Design

Arthur J. Redfern
axr180074@utdallas.edu
Sep 17, 2018

Outline

Motivation

The Previous Lectures Covered A Lot

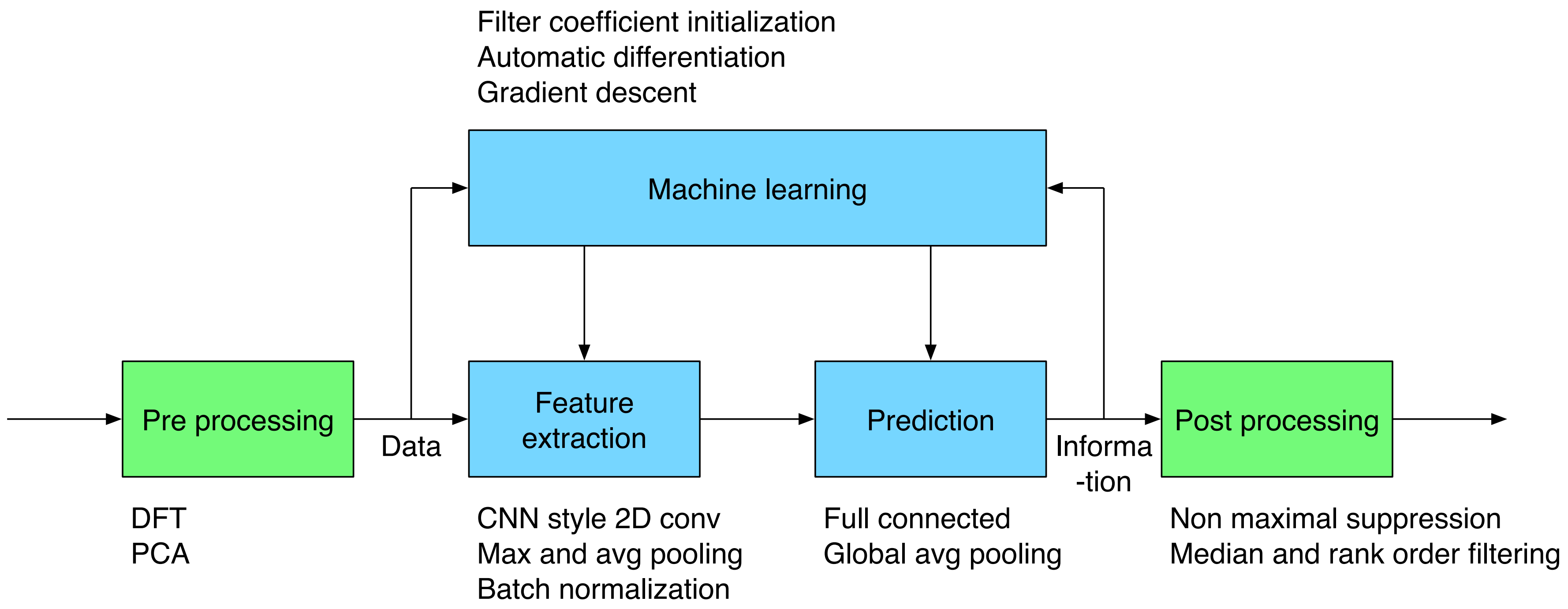
- We've briefly looked at a lot of different parts of basic math
    - 1 intro lecture where a framework for information extraction was introduced including pre processing, feature extraction, prediction and post processing
    - 2 lectures on linear algebra covering sets, fields, vectors, matrices, tensors, functions, vector spaces, normed vector spaces, inner product spaces, matrix vector multiplication, matrix matrix multiplication, CNN style 2D convolution, DFTs and PCA
    - 2 lectures on calculus covering derivatives, sub derivatives, partial derivatives, gradients, Jacobians, chain rule, critical points, gradient descent, automatic differentiation with reverse more accumulation and universal approximation
    - 2 lectures on probability covering 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 coding and arithmetic coding
    - 1/2 of a lecture on algorithms covering comparison sorts, sequential merge sort, parallel merge sort, pooling layers, median and rank order filtering and non maximal suppression

