Common AI Algorithms

(SENG 463 - Game Programming)

Dr.Çağatay ÜNDEĞER

Research and Innovation Director

SimBT Inc.

e-mail :

cagatay.undeger@simbt.com.tr

cagatay@undeger.com

Outline

- Finite-State Machines
- Decision Trees
- Instance-Based Learning
- Artificial Neural Networks
- Deep Learning

Finite-State Machines (FSMs)

- Defined by a set of states and transitions between them.
- Transition from a state to another state is triggered by a change (event or action) in the environment.

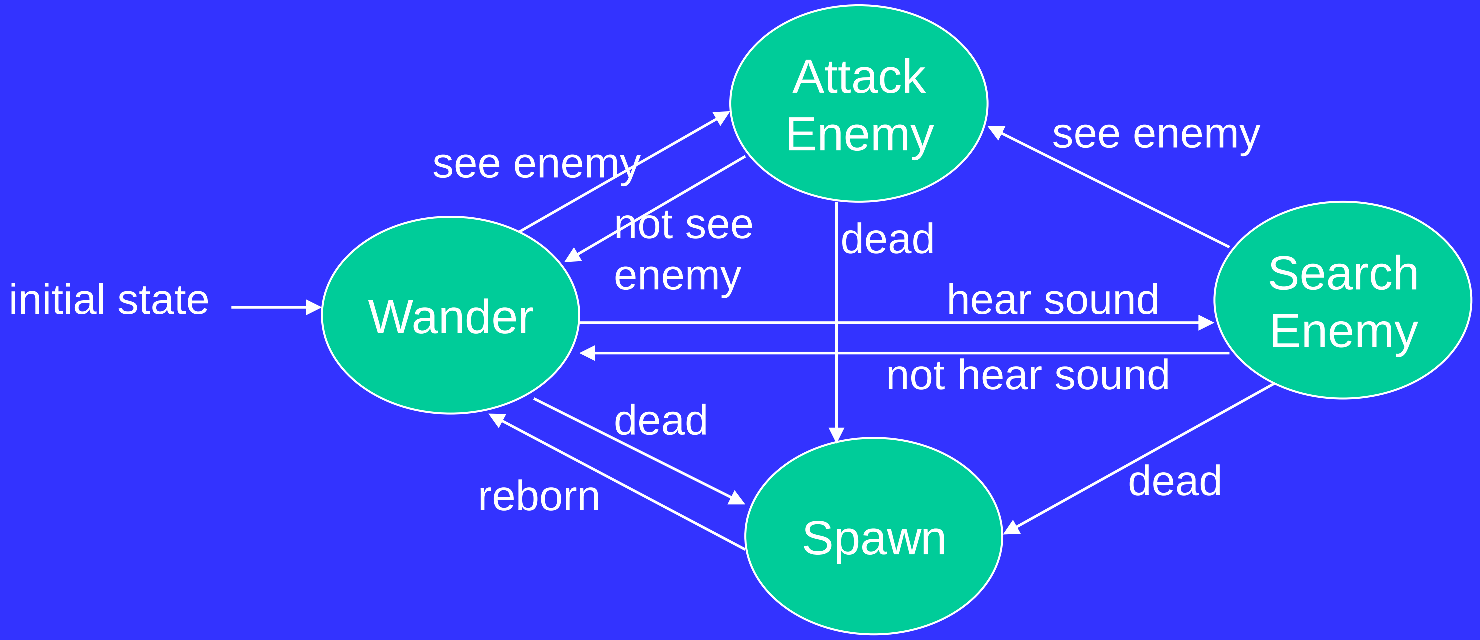
Finite-State Machines (FSMs)

- FSMs are used broadly in the video game industry.
  - Quake and Quake 2: a simple FSM system.
  - Warcraft III: a complex FSM system

- FSMs also have a large role outside of the video game industry.
  - For example, cars, airplanes, robots, etc. have complex FSMs for controlling the devices.

