PUG
Pune User Group

Unveiling the Neural Nets

Sri Harsha G

Data scientist by profession...Teacher by passion...

sriharsha@cyberigence.com

#PuneDevCon

Outline

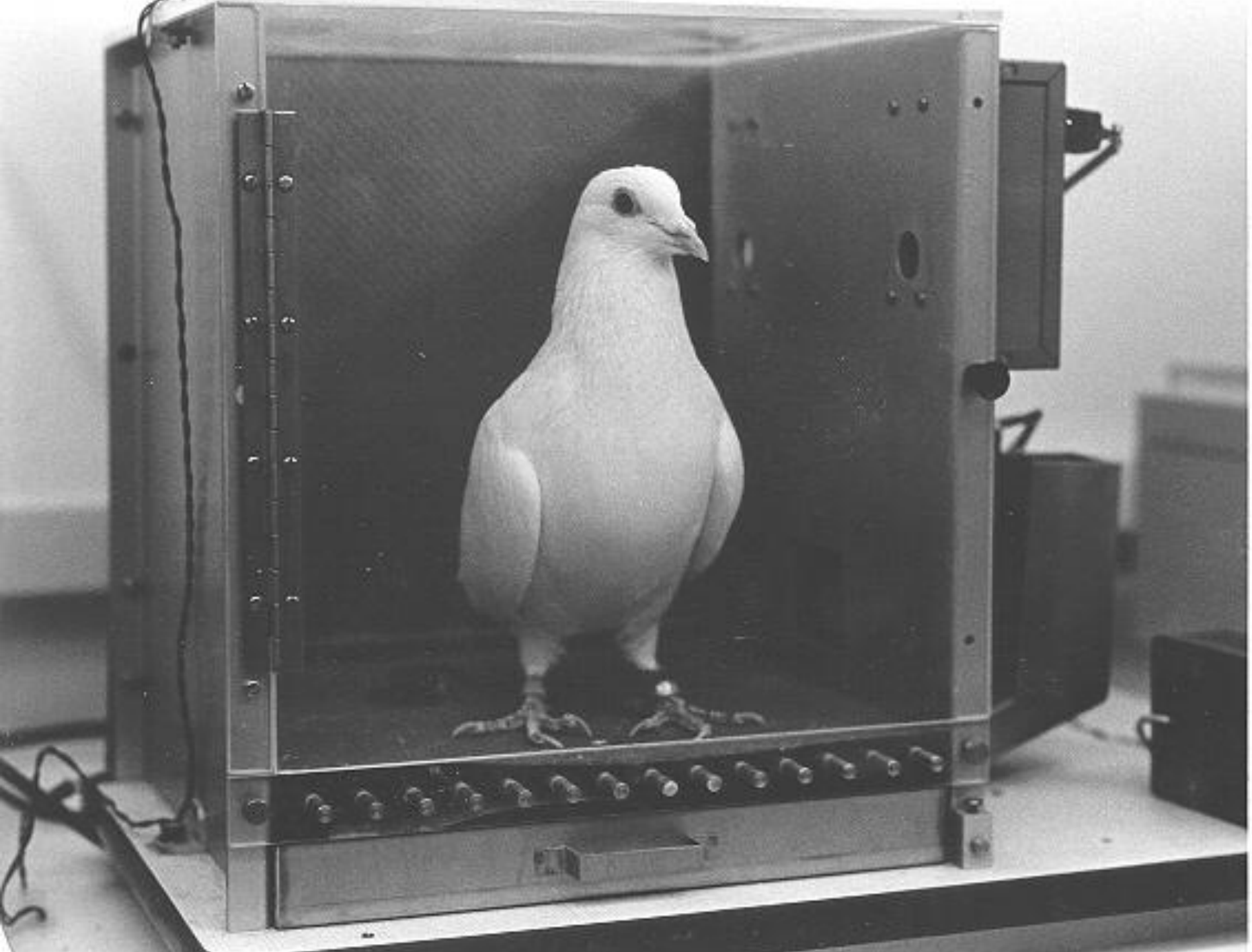
- What are Neural Networks?
- Biological Neural Networks
- ANN – The basics
- Feed forward net
- Training
- Example – Voice recognition
- Applications – Feedforward nets
- Recurrency
- ConvNets
- Conclusion

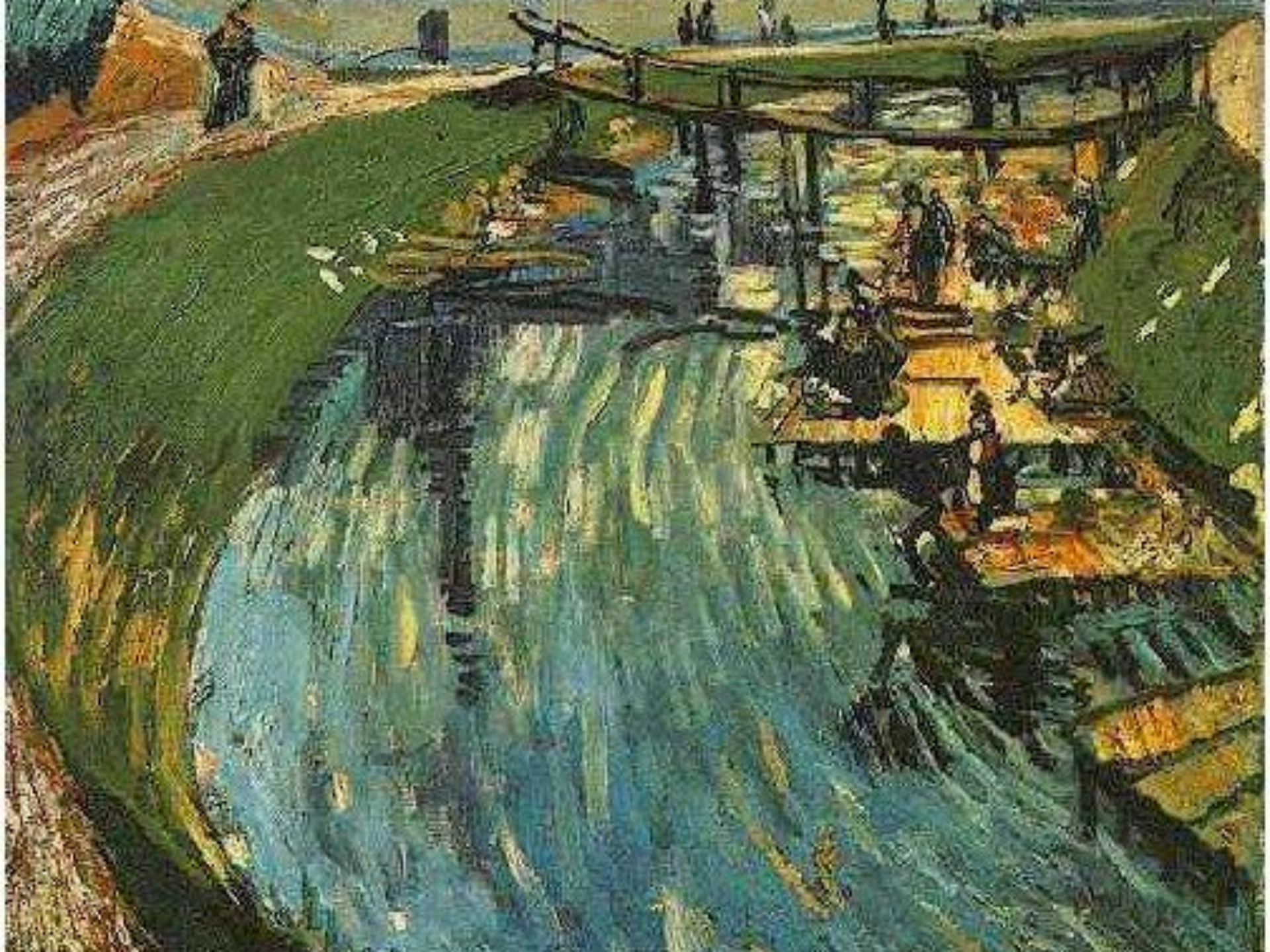
What are Neural Networks?

