Artificial neural networks

Steven Kerr

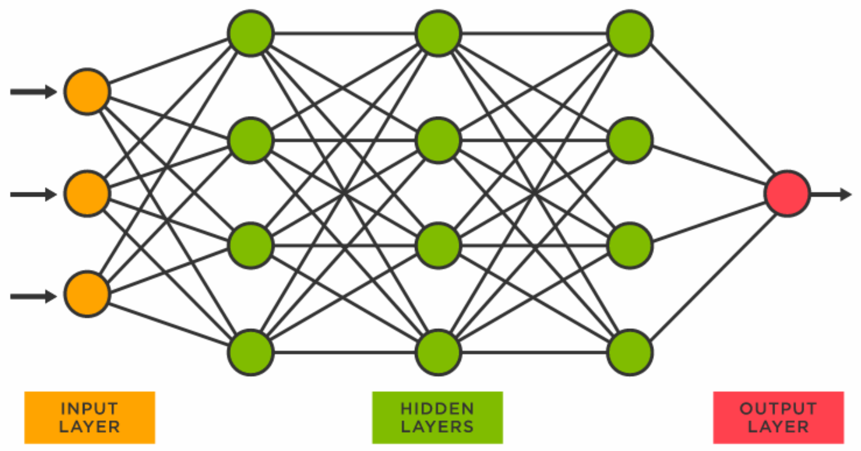
Supported by

What is an Artificial neural network (ANN)?

- A statistical/machine learning model inspired by biological neural networks.
- Consists of a network of interconnected artificial neurons.
- Each connection between two neurons has a weight, which determines how strongly they influence each other.
- The neuron itself is just a little ‘machine’ that takes inputs from multiple other neurons it is connected to, and produces a single output.
- That output is then fed into other neurons.
- In this way you can chain neurons together in arbitrarily complex networks.

What is an Artificial neural network?

- ANNs are typically organised into layers: input, hidden and output.
- Below is an example of a *feedforward* ANN – data flows forwards only, left to right.
- It is also *fully connected* – each neuron is connected to every neuron in the next layer.



Artificial Neural Networks

- It is possible to show that an ANN with sufficiently many neurons can be used to approximate a huge class of functions (Universal approximation theorem).
- What is a human mind except a very complicated function?
- For example, when a human plays chess, they take some input (the board position) and produce an output (the next move).
- ANNs hold the promise of (super) human-like intelligence.

Artificial Neural Networks

- AlphaZero, neural network based algorithm released by DeepMind in 2018.

- It is the strongest Chess, Shogi and Go playing entity that has ever existed.

- Many thought that machines would never be able to play Go better than humans.

Artificial Neural Networks

- In November 2019, Lee Sedol, one of the strongest Go players in the world, retired after losing a series of matches against AlphaGo Zero (predecessor of AlphaZero) earlier that year.
- ANNs are doing impressive things in many other areas (voice/text/image recognition, medical diagnosis, others).
- Very hot area for research and innovation.

Artificial Neural Networks

- ANNs can be huge. State of the art *foundation models* currently have *trillions* of parameters.

- Nice article: https://www.economist.com/interactive/briefing/2022/06/11/huge-foundation-models-are-turbo-charging-ai-progress.

- This requires specialised hardware and software.

Artificial Neural Networks

- Software: TensorFlow, Keras, Torch.

- ANNs are trained using techniques called *gradient descent* and *backpropagation*, which are very efficient and parallelisable.

- Hardware:
  - Graphical processing units (GPUs)
  - Tensor processing units (TPUs).

- Enables massive parallelisation.

Multilayer perceptrons

- Multilayer perceptrons are fully connected, feedforward ANNs.
- In sparklyr, use the following command to fit a multilayer perceptron classifier:

  ```
  ml_multilayer_perceptron_classifier(data, formula, layers)
  ```

- The layers argument specifies the number of neurons in each layer.

Important caveats!

- *Caveat 1*: Sparklyr implements a multilayer perceptron classifier only - the variable you're predicting must be categorical.
- *Caveat 2*: The labels of the categorical variable you're predicting need to be numbers in order, starting at zero.
- *Caveat 3*: By contrast, categorical variables included as predictors must be one-hot encoded.
- *Caveat 4*: The number of input and output layers in the layers argument must be correct, otherwise your code will fail.
- *Caveat 5*: If you choose to train a multilayer perceptron in your project, inputs variables that are continuous should be normalised.

Correct number of input/output layers

- ml_multilayer_perceptron_classifier(data, formula, layers)

- The layers argument is a vector that gives the number of neurons in each layer, starting left to right from input, then hidden layers, and finally output layer.

- The number of neurons in the input layer should be equal to the total number of features in the predictor variables. Remember that categorical variables used as predictors must be one-hot encoded!

- The number of neurons in the output layer should be equal to the number of classes in the variable that is being predicted.

Normalising continuous inputs

- The results of the training can depend strongly on the units in which continuous input quantities are measured.
- For example, say I wish to predict whether an individual is unemployed using age, height, sex, etc.
- Height could be measured in metres, feet, centimetres, nanometres...

Normalising continuous inputs

- We ‘level the playing field’ before we start. Most common way of doing this is ‘z-scoring’.

- For each numeric variable, subtract mean and divide by standard deviation,

$$z = \frac{x - \mu}{\sigma}$$