Complex-Network Modelling and Inference

Lecture 15: Modelling with Graphs, and Artificial Neural Networks

Matthew Roughan
<matthew.roughan@adelaide.edu.au>
https://roughan.info/notes/Network_Modelling/

School of Mathematical Sciences,
University of Adelaide

March 7, 2024

Section 1

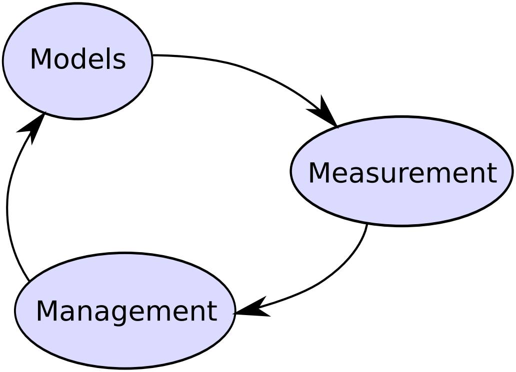
Modelling

Modelling

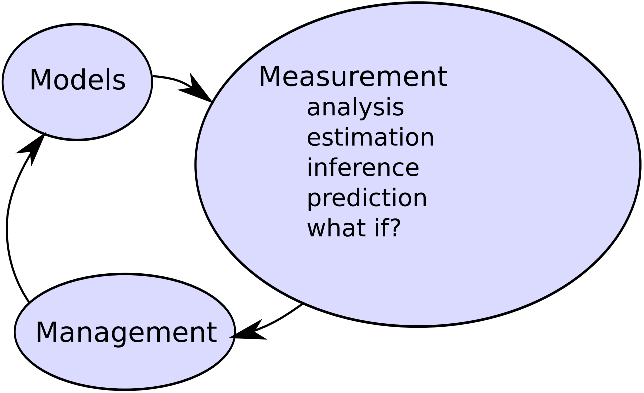
Actually...