~ 30 lectures / semester, 7.5 = 25% allocated to math; remaining ~ 25% CNNs, ~ 25% implementation, ~ 25% application; minus tests and projects

Now We Start To Put The Pieces Together

• Introduction:	flow of pre processing, information extraction, prediction and post processing
• Linear algebra:	CNN style 2D convolutional layers, fully connected layers, pre processing methods
• Algorithms:	pooling layers, post processing methods
• Probability:	initialization, information in feature maps and filter coefficients, batch normalization, error functions
• Calculus:	filter coefficient estimation, approximation

Now We Start To Put The Pieces Together



But Realize That There’s (A Lot) More

- A basic amount of material from linear algebra, calculus and probability was needed such everything hangs together

- But it’s possible to go much deeper and broader in each of these topics
  - We’ll do some of that selectively in subsequent lectures
  - And you’re well setup to do more in the future based on your own interests
  - Starting from a strong base makes learning new material a whole lot better

- Summary: the diagram on the previous slide is a nice start but there’s a lot more for us to consider together and for you to consider individually

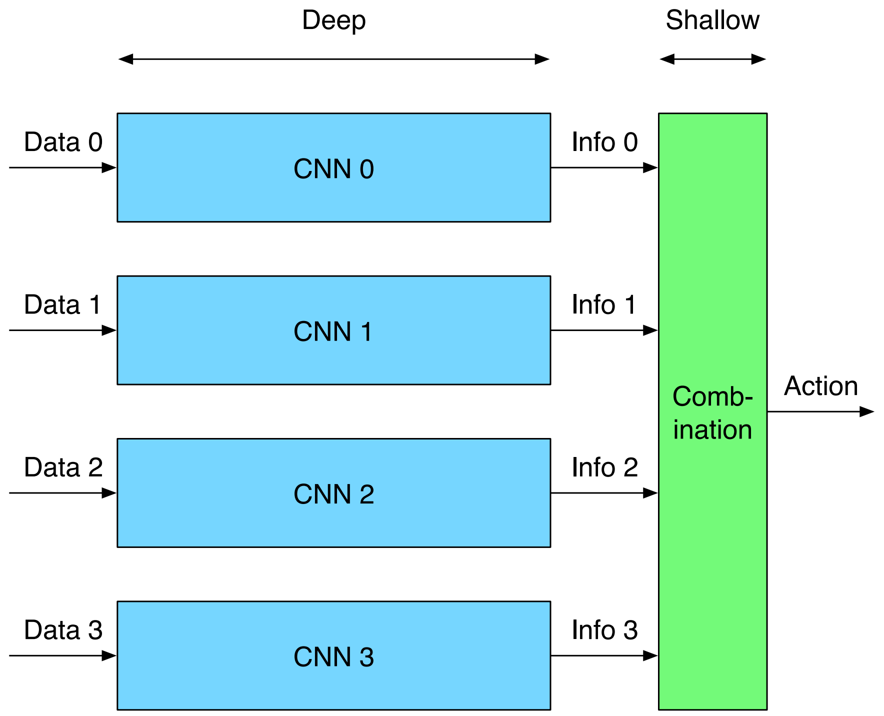
Goal

Keep Your Goals Simple

Both a deep network and general life comment

- You design a network to accomplish a goal
  - Never lose sight of this
- Keep the problems you address using deep networks like CNNs simple
  - Deep networks work best when there’s a whole lot of labeled data to train on
  - The simpler the task the more data you have related to that task
  - A simple combinatorics argument illustrates this
- Handle complex problems via a shallow combination of simple problems that were solved with deep networks
  - A smaller shallow structure likely requires less data to train
  - A CNN may or may not be used for this

Keep Your Goals Simple



Example: Crossing A Busy Street

- Goal: get from 1 side of the street to the other side in a safe, efficient and law abiding way

