentation, RNN transducer, attention, transition possibilities, alignments, text to speech, intermediate representation conversion and audio signal conversion

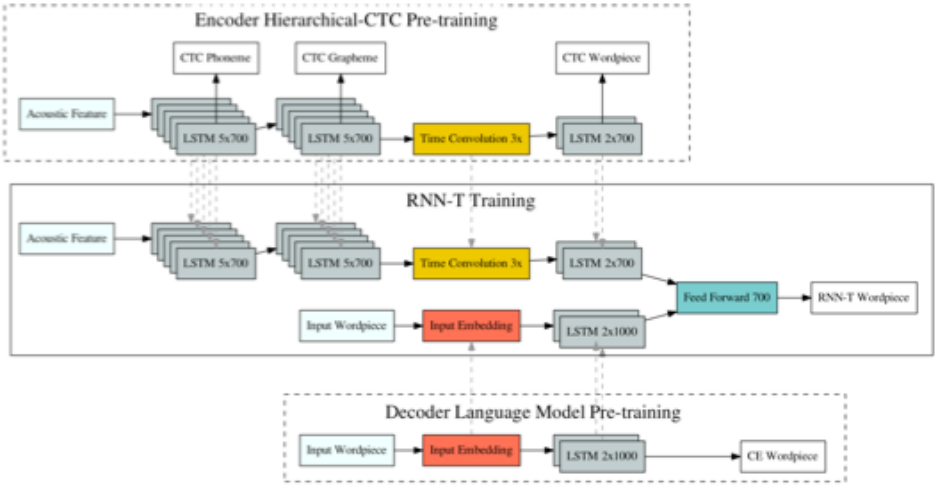


Figure from https://arxiv.org/abs/1801.00841

Applications: Language

- The language lectures focus on attention
  - Exploit localized information with attention (but also include CNNs and RNNs)
  - Use attention as a function that maps from language to information
  - Encoder decoder architectures with special attention paid to the localization of sequential information
- A laundry list of topics
  - Word embeddings, the distributional hypothesis, SVD based, continuous bag of words, skip grams, visualization, word similarity and analogies, task specific optimization, language modeling, N grams, neural language models, perplexity, character based, translation, sequence to sequence, greedy and beam search decoding, structured prediction, attention, architecture exploration, self attention, transformer and BLEU

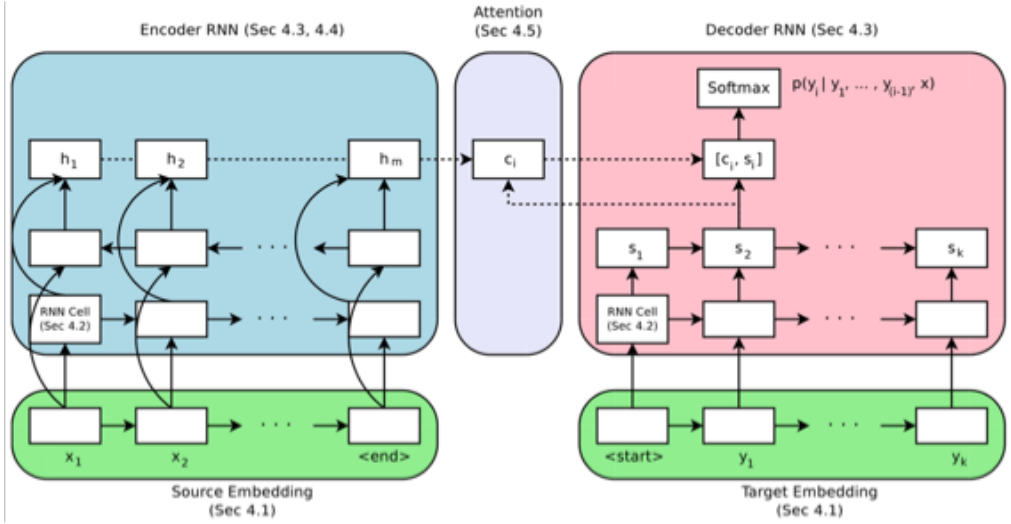


Figure from https://arxiv.org/abs/1703.03906

Applications: Games And Art

- New this semester
  - So the below topics have a higher level of variance with respect to what will actually be discussed
- Games
  - Reinforcement learning
  - Atari
  - Go and chess
- Art
  - Style transfer
  - Auto encoders and GANs
  - Image generation