Actually I mean Mathematical Modelling

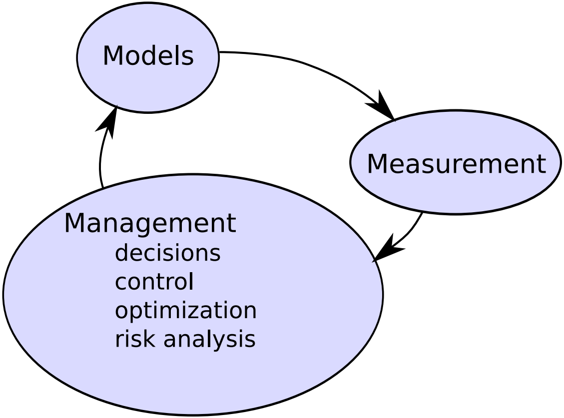
Purpose of our work



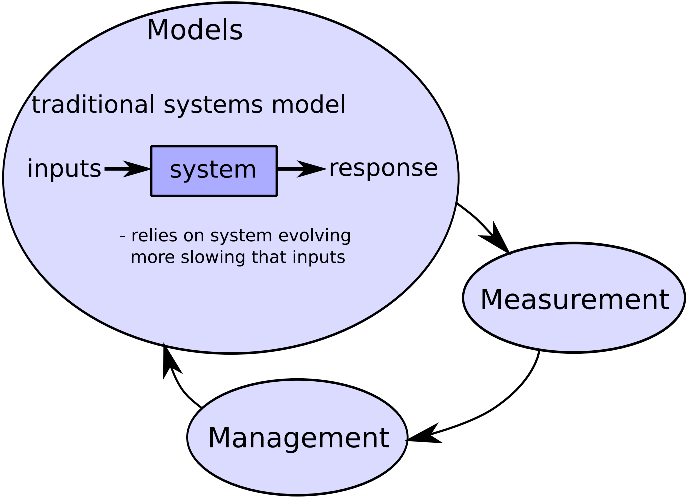
Purpose of our work



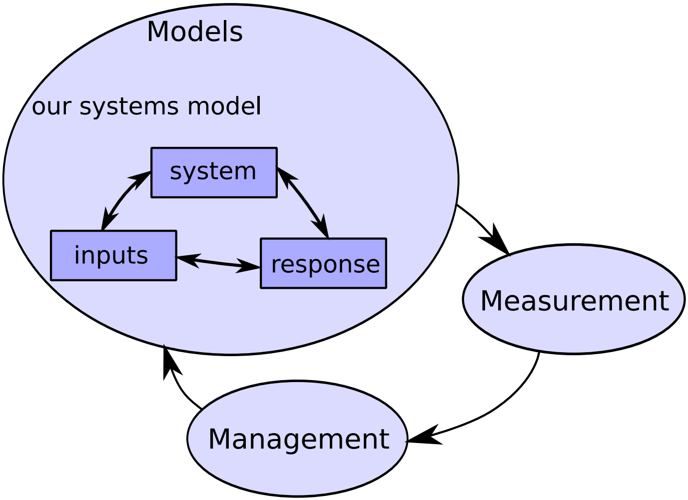
Purpose of our work



Purpose of our work

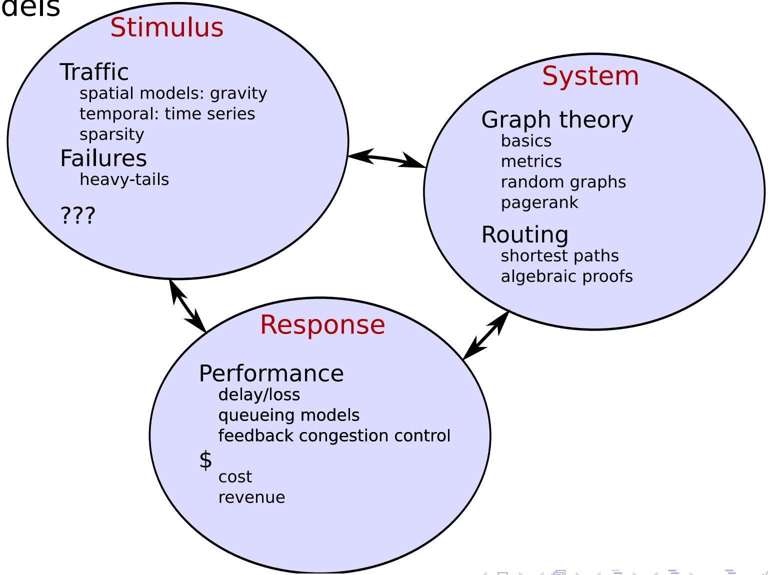


Purpose of our work



Systems model

Models



Why model

- simulations: particularly of higher-level phenomena or algorithms, or to test "what if" scenarios.
- extrapolation: consider what will happen as the network grows.
- sampling: understand the impact of sampling the graph.
- similarity: compare graphs
- visualisation and compression: how can we understand something too big to look at all at once.
- anonymization: we might not be able to publish data due to privacy, but we may be able to publish the model.
- structure: model can teach us about network structure and formation (maybe)
- null-model: a starting point to test hypotheses

Features of a good (random graph) model

- Can generate an ensemble
  - controlled variation
  - needed for statistics
- Captures real constraints and costs
  - otherwise results could be silly
- Prefer operationally meaningful parameters
  - prefer \$5 per mile to $\beta = 3.4$
- Tunability
  - want to be able to control output
  - transparent relationship between inputs and outputs
- Parameters should be measurable/estimatable
- Should generate a network, not just a graph
  - links and nodes have features
    - *e.g.*, link capacity and distance
- "Everything should be made as simple as it can be — but not simpler."
  – Attributed to A. Einstein
- Scalability to large networks

Realism, accuracy and fidelity

- Often "realism" means "we fit our favourite statistics"
  - you have seen a taste of how many stats there are for networks
  - people keep coming up with more
  - if you invent enough, you can always find one that fits
- We favour the idea that
  - it can fit statistics
  - but it must have a physically realistic model as well