Finite-State Machines (FSMs)

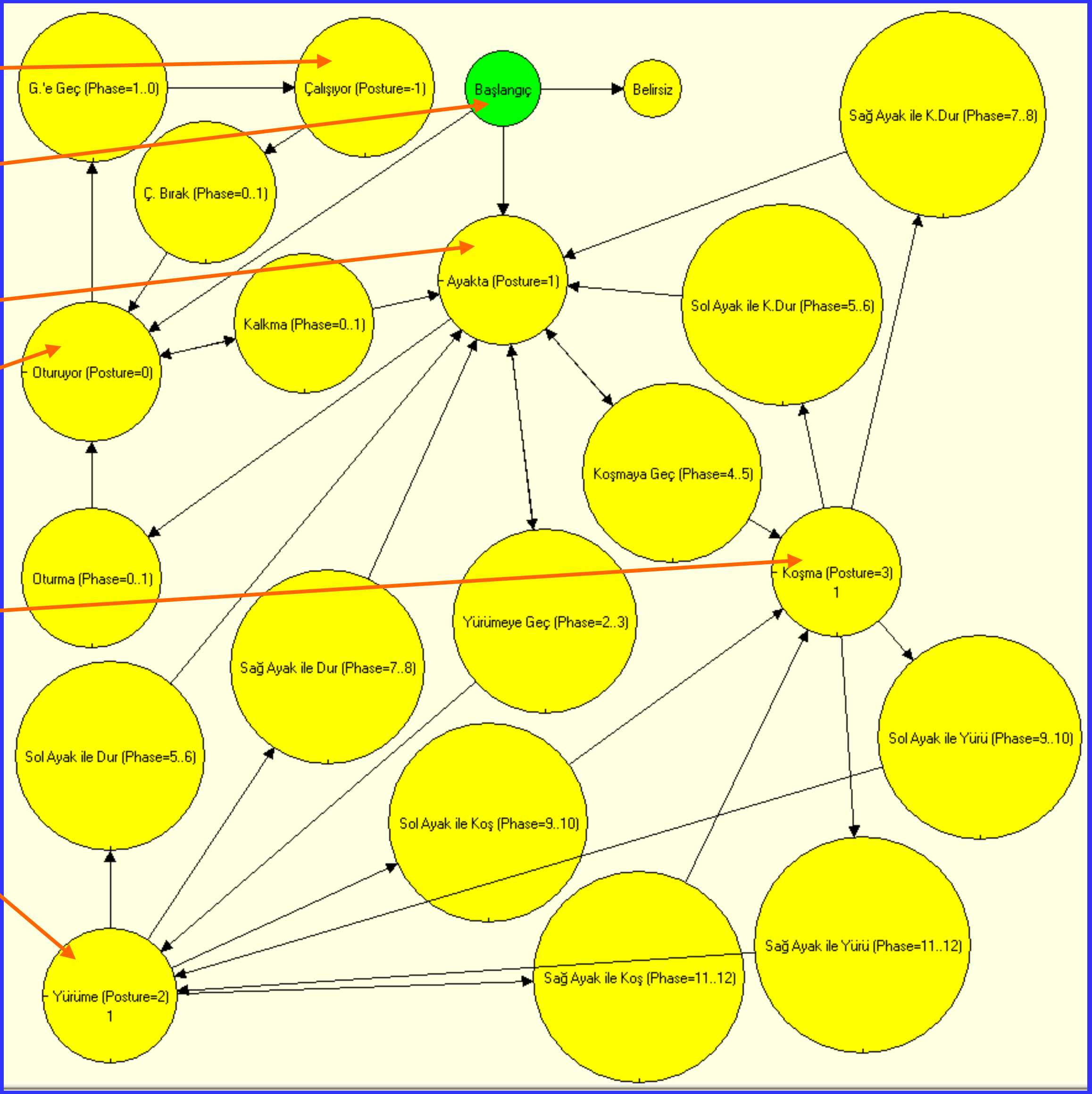
- Also called “State Transition Networks/Diagrams”



Finite-State Machines (FSMs)

- An FSM has an entry (initial) state for starting the execution.
- After execution starts, events and/or actions cause state transitions.
- Each state may execute a code while;
  - Entering a state
  - Maintaining a state in every step and/or
  - Leaving a state

A Sample


G.'e Geç (Phase=1..0)
Çalışıyor (Posture=-1)
Başlangıç
Belirsiz
Sağ Ayak ile K.Dur (Phase=7..8)
Ç. Bırak (Phase=0..1)
Ayakta (Posture=1)
Kalkma (Phase=0..1)
Oturuyor (Posture=0)
Sol Ayak ile K.Dur (Phase=5..6)
Koşmaya Geç (Phase=4..5)
Oturma (Phase=0..1)
Koşma (Posture=3)
1
Yürümeye Geç (Phase=2..3)
Sağ Ayak ile Dur (Phase=7..8)
Sol Ayak ile Dur (Phase=5..6)
Sol Ayak ile Yürü (Phase=9..10)
Sol Ayak ile Koş (Phase=9..10)
Sağ Ayak ile Yürü (Phase=11..12)
Sağ Ayak ile Koş (Phase=11..12)
Yürüme (Posture=2)
1