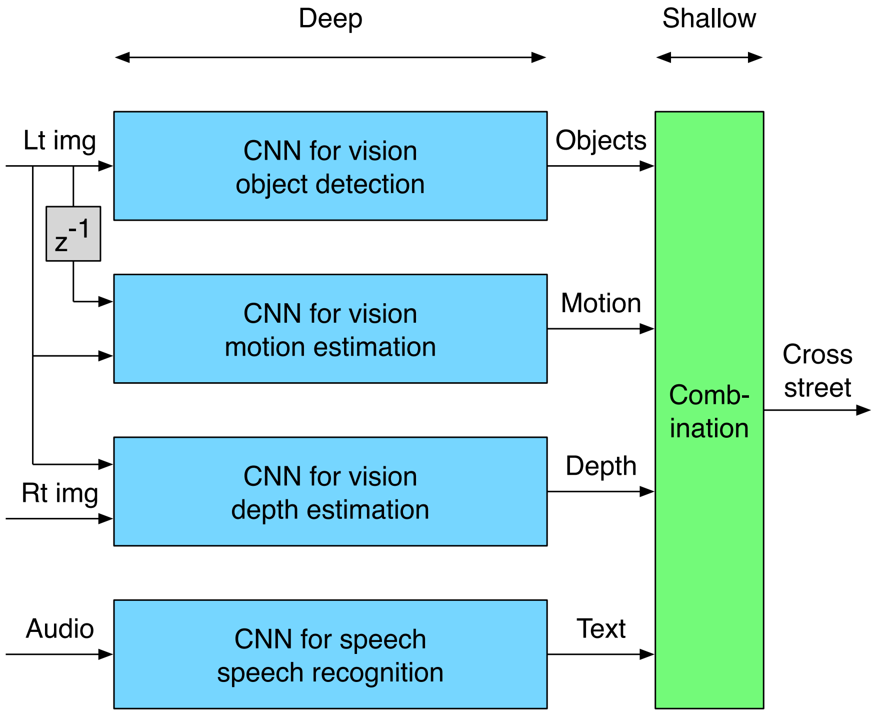
- How do you solve this problem as a human? An incomplete list includes ...

• Pre processing	look up from mobile phone
• Vision	recognition of many objects in the scene, approximate distances, approximate motion
• Sound	recognition of many sounds in the scene, approximate localization
• Modeling	prediction of object paths vs desirable object separation for different scenarios
• Risk analysis	from a personal health and legal perspective
• Action	decide whether to start crossing via some trajectory at some speed looking some way
• Repeat	
• Post processing	look down at mobile phone

https://www.freeimageslive.co.uk/files/images008/shibuya_busy_people.jpg

Example: Crossing A Busy Street

- How do you solve this problem as a computer?
  - Use CNNs to solve "simple" vision and speech problems, maybe part of the modeling prediction problem, maybe part of the action selection problem
  - Probably use other methods for risk analysis
  - Probably put the individual pieces together with a shallow structure


Deep
Shallow
Lt img
z^{-1}
Rt img
Audio
CNN for vision object detection
CNN for vision motion estimation
CNN for vision depth estimation
CNN for speech speech recognition
Objects
Motion
Depth
Text
Comb-ination
Cross street