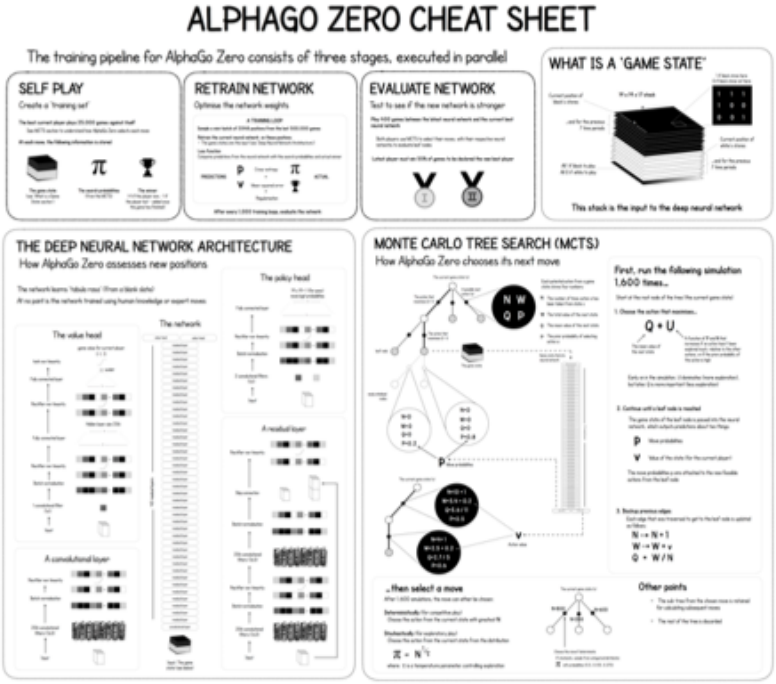


Figure from https://applied-data.science/static/main/res/alpha_go_zero_cheat_sheet.png

Depth And Breadth

- This slide title refers to topic coverage, not networks
  - Networks will always be deep :)
- We're going to talk about a lot of topics
  - Some in depth
  - Some in passing
- Why include the briefly mentioned topics?
  - There's a lot of stuff in this field that's useful to know that it exists and have a basic idea of the why and how it works
  - But it's not necessary or practical to know every detail about everything (trust me, you won't, but that's totally ok)

Figure from https://explorethecanyon.com/faqs/

So What's Missing?

- The amount of stuff that's not included >> the amount of stuff that's included
  - Missing sub topics within the topics that are covered
  - Missing topics altogether

- 1 role of a teacher is to be a guide
  - You see what I think is most important
  - But your only hope is to learn how to learn
  - The last paper is not written

Figure from http://www.arxiv-sanity.com/toptwtr

Logistics

Unofficial Course Objections

- Understand why xNNs work
- Understand how to design, train and implement xNNs
- Understand how xNNs are applied to vision, speech, language and other applications
- Understand how to apply xNNs to new applications
- Learn how to learn

Grades

- 3 in class closed everything pencil and paper only tests at the end of each section
  - 25% math test
  - 25% networks test
  - 25% applications test
- Homework throughout the semester
  - 25% total for all assignments
- No final
- eLearning for homework assignments and grade distribution

This is a change from last semester

Highlighting The Role Of Homework

- Homework will serve multiple purposes
  - **Reading**: extending in class knowledge and learning to learn via paper reading
  - **Theory**: evaluation and reinforcing of the concepts discussed in class
  - **Practice**: an avenue to gain familiarity with implementations

- Highlighting the implementation component of homework
  - The goal is to get everyone up and running as soon as possible with high level software libraries
  - You build skills via the relatively close reproduction of results of others
  - Which will lead to confidence in your ability to design, train, evaluate and revise networks that contain new ideas of your own

- This semester everyone will use Python / TensorFlow for the software implementation portion of their homework assignments
  - This isn't a negative reflection on other libraries (many are excellent)
  - This is a reflection on my bandwidth and need for simplification

Figures from https://www.python.org and https://www.tensorflow.org

Communication

- UT Dallas email
  - Individual emails
- eLearning
  - Full class emails
  - Discussion board
- GitHub
  - Web site — https://github.com/arthurredfern/UT-Dallas-CS-6301-CNNs
  - Syllabus — Course syllabus with approximate schedule
  - Lectures — Will continually update
  - References — Links to many
  - Code — Will add as appropriate

Office Hours

- Typically the 1/2 hour before class each day
- Will complement with an additional 1 hour on some Fridays at the student union Starbucks (while drinking coffee) or food court (while eating lunch)
  - Will confirm in class on Wed
- Will complement with a discussion board on eLearning to facilitate student to student discussions