Working

Initial state

Standing

Sitting

Running

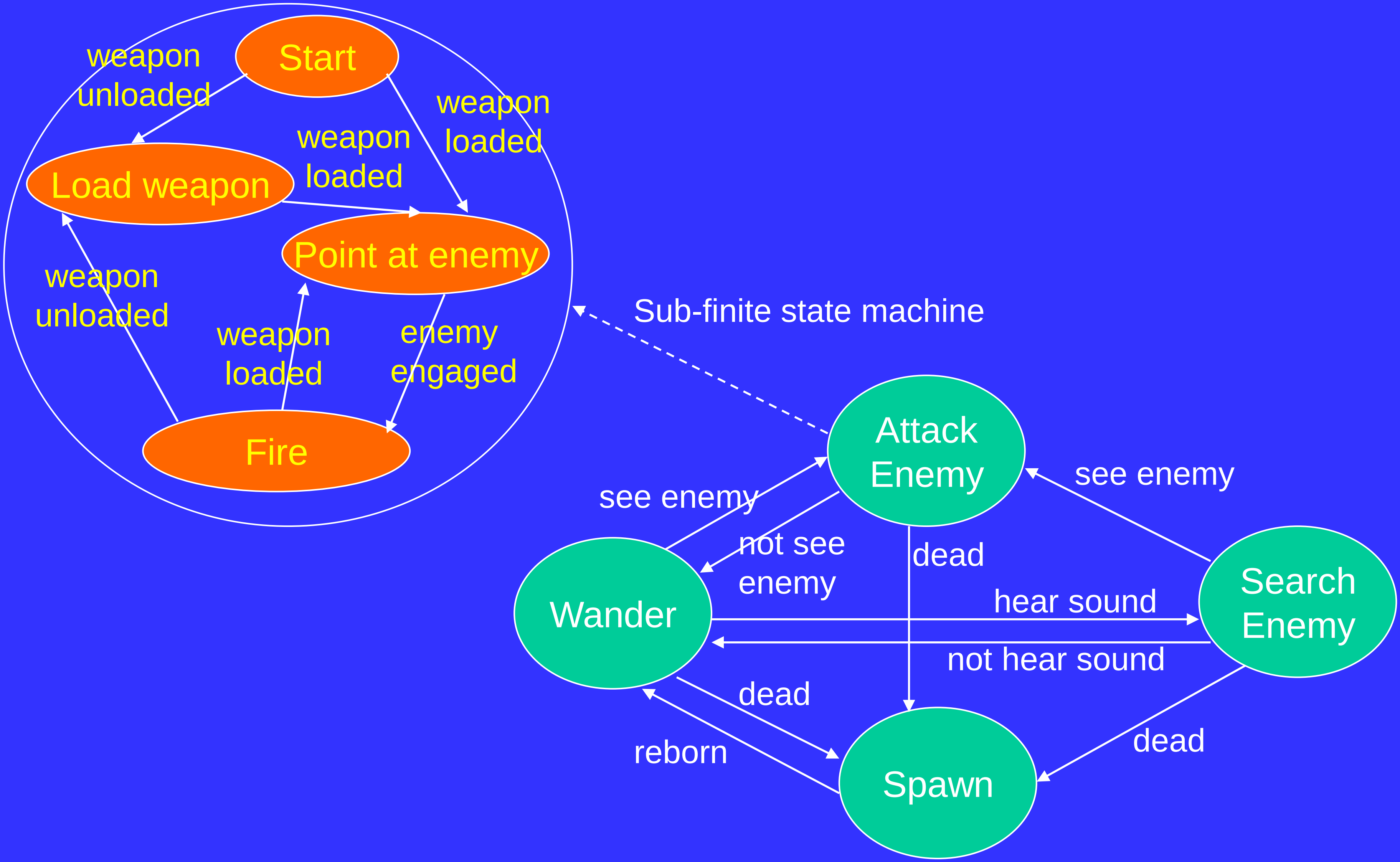
Walking

Hierarchical Finite-State Machines

- When the number of states increases,
  - It becomes very complicated to define the FSMs and
  - High probable to have some bugs !