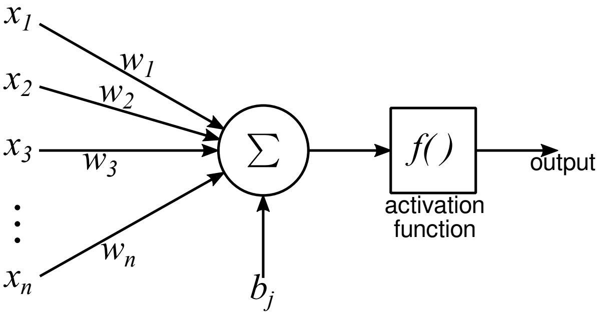
Section 2

Application: Artificial Neural Networks

Neural Networks

- Biological Neural Networks
  - *e.g., c. elegans*
- Artificial Neural Networks (ANNs)
  - used to understand how biological neural networks work
  - used to duplicate our cognitive abilities
    - often under the broader heading of Machine Learning (ML)

Artificial Neurons



- Model a "neuron" as follows:
  - take the sum of weighted (real) inputs x_i, and bias b_j
  - pass it through a *activation* function $f(\cdot)$
- Result

$$y_j = f\left(b_j + \sum_{i=1}^{n} w_i x_i\right)$$

Activation functions

- A common activation function is the *sigmoid*

$$f(x) = \frac{1}{1 + \exp(-x)}$$

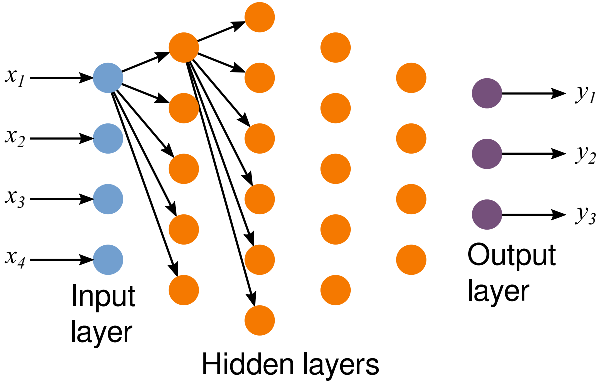
- There are many others:
  - hyperbolic tangent
  - hard threshold (step function)
- Reasons for choices
  - simple
  - differentiable
  - maps to values in $[0, 1]$
  - *non-linear*

Artificial Neural *Network*

- Now put these "neurons" together into a network
- Various organisations
- Commonalities
  - directed
  - usually organised in layers
    - multipartite – typically links between one layer to the next (only)
    - could be links crossing between layers, or backwards
  - highly structured, repeated patterns
  - other variations:
    - acyclic: also called *feed forward*
    - but can be "recurrent", *e.g.*, Hopfield networks

The Multi-layer Perceptron (MLP)

- Perceptron is one of the earliest (1957), and best known networks
  - basically the single neuron above
  - can't do that much by itself
- Multi-layer perceptron: link all in layer i to all in $i + 1$



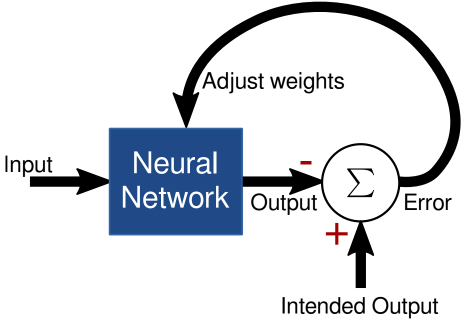
- Convolutional Neural Net (CNN) is similar

Pattern Recognition

- Pattern recognition is a VERY important part of "thinking"
- In essence, we can model pattern recognition as a function mapping a set of inputs, to an output
  - classification: output is the class of the input
- *Universal Approximation Theorem* states (roughly) that a MLP can model any such function with arbitrary precision (given enough nodes)
- Note that if $f(\cdot)$ were linear, then the network would just be a big linear combination, which would be pretty boring, but some linear pieces are OK
- *Deep learning* is called this because lots of layers (deep)

Training an ANN

- Training is just
  - set of inputs and desired outputs
    - think of these as defining a function $\mathbb{R}^n \rightarrow \mathbb{R}$
  - optimising the weights on each link such that the neural network has the best possible approximation to the training function
  - often done by *reinforcement learning*, *i.e.*, repeated iteration