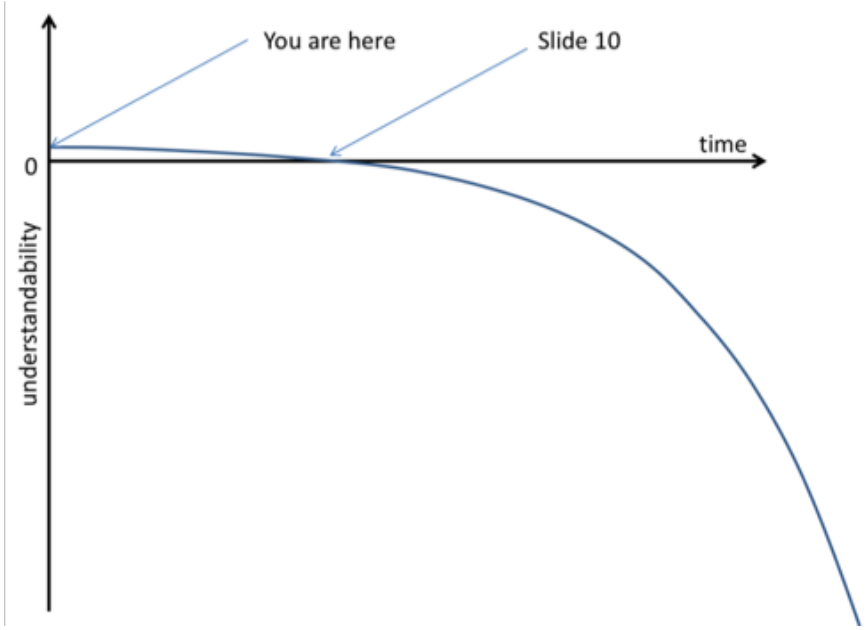
Classroom Citizenship

- Attendance is expected
- No computers in class
  - All slides are provided on the class web site
  - Pencil and paper for any additional notes you want to take
- Very limited use of mobile phones
  - If it becomes annoying to me I'll restrict it
  - Don't let it become annoying to me

Expectations

Of Me

- A logically laid out plan for both the whole course and individual lectures
- A willingness to modify the plan as needed
- While this is my 2nd time teaching the course and have a better idea on how things will go than the 1st time teaching ...
  - I'm also making changes to improve things which re introduces some ambiguity



I'll try and avoid this style of lecture (side note: this is 1 of the funniest / best figures I've ever seen in a presentation)

Figure from http://deeplearning.cs.cmu.edu/slides/lec23_24.RL.pdf

Of Me

- My best every class
- My opinions
  - It’s a special topics class
- That I’ll speak to adults like adults
  - It's a grad class
  - I want to be precise
  - But I don't want to make things unnecessarily complicated

Of You

- Honesty
  - In your work
  - In your interactions with other students
  - In your interactions with me
- Hard work
  - Nothing meaningful in life is easy
  - I promise you this won't be an exception
- Preparation
  - Review the lecture on your own before class
  - Listen to the lecture in class
  - Re review the lecture after class until it makes sense

My sources say no. This course covers a lot, some of it will be easy for you, some of it will be difficult. If you find yourself stuck on something, don't stress too much, come talk to me and we'll figure it out together.

Figure from https://i.redd.it/a5fk5dg6yflz.png

Of You

- Contribute to a friendly environment
  - It's great to shine as an individual through individual accomplishments
  - It's great to shine by helping others shine
    - This is a characteristic of a leader
- Be engaged
  - **Ask questions freely** – a goal of mine this semester is more interactive discussions, help me out with this
  - So speak up if something is unclear
  - And correct me if I'm wrong
- Help me learn your name
  - Say it when you ask a question

Opportunities

A Suggestion

- Look for opportunities beyond this course
  - Your own research
  - Your own company
  - UT Dallas HackAI or similar events (side note: I'd like a team of you to win this)
  - …
- At the end of this course you'll have a tool that can map from all sorts of different inputs to all sorts of different outputs, a way to train it and a way to implement it
  - What can you do with something like this?
- Want to bounce ideas off of someone? Come talk to me

Questions?

References

Recommendation

- For a curated list of generally useful references that apply to all parts of this course see
  - https://github.com/arthurredfern/UT-Dallas-CS-6301-CNNs/tree/master/References

- The references at the above web site are loosely divided into math, networks and applications, though there is a lot of overlap
  - The references tend to be relatively general and many are links to related courses with overlapping information (you're encouraged to check these out, they're excellent)
  - References that are more specific / focused in nature are listed in and at the end of each set of slides as appropriate