- A solution is
  - To define some higher level states, and
  - Refine the details of states hierarchically.

A Sample


Start
weapon unloaded
weapon loaded
Load weapon
weapon loaded
Point at enemy
weapon unloaded
weapon loaded
enemy engaged
Fire
Sub-finite state machine
Attack Enemy
see enemy
see enemy
not see enemy
dead
Wander
hear sound
not hear sound
Search Enemy
dead
reborn
dead
Spawn

Advantages

- Very fast to execute.
- Expressive enough for simple behaviors.
- Can create tools for non-programmers to build behaviors.
- Probabilistic transitions can be introduced to make unpredictable transitions/behaviors.

Disadvantages

- Number of states and arcs can grow very fast.
- Easy to do errors in complex FSMs.
- Difficult to;
  - Put propositional representations such as:
    - Pick up the best weapon
    - Attack the closest enemy
  - Count such as:
    - Wait until the third time I see the enemy, then attack
  - Perform actions in parallel

Improvements

- But they are still prefered in many games, and
- They can be improved with some special additions such as:
  - Parallel finite-state machines
  - Higher level scripts to perform complex transitions

Decision Trees

- Rules are defined as a tree.
- Conditions are non-leaf nodes.
- Actions/decisions are leaf nodes.

- Decision trees can be learned (e.g. ID3) from instances (via questions and known answers)

Sample Inputs & Outputs

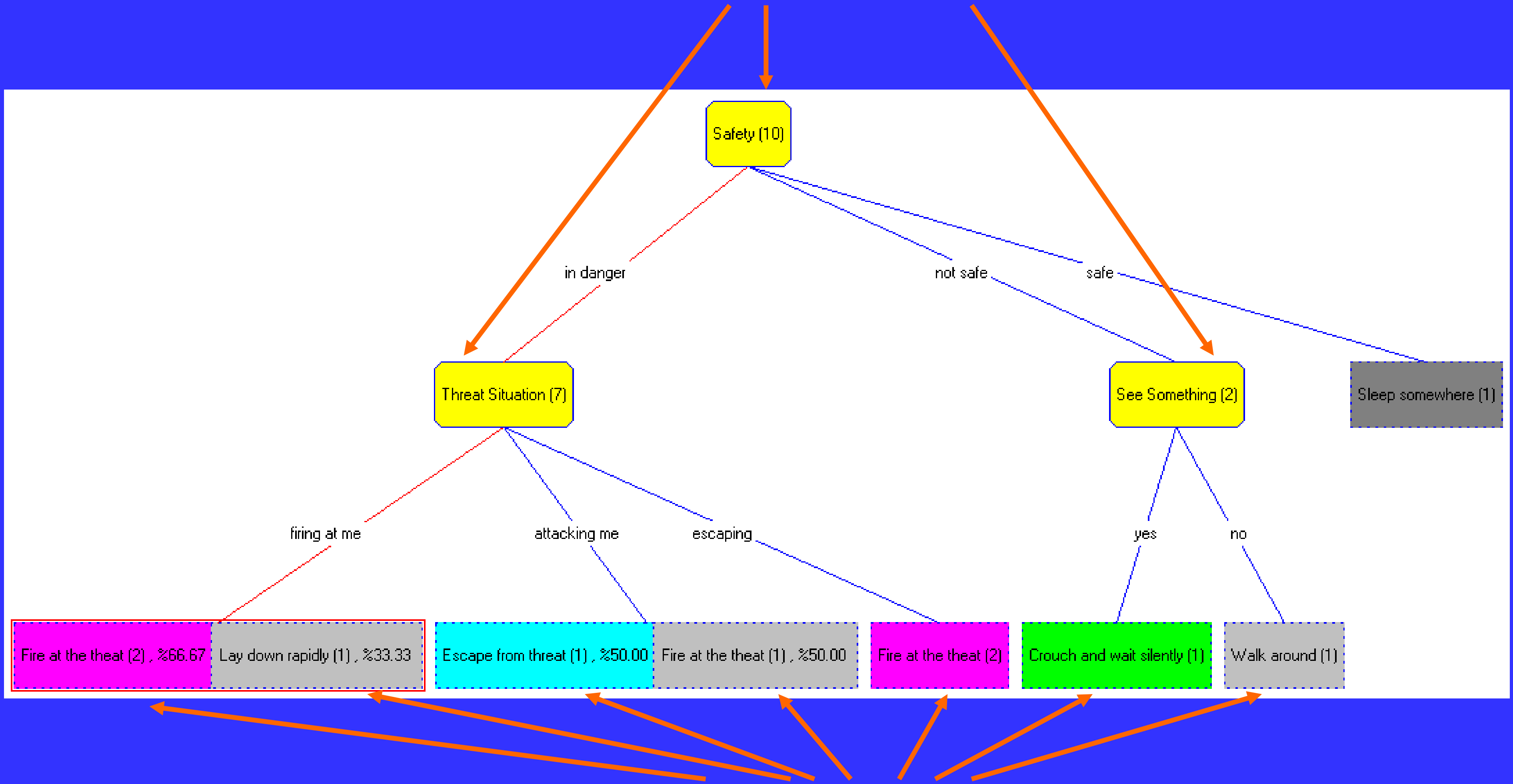
- Inputs (State Space variables)
  - Safety (in danger, not safe, safe)
  - See something (yes, no)
  - Theat situation (firing at me, attacting me, escaping)
- Outputs (Actions)
  - Fire at the threat
  - Lay down rapidly
  - Escape from threat
  - Crouch and wait silently
  - Walk around
  - Sleep somewhere