This Semester And This Class

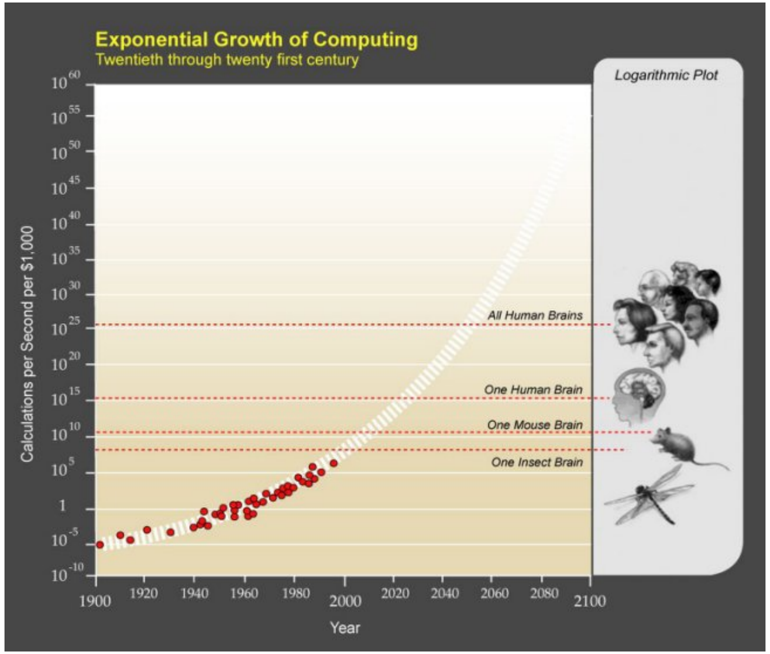
- This semester will focus on small simple problems solved via CNNs
  - It’s not that solving complex problems with shallow structures is not important (it is)
  - It’s just that simple problems are a pre requisite and / or subset of complex problems
  - Simple problems will also be sufficiently difficult to solve that we won’t get too bored
- It’s nice to have a default problem to think about new ideas in the context of
  - Will use image classification as our starting point in this lecture
  - Will later generalize in a natural way to more problems in subsequent lectures on applications

Sound Like A Good Plan?

Preliminaries

Inspiration

- It’s ok to look to nature for inspiration for basic principles that work
- But don’t get too hung up on biology and replicating brain structure
  - If you believe in physics, the brain is an existence proof of what can be done in 3 pounds of material and 20 W of power (20 % of an average human’s 100 W power budget)
  - If you believe in evolution, why build replica of the current human brain, why not build the better brain that people will have 1M years from now?
- Why have aliens never contacted us?



Theory Or Practice?

- Both (and that’s ok, it’s how science works)

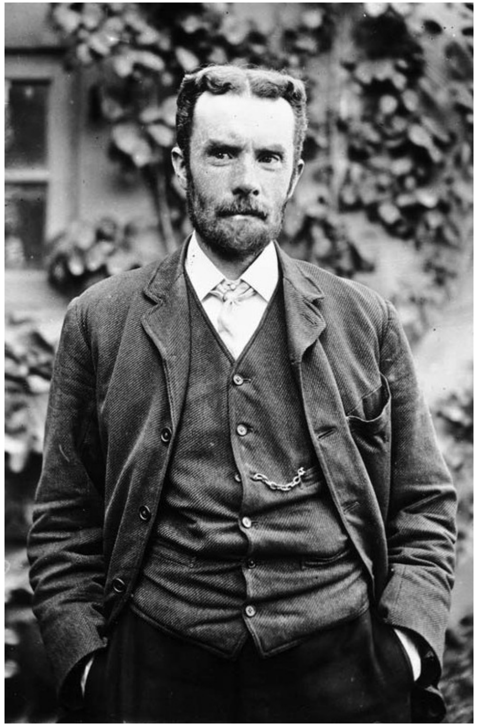
- A frequent critique of deep learning and CNNs is that no one knows why they work
  - This is way too general a statement
  - This is not correct
  - There are strong theoretical underpinnings and frameworks for understanding what's going on
  - Is every single detail known about every single thing? No. But what scientific field would answer yes?
  - We’ve seen some theory, realize that much more exists

- There’s a place in science for experimentation and trying new things
  - It’s a reasonable way to make progress
  - Experimentation works best when it’s guided by a combination of theory and intuition
  - Successes (→ practice) are nice in that they allow focused work on theory to explain