- Models of the brain and nervous system
- Highly parallel
  - Process information much more like the brain than a serial computer
- Learning
- Very simple principles
- Very complex behaviours
- Applications
  - As powerful problem solvers
  - As biological models

Biological Neural Nets

- Pigeons as art experts (Watanabe *et al.* 1995)
  - Experiment:
    - Pigeon in Skinner box
    - Present paintings of two different artists (e.g. Chagall / Van Gogh)
    - Reward for pecking when presented a particular artist (e.g. Van Gogh)





- Pigeons were able to discriminate between Van Gogh and Chagall with 95% accuracy (when presented with pictures they had been trained on)

- Discrimination still 85% successful for previously unseen paintings of the artists

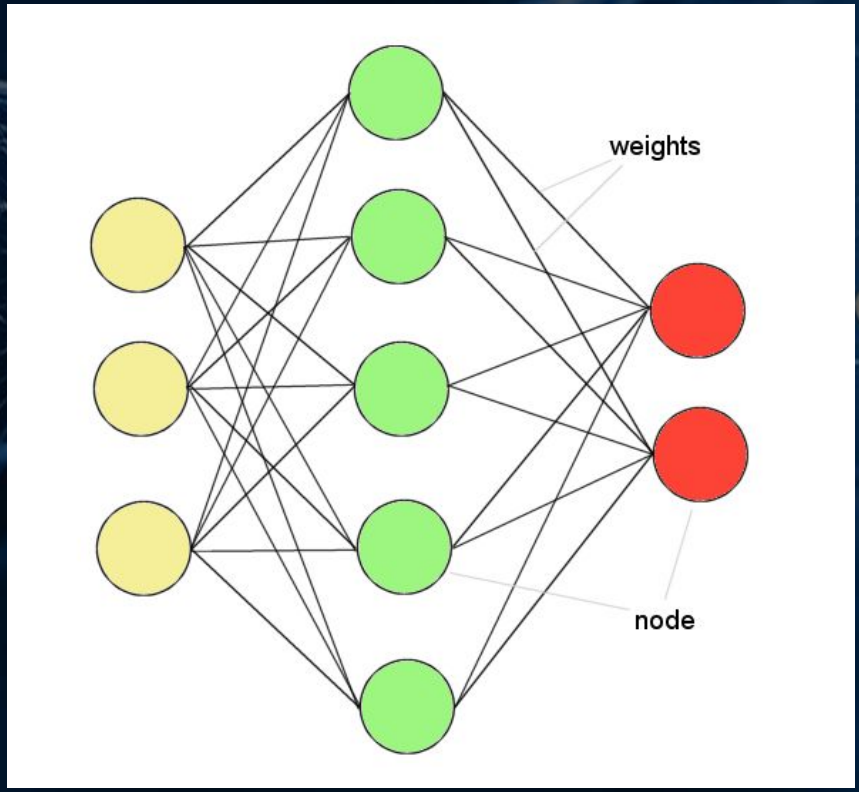
- Pigeons do not simply memorise the pictures
- They can extract and recognise patterns (the ‘style’)
- They generalise from the already seen to make predictions

- This is what neural networks (biological and artificial) are good at (unlike conventional computer)

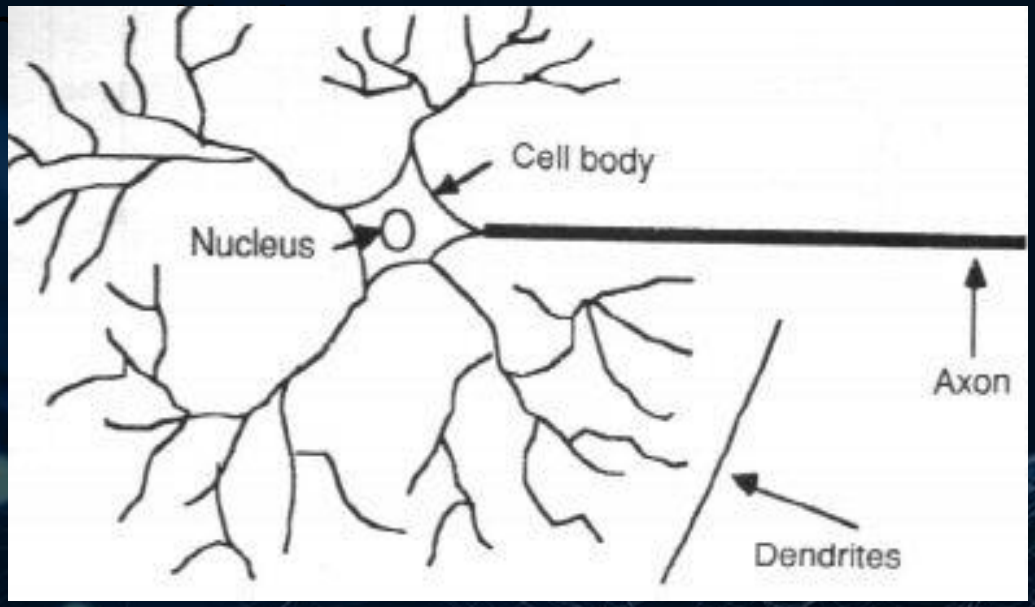
ANNs – The basics

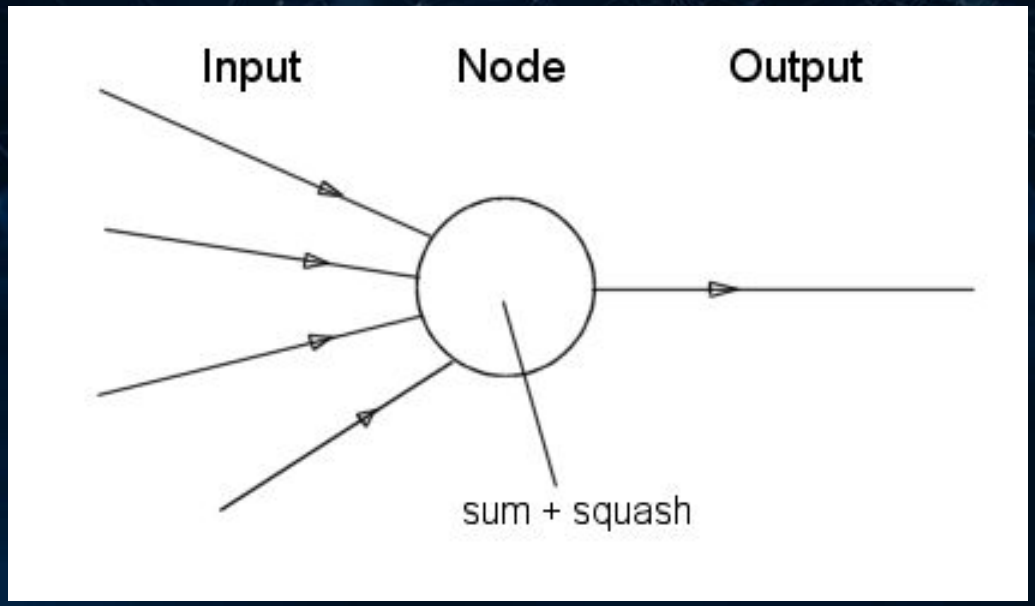
ANNs incorporate the two fundamental components of biological neural nets:

1. Neurons (nodes)
2. Synapses (weights)



- Neuron

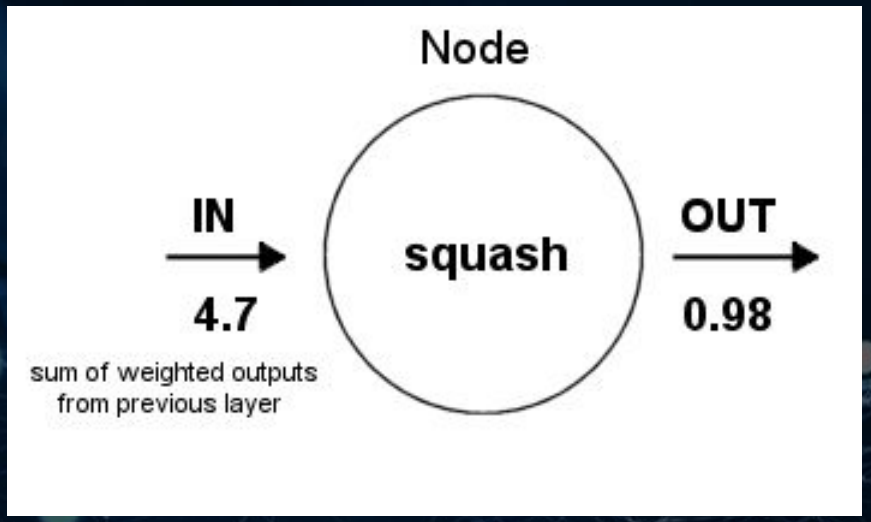




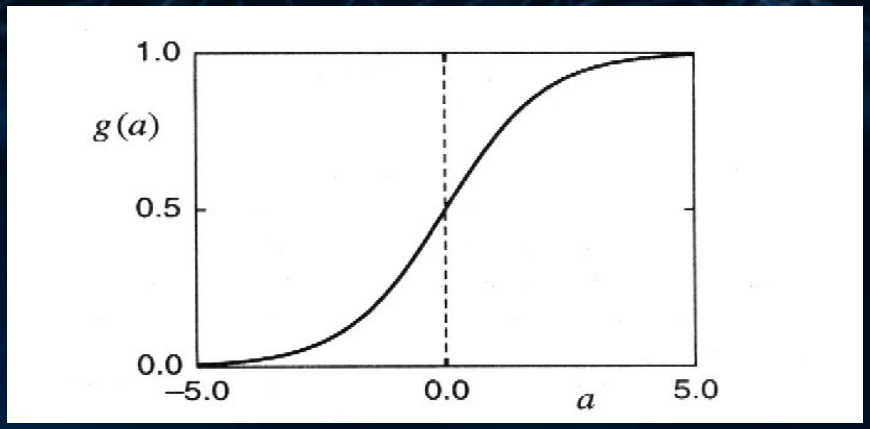
PUG
Pune User Group

Let’s look at a Neuron in detail

Structure of a node:



Squashing function limits node output:



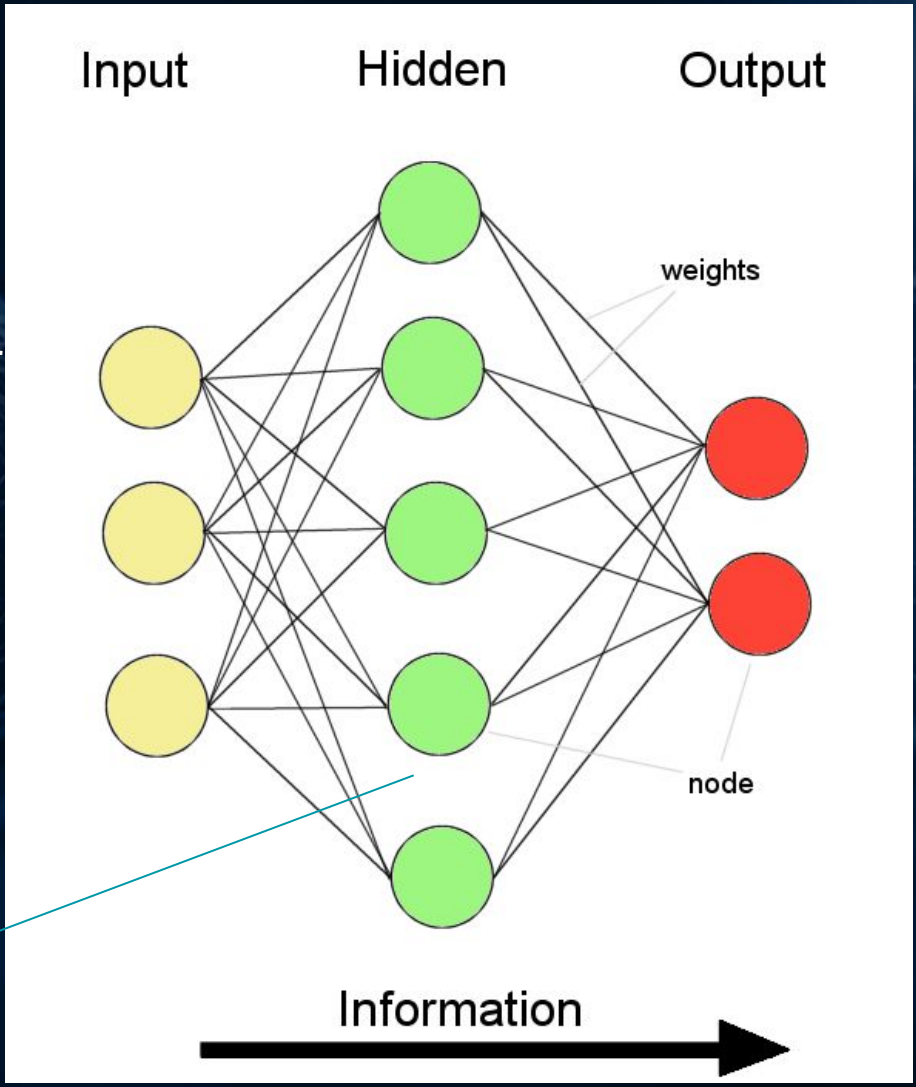
PUG
Pune User Group

Let’s look at some activation functions...

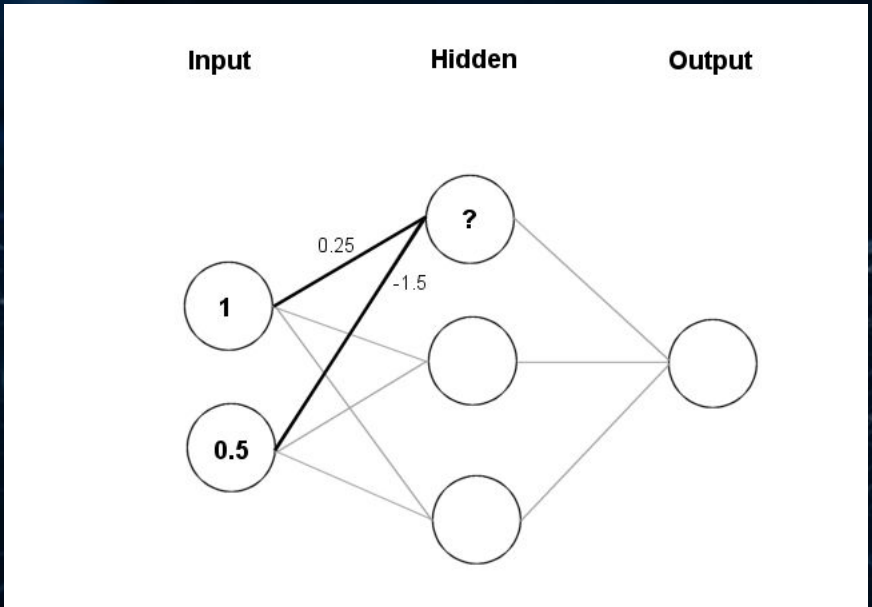
Feed-forward nets

- Information flow is unidirectional
  - Data is presented to *Input layer*
  - Passed on to *Hidden Layer*
  - Passed on to *Output layer*
- Information is distributed
- Information processing is parallel

Internal representation (interpretation) of data



Feeding data through the net:



$(1 \times 0.25) + (0.5 \times (-1.5)) = 0.25 + (-0.75) = -\mathbf{0.5}$

PUG
Pune User Group

What is hidden inside a neuron?

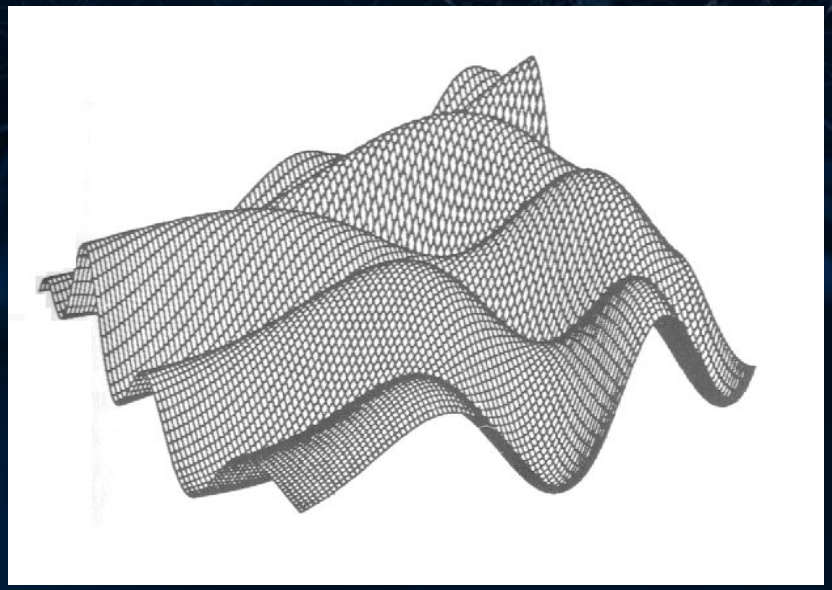
PUG
Pune User Group

How can we find the right weights?