Training an ANN

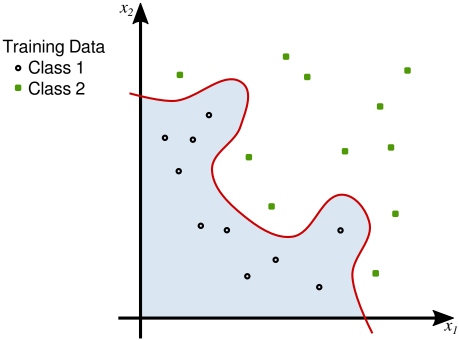
- Training is just
  - set of inputs and desired outputs
    - think of these as defining a function $\mathbb{R}^n \to \mathbb{R}$
  - optimising the weights on each link such that the neural network has the best possible approximation to the training function
  - often done by *reinforcement learning*, *i.e.*, repeated iteration
- This is harder to do than it sounds
  - you could need LOTS of training data to define the function with enough resolution
  - noise in training data
  - there are LOTS of weights forming your optimisation variable, hence computational cost
  - no guarantee (in general) that it will be a "nice" optimisation problem, *e.g.*, a convex program
  - over-fitting
- Perceptron training relatively straightforward
  - back-propagation
  - but not that relevant to this course, so I leave it to you to find out more

What can we say about these networks?

- By default, they tend to be dense
  - can improve performance sometimes by "sparsifying", *i.e.*, removing links that don't contribute a lot
  - use locality in the input to remove some edges
- Substructure
  - often some locality (neurons relate to a position in space, and more connections between close nodes)
  - some layers are *convolutional* – convolve outputs of a group from prior layer
  - some layers are *pooling* – pool outputs of a larger group
- They have very repetitive structures
  - more on this next week

Simple Example

Classification is a very common ML task

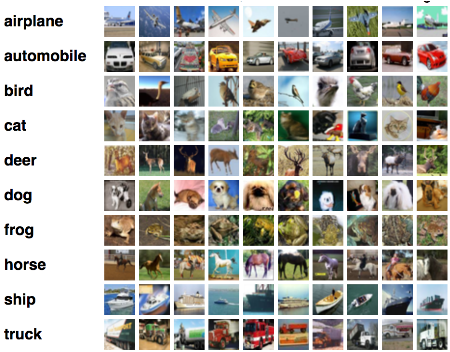


Function to learn

$$g(x) = \begin{cases} 1, & \text{inside shaded region} \\ 2, & \text{outside shaded region} \end{cases}$$

Slightly Harder Example

Input



But images are complex, so we construct a large set of *features* (*i.e.*, numbers) to describe each image. Perhaps as detailed as the value of each pixel.

"Deep Learning" Examples

Deep learning uses many-layered ANNs

- Google Neural Machine Translation
  https://ai.googleblog.com/2016/09/a-neural-network-for-machine.html
  https://arxiv.org/abs/1609.08144
  8 encoder, and 8 decoder layers, 36 M sentences used in training, 6 days of training using 96 Nvidia GPUs
- DeepMind WaveNet, synthesizes speech
  https://deepmind.com/blog/wavenet-generative-model-raw-audio/
  https://arxiv.org/pdf/1609.03499.pdf
  8 layers, 44 hours speech in training
- Google Quickdraw
  https://quickdraw.withgoogle.com/, recongnises sketches
  50 million drawings for training

Do we learn about our brain from ANNs?

- ANNs work, but they are very like "black boxes" in that they don't explain why they work
- Artificial neurons intended to model real neurons
    - dendrites bring input signals from other neurons
    - soma acts to sum inputs
    - axon sends pulse when activated

  But real neurons "fire" in discrete pulses
- Networks in ANNs are often
    - very regular
    - size
        - our brain has 86 billion neurons
        - biggest "deep learners" 160 billion *parameter*
        - even our gut has $\sim$ 500 million neurons
    - our brain does other things, *e.g.*, conciousness

Further reading I