Sample Instances For Learning

- If in danger, see something and threat firing at me
  - then Fire at the threat (instance count = 2)
- If in danger, see something and threat firing at me
  - then Lay down rapidly (instance count = 1)
- If in danger, see something and threat attacking me
  - then Escape from threat (instance count = 1)
- If in danger, see something and threat attacking me
  - then Fire at the threat (instance count = 1)
- If in danger, see something and threat escaping
  - then Fire at the threat (instance count = 2)
- If not safe and see something
  - then Crouch and wait silently (instance count = 1)
- If not safe and not see something
  - then Walk around (instance count = 1)
- If safe
  - then Sleep somewhere (instance count = 1)

A Sample Decision Tree

Inputs (state variables)



Outputs (actions)

Advantages

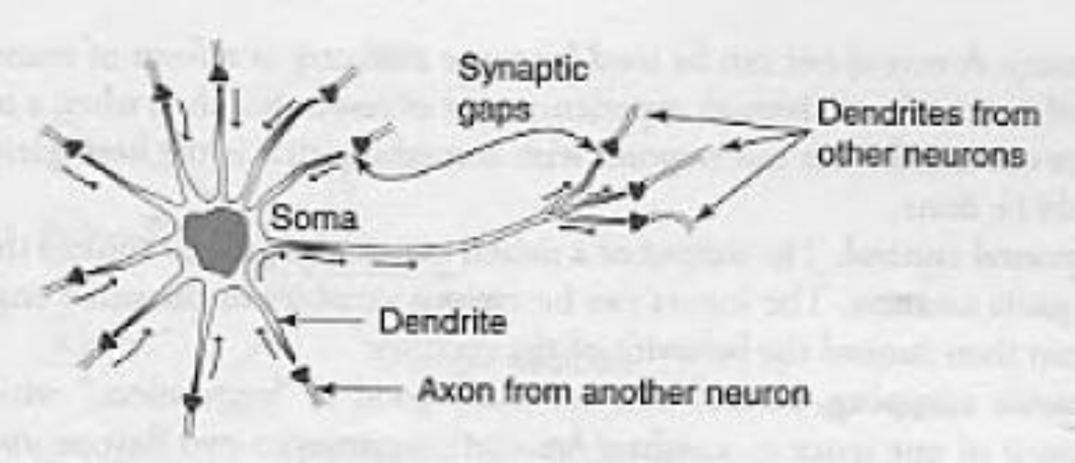
- A simple and compact representation.
- Easy to create and understand:
  - Can also be represented as rules
- Decision trees can be learned.

Disadvantages

- Decision tree learning algorithm is complex (hard to be coded).
- Need as many examples as possible.
- Sensitive to the errors in instances,
  - Learned decision trees may contain errors.