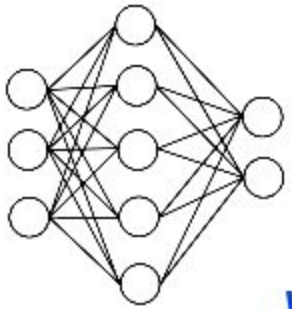
Training the Network - Learning

- Backpropagation
  - Requires training set (input / output pairs)
  - Starts with small random weights
  - Error is used to adjust weights (supervised learning)

Gradient descent on error landscape



1.

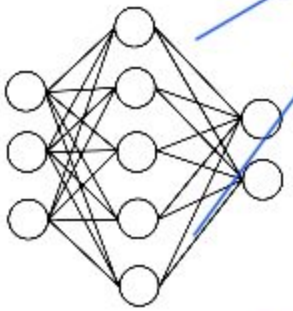


Wallace

Wallace - Darwin (calculate error)

adjust weights

2.

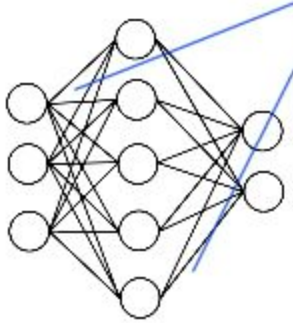


Wallace

Wallace - Darwin (calculate error)

adjust weights

3.



Darwin

- **Advantages**
  - It works!
  - Relatively fast

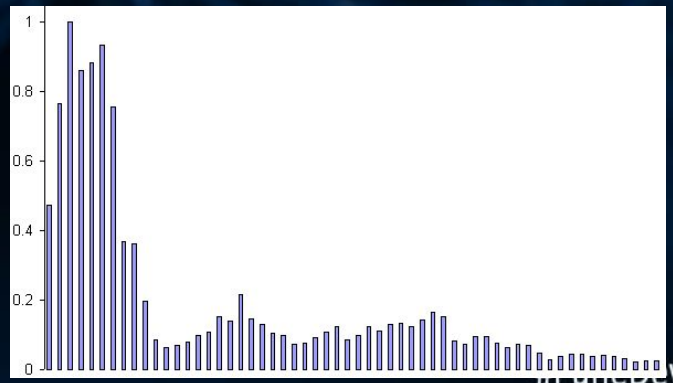
- **Downsides**
  - Requires a training set
  - Can be slow
  - Probably not biologically realistic

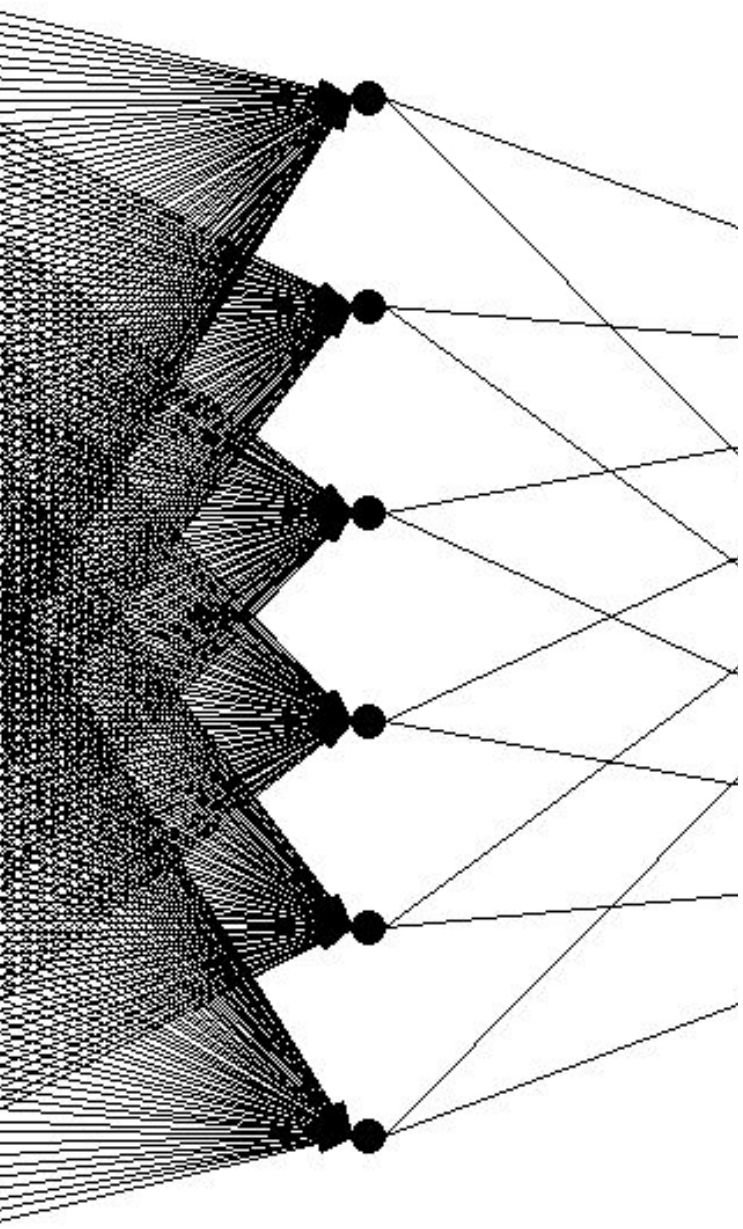
- **Alternatives to Backpropagation**
  - Hebbian learning
    - Not successful in feed-forward nets
  - Reinforcement learning
    - Only limited success
  - Artificial evolution
    - More general, but can be even slower than backprop

Example: Voice Recognition

- Task: Learn to discriminate between two different voices saying “Hello”

- Data
  - Sources
    - Steve Simpson
    - David Raubenheimer
  - Format
    - Frequency distribution (60 bins)
    - Analogy: cochlea

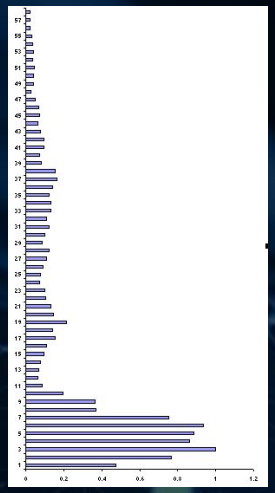


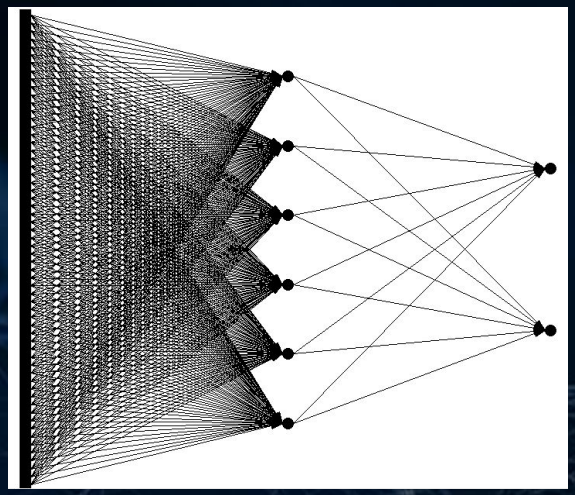


- Network architecture
  - Feed forward network
    - 60 input (one for each frequency bin)
    - 6 hidden
    - 2 output (0-1 for “Steve”, 1-0 for “David”)