Oliver Heavyside

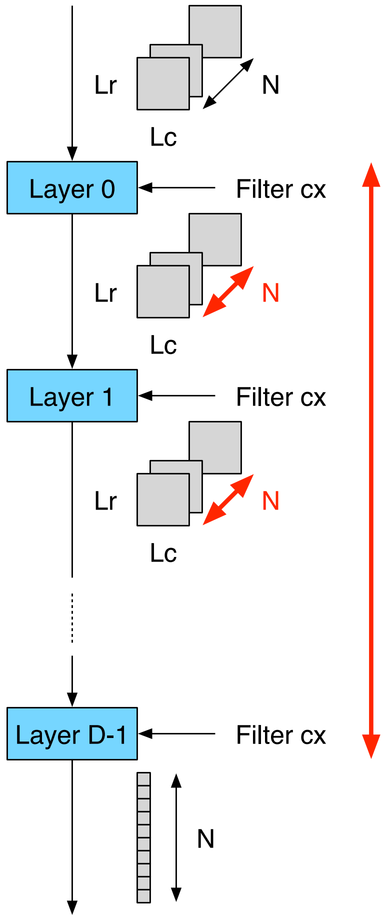
- “Mathematics is of two kinds, Rigorous and Physical. The former is Narrow: the latter Bold and Broad. To have to stop to formulate rigorous demonstrations would put a stop to most physico-mathematical inquiries. **Am I to refuse to eat because I do not fully understand the mechanism of digestion?**”

Quotes != proofs, but regardless, I like this 1


https://en.wikipedia.org/wiki/Oliver_Heaviside#/media/File:Oheaviside.jpg

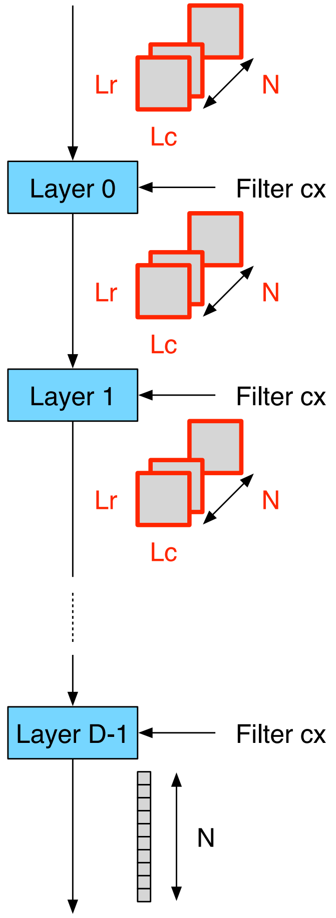
Network Size

- Network depth and breadth (D, $N_{i/o}$)
  - Deeper and wider networks can approximate more complex functions
  - Remember the universal approximation proof from the calculus lecture
  - Deeper and wider networks are enabled by more data and better hardware
  - Do you want a smaller brain or bigger brain?
  - How do you sell a house?
- Performance
  - A drawback of a deeper and wider network is increased complexity / reduced performance
  - Will look at this in detail later
  - For now, pay attention to compute at the beginning and memory at the end



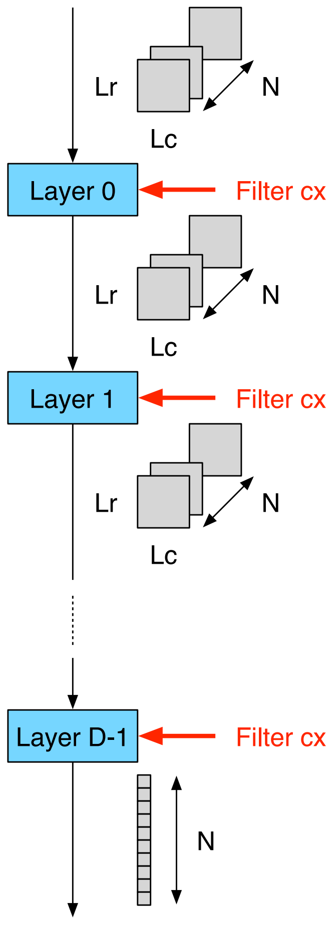
Feature Map Size

- Feature maps ($N_{i/o}$ x L_r x L_c per layer)
    - Feature maps encode information in the testing data
    - The information you're trying to extract is diffused within the feature maps
    - Need to track feature map data size as it flows through the network to make sure that there are no information killing bottlenecks
    - Remember the data processing inequality from the probability / information theory lecture