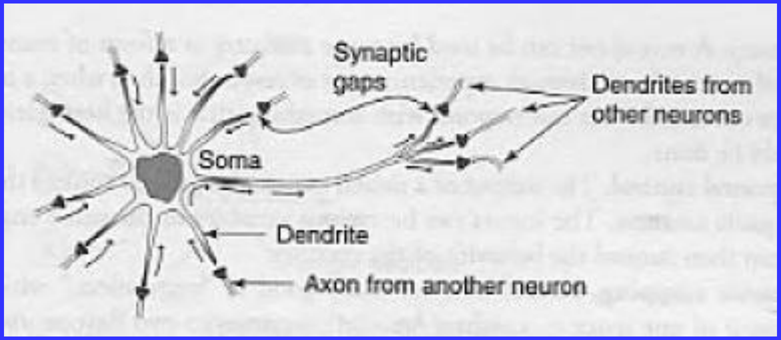
Artificial Neural Networks (NNets)

- Inspired by human brain.
- Fundamental functional units of brain are:
  - Neurons / soma / nerve cells and
  - Axons / dendrites connecting nerve cells.



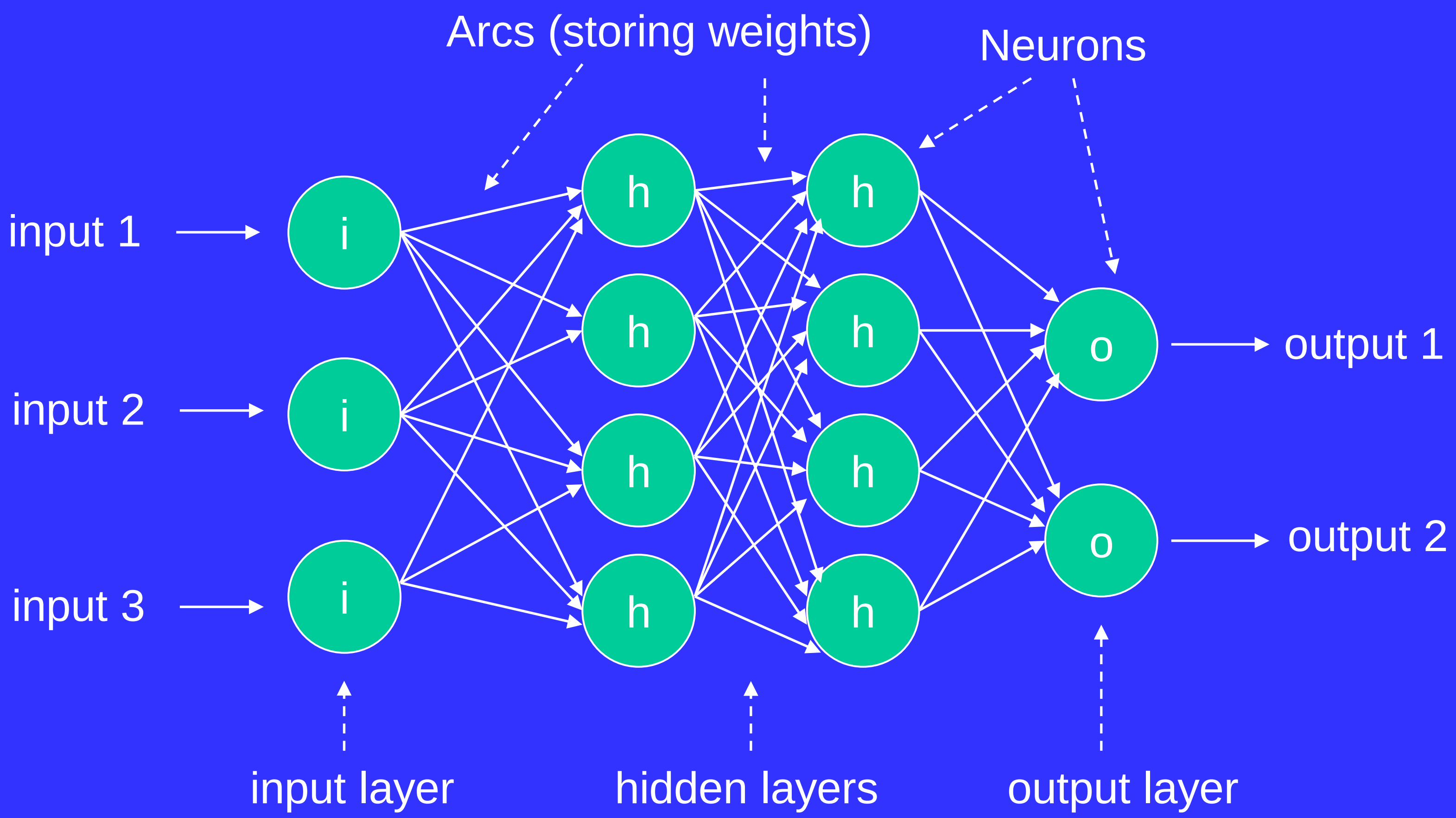
Artificial Neural Networks (NNets)