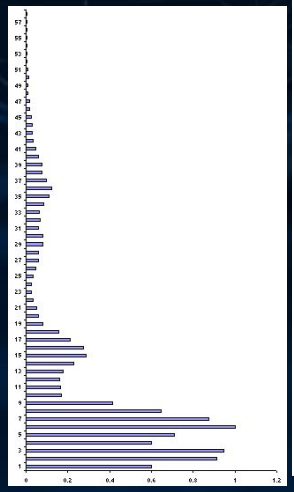
- Presenting the data

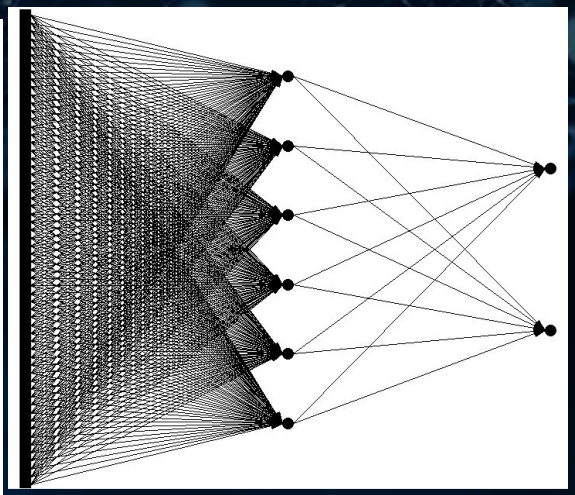
Steve





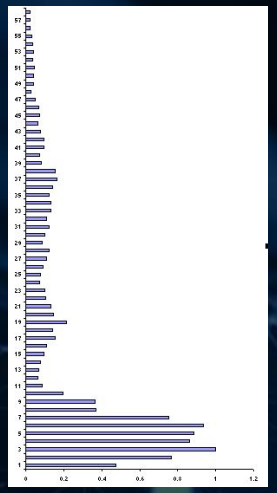
David

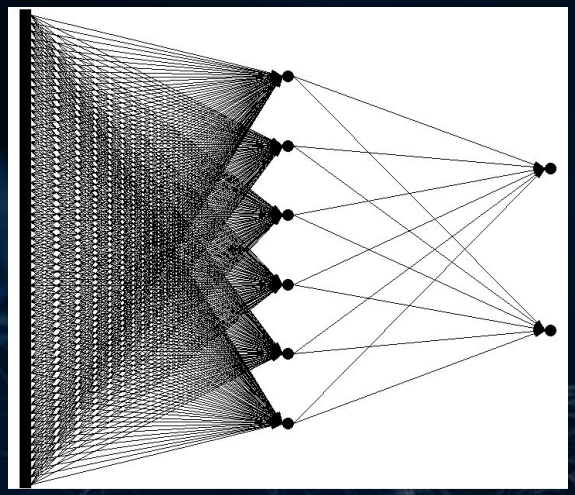




- Presenting the data (untrained network)