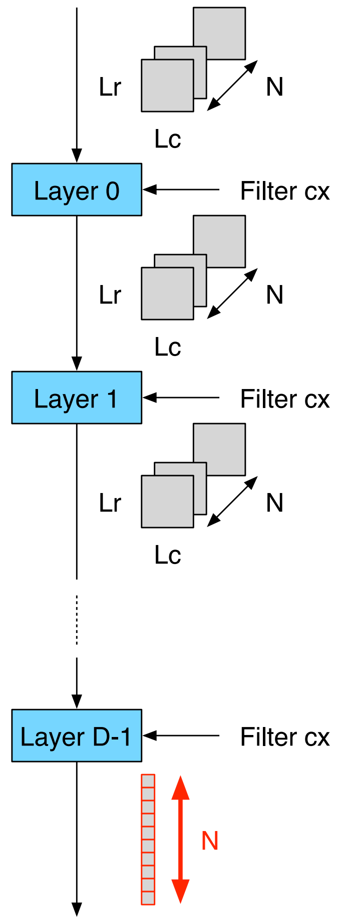
Filter Coefficient Size

- Filter coefficients (F_r x F_c x N_i x N_o and bias N_o per linear transformation)
  - Together with the network structure contain all information extracted from training data that will be applied to testing data
  - Remember automatic differentiation and gradient descent from the calculus lecture
  - Would like to have as few as possible because of performance reasons
  - But less filter coefficients tends to mean less extracted information, so there's a balance



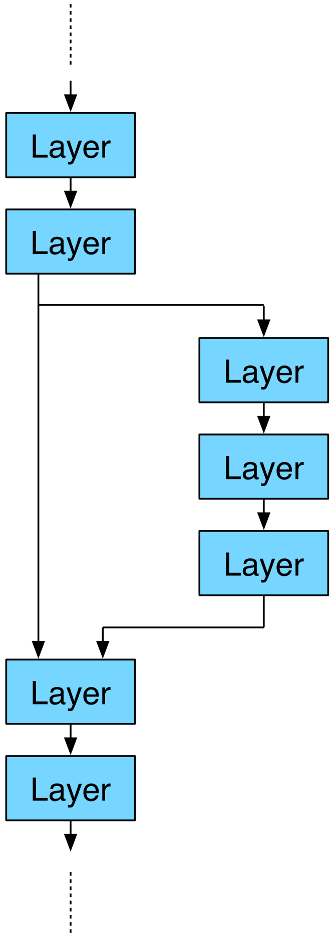
Problem Complexity

- It’s important to understand how complex the problem is you’re trying to solve
  - Classification to 2 classes, 1000 classes, regression to a real number, …
  - Easily separable classes or easily confused classes?
  - The simpler the function needed to solve the problem the simpler the network that can be used to solve it (and the converse is also true)
  - Remember the universal approximation proof from the calculus lecture
  - Lot’s of problems we care about are really pretty complex



Graph Specification

- Implicitly or explicitly, computational graphs are used for high level network descriptions (other algorithms too)
- Graph edges are memory and nodes are compute
  - Not shown are practicalities that we'll care about during the implementation including feature map location and data movement, instruction location and data movement, computation partitioning, computation algorithm selection, ...
  - We'll deal with this in the implementation section
- Graphs are used for training and testing
  - We've already seen this with automatic differentiation with reverse mode accumulation
  - Typically just need to specify the forward graph, the backward graph and weight update graph can be auto generated (with a few other differences between training and testing)



References

List

- References for individual networks are listed on the slide on which they occur

- Mathematics of deep learning
    - https://arxiv.org/pdf/1712.04741.pdf
    - http://www.vision.jhu.edu/tutorials/ICCV17-Tutorial-Math-Deep-Learning-Intro-Rene.pdf
- An introduction to deep learning (lecture 1)
    - http://www.cs.toronto.edu/~ranzato/files/ranzato_deeplearn17_lec1_vision.pdf
- Image classification with deep learning
    - http://www.cs.toronto.edu/~ranzato/files/ranzato_CNN_stanford2015.pdf
- Batch normalization: accelerating deep network training by reducing internal covariate shift
    - https://arxiv.org/abs/1502.03167
- Group normalization
    - https://arxiv.org/abs/1803.08494
- Netscope
    - https://dgschwend.github.io/netscope/quickstart.html