- Neural networks use a similar approach.
- Consist of:
  - Input arcs collect the signals,
  - Neuron sums the signals and
  - Output arcs transmit the summed signal based on a transmit function.



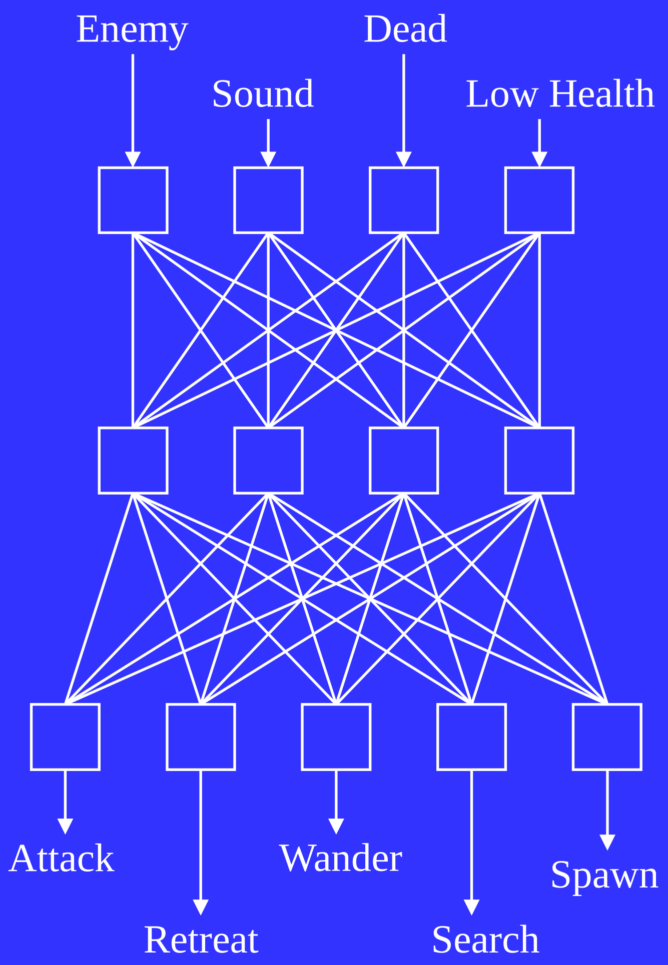
- Neural networks can be learned.

Neural Network Architecture



A Sample Neural Network

- The inputs are state variables that equal to either 0 (no) or 1 (yes).
- Outputs are action variables that are between 0 and 1.
- The action with the greatest output is selected.



Common Usage of NNets

- Environmental Scanning and Classification:
  - Interpret vision and auditory information
- Memory:
  - Can learn through experience
- Behavior Control:
  - Outputs can be used as actions to be performed.
- Response Mapping:
  - Any input output mapping and association.

Usage of NNets

- The usage of NNets is a little bit fuzzy.
- No exact cases and rules.
- Assume it as a tool and use your creativity to get whatever you like.

- It is the second best solution to a problem,
  - Use it if there is no known exact solution to the problem

Advantages

- Handle errors well.
- Graceful degradation.
- Can learn novel solutions.

Disadvantages

- Can’t understand how the learned network works, therefore they are the second best way to do something.
- Need as many examples as possible.
- Learning takes too much time / processing.
- Sometimes the network may not converge.

Deep Learning

- Deep Learning is an extended version of classical artificial neural networks.

- If you are just starting out in the field of deep learning or you had some experience with neural networks some time ago,
  - You may be confused.