Steve

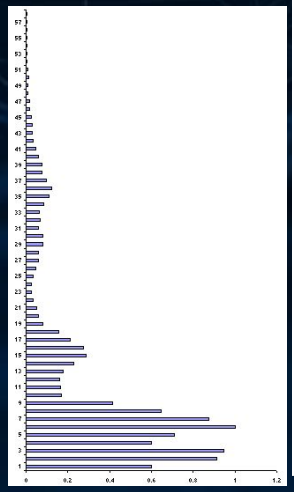


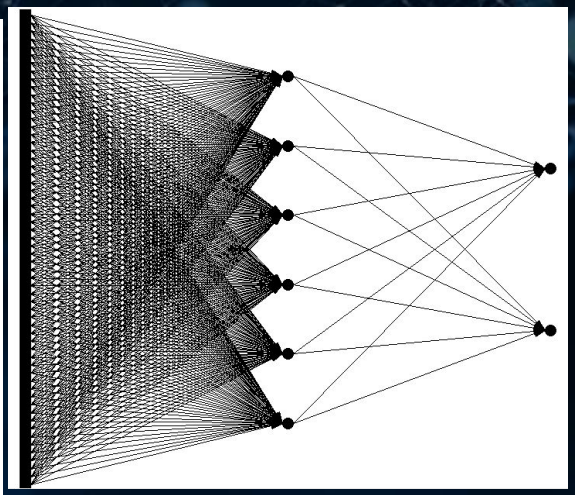


0.43

0.26

David



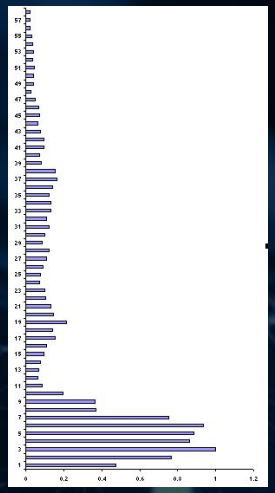


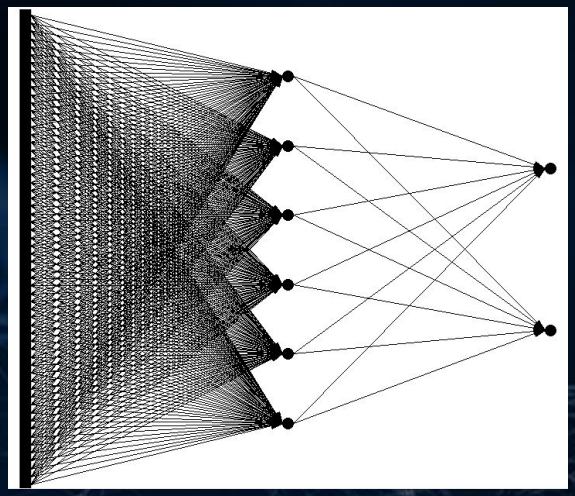
0.73

0.55

- Calculate error

Steve

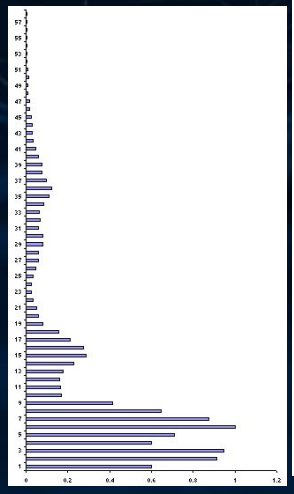


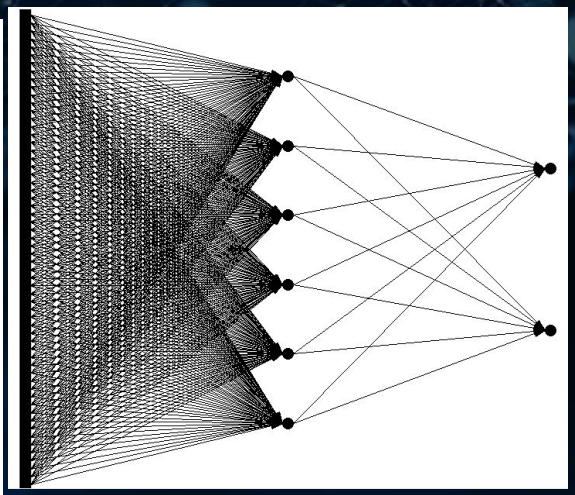


0.43 – 0 = 0.43

0.26 –1 = 0.74

David



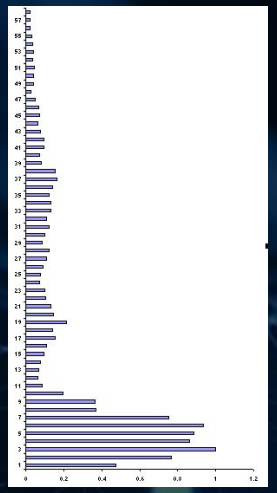


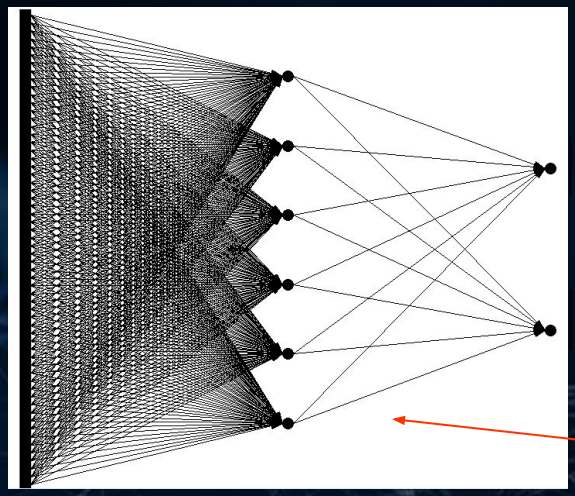
0.73 – 1 = 0.27

0.55 – 0 = 0.55

Backprop error and adjust weights

Steve



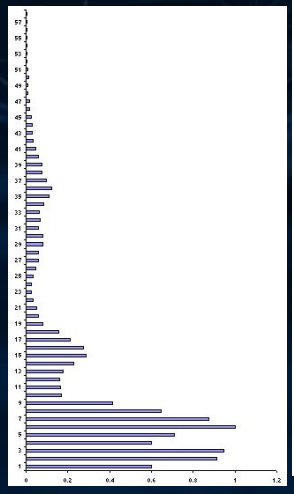


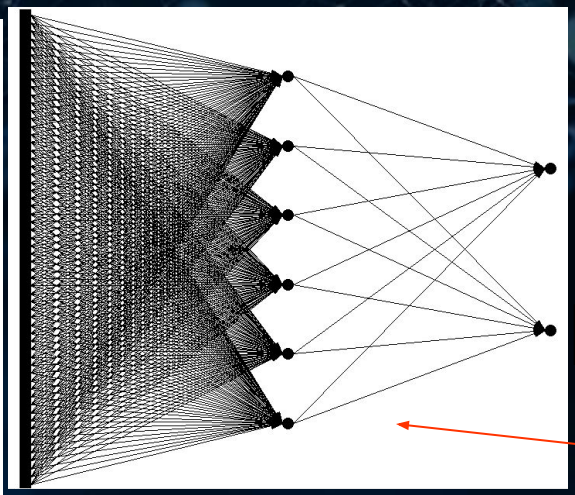
$$|0.43 - 0| = 0.43$$

$$|0.26 - 1| = 0.74$$

$$1.17$$

David





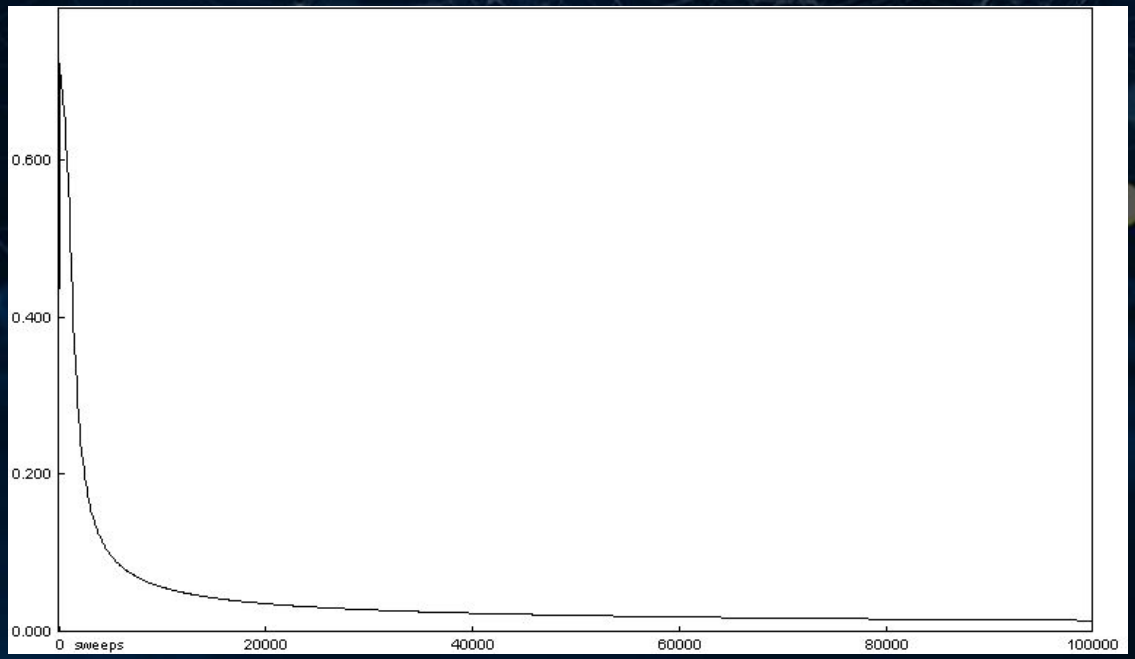
$$|0.73 - 1| = 0.27$$

$$|0.55 - 0| = 0.55$$

$$0.82$$

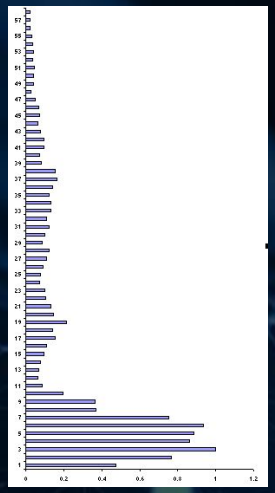
- Repeat process (sweep) for all training pairs
  - Present data
  - Calculate error
  - Backpropagate error
  - Adjust weights

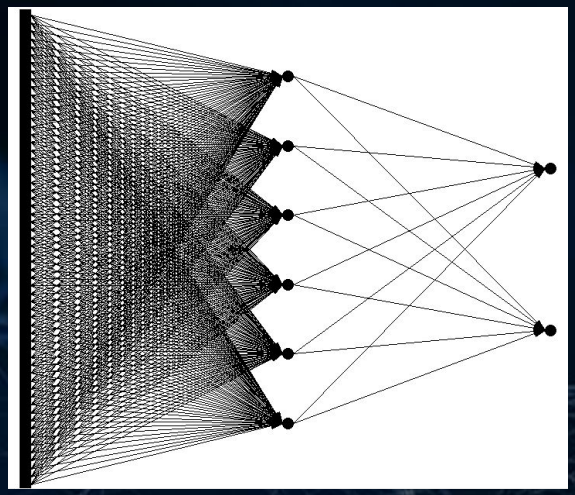
- Repeat process multiple times



- Presenting the data (trained network)

Steve

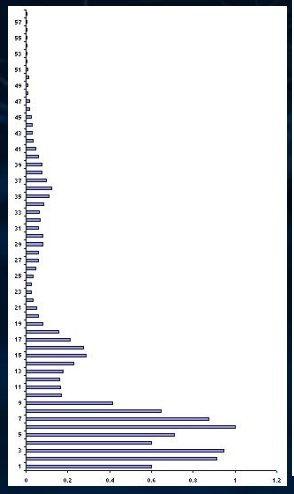


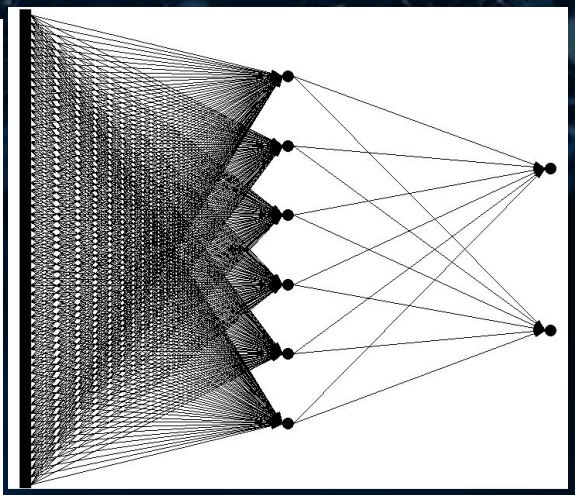


0.01

0.99

David



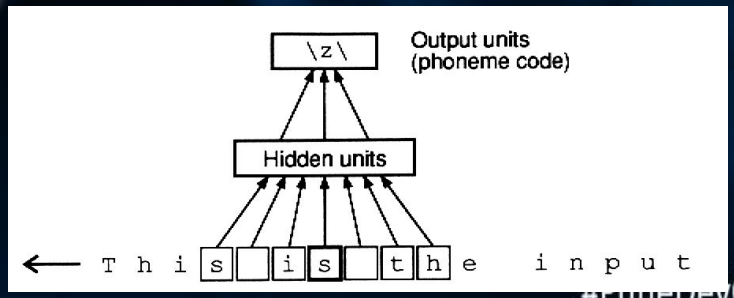


0.99

0.01

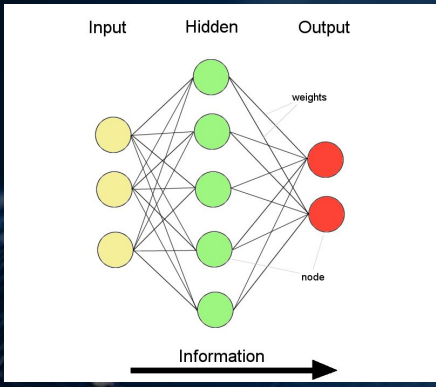
Applications of Feed-forward nets

- Pattern recognition
  - Character recognition
  - Face Recognition
- Sonar mine/rock recognition (Gorman & Sejnowksi, 1988)
- Navigation of a car (Pomerleau, 1989)
- Stock-market prediction
- Pronunciation (NETtalk)



Recurrent Networks

- Feed forward networks:
  - Information only flows one way
  - One input pattern produces one output
  - No sense of time (or memory of previous state)



- Recurrency
  - Nodes connect back to other nodes or themselves
  - Information flow is multidirectional
  - Sense of time and memory of previous state(s)

- Biological nervous systems show high levels of recurrency (but feed-forward structures exists too)

ConvNets

Motivation—Image Data

PUG
Pune User Group

- So far, the structure of our neural network treats all inputs interchangeably.
- No relationships between the individual inputs
- Just an ordered set of variables
- We want to incorporate domain knowledge into the architecture of a Neural Network.

Motivation

Image data has important structures, such as;

- ”Topology” of pixels
- Translation invariance
- Issues of lighting and contrast
- Knowledge of human visual system
- Nearby pixels tend to have similar values
- Edges and shapes
- Scale Invariance—objects may appear at different sizes in the image.

Motivation—Image Data

PUG
Pune User Group

- Fully connected would require a vast number of parameters
- MNIST images are small (32 x 32 pixels) and in grayscale
- Color images are more typically at least (200 x 200) pixels x 3 color channels (RGB) = 120,000 values.
- A single fully connected layer would require (200x200x3)2 = 14,400,000,000 weights!
- Variance (in terms of bias-variance) would be too high
- So we introduce “bias” by structuring the network to look for certain kinds of patterns

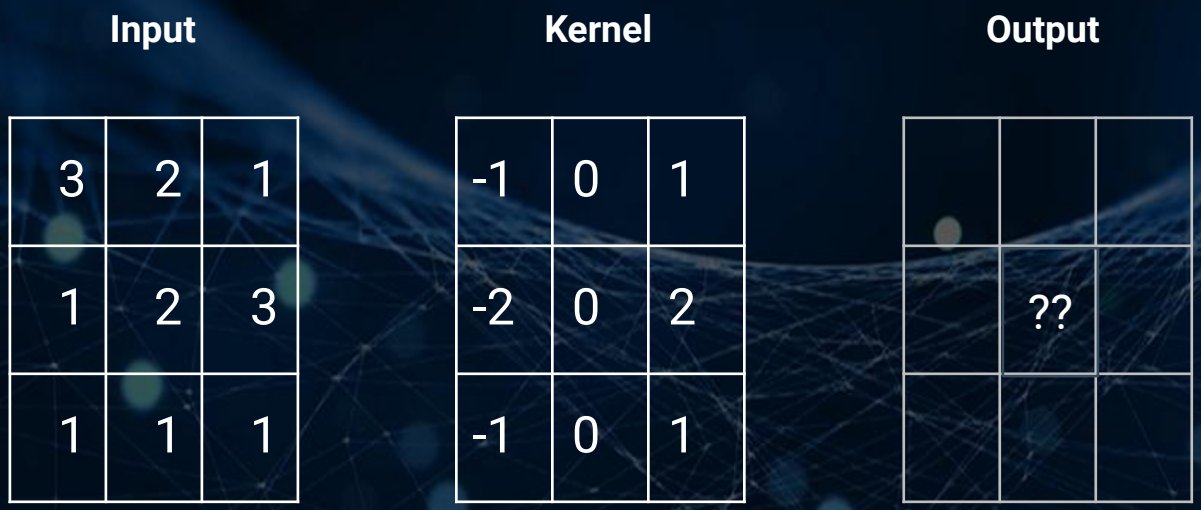
Motivation

- Features need to be “built up”
- Edges -> shapes -> relations between shapes
- Textures
- Cat = two eyes in certain relation to one another + cat fur texture.
- Eyes = dark circle (pupil) inside another circle.
- Circle = particular combination of edge detectors.
- Fur = edges in certain pattern.

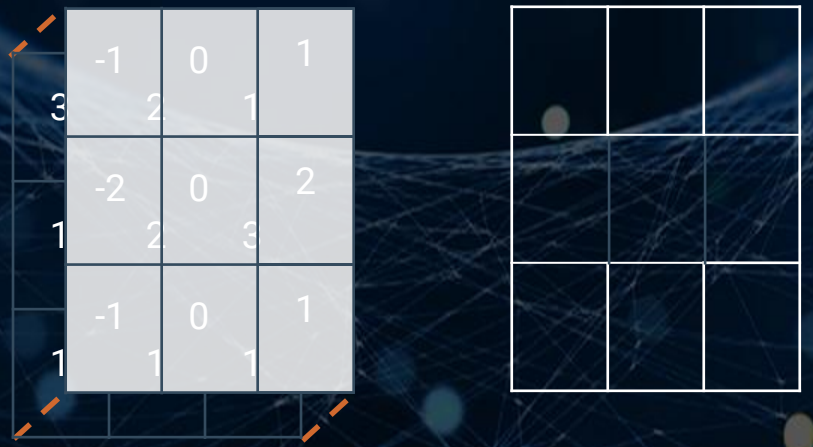
Kernels

- A *kernel* is a grid of weights “overlaid” on image, centered on one pixel
- Each weight multiplied with pixel underneath it
- Output over the centered pixel is $\sum_{p=1}^{P} W_p \cdot pixel_p$
- Used for traditional image processing techniques:
  - Blur
  - Sharpen
  - Edge detection
  - Emboss

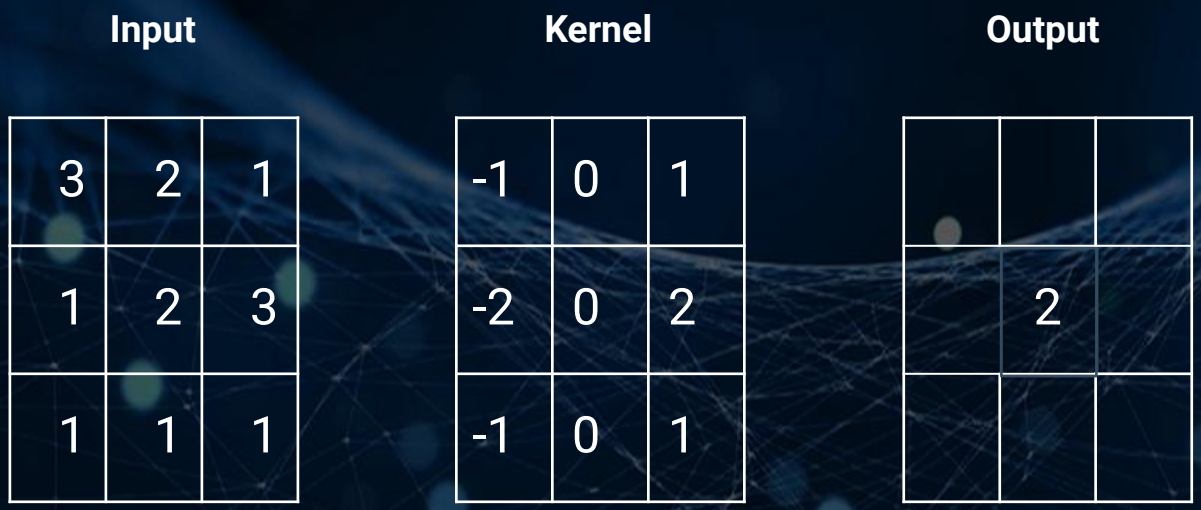
Kernel: 3x3 Example



Kernel: 3x3 Example



Kernel: 3x3 Example



= −3 + 1 − 2 + 6 − 1 + 1 = 2

Kernels as Feature Detectors

PUG
Pune User Group

Can think of kernels as a ”local feature detectors”

Vertical Line Detector

-1	1	-1
-1	1	-1
-1	1	-1

Horizontal Line Detector

-1	-1	-1
1	1	1
-1	-1	-1

Corner Detector

-1	-1	-1
-1	1	1
-1	1	1

Convolutional Neural Nets

PUG
Pune User Group

Primary Ideas behind Convolutional Neural Networks:

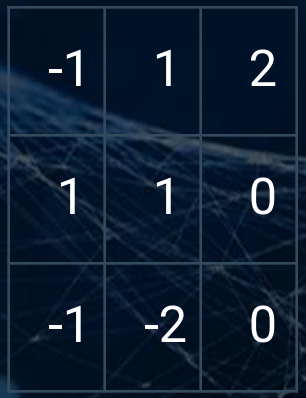
- Let the Neural Network learn which kernels are most useful
- Use same set of kernels across entire image (translation invariance)
- Reduces number of parameters and “variance” (from bias-variance point of view)

Padding

- Using Kernels directly, there will be an “edge effect”
- Pixels near the edge will not be used as “center pixels” since there are not enough surrounding pixels
- Padding adds extra pixels around the frame
- So every pixel of the original image will be a center pixel as the kernel moves across the image
- Added pixels are typically of value zero (zero-padding)

Without Padding

Input

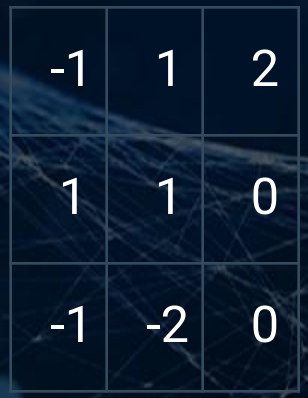


Kernel

Output

With Padding

Input



Kernel

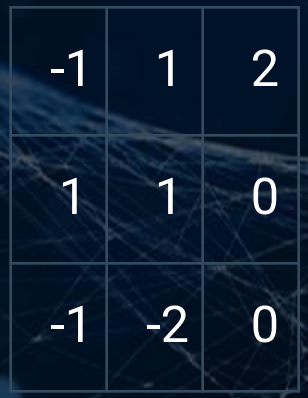
Output

Stride

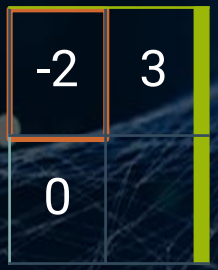
- The ”step size” as the kernel moves across the image
- Can be different for vertical and horizontal steps (but usually is the same value)
- When stride is greater than 1, it scales down the output dimension