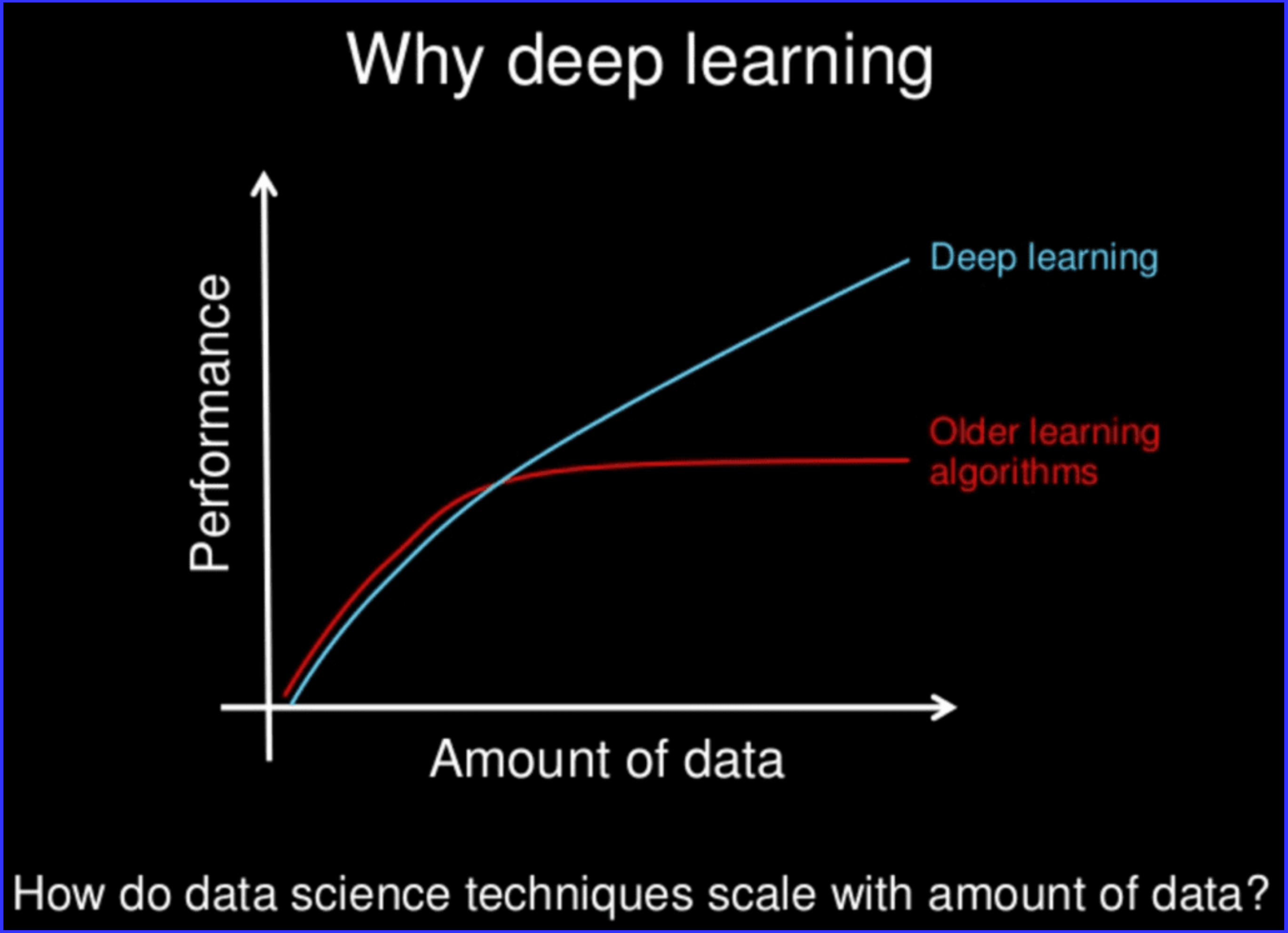
- Indeed Deep Learning is Large Multi-Layer Neural Network with new techniques introduced to handle large and deep networks better.

Deep Learning

- What was wrong with classical neural networks in back 1980s.
  - Our labeled datasets were thousands of times too smaller
  - Our computers were millions of times too slow
  - We initialized the weights in a wrong way
  - We used the wrong type of non-linearity

Deep Learning

- Improved learning performance



How do data science techniques scale with amount of data?

Advantages

- Highly Scalable.
- Ability to perform automatic feature extraction from raw data.
- Discover and learn good representations using feature learning.

Disdvantages

- Needs too much time and processing power to do training.
- Needs strong hardware to execute learned deep networks.