Stride 2 Example—No Padding

Input



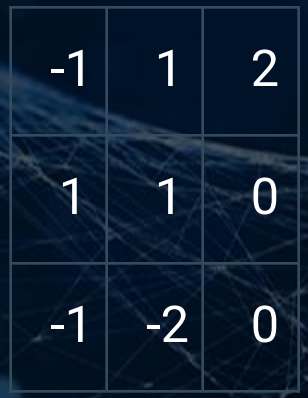
Kernel



Output

Stride 2 Example—with Padding

Input



Kernel

Output

Depth

- In images, we often have multiple numbers associated with each pixel location.
- These numbers are referred to as “channels”
  - RGB image—3 channels
  - CMYK—4 channels
- The number of channels is referred to as the “depth”
- So the kernel itself will have a “depth” the same size as the number of input channels
- Example: a 5x5 kernel on an RGB image
  - There will be 5x5x3 = 75 weights

Depth

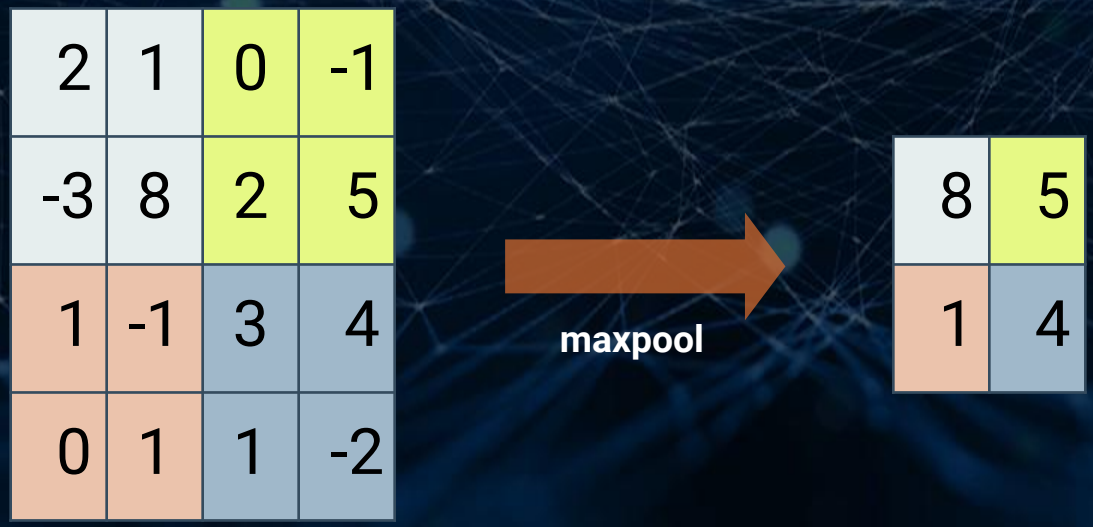
- The output from the layer will also have a depth
- The networks typically train many different kernels
- Each kernel outputs a single number at each pixel location
- So if there are 10 kernels in a layer, the output of that layer will have depth 10.

Pooling

- Idea: Reduce the image size by mapping a patch of pixels to a single value.
- Shrinks the dimensions of the image.
- Does not have parameters, though there are different types of pooling operations.

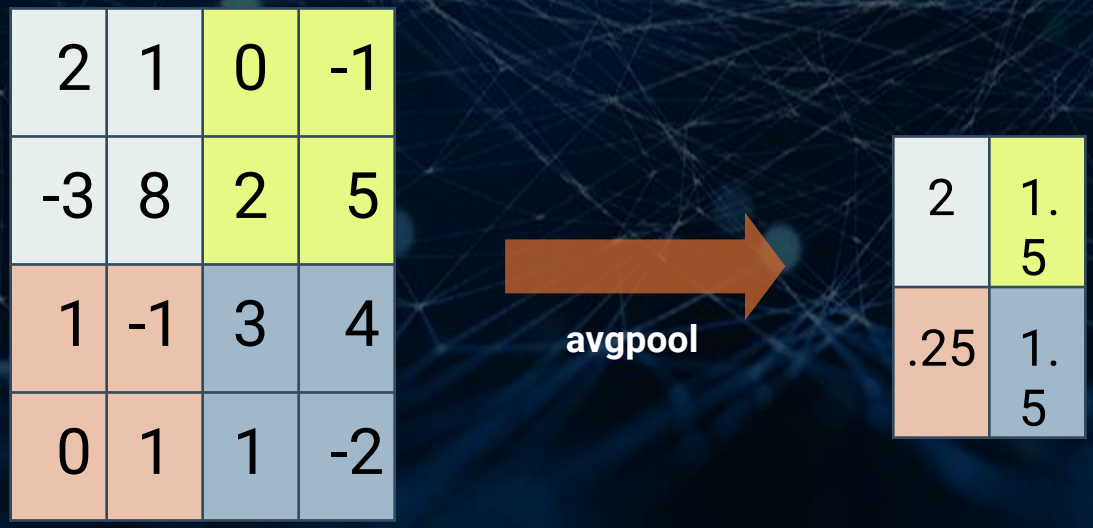
Pooling: Max-pool

- For each distinct patch, represent it by the maximum
- 2x2 maxpool shown below



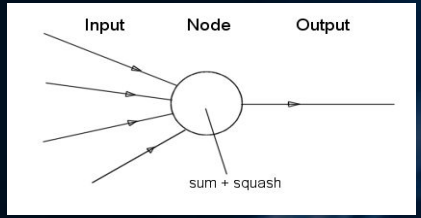
Pooling: Average-pool

- For each distinct patch, represent it by the average
- 2x2 avgpool shown below

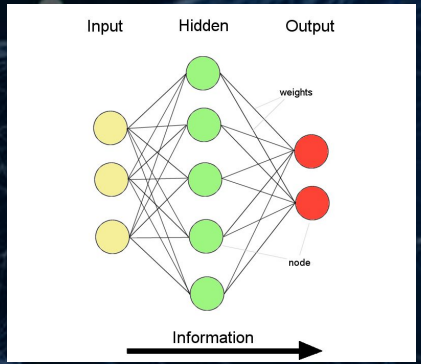


Recap – Neural Networks

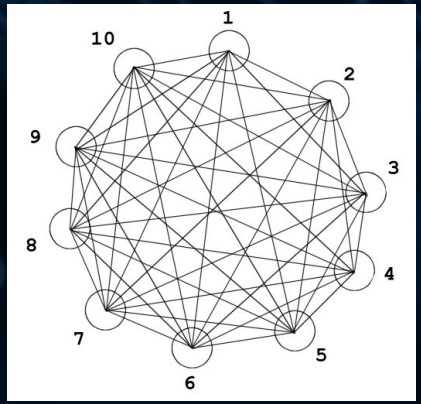
- Components – biological plausibility
  - Neuron / node
  - Synapse / weight



- Feed forward networks
  - Unidirectional flow of information
  - Good at extracting patterns, generalisation and prediction
  - Distributed representation of data
  - Parallel processing of data
  - Training: Backpropagation
  - Not exact models, but good at demonstrating principles



- Recurrent networks
  - Multidirectional flow of information
  - Memory / sense of time
  - Complex temporal dynamics (e.g. CPGs)
  - Various training methods (Hebbian, evolution)
  - Often better biological models than FFNs



Sri Harsha Gajavalli

Founder @ICyberSol | Lead @socaity | Startup Tech Mentor/AI consultant | Researcher, AI in CyberSec

PUG Pune User Group

- Intel Software Innovator
- Mentor of Change, Atal Innovation Mission
- Community Dev Lead, Google ; GDE [soon]
- Tech Visionary award winner, InterCon
- Google Summer of Code Mentor, OWASP [2018]
- National Entrepreneurship Award, GoI [2018]
- Fb Open Source Mentor [2018]
- Top 100, Fb Bug Hunter [2017]

Reach me:

@Email: sriharsha.g15@gmail.com

@Github: https://github.com/SriHarshaGajavalli

@LinkedIn: https://www.linkedin.com/in/sriharshagajavalli

@Facebook: https://facebook.com/harsha.gajavalli

@Twitter: @Sri_HarshaG